Oracle® Fusion Middleware
Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper
11g Release 1 (11.1.1.9.0)
E27739-08

October 2015
Documentation for developers that explains how to build and deploy Portal Framework applications, portlets, portal assets, and custom portal components using Oracle WebCenter Portal and Oracle JDeveloper.
Contents

Preface ................................................................................................................................. lxvii
  Audience ......................................................................................................................... lxvii
  Documentation Accessibility ......................................................................................... lxviii
  Related Documents ........................................................................................................ lxviii
  Conventions ..................................................................................................................... lxviii

What's New? ...................................................................................................................... lxix
  System Requirements and Specifications ...................................................................... lxix
  New and Changed Features for 11g Release 1 (11.1.1.9.0) ........................................ lxix
  New and Changed Features for 11g Release 1 (11.1.1.8.3) .......................................... lxx
  New and Changed Features for 11g Release 1 (11.1.1.8.0) .......................................... lxxi

Who's Who ....................................................................................................................... lxxv
  Knowledge Worker ....................................................................................................... lxxv
  Application Specialist ................................................................................................. lxxvi
  Web Developer ............................................................................................................. lxxvii
  Developer ...................................................................................................................... lxxvii
  System Administrator .................................................................................................. lxxviii

Part I Getting Started

1 Introduction to Oracle WebCenter Portal for Developers
  1.1 What Is the Purpose of this Guide? ............................................................................ 1-1
  1.2 What Is My Role as a WebCenter Portal Developer? .............................................. 1-2

2 Setting Up Your Development Environment
  2.1 Introduction ............................................................................................................. 2-1
  2.2 General Setup Tasks ............................................................................................... 2-1
  2.2.1 Installing Oracle JDeveloper .............................................................................. 2-2
  2.2.2 Installing the WebCenter Portal Extension for JDeveloper ................................ 2-2
  2.2.3 Setting the User Home Directory Environment Variable .................................. 2-3
  2.2.4 Managing the Integrated WebLogic Server ...................................................... 2-3
    2.2.4.1 What Is the Integrated WebLogic Server? .................................................. 2-3
    2.2.4.2 Starting and Stopping Integrated WebLogic Server .................................... 2-3
4.2.3 Setting Up an External Application Connection .......................................................... 4-12
4.3 Customizing How Services Render Resources ............................................................. 4-13
4.3.1 Overview of Resource Rendering Options ............................................................... 4-13
4.3.2 Rendering Resources in the Same Browser Window .................................................. 4-14
4.3.3 Rendering Resources in an ADF Inline Popup .......................................................... 4-14
4.3.4 Rendering Resources in a New Browser Window ...................................................... 4-15
4.3.5 Setting Rendering Behavior with the resourceActionBehavior Tag ......................... 4-15
4.3.6 Updating adf-config.xml in Deployed Framework Applications ............................... 4-16
4.4 Configuring General Settings ....................................................................................... 4-16
4.4.1 Building a Preferences User Interface .................................................................... 4-18
4.4.2 Using General Settings to Specify the User’s Locale ............................................... 4-18

Part II Building WebCenter Portal Framework Applications

5 Understanding WebCenter Portal Framework Applications
5.1 What Is a WebCenter Portal Framework Application? ................................................... 5-2
5.2 Creating a WebCenter Portal Framework Application .................................................. 5-2
5.3 Exploring the Features of a WebCenter Portal Framework Application ......................... 5-2
5.3.1 Summary of Features ............................................................................................... 5-2
5.3.2 Understanding Pages, Page Templates, and the Portal Page Hierarchy ...................... 5-3
5.3.3 Securing WebCenter Portal Framework Pages ....................................................... 5-8
5.3.4 Understanding the Navigation Model and the Navigation Registry ............................ 5-9
5.3.5 Understanding Resource Catalogs and the Catalog Registry .................................... 5-11
5.4 How are WebCenter Portal Framework Application Files Organized? ........................ 5-13
5.4.1 Understanding the Organization of a WebCenter Portal Framework Application .... 5-13
5.4.1.1 How Is a WebCenter Portal Framework Application Organized? ....................... 5-13
5.4.1.2 Why Is a WebCenter Portal Framework Application Organized the Way It Is? ....... 5-14
5.4.2 Viewing Your Portal Project on the Filesystem ....................................................... 5-15
5.4.3 Viewing Your Portal Project in JDeveloper ............................................................ 5-15
5.4.4 Showing Hidden Files in the Application Navigator ............................................... 5-17
5.5 What Is the Portal Life Cycle? ..................................................................................... 5-15
5.6 Developing Your Portal’s Look and Feel .................................................................... 5-18
5.7 Changing Default Portal Preferences ......................................................................... 5-19
5.7.1 What Are the Default Portal Preferences? .............................................................. 5-19
5.7.2 How to Change the Default Preferences in JDeveloper ........................................ 5-19
5.8 Using Iterative and Round-Trip Development Techniques .......................................... 5-20
5.9 Running and Testing a Portal ...................................................................................... 5-20
5.10 WebCenter Portal Framework Applications at Runtime ............................................ 5-21
5.10.1 Preserving Runtime Customizations on the Integrated WebLogic Server ............... 5-22
5.10.2 Browser-Based WebCenter Portal Administration Console .................................. 5-23
5.10.3 How Security Settings Affect the Runtime Portal .................................................. 5-24
5.10.4 Editing of Portal Resources Using Browser-Based Tools ...................................... 5-24
5.11 Changing the Default Home Page and Login/Logout Target Pages ............................ 5-25
5.11.1 Understanding How the Home Page Is Specified ................................................. 5-25
5.11.2 How to Change the Default Home Page ............................................................... 5-26
5.11.3 How Is the Default Index Page Specified? ............................................................ 5-27
6 Creating WebCenter Portal Framework Applications

6.1 Creating a New WebCenter Portal Framework Application ........................................ 6-1
6.2 Understanding the WebCenter Portal Framework Application Template .................. 6-6
6.3 Building a Portal ........................................................................................................ 6-7
6.4 Understanding the PortalWebAssets Project ............................................................ 6-7
6.4.1 Introduction .............................................................................................................. 6-7
6.4.2 Decoupling the Static Application Resources Project ........................................... 6-8
6.4.3 Deployment Options ................................................................................................. 6-9
6.4.4 Using EL to Manage Static Resources ................................................................. 6-10
6.4.4.1 Defining a Base URL Preference ........................................................................ 6-10
6.4.4.2 Mapping URLs Dynamically ............................................................................. 6-10
6.5 How Is Framework Application Security Configured by Default? ............................ 6-11
6.6 What Configuration Files Should I Know About? ....................................................... 6-12
6.6.1 The Framework Application Template Default web.xml File ............................... 6-12
6.6.2 The Framework Application Template Default faces-config.xml File .................... 6-13
6.6.3 The Framework Application Template Default trinidad-config.xml File ................. 6-14
6.6.4 The Framework Application Template Default adfc-config.xml File ..................... 6-15
6.7 Adding Excluded Files to the Application Navigator ............................................... 6-15
6.8 Developing Framework Portal Applications for High Availability ............................ 6-16
6.8.1 Introduction to High Availability for WebCenter Portal and JDeveloper ............... 6-17
6.8.2 Serializing Managed Beans ...................................................................................... 6-17
6.8.3 Minimizing the Session Footprint .......................................................................... 6-17
6.8.4 Application Configuration ....................................................................................... 6-18
6.8.5 Mutating Managed Beans ...................................................................................... 6-18
6.8.6 Replication Groups ............................................................................................... 6-19
6.9 Securing the Portal ..................................................................................................... 6-19

7 Deploying and Testing Your Portal Framework Application

7.1 Introduction to Oracle WebLogic Servers .................................................................. 7-1
7.2 Deploying a Portal Framework Application to the Integrated WebLogic Server .......... 7-2
7.2.1 Running a Portal Framework Application in the Integrated WebLogic Server ........ 7-3
7.2.2 Deploying a Portal Framework Application to the Integrated WebLogic Server ...... 7-4
7.3 Deploying a Portal Framework Application to a WebLogic Managed Server .......... 7-6
7.3.1 Deployment Roadmap ............................................................................................. 7-6
7.3.2 Packaging a Framework Application ....................................................................... 7-9
7.3.2.1 Packaging Database Connections and Application Security ............................. 7-9
7.3.2.1.1 Packaging the Database Connections ......................................................... 7-9
7.3.2.1.2 Packaging Application Security Data .......................................................... 7-11
7.3.2.2 Creating Deployment Profiles .......................................................................... 7-12
7.3.3 Preparing the Target Environment for Deployment ............................................... 7-14
7.3.3.1 Creating and Provisioning an Oracle WebLogic Managed Server Instance ...... 7-14
8 Understanding the WebCenter Portal Framework Application Life Cycle

8.2 What Are the Major Life Cycle Tasks? ..................................................................................... 8-3
8.2.1 One-Time Setup Tasks ........................................................................................................ 8-4
8.2.2 Development Environment Tasks ....................................................................................... 8-4
8.2.3 Nightly Build Environment Tasks ....................................................................................... 8-4
8.2.4 Testing Environment Tasks ................................................................................................ 8-5
8.2.5 Stage Environment Tasks .................................................................................................. 8-5
8.2.6 Production Environment Tasks .......................................................................................... 8-5
8.3 Who Participates in the Portal Life Cycle? ................................................................................ 8-6
8.4 Understanding the Build and Test Environments ..................................................................... 8-6
8.5 Understanding the Staging and Production Environments ....................................................... 8-8
8.5.1 Provisioning the Staging Environment ................................................................................ 8-9
8.5.2 Adding Content to the Staging Environment ...................................................................... 8-9
8.5.3 Moving the Portal from Staging to Production .................................................................... 8-9
8.6 Tools for Managing the Life Cycle ............................................................................................ 8-9
8.7 Understanding Iterative Development...................................................................................... 8-10
8.7.1 What Is Iterative Development? ......................................................................................... 8-10
8.7.2 How Does Iterative Development Work? ........................................................................... 8-11
8.7.3 Enabling Iterative Development ......................................................................................... 8-12
8.8 Configuring a Nightly Build Script .......................................................................................... 8-12
8.9 Setting Up a Staging or Production Environment for the First Time ........................................ 8-13
8.10 Moving a WebCenter Portal Framework Application to an Existing Environment .......... 8-13
8.11 Deploying to Managed Servers .............................................................................................. 8-13
8.12 Deploying and Configuring the Application on Targeted Servers ......................................... 8-14
8.12.1 Introduction ......................................................................................................................... 8-14
8.12.2 Using the Deploy and Configure Script .............................................................................. 8-14
8.12.3 Managing Post-Deployment Changes ................................................................................. 8-17
8.13 Using the Propagation Tool to Propagate From Staging to Production ................................ 8-17
8.13.1 Introduction to the Propagation Tool ................................................................................ 8-17
8.13.2 Configuring the Propagation Tool ...................................................................................... 8-19
8.13.2.1 Configuring the URL Connection With Oracle Enterprise Manager Fusion Middleware Control Console 8-19
8.13.2.2 Configuring the URL Connection with WLST ................................................................ 8-20
8.13.3 Propagating Portal Metadata ............................................................................................ 8-21
8.14 Propagating Content From Oracle WebCenter Content ....................................................... 8-21
8.15 Managing Security Through the Life Cycle ............................................................................ 8-21
8.15.1 Migrating Security Policies for First-Time Deployment ..................................................... 8-22
8.15.2 Migrating Security Policies On Redeployment ................................................................. 8-22
9 Introduction to Portal Resource Management

9.1 Introduction to Portal Resources ................................................................. 9-1
9.2 Creating Portal Resources ............................................................................ 9-2
9.3 Editing Portal Resources ............................................................................. 9-3
9.4 Deleting Portal Resources ........................................................................... 9-2
9.5 Working with Portal Resources at Runtime .................................................. 9-3
9.5.1 Enabling Runtime Administration of Portal Resources ......................... 9-4
9.5.1.1 How to Add the Assets Page to a Page .............................................. 9-5
9.5.1.2 How to Include a Portal Resource on the Assets Page ................... 9-6
9.5.1.3 How to Update Portal Resource Properties ...................................... 9-8
9.5.1.4 How to Remove a Portal Resource from the Assets Page ................ 9-8
9.5.1.5 How to Use the Assets Page at Runtime ......................................... 9-8
9.6 Working with Round-Trip Development ...................................................... 9-9
9.6.1 How to Download a Portal Resource Using the Assets Page ................ 9-11
9.6.2 How to Import a Portal Resource into JDeveloper ............................... 9-11
9.6.3 How to Export a Portal Resource from JDeveloper ................................. 9-12
9.6.4 How to Upload a Resource Using the Assets Page .................................. 9-13
9.7 Working with WebCenter Portal Resources ................................................ 9-13

10 Developing a Navigation Model

10.1 Introduction to Navigation Models ............................................................ 10-1
10.2 Creating a Navigation Model ..................................................................... 10-2
10.3 Editing a Navigation Model ....................................................................... 10-4
10.3.1 How to Drag and Drop a Resource onto a Navigation Model ............... 10-5
10.3.2 How to Add a Page to a Navigation Model .......................................... 10-6
10.3.3 How to Add a Folder to a Navigation Model ....................................... 10-7
10.3.4 How to Add a Separator to a Navigation Model ................................. 10-8
10.3.5 How to Add Other Resources to a Navigation Model ......................... 10-9
10.3.6 How to Add Dynamically Generated Resources to a Navigation Model 10-13
10.3.7 How to Set Display Options for a Navigation Model or Navigation Model Node 10-15
10.3.8 How to Rearrange Nodes in a Navigation Model ................................ 10-16
10.3.9 How to Show or Hide a Navigation Model or Navigation Model Node 10-17
10.3.10 How to Edit a Navigation Model Node .............................................. 10-18
10.3.11 How to Delete a Navigation Model Node ........................................... 10-18
10.4 Selecting the Default Navigation Model .................................................... 10-18
10.5 Filtering the Contents of a Navigation Model ............................................. 10-19
10.6 Deleting a Navigation Model .................................................................... 10-20
10.7 Editing the Navigation Registry File ........................................................... 10-20
10.8 Editing the Navigation Renderer ................................................................. 10-21
10.9 Navigation URL Reserved Keywords ......................................................... 10-22
10.10 Tips for Developing Navigation Models .................................................. 10-23
11 Developing Page Templates

11.1 Introduction to Developing Page Templates .......................................................... 11-1
  11.1.1 Understanding Page Template Structure ......................................................... 11-2
  11.1.2 Understanding Page Template Layout .............................................................. 11-3
  11.1.3 Understanding Page Template Layout Components ......................................... 11-4
  11.1.4 Understanding the Built-In Page Templates in a WebCenter Portal Framework Application .......................................................... 11-7
  11.2 Best Practices for Developing Page Templates .................................................... 11-9
    11.2.1 Best Practices for Creating Stretching Layouts .............................................. 11-13
    11.2.2 Best Practices for Creating Flowing Layouts ............................................... 11-14
    11.2.3 Best Practices for Developing Page Templates That Can Be Edited at Runtime ........................................................................... 11-15
    11.2.4 Best Practices for Customizing the Appearance of Components .................. 11-15
    11.2.5 Best Practices for Defining Scrolling in a Page Template ................................ 11-16
    11.2.6 Best Practices for Defining Margins, Borders, and Padding ............................ 11-17
    11.2.7 Best Practices for Hiding Content in Page Templates in a Portal Framework Application .......................................................... 11-17
  11.3 Creating a Page Template ....................................................................................... 11-17
  11.4 Editing a Page Template ....................................................................................... 11-21
  11.5 Applying a Page Template to a Page ..................................................................... 11-24
    11.5.1 How to Apply a Page Template During Page Creation in a WebCenter Portal Framework Application .......................................................... 11-24
    11.5.2 How to Change the Page Template Applied to a Page in a WebCenter Portal Framework Application .......................................................... 11-25
    11.5.3 How to Enable Dynamic Switching of Page Templates at Runtime in a WebCenter Portal Framework Application .......................................................... 11-26
    11.5.3.1 What Happens at Runtime ........................................................................ 11-26
  11.6 Selecting the Default Page Template ..................................................................... 11-27
  11.7 Deleting a Page Template ..................................................................................... 11-28
  11.8 Page Templates Tutorials and Examples ................................................................ 11-28
14.3.16 How to Control Visibility of Portlets in a Resource Catalog......................... 14-20
14.4 Selecting the Default Resource Catalog ............................................................. 14-21
14.4.1 How to Select the Default Resource Catalog for an Application.................. 14-21
14.4.2 How to Configure Multiple Resource Catalogs ............................................ 14-22
14.4.3 How to Select the Default Resource Catalog for a Page ................................. 14-24
14.5 Filtering Items in a Resource Catalog .................................................................. 14-24
14.6 Deleting a Resource Catalog ............................................................................... 14-25
14.7 Defining Task Flow Parameters and Attributes of the Enclosing Show Detail Frames for Task Flows 14-26
14.8 Troubleshooting Problems with Resource Catalogs ......................................... 14-27

15 Creating Pages and Adding Resources
15.1 Introduction to Pages ............................................................................................. 15-1
15.2 Creating Pages in a WebCenter Portal Framework Application ......................... 15-2
15.3 Adding WebCenter Portal Tools and Services Task Flows to a Page .................. 15-3
15.4 Adding Custom Task Flows to a Page .................................................................... 15-4
15.5 Adding Content from WebCenter Content to a Page ........................................... 15-5
15.6 Adding Portlets to a Page ....................................................................................... 15-6
15.7 Adding Pagelets to a Page ..................................................................................... 15-6
15.8 Adding Data Controls to a Page ............................................................................ 15-6
15.9 Using External Applications .................................................................................. 15-7

Part III Customizing Your Application and Extending Customization Options

16 Using WebCenter Portal Composer
16.1 Introduction to Composer ...................................................................................... 16-1
16.2 Understanding Application and User Customizations ........................................... 16-2
16.3 Understanding the Edit Mode and View Mode of a Page ..................................... 16-3
16.3.1 About Edit Mode .................................................................................................. 16-3
16.3.1.1 About Add Content View in Edit Mode .............................................................. 16-4
16.3.1.2 About Structure View in Edit Mode ................................................................. 16-5
16.3.1.3 About Design View in Edit Mode ..................................................................... 16-6
16.3.1.4 About Select View in Edit Mode ..................................................................... 16-6
16.3.1.5 About Preview View in Edit Mode ................................................................. 16-7
16.4 Customizing Capabilities in Page View Mode ....................................................... 16-7
16.4.1 Rearrange Components ...................................................................................... 16-7
16.4.2 Change the Layout .............................................................................................. 16-8
16.4.3 Expand and Collapse Components .................................................................... 16-9
16.5 Editing Capabilities in Design View and Add Content View in Page Edit Mode ...... 16-10
16.5.1 Add Content ....................................................................................................... 16-11
16.5.1.1 Add a Tab Set or a Tab to a Box Component .................................................. 16-14
16.5.1.2 Add Box Components Adjacent to Existing Ones ........................................ 16-15
16.5.1.2.1 Add Box Above .......................................................................................... 16-16
16.5.1.2.2 Add Box Below .......................................................................................... 16-17
16.5.1.2.3 Add Box Left ............................................................................................. 16-17
16.5.1.2.4 Add Box Right .......................................................................................... 16-17
17 Enabling Runtime Creation and Management of Pages

17.1 Introduction to Page Creation and Management ................................................. 17-1
17.2 Creating Pages and Task Flows ........................................................................... 17-2
17.2.1 How to Create Pages ..................................................................................... 17-3
17.2.1.1 How to Add the Page - Create New Task Flow ...................................... 17-3
17.2.1.2 Setting Security for Pages in JDeveloper ............................................... 17-4
17.2.1.3 How to Create Pages at Runtime ............................................................. 17-5
17.2.1.4 Structure of Pages Created at Runtime .................................................... 17-5
17.2.1.5 How to Access Pages Created at Runtime ............................................... 17-6
17.3 Defining Values for the Page - Create New Task Flow Parameters .................... 17-7
17.3.1 How to Access Page - Create New Task Flow Parameters ............................ 17-7
17.3.2 Setting Scope in a Page - Create New Task Flow ......................................... 17-8
17.3.3 Setting an Outcome Parameter ....................................................................... 17-9
17.3.4 Specifying Styles ......................................................................................... 17-11
17.3.4.1 Out-of-the-Box Styles ............................................................................ 17-11
19 Extending Runtime Editing Capabilities Using Composer

19.1 Overview of Extensibility Options .............................................................. 19-1
19.1.1 Composer Add-Ons ............................................................................... 19-2
19.1.2 Composer Custom Property Panels ......................................................... 19-2
19.1.3 Composer Events Handlers ...................................................................... 19-2
19.1.4 Drop Handlers ........................................................................................ 19-3
19.1.5 Component Property Filters ..................................................................... 19-3
19.1.6 Customization Manager ........................................................................... 19-3
19.1.7 Composer Toolbar Customization ............................................................... 19-3
19.1.8 Configuration Files .................................................................................. 19-4
19.2 Creating Composer Add-Ons ................................................................. 19-4
19.2.1 How to Create and Register Add-Ons ......................................................... 19-5
19.2.1.1 Creating an Add-On Task Flow ............................................................... 19-5
19.2.1.2 Registering Add-Ons with Composer ...................................................... 19-7
19.2.1.3 Registering Add-Ons in adf-config.xml ................................................ 19-8
19.2.2 What Happens at Runtime ...................................................................... 19-8
19.2.3 How to Exclude Composer Default Add-Ons ........................................... 19-9
19.2.4 How to Display the Customization Manager Add-On ............................... 19-10
19.2.5 How to Selectively Display Add-Ons ........................................................ 19-10
19.3 Creating Custom Property Panels .......................................................... 19-11
19.3.1 How to Create and Register Custom Property Panels ............................. 19-11
19.3.1.1 Creating a Custom Property Panel ......................................................... 19-12
19.3.1.2 Registering a Custom Property Panel for a Component ....................... 19-12
19.3.1.3 Registering a Custom Property Panel for a Task Flow ......................... 19-13
19.3.2 What Happens at Runtime ...................................................................... 19-15
19.3.3 How to Override Default Property Panels .............................................. 19-15
19.3.4 How to Exclude Default Property Panels .............................................. 19-16
19.3.5 How to Selectively Render Property Panels ............................................ 19-16
19.3.6 How to Display Properties and Parameters in a Custom Property Panel ... 19-16
19.4 Extending the Expression Builder ............................................................ 19-20
19.4.1 How to Extend the Expression Builder ...................................................... 19-21
19.4.2 How to Protect Expression Language ..................................................... 19-22
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>19.4.3</td>
<td>What Happens at Runtime</td>
<td>19-22</td>
</tr>
<tr>
<td>19.5</td>
<td>Configuring Custom LOVs or Pickers for Task Flow Parameters</td>
<td>19-23</td>
</tr>
<tr>
<td>19.5.1</td>
<td>How to Configure an LOV</td>
<td>19-26</td>
</tr>
<tr>
<td>19.5.1.1</td>
<td>Configuring a Static LOV</td>
<td>19-26</td>
</tr>
<tr>
<td>19.5.1.2</td>
<td>Configuring a Dynamic LOV</td>
<td>19-27</td>
</tr>
<tr>
<td>19.5.1.3</td>
<td>Configuring a Picker</td>
<td>19-28</td>
</tr>
<tr>
<td>19.5.1.3.1</td>
<td>Creating the Picker Task Flow</td>
<td>19-28</td>
</tr>
<tr>
<td>19.5.1.3.2</td>
<td>Registering the Picker for a Task Flow Parameter</td>
<td>19-32</td>
</tr>
<tr>
<td>19.5.1.4</td>
<td>Configuring an LOV from a Global List</td>
<td>19-33</td>
</tr>
<tr>
<td>19.5.2</td>
<td>What Happens at Runtime</td>
<td>19-34</td>
</tr>
<tr>
<td>19.6</td>
<td>Configuring Event Handlers for Composer UI Events</td>
<td>19-34</td>
</tr>
<tr>
<td>19.6.1</td>
<td>How to Create and Register Handlers for Composer UI Events</td>
<td>19-34</td>
</tr>
<tr>
<td>19.6.1.1</td>
<td>UI Events that Support Event Handler Registration</td>
<td>19-35</td>
</tr>
<tr>
<td>19.6.1.2</td>
<td>Creating a Save Event Handler: Example</td>
<td>19-36</td>
</tr>
<tr>
<td>19.6.1.3</td>
<td>Registering an Event Handler with Composer</td>
<td>19-38</td>
</tr>
<tr>
<td>19.6.2</td>
<td>What Happens When You Create and Register Event Handlers</td>
<td>19-38</td>
</tr>
<tr>
<td>19.6.3</td>
<td>Additional Composer Event Handler Configurations</td>
<td>19-38</td>
</tr>
<tr>
<td>19.6.3.1</td>
<td>Specifying a Sequence Number for an Event Handler</td>
<td>19-38</td>
</tr>
<tr>
<td>19.6.3.2</td>
<td>Terminating Event Processing</td>
<td>19-39</td>
</tr>
<tr>
<td>19.7</td>
<td>Configuring Drop Handlers in the Resource Catalog</td>
<td>19-40</td>
</tr>
<tr>
<td>19.7.1</td>
<td>How to Create and Register Java Drop Handlers</td>
<td>19-40</td>
</tr>
<tr>
<td>19.7.1.1</td>
<td>Creating a Drop Handler</td>
<td>19-40</td>
</tr>
<tr>
<td>19.7.1.2</td>
<td>Registering a Drop Handler with Composer</td>
<td>19-43</td>
</tr>
<tr>
<td>19.7.1.3</td>
<td>Adding an XML Component to the Resource Catalog</td>
<td>19-44</td>
</tr>
<tr>
<td>19.7.2</td>
<td>What Happens at Runtime</td>
<td>19-44</td>
</tr>
<tr>
<td>19.8</td>
<td>Defining Property Filters</td>
<td>19-45</td>
</tr>
<tr>
<td>19.8.1</td>
<td>How to Define Property Filters</td>
<td>19-45</td>
</tr>
<tr>
<td>19.8.2</td>
<td>What Happens at Runtime</td>
<td>19-46</td>
</tr>
<tr>
<td>19.8.3</td>
<td>How to Remove Property Filters</td>
<td>19-46</td>
</tr>
<tr>
<td>19.9</td>
<td>Enabling Parameter Support on the Customization Manager Task Flow</td>
<td>19-47</td>
</tr>
<tr>
<td>19.9.1</td>
<td>How to Set the Customization Manager Task Flow Parameters</td>
<td>19-49</td>
</tr>
<tr>
<td>19.9.2</td>
<td>What Happens When You Enable Parameter Support</td>
<td>19-50</td>
</tr>
<tr>
<td>19.10</td>
<td>Customizing the Composer Toolbar</td>
<td>19-50</td>
</tr>
<tr>
<td>19.10.1</td>
<td>How to Rearrange or Hide Toolbar Elements</td>
<td>19-51</td>
</tr>
<tr>
<td>19.10.2</td>
<td>What Happens at Runtime</td>
<td>19-51</td>
</tr>
<tr>
<td>19.10.3</td>
<td>How to Add New Sections to the Toolbar</td>
<td>19-52</td>
</tr>
<tr>
<td>19.10.4</td>
<td>How to Override a Toolbar Section to Display Custom Content</td>
<td>19-53</td>
</tr>
<tr>
<td>19.11</td>
<td>Enabling Direct Select in Design or Add Content View</td>
<td>19-53</td>
</tr>
<tr>
<td>19.11.1</td>
<td>How to Enable and Configure Direct Select</td>
<td>19-53</td>
</tr>
<tr>
<td>19.11.2</td>
<td>What Happens at Runtime</td>
<td>19-56</td>
</tr>
<tr>
<td>19.12</td>
<td>Troubleshooting Problems with Composer Extensibility Features</td>
<td>19-57</td>
</tr>
</tbody>
</table>

## 20 Performing Advanced Composer Configurations

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>20.1</td>
<td>Enabling Custom Actions on a Show Detail Frame Component by Using Facets</td>
<td>20-1</td>
</tr>
<tr>
<td>20.2</td>
<td>Enabling Custom Actions on a Show Detail Frame Component by Using Facets: Example</td>
<td>20-2</td>
</tr>
<tr>
<td>20.2.1</td>
<td>How to Create an ADF Task Flow</td>
<td>20-2</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
<td></td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>20.2.2 How to Include an Additional Actions Facet</td>
<td>20-4</td>
<td></td>
</tr>
<tr>
<td>20.2.3 How to Create a Redirection Page</td>
<td>20-4</td>
<td></td>
</tr>
<tr>
<td>20.2.4 How to Create Navigation Rules Between Pages</td>
<td>20-5</td>
<td></td>
</tr>
<tr>
<td>20.2.5 What Happens at Runtime</td>
<td>20-6</td>
<td></td>
</tr>
<tr>
<td>20.3 Enabling Custom Actions on a Task Flow</td>
<td>20-6</td>
<td></td>
</tr>
<tr>
<td>20.3.1 How to Enable Custom Actions Directly on a Task Flow</td>
<td>20-7</td>
<td></td>
</tr>
<tr>
<td>20.3.2 How to Enable Custom Actions on a Show Detail Frame Enclosing a Task Flow</td>
<td>20-8</td>
<td></td>
</tr>
<tr>
<td>20.3.2.1 Defining Custom Actions at the Instance Level</td>
<td>20-8</td>
<td></td>
</tr>
<tr>
<td>20.3.2.2 Defining Custom Actions at the Global Level</td>
<td>20-9</td>
<td></td>
</tr>
<tr>
<td>20.3.2.3 Configuring Custom Actions that Display Task Flow Views in a Separate Browser Window</td>
<td>20-11</td>
<td></td>
</tr>
<tr>
<td>20.3.3 What Happens at Runtime</td>
<td>20-13</td>
<td></td>
</tr>
<tr>
<td>20.3.4 How to Enable Custom Actions On a Show Detail Frame Enclosing a Task Flow: Example</td>
<td>20-13</td>
<td></td>
</tr>
<tr>
<td>20.4 Enabling Custom Actions that Display on Task Flows in the Component Navigator</td>
<td>20-15</td>
<td></td>
</tr>
<tr>
<td>20.4.1 How to Configure Custom Actions in the Component Navigator</td>
<td>20-15</td>
<td></td>
</tr>
<tr>
<td>20.4.1.1 Defining the Logic for the Custom Action</td>
<td>20-16</td>
<td></td>
</tr>
<tr>
<td>20.4.1.2 Creating a JSPX Page Containing the Custom Action</td>
<td>20-17</td>
<td></td>
</tr>
<tr>
<td>20.4.1.3 Calling the JSPX Page from the Application Page Containing the Task Flow</td>
<td>20-18</td>
<td></td>
</tr>
<tr>
<td>20.4.2 What Happens at Runtime</td>
<td>20-19</td>
<td></td>
</tr>
<tr>
<td>20.5 Configuring a Keyboard Shortcut to Access Composer</td>
<td>20-19</td>
<td></td>
</tr>
<tr>
<td>20.5.1 How to Enable Linkless Entry Into Edit Mode</td>
<td>20-19</td>
<td></td>
</tr>
<tr>
<td>20.5.2 How to Configure an Event Handler for the Shortcut Key</td>
<td>20-20</td>
<td></td>
</tr>
<tr>
<td>20.5.3 What Happens at Runtime</td>
<td>20-20</td>
<td></td>
</tr>
<tr>
<td>20.6 Creating Event-Enabled Task Flows</td>
<td>20-20</td>
<td></td>
</tr>
<tr>
<td>20.7 Configuring an Application Page to Display in Structure View by Default</td>
<td>20-21</td>
<td></td>
</tr>
<tr>
<td>20.7.1 How to Configure an Application Page to Display in Structure View by Default</td>
<td>20-21</td>
<td></td>
</tr>
<tr>
<td>20.7.2 What Happens at Runtime</td>
<td>20-21</td>
<td></td>
</tr>
<tr>
<td>20.8 Disabling Structure View for the Application</td>
<td>20-21</td>
<td></td>
</tr>
<tr>
<td>20.8.1 How to Disable Structure View</td>
<td>20-22</td>
<td></td>
</tr>
<tr>
<td>20.8.2 What Happens at Runtime</td>
<td>20-22</td>
<td></td>
</tr>
<tr>
<td>20.9 Disabling Task Flow Zoom Capability</td>
<td>20-22</td>
<td></td>
</tr>
<tr>
<td>20.10 Applying Styles to Components</td>
<td>20-23</td>
<td></td>
</tr>
<tr>
<td>20.11 Configuring the Persistence Change Manager</td>
<td>20-24</td>
<td></td>
</tr>
<tr>
<td>20.12 Configuring Runtime Resource String Editing</td>
<td>20-26</td>
<td></td>
</tr>
<tr>
<td>20.12.1 Overview of the Resource String Editor</td>
<td>20-27</td>
<td></td>
</tr>
<tr>
<td>20.12.2 How to Enable the Resource String Editor in Your Application</td>
<td>20-28</td>
<td></td>
</tr>
<tr>
<td>20.12.3 How to Configure the Override Bundle</td>
<td>20-28</td>
<td></td>
</tr>
<tr>
<td>20.12.4 What Happens at Runtime</td>
<td>20-29</td>
<td></td>
</tr>
<tr>
<td>20.13 Troubleshooting Problems with Advanced Composer Configurations</td>
<td>20-29</td>
<td></td>
</tr>
</tbody>
</table>

## 21 Performing Composer-Specific MDS Configurations

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>21.1 Introduction to MDS</td>
<td>21-1</td>
</tr>
<tr>
<td>21.2 Using Composer Sandbox</td>
<td>21-2</td>
</tr>
<tr>
<td>21.2.1 How to Enable Composer Sandbox Creation</td>
<td>21-3</td>
</tr>
<tr>
<td>21.2.1.1 Updating Your Application’s adf-config.xml File</td>
<td>21-3</td>
</tr>
<tr>
<td>21.2.1.2 Updating Your Application’s web.xml File</td>
<td>21-4</td>
</tr>
</tbody>
</table>
21.2.1.3 Selecting a Database Store for Your Application .................................................. 21-5
21.2.1.4 Enabling a Full Page Refresh at Runtime ............................................................ 21-5
21.2.2 What Happens at Runtime ......................................................................................... 21-5
21.2.3 How to Disable Sandbox for an Application .......................................................... 21-6
21.2.4 How to Destroy Stale Sandboxes .............................................................................. 21-7
21.3 Adding Customization Layers to View and Edit Modes: Example ................................ 21-7
21.3.1 How to Add Composer to a JSF Page ...................................................................... 21-8
21.3.2 How to Create a Custom SiteCC Tip Layer .............................................................. 21-9
21.3.3 How to Create a Custom UserCC Tip Layer ............................................................ 21-10
21.3.4 How to Implement the ComposerSessionOptionsFactory Class.......................... 21-11
21.3.5 How to Register the Implementation with Composer ............................................ 21-13
21.3.6 How to Configure WebCenterComposerFilter ....................................................... 21-13
21.3.7 How to Redirect the Servlet to Enable Switch Between MDS Customization Layers...... 21-14
  21.3.7.1 How to Create the AppNavigationUtils Class (start here) ................................ 21-15
  21.3.7.2 How to Create AppCloseHandler ..................................................................... 21-15
  21.3.7.3 How to Register the AppCloseHandler ............................................................ 21-16
  21.3.7.4 How to Create AppModeBean .......................................................................... 21-16
21.3.8 What Happens at Runtime ......................................................................................... 21-17
21.4 Troubleshooting MDS-Related Problems with Composer ............................................ 21-17

22 Modifying Default Security Behavior of Composer Components

22.1 Applying Component-Level Restrictions by Defining Customization Policies .......... 22-1
  22.1.1 How to Define Type-Level Customization Policies ................................................. 22-2
  22.1.2 How to Define Instance-Level Customization Policies ........................................ 22-3
22.2 Applying a Component Instance-Level Customization Restriction By Using Security Roles: Example 22-4
  22.2.1 How to Configure ADF Security ............................................................................. 22-5
  22.2.2 How to Define Roles and Grant Privileges in the jazn-data.xml File ..................... 22-5
  22.2.3 How to Customize the SessionOptions Object to Include Customization Policy ... 22-8
  22.2.4 How to Register the Implementation with Composer ............................................ 22-10
  22.2.5 How to Configure WebCenterComposerFilter .................................................... 22-10
  22.2.6 How to Apply an Instance-Level Customization Restriction ................................ 22-10
  22.2.7 What Happens at Runtime ..................................................................................... 22-11
22.3 Applying Tag-Level Security Using the customizationAllowed Attribute ............... 22-12
  22.3.1 How to Enable Application Customization on an Image Component ..................... 22-13
  22.3.2 How to Restrict Customization on an Image Component ..................................... 22-13
22.4 Applying Attribute-Level Security .............................................................................. 22-13
  22.4.1 How to Define Change Persistence at the Component Level ................................ 22-14
  22.4.2 What Happens at Runtime .................................................................................... 22-14
22.5 Applying Action-Level Restrictions on Panel Customizable and Show Detail Component Actions 22-15
  22.5.1 How to Add an enableSecurity Section to adf-config.xml .................................. 22-15
  22.5.2 Defining Security at the Actions Category Level ................................................... 22-16
  22.5.3 Defining Security at the Actions Level .................................................................. 22-18
22.6 Implementing Task Flow Security .............................................................................. 22-20
  22.6.1 Granting Permissions on Task Flows .................................................................... 22-20
Part IV Integrating and Publishing Content

24 Introduction to Integrating and Publishing Content

25 Configuring Content Repository Connections

25.1 Overview of Content Adapters

25.2 Configuring Content Repository Connections

25.2.1 Creating a Content Repository Connection Based on the Oracle Content Server Adapter

25.2.2 Creating a Content Repository Connection Based on the Oracle Portal Adapter

25.2.3 Creating a Content Repository Connection Based on the Oracle WebCenter Adapter for SharePoint

25.2.3.1 Installing the Oracle WebCenter Adapter for SharePoint
| 25.2.3.2  | Creating a Content Repository Connection Based on the Oracle WebCenter Adapter for SharePoint | 25-12 |
| 25.2.3.3  | Mapping Microsoft SharePoint Content and Services | 25-14 |
| 25.2.4  | Creating a Content Repository Connection Based on the File System Adapter | 25-15 |
| 25.2.5  | What Happens When You Create a Repository Connection | 25-17 |
| 25.2.6  | About Creating a Repository Connection | 25-18 |
| 25.2.6.1  | About Using Identity Propagation and External Application Authentication Methods | 25-18 |
| 25.2.6.2  | About Oracle Portal | 25-19 |
| 25.2.6.3  | About Oracle WebCenter Adapter for SharePoint | 25-19 |
| 25.3  | Editing Content Repository Connections | 25-22 |
| 25.3.1  | How to Edit a Common Repository Connection | 25-22 |
| 25.3.2  | Editing a Framework Application-Specific Content Repository Connection | 25-23 |
| 25.4  | Using an Existing Repository Connection for a New Portal Framework Application | 25-24 |

26  Working with Content Data Controls

| 26.1  | Overview of Content Data Controls | 26-1 |
| 26.1.1  | getItems Method | 26-2 |
| 26.1.2  | search Method | 26-2 |
| 26.1.3  | advancedSearch Method | 26-3 |
| 26.1.4  | getURI Method | 26-4 |
| 26.1.5  | getAttributes Method | 26-4 |
| 26.2  | Configuring Content Data Controls for JCR Adapters | 26-4 |
| 26.2.1  | Configuring a Content Repository Data Control | 26-5 |
| 26.2.2  | What Happens When You Configure a Content Repository Data Control | 26-6 |
| 26.3  | Editing Content Repository Data Controls | 26-8 |
| 26.4  | Securing a Content Repository Data Control | 26-9 |
| 26.5  | Integrating Content Using Content Data Controls | 26-9 |
| 26.5.1  | Publishing Content As Links | 26-10 |
| 26.5.1.1  | Publishing Content As a Textual Link | 26-11 |
| 26.5.1.2  | Creating a Clickable Image to Link to a Document | 26-15 |
| 26.5.2  | What Happens at Runtime | 26-16 |
| 26.5.3  | Publishing Content in a Table | 26-17 |
| 26.5.3.1  | Displaying Files and Folders in Read-Only Format | 26-17 |
| 26.5.3.2  | Displaying the Name Attribute As a Go Link | 26-21 |
| 26.5.3.3  | Configuring a Table to Show Only Files | 26-24 |
| 26.5.4  | What Happens at Runtime | 26-26 |
| 26.5.5  | Publishing Folder Content in a Tree | 26-27 |
| 26.5.5.1  | Displaying Files and Folders in Read-Only Format | 26-27 |
| 26.5.5.2  | Displaying File Names As Hyperlinks | 26-31 |
| 26.5.6  | What Happens at Runtime | 26-33 |
| 26.5.7  | Adding Search Capabilities to Content Repositories | 26-33 |
| 26.5.7.1  | Adding Simple Search Capabilities | 26-34 |
| 26.5.7.2  | Adding Advanced Search Capabilities | 26-35 |
| 26.5.8  | What Happens at Runtime | 26-38 |
| 26.5.9  | Using Search Capabilities | 26-39 |
31 Content Management REST API
31.1 CMIS Domain Model ................................................................. 31-1
31.1.1 Data Model ........................................................................ 31-2
31.1.1.1 Repository ..................................................................... 31-2
31.1.1.2 Object ........................................................................... 31-2
31.1.1.3 Object-Type ................................................................... 31-2
31.1.1.4 Document-Type ............................................................ 31-3
31.1.1.5 Folder Object ................................................................. 31-3
31.1.1.6 Relationship Object ....................................................... 31-3
31.1.1.7 Policy Object ................................................................. 31-3
31.1.1.8 Access Control .............................................................. 31-3
31.1.1.8.1 AllowableActions Mapping ....................................... 31-3
31.1.1.9 Versioning .................................................................... 31-4
31.1.1.10 Query ......................................................................... 31-4
31.1.2 Services ........................................................................... 31-8
31.2 CMIS Part II: RESTful AtomPub Binding ................................. 31-8
31.2.1 Service Document .............................................................. 31-8
31.2.2 Response Formats .............................................................. 31-8
31.2.3 Additional Functionality ..................................................... 31-9
31.2.3.1 Folder Children Collection ........................................... 31-9
31.2.3.2 Document Entry ............................................................ 31-10
31.2.3.3 Content Stream ............................................................ 31-10
31.3 Content Management REST Service Best Practices and Examples ......................................................................... 31-11
31.3.1 Best Practices ..................................................................... 31-11
31.3.2 Content Management REST Service Examples .................. 31-12

Part V Enabling Communication and Collaboration

32 Integrating Announcements
32.1 Introduction to Announcements ................................................ 32-1
32.1.1 Understanding Announcements .......................................... 32-1
32.1.2 Requirements for Announcements ..................................... 32-2
32.1.3 What Happens at Runtime .................................................. 32-2
32.2 Basic Configuration for Announcements ................................ 32-3
32.2.1 Setting Up a Connection for Announcements ...................... 32-3
32.2.1.1 Announcements Connections ....................................... 32-3
32.2.1.2 How to Set Up Connections for Announcements .......... 32-4
32.2.2 Adding Announcements at Design Time ............................ 32-7
33 Integrating Discussions

33.1 Introduction to Discussions................................................................... 33-1
33.1.1 Understanding Discussions .............................................................. 33-1
33.1.2 Requirements for Discussions............................................................ 33-2
33.1.3 What Happens at Runtime................................................................. 33-2
33.2 Basic Configuration for Discussions....................................................... 33-3
33.2.1 Setting up Connections for Discussions............................................... 33-3
33.2.1.1 Discussions Connections ............................................................ 33-3
33.2.1.2 How to Set Up Connections for Discussions.................. 33-3
33.2.2 Adding Discussions Functionality at Design Time......................... 33-6
33.2.2.1 Discussions Task Flows ............................................................. 33-6
33.2.2.2 How to Add Discussions to a Page........................................ 33-7
33.2.2.3 How to Modify the Discussions Task Flow Parameters............... 33-8
33.2.3 Setting Security for Discussions ....................................................... 33-11
33.3 Advanced Information for Discussions................................................ 33-12
33.3.1 Adding the Discussions - Popular Topics Task Flow.................... 33-12
33.3.2 Adding the Discussions - Recent Topics Task Flow.................... 33-13
33.3.3 Adding the Discussions - Watched Forums Task Flow.................. 33-13
33.3.4 Adding the Discussions - Watched Topics Task Flow................. 33-14
33.3.5 Adding the Discussions - Quick View Task Flow......................... 33-14
33.3.6 Obtaining Portal RSS News Feed URL for Discussions............ 33-15
33.3.7 Using Custom Discussions APIs .................................................... 33-15
33.3.8 Using the Discussions REST API..................................................... 33-15
33.3.8.1 Discussions Entry Point ............................................................. 33-16
33.3.8.2 Discussions Resource Type Taxonomy ...................................... 33-16
33.3.8.3 Security Considerations............................................................... 33-16
33.3.8.4 Discussions Resource Types ....................................................... 33-16
33.3.8.4.1 urn:oracle:webcenter:discussions:forums........................................ 33-17
33.3.8.4.2 urn:oracle:webcenter:discussions:forum ........................................ 33-17
33.3.8.4.3 urn:oracle:webcenter:discussions:forum:topics............................ 33-19
33.3.8.4.4 urn:oracle:webcenter:discussions:forum:topic........................... 33-20
33.3.8.4.5 urn:oracle:webcenter:discussions:forum:topic:messages.......... 33-22
33.3.8.4.6 urn:oracle:webcenter:discussions:forum:topic:message........ 33-22
33.3.9 Troubleshooting Discussions .......................................................... 33-24

34 Integrating Instant Messaging and Presence

34.1 Introduction to Instant Messaging and Presence................................... 34-1
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>36.2.3 Setting Security for Polls</td>
<td>36-5</td>
</tr>
<tr>
<td>36.3 Advanced Information for Polls</td>
<td>36-5</td>
</tr>
<tr>
<td>36.3.1 How to Add the Polls Manager Task Flow</td>
<td>36-5</td>
</tr>
<tr>
<td>36.3.2 How to Add the Take Polls Task Flow</td>
<td>36-6</td>
</tr>
<tr>
<td>36.3.3 How to Add the View Poll Results Task Flow</td>
<td>36-6</td>
</tr>
<tr>
<td>36.3.4 Using the Polls Data Controls</td>
<td>36-6</td>
</tr>
<tr>
<td>36.3.4.1 Take Poll Data Control</td>
<td>36-6</td>
</tr>
<tr>
<td>36.3.4.1.1 createResponse</td>
<td>36-6</td>
</tr>
<tr>
<td>36.3.4.1.2 createResponseForLatestAvailablePoll</td>
<td>36-7</td>
</tr>
<tr>
<td>36.3.4.1.3 getPublishedPoll</td>
<td>36-7</td>
</tr>
<tr>
<td>36.3.4.1.4 getResult</td>
<td>36-7</td>
</tr>
<tr>
<td>36.3.4.1.5 saveResponse</td>
<td>36-7</td>
</tr>
<tr>
<td>36.3.4.2 Polls Manager Data Control</td>
<td>36-7</td>
</tr>
<tr>
<td>36.3.4.2.1 addQuestion</td>
<td>36-8</td>
</tr>
<tr>
<td>36.3.4.2.2 addQuestion</td>
<td>36-8</td>
</tr>
<tr>
<td>36.3.4.2.3 closePoll</td>
<td>36-8</td>
</tr>
<tr>
<td>36.3.4.2.4 closePollAfterDate</td>
<td>36-8</td>
</tr>
<tr>
<td>36.3.4.2.5 closePollAfterDays</td>
<td>36-9</td>
</tr>
<tr>
<td>36.3.4.2.6 closePollAfterResponses</td>
<td>36-9</td>
</tr>
<tr>
<td>36.3.4.2.7 createPoll</td>
<td>36-9</td>
</tr>
<tr>
<td>36.3.4.2.8 deletePoll</td>
<td>36-9</td>
</tr>
<tr>
<td>36.3.4.2.9 editPoll</td>
<td>36-9</td>
</tr>
<tr>
<td>36.3.4.2.10 editPoll</td>
<td>36-10</td>
</tr>
<tr>
<td>36.3.4.2.11 findPoll</td>
<td>36-10</td>
</tr>
<tr>
<td>36.3.4.2.12 getPolls</td>
<td>36-10</td>
</tr>
<tr>
<td>36.3.4.2.13 getPollsByUser</td>
<td>36-10</td>
</tr>
<tr>
<td>36.3.4.2.14 getPollsByUserCount</td>
<td>36-10</td>
</tr>
<tr>
<td>36.3.4.2.15 publishPoll</td>
<td>36-10</td>
</tr>
<tr>
<td>36.3.4.2.16 publishPollOnDate</td>
<td>36-11</td>
</tr>
</tbody>
</table>

### Part VI Working with People Connections

#### 37 Introducing the People Connections Service

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>37.1 Overview of the People Connections Service</td>
<td>37-1</td>
</tr>
<tr>
<td>37.1.1 People Connections Service Task Flows</td>
<td>37-2</td>
</tr>
<tr>
<td>37.1.1.1 Activity Stream</td>
<td>37-2</td>
</tr>
<tr>
<td>37.1.1.2 Connections</td>
<td>37-5</td>
</tr>
<tr>
<td>37.1.1.3 Feedback</td>
<td>37-7</td>
</tr>
<tr>
<td>37.1.1.4 Message Board</td>
<td>37-8</td>
</tr>
<tr>
<td>37.1.1.5 Profile</td>
<td>37-10</td>
</tr>
<tr>
<td>37.1.1.6 The Publisher Task Flow</td>
<td>37-12</td>
</tr>
<tr>
<td>37.1.2 People Connections Service Requirements</td>
<td>37-13</td>
</tr>
<tr>
<td>37.2 Troubleshooting the People Connections Service</td>
<td>37-13</td>
</tr>
</tbody>
</table>

#### 38 Basic Configuration for the People Connections Service

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>38.1 How to Set Up a Database Connection for the People Connections Service</td>
<td>38-1</td>
</tr>
</tbody>
</table>
38.2 How to Add People Connections Task Flows to a Page .................................................. 38-1
38.3 Setting Security for the People Connections Service .................................................. 38-2
38.4 Establishing Site-Level Settings for People Connections Features ............................. 38-3
38.4.1 Understanding the People Connections Site-Level Setting Scripts .......................... 38-3
38.4.1.1 Preparing and Running settings-insert.sql .......................................................... 38-3
38.4.1.2 Preparing and Running settings-update.sql .......................................................... 38-4
38.4.2 Supported Site-Level Settings for People Connections Features ............................. 38-5
38.4.2.1 Activity Stream Site-Level Settings ................................................................. 38-5
38.4.2.2 Connections Site-Level Settings ......................................................................... 38-6
38.4.2.3 Feedback Site-Level Settings ............................................................................. 38-7
38.4.2.4 Message Board Site-Level Settings ................................................................... 38-8
38.4.2.5 Profile Site-Level Settings .................................................................................. 38-9
38.4.3 Example: Configuring Connections to Accept Invitations Automatically ............... 38-11

39 People Connections Task Flow Binding Parameters

39.1 About People Connections Task Flow Binding Parameters ........................................... 39-1
39.2 How to Revise People Connections Task Flow Binding Parameters ............................ 39-1
39.3 People Connections Task Flow Binding Parameters ...................................................... 39-3
39.4 About the Activity Stream Advanced Query Option ..................................................... 39-12

40 Using People Connections Data Controls and Java APIs

40.1 Using People Connections Data Controls ...................................................................... 40-1
40.1.1 Adding a Data Control to Your Project ...................................................................... 40-1
40.1.2 Working with People Connections Management Data Control ............................... 40-3
40.1.2.1 connectionLists .................................................................................................. 40-4
40.1.2.2 groupedPeopleCounts ...................................................................................... 40-4
40.1.2.3 receivedInvitations ............................................................................................ 40-4
40.1.2.4 sentInvitations .................................................................................................. 40-5
40.1.2.5 acceptInvitation(String invitationId) .................................................................. 40-5
40.1.2.6 addConnectionsToList(List userids, String listName) ....................................... 40-5
40.1.2.7 createConnectionList(String listName) ............................................................. 40-6
40.1.2.8 declineInvitation(String invitationId) ................................................................. 40-6
40.1.2.9 dropConnectionList(String listName) ................................................................. 40-6
40.1.2.10 getConnectionList(String connectionListName) ............................................ 40-6
40.1.2.11 getConnections(String userid, String connectionListName, String filterPattern, String sortBy, int startIndex, int fetchSize) .................................................. 40-7
40.1.2.12 getNumberOfConnections(String userid, String connectionListName, String filterPattern) ................................................................. 40-7
40.1.2.13 ignoreInvitation(String invitationId) .................................................................. 40-7
40.1.2.14 inviteUserForConnection(String userid, String invitationMessage, Collection connectionListNames) ................................................................. 40-8
40.1.2.15 isConnectedListModifiable(String listName) .................................................... 40-8
40.1.2.16 isConnectedPartOfUnmodifiableList(String connecteeGuid) ............................ 40-9
40.1.2.17 removeConnection(String userid, boolean fromListOnly, String listName) .......... 40-9
40.1.2.18 searchUsers(String filterPattern, int startIndex, int fetchSize) ....................... 40-9
40.1.2.19 updateConnectionListsMembership(String member, Collection addListIds, Collection removeListIds) ................................................................. 40-10
## Part VII  Helping Users Find Content

### 43 Integrating Links

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

### 44 Integrating Tags

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>44.1 Introduction to Tags</td>
<td>44-1</td>
</tr>
<tr>
<td>44.1.1 Understanding Tags</td>
<td>44-2</td>
</tr>
<tr>
<td>44.1.2 Requirements for Tags</td>
<td>44-2</td>
</tr>
<tr>
<td>44.1.3 What Happens at Runtime</td>
<td>44-3</td>
</tr>
<tr>
<td>44.2 Basic Configuration for Tags</td>
<td>44-3</td>
</tr>
<tr>
<td>44.2.1 Setting up Connections for Tags</td>
<td>44-3</td>
</tr>
<tr>
<td>44.2.2 Adding Tags Functionality at Design Time</td>
<td>44-3</td>
</tr>
<tr>
<td>44.2.2.1 Tags Components</td>
<td>44-3</td>
</tr>
<tr>
<td>44.2.2.2 Tags Task Flows</td>
<td>44-4</td>
</tr>
<tr>
<td>44.2.2.3 How to Add Tags Functionality to your Page</td>
<td>44-5</td>
</tr>
<tr>
<td>44.2.2.4 How to Modify Tags Task Flow Parameters</td>
<td>44-7</td>
</tr>
<tr>
<td>44.2.3 Setting Security for Tags</td>
<td>44-8</td>
</tr>
<tr>
<td>44.3 Advanced Information for Tags</td>
<td>44-9</td>
</tr>
<tr>
<td>44.3.1 Using the Tags Java APIs</td>
<td>44-9</td>
</tr>
<tr>
<td>44.3.2 Using the Tags REST APIs</td>
<td>44-9</td>
</tr>
<tr>
<td>44.3.2.1 Tags Entry Point</td>
<td>44-10</td>
</tr>
<tr>
<td>44.3.2.2 Tags Resource Type Taxonomy</td>
<td>44-10</td>
</tr>
<tr>
<td>44.3.2.3 Security Considerations</td>
<td>44-10</td>
</tr>
<tr>
<td>44.3.2.4 Tags Resource Types</td>
<td>44-11</td>
</tr>
<tr>
<td>44.3.2.4.1 urn:oracle:webcenter:tagging:tags</td>
<td>44-11</td>
</tr>
<tr>
<td>44.3.2.4.2 urn:oracle:webcenter:tagging:tag</td>
<td>44-12</td>
</tr>
</tbody>
</table>
45 Integrating Search

45.1 Introduction to Search ................................................................. 45-1
45.1.1 Understanding Search ................................................................. 45-2
45.1.1.1 Understanding Search with WebCenter Portal Live Search ........ 45-2
45.1.1.2 Understanding Search with the Oracle SES Adapter ................. 45-2
45.1.1.3 Click Actions on Search Results .............................................. 45-3
45.1.2 What Happens at Runtime .......................................................... 45-3
45.2 Basic Configuration for Search .......................................................... 45-3
45.2.1 Configuration Roadmap - Oracle SES ........................................... 45-3
45.2.2 Setting up Connections for Oracle SES ......................................... 45-7
45.2.3 Adding Search at Design Time ....................................................... 45-9
45.2.3.1 Search Task Flows ................................................................. 45-9
45.2.3.2 How to Add Search to Your Page ............................................ 45-10
45.2.3.3 How to Customize Search ..................................................... 45-11
45.2.4 Setting Security for Search ............................................................ 45-11
45.3 Advanced Information for Search .................................................... 45-11
45.3.1 Configuring Search with the Oracle SES Adapter ......................... 45-12
45.3.2 Configuring Search with WebCenter Portal’s Live Search ............. 45-13
45.3.3 Using the Search Java APIs ......................................................... 45-14
45.3.4 Using the Search REST APIs ....................................................... 45-14
45.3.4.1 Search Entry Point ................................................................. 45-15
45.3.4.2 Search Resource Type Taxonomy ........................................... 45-15
45.3.4.3 Security Considerations ....................................................... 45-18
45.3.4.4 Search Resource Types ......................................................... 45-18
45.3.4.4.1 urn:oracle:webcenter:searchcollection:links ....................... 45-18
45.3.4.4.2 urn:oracle:webcenter:searchcollection:results .................... 45-18
45.3.4.4.3 urn:oracle:webcenter:searchcollection:facets ....................... 45-18
45.3.5 Using the Search Data Control ..................................................... 45-18
45.3.5.1 Search Data Control Methods, Attributes, and Classes ............. 45-18
45.3.5.1.1 SearchServiceDCRows ..................................................... 45-19
45.3.5.1.2 SearchServiceDCRefiner .................................................. 45-19
45.3.5.1.3 SearchServiceDCRefinement ............................................. 45-19
45.3.5.2 Integrating the Search Data Control ........................................ 45-20
45.3.6 Building Adapters for Search .................................................... 45-20
45.3.6.1 How to Add a Search Source .................................................. 45-20
45.3.6.2 How to Register a Custom Adapter .......................................... 45-27
46  Integrating the Activity Graph

46.1  Introduction to the Activity Graph .............................................................. 46-1
46.1.1  Understanding the Activity Graph .......................................................... 46-1
46.1.2  Requirements for the Activity Graph ...................................................... 46-8
46.1.3  What Happens at Runtime ..................................................................... 46-9
46.2  Basic Configuration for the Activity Graph ................................................. 46-9
46.2.1  Configuration Roadmap for the Activity Graph ...................................... 46-9
46.2.2  Setting Up Connections for the Activity Graph ...................................... 46-11
46.2.3  Adding the Activity Graph at Design Time ............................................ 46-12
46.2.3.1  Activity Graph Task Flows .................................................................. 46-12
46.2.3.1.1  The Recommended Connections Task Flow .................................. 46-13
46.2.3.1.2  The Similar Portals Task Flow ....................................................... 46-13
46.2.3.1.3  The Similar Items Task Flow .......................................................... 46-14
46.2.3.1.4  The Top Items Task Flow ............................................................... 46-15
46.2.3.2  How to Add Activity Graph Task Flows to a Page .............................. 46-15
46.2.3.3  How to Modify Activity Graph Task Flow Parameters ....................... 46-16
46.2.3.4  Activity Graph Task Flows and Task Flow Parameters ....................... 46-17
46.3  Advanced Information for the Activity Graph ............................................. 46-18
46.3.1  Using the Recommendation Data Control .............................................. 46-19
46.3.2  Using the Activity Graph REST APIs ................................................... 46-20
46.3.2.1  Activity Graph Entry Point ............................................................... 46-20
46.3.2.2  Activity Graph Resource Type Taxonomy ......................................... 46-20
46.3.2.3  Security Considerations .................................................................... 46-21
46.3.2.4  Activity Graph Resource Types ......................................................... 46-21
46.3.2.4.1  urn:oracle:webcenter:activitygraph:recommendations ............... 46-21
46.3.2.4.2  urn:oracle:webcenter:activitygraph:recommendations:recommendation 46-23
46.3.2.4.3  urn:oracle:webcenter:activitygraph:items .................................. 46-23
46.3.2.4.4  urn:oracle:webcenter:activitygraph:items:item .......................... 46-24
46.4  Extending the Activity Graph .................................................................... 46-26
46.4.1  Defining Custom Node Classes .............................................................. 46-26
46.4.2  Defining Custom Actions ....................................................................... 46-27
46.4.3  Defining Custom Similarity Calculations .............................................. 46-29
46.4.4  Defining Custom Rank Calculations ..................................................... 46-30
46.4.5  Registering Custom Activity Providers .................................................. 46-32
47 Integrating Analytics

47.1 Introduction to Analytics ........................................................................................................... 47-1
47.1.1 Understanding Analytics ....................................................................................................... 47-1
47.1.2 Requirements for the Analytics Service .................................................................................. 47-3
47.1.3 What Happens at Runtime .................................................................................................... 47-3
47.2 Basic Configuration for Analytics .............................................................................................. 47-4
47.2.1 Configuration Roadmap for Analytics .................................................................................... 47-4
47.2.2 Setting up Connections for Analytics ..................................................................................... 47-6
47.2.2.1 How to Set Up a Connection to the Analytics Database .................................................... 47-7
47.2.2.2 How to Set Analytics Collector Properties ......................................................................... 47-7
47.2.2.3 How to Set Up a Connection to the Analytics Collector ................................................... 47-8
47.2.3 Adding Analytics Event Code to Your Application ............................................................... 47-9
47.2.3.1 Including Event Code for Page Views ................................................................................. 47-9
47.2.4 Configuring a Namespace for Analytics Customizations in MDS ........................................... 47-10
47.2.5 Adding Analytics Task Flows at Design Time ....................................................................... 47-10
47.2.5.1 Analytics Task Flows ......................................................................................................... 47-11
47.2.5.2 How to Add Analytics Task Flows to a Page .................................................................... 47-11
47.2.5.3 How to Modify Analytics Task Flow Parameters ............................................................... 47-13
47.2.5.4 How to Allow End Users to Customize Analytics Task Flows at Runtime ..................... 47-14
47.2.5.5 Analytics Task Flows and Task Flow Parameters ............................................................... 47-15
47.2.6 Setting up Security for Analytics Task Flows and Usage Data ............................................. 47-16
47.3 Building Analytics Reports ...................................................................................................... 47-17
47.3.1 Using SQL Data Controls .................................................................................................... 47-17
47.3.2 SQL Statements for Out-of-the-Box Analytics Reports .......................................................... 47-20
47.3.2.1 Analytics SQL: WebCenter Portal Traffic ....................................................................... 47-20
47.3.2.2 Analytics SQL: Page Traffic ............................................................................................. 47-22
47.3.2.3 Analytics SQL: Login Metrics ........................................................................................... 47-22
47.3.2.4 Analytics SQL: Portal Traffic ........................................................................................... 47-22
47.3.2.5 Analytics SQL: Portal Response Time ............................................................................. 47-22
47.3.2.6 Analytics SQL: Portlet Traffic .......................................................................................... 47-23
47.3.2.7 Analytics SQL: Portlet Instance Traffic ........................................................................... 47-23
47.3.2.8 Analytics SQL: Portlet Response Time ............................................................................ 47-23
47.3.2.9 Analytics SQL: Portlet Instance Response Time ............................................................... 47-24
47.3.2.10 Analytics SQL: Search Metrics ......................................................................................... 47-24
47.3.2.11 Analytics SQL: Document Metrics ................................................................................... 47-24
47.3.2.12 Analytics SQL: Wiki Metrics .......................................................................................... 47-25
47.3.2.13 Analytics SQL: Blog Metrics .......................................................................................... 47-25
47.3.2.14 Analytics SQL: Discussion Metrics .................................................................................. 47-25
47.3.3 Sample Queries for User Metrics ........................................................................................... 47-25
47.3.3.1 Sample SQL: Filter by User Property ............................................................................... 47-26
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>47.3.3.2</td>
<td>Sample SQL: Group by User Property</td>
<td>47-26</td>
</tr>
<tr>
<td>47.3.4</td>
<td>Sample Analytics Database Queries for Specific Metrics</td>
<td>47-26</td>
</tr>
<tr>
<td>47.3.4.1</td>
<td>Sample SQL: User Activities</td>
<td>47-26</td>
</tr>
<tr>
<td>47.3.4.2</td>
<td>Sample SQL: Portal Activities</td>
<td>47-27</td>
</tr>
<tr>
<td>47.3.4.3</td>
<td>Sample SQL: Portal Activities for a Specific Time Period</td>
<td>47-28</td>
</tr>
<tr>
<td>47.3.4.4</td>
<td>Sample SQL: Activity for a Service During a Specific Time Period</td>
<td>47-29</td>
</tr>
<tr>
<td>47.3.4.5</td>
<td>Sample SQL: Search Phrases</td>
<td>47-29</td>
</tr>
<tr>
<td>47.3.4.6</td>
<td>Sample SQL: Page Views</td>
<td>47-30</td>
</tr>
</tbody>
</table>

**Part VIII  Helping Users Keep Track**

### 48 Integrating Events

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>48.1</td>
<td>Introduction to Events</td>
<td>48-1</td>
</tr>
<tr>
<td>48.1.1</td>
<td>Understanding Events</td>
<td>48-1</td>
</tr>
<tr>
<td>48.1.2</td>
<td>Requirements for Events</td>
<td>48-2</td>
</tr>
<tr>
<td>48.1.3</td>
<td>What Happens at Runtime</td>
<td>48-2</td>
</tr>
<tr>
<td>48.2</td>
<td>Basic Configuration for Events</td>
<td>48-2</td>
</tr>
<tr>
<td>48.2.1</td>
<td>Configuration Roadmap for Events</td>
<td>48-3</td>
</tr>
<tr>
<td>48.2.2</td>
<td>Setting Up a Connection for Events</td>
<td>48-4</td>
</tr>
<tr>
<td>48.2.3</td>
<td>Adding Events Functionality at Design Time</td>
<td>48-6</td>
</tr>
<tr>
<td>48.2.3.1</td>
<td>Events Task Flows</td>
<td>48-6</td>
</tr>
<tr>
<td>48.2.3.2</td>
<td>How to Add Events Functionality to Your Application</td>
<td>48-7</td>
</tr>
<tr>
<td>48.2.4</td>
<td>Setting Security for Events</td>
<td>48-8</td>
</tr>
<tr>
<td>48.3</td>
<td>Using the Events REST API</td>
<td>48-8</td>
</tr>
<tr>
<td>48.3.1</td>
<td>Events Entry Point</td>
<td>48-8</td>
</tr>
<tr>
<td>48.3.2</td>
<td>Events Resource Type Taxonomy</td>
<td>48-8</td>
</tr>
<tr>
<td>48.3.3</td>
<td>Security Considerations</td>
<td>48-9</td>
</tr>
<tr>
<td>48.3.4</td>
<td>Events Resource Types</td>
<td>48-9</td>
</tr>
<tr>
<td>48.3.4.1</td>
<td>urn:oracle:webcenter:events:gsEvents</td>
<td>48-9</td>
</tr>
<tr>
<td>48.3.4.2</td>
<td>urn:oracle:webcenter:events:gsEvent</td>
<td>48-10</td>
</tr>
<tr>
<td>48.3.4.3</td>
<td>urn:oracle:webcenter:events:gsCategories</td>
<td>48-12</td>
</tr>
<tr>
<td>48.4</td>
<td>Troubleshooting Events</td>
<td>48-12</td>
</tr>
</tbody>
</table>

### 49 Integrating Lists

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>49.1</td>
<td>Introduction to Lists</td>
<td>49-1</td>
</tr>
<tr>
<td>49.1.1</td>
<td>Understanding Lists</td>
<td>49-1</td>
</tr>
<tr>
<td>49.1.2</td>
<td>What Happens to Lists at Runtime</td>
<td>49-1</td>
</tr>
<tr>
<td>49.2</td>
<td>Basic Configuration for Lists</td>
<td>49-2</td>
</tr>
<tr>
<td>49.2.1</td>
<td>Setting Up Connections for Lists</td>
<td>49-2</td>
</tr>
<tr>
<td>49.2.2</td>
<td>Adding Lists Functionality at Design Time</td>
<td>49-2</td>
</tr>
<tr>
<td>49.2.2.1</td>
<td>Lists Task Flow</td>
<td>49-2</td>
</tr>
<tr>
<td>49.2.2.2</td>
<td>How to Add Lists Functionality to a Framework Application</td>
<td>49-3</td>
</tr>
<tr>
<td>49.2.3</td>
<td>Enabling Users to Work with Lists at Runtime</td>
<td>49-3</td>
</tr>
<tr>
<td>49.3</td>
<td>Adding Lists to a Resource Catalog</td>
<td>49-3</td>
</tr>
<tr>
<td>49.4</td>
<td>Using the Lists REST API</td>
<td>49-4</td>
</tr>
<tr>
<td>49.4.1</td>
<td>Entry Point for Lists</td>
<td>49-4</td>
</tr>
</tbody>
</table>
50.3.1 Using Notifications Java APIs ................................................................. 50-15
50.3.1.1 Configuration Settings Required to Use Notifications Java APIs .................... 50-16
50.3.1.2 Introduction to Notifications Java APIs ..................................................... 50-16
50.3.1.3 How to Set up Your Application to Use Notifications Java APIs .................... 50-16
50.3.2 Using Notifications Data Controls ............................................................. 50-18
50.3.2.1 Adding a Data Control to Your Project .................................................... 50-18
50.3.2.2 Understanding the Notification Subscription Data Control ......................... 50-19
50.3.2.3 Notification Subscription Data Control Methods and Attributes ................. 50-19
50.3.2.3.1 Method: getUserSubscriptionPreferences ............................................. 50-19
50.3.2.3.2 Method: getScopeSubscriptionPreferences .......................................... 50-19
50.3.2.3.3 Method: saveUserSubscriptionPreferences ........................................... 50-20
50.3.2.3.4 Method: saveScopeSubscriptionPreferences ......................................... 50-20
50.3.2.3.5 Method: getSubscriptions ..................................................................... 50-20
50.3.2.3.6 Method: unsubscribe ........................................................................... 50-21
50.3.2.4 Notification Subscription Data Control Classes ........................................... 50-21
50.3.2.4.1 SubscriptionPreference ....................................................................... 50-21
50.3.2.4.2 Subscription ....................................................................................... 50-22

51 Integrating Recent Activities

51.1 Introduction to Recent Activities ........................................................................ 51-1
51.1.1 Understanding Recent Activities .................................................................... 51-2
51.1.2 What Happens at Runtime ............................................................................ 51-2
51.2 Basic Configuration for Recent Activities .......................................................... 51-3
51.2.1 Setting up Connections for Recent Activities ................................................ 51-3
51.2.2 Adding Recent Activities at Design Time ..................................................... 51-3
51.2.3 Modifying Recent Activities Task Flow Parameters ....................................... 51-6
51.2.4 Setting Security for Recent Activities ............................................................ 51-6
51.3 Advanced Information for Recent Activities ..................................................... 51-6
51.3.1 Refining the Behavior of Recent Activities .................................................... 51-6
51.3.2 Obtaining RSS News Feed URL for Recent Activities .................................... 51-7

52 Integrating RSS

52.1 Introduction to RSS ........................................................................................ 52-1
52.1.1 Understanding RSS .................................................................................... 52-1
52.1.2 Requirements for RSS ................................................................................ 52-2
52.1.3 What Happens at Runtime ......................................................................... 52-2
52.2 Basic Configuration for the RSS ................................................................. 52-3
52.2.1 Setting Up a Proxy Server for RSS ............................................................. 52-3
52.2.2 Adding RSS Functionality at Design Time ................................................ 52-4
52.2.2.1 About RSS Viewer Task Flow ................................................................. 52-4
52.2.2.2 How to Add the RSS Viewer Task Flow to Your Application .................. 52-4
52.2.2.3 How to Modify RSS Viewer Task Flow Parameters ................................ 52-5
52.2.3 Setting Security for the RSS Viewer ........................................................... 52-6

Part IX Extending WebCenter Portal and Portal Framework Applications
Consuming a Siebel Web Service in a Framework Application ........................................ 54-5
Creating a Data Control Based on a Siebel Web Service ........................................ 54-5
Creating a Page Flow Diagram .................................................................................. 54-6
Adding Pages and Navigation to the Page Flow Diagram ........................................ 54-6
Creating the Query Page ........................................................................................... 54-7
Creating the Results Page .......................................................................................... 54-8
Testing the Application .............................................................................................. 54-8
Consuming a Siebel Web Service Data Control in a Portal ........................................ 54-9
Integrating Siebel Applications Using the Siebel Web Engine ................................... 54-10
Integrating E-Business Suite Applications .................................................................. 54-11
Introduction to Integrating EBS Applications ............................................................ 54-11
Understanding EBS Integration .................................................................................. 54-11
Requirements for Integrating EBS Applications .......................................................... 54-11
Required Configurations for Integrating EBS ............................................................. 54-12
Preparing OID for Use Without Single Sign-On ......................................................... 54-12
Creating a User in EBS and Assigning a Responsibility ............................................. 54-13
Configuring the EBS Applications Profile Options .................................................... 54-15
Adding the WebCenter Host as a Trusted Portal Using AutoConfig .......................... 54-17
Integrating EBS Applications as WSRP Portlets ....................................................... 54-17
Preparing the EBS Portlet for Remote Access ........................................................... 54-17
Integrating EBS Applications in a Framework Application ......................................... 54-20
Preparing the EBS Portlet for Remote Access ........................................................... 54-20
Registering the EBS WSRP Producer in the Framework Application ...................... 54-20
Creating a JSF Page to Consume the Remote Producer ........................................... 54-20
Testing the Framework Application .......................................................................... 54-21
Integrating EBS Applications in a Portal .................................................................... 54-21
Preparing the EBS Portlet for Remote Access ........................................................... 54-21
Registering the EBS WSRP Producer in WebCenter Portal ....................................... 54-21
Adding the EBS Portlet to a Portal Page ..................................................................... 54-22
Testing the Portal Portlet Connection ....................................................................... 54-23
Integrating EBS Applications as Data Controls in WebCenter Portal ....................... 54-23
Generating the WSDL ............................................................................................... 54-23
Adding a Web Service Data Control to a Portal Page ............................................... 54-24
Integrating JD Edwards Applications ......................................................................... 54-26
Integrating JD Edwards Applications in a Framework Application ......................... 54-26
Preparing the JD Edwards Application for Remote Access ...................................... 54-26
Registering the JD Edwards WSRP Producer in the Framework Application ............ 54-26
Creating the JSF Page to Consume the Remote Producer ........................................ 54-27
Testing the Framework Application .......................................................................... 54-27
Integrating JD Edwards Applications in a Portal ....................................................... 54-27
Registering the Producer ......................................................................................... 54-27
Adding the JD Edwards Portlet to a Portal Page ....................................................... 54-28
Testing the Portal Portlet Connection ....................................................................... 54-29
Integrating PeopleSoft Applications ........................................................................ 54-29
Introduction to Integrating PeopleSoft Applications .................................................. 54-29
Understanding PeopleSoft Integration ..................................................................... 54-29
Requirements for Integrating PeopleSoft Applications ............................................. 54-29
54.5.2 Integrating PeopleSoft Applications as WSRP Portlets ............................................ 54-30
54.5.2.1 Preparing the PeopleSoft Application for Remote Access ........................................ 54-30
54.5.2.2 Configuring WS-Security for PeopleTools 8.52 and Later ........................................ 54-33
54.5.2.3 Attaching a WS-Security Policy to WebCenter Portal ........................................... 54-38
54.5.2.3.1 Configuring WSS 1.0 SAML Token with Message Integrity ............................. 54-38
54.5.2.3.2 Configuring WSS 1.0 Username Token Without Password ............................ 54-39
54.5.2.3.3 Configuring WSS 1.0 SAML Token with Message Protection ....................... 54-41
54.5.2.3.4 Configuring WSS 1.0 Username Token with Password ................................... 54-41
54.5.2.4 Integrating PeopleSoft Applications in a Portal .................................................. 54-42
54.5.2.4.1 Registering the PeopleSoft WSRP Producer for a Portal .................................. 54-42
54.5.2.4.2 Adding the PeopleSoft Portlet to a Portal Page ................................................ 54-43
54.5.2.4.3 Testing the Portal Portlet Connection .............................................................. 54-43
54.5.2.5 Integrating PeopleSoft Applications in a Framework Application ......................... 54-44
54.5.2.5.1 Registering the PeopleSoft WSRP Producer in the Framework Application ....... 54-44
54.5.2.5.2 Creating the JSF Page to Consume the Remote (WSRP) Producer .................. 54-46
54.5.2.5.3 Testing the Framework Application ..................................................................... 54-46
54.5.2.6 Configuring WS-Security for PeopleTools 8.51 ................................................... 54-46
54.5.2.6.1 Configuring WS-Security for WSS10 SAML Token with Message Integrity ....... 54-47
54.5.2.6.2 Configuring WS-Security for WSS10 SAML Token with Message Protection .... 54-47
54.5.2.6.3 Configuring WS-Security for WSS10 Username Token with Password ......... 54-48
54.5.3 Integrating PeopleSoft Applications as Data Controls in WebCenter Portal .......... 54-49
54.5.3.1 Preparing the WSDL ............................................................................................ 54-49
54.5.3.2 Creating a Web Service Data Control ................................................................. 54-54
54.6 Integrating Oracle Business Intelligence Presentation Services .................................... 54-56
54.6.1 Introduction to Integrating Oracle Business Intelligence Presentation Services ....... 54-56
54.6.1.1 Understanding Oracle Business Intelligence Presentation Services Integration .... 54-56
54.6.1.2 Requirements for Integrating Oracle Business Intelligence Presentation Services .... 54-57
54.6.1.3 Advanced Integration Options .............................................................................. 54-57
54.6.2 Configuring Credentials for Connecting to the Oracle BI Presentation Catalog .. 54-58
54.6.2.1 Checking for the BIImpersonatorUser ................................................................ 54-58
54.6.2.2 Creating the BIImpersonatorUser ...................................................................... 54-59
54.6.2.3 Granting Permissions to BIImpersonatorUser .................................................... 54-60
54.6.3 Integrating Oracle Business Intelligence Objects in WebCenter Portal ................. 54-61
54.6.3.1 Adding or Modifying a Presentation Services Connection After Deployment .......... 54-62
54.6.3.2 Adding Oracle BI Objects to the WebCenter Portal Resource Catalog ............. 54-63
54.6.3.3 Adding Oracle BI Content at Runtime ................................................................. 54-64
54.6.3.4 Modifying a Business Intelligence Object's Prompt Values .............................. 54-64
54.6.3.5 Modifying a Business Intelligence Task Flow’s Initialization Parameters .......... 54-65

55 Developing Components for WebCenter Portal Using JDeveloper

55.1 Developing Assets for WebCenter Portal ....................................................................... 55-1
55.1.1 About the WebCenterSpacesResources Project .......................................................... 55-1
55.1.2 Setting Up Your JDeveloper Environment for Asset Development ..................... 55-3
55.1.3 Enabling Direct Download and Uploads to WebCenter Portal ............................ 55-5
55.1.4 Opening and Exploring the WebCenterSpacesResources Project ..................... 55-7
55.1.5 Editing WebCenter Portal Assets in JDeveloper .................................................. 55-9
55.1.5.1 Downloading an Asset Directly from WebCenter Portal to JDeveloper .......... 55-10
55.1.5.2 Downloading an Asset to an Archive from WebCenter Portal ....................... 55-12
55.1.5.3 Importing a WebCenter Portal Asset to JDeveloper ....................................... 55-12
55.1.5.4 How to Build Assets for WebCenter Portal ................................................... 55-15
55.1.6 Uploading Assets Directly to WebCenter Portal ................................................ 55-15
55.1.7 Exporting WebCenter Portal Assets to an Archive ............................................ 55-17
55.1.8 Troubleshooting Asset Round-Trip Development ............................................. 55-17
55.2 Developing Task Flows, Data Controls, and Managed Beans for WebCenter Portal .. 55-18
55.2.1 Understanding the WebCenter Portal Server Extension Template .................... 55-19
55.2.1.1 How the WebCenter Portal Server Extension Workspace Is Organized ........... 55-20
55.2.1.2 The PortalSharedLibrary Project ................................................................... 55-20
55.2.1.2.1 Versioning extend.spaces.webapp.war ..................................................... 55-21
55.2.1.3 The PortalExtension Project ......................................................................... 55-21
55.2.2 Understanding Shared Library Development for WebCenter Portal .................. 55-22
55.2.2.1 General Documentation for ADF Library Development ............................... 55-25
55.2.2.2 General Documentation for Shared Library Deployment ............................. 55-25
55.2.3 Creating a WebCenter Portal Server Extension Workspace ............................ 55-26
55.2.4 Registering Additional Shared Libraries With WebCenter Portal .................... 55-30
55.2.5 Deploying Extensions to the WebCenter Portal Shared Library (extend.spaces.webapp.war) 55-32
55.2.5.1 Deploying Extensions Directly to the Portal Server ....................................... 55-32
55.2.5.2 Deploying Extensions to an Archive ............................................................. 55-37
55.2.5.3 Reverting to a Previous WebCenter Portal Shared Library Version ............... 55-38
55.2.6 Developing ADF Library Components for WebCenter Portal Using the PortalExtension Project 55-39
55.2.7 Troubleshooting WebCenter Portal Shared Library Deployment ..................... 55-39

56 Integrating with WebCenter Portal

56.1 Exposing Portals in Your Application Using the WebCenter Portal API .................. 56-1
56.1.1 Introduction to the WebCenter Portal API ......................................................... 56-2
56.1.2 Case Study 1: Purchasing Application Uses a Portal to Evaluate Suppliers .......... 56-3
56.1.3 Case Study 2: Customer Support Center Application Uses a Portal to Discuss Customer Escalations 56-5
56.1.4 How to Set Up Your Framework Application to Use the WebCenter Portal API 56-6
56.1.4.1 Verifying That WebCenter Portal Is Up and Running .................................. 56-6
56.1.4.2 Setting Up WebCenter Portal to Use the WebCenter Portal API .................. 56-7
56.1.4.3 Securing the Connection Between the Portal Framework Application and WebCenter Portal 56-9
56.1.4.4 Setting Up the WebCenter Portal Client Context ........................................ 56-9
56.1.5 Providing WebCenter Portal Functionality in Your Portal Framework Application ........ 56-10
56.1.5.1 Managing Portals and Portal Templates ....................................................... 56-12
56.1.5.1.1 Creating a Portal .................................................................................... 56-12
56.1.5.1.2 Creating a Custom Attribute .......................................................... 56-12
56.1.5.1.3 Creating a Portal Template ............................................................ 56-13
56.1.5.1.4 Deleting a Portal ............................................................................ 56-13
56.1.5.2 Managing Portal Membership .......................................................... 56-14
56.1.5.2.1 Adding Members to a Portal ........................................................ 56-14
56.1.5.2.2 Inviting Users to Join a Portal....................................................... 56-15
56.1.5.2.3 Removing Members from a Portal .............................................. 56-15
56.1.5.2.4 Retrieving Role Information ....................................................... 56-16
56.1.5.3 Retrieving Information for Portals and Portal Templates .............. 56-16
56.1.5.3.1 Retrieving a List of Portals .......................................................... 56-17
56.1.5.3.2 Retrieving a List of Public Ports .................................................. 56-17
56.1.5.3.3 Retrieving a List of Discoverable Ports ...................................... 56-17
56.1.5.3.4 Retrieving Portal Metadata ......................................................... 56-18
56.1.5.3.5 Retrieving a List of Portal Templates ......................................... 56-20
56.1.5.3.6 Retrieving Portal Template Metadata ....................................... 56-20
56.1.5.3.7 Retrieving the Portal URL .......................................................... 56-21
56.1.5.3.8 Retrieving a Portal URL .............................................................. 56-21
56.1.5.3.9 Retrieving RSS Feed URLs for WebCenter Portal Tools and Services ... 56-22
56.1.6 How to Handle Exceptions Raised by the WebCenter Portal API ....... 56-23
56.1.6.1 Providing Localized Error Messages ............................................. 56-24
56.1.6.2 Listing the Error Stack ................................................................... 56-24
56.1.7 Finding More Information on the WebCenter Portal API ............... 56-25
56.1.8 Troubleshooting Issues with the WebCenter Portal API .................. 56-25
56.2 Using the WebCenter Portal REST API ............................................... 56-26
56.2.1 Portal Entry Point ............................................................................. 56-26
56.2.2 Portal Resource Type Taxonomy ....................................................... 56-27
56.2.3 Portal Security Considerations ......................................................... 56-27
56.2.4 Portal Resource Types ...................................................................... 56-27
56.2.4.1 um:oracle:webcenter:spaces ...................................................... 56-28
56.2.4.2 um:oracle:webcenter:siteresources ........................................... 56-29
56.2.4.3 um:oracle:webcenter:spaces:resource:templates ..................... 56-33
56.2.4.4 um:oracle:webcenter:space ........................................................ 56-33
56.2.4.5 um:oracle:webcenter:space:resource:index .............................. 56-36
56.2.4.6 um:oracle:webcenter:space:roles .............................................. 56-37
56.2.4.7 um:oracle:webcenter:space:role ................................................ 56-37
56.2.4.8 um:oracle:webcenter:space:icon .............................................. 56-38
56.2.4.9 um:oracle:webcenter:space:attributes ...................................... 56-38
56.2.4.10 um:oracle:webcenter:space:attribute ...................................... 56-39
56.2.4.11 um:oracle:webcenter:space:members ...................................... 56-40
56.2.4.12 um:oracle:webcenter:space:member ........................................ 56-41
56.2.4.13 um:oracle:webcenter:space:lists ............................................... 56-42
56.2.4.14 um:oracle:webcenter:space:list ................................................. 56-43
56.2.4.15 um:oracle:webcenter:space:list:rows ..................................... 56-45
56.2.4.16 um:oracle:webcenter:space:list:row ........................................ 56-46
56.2.4.17 um:oracle:webcenter:space:list:columns ................................ 56-48
56.2.4.18 um:oracle:webcenter:space:list:column .................................. 56-49
56.3 Exposing Enterprise Applications in Portals ....................................... 56-50

xxxix
Part X  Working with Portlets and Pagelets

57  Introduction to Portlets
57.1  Introduction to Portlets ................................................................. 57-1
57.2  Portlet Anatomy ............................................................................ 57-2
57.3  Portlet Resources .......................................................................... 57-4
57.3.1  Prebuilt Portlets ................................................................. 57-4
57.3.2  Java Server Faces (JSF) Portlets ........................................... 57-4
57.3.3  Custom Java Portlets ............................................................ 57-5
57.3.4  OmniPortlet ............................................................................ 57-5
57.3.5  Web Clipping Portlet ............................................................ 57-6
57.3.6  Portlet Technologies ................................................................ 57-7
57.3.7  When To Use Portlets Versus Task Flows? ............................ 57-13
57.4  Portlet Development .................................................................... 57-14
57.4.1  Portlet Producer Applications ...................................................... 57-14
57.4.2  Portlet Modes ........................................................................ 57-16
57.4.2.1  View Mode ........................................................................ 57-17
57.4.2.1.1  HTML Guidelines for Rendering Portlets .................... 57-17
57.4.2.1.2  CSS Guidelines for Rendering Portlets ................. 57-18
57.4.2.2  Edit Mode .......................................................................... 57-18
57.4.2.2.1  Guidelines for Edit Mode Operations .................. 57-19
57.4.2.2.2  Guidelines for Buttons in Edit Mode .................... 57-19
57.4.2.3  Edit Defaults Mode .......................................................... 57-19
57.4.2.3.1  Guidelines for Edit Defaults Mode Operation .... 57-20
57.4.2.3.2  Guidelines for Buttons in Edit Defaults Mode ...... 57-20
57.4.2.3.3  Guidelines for Rendering Customization Values .... 57-20
57.4.2.4  Help Mode ........................................................................ 57-20
57.4.2.4.1  Guidelines for Help Mode ........................................ 57-21
57.4.2.5  About Mode ....................................................................... 57-21
57.4.2.5.1  Guidelines for About Mode .................................... 57-21
57.4.3  Interportlet Communication ..................................................... 57-21
57.4.4  Portlet Personalization and Customization ............................ 57-22
57.4.5  Portlet Performance ............................................................... 57-22
57.4.6  Multilanguage Portlets ............................................................ 57-24
57.4.7  Portlet Deployment ................................................................. 57-25

58  Creating Portlets from JSF Applications Using the Oracle JSF Portlet Bridge
58.1  Introduction to the Oracle JSF Portlet Bridge ......................... 58-1
58.2  Creating a Portlet from a JSF Application ............................... 58-2
58.2.1  How to Create a JSF Portlet Based on a Page ............... 58-2
58.2.2  What Happens When You Create a JSF Portlet Based on a Page ................................................................. 58-3
58.2.3  How to Create a JSF Portlet Based on a Task Flow ......... 58-4
Creating a Portlet From a Task Flow Using the Create Portlet Entry Dialog... 58-5
Creating a Portlet From a Task Flow Using the Manage Portlet Entries of Task Flows Dialog 58-6
What Happens When You Create a Portlet Based on a Task Flow ......................... 58-6
Updating the Portlet Entry for a Portletized Page or Task Flow .......................... 58-7
Testing a Portletized Page or Task Flow in JDeveloper ........................................ 58-8
Testing a JSF Portlet Using the Integrated WebLogic Server .............................. 58-9
Deploying JSF Portlets to a WebLogic Managed Server ....................................... 58-10
Using Events to Link JSF Portlets with Other Portlets .......................................... 58-11
How to Link a JSF Portlet to Another JSF Portlet .................................................. 58-11
How to Link a JSF Portlet to a JSR 286 Portlet ...................................................... 58-15
How to Link a JSR 286 Portlet to a JSF Portlet ....................................................... 58-17
What You May Need to Know About Portlet Events in the Oracle JSF Portlet Bridge.... 58-20
What You May Need to Know When Creating a JSF Portlet .................................... 58-21
General Guidelines .................................................................................................. 58-21
Application Must Run as a Web Application ......................................................... 58-21
The Portlet Is Contained by a Different Page .......................................................... 58-22
Servlet Dependencies ............................................................................................. 58-22
Secured Applications .............................................................................................. 58-22
Deep Links .............................................................................................................. 58-22
Java EE Authentication ............................................................................................ 58-23
Timeout Period ........................................................................................................ 58-23
Portlet Guidelines .................................................................................................... 58-23
Portlet Sizing ............................................................................................................ 58-23
Resources and Links ............................................................................................... 58-24
Redirecting Requests ............................................................................................. 58-24
Memory Consumption ............................................................................................ 58-24
WSRP Version ........................................................................................................ 58-24
Security Guidelines ................................................................................................ 58-24
Oracle ADF Secured Applications ......................................................................... 58-24
Role Based Authorization ....................................................................................... 58-25
JSF Guidelines ........................................................................................................ 58-25
Using h:commandLink ......................................................................................... 58-25
Oracle ADF Guidelines .......................................................................................... 58-25
Task Flow Returns .................................................................................................. 58-26
Mismatched Style Sheets ....................................................................................... 58-26
ADF Faces Dialog Framework ............................................................................... 58-27
Composer Components ......................................................................................... 58-28
The fileDownloadActionListener Component .................................................... 58-28
The prepareModel Phase ...................................................................................... 58-28
Accessing Request Parameters ............................................................................. 58-28
Referencing Page Parameters ............................................................................... 58-28
Trinidad Components ............................................................................................ 58-29
Using ADF Faces Client-Side APIs to Find Components ....................................... 58-29
Encoded URLs ....................................................................................................... 58-29
Issues with goLink ................................................................................................ 58-30
59 Building Standards-Based Java Portlets Using JSR 286

59.1 Introduction to Standards-Based Java Portlets .......................................................... 59-1
59.2 Creating a JSR 286 Java Portlet ............................................................................. 59-2
59.2.1 How to Create a JSR 286 Java Portlet ............................................................... 59-3
59.2.2 What Happens When You Create a JSR 286 Portlet Using the JDeveloper Wizard 59-9
59.3 Developing JSR 286 Java Portlets ....................................................................... 59-10
59.3.1 How to Edit the Portlet Deployment Descriptor File ........................................... 59-11
59.3.2 How to Add Custom Portlet Modes to JSR 286 Portlets ................................. 59-12
59.3.3 How to Access User Information in JSR 286 Portlets ....................................... 59-13
59.3.4 How to Customize the Runtime Environment for JSR 286 Portlets .................. 59-14
59.3.4.1 Supported Container Runtime Options ......................................................... 59-14
59.3.4.2 Setting Container Runtime Options for All Portlets in an Application ...... 59-22
59.3.4.3 Setting Container Runtime Options for Individual Portlets ....................... 59-22
59.3.5 How to Use Public Render Parameters in JSR 286 Portlets .............................. 59-23
59.3.5.1 Declaring a Public Render Parameter at the Application Level .................. 59-23
59.3.5.2 Defining a Public Render Parameter for a Portlet .......................................... 59-25
59.3.5.3 Example: Public Render Parameters ........................................................... 59-26
59.3.6 How to Use Portlet Events in JSR 286 Portlets .................................................. 59-28
59.3.6.1 Declaring a Portlet Event at the Application Level ..................................... 59-28
59.3.6.2 Defining a Publishing Event for a Portlet ...................................................... 59-30
59.3.6.3 Defining a Publishing Event for a Portlet ...................................................... 59-30
59.3.6.4 Example: Portlet Events ............................................................................ 59-31
59.3.7 How to Add Portlet Preferences to JSR 286 Portlets ......................................... 59-34
59.3.7.1 Adding a Portlet Preference to a JSR 286 Portlet ......................................... 59-34
59.3.7.2 Example: Simple Portlet Personalization ..................................................... 59-35
59.3.8 How to Use Portlet Filters in JSR 286 Portlets .................................................. 59-38
59.3.8.1 Adding a Portlet Filter to an Application ..................................................... 59-38
59.3.8.2 Applying a Portlet Filter to a Portlet .......................................................... 59-39
59.3.9 How to Implement Interportlet Communication Across Different Pages ........ 59-40
59.3.10 How to Enhance JSR 286 Portlet Performance with Caching ........................ 59-42
59.3.10.1 Implementing Expiry-Based Caching in JSR 286 Portlets ............................ 59-43
59.3.10.2 Implementing Validation-Based Caching in JSR 286 Portlets ...................... 59-44
59.3.11 How to Implement Rewritten URLs for Resource Proxy ............................... 59-45
59.3.12 How to Implement Stateless Resource Proxying ............................................. 59-45
59.3.13 How to Implement Security for JSR 286 Portlets ........................................... 59-46
59.3.14 How to Manage the Persistence Store for JSR 286 Portlets ............................ 59-46
59.3.14.1 Setting Up a Persistence Store for a WSRP Producer ................................. 59-46
59.3.14.2 Migrating a WSRP Producer Persistence Store ........................................... 59-48
59.3.14.2.1 Migrating a Database or File-Based Persistence Store ......................... 59-48
59.3.14.2.2 Migrating to or from a Consumer Persistence Store .............................. 59-50
59.3.14.2.3 Migrating a WSRP Portlet Producer ...................................................... 59-51
59.3.14.3 Enabling Java Object Cache for Database Persistence Store Access ............ 59-51
59.3.15 How to Export and Import Portlet Producers at Design Time ....................... 59-52
59.3.15.1 Exporting Portlet Producers at Design Time .............................................. 59-52
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>62.1.2.5</td>
<td>Pagelet Producer Proxy</td>
<td>62-7</td>
</tr>
<tr>
<td>62.1.3</td>
<td>Requirements</td>
<td>62-8</td>
</tr>
<tr>
<td>62.2</td>
<td>Configuring Pagelet Producer Settings</td>
<td>62-8</td>
</tr>
<tr>
<td>62.2.1</td>
<td>Configuring a WCI Data Source</td>
<td>62-8</td>
</tr>
<tr>
<td>62.2.2</td>
<td>Configuring Logging Settings</td>
<td>62-9</td>
</tr>
<tr>
<td>62.2.3</td>
<td>Configuring Proxy Settings</td>
<td>62-9</td>
</tr>
<tr>
<td>62.2.4</td>
<td>Configuring Transform Settings</td>
<td>62-10</td>
</tr>
<tr>
<td>62.2.5</td>
<td>Configuring CSP Settings</td>
<td>62-11</td>
</tr>
<tr>
<td>62.2.6</td>
<td>Configuring Kerberos Settings</td>
<td>62-11</td>
</tr>
<tr>
<td>62.2.7</td>
<td>Configuring OpenSocial Settings</td>
<td>62-11</td>
</tr>
<tr>
<td>62.3</td>
<td>Creating Pagelet Producer Objects</td>
<td>62-12</td>
</tr>
<tr>
<td>62.3.1</td>
<td>Creating Resources</td>
<td>62-13</td>
</tr>
<tr>
<td>62.3.1.1</td>
<td>Configuration Pages: Web and CSP Resources</td>
<td>62-14</td>
</tr>
<tr>
<td>62.3.1.1.1</td>
<td>Configuring General Settings</td>
<td>62-14</td>
</tr>
<tr>
<td>62.3.1.1.2</td>
<td>Configuring CSP settings</td>
<td>62-15</td>
</tr>
<tr>
<td>62.3.1.1.3</td>
<td>Configuring Policy Settings</td>
<td>62-15</td>
</tr>
<tr>
<td>62.3.1.1.4</td>
<td>Configuring Autologin</td>
<td>62-16</td>
</tr>
<tr>
<td>62.3.1.1.5</td>
<td>Configuring Autologin: Form Login</td>
<td>62-17</td>
</tr>
<tr>
<td>62.3.1.1.6</td>
<td>Configuring Autologin: Basic Login and NTLM Login</td>
<td>62-18</td>
</tr>
<tr>
<td>62.3.1.1.7</td>
<td>Configuring Autologin: Kerberos Login</td>
<td>62-18</td>
</tr>
<tr>
<td>62.3.1.1.8</td>
<td>Configuring Autologin: Authentication Sources</td>
<td>62-18</td>
</tr>
<tr>
<td>62.3.1.1.9</td>
<td>Configuring Headers</td>
<td>62-19</td>
</tr>
<tr>
<td>62.3.1.2</td>
<td>Configuring WSRP and JPDK Resources</td>
<td>62-20</td>
</tr>
<tr>
<td>62.3.1.2.1</td>
<td>Configuring General Settings</td>
<td>62-20</td>
</tr>
<tr>
<td>62.3.1.2.2</td>
<td>Configuring Policy Settings</td>
<td>62-21</td>
</tr>
<tr>
<td>62.3.1.3</td>
<td>Configuring OpenSocial Resources (OpenSocial Gadget Producers)</td>
<td>62-21</td>
</tr>
<tr>
<td>62.3.1.3.1</td>
<td>Configuring General Settings</td>
<td>62-21</td>
</tr>
<tr>
<td>62.3.1.3.2</td>
<td>Configuring Policy Settings</td>
<td>62-21</td>
</tr>
<tr>
<td>62.3.2</td>
<td>Creating Pagelets</td>
<td>62-22</td>
</tr>
<tr>
<td>62.3.2.1</td>
<td>Configuring General Settings</td>
<td>62-22</td>
</tr>
<tr>
<td>62.3.2.2</td>
<td>Configuring Preferences</td>
<td>62-23</td>
</tr>
<tr>
<td>62.3.2.3</td>
<td>Configuring Parameters</td>
<td>62-23</td>
</tr>
<tr>
<td>62.3.2.4</td>
<td>Configuring the Clipper</td>
<td>62-24</td>
</tr>
<tr>
<td>62.3.2.5</td>
<td>Accessing the Pagelet and Preference Editor</td>
<td>62-25</td>
</tr>
<tr>
<td>62.3.3</td>
<td>Creating Web Injectors</td>
<td>62-25</td>
</tr>
<tr>
<td>62.3.3.1</td>
<td>Configuring General Settings</td>
<td>62-25</td>
</tr>
<tr>
<td>62.3.3.2</td>
<td>Injecting Content</td>
<td>62-26</td>
</tr>
<tr>
<td>62.3.4</td>
<td>Creating Custom Parsers</td>
<td>62-27</td>
</tr>
<tr>
<td>62.3.5</td>
<td>Creating Hosted Files</td>
<td>62-28</td>
</tr>
<tr>
<td>62.4</td>
<td>Working with Pagelet Chrome for WSRP and Oracle JPDK Portlets</td>
<td>62-29</td>
</tr>
<tr>
<td>62.5</td>
<td>Working with OpenSocial Gadgets</td>
<td>62-31</td>
</tr>
<tr>
<td>62.5.1</td>
<td>Configuring Authentication</td>
<td>62-31</td>
</tr>
<tr>
<td>62.5.2</td>
<td>Storing User Preferences</td>
<td>62-31</td>
</tr>
<tr>
<td>62.5.3</td>
<td>Accessing WebCenter Portal Profile Information</td>
<td>62-32</td>
</tr>
<tr>
<td>62.5.4</td>
<td>Accessing a User’s Activity Stream</td>
<td>62-32</td>
</tr>
<tr>
<td>62.5.5</td>
<td>Using Gadget Eventing</td>
<td>62-33</td>
</tr>
<tr>
<td>62.5.6</td>
<td>Example: Consuming an External OpenSocial Gadget</td>
<td>62-33</td>
</tr>
</tbody>
</table>
62.8.4.2.1 Establishing Identity in Pagelet Producer ............................................. 62-79
62.8.4.2.2 Using a Login Page Supplied by Pagelet Producer ................................. 62-80
62.8.4.3 Propagating Identity from Pagelet Producer to the Backend Application .... 62-81
62.8.4.3.1 WSRP/JSDK Portlets .............................................................................. 62-81
62.8.4.3.2 Stand-alone Backend Application Protected with Native Authentication (Non-SSO) 62-81
62.8.4.3.3 Identity Propagation with Autologin.................................................... 62-81
62.8.4.3.4 Stand-alone Backend Application Protected with SSO .......................... 62-82
62.8.4.3.5 Using Direct Identity Propagation to OAM Protected Backend Application .... 62-82
62.8.4.3.6 Using Identity Propagation with Autologin and SSO ............................. 62-84
62.8.4.3.7 Reference: Common Autologin Configurations ..................................... 62-84
62.8.4.3.8 Autologin Settings for Oracle Access Manager (OAM)............................ 62-84
62.8.4.3.9 Autologin Settings for Oracle SSO (OSSO) ........................................... 62-85
62.8.5 Consuming WSRP Portlets as Pagelets ..................................................... 62-85
62.8.5.1 Exposing Custom ADF Task Flows as WSRP Portlets............................... 62-86
62.8.5.1.1 Exposing WSRP Portlets Developed for Oracle WebLogic Portal ............ 62-86
62.8.5.1.2 Exposing WLP Portlets Using WLP as WSRP Producer ........................ 62-86
62.8.5.1.3 Configuring WS Security between WLP WSRP Producer and WebCenter Consumer 62-87
62.8.5.1.4 Registering the WLP WSRP Producer in Pagelet Producer .................. 62-87
62.8.5.1.5 Adding WLP WSRP Portlets to Sites .................................................. 62-88
62.8.6 Consuming WebCenter Portal Services as Pagelets in Sites ........................ 62-88
62.8.6.1 Requirements ....................................................................................... 62-88
62.8.6.2 Configuring Security and Single Sign-On ................................................ 62-89
62.8.6.2.1 Establishing User Identity Propagation: OAM Configuration ................ 62-89
62.8.6.2.2 Configuring the GUID Attribute in the Identity Store ............................ 62-89
62.8.6.2.3 Configuring SSO for Discussions Server ............................................ 62-90
62.8.6.3 Registering WebCenter Services Exposed as WSRP Portlets .................. 62-90
62.8.6.3.1 Adding Pagelets to Sites Pages ......................................................... 62-90
62.8.7 Consuming Applications as Pagelets Using EBS11i ...................................... 62-91
62.8.7.1 Creating a Resource for Basic URL Mapping (Proxy) ............................... 62-92
62.8.7.2 Configuring Autologin ............................................................................ 62-92
62.8.7.3 Creating a Pagelet .................................................................................. 62-93
62.8.7.4 Making Corrective Configurations ......................................................... 62-93
62.8.7.4.1 Creating a Pluggable Parser for Popup URL Rewriting ......................... 62-93
62.8.7.4.2 Creating an Injector to Disable Frame-busting ..................................... 62-94
62.8.7.4.3 Creating an Injector to Auto-Resize the iFrame .................................... 62-95
62.8.7.4.4 Setting Up Clipping by JavaScript Injection ......................................... 62-95
62.8.7.4.5 Creating an Injector for Pagelet Restyling .......................................... 62-96
62.8.7.4.6 Testing the Pagelets ............................................................................ 62-97
62.8.7.4.7 Final Result ......................................................................................... 62-98
62.8.7.4.8 Troubleshooting ............................................................................... 62-99
62.8.8 Consuming OpenSocial Gadgets Using Pagelet Producer ............................. 62-99
62.8.9 Manipulating HTML Markup ....................................................................... 62-100
62.8.9.1 Using an Injector with JavaScript .......................................................... 62-100
63 Consuming Portlets

63.1 Introduction to Consuming Portlets ....................................................... 63-1
63.2 Registering Portlet Producers with a WebCenter Portal Framework Application ........ 63-3
63.2.1 Registering a WSRP Portlet Producer with a WebCenter Portal Framework Application 63-3
63.2.1.1 How to Register a WSRP Portlet Producer ........................................ 63-3
63.2.1.2 How to Map a Producer’s Declared User Categories to an Application’s Defined Java EE Security Roles 63-8
63.2.2 Registering an Oracle PDK-Java Portlet Producer with a WebCenter Portal Framework Application 63-9
63.3 Managing Portlet Producer Connections .................................................. 63-11
63.3.1 How to Edit Portlet Producer Registration Settings ............................... 63-11
63.3.2 How to Test a Portlet Producer Connection ........................................... 63-12
63.3.3 How to Refresh a Portlet Producer ......................................................... 63-12
63.3.4 How to Delete a Portlet Producer ............................................................ 63-13
63.3.5 How to Edit Portlet Client Configuration ............................................... 63-13
63.4 Adding the Portlet Producer Task Flow .................................................... 63-16
63.5 Adding Portlets to a Page ......................................................................... 63-17
63.5.1 How to Add a Portlet to a Page ............................................................... 63-18
63.5.2 What Happens When You Add a Portlet to a Page ............................... 63-19
63.5.3 What You May Need to Know About Interportlet Communication ............. 63-21
63.5.4 What You May Need to Know About Inline Frames ............................... 63-22
63.5.5 What You May Need to Know About Portlet Sizing ................................ 63-23
63.5.6 What You May Need to Know About Minimize, Restore, and Move .......... 63-25
63.5.7 What Happens at Runtime ................................................................... 63-26
63.6 Setting Attribute Values for the Portlet Tag ............................................. 63-26
63.6.1 How to Set Attribute Values for the Portlet Tag Using the Property Inspector.... 63-27
63.6.2 How to Set Attribute Values for the Portlet Tag in Source Code ................ 63-27
63.6.3 Common Attributes of the Portlet Tag ................................................... 63-28
63.6.4 Appearance Attributes of the Portlet Tag .............................................. 63-30
63.6.5 Behavior Attributes of the Portlet Tag .................................................... 63-33
63.6.6 Portlet Modes Attributes of the Portlet Tag ............................................ 63-34
63.6.7 Style Attributes of the Portlet Tag ......................................................... 63-35
63.6.8 Binding Attributes of the Portlet Tag ..................................................... 63-36
63.6.9 Customization Attributes of the Portlet Tag .......................................... 63-36
63.6.10 Other Attributes of the Portlet Tag ....................................................... 63-36
63.7 Manually Wiring Parameters and Events in JSR 286 Portlets ....................... 63-37
63.7.1 How to Manually Link Portlets with Public Render Parameters .................. 63-37
63.7.1.1 Manually Linking Parameters Using Page Variables ............................ 63-38
63.7.1.2 Manually Linking Parameters Using the ParametersChange ADF Contextual Event 63-39
63.7.2 How to Manually Link Portlets with Portlet Events .................................. 63-40
63.8 Copying Portlets ................................................................. 63-40
63.8.1 How to Copy a Portlet and Place it on the Same Page .... 63-41
63.8.2 How to Copy a Portlet from One Application Page to Another ... 63-43
63.9 Deleting Portlets from Application Pages .................................. 63-44
63.10 Consuming WebCenter Services Portlets ........................... 63-45
63.10.1 How to Consume WebCenter Services Portlets in Oracle Portal ... 63-46
63.10.2 How to Consume WebCenter Services Portlets in Oracle WebLogic Portal ... 63-47
63.10.3 How to Consume WebCenter Services Portlets in Oracle WebCenter Interaction .... 63-47
63.10.4 How to Set WebCenter Services Portlets Parameters............... 63-48
63.10.4.1 Common WebCenter Services Portlets Parameters .................. 63-48
63.10.4.2 Document Manager Portlet Parameters ............................. 63-48
63.10.4.3 Discussion Forums Portlet Parameters ................................. 63-50
63.10.4.4 Blogs Portlet Parameters .......................................................... 63-50
63.10.4.5 Lists Portlet Parameters ......................................................... 63-51
63.10.4.6 Polls Manager Portlet Parameters .............................................. 63-51
63.10.4.7 Take Polls Portlet Parameters ................................................... 63-51
63.10.4.8 Announcements Portlet Parameters ........................................... 63-51
63.10.4.9 Mail Portlet Parameters ............................................................ 63-51
63.10.4.10 Activity Stream Portlet Parameters ......................................... 63-52
63.10.4.11 Tag Cloud Portlet Parameters ................................................. 63-53
63.10.5 WebCenter Services Portlets Limitations ............................. 63-53
63.10.6 Troubleshooting Problems with WebCenter Services Portlets .......... 63-54
63.10.6.1 External Link in Announcements or Discussion Forum Portlet Not Working ........ 63-54
63.10.6.2 Get a Link Option in Document Manager and Blogs Portlets Not Working ... 63-55
63.10.6.3 Recommendations Tab in Document Manager and Blogs Portlets Is Empty ... 63-55
63.10.6.4 Unable to Embed an Image into a Blog ....................................... 63-56
63.10.6.5 Selected Text Is Lost When Creating a Link in a Blog Post or Wiki ........ 63-56
63.11 Troubleshooting Portlets ................................................... 63-56
63.11.1 Diagnostic Tools for Troubleshooting Portlets ...................... 63-56
63.11.1.1 Identify the Portlet Instance ......................................................... 63-57
63.11.1.2 Examine the Portlet Consumer Test Page ................................. 63-58
63.11.1.2.1 Access the Portlet Consumer Test Page .................................. 63-59
63.11.1.2.2 Locate the Portlet Producer ...................................................... 63-60
63.11.1.2.3 Locate and Run the Portlet Instance ....................................... 63-60
63.11.1.2.4 Perform Sanity Checks .......................................................... 63-61
63.11.1.2.5 Check Consumer Configuration Entries ............................... 63-62
63.11.1.3 Examine the Producer Test Page .............................................. 63-62
63.11.1.3.1 Access the Producer Test Page ............................................... 63-64
63.11.1.3.2 Run the JSF Portlet as a Servlet ............................................. 63-64
63.11.1.3.3 Examine the SOAP Monitor .................................................. 63-64
63.11.2 Configuring the Portlet Logging File ...................................... 63-65
63.11.3 Portlet Displays a Portlet Consumer Error ............................ 63-67
63.11.4 Portlet Displays a Portlet Timeout ......................................... 63-68
63.11.5 Portlet Displays a Remote Portlet Communication Error ............ 63-70
63.11.6 Portlet Displays a Remote Portlet Error .................................... 63-71
64  Creating Portlets with OmniPortlet
64.1  Introduction to OmniPortlet................................................................. 64-1
64.2  Adding OmniPortlet to Your Application.............................................. 64-2
64.3  Customizing OmniPortlet........................................................................ 64-2
64.4  OmniPortlet Configuration Tips............................................................. 64-3
64.4.1  Configuring the OmniPortlet Producer to Access Data Outside a Firewall .... 64-4
64.4.2  Configuring the OmniPortlet Producer to Access Other Relational Databases .... 64-7
64.4.2.1  Installing DataDirect JDBC Drivers .................................................... 64-7
64.4.2.2  Registering DataDirect Drivers in OmniPortlet................................. 64-7
64.4.3  Configuring Portal Tools and Web Producers (Optional)....................... 64-10
64.5  Troubleshooting OmniPortlet Problems................................................ 64-12

65  Creating Content-Based Portlets with Web Clipping
65.1  Introduction to Web Clipping................................................................. 65-1
65.2  Adding Web Clipping to Your Application............................................. 65-3
65.3  Integrating Authenticated Web Content Using Single Sign-On.................. 65-4
65.4  Advanced Features of Web Clipping....................................................... 65-6
65.4.1  Using the Web Clipping Transport API.................................................. 65-6
65.4.1.1  Implementing the Web Clipping Transport API ...................................... 65-6
65.4.1.2  Deploying the Web Clipping Transport API......................................... 65-7
65.4.2  Rewriting Image Links to Use a Resource Proxy.................................... 65-7
65.5  Web Clipping Portlet Configuration Tips ................................................. 65-8
65.5.1  Web Clipping Repository Configuration.............................................. 65-8
65.5.1.1  Using Oracle Metadata Services (MDS) as the Web Clipping Repository .... 65-10
65.5.1.2  Using an Oracle Database as the Web Clipping Repository.................. 65-10
65.5.1.2.1  Using the Database Schema Created by RCU ..................................... 65-11
65.5.1.2.2  Creating a Database Schema............................................................. 65-11
65.5.1.3  Configuring Web Clipping Repository in provider.xml ....................... 65-12
65.5.1.4  Attributes and Child Tags of the repositoryInfo Tag........................... 65-13
65.5.1.5  Migrating the Web Clipping Repository.............................................. 65-15
65.5.2  HTTP or HTTPS Proxy Configuration................................................ 65-15
65.5.3  Web Clipping Producer Security........................................................ 65-17
65.5.3.1  Adding Certificates for Trusted Sites............................................... 65-17
65.5.3.2  Configuring Oracle Advanced Security for the Web Clipping Producer .... 65-18
65.6  Current Limitations of Web Clipping..................................................... 65-19
65.7  Troubleshooting Web Clipping ............................................................. 65-20

Part XI  Delivering Personalized Content

66  Personalizing Oracle WebCenter Portal Applications
66.1  Introduction to Personalization............................................................. 66-1
66.1.1  Personalization Architecture............................................................... 66-1
66.1.2  The Conductor..................................................................................... 66-2
66.1.2.1  Scenarios ......................................................................................... 66-2
66.1.2.2  Personalized Application Flow ........................................................ 66-3
66.1.2.3  Property Set Integration................................................................. 66-3
68 Implementing Custom Function Providers: Introduction

68.1 Introduction to Function Providers ................................................................. 68-1
68.2 Do I Need to Write a Function Provider? ....................................................... 68-1
68.3 Discovering Function Providers .................................................................... 68-2
68.4 Implementing a Simple Custom Function Provider ....................................... 68-3
68.4.1 Overview of the Example and Prerequisites ............................................. 68-3
68.4.2 Example Function Provider ....................................................................... 68-4
68.4.3 Adding Java Class Annotations ................................................................. 68-6
68.4.4 Writing Function Provider Methods ....................................................... 68-7
68.5 Deploying a Custom Function Provider ....................................................... 68-7
68.5.1 Compile Your Function Provider Class(es) ............................................. 68-8
68.5.2 Create a Resource Properties File ........................................................... 68-8
68.5.3 Create a Configuration File in META-INF/services ............................... 68-9
68.5.4 Set Up the JAR File Structure ................................................................. 68-10
68.5.5 Create a JAR File .................................................................................... 68-12
68.5.6 Copy the JAR File to the Server .............................................................. 68-12
68.5.7 Test the Deployment ................................................................................ 68-13
68.6 Calling a Function Provider in a Scenario ................................................... 68-13

69 Function Provider Reference

Cache Methods ..................................................................................................... 69-2
getCache ............................................................................................................... 69-3
get ......................................................................................................................... 69-4
clear .................................................................................................................... 69-5
getCustomCache ................................................................................................. 69-6
getLargeCache ..................................................................................................... 69-7
getch MediumCache .......................................................................................... 69-8
getSmallCache ..................................................................................................... 69-9
put ......................................................................................................................... 69-10
Maps Methods ..................................................................................................... 69-11
70 Using the Property Service

70.1 Introduction to the Property Service ................................................................. 70-1
70.1.1 What Is the Property Service Used For? ......................................................... 70-2
70.1.2 What Are Property Sets? .................................................................................. 70-2
70.1.3 What Is a Locator? ............................................................................................ 70-3
70.1.4 Caching for the Property Service ..................................................................... 70-4
70.1.5 Using the Property Service with Scenarios ..................................................... 70-4
70.1.6 Architectural Overview ..................................................................................... 70-4
70.2 Creating a Property Set Definition in JDeveloper .............................................. 70-4
70.3 Deploying the Property Set Definition ............................................................... 70-6
70.4 Testing the Property Set Definition Deployment ............................................... 70-7
70.4.1 Locating the Property Set Definition Using REST ........................................... 70-7
70.4.2 Locating the Property Set Definition Using the Data Provider ....................... 70-8
70.5 Using the Property Service Data Provider ......................................................... 70-8
70.5.1 Accessing the Property Service Data Provider in a Scenario ......................... 70-8
70.5.2 Property Service Data Provider Method Reference .......................................... 70-9
70.5.2.1 getProperty(definition, set, property) ......................................................... 70-10
70.5.2.2 getProperty(namespace, definition, set, property) ....................................... 70-10
70.5.2.3 getPropertySet(definition, set) ................................................................. 70-10
70.5.2.4 getPropertySet(namespace, definition, set) ................................................ 70-11
70.5.2.5 getPropertySets(definition, search, restrictions) ........................................ 70-11
70.5.2.6 getPropertySets(namespace, definition, search, restrictions) ....................... 70-11
70.6 Using the Property Service Caching Service ..................................................... 70-12
70.7 Implementing a Custom Locator ....................................................................... 70-12
70.7.1 Overview of the Example and Prerequisites .................................................... 70-13
70.7.2 Creating a Property Set Definition ................................................................. 70-13
70.7.3 Making Up Data: Implement a Simple Library Class ...................................... 70-14
70.7.4 Backing the Data with a JavaBean ................................................................. 70-14
70.7.5 Implementing a Java Data Access Object ......................................................... 70-16
70.7.6 Implementing the ILocator Class ..................................................................... 70-17
### 71 Property Service REST APIs

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>71.1</td>
<td>Introduction to Property Service REST APIs</td>
<td>71-1</td>
</tr>
<tr>
<td>71.2</td>
<td>Property Service REST APIs</td>
<td>71-1</td>
</tr>
<tr>
<td>71.2.1</td>
<td>Property Service Resource Index</td>
<td>71-2</td>
</tr>
<tr>
<td>71.2.2</td>
<td>Namespace CRUD</td>
<td>71-5</td>
</tr>
<tr>
<td>71.2.2.1</td>
<td>Create Namespace</td>
<td>71-5</td>
</tr>
<tr>
<td>71.2.2.2</td>
<td>Retrieve Namespace</td>
<td>71-8</td>
</tr>
<tr>
<td>71.2.2.3</td>
<td>Update Namespace</td>
<td>71-10</td>
</tr>
<tr>
<td>71.2.2.4</td>
<td>Delete Namespace</td>
<td>71-11</td>
</tr>
<tr>
<td>71.2.2.5</td>
<td>Retrieve all Namespaces</td>
<td>71-12</td>
</tr>
<tr>
<td>71.2.2.6</td>
<td>Delete all Namespaces</td>
<td>71-25</td>
</tr>
<tr>
<td>71.2.3</td>
<td>Property Definition CRUD</td>
<td>71-26</td>
</tr>
<tr>
<td>71.2.3.1</td>
<td>Create Property Definition - Create Integer Property Definition</td>
<td>71-26</td>
</tr>
<tr>
<td>71.2.3.2</td>
<td>Create Property Definition - Create Integer Array Property Definition</td>
<td>71-28</td>
</tr>
<tr>
<td>71.2.3.3</td>
<td>Retrieve Property Definition</td>
<td>71-31</td>
</tr>
<tr>
<td>71.2.3.4</td>
<td>Update Property Definition</td>
<td>71-33</td>
</tr>
<tr>
<td>71.2.3.5</td>
<td>Delete Property Definition</td>
<td>71-34</td>
</tr>
<tr>
<td>71.2.3.6</td>
<td>Retrieve all Property Definitions</td>
<td>71-35</td>
</tr>
<tr>
<td>71.2.3.7</td>
<td>Delete all Property Definitions</td>
<td>71-38</td>
</tr>
<tr>
<td>71.2.4</td>
<td>Property Set Definition CRUD</td>
<td>71-39</td>
</tr>
<tr>
<td>71.2.4.1</td>
<td>Create Property Set Definition</td>
<td>71-39</td>
</tr>
<tr>
<td>71.2.4.2</td>
<td>Retrieve Property Set Definition</td>
<td>71-43</td>
</tr>
<tr>
<td>71.2.4.3</td>
<td>Retrieve All Property Set Definitions</td>
<td>71-44</td>
</tr>
<tr>
<td>71.2.4.4</td>
<td>Delete all Property Set Definitions</td>
<td>71-45</td>
</tr>
<tr>
<td>71.2.5</td>
<td>Property Set CRUD</td>
<td>71-46</td>
</tr>
<tr>
<td>71.2.5.1</td>
<td>Create Property Set</td>
<td>71-46</td>
</tr>
<tr>
<td>71.2.5.2</td>
<td>Retrieve Property Set</td>
<td>71-52</td>
</tr>
<tr>
<td>71.2.5.3</td>
<td>Update Property Set</td>
<td>71-57</td>
</tr>
<tr>
<td>71.2.5.4</td>
<td>Delete Property Set</td>
<td>71-60</td>
</tr>
<tr>
<td>71.2.5.5</td>
<td>Retrieve all Property Sets</td>
<td>71-61</td>
</tr>
<tr>
<td>71.2.5.6</td>
<td>Delete all Property Sets</td>
<td>71-62</td>
</tr>
<tr>
<td>71.2.6</td>
<td>Property CRUD</td>
<td>71-63</td>
</tr>
<tr>
<td>71.2.6.1</td>
<td>Create Property</td>
<td>71-63</td>
</tr>
<tr>
<td>71.2.6.2</td>
<td>Retrieve Property</td>
<td>71-65</td>
</tr>
<tr>
<td>71.2.6.3</td>
<td>Update Property</td>
<td>71-67</td>
</tr>
<tr>
<td>71.2.6.4</td>
<td>Delete Property</td>
<td>71-68</td>
</tr>
<tr>
<td>71.2.6.5</td>
<td>Delete all Properties</td>
<td>71-69</td>
</tr>
<tr>
<td>71.2.7</td>
<td>JSON Payload</td>
<td>71-70</td>
</tr>
</tbody>
</table>
71.2.8 Exceptions ........................................................................................................................ 71-70
71.2.9 Query Parameters ........................................................................................................... 71-70
71.2.10 Search Expressions ...................................................................................................... 71-71
71.2.10.1 Format .................................................................................................................... 71-71
71.2.10.2 Supported Operators ............................................................................................... 71-71
71.2.10.3 Supported Attribute Names .................................................................................... 71-71

72 Conductor API Reference

72.1 Using Java APIs ................................................................................................................ 72-1
72.2 Using the Conductor REST APIs ..................................................................................... 72-1
72.2.1 Using the Data Provider Management REST APIs .................................................. 72-2
72.2.1.1 Getting a Collection of Data Providers (GET) ....................................................... 72-2
72.2.1.2 Getting a Single Data Provider's Metadata (GET) ................................................. 72-6
72.2.1.3 Getting Information About a Data Provider Connection (GET) ...................... 72-17
72.2.1.4 Getting Information about an Executable Resource for a Data Provider and Connection (GET) 72-20
72.2.1.5 Getting Method Information for an Executable Resource for a Data Provider and Connection (GET) 72-30
72.2.1.6 Getting Entire Data Provider Information with Details Projection (GET) ...... 72-32
72.2.2 Using the Function Provider Management APIs ..................................................... 72-50
72.2.2.1 Getting a Collection of Function Providers (GET) .............................................. 72-50
72.2.2.2 Getting Metadata for a Single Function Provider (GET) ..................................... 72-76
72.2.3 Using the Namespace Management APIs ................................................................. 72-78
72.2.3.1 Getting a Collection of Namespaces (GET) .......................................................... 72-78
72.2.3.2 Retrieving a Single Namespace (GET) ................................................................ 72-82
72.2.3.3 Creating a Namespace (POST) ............................................................................. 72-84
72.2.3.4 Deleting a Namespace (DELETE) ........................................................................ 72-87
72.2.4 Using the Scenario Management APIs ..................................................................... 72-87
72.2.4.1 Getting a Collection of Scenarios (GET) ............................................................. 72-87
72.2.4.2 Retrieving a Single Scenario (GET) ..................................................................... 72-90
72.2.4.3 Creating a Scenario (POST) ................................................................................ 72-92
72.2.4.4 Updating a Scenario (PUT) ................................................................................ 72-95
72.2.4.5 Deleting a Scenario (DELETE) .......................................................................... 72-98
72.2.4.6 Execute a Scenario (POST) ................................................................................ 72-99
72.3 Calling Data Integration Client Services Using ELs ....................................................... 72-100
72.3.1 Calling Data Integration Client Services from JSF Pages ....................................... 72-100
72.3.2 Calling Data Integration Client Services from JSP Pages ..................................... 72-101
72.3.3 Data Integration Context Object Method Reference ............................................. 72-101
72.3.4 Enabling Single Sign-on ......................................................................................... 72-103

73 Cache Management for Personalization

73.1 Introduction to Cache Management for Personalization ............................................. 73-1
73.2 Locating and Updating Cache Configuration Files ..................................................... 73-1
73.2.1 Where are the Cache Configuration Files Located? ............................................. 73-2
73.2.2 Using the Cache Configuration Files ..................................................................... 73-2
73.2.2.1 Internal Cache Configuration File...................................................................... 73-2
73.2.2.2 Data Provider Cache File .................................................................................. 73-2
### Part XII  Completing Your WebCenter Portal Application

#### 74  Securing Your WebCenter Portal Framework Application

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>74.1  Introduction to Portal Framework Application Security</td>
<td>74-2</td>
</tr>
<tr>
<td>74.2  Creating an Application Role</td>
<td>74-2</td>
</tr>
<tr>
<td>74.3  Configuring ADF Security</td>
<td>74-3</td>
</tr>
<tr>
<td>74.3.1 Configuring ADF Security Settings</td>
<td>74-3</td>
</tr>
<tr>
<td>74.3.2 Automated Security Grants for Portal Framework Applications</td>
<td>74-8</td>
</tr>
<tr>
<td>74.4  Using the Role Manager Task Flow</td>
<td>74-16</td>
</tr>
<tr>
<td>74.5  Using the Enterprise Role Member Task Flows</td>
<td>74-17</td>
</tr>
<tr>
<td>74.6  Using the Page Hierarchy Security Editor</td>
<td>74-17</td>
</tr>
<tr>
<td>74.6.1 Introduction to Page Hierarchy Security</td>
<td>74-17</td>
</tr>
<tr>
<td>74.6.1.1 Page Hierarchy Model</td>
<td>74-18</td>
</tr>
<tr>
<td>74.6.1.2 Page-level and Page Hierarchy Security</td>
<td>74-19</td>
</tr>
<tr>
<td>74.6.1.3 Inherited and Delegated Security</td>
<td>74-19</td>
</tr>
<tr>
<td>74.6.2 Building a Page Hierarchy</td>
<td>74-20</td>
</tr>
<tr>
<td>74.6.2.1 Adding a Page Hierarchy to a Navigation</td>
<td>74-20</td>
</tr>
<tr>
<td>74.6.2.2 Adding Pages to a Page Hierarchy</td>
<td>74-20</td>
</tr>
<tr>
<td>74.6.2.3 Delegating Security</td>
<td>74-21</td>
</tr>
<tr>
<td>74.7  Creating Login Pages and a Login Component</td>
<td>74-22</td>
</tr>
<tr>
<td>74.8  Creating a Login Portlet</td>
<td>74-23</td>
</tr>
<tr>
<td>74.9  Adding Portlets to a Login Page</td>
<td>74-23</td>
</tr>
<tr>
<td>74.10 Creating a Self-Registration Page</td>
<td>74-24</td>
</tr>
<tr>
<td>74.10.1 Integrating the WebCenter Portal Self-Registration Task Flow</td>
<td>74-24</td>
</tr>
<tr>
<td>74.10.2 Building a Custom Self-Registration Page</td>
<td>74-25</td>
</tr>
<tr>
<td>74.10.3 Creating a Self-Registration Invitation Page</td>
<td>74-25</td>
</tr>
<tr>
<td>74.11 Creating a Reset Password Page</td>
<td>74-26</td>
</tr>
<tr>
<td>74.12 Configuring Basic Authentication for Testing Portlet Personalization</td>
<td>74-26</td>
</tr>
<tr>
<td>74.13 Working with External Applications</td>
<td>74-27</td>
</tr>
<tr>
<td>74.13.1 Using External Applications</td>
<td>74-28</td>
</tr>
<tr>
<td>74.13.1.1 Secured Service Connections</td>
<td>74-28</td>
</tr>
<tr>
<td>74.13.1.2 Automated Single Sign-On</td>
<td>74-30</td>
</tr>
<tr>
<td>74.13.2 Supplying User Credentials</td>
<td>74-31</td>
</tr>
<tr>
<td>74.13.3 Managing External Applications</td>
<td>74-32</td>
</tr>
<tr>
<td>74.13.3.1 Adding External Application Task Flows</td>
<td>74-32</td>
</tr>
</tbody>
</table>
74.13.3.2 Working with External Applications in Oracle JDeveloper ...................... 74-34
74.13.3.2.1 Registering an External Application in Oracle JDeveloper .................. 74-34
74.13.3.2.2 Editing External Application Registration Details in Oracle JDeveloper .......... 74-36
74.13.3.2.3 Deleting External Application Registration Details in Oracle JDeveloper ........... 74-36
74.13.3.3 Working with External Applications in Enterprise Manager ...................... 74-37
74.13.3.4 Working with External Applications Using WLST ........................................ 74-37
74.14 Registering Custom Certificates with the Keystore ........................................... 74-37
74.15 Overriding Inherited Security on Portlets and Customizable Components ............... 74-39
74.15.1 Portlets Security ............................................................................................... 74-41
74.15.1.1 Defining Security at the Actions Category Level ............................................. 74-41
74.15.1.2 Defining Security at the Actions Level ............................................................ 74-43
74.16 Identity Propagation Mechanisms ........................................................................ 74-45
74.17 Securing Identity Propagation Through WSRP Producers with WS-Security .......... 74-46
74.17.1 Identity Propagation Without WS-Security ........................................................ 74-46
74.17.2 Identity Propagation with WS-Security ............................................................. 74-47
74.17.3 Configuring Security for WSRP Portlets .......................................................... 74-49
74.18 Implementing PDK-Java Portlet Security ............................................................ 74-49
74.18.1 Assumptions ........................................................................................................ 74-50
74.18.2 Introduction to PDK-Java Portlet Security Features ........................................... 74-50
74.18.2.1 Identity Propagation ...................................................................................... 74-50
74.18.2.2 Authorization ............................................................................................... 74-50
74.18.2.3 Message-level Security .................................................................................. 74-50
74.18.3 Single Sign-On ................................................................................................. 74-51
74.18.3.1 External Application ...................................................................................... 74-51
74.18.3.2 No Application Authentication ....................................................................... 74-52
74.18.4 Portlet Security Managers .................................................................................. 74-52
74.18.4.1 Implementing Your Own Security Manager ................................................... 74-53
74.18.5 Message Authentication ...................................................................................... 74-53
74.18.6 User Input Escape ............................................................................................. 74-53
74.18.6.1 Default Container Encoding ......................................................................... 74-54
74.18.6.2 Escape Methods ........................................................................................... 74-54
74.19 Using WebCenter Portal Impersonation ELs and APIs .......................................... 74-54
74.20 Troubleshooting Security Issues ............................................................................ 74-55
74.20.1 Error Message Appears When Running a Page with a Content Repository Data Control Method Being Consumed 74-56

75 Building Multilanguage Portals
75.1 Guidelines for Building Multilanguage Portals ....................................................... 75-1
75.2 Adding Support for a New Language ...................................................................... 75-2
75.3 Language Support in ADF Faces Components ...................................................... 75-2
75.4 Using Resource Bundles to Support Multiple Languages ...................................... 75-2

Part XIII Appendixes

A WebCenter Portal Files
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-2
  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-3
  A.3.11 pageTemplate_globe.jspx .................................................. A-3
  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-7
  A.5.3 index.jsp ............................................................................. A-8
  A.5.4 portlet_name_modePage.jsp ................................................ A-8
  A.5.5 provider.xml ........................................................................ A-8
  A.5.6 web.xml ............................................................................... A-9
A.6 System Files ................................................................................................................. A-9
  A.6.1 adf-config.xml ................................................................. A-9
  A.6.2 DataBindings.cpx ............................................................... A-10
  A.6.3 faces-config.xml ............................................................... A-10
  A.6.4 page_name.jspx ............................................................... A-10
  A.6.5 PageDef.xml ................................................................. A-10
  A.6.6 trinidad-config.xml .......................................................... A-12
  A.6.7 trinidad-skins.xml ............................................................ A-13
  A.6.8 web.xml ............................................................................. A-13
  A.6.9 weblogic-application.xml .................................................. A-13
  A.6.10 mds Subdirectory ............................................................ A-13
  A.6.11 wsdl Subdirectory .......................................................... A-13
A.7 Files Related to Security .............................................................................................. A-13
  A.7.1 connections.xml ................................................................. A-14
  A.7.2 jazn-data.xml ................................................................. A-18
  A.7.3 cwallet.sso ........................................................................ A-20
  A.7.4 jps-config.xml ................................................................. A-20
  A.7.5 adf-config.xml ................................................................. A-20
A.8 Oracle JDeveloper Files .............................................................................................. A-21
B Composer Component Properties and Files

B.1 Composer Component Properties ....................................................... B-1
  B.1.1 Page Customizable Component .................................................. 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-12
  B.1.5 Show Detail Frame Component ............................................... B-17
  B.1.6 Custom Action ........................................................................ B-26
  B.1.7 Show Property ......................................................................... B-29

B.2 Composer-Specific Files and Configurations .................................. B-33
  B.2.1 pe_ext.xml ................................................................................. B-33
    B.2.1.1 addon-config ....................................................................... B-34
    B.2.1.2 property-panels ................................................................. B-35
    B.2.1.3 lov-config ........................................................................... B-36
    B.2.1.4 event-handlers ................................................................... B-37
    B.2.1.5 drop-handlers .................................................................... B-37
    B.2.1.6 filter-config ........................................................................ B-38
    B.2.1.7 elbuilder-config ................................................................. B-40
    B.2.1.8 selection-config ................................................................. B-40
    B.2.1.9 Sample pe_ext.xml File ....................................................... B-41
    B.2.2 adf-config.xml ....................................................................... B-42
      B.2.2.1 page-editor-config ............................................................ B-43
      B.2.2.2 addon-panels ................................................................. B-43
      B.2.2.3 sandbox-namespaces ...................................................... B-44
      B.2.2.4 session-options-factory ................................................. B-44
      B.2.2.5 enable-design-views ....................................................... B-44
      B.2.2.6 allow-el and protect-el .................................................... B-45
      B.2.2.7 rcv-config ........................................................................ B-45
      B.2.2.8 customizableComponentsSecurity .................................. B-46
      B.2.2.9 mds-config ...................................................................... B-47
      B.2.2.10 resource-string-editor ................................................... B-47
      B.2.2.11 enable-source-view ....................................................... B-48
      B.2.2.12 enable-zoom ................................................................... B-48
      B.2.2.13 persistent-change-manager .......................................... B-48
      B.2.2.14 taglib-config ................................................................. B-49
      B.2.2.15 security-config .............................................................. B-49
      B.2.2.16 Sample adf-config.xml File ............................................. B-50
      B.2.3 adf-settings.xml ................................................................. B-52
        B.2.3.1 custComps-config ......................................................... B-52
        B.2.4 web.xml .......................................................................... B-53
          B.2.4.1 CHANGE_PERSISTENCE Context Parameter .................. B-53
          B.2.4.2 WebCenterComposerFilter ........................................ B-54
  B.3 Composer Default Add-Ons and Property Panels .......................... B-54
    B.3.1 Default Add-Ons ................................................................... B-54
    B.3.2 Default Property Panels ....................................................... B-55
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
G Expression Language Expressions

G.1 Introduction to Expression Language (EL) Expressions ........................................ G-1
G.1.1 Introducing the Expression Builder ........................................................................ G-2
G.1.2 Introducing the Expression Editor in WebCenter Portal ........................................ G-3
G.2 Expression Language Example Use Case ................................................................ G-5
G.3 ELs Related to Applications ..................................................................................... G-6
G.4 ELs Related to Security ............................................................................................. G-6
G.4.1 Example: Restricting Access to Structure View in Composer Using EL Expressions ...... G-7
G.5 ELs Related to General Settings ................................................................................ G-8
G.6 ELs Related to Portal Resources .............................................................................. G-9
G.7 ELs Related to Navigation ....................................................................................... G-13
G.7.1 Example: Using EL Expressions for Navigation ................................................ G-19
G.8 ELs Related to Tools and Services .......................................................................... G-20
G.9 ELs Related to Documents ...................................................................................... G-22
G.10 ELs Related to People Connections ....................................................................... G-22
G.11 ELs Related to Personalization .............................................................................. G-25
G.12 ELs Related to Impersonation ................................................................................ G-27
G.13 EL Expressions Relevant to Composer ................................................................... G-28
G.13.1 Example: Using EL Expressions for Composer ................................................ G-29
G.14 EL Expressions Relevant Only to WebCenter Portal ............................................ G-29
G.14.1 Desupport of Freeform JPQL WHERE and SORT Clauses in Portal Queries ...... G-29
G.14.2 Seeded Expressions in the Expression Editor .................................................... G-32
G.14.2.1 Application Info Seeded ELs ............................................................................ G-33
G.14.2.2 Asset Info Seeded ELs .................................................................................... G-34
G.14.2.3 Page Info Seeded ELs .................................................................................... G-36
G.14.2.4 Page Parameter Seeded ELs .......................................................................... G-36
G.14.2.5 Portal Info Seeded ELs .................................................................................. G-37
G.14.2.6 Portal Page Info Seeded Paths ....................................................................... G-41
G.14.2.7 System Seeded ELs ....................................................................................... G-42
G.14.2.8 User Info Seeded ELs .................................................................................... G-43
G.14.2.9 WebCenter Events Seeded ELs ...................................................................... G-45
G.14.3 EL Expressions Relevant to WebCenter Portal Information ................................ G-45
G.14.4 EL Expressions Relevant to Assets ..................................................................... G-47
G.14.4.1 Example: Using EL Expressions for Assets ................................................ G-50
G.14.5 EL Expressions Relevant to Specific Portals ..................................................... G-51
G.14.5.1 Example: Using EL Expressions for Various Portals .................................... G-55
G.14.6 EL Expressions Relevant to Specific Pages ....................................................... G-57
G.14.7 EL Expressions Relevant to Portal Event Contexts ......................................... G-58
G.14.8 Utilitarian EL Expressions .............................................................................. G-60
H WebCenter Portal Analytics Database Schema

H.1 Analytics Database Schema Overview ................................................................. H-1
H.1.1 Analytics Database Model Overview ............................................................. H-1
H.1.2 Analytics Database Table Overview ............................................................. H-3
H.1.2.1 Sample SQL Query: Finding Table Names Associated with Specific Events.... H-7
H.2 Analytics Fact Table Descriptions ..................................................................... H-8
H.2.1 ASFACT_WC_PAGECRE_0 ............................................................................ H-8
H.2.2 ASFACT_WC_PAGEDEL_0 ............................................................................ H-8
H.2.3 ASFACT_WC_PAGEEDI_0 ............................................................................ H-9
H.2.4 ASFACT_WC_PAGETAG_0 .......................................................................... H-9
H.2.5 ASFACT_WC_PAGEVIE_0 ............................................................................ H-10
H.2.6 ASFACT_WC_DISCUSS_0 ............................................................................ H-10
H.2.7 ASFACT_WC_DISCUSS_1 ............................................................................ H-11
H.2.8 ASFACT_WC_DISCUSS_2 ............................................................................ H-11
H.2.9 ASFACT_WC_DISCUSS_3 ............................................................................ H-12
H.2.10 ASFACT_WC_DISCUSS_4 .......................................................................... H-12
H.2.11 ASFACT_WC_DISCUSS_5 .......................................................................... H-13
H.2.12 ASFACT_WC_DISCUSS_6 .......................................................................... H-13
H.2.13 ASFACT_WC_DISCUSS_7 .......................................................................... H-13
H.2.14 ASFACT_WC_DISCUSS_8 .......................................................................... H-14
H.2.15 ASFACT_WC_DISCUSS_9 .......................................................................... H-14
H.2.16 ASFACT_WC_DISCUSS_A .......................................................................... H-15
H.2.17 ASFACT_WC_DISCUSS_B .......................................................................... H-15
H.2.18 ASFACT_WC_DISCUSS_C .......................................................................... H-16
H.2.19 ASFACT_WC_DOCLIB_0 .......................................................................... H-16
H.2.20 ASFACT_WC_DOCLIB_1 .......................................................................... H-17
H.2.21 ASFACT_WC_DOCLIB_2 .......................................................................... H-18
H.2.22 ASFACT_WC_DOCLIB_3 .......................................................................... H-18
H.2.23 ASFACT_WC_DOCLIB_4 .......................................................................... H-19
H.2.24 ASFACT_WC_DOCLIB_5 .......................................................................... H-19
H.2.25 ASFACT_WC_DOCLIB_6 .......................................................................... H-20
H.2.26 ASFACT_WC_DOCLIB_7 .......................................................................... H-20
H.2.27 ASFACT_WC_EVENT_C_0 .......................................................................... H-21
H.2.28 ASFACT_WC_EVENT_D_0 .......................................................................... H-21
H.2.29 ASFACT_WC_EVENT_E_0 .......................................................................... H-22
H.2.30 ASFACT_WC_GROUPSP_0 .......................................................................... H-22
H.2.31 ASFACT_WC_GROUPSP_1 .......................................................................... H-23
H.2.32 ASFACT_WC_GROUPSP_2 .......................................................................... H-23
H.2.33 ASFACT_WC_GROUPSP_3 .......................................................................... H-24
H.2.34 ASFACT_WC_LIST_CR_0 .......................................................................... H-24
H.2.35 ASFACT_WC_LIST_DE_0 .......................................................................... H-25
H.2.36 ASFACT_WC_LIST_ED_0 .......................................................................... H-25
H.2.37 ASFACT_WC_LOGINS_0 .......................................................................... H-26
H.2.38 ASFACT_WC_PEOPLEC_0 .......................................................................... H-26
I WebCenter Portal Accessibility Features

I.1 Overview of WebCenter Accessibility Features ................................................................. I-1
I.1.1 Generating Accessible HTML ...................................................................................... I-1
I.1.2 Accessibility Features at Runtime ............................................................................... I-2
I.1.3 Accessibility Considerations for Portlets ................................................................. I-2
Preface

This guide explains how to build portal applications and custom components using WebCenter Portal Framework and JDeveloper, 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 build, deploy, and manage a Portal Framework application.
- How to secure, administer, monitor, and maintain a Portal 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 Portal Framework application.
- How to deliver personalized content to a Portal Framework application.
- How to plan, build, deploy, and manage portlets for a Portal 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 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 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 that have purchased support 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 11g Release 1 (11.1.1.9.0) documentation set or on Oracle Technology Network (OTN) at http://www.oracle.com/technology/index.html.

- Administering Oracle WebCenter Portal
- Building Portals with Oracle WebCenter Portal
- Using Oracle WebCenter Portal
- Error Messages Reference
- Fusion Developer’s Guide for Oracle Application Development Framework

Conventions
The following text conventions are used in this document:

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>boldface</strong></td>
<td>Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.</td>
</tr>
<tr>
<td><em>italic</em></td>
<td>Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.</td>
</tr>
<tr>
<td>monospace</td>
<td>Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.</td>
</tr>
</tbody>
</table>
What's New?

The following topics introduce the new and changed features in Oracle WebCenter Portal 11g Release 1, and provide pointers to additional information:

- System Requirements and Specifications
- New and Changed Features for 11g Release 1 (11.1.1.9.0)
- New and Changed Features for 11g Release 1 (11.1.1.8.3)
- New and Changed Features for 11g Release 1 (11.1.1.8.0)

System Requirements and Specifications

For system and platform-specific information for Oracle Fusion Middleware products for 11g Release 1, refer to Oracle Fusion Middleware System Requirements and Specifications. That guide describes how to find out what is certified and how to verify the requirements of the certification. Your system administrator performs these tasks prior to installing Oracle WebCenter Portal.

New and Changed Features for 11g Release 1 (11.1.1.9.0)

WebCenter Portal 11g Release 1 (11.1.1.9.0) includes the following new and changed features:

- Support for running WebCenter Portal on cloud technologies:
  - Portal on-premise: WebCenter Portal deployment in a customer data center.
  - Oracle Private Cloud: provisioning and running WebCenter Portal on Oracle MWaaS; that is, using Oracle EM Cloud Control.

- Support for migration of the WebCenter Portal and Portal Framework application content from the Folders_g folder service to the FrameworkFolders folder service. See the "Migrating Folders_g to FrameworkFolders” appendix in Administering Oracle WebCenter Portal.

- Support for upgrading a pre-11.1.1.8.0 portal to the current release. See the "Upgrading a Pre-11.1.1.8.0 Portal” section in Building Portals with Oracle WebCenter Portal.

- Addition of the childCreation property for the Show Detail Frame component to specify when the children (contents) of the Show Detail Frame component are created, thus reducing processing time. See childCreation in Table B–7, "Attributes of a Show Detail Frame Component".
New and Changed Features for 11g Release 1 (11.1.1.8.3)

WebCenter Portal 11g Release 1 (11.1.1.8.3) included the following new and changed features:

- Support for four new page templates that provide efficiency and performance enhancements over the existing page templates. See "New Page Templates."
- Support for the FrameworkFolders component that provides a scalable, high-performing folder service from Oracle WebCenter Content as an alternative to Folders_g. See "FrameworkFolders Support."

New Page Templates

Four new page templates provide efficiency and performance improvements over the existing page templates. The files included are:

- Four page templates: Skyros Side Navigation v2, Skyros Side Navigation (Stretch) v2, Skyros Top Navigation v2, and Skyros Top Navigation (Stretch) v2
- One skin: Skyros v2
  
  This is the preferred skin for the new page templates. Do not attempt to use the new Skyros v2 skin with existing page templates. Likewise, do not attempt to use the new page templates with the existing skins.

- Two task flows: Portal Side Navigation and Portal Top Navigation
  
  These task flows implement the navigation in the page templates, either in a side pane or as tabs along the top of a portal. The styling of the navigation in these task flows relies on the CSS in the skin file available in this patch. If you want to use these task flows in a page template that uses a different skin, be aware that the navigation may not look as expected due to CSS mismatches in the skin. However, if you copy and paste the navigation sections from the new Skyros v2 skin source code into your skin source code, you can achieve the expected results.

With WebCenter Portal 11g Release 1 (11.1.1.9.0), these files are pre-installed, and you will see them on the Shared Assets page, which is available if you have the permissions of the Administrator or Application Specialist role:

<table>
<thead>
<tr>
<th>Asset Type</th>
<th>File Names</th>
</tr>
</thead>
<tbody>
<tr>
<td>Page Templates</td>
<td>pageTemplate_Skyros_Side_Navigation_Stretch_v2.ear</td>
</tr>
<tr>
<td></td>
<td>pageTemplate_Skyros_Side_Navigation_v2.ear</td>
</tr>
<tr>
<td></td>
<td>pageTemplate_Skyros_Top_Navigation_Stretch_v2.ear</td>
</tr>
<tr>
<td></td>
<td>pageTemplate_Skyros_Top_Navigation_v2.ear</td>
</tr>
<tr>
<td>Skins</td>
<td>skin_Skyros_v2_Skin.ear</td>
</tr>
<tr>
<td>Task Flows</td>
<td>taskFlow_Portal_Side_Navigation.ear</td>
</tr>
<tr>
<td></td>
<td>taskFlow_Portal_Top_Navigation.ear</td>
</tr>
</tbody>
</table>

Perform the following steps to use the new page templates:

1. To set a new default page template for pages in the Home portal and all new portals (when the portal’s template does not specify that a particular page
template must be used), see the "Choosing a Default Page Template" section in *Administering Oracle WebCenter Portal*.

2. To change the page template used by an individual portal, see the "Changing the Page Template for a Portal" section in *Building Portals with Oracle WebCenter Portal*.

3. To change the preferred skin used by a page template, see the "Setting the Preferred Skin for a Page Template" section in *Building Portals with Oracle WebCenter Portal*.

For more information, see the "Working with Page Templates" chapter in *Building Portals with Oracle WebCenter Portal* and Chapter 11, "Developing Page Templates."

**FrameworkFolders Support**

Previously, Oracle WebCenter Portal supported only Folders_g. WebCenter Portal 11g Release 1 (11.1.1.8.3) enables new installations of Oracle WebCenter Portal to be integrated with FrameworkFolders. Existing installations of Oracle WebCenter Portal patched to 11g Release 1 (11.1.1.8.3) must continue to use Folders_g.

The FrameworkFolders component provides a scalable hierarchical folder interface for accessing and organizing repository content, and is the recommended enterprise folder service for Content Server.

**New and Changed Features for 11g Release 1 (11.1.1.8.0)**

WebCenter Portal 11g Release 1 (11.1.1.8.0) included the following new and changed features:

- Terminology changes:
  - Prior Releases | 11g Release 1 (11.1.1.8.0)  
  | WebCenter Portal: Spaces | WebCenter Portal  
  | space | portal  
  | space template | portal template  
  | resource | asset

- End-User Experience
  - Updated profile user interface that includes improved organization of profile information, click to edit, and clear profile photo functionality. See the "Managing Your Profile" chapter in *Using Oracle WebCenter Portal*. Additional documentation for the rich user profile is referenced under *Portal Builder, Administration*, and *Development Environment*.
  - Improved search experience (supported with Oracle SES 11.2.2.2) that includes faceted search and document thumbnails. See the "Searching for Information" chapter in *Using Oracle WebCenter Portal*.

- Portal Builder
  - Simplified portal creation that includes in-place page creation. See the "Creating and Building a New Portal" chapter in *Building Portals with Oracle WebCenter Portal*.
  - Redesigned portal edit and administration user interface (Portal Builder) that consolidates tasks into fewer steps. See the "Editing a Portal" and
"Administering a Portal” chapters in *Building Portals with Oracle WebCenter Portal*.

- Simplified page creation and editing: Web (for editing) and Data (for managing) views, inline resource catalog (with support for component drag-and-drop onto a page), and Select view. See the "Working with Portal Pages” chapter in *Building Portals with Oracle WebCenter Portal*.

- Automatic update of portal navigation as new pages are created. See the "Creating a Page or Subpage in an Existing Portal" section in *Building Portals with Oracle WebCenter Portal*.

- "Lazy provisioning” of tools—WebCenter Portal configures the back-end server at first use of a tool rather than at portal creation to speed the successful creation of a new portal. See the "About Creating a New Portal” section in *Building Portals with Oracle WebCenter Portal*.

- Hierarchical page support (subpages). See the "Creating Pages or Subpages in a Portal” section in *Building Portals with Oracle WebCenter Portal*.

- Updated profile user interface that includes improved organization of profile information, click to edit, and clear profile photo functionality; new component properties for improved control of people connections and activity graph components. See the "Adding Activity Graphs and Recommendations to a Portal,” "Adding Connections to a Portal,” and "Adding Profiles to aPortal” chapters in *Building Portals with Oracle WebCenter Portal*. Additional documentation for the rich user profile is referenced under End-User Experience, Administration, and Development Environment.

- Device Settings that control how your portal pages render on different devices, such as smartphones, tablets, and desktop browsers. Page variants can be created to target and optimally render a portal on specific groups of devices like iOS phones, iOS tablets, and others. See the "Administering Device Settings in a Portal” section, the "Managing Device Groups for a Portal" chapter, and the "Creating a Page Variant for a Device Group” section in *Building Portals with Oracle WebCenter Portal*. Additional documentation for mobile support is referenced under Administration and Development Environment.

- Responsive Content Presenter templates that provide an example of how you can use Content Presenter and CSS3 media queries to produce a responsive layout that adjusts to the width of the browser (for example, on smartphones, tablets, and desktop browsers). See the "Using Responsive Templates” section in *Building Portals with Oracle WebCenter Portal*. Additional documentation for mobile support is referenced under Development Environment.

- Development Environment

- Updated profile user interface that includes improved organization of profile information, click to edit, and clear profile photo functionality; new component properties for improved control of people connections and activity graph components. See Chapter 37, "Introducing the People Connections Service,” Chapter 39, "People Connections Task Flow Binding Parameters,” and Chapter 46, "Integrating the Activity Graph.” Additional documentation for the rich user profile is referenced under End-User Experience, Portal Builder, and Administration.

- Developers can use Expression Language (EL) to retrieve information about Device Settings. Device Settings control how your portal pages render on different devices including smartphones, tablets, and desktop browsers. See
---

Section G.15, "EL Expressions Related to Device Settings." Additional documentation for mobile support is referenced under Portal Builder and Administration.

- Responsive Content Presenter templates that provide an example of how you can use Content Presenter and CSS3 media queries to produce a responsive layout that adjusts to the width of the browser (for example, on phones, tablets, or personal computers). See Section 27.3.4, "Using Responsive Templates" and Section 27.3.5, "Extending Responsive Templates." Additional documentation for mobile support is referenced under Portal Builder.

- Simplified custom shared library development and deployment. WebCenter Portal provides a new JDeveloper template that enables you to build custom components, such as task flows, data controls, and managed beans and deploy them in shared libraries directly to the WebCenter Portal server. See Chapter 55, "Developing Components for WebCenter Portal Using JDeveloper."

- Administration
  
  - Simplified WebCenter Portal administration that includes a power user oriented experience with familiar concepts for legacy WebCenter Portal customers. See the "Managing Portals in Portal Builder Administration" part in Administering Oracle WebCenter Portal.

  - New profile configuration settings that include properties to specify whether to show the new or legacy profile user interface and to specify profile synchronization settings. See the "Managing People Connections" chapter in Administering Oracle WebCenter Portal. Additional documentation for the rich user profile is referenced under End-User Experience, Portal Builder, and Development Environment.

  - Device Settings that control how your portal pages render on different devices, such as smart phones, tablets, and desktop browsers. Page variants can be created to target and optimally render a portal on specific groups of devices like iOS phones, iOS tablets, and others. See the "Deploying Devices and Device Groups" section, the "Creating a Page Variant of a System Page for Device Groups" section, and the "Administering Device Settings" chapter in Administering Oracle WebCenter Portal. Additional documentation for mobile support is referenced under Portal Builder and Development Environment.

  - Impersonation, which allows a privileged user to impersonate another user for the purposes of verifying the other user’s experience in WebCenter Portal and troubleshooting unexpected results. See the "Managing Impersonation" chapter in Administering Oracle WebCenter Portal.

  - Improved portal lifecycle tools that enable export/import and backup/recovery of one or more portals with minimal downtime. See the "Deploying Portals, Templates, Assets, and Extensions" and "Managing WebCenter Portal Backup, Recovery, and Cloning" chapters in Administering Oracle WebCenter Portal.

  - Integrated Oracle WebCenter Portal’s Pagelet Producer user interface within WebCenter Portal’s administrative user interface to make system administrators aware of the existence of Pagelet Producer pagelets and to allow them to make these pagelets available to end users. Integrating the UIs also provides Pagelet Producer developers to easily navigate from WebCenter Portal where they see the pagelets to the Pagelet Producer Admin UI so they can create new or edit existing pagelets. See the "Managing the Pagelet Producer" chapter in Administering Oracle WebCenter Portal.

---
- New page performance analyzer that shows you how long individual components take to display on a portal page, as well as the overall time taken to display a page. This new tool is useful to developers who are performing first level performance analysis, customers who build their own pages, and any user who customizes pages in WebCenter Portal. See the "How to Identify Slow Page Components" section in *Administering Oracle WebCenter Portal*.

- Restructured documentation library according to personas and their roles in WebCenter Portal:
  - *Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper* (this guide) covers information needed by a developer who primarily works with JDeveloper to provide support for both portals and WebCenter Portal Framework applications.
  - *Using Oracle WebCenter Portal* covers information needed by a knowledge worker who typically uses WebCenter Portal to contribute and review content, participate in social interactions, and leverage the Home portal to manage her own documents and profile.
  - *Building Portals with Oracle WebCenter Portal* covers information needed by an application specialist who works in Portal Builder to create and administer portals, their structure (hierarchy of pages, navigation, security), and their content (components on a page, layout, behavior, and so on).
  - *Administering Oracle WebCenter Portal* covers information needed by a system administrator who fields requests from IT employees and business users to set up new machines; clone or back up existing applications systems and databases; install patches, packages, and applications; and perform other administration-related tasks.

For more information, see "Who's Who."
Who's Who

The WebCenter Portal documentation is organized so that the tasks in a particular guide address a specific user persona. Each persona is associated with a set of skills required to work with WebCenter Portal, from basic to advanced. For example, this guide is aimed at the Developer persona.

This preface introduces you to the WebCenter Portal personas and describes the ways in which they might interact with WebCenter Portal. Each persona is assigned a default role provided out-of-the-box with WebCenter Portal. The default roles are given a unique set of permissions appropriate for the work that each persona will typically do. Note that you can modify these default roles or configure new roles to meet the unique needs of your organization.

The people who interact with WebCenter Portal typically work together as a team that is comprised of the following personas:

- Knowledge Worker
- Application Specialist
- Web Developer
- Developer
- System Administrator

Knowledge Worker

Karen is a knowledge worker who typically uses WebCenter Portal to contribute and review content, participate in social interactions, and leverage the Home portal to manage her own documents and profile.

At the application level, Karen has permissions such as those granted to the default Authenticated-User role, which may be customized for the specific needs of the organization. At the portal level, the portal Moderator will likely assign Karen the Viewer or Participant role, or a custom role that offers a similar set of permissions.
Knowledge Worker Tasks in WebCenter Portal

Tasks that are typical of a knowledge worker like Karen include:

- Connecting to and collaborating with other WebCenter Portal users by sharing information, files, and links; and by interacting through instant messaging, mail, message boards, discussions, wikis, and blogs
- Uploading, sharing, and managing documents stored in WebCenter Content
- Joining a team or project portal
- Keeping up with changes in WebCenter Portal by receiving notifications when content is updated, exploring recommendations from other users, viewing the activities of the portals she is a member of and users she’s connected to, viewing announcements, taking polls, and monitoring WebCenter Portal RSS feeds
- Staying organized through the use of favorites, notes, calendars, lists, links to portal objects, and tags
- Viewing and responding to worklist items

As Karen becomes more familiar with the functionality available in WebCenter Portal, she may begin to perform more advanced tasks, such as creating portals. As a more advanced knowledge worker, her role may evolve to overlap with application specialist tasks.

Information targeted for knowledge workers like Karen is in Using Oracle WebCenter Portal. Advanced tasks that overlap with those of an application specialist are covered in Building Portals with Oracle WebCenter Portal.

Application Specialist

Ari is an application specialist who works in Portal Builder to create and administer portals, their structure (hierarchy of pages, navigation, security), and their content (components on a page, layout, behavior, and so on). In a typical project, Ari coordinates the efforts of Karen (knowledge worker), Wendy (web developer), and Dave (developer).

At the application level, Ari has permissions such as those granted to the default Application Specialist role, which may be customized for the specific needs of the organization. In a portal that Ari creates, he performs actions available to the Moderator role to manage the portal.

For more information about roles and permissions, see the "About Roles and Permissions for a Portal" section in Building Portals with Oracle WebCenter Portal.

Application Specialist Tasks in WebCenter Portal

Tasks that are typical of an application specialist like Ari include:
Planning and creating new portals
Editing and administering the portals he owns
Creating and building portal pages using the page editor (Composer) and the resource catalog to add and configure page components
Creating and managing portal assets, tools, and services
Managing shared assets and portal templates across all portals

Information targeted for application specialists like Ari is in *Building Portals with Oracle WebCenter Portal*. To work with his personal view of the Home portal, Ari will also refer to *Using Oracle WebCenter Portal*.

**Web Developer**

Wendy is a *web developer* who focuses on delivering a consistent, branded look and feel to all portals. Wendy provides graphics designs and HTML markup from which Ari (application specialist in Portal Builder) or Dave (developer in JDeveloper) can create content or page style templates, skins, and so on. Once these assets are created, Ari can leverage them to create portal pages. Wendy typically does not interact with WebCenter Portal directly.

**Web Developer Tasks in WebCenter Portal**
Tasks that are typical of a web developer like Wendy include:

- Developing a corporate portal look and feel
- Designing new portal page templates

Information targeted for web developers like Wendy is in the "Creating a Look and Feel for Portals" chapter in *Building Portals with Oracle WebCenter Portal*.

**Developer**

Dave is a *developer* who provides support for both portals and WebCenter Portal Framework applications:

- Portals (Portal Builder)
Dave is primarily responsible for developing components (such as task flows, page templates, and content templates), which are published and leveraged by Ari (the application specialist). Dave primarily works with JDeveloper and leverages the WebCenter Portal Extension/WebCenter Portal Service Extension projects.

### Framework Applications

Dave primarily works with JDeveloper to develop WebCenter Portal Framework applications. Once he has developed the application, he can package it as an EAR file and deploy it on the application server. In a typical environment, Dave would have JDeveloper configured with a SCM system and be working within a team with automated build and deploy processes.

### Developer Tasks

Tasks that are typical of a developer like Dave include:

- Building and maintaining WebCenter Portal Framework applications
- Developing custom assets, like page templates and navigation components for portals in WebCenter Portal
- Developing Java portlets
- Developing and deploying task flows, managed beans, and other custom components
- Developing custom personalization components
- Maintaining the source control system
- Maintaining a build system

Information targeted for developers like Dave is in *Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper*.

### System Administrator

Syed is a system administrator who fields requests from IT employees and business users to set up new machines; clone or back up existing applications systems and databases; install patches, packages, and applications; and perform other administration-related tasks. As the system administrator, Syed works with other tools such as Fusion Middleware Control and command line tools. He leverages Enterprise Manager to configure portal settings, and also configures integrations such as WebCenter Content and other Fusion Middleware products and Oracle applications.

In WebCenter Portal’s Portal Builder, he has permissions such as those granted to the default Administrator role, which provides exclusive access to administer and set global options for all portals (including the Home portal).

For more information about application level roles and permissions, see the "About Application Roles and Permissions" section in *Administering Oracle WebCenter Portal*. 
System Administrator Tasks

Tasks that are typical of a system administrator like Syed include:

- Uses Portal Builder administration to administer all portals (including import and export of portals) and security site-wide
- Uses Portal Builder administration to manage site-wide system pages, business role pages, and personal pages
- Uses Portal Framework application administration console to manage application-wide preferences, manage users and roles, manage assets, configure the content repository, create polls, register producers and external applications
- Leads security, taxonomy, metadata, workflow, governance
- Uses the management console for administrative functions
- Executes command line utilities for administrative functions
- Installs and configures production versions of developers' efforts
- Performs patching of the production versions and the operating system
- Creates clones and backups of the production versions
- Performs restores of production versions
- Monitors the operating system for issues with the production version
- Deploys and redeploys applications

Information targeted for system administrators like Syed is in Administering Oracle WebCenter Portal and WebLogic Scripting Tool Command Reference.
Part I contains the following chapters:

- Chapter 1, "Introduction to Oracle WebCenter Portal for Developers"
- Chapter 2, "Setting Up Your Development Environment"
- Chapter 3, "Working Productively in Teams"
- Chapter 4, "Preparing Your Application for WebCenter Portal Tools and Services"
This guide is the primary resource for developers using JDeveloper to create Portal Framework applications, portlets, and custom portal components.

This chapter includes the following topics:

- Section 1.1, "What Is the Purpose of this Guide?"
- Section 1.2, "What Is My Role as a WebCenter Portal Developer?"

1.1 What Is the Purpose of this Guide?

This developer's guide discusses developing Portal Framework applications, portlets, and custom portal components using Oracle JDeveloper. Many of the tasks described in this guide involve activities that require Java, CSS, Application Development Framework (ADF), Expression Language (EL), and related experience.

Major activities described in this guide include:

Setting Up Your Environment
See chapters on setting up your environment and your team environment in Part I, "Getting Started."

Building Portals with WebCenter Portal Framework
A WebCenter Portal Framework application is a standard ADF web application that includes portal features, like navigation, pages, page templates, content, and more. The chapters in Part II, "Building WebCenter Portal Framework Applications" explain how to create Framework applications and develop assets like skins, templates, and navigation components.

Several parts of this guide contain chapters devoted to working with portal pages, content, look and feel, collaborative features, and so on.

- Part III, "Customizing Your Application and Extending Customization Options" discusses creating, customizing, and editing portal pages and customizing task flows used in your portal.
- Part IV, "Integrating and Publishing Content" explains how to integrate content into your portal by connecting to content repositories and working with content display components, like Content Presenter. The part also includes chapters on integrating wikis and blogs into the portal.
- Part IV, "Integrating and Publishing Content" explains how to integrate content into your portal by connecting to content repositories and working with content
display components, like Content Presenter. The part also includes chapters on integrating wikis and blogs into the portal.

- Part V, "Enabling Communication and Collaboration" discusses integrating features like discussions, mail, and presence into the portal.
- Part VI, "Working with People Connections" explains how to integrate people connections features into the portal. People connections provide 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, Connections, Feedback, Message Board, Profile, and Publisher.
- Part VII, "Helping Users Find Content" discusses how to add links, tags, search, and activity tracking to the portal.
- Part VIII, "Helping Users Keep Track" discusses adding features like analytics, lists, notifications, and related features to the portal.

Developing Custom Components and Extending WebCenter Portal

WebCenter Portal Framework applications can be integrated with other applications, like Siebel, Oracle E-Business Suite, and PeopleSoft. You can use JDeveloper to create custom components, like task flows and managed beans, that can be deployed to the Portal Server and used in Portal Builder. These topics are covered in the chapters of Part IX, "Extending WebCenter Portal and Portal Framework Applications."

Working with Portlets and Pagelets

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 several different portlets can be placed on a single page, users benefit from a single-source experience even though, in reality, the content may be derived from multiple sources. See the chapters of Part X, "Working with Portlets and Pagelets," for detailed information.

Pagelets are sub-components of a Web page accessed through Pagelet Producer that can be injected into any proxied application. Any application on a Pagelet Producer resource that returns markup can be registered as a pagelet, which can then be displayed in a portal or Framework application, or any other Web application. See the chapters of Part X, "Working with Portlets and Pagelets," for detailed information.

Delivering Personalized Content

Personalization delivers targeted content based on both user and application context for Portal Framework applications, and WebCenter Portal pages using Expression Language (EL) expressions. See Part XI, "Delivering Personalized Content."

Securing Your Portal

The security chapter describes security mechanisms and features provided in a Framework application, and how you can use Oracle ADF Security to handle authentication and authorization. See Part XII, "Completing Your WebCenter Portal Application." This part also includes guidelines for creating multi-language portals.

1.2 What Is My Role as a WebCenter Portal Developer?

In some cases, a portal requires custom components that do not exist out-of-the-box or cannot easily be created with the browser-based Portal Builder. For example, design requirements might call for page templates, page styles, and skins that are branded
and organized in a precise way. In this case, you may need to use ADF Faces and CSS to achieve the desired look and feel, and JDeveloper is well-suited to such development. In another use case, custom components may be required that allow for interaction between portal elements and data (for example, using ADF task flows or portlets with event handling). Many such use cases exist.

As a WebCenter Portal developer, you may be asked to create and configure:

- Custom portal extensions like task flows, data controls, and backing beans
- Portal infrastructure components like page templates and page styles
- Personalization components like custom data providers, function providers, and property set locators
- JSR 286 portlets (also called "Java portlets")
- WebCenter Portal Framework applications
- WebCenter Portlet Producer applications

Large-scale portal development projects, like intranets and extranets, require the effort of a multi-role team. The team may include user interface experts, application specialists, QA engineers, Java developers, and others who participate in the portal's overall development and maintenance. See also Section 3, "Working Productively in Teams."
What Is My Role as a WebCenter Portal Developer?
Setting Up Your Development Environment

This chapter provides guidance and tips to help you get started using JDeveloper for WebCenter Portal development activities.

This chapter includes the following topics:

- Section 2.1, "Introduction"
- Section 2.2, "General Setup Tasks"
- Section 2.3, "Setup Tasks Specific to WebCenter Portal Framework Applications"
- Section 2.4, "Setup Tasks Specific to WebCenter Portlet Producer Applications"
- Section 2.5, "Setting Up JDeveloper for Personalization"

2.1 Introduction

This chapter distinguishes between tasks that are general and tasks that are application-specific. General tasks apply no matter what kind of application you are developing. For instance, installing JDeveloper and the WebCenter Portal Extension are general setup tasks. Other tasks described in this chapter apply only if you are developing specific kinds of applications, like WebCenter Portal Server Extension or WebCenter Portal Framework applications.

2.2 General Setup Tasks

This section describes setup tasks that do not depend on what kind of application you are developing.

- Section 2.2.1, "Installing Oracle JDeveloper"
- Section 2.2.2, "Installing the WebCenter Portal Extension for JDeveloper"
- Section 2.2.3, "Setting the User Home Directory Environment Variable"
- Section 2.2.4, "Managing the Integrated WebLogic Server"
- Section 2.2.5, "Configuring WebCenter Back-End Services"
- Section 2.2.6, "Creating Application Resource Connections"
- Section 2.2.7, "Preparing for Team Development and Source Control"
2.2.1 Installing Oracle JDeveloper

Oracle JDeveloper provides an integrated development environment (IDE) for developing portals and custom portal components. For information on obtaining and installing Oracle JDeveloper, see the Oracle JDeveloper page on OTN at:


Once JDeveloper is installed, Oracle recommends that you increase the maximum PermGen size as follows:

1. Open the following file in a text editor:
   
   \[JDEV_HOME/jdev/bin/jdev.conf\]

   where \[JDEV_HOME\] is the location where JDeveloper is installed on your machine.

2. Set \[AddVMOption -XX:MaxPermSize=512\].

3. Restart JDeveloper.

2.2.2 Installing the WebCenter Portal Extension for JDeveloper

The WebCenter Portal extension is an add-in that provides JDeveloper with the complete set of WebCenter Portal capabilities and features. 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.


7. Check Oracle Fusion Middleware Products and Official Oracle Extensions and Updates and click Next.

8. From the generated list, search for the WebCenter Framework and Services Design Time extension, select it, and then click Finish.

9. When prompted, restart JDeveloper.

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

http://www.oracle.com/technetwork/middleware/webcenter/portal/overview/index.html

See also the Installation Guide for Oracle WebCenter Portal.
2.2.3 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 the “Setting the User Home Directory” section in the Installation Guide for Oracle JDeveloper.

2.2.4 Managing the Integrated WebLogic Server

This section discusses the Integrated WebLogic Server used by JDeveloper.

- Section 2.2.4.1, "What Is the Integrated WebLogic Server?"
- Section 2.2.4.2, "Starting and Stopping Integrated WebLogic Server"
- Section 2.2.4.3, "Configuring the JVM for the Integrated WebLogic Server"
- Section 2.2.4.4, "The WebCenter Preconfigured Server Readme File"

2.2.4.1 What Is the Integrated WebLogic Server?

The Integrated WebLogic Server is a runtime service that references an instance of Oracle WebLogic Server and is bundled with JDeveloper. The Integrated WebLogic Server allows developers to run, test, and debug WebCenter Portal Framework applications from within JDeveloper.

2.2.4.2 Starting and Stopping Integrated WebLogic Server

For detailed information managing the Integrated WebLogic Server, see the topic "Deploying to Integrated WebLogic Server" in the JDeveloper Online Help. Other options, such as running in debug mode and logging into the WebLogic Server Administration Console are also covered in that section.

2.2.4.3 Configuring the JVM for the Integrated WebLogic Server

Although it is not required, you have the option of changing the default Java Virtual Machine (JVM) settings the Integrated WLS in the setDomainEnv.sh. This file is located in 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 Portal 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.

2.2.4.4 The WebCenter Preconfigured Server Readme File

The WebCenter Preconfigured Server Read Me file contains valuable information about how to use Integrated WLS. In addition, the file contains links to preconfigured portlet producer test pages.
2.2.5 Configuring WebCenter Back-End Services

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 2–1 lists the schemas that are used with WebCenter Portal 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 schema, you have the option of installing with a SQL script, as explained below.

**Table 2–1  Database Schema Summary**

<table>
<thead>
<tr>
<th>Schema</th>
<th>Description</th>
<th>Installation Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>WebCenter</td>
<td>To use Tag, Links, Lists, Polls, and People Connections, you must have WebCenter Portal's schema installed in your database.</td>
<td>• RCU (preferred)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SQL script (On Oracle databases only. See below for details.)</td>
</tr>
<tr>
<td>Portlets</td>
<td>For storing customizations in portlet producers.</td>
<td>RCU only</td>
</tr>
<tr>
<td>Activities</td>
<td>For Activity Graph and Analytics products.</td>
<td>RCU only</td>
</tr>
<tr>
<td>Discussions/WebCenter Portal Discussions Crawler</td>
<td>Used as the back end for discussions and announcements.</td>
<td>RCU only</td>
</tr>
</tbody>
</table>

2.2.5.1 Installing Schemas with RCU

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

2.2.5.2 Installing WebCenter Portal Schema with a SQL Script

To use Tag, Links, and People Connections, you must have the WebCenter Portal schema installed in your database.

To install the WebCenter Portal schema:

1. From the Tools menu, select Database, and then select SQL Worksheet.
2. In the Select Connection Dialog you can either select an existing connection or create a new connection. If you already have a database connection for an...
administrator user to your database you can select that connection. If you don't have a connection then follow the next steps to create a new connection:

1. In the Select Connection dialog, click the plus icon to create a connection.
2. Use the Create Database Connection dialog to create a connection to the database where you will create the schema. This connection is not associated to the WebCenter Portal Framework Application.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Recommended Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create Connection In</td>
<td>IDE Connections</td>
</tr>
<tr>
<td>Connection Name</td>
<td>Use any name of your choice, for example: my_connection.</td>
</tr>
<tr>
<td>Connection Type</td>
<td>Oracle JDBC</td>
</tr>
<tr>
<td>Username</td>
<td>SYS</td>
</tr>
<tr>
<td>Password</td>
<td>Enter the SYS user password.</td>
</tr>
<tr>
<td>Role</td>
<td>SYSDBA</td>
</tr>
<tr>
<td>Driver</td>
<td>thin</td>
</tr>
<tr>
<td>Host Name</td>
<td>Enter the server name where the database is running. For example: myserver.example.com.</td>
</tr>
<tr>
<td>SID</td>
<td>Enter the database SID, for example: mydb101.</td>
</tr>
<tr>
<td>JDBC Port</td>
<td>Enter the database port, for example: 1521.</td>
</tr>
</tbody>
</table>

3. Click **Test Connection** to verify that the connection works.
4. Click **OK** to close the Create Database Connection dialog.

### 2.2.5.3 Installing WebCenter Portal Schema with a SQL Script

To use Tag, Links, and People Connections, you must have the WebCenter Portal schema installed in your database.

To install the WebCenter Portal schema:

1. From the **Tools** menu, select **Database**, and then select **SQL Worksheet**.
2. In the Select Connection dialog, click the pencil icon to edit the connection.

**Note:** For information on creating database connections, see Section 2.2.6, "Creating Application Resource Connections."

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:

```sql
@@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.

### 2.2.6 Creating Application Resource Connections

Connections allow applications 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 discusses connections and describes the different ways to access the wizards for creating new connections. See also Section 2.2.6.3, "Where Can I Learn More About Connections?"

#### 2.2.6.1 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 features, 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 chapters on tools and services for WebCenter Portal. See

- **Under IDE Connections in the Resource Palette**

  Connections created here can be reused across 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.

#### 2.2.6.2 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.

### 2.2.6.3 Where Can I Learn More About Connections?

For information about creating and consuming connections, see:

- Chapter 25, "Configuring Content Repository Connections"
- Chapter 63, "Consuming Portlets"
- Part IV, "Integrating and Publishing Content"
- Chapter 7, "Deploying and Testing Your Portal Framework Application"

### 2.2.7 Preparing for Team Development and Source Control

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 portals.

**Tip:** Typically, one member of the development team creates a new portal application in JDeveloper and checks it in to the source control repository. A team member 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 3, "Working Productively in Teams."

### 2.3 Setup Tasks Specific to WebCenter Portal Framework Applications

This section describes tasks that you only need to perform if you are creating WebCenter Portal Framework application.

- **Section 2.3.1, "Preparing for Iterative Development in a Portal Framework Application"
- **Section 2.3.2, "Additional Configurations"

### 2.3.1 Preparing for Iterative Development in a Portal Framework Application

This section discusses the iterative development feature for WebCenter Portal Framework applications.
Enabling iterative development lets you make changes to your portal application while it is running on the Integrated WebLogic Server and immediately see the effect of those changes by refreshing the browser. For more information on iterative development, see Section 8.7, "Understanding Iterative Development."

**Note:** The iterative development option disables many performance optimization features, which can cause your application to run more slowly.

- Section 2.3.1.1, "Enabling Iterative Development"
- Section 2.3.1.2, "Turning Off Iterative Development"

### 2.3.1.1 Enabling Iterative Development

This section explains how to enable iterative development, if it has been disabled. When you enable this feature, you are turning off all caching, which can cause your application to run more slowly.

**Note:** The iterative development feature is enabled by default when you create a new Portal 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 2-1.

**Figure 2-1  Enabling Iterative Development**

5. Under the **Run** node in the left pane of the dialog, select **MDS**.
6. 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.

7. Click **OK**.

### 2.3.1.2 Turning Off Iterative Development

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

There are other several reasons why you might consider turning off iterative development, such as:

- Because turning on iterative development disables all caching, you might turn off this feature if you have a very large portal and the lack of caching is causing performance problems.
- When this feature is turned off, there is no difference between a portal instance that is running on the Integrated WebLogic Server and an instance that is deployed to a managed server.

To turn off iterative development:

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**.

### 2.3.2 Additional Configurations

Additional configurations that may be required for WebCenter Portal Framework applications are covered in [Section 2.2.5, "Configuring WebCenter Back-End Services"] and [Section 2.2.6, "Creating Application Resource Connections."]

### 2.4 Setup Tasks Specific to WebCenter Portlet Producer Applications

This section describes tasks that you only need to perform if you are creating WebCenter Portlet Producer application. For more information about WebCenter Portlet Producer applications, see [Chapter 57, "Introduction to Portlets."]

#### 2.4.1 What You May Need to Know About Preconfigured Portlet Producers

WebCenter Portal provides a variety of ready-to-use portlets that you can add to your portal pages. This section provides a brief description of the preconfigured producers and the portlets they provide. It contains the following subsections:

- [Section 2.4.1.1, "Deploying the Preconfigured Portlet Producers"]
- [Section 2.4.1.2, "Registering Deployed Portlet Producers"]
- [Section 2.4.1.3, "Accessing OmniPortlet and Web Clipping"]
- [Section 2.4.1.4, "The WSRP Sample Portlet Producers and Portlets"]
- [Section 2.4.1.5, "The PDK-Java Sample Portlet Producer and Portlets"]
2.4.1.1 Deploying the Preconfigured Portlet Producers

The preconfigured portlet producers are not deployed by default – you must deploy them manually, as explained in this section.

**Note:** These producer applications consume memory and require CPU time during deployment. This contributes to the WebLogic Server startup time and base memory footprint. For these reasons, it makes sense to only deploy the applications you need. For more information on tuning the deployment of internal applications, see the "On-demand Deployment of Internal Applications" section in Deploying Applications to Oracle WebLogic Server.

1. If it is not running, start the Integrated WebLogic Server. The server must be running for this feature to be enabled.
2. From the Run menu, select WebCenter Portal Deployments.
3. In the WebCenter Portal Deployments dialog box, select the producer you wish to deploy and click OK. The dialog is shown in Figure 2–2. Each of the preconfigured producers is described in more detail in the following sections.

[Figure 2–2 WebCenter Portal Deployments Dialog]

2.4.1.2 Registering Deployed Portlet Producers

After a portlet producer is deployed, you can register it. Portlets from registered producers can then be selected from Oracle JDeveloper’s Application Resources panel.

For information about registering portlet producers, see Chapter 63, "Consuming Portlets." For information about adding portlets to pages, see Section 63.5, "Adding Portlets to a Page."

2.4.1.3 Accessing 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.

To access the OmniPortlet and Web Clipping portlet producers:

2. From the Help menu, select WebCenter Preconfigured Server Readme.
3. In the Readme file, go to the heading **Accessing the PortalTools Portlet Producers Test Pages**, and click the **PortalTools Welcome Page** link.

This opens the PortalTools Welcome page.

---

**Note:** The Portal Tools Portlet Producers component must be deployed before the PortalTools Welcome page can be accessed through the WebCenter Preconfigured Server Readme page. For details, see Section 2.4.1.1, "Deploying the Preconfigured Portlet Producers."

---

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 Chapter 63, "Consuming Portlets."

---

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 portal page. (For information about adding portlets to pages, see Section 63.5, "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 65, "Creating Content-Based Portlets with Web Clipping."

---

**Note:** Instead of using the Web Clipping portlet, consider using a clipper pagelet using Oracle WebCenter Portal's Pagelet Producer. For more information, see the "Managing the Pagelet Producer" chapter in the *Administering Oracle WebCenter Portal*.

---

- **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, and web page. 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 64, "Creating Portlets with OmniPortlet."

---

- **The Sample Portlet Producer** includes portlets built with OmniPortlet that are for demonstration purposes only. The portlet samples include a scrolling RSS portlet and a simple RSS portlet. Note that the sample producer is for demonstration and should not be used to create real-use portlet instances.
2.4.1.4 The WSRP Sample Portlet Producers and Portlets

The Integrated WLS includes sample WSRP portlet producers and portlets you can use with your application.

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:

2. From the Help menu, select WebCenter Portal Preconfigured Server Readme.
3. In the Readme file, go to the heading Accessing the WSRP Portlet Producers Test Pages, and click the link for the WSRP Tools Portlet Producers or Sample Portlet Producer.

Note: The portlet producers are not deployed by default. You must deploy a producer application before it can be accessed through the WebCenter Preconfigured Server Readme page. For details, see Section 2.4.1.1, "Deploying the Preconfigured 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 Chapter 63, "Consuming Portlets."

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 Portal Framework application pages.

2.4.1.5 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:

2. From the Help menu, select WebCenter Preconfigured Server Readme.

3. In the Readme file, go to the heading Accessing the PDK-Java Portlet Producers Test Pages, and copy the link location for the PDK-Java Sample Producer or PDK-Java Struts Sample Producer link.

Note: These producers must be deployed before they can be accessed through the WebCenter Preconfigured Server Readme page. For details, see Section 2.4.1.1, "Deploying the Preconfigured 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 Chapter 63, "Consuming Portlets."

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 Portal Framework application pages.

2.4.1.6 The Portlet Bridge Tester Consumer
The tester is used to test portletized page or task flow in JDeveloper. The tester is deployed by default. For more information, see Section 58.4, "Testing a Portletized Page or Task Flow in JDeveloper."

2.4.1.7 WCPS Services
WebCenter Personalization Services is deployed by default, and is unrelated to portlet producers. For information about WebCenter Personalization, see Chapter 66, "Personalizing Oracle WebCenter Portal Applications."

2.5 Setting Up JDeveloper for Personalization
Personalization provides a dynamically derived user experience for your Portal Framework 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. For more information, see Chapter 66, "Personalizing Oracle WebCenter Portal Applications."

Before you can develop and use personalization components you need to add certain components to your portal project. JDeveloper makes this process easy to do:

1. Be sure the WebCenter Portal Extension is installed into your JDeveloper environment. The WebCenter Portal Extension includes the required data integration component JAR files. See Section 2.2.2, "Installing the WebCenter Portal Extension for JDeveloper."

2. Create an application, such as a WebCenter Portal Framework application.
3. Add the WebCenter Personalization technology scope to your project. This technology scope adds the JDeveloper-specific tooling features, like the scenario editor, to the project.

1. Right-click your project and select Project Properties.
2. Select Technology Scope in the Project Properties dialog.
3. Select WebCenter Personalization in the list of available scopes and move it to the Selected column.
4. Click OK.

After you complete these steps, you are ready to develop and test personalization components in JDeveloper. For more information, see Chapter 66, "Personalizing Oracle WebCenter Portal Applications."
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.

This chapter includes the following topics:

- Section 3.1, "Using Source Control for WebCenter Portal Development"
- Section 3.2, "Using Oracle Team Productivity Center"
- Section 3.3, "Deciding Which Files to Check in to Source Control"
- Section 3.4, "Implementing Common Requirements One Time"
- Section 3.5, "Portlet Producer Considerations"

3.1 Using Source Control for WebCenter Portal Development

You can use source control in WebCenter Portal 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 to 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. For details, see the "Proxy Settings and JDeveloper" topic in the Oracle JDeveloper online help.

For more complete and extensive information about setting up and using SubVersion and other source control systems with JDeveloper, see the "Developing in Teams" topic in the Oracle JDeveloper online help.

3.2 Using 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 portals 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.

### 3.3 Deciding Which Files to Check in to Source Control

An important aspect of working with any source control system is understanding which files to check in and which actions might affect those files. In general, as you your team members begin creating new projects, you will see which files are created, and then can determine what you wish to check into source control. It’s difficult to summarize all the possible interactions with files you may encounter; however, this section offers a few general tips.

Table 3–1 lists a few of the components with which developers typically interact, and provides links to more information about those components and their related files.

<table>
<thead>
<tr>
<th>Component</th>
<th>For More Information...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pages</td>
<td>For information about page metadata files, see the &quot;Introduction to the ADF Metadata Files&quot; section in the Fusion Developer’s Guide for Oracle Application Development Framework.</td>
</tr>
<tr>
<td>Portlets</td>
<td>For information about portlet and producer metadata files, see Appendix A, &quot;WebCenter Portal Files.&quot; See also, Section 59, &quot;Building Standards-Based Java Portlets Using JSR 286.&quot;</td>
</tr>
<tr>
<td>Task Flows</td>
<td>For detailed information about ADF task flows, see the &quot;Getting Started with ADF Task Flows&quot; section in Fusion Developer’s Guide for Oracle Application Development Framework.</td>
</tr>
<tr>
<td>Data Controls</td>
<td>See the Fusion Developer’s Guide for Oracle Application Development Framework for detailed information about data controls.</td>
</tr>
</tbody>
</table>

For information on other features and related files you may wish to review when planning your source control strategy, see Appendix A, "WebCenter Portal Files." The appendix lists and describes Portal Framework files, system files, JDeveloper files, and others.

### 3.4 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 a portal. 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.

3.5 Portlet Producer Considerations

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

- Section 3.5.1, "Portlet Producer Connections"
- Section 3.5.2, "Combining Portlets from Different Portlet Producers"

3.5.1 Portlet Producer Connections

When building an EAR file for your project, connections are loaded for the entire portal rather than individual projects. For example, suppose you have a portal 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 portals and projects.

3.5.2 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 portal.

- 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 a portal 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.
Preparing Your Application for WebCenter Portal Tools and Services

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

This chapter includes the following topics:

- Section 4.1, "Understanding WebCenter Portal Tools and Services"
- Section 4.2, "Preparing Your Framework Application for Tools and Services"
- Section 4.3, "Customizing How Services Render Resources"
- Section 4.4, "Configuring General Settings"

4.1 Understanding WebCenter Portal Tools and Services

Oracle WebCenter Portal tools and services enrich portals and Web sites with social computing services, personal productivity services, online awareness and communications, content integration, and Web analytics.

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

**Note:** WebCenter Portal tools and services expose their functionality through task flows that you can add to the pages of your application. If these task flows do not quite meet your requirements, you can customize them to change their look and feel or functionality. For example, you can add text, change labels, remove regions or components, or show additional attributes. For more information, see Chapter 23, "Customizing WebCenter Portal Tools and Services Task Flows."

This section includes the following subsections:

- Section 4.1.1, "Understanding WebCenter Portal Horizontal Tools and Services"
- Section 4.1.2, "Understanding WebCenter Portal APIs"
- Section 4.1.3, "Using WebCenter Portal Data Controls"

4.1.1 Understanding WebCenter Portal Horizontal Tools and Services

Some tools and services in WebCenter Portal are considered horizontal in that they interact with other tools and services across an application.
Table 4–1 lists integration points in WebCenter Portal.

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

Note: Search (listed in the table) uses WebCenter Portal search adapters to search each available component. 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

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

4.1.2 Understanding WebCenter Portal APIs

This section describes the APIs available for WebCenter Portal tools and services.

Note the following considerations when multiple customization options are available:

- Use REST APIs to build a custom client for accessing WebCenter Portal; for example, an iPhone application that interacts with WebCenter Portal.
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 WebCenter Portal.

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 4–2 lists tools and services with available APIs.

<table>
<thead>
<tr>
<th>WebCenter Portal Tool or Service</th>
<th>Data Controls</th>
<th>Java APIs</th>
<th>REST APIs</th>
<th>Back End APIs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activity Graph</td>
<td>Section 46.3.1, &quot;Using the Recommendation Data Control&quot;</td>
<td>Java API Reference for Oracle WebCenter Portal</td>
<td>Section 46.3.2, &quot;Using the Activity Graph REST APIs&quot;</td>
<td></td>
</tr>
<tr>
<td>Analytics</td>
<td>Section 47.3.1, &quot;Using SQL Data Controls&quot;</td>
<td>Java API Reference for Oracle WebCenter Portal</td>
<td></td>
<td>Web Services with Oracle WebCenter Portal’s Discussions Server (requires configuration, because WebCenter Portal is set up with custom authentication)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>See the Jive Forums documentation on the Oracle Fusion Middleware documentation library (in the WebCenter Portal product area)</td>
</tr>
<tr>
<td>Announcements</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Comments &amp; Likes</td>
<td></td>
<td>Java API Reference for Oracle WebCenter Portal</td>
<td>Section 42.1, &quot;Activity Stream REST API&quot;</td>
<td>Web Services with Oracle WebCenter Portal’s Discussions Server (requires configuration, because WebCenter Portal is set up with custom authentication)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>See the Jive Forums documentation on the Oracle Fusion Middleware documentation library (in the WebCenter Portal product area)</td>
</tr>
<tr>
<td>Discussions</td>
<td></td>
<td></td>
<td>Section 33.3.8, &quot;Using the Discussions REST API&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Documents (with wiki and blogs)</td>
<td>Chapter 26, &quot;Working with Content Data Controls&quot;</td>
<td>Java API Reference for Oracle WebCenter Portal</td>
<td>Chapter 42, &quot;Using the People Connections REST APIs&quot;</td>
<td>See Java API Reference for Oracle WebCenter Content Remote Intradoc Client (RIDC)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Services Reference for Oracle WebCenter Content</td>
</tr>
</tbody>
</table>
### Table 4–2  (Cont.) Supported Technologies for Tools and Services

<table>
<thead>
<tr>
<th>WebCenter Portal Tool or Service</th>
<th>Data Controls</th>
<th>Java APIs</th>
<th>REST APIs</th>
<th>Back End APIs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Events</td>
<td></td>
<td></td>
<td>Section 48.3, &quot;Using the Events REST API&quot;</td>
<td>Web Services with Microsoft Exchange</td>
</tr>
<tr>
<td>General Settings</td>
<td></td>
<td>Java API Reference for Oracle WebCenter Portal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Links</td>
<td></td>
<td></td>
<td>Section 43.3.1, &quot;Using the Links REST API&quot;</td>
<td></td>
</tr>
<tr>
<td>Lists</td>
<td></td>
<td></td>
<td>Section 49.4, &quot;Using the Lists REST API&quot;</td>
<td></td>
</tr>
<tr>
<td>Notifications</td>
<td>Section 50.3.2, &quot;Using Notifications Data Controls&quot;</td>
<td>Java API Reference for Oracle WebCenter Portal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pagelet Producer</td>
<td></td>
<td></td>
<td>Section 62.7.2.2, &quot;Accessing Pagelets Using REST&quot;</td>
<td></td>
</tr>
<tr>
<td>People Connections</td>
<td>Section 40.1, &quot;Using People Connections Data Controls&quot;</td>
<td>Java API Reference for Oracle WebCenter Portal</td>
<td>Chapter 42, &quot;Using the People Connections REST APIs&quot; (for Profile and Activity Stream)</td>
<td></td>
</tr>
<tr>
<td>Personalization</td>
<td></td>
<td>Java API Reference for Oracle WebCenter Portal (for developing custom Providers and Locators)</td>
<td>Chapter 71, &quot;Property Service REST APIs&quot; Chapter 72, &quot;Conductor API Reference&quot;</td>
<td></td>
</tr>
<tr>
<td>Polls</td>
<td>Section 36.3.4, &quot;Using the Polls Data Controls&quot;</td>
<td>Java API Reference for Oracle WebCenter Portal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Resource Action Handler Framework</td>
<td></td>
<td>Java API Reference for Oracle WebCenter Portal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Search</td>
<td>Section 45.3.5, &quot;Using the Search Data Control&quot;</td>
<td>Java API Reference for Oracle WebCenter Portal</td>
<td>Section 45.3.4, &quot;Using the Search REST APIs&quot; Note: Only Oracle SES search supports the search data control.</td>
<td>Web Services and Java APIs with Oracle Secure Enterprise Search</td>
</tr>
<tr>
<td>Tags</td>
<td></td>
<td>Java API Reference for Oracle WebCenter Portal</td>
<td>Section 44.3.2, &quot;Using the Tags REST APIs&quot;</td>
<td></td>
</tr>
</tbody>
</table>
4.1.3 Using WebCenter Portal Data Controls

Several tools and 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 resource catalog for a page template in a portal in a WebCenter Portal application.

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

To use a built-in data control:

1. In JDeveloper, create a WebCenter Portal Framework application with any connections required.

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.

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

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 tools and services.

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

Note:  For ADF Web service data controls, connection details are not propagated from stage to production. For the data control to work, you must import the connections at the target instance.
4. Drag and drop methods and properties from the data control to an 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.

### 4.2 Preparing Your Framework Application for Tools and Services

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

- Section 4.2.1, "How to Prepare Your Application to Consume Tools and Services"
- Section 4.2.2, "Setting Up a Database Connection"
- Section 4.2.3, "Setting Up an External Application Connection"

#### 4.2.1 How to Prepare Your Application to Consume Tools and Services

You can configure any application to include a tool or 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 task flow or component, the necessary libraries are automatically added to the project.

Depending upon the tool or service you plan to consume, your application must meet certain prerequisites. For example, if the tool 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 4.2.1.1, "Implementing Security for Tools and Services"
Preparing Your Framework Application for Tools and Services

4.2.1 Implementing Security for Tools and Services

Some tools must know the identity of the user (for example, search needs the user's identity for saving searches). For these tools and 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 74.3, "Configuring ADF Security" for details on how to implement a basic security solution for your Framework application
- Chapter 74, "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 4–2). The Resource Type drop down 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 4–2  ADF Security Policies in the jazn-data.xml File**
4.2.1.2 Setting Up SSL-Protected Connections

If you are setting up a back-end connection in JDeveloper, 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 connections:

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

   The Preferences dialog displays (Figure 4–3).

**Figure 4–3 Preferences Dialog**

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

### 4.2.2 Setting Up a Database Connection

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

**Table 4–3** lists the tools and services that require a database connection.

**Table 4–3  Associated Data Source**

<table>
<thead>
<tr>
<th>Tool or Service</th>
<th>Database Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activity Graph</td>
<td>ACTIVITIES schema</td>
</tr>
<tr>
<td>Activity Stream</td>
<td>WEBCENTER schema</td>
</tr>
<tr>
<td>Analytics</td>
<td>ACTIVITIES schema</td>
</tr>
<tr>
<td>Blogs</td>
<td>WEBCENTER schema</td>
</tr>
<tr>
<td>Comments</td>
<td>WEBCENTER schema</td>
</tr>
<tr>
<td>Documents (including wikis and blogs) that want to</td>
<td>WEBCENTER schema</td>
</tr>
<tr>
<td>include Comments and Activity Stream</td>
<td></td>
</tr>
<tr>
<td>Links</td>
<td>WEBCENTER schema</td>
</tr>
<tr>
<td>Lists</td>
<td>WEBCENTER schema</td>
</tr>
</tbody>
</table>
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

---

**Table 4–3 (Cont.) Associated Data Source**

<table>
<thead>
<tr>
<th>Tool or Service</th>
<th>Database Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle Portal Adapter</td>
<td>PORTAL schema (name of the Oracle Portal schema) For more information, see the section “Setting Up the Environment to Use the Federated Portal Adapter” in the Administrator’s Guide for Oracle Portal.</td>
</tr>
<tr>
<td>People Connections</td>
<td>WEBCENTER schema</td>
</tr>
<tr>
<td>Polls</td>
<td>WEBCENTER schema</td>
</tr>
<tr>
<td>Tags</td>
<td>WEBCENTER schema</td>
</tr>
<tr>
<td>Announcements</td>
<td>DISCUSSIONS schema</td>
</tr>
<tr>
<td>Discussions</td>
<td>DISCUSSIONS schema</td>
</tr>
</tbody>
</table>
Preparing Your Framework Application for Tools and Services

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 the Fusion Developer’s Guide for Oracle Application Development Framework.

**Figure 4–5 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 4–6*).
Preparing Your Framework Application for Tools and Services

Figure 4–6  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.

4.2.2.1 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 4–7).

Figure 4–7  Associating to Data Source for Existing Connections

4.2.3 Setting Up an External Application Connection

When a tool or 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 tools and services permit the use of an external application to connect with it and define authentication for it:

- Documents
- Events
Customizing How Services Render Resources

Many services used with WebCenter Portal provide access to resources. For example, search results can include links to documents, tags, activities, and other types of resources. This section explains how to specify the way such resources are rendered in the portal.

- Section 4.3.1, "Overview of Resource Rendering Options"
- Section 4.3.2, "Rendering Resources in the Same Browser Window"
- Section 4.3.3, "Rendering Resources in an ADF Inline Popup"
- Section 4.3.4, "Rendering Resources in a New Browser Window"
- Section 4.3.5, "Setting Rendering Behavior with the resourceActionBehavior Tag"
- Section 4.3.6, "Updating adf-config.xml in Deployed Framework Applications"

4.3 Customizing How Services Render Resources

Figure 4–8 shows sample results for a search. There are numerous clickable elements in the results, including links to user profiles, documents, tags, blogs, and wiki pages.

Figure 4–8  Sample Search Results

See Also:

- Chapter 28, "Integrating Documents," Chapter 34, "Integrating Instant Messaging and Presence," Chapter 35, "Integrating Mail," and Chapter 52, "Integrating RSS" for more information about setting up an external application
- Section 74.13, "Working with External Applications" for more information about external applications in general
For Portal Framework applications, most services render their resources directly in the browser window.

The possible resource rendering behaviors available within WebCenter Portal are:

- **Full-screen rendering in the same browser window.** This option is the default for most services. The resource is opened in a viewer task flow by navigating from the browser to the resource using the same browser window. See Section 4.3.2, "Rendering Resources in the Same Browser Window."

- **ADF inline popup.** This option opens the resource in an inline popup (a dialog-like window that pops up and is contained within the current browser window.) See Section 4.3.3, "Rendering Resources in an ADF Inline Popup."

- **New browser window.** This option renders the resource in a new browser window. See Section 4.3.4, "Rendering Resources in a New Browser Window."

### 4.3.2 Rendering Resources in the Same Browser Window

By default, most resources are rendered full-screen in the same browser window.

---

**Note:** In some special cases, resources are intentionally rendered by default in an inline popup dialog, as explained in Section 4.3.5, "Setting Rendering Behavior with the resourceActionBehavior Tag."

---

To explicitly set the default rendering to the browser window, you can do the following:

1. In the Application Navigator, open the Application Resources list.
2. Open the Descriptors folder and then open the ADF META-INF folder.
3. Open the adf-config.xml file in the editor and add a `<resource-handler>` element, as shown in Example 4–1. In this example, the DefaultResourceActionHandler (the default class to which rendering is delegated) specifies the desired behavior. If you were to remove the `<resource-handler>` element from adf-config.xml entirely, the behavior would be exactly the same. The behavior specified in Example 4–1 applies to all services.

#### Example 4–1  Specifying In-Browser Rendering

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

### 4.3.3 Rendering Resources in an ADF Inline Popup

To change the default rendering from the browser window to an ADF inline popup dialog, do the following:

1. In the Application Navigator, open the Application Resources list.
2. Open the Descriptors folder and then open the ADF META-INF folder.
3. Open the adf-config.xml file in the editor and add a `<resource-handler>` element, as shown in Example 4–2. In this example, the
DefaultResourceActionHandler (the default class to which rendering is delegated) is overridden by the display-as-popup=true attribute.

Example 4–2  Specifying ADF Inline Popup for Rendering

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

If the user clicks the **Maximize** button in an inline popup, then the currently specified default handler is invoked. For DefaultResourceActionHandler, the resource appears full-screen in the browser. If PopUpResourceActionHandler is specified in `adf-config.xml`, then it displays in a new browser window. See also Section 4.3.4, "Rendering Resources in a New Browser Window."

4.3.4 Rendering Resources in a New Browser Window

To change the default rendering from within the browser window to a new browser window, do the following:

1. In the Application Navigator, open the **Application Resources** list.
2. Open the Descriptors folder and then open the **ADF META-INF** folder.
3. Open the `adf-config.xml` file in the editor and add the `<resource-handler>` element, as shown in Example 4–3. In this case, rendering is delegated to the PopUpResourceActionHandler class.

Example 4–3  Specifying a Separate Browser for Rendering

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

4.3.5 Setting Rendering Behavior with the resourceActionBehavior Tag

Rendering behavior that is set in `adf-config.xml` can be overridden on the resource link itself using the `resourceActionBehavior` tag and setting the `useResourcePopup` attribute to values of always, never or appDefined.

WebCenter Portal uses this technique for targeted cases 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.

If `useResourcePopup` is not set, or if it is set to appDefined on the resource link, then the `display-as-popup` attribute in `adf-config.xml` is honored.

**Note:** In cases where WebCenter Portal services override the `<resource-handler>` by design, the `useResourcePopup` tag is set to always in their `resourceActionBehavior` tags. You can change this by customizing the task flow itself.
4.3.6 Updating adf-config.xml in Deployed Framework Applications

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 WebLogic Scripting Tool Command Reference.

4.4 Configuring General Settings

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

General Settings 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 4–9 shows the Insert EL Expression dialog with General Settings.
Table 4–4 describes the preference values for General Settings.

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

<table>
<thead>
<tr>
<th>General Settings Preference Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>formattedCurrentDate</td>
<td>Displays the current date in the user's selected locale.</td>
</tr>
<tr>
<td>formattedCurrentDateTime</td>
<td>Displays the current date and time in the user's selected locale.</td>
</tr>
<tr>
<td>formattedCurrentTime</td>
<td>Displays the current time in the user's selected locale.</td>
</tr>
<tr>
<td>preferredAccessibilityMode</td>
<td>Preferred accessibility mode (default, inaccessible, or screenReader) for use in the accessibility-mode setting in trinidad-config.xml.</td>
</tr>
<tr>
<td>preferredDateStyle</td>
<td>Java date style to be used in dateStyle attributes when displaying dates and times.</td>
</tr>
<tr>
<td>preferredTimeStyle</td>
<td>Java time style to be used in timeStyle attributes when displaying dates and times.</td>
</tr>
<tr>
<td>preferredDateFormat</td>
<td>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.</td>
</tr>
<tr>
<td>preferredTimePattern</td>
<td>Java time pattern to be used when displaying times, and not dates. This takes precedence over preferredTimeStyle.</td>
</tr>
<tr>
<td>preferredDateTimePattern</td>
<td>Java date/time pattern to be used when displaying dates and times. This takes precedence over preferredDateStyle and preferredTimeStyle.</td>
</tr>
<tr>
<td>userTimeZone</td>
<td>Java time zone to be used in timeZone attributes when displaying dates and times.</td>
</tr>
<tr>
<td>preferredSkinName</td>
<td>Preferred skin name to use with the skin-family setting in trinidad-config.xml.</td>
</tr>
</tbody>
</table>
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 4–4  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 4–5  Code for Displaying Specified Date and Time**

<af:outputText value="#{row.date_time_value}">
  <af:convertDateTime type="both" 
  dateStyle="#{generalSettings.preferred_date_style}"
  timeStyle="#{generalSettings.preferred_time_style}"
  timeZone="#{generalSettings.user_time_zone}"
  locale="#{facesContext.externalContext.requestLocale}" />
  pattern="#{generalSettings.preferred_datetime_pattern}"  
</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 4–6  Setting the Accessibility Mode in trinidad-config.xml**

<accessibility-mode>#{generalSettings.preferred_accessibility_mode}</accessibility-mode>

### 4.4.1 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 4–4.

Any tool or service in WebCenter Portal that displays a date and time uses the settings configured in General Settings. For information on using the General Settings 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: Java API Reference for Oracle WebCenter Portal

### 4.4.2 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 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. If the General Settings locale preference does not exist, then it reverts to the non-filter default behavior.
Part II

Building WebCenter Portal Framework Applications

This part contains the following chapters:

- Chapter 5, "Understanding WebCenter Portal Framework Applications"
- Chapter 6, "Creating WebCenter Portal Framework Applications"
- Chapter 7, "Deploying and Testing Your Portal Framework Application"
- Chapter 8, "Understanding the WebCenter Portal Framework Application Life Cycle"
- Chapter 9, "Introduction to Portal Resource Management"
- Chapter 10, "Developing a Navigation Model"
- Chapter 11, "Developing Page Templates"
- Chapter 12, "Developing Page Styles and Task Flow Styles"
- Chapter 13, "Developing Skins"
- Chapter 14, "Developing Resource Catalogs"
- Chapter 15, "Creating Pages and Adding Resources"
This chapter introduces the architecture, components, and features of a WebCenter Portal Framework application.

**Note:** Oracle recommends creating new portals using Portal Builder, rather than Portal Framework. While custom Portal Framework applications may provide a greater degree of flexibility when developing your portal solution, such flexibility typically forfeits product supportability and the ability to upgrade to new WebCenter Portal releases. Using Portal Builder avoids these constraints without sacrificing complexity or scalability. Building portals using Portal Builder is covered in *Building Portals with Oracle WebCenter Portal*.

This chapter includes the following topics:

- Section 5.1, "What Is a WebCenter Portal Framework Application?"
- Section 5.2, "Creating a WebCenter Portal Framework Application"
- Section 5.3, "Exploring the Features of a WebCenter Portal Framework Application"
- Section 5.4, "How are WebCenter Portal Framework Application Files Organized?"
- Section 5.5, "What is the Portal Life Cycle?"
- Section 5.6, "Developing Your Portal’s Look and Feel"
- Section 5.7, "Changing Default Portal Preferences"
- Section 5.8, "Using Iterative and Round-Trip Development Techniques"
- Section 5.9, "Running and Testing a Portal"
- Section 5.10, "WebCenter Portal Framework Applications at Runtime"
- Section 5.11, "Changing the Default Home Page and Login/Logout Target Pages"
- Section 5.12, "Customizing Session Timeouts"
- Section 5.13, "Basic Portal Development Tasks"
5.1 What Is a WebCenter Portal Framework Application?

A WebCenter Portal Framework application is a standard ADF web application that includes portal features, like navigation, pages, page templates, content, and more. This chapter discusses these features in detail.

5.2 Creating a WebCenter Portal Framework Application

**Note:** Oracle recommends creating new portals using Portal Builder, rather than Portal Framework. While custom Portal Framework applications may provide a greater degree of flexibility when developing your portal solution, such flexibility typically forfeits product supportability and the ability to upgrade to new WebCenter Portal releases. Using Portal Builder avoids these constraints without sacrificing complexity or scalability. Building portals using Portal Builder is covered in *Building Portals with Oracle WebCenter Portal*.

JDeveloper makes it easy to create a new WebCenter Portal Framework application. A wizard guides you through the details and creates a workspace with the required project structures and files. You develop, deploy, and test your portal from within this workspace.

The basic steps for creating a framework application and the resulting project structures are described in Chapter 6, “Creating WebCenter Portal Framework Applications.”

5.3 Exploring the Features of a WebCenter Portal Framework Application

Oracle WebCenter Portal Framework adds a number of portal-specific features to an ADF web application. These features are included by default when you create a new application using the WebCenter Portal Framework Application template.

This section discusses some of the main portal features you’ll need to know about.

- **Section 5.3.1, “Summary of Features”**
- **Section 5.3.2, “Understanding Pages, Page Templates, and the Portal Page Hierarchy”**
- **Section 5.3.3, “Securing WebCenter Portal Framework Pages”**
- **Section 5.3.4, “Understanding the Navigation Model and the Navigation Registry”**
- **Section 5.3.5, “Understanding Resource Catalogs and the Catalog Registry”**
- **Section 5.4.4, “Showing Hidden Files in the Application Navigator”**

5.3.1 Summary of Features

The basic features provided in a Portal Framework application include:

- **Page hierarchies** – A 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 5.3.2, “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. See Section 5.3.4, "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. For information on the Resource Manager, see the "Managing Assets for a Portal Framework Application" section in the Administering Oracle WebCenter Portal. See also Chapter 10, "Developing a Navigation Model."

■ **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 JDeveloper and browser-based 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 74, "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 15, "Creating Pages and Adding Resources."

■ **Page templates** – 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 Section 5.6, "Developing Your Portal’s Look and Feel." See also the “Using Page Templates” section in the 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. See Chapter 14, "Developing Resource Catalogs."

■ **Catalog Registries** – A catalog registry defines the set of items that are available for inclusion in your resource catalog. See Chapter 14, "Developing 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 13, "Developing Skins."

■ **Portal Preferences** – You can change default preferences for a portal, including default navigation model, page template, and other components. See Section 5.7, "Changing Default Portal Preferences."

JDeveloper 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, browser-based tools are provided to allow end users and administrators to perform some of these functions. See also Section 5.10.2, "Browser-Based WebCenter Portal Administration Console."

5.3.2 **Understanding Pages, Page Templates, and the Portal Page Hierarchy**

This section introduces three related pieces of an Oracle WebCenter Portal Framework application: 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 Portal 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 5–1.

**Note:** For important information on the organization of files in Portal Framework applications, see Section 5.4, "How are WebCenter Portal Framework Application Files Organized?"

**Note:** For pages to be used in the page hierarchy, they must be located under oracle/webcenter/portalapp.

*Figure 5–1  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 5–2. Notice that the folder contains two default page templates that are included out of the box.
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 5–3.

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:

**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.
1. Open the portal project in the Application Navigator, as shown in Figure 5–3.

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 5–4.

**Figure 5–4  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 74.6.2, "Building a Page Hierarchy."
Figure 5–5 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 5–5  Page Hierarchy Editor**

Notice how this hierarchy is realized when you run the portal, shown in Figure 5–6. 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 5.6, "Developing Your Portal's Look and Feel." For more information on the default navigation model and on creating navigation models and user interfaces, see Section 10, "Developing a Navigation Model."

**Figure 5–6  Portal Navigation Bar**

For more information on page hierarchies, see Section 74.6.2, "Building a Page Hierarchy."
5.3.3 Securing WebCenter Portal Framework Pages

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 7.4.6, "Using the Page Hierarchy Security Editor."

As shown in Figure 5–7, 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 5–7, 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 5.3.4, "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.
5.3.4 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, portlets, and 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 10, "Developing a Navigation Model."

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 5–8.

**Figure 5–8  Contents of the navigations Folder**

![Contents of the navigations Folder](image)

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 10, “Developing a Navigation Model.”

The default navigation model, shown in Figure 5–9, 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 5–6, 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 5–9  The Default Navigation Model**

![The Default Navigation Model](image)

Drag ADF pages, tasklets, portlets, or other navigation definitions and drop them in the navigation tree below.
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 5–10 Adding a Link to the Navigation Model**

For more information, see Chapter 10, "Developing a Navigation Model."

### 5.3.5 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 Portal 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 14, "Developing 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 5–11.

To create a new catalog, select New from the File menu. In the New Gallery dialog, select the All Technologies tab, then select Portal Framework under the Web Tier node. Then, select 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.
The Resource Catalog editor is shown in Figure 5–12.

For example you can create separate catalogs for each of several departments.
How are WebCenter Portal Framework Application Files Organized?

When you run through the Portal 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 in the Application Navigator. You can also view your portal project directly on your filesystem.

- Section 5.4.1, "Understanding the Organization of a WebCenter Portal Framework Application"
- Section 5.4.2, "Viewing Your Portal Project on the Filesystem"
- Section 5.4.3, "Viewing Your Portal Project in JDeveloper"
- Section 5.4.4, "Showing Hidden Files in the Application Navigator"

5.4.1 Understanding the Organization of a WebCenter Portal Framework Application

This section explains how a Portal Framework application is organized, and why it is organized the way it is.

5.4.1.1 How Is a WebCenter Portal Framework Application Organized?

When you create a new Portal 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 two projects. The first is called (by default) Portal, as shown in Figure 5-14. You can 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.

The second project is called PortalWebAssets. 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 6.4, "Understanding the PortalWebAssets Project."
5.4.1.2 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 Portal Framework application is structured this way 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.

5.4.2 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 5–15 shows the filesystem organization of a sample Portal Framework application. Many of the project's files are organized under the public_html folder.

**Figure 5–15  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 5.4.3, "Viewing Your Portal Project in JDeveloper."

5.4.3 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 5–16.
How are WebCenter Portal Framework Application Files Organized?

The **Application Sources** folder is primarily a repository for source code and page definition files. 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. 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.

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 Portal Framework application.

Figure 5–17 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 subfolder. This folder’s contents – catalogs, navigations, page hierarchies, pages, skins, and pagetemplates – make up the basic components of a portal. For detailed information on these portal components, see Section 5.4.1, "Understanding the Organization of a WebCenter Portal Framework Application."
How are WebCenter Portal Framework Application Files Organized?

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 6.4, "Understanding the PortalWebAssets Project."

5.4.4 Showing Hidden Files in the Application Navigator

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 5–18.

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.
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.jsp` 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.jsp` 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.

### 5.5 What Is the Portal Life Cycle?

The portal life cycle describes the creation process of a portal from development through staging and testing 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 8, "Understanding the WebCenter Portal Framework Application Life Cycle."

### 5.6 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 5.6, "Developing Your Portal's Look and Feel."

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 13, "Developing 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 15, "Creating Pages and Adding Resources."

### 5.7 Changing Default Portal Preferences

A Portal 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.

#### 5.7.1 What Are the Default Portal Preferences?

Table 5–1 lists the portal preferences and their default configuration files.

<table>
<thead>
<tr>
<th>Preference</th>
<th>Default Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Navigation Model</td>
<td><code>/oracle/webcenter/portalapp/navigations/default-navigation-model.xml</code></td>
</tr>
<tr>
<td>Resource Catalog</td>
<td><code>/oracle/webcenter/portalapp/catalogs/default-catalog.xml</code></td>
</tr>
<tr>
<td>Page Template</td>
<td><code>/oracle/webcenter/portalapp/pagetemplates/pageTemplate_globe.jspx</code></td>
</tr>
<tr>
<td>Navigation Renderer</td>
<td><code>/oracle/webcenter/portalapp/pages/navigation-renderer.jspx</code></td>
</tr>
<tr>
<td>Skin</td>
<td><code>portal</code></td>
</tr>
</tbody>
</table>

#### 5.7.2 How to Change the Default Preferences in JDeveloper

To change the default preferences in JDeveloper, 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 5–19.

![Figure 5–19 Location of the adf-config.xml File in JDeveloper](image-url)
For example, to change the default navigation model file, edit the value of this preference:

```xml
<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 the "Configuring Defaults for Portal Framework Applications" section in the Administrating Oracle WebCenter Portal.

### 5.8 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 Portal 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 information on enabling the iterative development feature in Portal Framework applications, see Section 2.3.1, "Preparing for Iterative Development in a Portal Framework Application." See also Section 8.7, "Understanding 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.

### 5.9 Running and Testing a Portal

JDeveloper provides several ways to run a Portal Framework application for testing purposes in the Integrated WebLogic Server in JDeveloper:

- Right-click the portal project name in the Application Manager and select Run, as shown in Figure 5–20. 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 5.11, "Changing the Default Home Page and Login/Logout Target Pages."
Right-click a page in the pages folder in the Application Manager and select Run, as shown in Figure 5–21. If the server is not running, this action starts the server, (re)deploys the application, and displays the selected page.

You can also run a portal by selecting Run portal name from the Run menu.

5.10 WebCenter Portal Framework Applications at Runtime

Oracle WebCenter Portal Framework provides a number of interesting runtime features. Runtime features refer to features that are accessible from a browser.

- Section 5.10.1, "Preserving Runtime Customizations on the Integrated WebLogic Server"
- Section 5.10.2, "Browser-Based WebCenter Portal Administration Console"
- Section 5.10.3, "How Security Settings Affect the Runtime Portal"
- Section 5.10.4, "Editing of Portal Resources Using Browser-Based Tools"
5.10.1 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 15, "Creating Pages and Adding Resources."

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 refresh 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 5–22.
5.10.2 Browser-Based WebCenter Portal Administration Console

WebCenter Portal Framework applications include a WebCenter Portal Administration Console that 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 5–23.

Figure 5–23 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 5.10.4, "Editing of Portal Resources Using Browser-Based Tools" and Chapter 15, "Creating Pages and Adding Resources." Security topics are discussed in Chapter 74, "Securing Your WebCenter Portal Framework Application."

5.10.3 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 5.3.3, "Securing WebCenter Portal Framework Pages." Oracle WebCenter Portal Framework respects these security policies 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.

5.10.4 Editing of Portal Resources Using Browser-Based Tools

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.
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 5–24.

**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.

![Editing a Page Template Source File at Runtime](image)

### 5.11 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 Portal 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 5.11.1, "Understanding How the Home Page Is Specified"
- Section 5.11.2, "How to Change the Default Home Page"
- Section 5.11.3, "How Is the Default Index Page Specified?"
- Section 5.11.4, "Specifying the Target Page After a Login or Logout"

#### 5.11.1 Understanding How the Home Page Is Specified

The default Portal Framework application includes an index.html file, which is located in the portal project directory. This index file contains the following redirect statement:

```html
<meta http-equiv="refresh" content="0;url=./faces/wcnav_defaultSelection" />
```
Changing the Default Home Page and Login/Logout Target Pages

5.11.2 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:

```html
<meta http-equiv="refresh" content="0;url=./faces/wcnav_defaultSelection" />
```

where .faces/wcnav_defaultSelection is the pretty URL that points to the link element defined in the navigation model.

**Figure 5–27  Navigation Model with New Home Page Element**

Using this example, if access the portal in a browser using this URL:

```
```

the portal will render a home page that contains the content from the new home page, as shown in Figure 5–28.

**Figure 5–28  The Oracle Website Displays as the Content for the Portal Home Page**

5.11.3 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:

```xml
<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 5–29.

Figure 5–29  Edit Run Configuration “Default” Dialog

5.11.4 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 5–1. You can change the target pages to any other navigation resource’s pretty URL or Faces page.

Example 5–1  Specifying the Target Page for Login and Logout

```xml
<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-dropdown 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:
After a successful logout, the user is redirected to the specified page.

### 5.12 Customizing Session Timeouts

You can customize session timeout popups in a Portal Framework application by configuring the default timeout value. You can also customize the text that appears in the popup by modifying the resource strings.

This section contains the following topics:

- Section 5.12.1, "Customizing the Session Timeout Default Value"
- Section 5.12.2, "Customizing the Session Timeout Default Message"

#### 5.12.1 Customizing the Session Timeout Default Value

For better performance, Oracle recommends that the HTTP session timeout value for Portal Framework applications be set to a lower value (for example, 10 minutes) than the out-of-the-box value of 45 minutes. The lower number helps the server's performance, particularly during significant load. The longer the session time, the longer the server will have to hold memory until the inactive sessions expire.

Configuring the session timeout value to a lower number, for example, 10 minutes, can be done through the `web.xml` file:

```xml
<session-config>
  <session-timeout>10</session-timeout>
</session-config>
```

After this value is set, a warning popup appears, with a default value of two minutes, before the session is set to expire (Figure 5–30).

![Figure 5–30 Timeout Warning Popup](image)

After the warning has expired, the message that the page has expired appears (Figure 5–31).

![Figure 5–31 Page Expired Popup](image)

Although these messages are usually valid in many Portal Framework deployments, in the majority of Portal Framework applications, a session is equivalent to being
logged in. For example, in the instance of a public user, the appearance of these messages may not be expected behavior. However, you can easily configure to disable these messages and keep them from appearing.

When a request is sent to the server, a session timeout value is written to the page and the session timeout warning interval is defined by the context parameter `oracle.adf.view.rich.sessionHandling.WARNING_BEFORE_TIMEOUT`.

Use the `oracle.adf.view.rich.sessionHandling.WARNING_BEFORE_TIMEOUT` context parameter to set the number of seconds prior to the session time out when a warning message is displayed. If the value of `WARNING_BEFORE_TIMEOUT` is less than 120 seconds, if client state saving is used for the page, or if the session has been invalidated, the feature is disabled. The session time-out value is taken directly from the session.

In Example 5–2, the `STATE_SAVING_METHOD` value is set to `client`, and the value correlates to the value needed for a page's `af:document` tag. This tag, has the `stateSaving` property set to default, which means it gets the value from the `init-context-param`. Notice that the `WARNING_BEFORE_TIMEOUT` value has been set to 0. Out-of-the-box, the default value is 120 seconds.

**Example 5–2 Session Timeout Warning Configuration**

```
<context-param>
  <param-name>javax.faces.STATE_SAVING_METHOD</param-name>
  <param-value>client</param-value>
</context-param>

<context-param>
  <param-name>oracle.adf.view.rich.sessionHandling.WARNING_BEFORE_TIMEOUT</param-name>
  <param-value>0</param-value>
</context-param>
```

**WARNING:** Any value less than the default value of 120 seconds will disable the popup.

This configuration only prevents the warning message from appearing. Any subsequent page action, partial-page event, or navigation through an `af:commandLink` will produce the un-handled, `ViewExpiredException` (ADF_FACES-60098) in the Faces, `RESTORE_VIEW`, lifecycle (Figure 5–32):

**Figure 5–32 Inactivity Timeout Message**

![Inactivity Timeout Message]

The following log shows the details of this issue:

```
javax.faces.application.ViewExpiredException: viewId:/programs – ADF_FACES-30108:The view state of the page has expired because of inactivity. Reload the page.
at oracle.adfinternal.view.faces.lifecycle.LifecycleImpl._restoreView(LifecycleImpl.java:650)
```
A way of handling this issue is to extend the Portal Framework application to support a custom exception handler. However, an alternative approach is discussed.

The solution is to use a servlet filter to handle the exception and then enable the application to control the recovery destination. For example, one scenario is to be able to either return to the page that the user was previously on, or navigate to the page that the user chooses through one of the (navigation model) page links. The following code supports this scenario:

```java
class SessionExpiryFilter implements Filter {
    private FilterConfig _filterConfig = null;

    public SessionExpiryFilter() {
        super();
    }

    public void init(FilterConfig filterConfig) throws ServletException {
        _filterConfig = filterConfig;
    }

    public void doFilter(ServletRequest servletRequest, ServletResponse servletResponse, FilterChain filterChain) throws IOException, ServletException {
        String requestedSession = ((HttpServletRequest)servletRequest).getRequestedSessionId();
        String currentWebSession = ((HttpServletRequest)servletRequest).getSession().getId();
        String requestURI = ((HttpServletRequest)servletRequest).getRequestURI();
    }
}
```

**Note:** For a Portal Framework application, the use of `goLinks` that use the pretty URL navigation model as the destination is preferred. `goLinks` will not have this particular issue, since the framework will handle creating a new state.
boolean sessionOk =
    currentWebSession.equalsIgnoreCase(requestedSession);
System.out.println("currentWebSession == requestedSession? : " +
    sessionOk);
if (!sessionOk && requestedSession != null) {
    ((HttpServletResponse)servletResponse).sendRedirect(requestURI);
    System.out.println("redirecting to : " + requestURI);
} else {
    filterChain.doFilter(servletRequest, servletResponse);
    System.out.println("session is OK");
}
}

public void destroy() {
    _filterConfig = null;
}

The code essentially compares the value of the session IDs. If the IDs do not match,
than a redirect executes to the page that is determined by the 
getRequestURI(). Otherwise, the request is handled normally.

The following is how the servlet filter is registered with the application through the web.xml file:

<filter>
    <filter-name>AppSessionExpiryFilter</filter-name>
    <filter-class>ateam.sample.SessionExpiryFilter</filter-class>
</filter>
<filter-mapping>
    <filter-name>AppSessionExpiryFilter</filter-name>
    <servlet-name>Faces Servlet</servlet-name>
</filter-mapping>

Note: Both the popup and inactive session error messages can be
easily managed. However, since WebCenter ADF is an application
framework, once the session has been timed out (invalidated), the
state of the application is lost as well. This means any managed beans,
ADF bindings, and so on, is reset to the initial state. In addition, any
information that was being persisted in a managed bean is also lost.

5.12.2 Customizing the Session Timeout Default Message

To customize the session timeout default messages, you need to customize the default
strings provided in the out-of-the-box session timeout popups. The steps to create
custom resource strings, which will override the default values, are detailed in this
section.

The default session timeout messages are shown in Figure 5–30 and Figure 5–31.

All the text-based resource strings (for example, Expiration Warning, and so on) are
customizable. These resource strings are contained in a bundle (RichBundle.java),
which is declared in the skin (CSS). In order to override the defaults, a custom resource
bundle must be created and then declared in a custom skin.

Note: The custom skin declaration in the trindad-skins.xml should also declare to extend the default skin, so the other default resource strings can be picked up.
Example 5–3 is an example of the trinidad-skins.xml, which declares the custom resource bundle:

**Example 5–3 Custom Resource Bundle in trinidad-skins.xml**

```xml
<?xml version="1.0" encoding="windows-1252" ?>
<skins xmlns="http://myfaces.apache.org/trinidad/skin">
  <skin>
    <id>myskin.custom.desktop</id>
    <family>myskindemo</family>
    <render-kit-id>org.apache.myfaces.trinidad.desktop</render-kit-id>
    <style-sheet-name>css/myskindemo.css</style-sheet-name>
    <extends>fusionFx-v1.desktop</extends>
    <bundle-name>com.ateam.view.MyBundleOverride</bundle-name>
  </skin>
</skins>
```

After the skin is declared and selected in the trinidad-skins.xml, the last step is to create the resource bundle file that will hold the custom strings.

For this example, a java-based version for declaring the strings in the resource bundle is created. There are also other ways to declare the strings (for example, .properties and .xliff).

After the configuration to override the default bundle is done, the next step is to declare the strings themselves.

The five default resource strings used are as follows:

- `af_document.PRE_SESSION_TIMEOUT_CONFIRM_TITLE` = Expiration Warning
- `af_document.PRE_SESSION_TIMEOUT_MSG` = This page will expire unless a response is received within {0} minutes. Click OK to prevent expiration.
- `af_document.PRE_SESSION_TIMEOUT_MSG_SECOND` = This page will expire unless a response is received within {0} seconds. Click OK to prevent expiration.
- `af_document.POST_SESSION_TIMEOUT_MSG` = The page has expired
- `af_document.POST_SESSION_TIMEOUT_MSG_CONTINUE` = Click OK to continue.
- `af_document.POST_SESSION_TIMEOUT_ALERT_TITLE` = Page Expired

Example 5–4 shows an example of MyBundleOverride.java, which contains the custom resource strings:

**Example 5–4 Example of Custom Resource Strings**

```java
public class MyBundleOverride extends ListResourceBundle {
  @Override
  public Object[][] getContents() {
    return _CONTENTS;
  }

  static private final Object[][] _CONTENTS = {
    { "af_document.PRE_SESSION_TIMEOUT_CONFIRM_TITLE", "af_document.PRE_SESSION_TIMEOUT_CONFIRM_TITLE : Custom Expiry Warning" },
    { "af_document.PRE_SESSION_TIMEOUT_MSG", "PRE_SESSION_TIMEOUT_MSG : Custom: within {0} minutes. Click OK to prevent expiration." },
    { "af_document.PRE_SESSION_TIMEOUT_MSG_SECOND", "PRE_SESSION_TIMEOUT_MSG_SECOND : Custom: within {0} seconds. Click OK to prevent expiration." }
  };
```
Based on the examples provided in this section, the customized messages before and after a session timeout appear, as shown in Figure 5–33 and Figure 5–34:

Figure 5–33 Custom Warning Message Before Timeout

![Custom Warning Message Before Timeout](image)

Figure 5–34 Custom Message After Timeout

![Custom Message After Timeout](image)

**Tip:** Clear the browser cache if you cannot view the updated strings.

### 5.13 Basic Portal Development Tasks

This section lists some of the basic tasks involved with developing Portal Framework applications.

- Set up an integrated team development environment with source control, a common database, and common content repository.
- Create a Portal 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 features 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 4.1, "Understanding WebCenter Portal Tools and 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.
This chapter explains how to create a new WebCenter Portal Framework application and discusses the organization and contents of a Portal Framework application workspace in JDeveloper.

This chapter includes the following topics:

- Section 6.1, "Creating a New WebCenter Portal Framework Application"
- Section 6.2, "Understanding the WebCenter Portal Framework Application Template"
- Section 6.3, "Building a Portal"
- Section 6.4, "Understanding the PortalWebAssets Project"
- Section 6.5, "How Is Framework Application Security Configured by Default?"
- Section 6.6, "What Configuration Files Should I Know About?"
- Section 6.7, "Adding Excluded Files to the Application Navigator"
- Section 6.8, "Developing Framework Portal Applications for High Availability"
- Section 6.9, "Securing the Portal"

6.1 Creating a New WebCenter Portal Framework Application

This section explains how to use the Create WebCenter Portal Framework Application wizard to create a new portal.

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 6–1.

   Note: When you select WebCenter Portal Framework Application, you are selecting an application template that populates the application with a pre-defined set of projects and files. For more information about this template, see Section 6.2, "Understanding the WebCenter Portal Framework Application Template."
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 Application**. 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 five pages, which are explained in the following steps.

2. Enter a name for the application in the **Application Name** field, as shown in Figure 6–2.
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 6–3.
7. Enter a directory in which to place the project, or use the Browse button to select a directory.

8. 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 6.6, "What Configuration Files Should I Know About?"

9. Click Next.

10. If you wish, enter a Default Package name and directories for the default Java source files and output class files, as shown in Figure 6–4.
11. In the next wizard page, two check boxes appear, as shown in Figure 6–5.
   Generally, the default settings are recommended; however, see the following note
   for more information on this wizard page.

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

Figure 6–5  Create WebCenter Portal Framework Wizard Step 4 of 5
Understanding the WebCenter Portal Framework Application Template

**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.

*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.

12. Click **Next**.

13. 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 6.4, "Understanding the PortalWebAssets Project."

![Create WebCenter Portal Framework Application Wizard Step 5 of 5](image)

14. Click **Finish** to create the WebCenter Portal Framework application.

## 6.2 Understanding the WebCenter Portal Framework Application Template

To readily find and use the appropriate components in your WebCenter Portal Framework 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 an out-of-the-box application template to ensure that scopes are set properly and the right tag and Java libraries are added to create WebCenter Portal Framework applications.

The **WebCenter Portal Framework Application** template 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 6.1, "Creating a New WebCenter Portal Framework Application" and Section 6.4, "Understanding the PortalWebAssets Project."

---

**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 tools and services in your application. The Framework Catalog contains a collection of components that are most commonly used for building portals. For more information see Chapter 14, "Developing Resource Catalogs."

---

### 6.3 Building a Portal

After you have created a new Portal Framework workspace, you can begin setting up navigation, adding pages, developing the look and feel, adding content, and working with other features or "assets." For a general overview of portal features, see Chapter 5, "Understanding WebCenter Portal Framework Applications." For more information on adding features to a portal, see Chapter 9, "Introduction to Portal Resource Management."

### 6.4 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 Portal Framework applications.

- Section 6.4.1, "Introduction"
- Section 6.4.2, "Decoupling the Static Application Resources Project"
- Section 6.4.3, "Deployment Options"
- Section 6.4.4, "Using EL to Manage Static Resources"

#### 6.4.1 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. Portal 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 6–7.

**Figure 6–7  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 Portal Framework application project. Do not attempt to place skin CSS files in the static application resources project.

---

### 6.4.2 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 6–8.
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 6–9.

4. Generate two distinct archives (*Portal.ear* and *PortalWebAssets.jar*), and deploy them to separate servers. See Section 6.4.3, "Deployment Options."

### 6.4.3 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 7, “Deploying and Testing Your
6.4.4 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.

6.4.4.1 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:

```xml
<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 6.4.4.2, "Mapping URLs Dynamically."

6.4.4.2 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/images/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/webcenter/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:
Creating WebCenter Portal Framework Applications

6.5 How Is Framework Application Security Configured by Default?

By default, a WebCenter Portal Framework application is configured with ADF security.

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

6.6 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 Portal Framework application.

This section briefly describes these file:

- Section 6.6.1, "The Framework Application Template Default web.xml File"
- Section 6.6.2, "The Framework Application Template Default faces-config.xml File"
- Section 6.6.3, "The Framework Application Template Default trinidad-config.xml File"
- Section 6.6.4, "The Framework Application Template Default adfc-config.xml File"

6.6.1 The Framework Application Template Default web.xml File

Part of Portal Framework application configuration is determined by the content of its Java EE 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 Portal Framework application.

When you first create your Portal 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.
What Configuration Files Should I Know About?

Creating WebCenter Portal Framework Applications
6-13

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 the "Oracle ADF XML Files" section in the Fusion Developer's Guide for Oracle Application Development Framework.

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

Use the faces-config.xml file to register a Portal 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.

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 Fusion Developer’s Guide for Oracle Application Development Framework.

Example 6–2 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 6–2 Default faces-config.xml File Provided Through the Framework Application Template

```xml
<?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>
  <lifecycle>
    <phase-listener>oracle.webcenter.skin.view.SkinPhaseListener</phase-listener>
  </lifecycle>
  <managed-bean>
    <managed-bean-name>preferenceBean</managed-bean-name>
    <context-param>
      <param-name>javax.faces.DEFAULT_SUFFIX</param-name>
      <param-value>.jsf</param-value>
    </context-param>
  </managed-bean>
</faces-config>
```

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

```xml
<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 the "Oracle ADF XML Files" section in the Fusion Developer’s Guide for Oracle Application Development Framework.
<managed-bean-class>oracle.webcenter.portalframework.sitestructure.preference.PortalPreferences</managed-bean-class>
<managed-bean-scope>application</managed-bean-scope>
</managed-bean>
<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>
</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 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 the "ADF Faces Configuration" section in the Fusion Developer’s Guide for Oracle Application Development Framework.

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

The default trinidad-config.xml file, available when you create a Portal 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 6–3 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 6–3 Default trinidad-config.xml File Provided Through the Framework Application Template

```xml
<?xml version="1.0" encoding="US-ASCII"?>
<trinidad-config xmlns="http://myfaces.apache.org/trinidad/config">
    <skin-family>#{preferenceBean.defaultTrinidadSkin}</skin-family>
    <skin-version>v1</skin-version>
</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 the "ADF Faces Configuration" section in the Web User Interface Developer’s Guide for Oracle Application Development Framework.

### 6.6.4 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.

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 Portal 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 the "Oracle ADF XML Files" section in the Fusion Developer’s Guide for Oracle Application Development Framework.

### 6.7 Adding Excluded Files to the Application Navigator

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 potentially some advanced 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 6–10.

**Figure 6–10  The Exclude Tab**

![Image of the Exclude Tab](image)

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.jsp` 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.jsp` 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.

### 6.8 Developing Framework Portal Applications for High Availability

High availability is very important in large WebCenter installations. It allows users to gracefully switch from one node to another node in the cluster. This allows users to keep their session in case a node goes down.

This section explains some guidelines to keep in mind to fully support an HA environment when implementing Framework Portal applications. Even if you are not implementing with HA in mind, these guidelines may help make your application perform better and with more stability.

The following subsections provide guidelines for implementing for a High Available environment:

- Section 6.8.1, “Introduction to High Availability for WebCenter Portal and JDeveloper”
- Section 6.8.2, “Serializing Managed Beans”
- Section 6.8.3, “Minimizing the Session Footprint”
- Section 6.8.4, “Application Configuration”
- Section 6.8.5, “Mutating Managed Beans”
6.8.1 Introduction to High Availability for WebCenter Portal and JDeveloper

In order to know how we can implement a solution that supports High Availability (HA) we first need to understand how HA works and how WebLogic handles a fail-over.

Whenever you have configured a cluster of managed server for your application, WebLogic will replicate the HTTP session to a secondary machine. When the primary machine where the session is running on fails, WebLogic will point the user to the secondary machine and the user will be able to continue the work without losing hit data.

WebLogic duplicates those sessions by serializing the objects in the session and than transfers it to the secondary machine. Only the object that are stored in the session will be replicated. From an ADF perspective this means that managed bean with pageFlowScope and above will be replicated.

In order for this to work all object in the session need to be serializable. If not, the session cannot completely be replicated which means the user will lose data upon fail-over. WebLogic also need to be notified when the state of an object changes so it can propagate these changes to the secondary machine.

6.8.2 Serializing Managed Beans

One of the key items for a successful HA environment is that all object in the session are serializable. If objects cannot be serializable than WebLogic will not be able to replicate those object to the fail-over server.

ADF UI components are not serializable by design. This means that there are bindings in managed bean with PageFlowScope or SessionScope, the replication will fail.

If there is a need for referencing those UI components in managed bean than we recommend following the ComponentReference pattern which is described in following post:

In addition to this, if custom objects are referenced in the managed bean, you always have to make sure those objects implement the Serializable interface. If not, replication will fail and the user will lose data upon fail-over.

6.8.3 Minimizing the Session Footprint

The session footprint is an important factor when replicating the session. The bigger the session object is, the more traffic the replication will require to propagate the session to the secondary server.

Therefore it is important to keep track of your session footprint. The session footprint can be tracked with tools like JRockit Mission Control (for JRockit JVM) or VisualVM (for Hotspot JVM).

A session footprint of 3MB is considered very high in an ADF application. The reasons for a large footprint can be any of the following:

- Referencing UI components in a managed bean using pageFlowScope or above
- Referencing large data sets (View Objects) in managed beans using pageFlowScope or above
In order to minimize the footprint some design considerations can take place. When putting large objects in managed beans you always need to ask yourself the following question: Does the object really need to be stored in the pageFlowScope? Is ViewScope or RequestScope not enough? In a lot of the cases ViewScope will be sufficient.

Even if you need to pass on information from one view to another you still can use techniques like contextual events or passing parameters instead of storing an entire object in the session.

Lowering the session footprint will not only help in an HA environment. It will also help to optimize the heap size of the JVM. This will allow more users on the node than when a non-optimized implementation is used.

6.8.4 Application Configuration

In order for the application to work in a fail-over environment, we need to tell the application that it needs to behave like an HA application.

This need to be done in two files: weblogic.xml and adf-config.xml:

**weblogic.xml**

In order to enable support for session replication you need to tell WebLogic to replicate the persistent store in a clustered environment. This can be done with following code:

```xml
<weblogic-web-app>
  <session-descriptor>
    <persistent-store-type>
      replicated_if_clustered
    </persistent-store-type>
  </session-descriptor>
</weblogic-web-app>
```

**adf-config.xml**

In order for HA to work in an ADF environment we need to tell the controller to replicate the beans in pageFlowScope and ViewScope. This can be done by setting adf-scope-ha-support to true:

```xml
<adf-controller-config xmlns="http://xmlns.oracle.com/adf/controller/config">
  <adf-scope-ha-support>true</adf-scope-ha-support>
</adf-controller-config>
```

**Business components**

Business components also need some additional configurations in order to support HA:

- jbo.dofailover='true'
- jbo.ampooling='true'

6.8.5 Mutating Managed Beans

Managed beans with a scope higher than RequestScope (ViewScope, pageFlowScope) need to be propagated on the secondary node. The controller won't know automatically when those managed beans are changed.

Whenever you make changes in a managed bean and those changes need to be replicated, you need to notify the controller so it can make sure the updated value of the bean will be passed on the secondary node.

This can be done using the following code:
Map<String, Object> viewScope =
    AdfFacesContext.getCurrentInstance().getViewScope();
controllerContext ctx = ControllerContext.getInstance();
ctx.markScopeDirty(viewScope);

6.8.6 Replication Groups

Within an HA environment, each session will be replicated to a second node. Not all
nodes in the cluster will have all of the session. This is to save network overhead and
performance.

In order to have the best possible outcome, WebLogic will automatically replicate the
session to a different machine than the current node. This is safe because if the
machine fails, the session is available on a different machine.

If you have a big cluster with many machines it is possible that those machines are set
up in different data centers. Therefore, WebLogic allows you to configure replication
groups. Those are configured in order to tell WebLogic which secondary machine to
use for each machine.

In order to configure the replication group you need to configure the cluster settings in
the WebLogic console as shown in Figure 6–11:

Figure 6–11  WebLogic Console - Cluster Setting

6.9 Securing the Portal

You can define security for an entire portal, a collection of pages and sub-pages, or a
single page. You can define security for individual custom components and actions
provided by customizable components. In addition, for Oracle WebCenter Portal tools
and 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 Portal Framework application. You need not synchronize
roles within the portal 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 portal 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 a portal. By associating permissions with an application role defined within the policy store, you can keep them self contained within the portal. 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 portals 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 portal. The portal must map the email user to the portal 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 Portal Framework application, see Chapter 74, “Securing Your WebCenter Portal Framework Application.”
This chapter describes how to run, deploy, and test Portal Framework applications using the Integrated WebLogic Server (IntegratedWebLogicServer), and how to deploy your applications from JDeveloper to a WebLogic Managed Server.

This chapter includes the following topics:

- Section 7.1, "Introduction to Oracle WebLogic Servers"
- Section 7.2, "Deploying a Portal Framework Application to the Integrated WebLogic Server"
- Section 7.3, "Deploying a Portal Framework Application to a WebLogic Managed Server"
- Section 7.4, "Transporting Customizations Between Environments"

---

**Note:** For information about deploying Oracle WebCenter Portal Personalization files, see Section 66.3, "Deploying Personalization Files." For information about deploying 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 "Deploying Portal Framework Applications" chapter in *Administering Oracle WebCenter Portal*.

---

### 7.1 Introduction to Oracle WebLogic Servers

This section provides an overview of the Integrated WebLogic Server and Oracle WebLogic Managed Servers. Using Oracle JDeveloper, you can test and deploy your applications on the Integrated WebLogic Server, 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 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 WebLogic Server, 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 WebLogic Server instance for that application automatically starts. You can also start the instance manually by choosing the Run > Start Server Instance (IntegratedWebLogicServer) 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 Portal Framework applications on the Integrated WebLogic Server 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 WebLogic Server, however, applications running on it are automatically undeployed.

To learn more about testing a Portal Framework application on the Integrated WebLogic Server, refer to Section 7.2, "Deploying a Portal 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 Portal Framework application to an Oracle WebLogic Managed Server, see Section 7.3, "Deploying a Portal Framework Application to a WebLogic Managed Server."

### 7.2 Deploying a Portal Framework Application to the Integrated WebLogic Server

The Integrated WebLogic Server is preconfigured so that you can run applications within Oracle JDeveloper without needing to create deployment profiles. However, when the Integrated WebLogic Server 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 WebLogic Server. These two options are described in the following subsections:

- **Section 7.2.1, "Running a Portal Framework Application in the Integrated WebLogic Server"
- **Section 7.2.2, "Deploying a Portal Framework Application to the Integrated WebLogic Server"**
7.2.1 Running a Portal Framework Application in the Integrated WebLogic Server

To run a Portal Framework application, you can right-click the Portal project in the Projects section 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 Portal Framework application on an Integrated WebLogic Server instance named after the application. You can stop instances of the Integrated WebLogic Server by using the Run Manager panel (shown in Figure 7–1). You can access the Run Manager panel by selecting Run Manager from the View menu. To stop the Integrated WebLogic Server instance, just select IntegratedWebLogicServer and click the red Terminate icon on the Run Manager tab. You can also stop the instance using the red Terminate icon on the toolbar or the one in the Log window, or by selecting the Terminate option from the Run menu.

Figure 7–1 Run Manager

Use the IntegratedWebLogicServer Log window (shown in Figure 7–2) to view how the activity is progressing.

Figure 7–2 DefaultServer Log Window

Note: Secure attributes and credentials are migrated by default when running a Portal Framework application on the Integrated WebLogic Server.
7.2.2 Deploying a Portal Framework Application to the Integrated WebLogic Server

When you run your Portal Framework application as described in Section 7.2.1, "Running a Portal Framework Application in the Integrated WebLogic Server", the application is undeployed and becomes unavailable if the Integrated WebLogic Server instance stops. Even if you start IntegratedWebLogicServer again, the application remains unavailable. For a more persistent testing scenario, you can deploy your application to the Integrated WebLogic Server by using the Deploy option. This deployed application will always be available while the Integrated WebLogic Server is running.

For more information about deploying applications, see the "Deploying Fusion Web Applications" chapter in Fusion Developer’s Guide for Oracle Application Development Framework.

Note: Before deploying your Portal Framework application if you need to create local data sources for Oracle WebCenter Portal and for Activities used by the Analytics task flow, see Section 4.2.2, "Setting Up a Database Connection."

To deploy your application to the Integrated WebLogic Server:

1. Open your application in JDeveloper.
2. From the Run menu, select Start Server Instance (IntegratedWebLogicServer) to start the server instance.
3. From the Application menu, select Deploy, then select the default deployment profile, which is displayed in the format application name_application1 (Figure 7–3)

4. In the Deploy dialog, ensure Deploy to Application Server is selected, then click Next.
5. Select IntegratedWebLogicServer, and click Next.
6. On the Weblogic Options screen, to accept default selections click Next.
7. On the Summary screen, verify the deployment details, and click **Finish**.

8. In the Deployment Configuration dialog, click **Deploy**.

   The Deployment Configuration dialog enables you to choose the target metadata repository or shared metadata repositories. The file system MDS repository, pre-created by JDeveloper, displays in the Repository Name field (Figure 7–4).

**Figure 7–4  Deployment Configuration Dialog**

Note that when you try to redeploy this application, the **Application > Deploy** menu provides the option to select the previous deployment target for your application, as shown in Figure 7–5. This is useful to skip the Deploy dialog screens and directly go to the Deployment Configuration dialog for redeploying to the previous target.
7.3 Deploying a Portal Framework Application to a WebLogic Managed Server

When you are ready to test your Portal 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 Portal Framework applications using Fusion Middleware Control, Oracle WebLogic Administration Console, and WLST commands, see the "Deploying Portal Framework Applications" chapter in Administering Oracle WebCenter Portal.

This section includes the following subsections:
- Section 7.3.1, "Deployment Roadmap"
- Section 7.3.2, "Packaging a Framework Application"
- Section 7.3.3, "Preparing the Target Environment for Deployment"
- Section 7.3.4, "Deploying a Portal Framework Application to a Managed Server"

7.3.1 Deployment Roadmap

The flowchart and table in this section provide an overview of the prerequisites and tasks required to deploy a Portal Framework application to an Oracle WebLogic Managed Server. Figure 7–6 shows the steps to deploy a Portal Framework application, and the roles that will carry them out.
Table 7–1 shows the tasks, sub-tasks and who will need to carry them out to deploy a Portal Framework application from JDeveloper.
## Table 7–1  Deploying a Framework Application to a Managed Server

<table>
<thead>
<tr>
<th>Actor</th>
<th>Task</th>
<th>Sub-task</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developer</td>
<td>1. Package the Application</td>
<td>1.a Select the data source type (package database connections)</td>
<td>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). <strong>Note:</strong> 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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.b Package application security data</td>
<td>This sub-task consists of packaging the credentials, identity data, and application policies.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.c Create deployment profiles</td>
<td>This sub-task consists of creating the WAR and EAR files. Out-of-the-box, there are default deployment profiles available. However, you may want to create your own deployment profile to specify the context root and application name as per your requirements.</td>
</tr>
<tr>
<td>Administrator</td>
<td>2. Prepare the Target Environment</td>
<td>2.a Create and provision the Managed Server</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.b Create and register the MDS repository</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.c Configure the target environment</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.d Create the server connection</td>
<td></td>
</tr>
<tr>
<td>Developer</td>
<td>3. Deploy the Application to a Managed Server from JDeveloper</td>
<td></td>
<td>The final step is to deploy the application to the Managed Server from JDeveloper.</td>
</tr>
</tbody>
</table>
7.3.2 Packaging a Framework Application

When you deploy 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 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 7.3.2.1, "Packaging Database Connections and Application Security"
- Section 7.3.2.2, "Creating Deployment Profiles"

7.3.2.1 Packaging Database Connections and Application Security

When you deploy 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 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 7.3.2.1.1, "Packaging the Database Connections"
- Section 7.3.2.1.2, "Packaging Application Security Data"

7.3.2.1.1 Packaging the Database Connections

If your 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** check box 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 about creating global data sources, see the "Creating a JDBC Data Source" section in 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 WebLogic Server. That
is, when the **Auto Generate and Synchronize weblogic-jdbc.xml Descriptors During Deployment** check box is selected in the Application Properties dialog, as shown in **Figure 7–7**.

**Figure 7–7  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 "Creating a JDBC Data Source" section in 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.

7.3.2.1.2 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** check box in the **Security Deployment Options** section in the Application Properties dialog (see Figure 7–8).

*Figure 7–8 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 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 security-related sections in Administering Oracle WebCenter Portal.

For more information about credential migration behavior, see the “Managing the Credential Store” section in Securing Applications with Oracle Platform Security Services.

- When you deploy a Portal 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 7–8, in the Security Deployment Options section, the Users and Groups check box 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 identity 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 check box 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 check box 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.

### 7.3.2.2 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 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).
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 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 Portal 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.

If your application, however, was not created using the WebCenter Portal Framework Application template, then you may 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. For information about creating an EAR deployment profile for an application, see the “Deploying Fusion Web Applications” chapter in Fusion Developer’s Guide for Oracle Application Development Framework.

To create an EAR file:

1. Open your application in JDeveloper.
2. From the Application menu, choose Deploy, then select the name of the deployment profile. An EAR-level deployment profile is available out of the box, named `application name_application1`.
3. In the Deploy dialog, on the Deployment Action screen, select Deploy to EAR, then click Next.
4. On the Summary screen, verify the deployment details, and click Finish.

This creates the EAR file in the `deploy` folder located in `JDEV_HOME\mywork\application_name\deploy`, as shown in Figure 7–9.

**Figure 7–9 EAR Status in Deployment - Log**

<table>
<thead>
<tr>
<th>Message</th>
<th>Time</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Running dependency analysis...</td>
<td>11:39:27</td>
<td>Message</td>
</tr>
<tr>
<td>Building...</td>
<td>11:39:27</td>
<td>Message</td>
</tr>
<tr>
<td>Deploying 3 profiles...</td>
<td>11:39:27</td>
<td>Message</td>
</tr>
<tr>
<td>Exporting portlet metadata and customizations...</td>
<td>11:39:27</td>
<td>Message</td>
</tr>
<tr>
<td>Writing EAR file to D:\emple\deploy\Application\deploy\AutoGeneratedMar.mar</td>
<td>11:39:27</td>
<td>Message</td>
</tr>
<tr>
<td>Info: Namespace '/oracle/adf/share/preferences' is mapped to deploy-target metadata store usage 'WebCenterFile'</td>
<td>11:39:31</td>
<td>Message</td>
</tr>
<tr>
<td>Info: Namespace '/oracle/webcenter/31/preferences' is mapped to deploy-target metadata store usage 'WebCenterFile'</td>
<td>11:39:31</td>
<td>Message</td>
</tr>
<tr>
<td>Info: Namespace '/oracle/webcenter/sf' is mapped to deploy-target metadata store usage 'WebCenterFile'</td>
<td>11:39:31</td>
<td>Message</td>
</tr>
<tr>
<td>Info: Namespace '/oracle/webcenter/sf' is mapped to deploy-target metadata store usage 'WebCenterFile'</td>
<td>11:39:31</td>
<td>Message</td>
</tr>
<tr>
<td>Time: Flapped time for deployment: 44 seconds</td>
<td>11:39:32</td>
<td>Message</td>
</tr>
<tr>
<td>**** Deployment finished.</td>
<td>11:39:32</td>
<td>Message</td>
</tr>
</tbody>
</table>

---

**Note:** You can deploy Oracle ADF applications, such as Portal Framework applications, only as EAR files.
7.3.3 Preparing the Target Environment for Deployment

This section describes how to create and provision a WebLogic Managed Server instance prior to deploying your Portal Framework application.

This section includes the following sub-sections:

- Section 7.3.3.1, "Creating and Provisioning an Oracle WebLogic Managed Server Instance"
- Section 7.3.3.2, "Creating and Registering the Metadata Service Repository"
- Section 7.3.3.3, "Configuring the Target Environment"
- Section 7.3.3.4, "Creating a WebLogic Managed Server Connection"

7.3.3.1 Creating and Provisioning an Oracle WebLogic Managed Server Instance

Before deploying a 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 an MDS Repository. For instructions on how to create a new managed server, see the "Extending an Existing Domain" section in Installation Guide for Oracle WebCenter Portal.

Note: Oracle does not recommend deploying Portal Framework applications to any of the preconfigured Managed Servers created during Oracle WebCenter Portal installation, or to a domain Administration Server.

7.3.3.2 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 "Creating and Registering the Metadata Service Repository" section in Administering Oracle WebCenter Portal.

7.3.3.3 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 "Creating a JDBC Data Source" section in Configuring and Managing JDBC Data Sources 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.

7.3.3.4 Creating a WebLogic Managed Server Connection

Before you can deploy your applications using JDeveloper to an Oracle WebLogic Managed Server instance that resides 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 Weblogic Domain field, specify the name of the domain in which the WebLogic Managed Server instance is created (for example, wc_domain), 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.

9. Click Finish.

7.3.4 Deploying a Portal Framework Application to a Managed Server

You can deploy a Portal 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 configuration 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 "Choosing the Data Source" section in Administering Oracle WebCenter Portal.

This section contains the following subsections:

- Section 7.3.4.1, "Deploying to a Managed Server Using Local Data Sources"
- Section 7.3.4.2, "Deploying to a Managed Server Using Global Data Sources"

7.3.4.1 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 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 a local data source if your application contains task flows that use the WebCenter schema. If your application uses Analytics task flows, create a separate
local data source for the Activities schema. For information, see Section 4.2.2,
"Setting Up a Database Connection."

3. From the Application menu, select Application Properties.
The Application Properties dialog displays (Figure 7–10).

![Figure 7–10 Application Properties Dialog](image)

4. Choose Deployment from the navigation panel to display the Deployment options.

5. Make sure that the Auto Generate and Synchronize weblogic-jdbc.xml
Descriptors During Deployment check box is selected and click OK.

6. From the Application menu, select Deploy and then the application name.
The Deployment Action dialog displays (Figure 7–11).
7. Select **Deploy to Application Server** and click **Next**.

8. Select the connection name of the Managed Server (for example, **WC_CustomPortal**), and click **Next** (**Figure 7–12**).

   If the server to which you want to deploy is not listed, click the Add icon (+) and complete the connection details to add the server to the list.

**Figure 7–12  Select Server Dialog**

9. Select **Deploy to selected instance in the domain**, select the managed server from the list (for example, **WC_CustomPortal**), and click **Next** (**Figure 7–13**).
In most cases you will want to deselect **Deploy to all instances in the domain** and select the specific servers and clusters to deploy to. Refer to the online help for more information about using these options.

**Figure 7–13  Deployment WebLogic Options dialog**

![Deployment WebLogic Options dialog](image)

10. On the Deployment Summary dialog, verify the deployment details. Click **Finish** to start the deployment.

   The Deployment Configuration dialog displays (**Figure 7–14**).

11. Select the Repository Name (for example, mds-CustomPortalDS), enter a Partition Name (if the partition does not already exist then it will be created during deployment), and click **Deploy**.
12. Continue by doing any post-deployment security or data source connection configurations that may be required as described in the "Post-Deployment Configuration" section in Administering Oracle WebCenter Portal.

7.3.4.2 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 nondefault 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 "Choosing the Data Source" section in Administering 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:
3. From the Application menu, select Application Properties.

4. Choose Deployment from the navigation panel to display the Deployment options.

5. Deselect the Auto Generate and Synchronize weblogic-jdbc.xml Descriptors During Deployment check box if selected and click OK.

6. From the Application menu, select Deploy, and then the application name.

7. Select Deploy to Application Server and click Next (Figure 7–15).

**Figure 7–15 Choosing a Deployment Action**

8. Select the connection name of the Managed Server (for example, WC_CustomPortal), and click Next (Figure 7–16).

   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.

<table>
<thead>
<tr>
<th>Data Source</th>
<th>JDeveloper JNDI Name</th>
<th>Default JNDI Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>WebCenterDS</td>
<td>jdbc/WebCenterConnectionNameDS</td>
<td>jdbc/webcenter/CustomPortalDS</td>
</tr>
<tr>
<td></td>
<td>(where WebCenterConnectionName is the name of the WebCenter schema database connection)</td>
<td></td>
</tr>
<tr>
<td>ActivitiesDS</td>
<td>jdbc/ActivitiesConnectionNameDS</td>
<td>jdbc/activities/CustomPortalDS</td>
</tr>
<tr>
<td></td>
<td>(where ActivitiesConnectionName is the name of the Activities schema database connection)</td>
<td></td>
</tr>
</tbody>
</table>
9. Select Deploy to selected instance in the domain, select the managed server from the list (for example, WC_CustomPortal), and click Next (Figure 7–17).

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 these options.

10. On the Deployment Summary dialog, verify that the deployment options are correct and click Finish to start the deployment.

The Deployment Configuration dialog displays (Figure 7–18).
11. Select the Repository Name (for example, mds-CustomPortalDS), enter a Partition Name (if the partition does not already exist then it will be created during the deployment), and click **Deploy**.

**Figure 7–18  Deployment Configuration Dialog**

![Deployment Configuration Dialog]

12. Continue by doing any post-deployment security or data source connection configurations that may be required as described in the "Post-Deployment Configuration" section in *Administering Oracle WebCenter Portal*.

### 7.4 Transporting Customizations Between Environments

When migrating a deployed application to a new environment, post-deployment customizations made to pages, WebCenter Portal tools and services, 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 "Managing Export, Import, Backup, and Recovery for Portal Framework Applications" chapter in *Administering Oracle WebCenter Portal*. 
Understanding the WebCenter Portal Framework Application Life Cycle

This chapter discusses tasks, tools, and techniques for managing a WebCenter Portal Framework application throughout its life cycle.

This chapter includes the following topics:

- Section 8.1, "What Is the WebCenter Portal Framework Application Life Cycle?"
- Section 8.2, "What Are the Major Life Cycle Tasks?"
- Section 8.3, "Who Participates in the Portal Life Cycle?"
- Section 8.4, "Understanding the Build and Test Environments"
- Section 8.5, "Understanding the Staging and Production Environments"
- Section 8.6, "Tools for Managing the Life Cycle"
- Section 8.7, "Understanding Iterative Development"
- Section 8.8, "Configuring a Nightly Build Script"
- Section 8.9, "Setting Up a Staging or Production Environment for the First Time"
- Section 8.10, "Moving a WebCenter Portal Framework Application to an Existing Environment"
- Section 8.11, "Deploying to Managed Servers"
- Section 8.12, "Deploying and Configuring the Application on Targeted Servers"
- Section 8.13, "Using the Propagation Tool to Propagate From Staging to Production"
- Section 8.14, "Propagating Content From Oracle WebCenter Content"
- Section 8.15, "Managing Security Through the Life Cycle"
- Section 8.16, "Migrating Portlet Preferences"
- Section 8.17, "Rolling Back Production Site Changes"

8.1 What Is the WebCenter Portal Framework Application Life Cycle?

The WebCenter Portal Framework application life cycle refers to the path a Framework Portal application 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 8–1.

<table>
<thead>
<tr>
<th>Life Cycle Phase</th>
<th>Primary Actors/Roles</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development</td>
<td>- Developers</td>
<td>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:</td>
</tr>
<tr>
<td></td>
<td>- Content Modelers</td>
<td>■ portlets</td>
</tr>
<tr>
<td></td>
<td>- Content Contributors</td>
<td>■ task flows</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ shared libraries</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ skins</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ navigation models</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ page templates</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ display templates</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ content models</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ data transfer and interportlet communication</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ security</td>
</tr>
<tr>
<td></td>
<td></td>
<td>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 8.8, &quot;Configuring a Nightly Build Script.&quot;</td>
</tr>
</tbody>
</table>
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 section provides an overview of the kinds of tasks that are performed during each phase of the portal life cycle.

- **Section 8.2.1, "One-Time Setup Tasks"
- **Section 8.2.2, "Development Environment Tasks"
- **Section 8.2.3, "Nightly Build Environment Tasks"
- **Section 8.2.4, "Testing Environment Tasks"
- **Section 8.2.5, "Stage Environment Tasks"
- **Section 8.2.6, "Production Environment Tasks"
8.2.1 One-Time Setup Tasks

You must perform certain preparatory steps to set up the development, build/test, stage, and production environments. Table 8–2 provides a general list of these preliminary setup tasks and the environments to which they apply. See also Section 8.8, "Configuring a Nightly Build Script" and Section 8.9, "Setting Up a Staging or Production Environment for the First Time."

<table>
<thead>
<tr>
<th>Setup Task</th>
<th>Development</th>
<th>Build/Test</th>
<th>Stage</th>
<th>Production</th>
</tr>
</thead>
<tbody>
<tr>
<td>Install JDeveloper</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Install Oracle WebCenter Portal</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Install Oracle WebLogic Server; create a domain and managed servers</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Create required database schemas using RCU</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Install and configure Oracle WebCenter Content</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Install identity management components, such as Oracle Access Manager</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Create the required Oracle Platform Security Services policies in the policy store</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Create required user credentials in the credential store</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Create connections to back end servers</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Create build scripts</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Create deploy and configure scripts</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Integrate/configure Oracle WebCenter Portal Personalization *</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

* For more information on Oracle WebCenter Portal Personalization, see the chapter "Managing Personalization" in the Administering Oracle WebCenter Portal and Chapter 66, "Personalizing Oracle 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 the section "Moving from a Test to a Production Environment" in the Administrator’s Guide.

8.2.2 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 2, "Setting Up Your Development Environment" and Chapter 3, "Working Productively in Teams."

8.2.3 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 8.8, "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 8.4, "Understanding the Build and Test Environments."

For information on integrating Oracle WebCenter Portal Personalization into your development environment, see Section 66.2, "Integrating Personalization in Your Application."

8.2.4 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 8.4, "Understanding the Build and Test Environments."

8.2.5 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 8.5.1, "Provisioning the Staging Environment" and Section 8.12, "Deploying and Configuring the Application on Targeted Servers."

For information on integrating and configuring Oracle WebCenter Portal Personalization, see the section 'Personalization Prerequisites and Limitations' in the Administering Oracle WebCenter Portal.

8.2.6 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 8.13, "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 8.16, "Migrating Portlet Preferences."
8.3 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 8–1) fall into one or more of these general roles:

- **Developer** – Uses JDeveloper and the build scripts described in Section 8.8, "Configuring a Nightly Build Script."

- **IT Administrator** – Uses the deploy and configuration scripts described in Section 8.12, "Deploying and Configuring the Application on Targeted Servers."

- **Site managers** – Have administrative privileges for the Portal Framework application. Site managers use the WebCenter Portal Administration Console to modify the layout, content, and security settings of the portal. See also *Administering Oracle WebCenter Portal* and Section 8.15, "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.

8.4 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 Portal 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 8–1* illustrates the general flow from development to build to test environments.

---

**Note:** *Figure 8–1* does not depict all possible portal features. For example, Oracle WebCenter Portal Personalization is not depicted in the diagram. For more information, see Section 66.3, "Deploying Personalization Files."
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 8.8, "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 8.14, "Propagating Content From Oracle WebCenter Content." For information on deploying portlets, see Chapter 61, "Deploying Portlet Producers."
8.5 Understanding the Staging and Production Environments

This section discusses the staging and production phases of the portal life cycle. Figure 8–2 illustrates the general flow from staging to production environments.

Note: Figure 8–2 does not depict all possible portal features. For example, Oracle WebCenter Portal Personalization is not depicted in the diagram. For more information, see Section 66.3, "Deploying Personalization Files.”
8.5.1 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 8–2.

If you are setting up the staging environment for the first time, see Section 8.9, "Setting Up a Staging or Production Environment for the First Time." For information on making incremental changes to the staging environment, see Section 8.12, "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 8.14, "Propagating Content From Oracle WebCenter Content." For information on migrating portlets, see Section 8.16, "Migrating Portlet Preferences."

For a complete list of requirements, dependencies, and options for WebCenter Personalization, see the section "Personalization Prerequisites and Limitations" in the Administering Oracle WebCenter Portal.

8.5.2 Adding Content to the Staging Environment

Content developers can add content directly to the staging server using Oracle WebCenter Content content contribution tools. Content workflow features of Oracle WebCenter Content can be used to manage content approvals. WebCenter also provides browser-based 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

8.5.3 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 8.13, "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 8.14, "Propagating Content From Oracle WebCenter Content."

WebCenter provides WLST commands for importing and exporting portlet preferences. For more information, see Section 8.16, "Migrating Portlet Preferences."

8.6 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:
Understanding Iterative Development

- **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 Portal 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 2.3.1, "Preparing for Iterative Development in a Portal Framework Application."

- **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 9.6, "Working with Round-Trip Development."

- **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 8.12, "Deploying and Configuring the Application on Targeted Servers" and Section 8.13, "Using the Propagation Tool to Propagate From Staging to Production." For more information about deployment, see Chapter 7, "Deploying and Testing Your Portal Framework Application."

---

**Note:** Services for WebCenter Portal 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 Portal Framework applications and WebCenter Portal tools and 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. Portal Framework application metrics are delivered using the familiar Application Server Control monitoring and management interface. For more information, see *Administering Oracle WebCenter Portal*.

---

### 8.7 Understanding Iterative Development

- Section 8.7.1, "What Is Iterative Development?"

- Section 8.7.2, "How Does Iterative Development Work?"

- Section 8.7.3, "Enabling Iterative Development"

### 8.7.1 What Is Iterative Development?

Iterative development lets you make changes to your Portal 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 portal.
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, and so on)
- Editing any configuration file, like web.xml or adfc-config.xml.
- 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.

### 8.7.2 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:

```xml
<context-param>
  <param-name>org.apache.myfaces.trinidad.DISABLE_CONTENT_COMPRESSION</param-name>
  <param-value>true</param-value>
</context-param>
```
Note: When consuming Oracle JSF Portlet Bridge Portlets in your portal, 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 58, “Creating Portlets from JSF Applications Using 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**.

### 8.7.3 Enabling Iterative Development

For details, see **Section 2.3.1.1, "Enabling Iterative Development."**

### 8.8 Configuring a Nightly Build Script

When you create a Portal 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 Oracle WebCenter Portal Personalization to provide personalization features for your portal, you must perform a manual update to the build.xml file. For details, see Section 66.3, "Deploying Personalization Files."

8.9 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 the section "Moving Oracle WebCenter Portal to a New Target Environment," in the Administrator’s Guide. See also Section 8.15, "Managing Security Through the Life Cycle" for information on moving security policies and credentials to an environment for the first time.

8.10 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 configurations.

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 8.12, "Deploying and Configuring the Application on Targeted Servers" and Section 66.3, "Deploying Personalization 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 8.13, "Using the Propagation Tool to Propagate From Staging to Production."

For information on moving Oracle WebCenter Content content, see Section 8.14, "Propagating Content From Oracle WebCenter Content."

For information on moving security policies and credentials, see Section 8.15, "Managing Security Through the Life Cycle."

For more information, see the section "Moving Oracle WebCenter Portal to an Existing Target Environment," in the Administrator’s Guide.

8.11 Deploying to Managed Servers

Portal 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 7.3, "Deploying a Portal Framework Application to a WebLogic Managed Server" and Section 66.3, "Deploying Personalization Files." See also the section "Creating a Managed Server" in Administering Oracle WebCenter Portal.
8.12 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 Portal Framework application on a targeted server.

- Section 8.12.1, "Introduction"
- Section 8.12.2, "Using the Deploy and Configure Script"

8.12.1 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 8.13, "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 Portal Framework application do not need to change when you redeploy it to another server. For example, the name of the application doesn’t change.

8.12.2 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 8.14, "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 8.13, "Using the Propagation Tool to Propagate From Staging to Production." See also Section 8.15, "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 \texttt{WEBCENTER\_HOME/webcenter/scripts/stage2prod}, where \texttt{WEBCENTER\_HOME} is the directory where WebCenter is installed.

\textbf{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 \texttt{setup.properties} file, like the target server URL, user name, and password. The \texttt{setup.properties} file is located in \texttt{WEBCENTER\_HOME/webcenter/scripts/stage2prod/setup}. Simply open the file \texttt{setup.properties} and add the appropriate values for the target environment. A sample file is shown in \textbf{Example 8–1}.

\textbf{Note:} For WebLogic Server deployments, you can ignore properties listed under WebSphere Server properties. See also the section "Using the Deploy and Configure Script for Applications Deployed on WebSphere" in the \textit{Third-Party Application Server Guide}.

\textit{Example 8–1 Sample setup.properties File}

\begin{verbatim}
# 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
\end{verbatim}

3. Update the \texttt{create_profile.csh} and \texttt{deploy_and_config.csh} to reflect the deployed environment.

\begin{verbatim}
setenv WC_HOME <webcenter_home>
setenv SCRIPTS_DIR <scripts_home>
\end{verbatim}

\texttt{WC_HOME} is the WebCenter Home and \texttt{SCRIPTS_DIR} is where the scripts are located. By default, the scripts are here: \texttt{WC\_HOME/webcenter/scripts/stage2prod}. If you copied the scripts to another location, then set \texttt{SCRIPTS\_DIR} to that location.

4. Run the \texttt{create_profile} script. The input to this script is the \texttt{setup.properties} file. For example, in a Linux environment, enter:

\begin{verbatim}
./create_profile.csh
\end{verbatim}

This script examines your application environment and produces an output properties file called \texttt{wcconfig.properties} in \texttt{WEBCENTER\_HOME/webcenter/scripts/stage2prod/setup}.

5. If you wish, rename the output file, \texttt{wcconfig.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 \texttt{wstage.properties}. 

\textbf{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).
The `wcconfig.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, Oracle WebCenter Portal Personalization and others. Example 8–2 shows a sample profile.properties file.

**Example 8–2  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
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. Edit the `deploy_and_config` script to accept as input the `wcconfig.properties` file (or whatever you renamed it).

8. Run the `deploy_and_config` 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 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 Portal Framework application to run on the target environment.
8.12.3 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 the section “Oracle WebCenter Portal Administration Tools” in the Administering Oracle WebCenter Portal.

---

**Note:** 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.

---

8.13 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 Oracle WebCenter Portal Personalization files. For more information about propagating Personalization files, see Section 66.3.4, “Propagating Personalization 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 8.16, “Migrating Portlet Preferences.”

---

- Section 8.13.1, "Introduction to the Propagation Tool"
- Section 8.13.2, "Configuring the Propagation Tool"
- Section 8.13.3, "Propagating Portal Metadata"

8.13.1 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 8–3.

**Figure 8–3  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 8–4.

**Figure 8–4  The Propagation Tab with Labels Opened**
8.13.2 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.

8.13.2.1 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 8–5.

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 8–5.

Figure 8–5 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 8–6.

Note: You must name the connection ProductionURLConnection. The name is case sensitive.
Using the Propagation Tool to Propagate From Staging to Production

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 8–7.

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.

### 8.13.2.2 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 8.13.2.1, “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:
Managing Security Through the Life Cycle

Understanding the WebCenter Portal Framework Application Life Cycle

8.13.3 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 8.13.2, "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

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 8.13.2, "Configuring the Propagation Tool."

4. Click Propagate to transfer the files to the production server.

8.14 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 the chapter "Understanding 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.

8.15 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.

### 8.15.1 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 8–3 shows the recommended settings in `weblogic-application.xml` to achieve automatic policy migration for first-time deployment.

**Example 8–3 Settings for Automatic Policy Migration (weblogic-application.xml)**

```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.JpsApplicationLifecycleListener</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, policies packaged in `jazn-data.xml` will be always migrated during first 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.

### 8.15.2 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.

### 8.15.3 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 8–3 shows the recommended settings in weblogic-application.xml to achieve automatic credential migration for first-time deployment.

Example 8–4  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.

8.15.4 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.

8.16 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 “WebCenter Portal Custom WLST Commands” in the WebLogic Scripting Tool Command Reference. Also ensure that export and import are enabled. See Section 59.3.15, “How to Export and Import Portlet Producers at Design Time.”

See also Section 61.5, "Deploying a Portlet Producer to a WebLogic Managed Server.”

8.17 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.
Rolling Back Production Site Changes
This chapter describes the different portal resources available for you to use in your WebCenter Portal Framework applications, and some of the common operations you can perform on them. Portal resources include navigation models, page templates, page styles, resource catalogs, and so on.

This chapter includes the following topics:

- Section 9.1, "Introduction to Portal Resources"
- Section 9.2, "Creating Portal Resources"
- Section 9.3, "Editing Portal Resources"
- Section 9.4, "Deleting Portal Resources"
- Section 9.5, "Working with Portal Resources at Runtime"
- Section 9.6, "Working with Round-Trip Development"
- Section 9.7, "Working with WebCenter Portal Resources"

**Tip:** Portal resources are also sometimes referred to as assets.

### 9.1 Introduction to Portal Resources

Portal Framework applications provide the following portal resources to help you define the structure, look and feel, and content of your portal:

- **Page Templates** define the interface that surrounds page content and help to apply a consistent look and feel across groups of pages. For more information, see Chapter 11, "Developing Page Templates."

- **Navigation Models** define how to link together information from multiple sources, such as pages, content repositories, and external web pages. For more information, see Chapter 10, "Developing a Navigation Model."

- **Resource Catalogs** define the components that users can add to their pages, page templates, and task flows. For more information, see Chapter 14, "Developing Resource Catalogs."

- **Skins** define the appearance and look and feel, including colors and fonts, of a specific portal or the entire application. For more information, see Chapter 13, "Developing Skins."

- **Page Styles** define the layout of a newly created page, and may also dictate the type of content the page supports. For more information, see Section 12.2, "Developing Page Styles."
Creating Portal Resources

Content Presenter Display Templates define templates for displaying content. For more information, see Chapter 27, "Creating Content Presenter Display Templates."

Task Flow Styles are used at runtime to define the layout of task flows created using Data Presenter to display data controls. For more information, see Section 12.3, "Developing Task Flow Styles."

9.2 Creating Portal Resources

The process for creating a portal resource depends on the type of portal resource. Table 9–1 shows where to find further information about creating the different portal resources.

<table>
<thead>
<tr>
<th>Portal Resource</th>
<th>More Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Page Template</td>
<td>Section 11.3, &quot;Creating a Page Template&quot;</td>
</tr>
<tr>
<td>Navigation Model</td>
<td>Section 10.2, &quot;Creating a Navigation Model&quot;</td>
</tr>
<tr>
<td>Resource Catalog</td>
<td>Section 14.2, &quot;Creating a Resource Catalog&quot;</td>
</tr>
<tr>
<td>Skin</td>
<td>Section 13.3, &quot;Creating a Skin&quot;</td>
</tr>
<tr>
<td>Page Style</td>
<td>Section 12.2.1, &quot;How to Create a Page Style&quot;</td>
</tr>
<tr>
<td>Content Presenter Display Template</td>
<td>Section 27.3, &quot;Creating Content Presenter Display Templates&quot;</td>
</tr>
<tr>
<td>Task Flow Style</td>
<td>Section 12.3.1, &quot;How to Create a Task Flow Style&quot;</td>
</tr>
</tbody>
</table>

9.3 Editing Portal Resources

The process for editing a portal resource depends on the type of portal resource. Table 9–2 shows where to find further information about editing the different portal resources.

<table>
<thead>
<tr>
<th>Portal Resource</th>
<th>More Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Page Template</td>
<td>Section 11.4, &quot;Editing a Page Template&quot;</td>
</tr>
<tr>
<td>Navigation Model</td>
<td>Section 10.3, &quot;Editing a Navigation Model&quot;</td>
</tr>
<tr>
<td>Resource Catalog</td>
<td>Section 14.3, &quot;Editing a Resource Catalog&quot;</td>
</tr>
<tr>
<td>Skin</td>
<td>Section 13.4, &quot;Editing a Skin&quot;</td>
</tr>
<tr>
<td>Page Style</td>
<td>Section 12.2.1, &quot;How to Create a Page Style&quot;</td>
</tr>
<tr>
<td>Content Presenter Display Template</td>
<td>Section 27.3, &quot;Creating Content Presenter Display Templates&quot;</td>
</tr>
<tr>
<td>Task Flow Style</td>
<td>Section 12.3.1, &quot;How to Create a Task Flow Style&quot;</td>
</tr>
</tbody>
</table>

9.4 Deleting Portal Resources

When a portal resource is no longer required, you may want to remove it.
To delete a portal resource:

1. In the Application Navigator, right-click the portal resource that you want to delete and choose **Delete**.

2. The Confirm Delete dialog identifies if the portal resource 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 portal resource.

3. Click **Yes** to continue and delete the navigation model. Click **No** to cancel the operation.

### 9.5 Working with Portal Resources at Runtime

**Tip:** At runtime, portal resources are referred to as assets.

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 portal resources, such as pages, skins, and data controls at runtime. For more information, see Section 9.5.1.5, "How to Use the Assets Page at Runtime."

- **Round-trip development**—Pull portal 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 9.6, "Working with Round-Trip Development."

- **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 2.3.1, "Preparing for Iterative Development in a Portal Framework Application."

- **Extension of WebCenter Portal**—Build and manage new WebCenter Portal resources in JDeveloper and package them on a periodic basis for distribution to the runtime portal. For more information, see Section 55.1, "Developing Assets for WebCenter Portal."

Portal Framework applications enable runtime portal resource management through the Assets page of the runtime administration console. Using the administration console, users with the appropriate privileges can continue developing the portal after the application has been deployed (Figure 9–1).
At runtime, users with the appropriate privileges can create and manage the following resources:

- Pages
- Page templates
- Navigation models
- Resource catalogs
- Skins
- Page styles (can be created at runtime only by copying an existing page style)
- Content Presenter display templates (cannot be created at runtime)
- Task flow styles (can be created at runtime only by copying an existing task flow style)
- Task flows
- Data controls

Using the Assets page of the runtime administration console, users can also download portal 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 portal resources from scratch in JDeveloper, export them to an archive file, 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.

This process is known as round-trip development. For more information, see Section 9.6, "Working with Round-Trip Development."

### 9.5.1 Enabling Runtime Administration of Portal Resources

You can enable portal administrators to manage portal resources at runtime by exposing them on the Assets page of your application.
If you use the WebCenter Portal Framework Application template to create your application, the Assets page is automatically provided as part of the runtime administration console.

If you prefer, you can add the Assets page to any other page in your application by adding the List of Resource Types task flow to that page.

For a portal resource to be exposed on the Assets page, and therefore available for runtime administration, it must be located under the `oracle/webcenter/portalapp` directory of the application project and explicitly identified as a portal resource.

This section includes the following topics:

- Section 9.5.1.1, "How to Add the Assets Page to a Page"
- Section 9.5.1.2, "How to Include a Portal Resource on the Assets Page"
- Section 9.5.1.3, "How to Update Portal Resource Properties"
- Section 9.5.1.4, "How to Remove a Portal Resource from the Assets Page"
- Section 9.5.1.5, "How to Use the Assets Page at Runtime"

### 9.5.1.1 How to Add the Assets Page to a Page

For applications created using the WebCenter Portal Framework Application template, the Assets page is available out of the box as part of the runtime administration console (Figure 9–1).

In addition, you can add the Assets page to any page in your application. This is especially useful if you do not want to use the WebCenter Portal Framework Application template, and the runtime administration console is therefore not part of your project.

When you add the Assets page to a page, anyone with view access to the page can view the Assets page and perform view related operations on portal resources, such as viewing resource properties, previewing resources, and so on. For users to be able to perform other operations on portal resources, such as creating new resources, editing resources, or deleting resources, they must have the appropriate permissions on the type of portal resource with which they want to work. These permissions can be provisioned using the Role Manager task flow. For more information, see Section 74.4, "Using the Role Manager Task Flow."

**Tip:** By default, the Administrator role has manage access on all portal resource types available on the Assets page.

To add the Assets page:

1. Open the page to which you want to add the Assets page.
2. In the Resource Palette, expand the **WebCenter Portal - Services Catalog** and then **Task Flows**.
3. Select the **List of Asset 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 Assets page library to the project. Confirm by clicking **Add Library**.
5. The Assets page is added to the page.
6. To enable users to upload and download portal resources at runtime, the `usesUpload` property for the form that contains the Assets page task flow must be set to true:

   `<af:form id="f1" usesUpload="true">`

9.5.1.2 How to Include a Portal Resource on the Assets Page

To include a portal resource on the Assets page, and therefore enable runtime administration of it, you must explicitly identify the resource as a portal resource.

For navigation models and resource catalogs, you have the option of identifying the resource as a portal resource when you first create it by selecting the **Create as Portal Resource** check box. For other resources, for example, page templates and skins, you must identify the resource as a portal resource after you create it.

To include a portal resource on the Assets page:

1. In the Application Navigator, right-click the portal resource that you want to include on the Assets page 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 9–2**), in the **Display Name** field, edit the name if required.

   This is the name that is listed on the Assets page at runtime, so make sure that users are able to easily identify the resource from the display name.

**Figure 9–2  The Create Portal Resource Dialog**
3. The **Resource Type** field identifies the kind of resource you are including on the Assets page. 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 portal resource.
   
   This image is used for page styles only to specify the icon to display in the Create Page dialog.

5. In the **Description** field, enter a description for the portal resource. This description should help users determine if they want to use the resource in their application.
   
   The description is displayed below the portal resource’s name on the Assets page.

6. Provide values for any additional attributes supported by the portal resource.

7. If you want to specify resource-level security for the portal resource, click the **Security** tab. Otherwise, skip to step 11.
   
   Resource-level security enables you to override the application-level security for individual portal resources to specify who can edit and delete the resource at runtime.

8. Select **Override default access policy** (Figure 9–3).

   ![](image)

   **Figure 9–3  Create Portal Resource - Security Tab**

9. Click the **Add permissions to new role** icon (represented by a plus sign) and select the role for which you want to grant permissions.

10. Select the **Manage** check box to grant full access to the portal resource at runtime. Users with this role can edit and delete the portal resource.

   Select the **Update** check box to grant the permission to edit the portal resource at runtime. Users with this role can edit the portal resource, but they cannot delete it.

---

Introduction to Portal Resource Management  9-7
11. Click OK.

The portal resource is exposed on the Assets page and users with the appropriate
privileges can manage the resource at runtime.

9.5.1.3 How to Update Portal Resource Properties

You can update the details of portal resources that are exposed on the Assets page.

To update portal resource properties:

1. In the Application Navigator, right-click the portal 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 9.5.1.2, "How to Include
   a Portal Resource on the Assets Page."

3. When you are done, click OK.

9.5.1.4 How to Remove a Portal Resource from the Assets Page

If you no longer want users to be able to administer a particular portal resource at
runtime, you can remove it from the Assets page.

To remove a portal resource from the Assets page:

1. In the Application Navigator, right-click the portal 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 portal resource from the Assets page.

   The portal resource is removed from the Assets page and can no longer be
   managed at runtime. The portal resource itself is not deleted, so it is still available
   for use within the Portal Framework application.

9.5.1.5 How to Use the Assets Page at Runtime

At runtime, the Assets page provides access to administration tools for the resources
available to the portal. For example, the administrator can create new portal resources
(navigation models, page templates, skins, and so on), edit existing resources, upload
resources exported from other applications, and so on.

For information about how to manage portal resources at runtime using the Assets
page of the runtime administration console, see the "Managing Assets for a Portal
Framework Application" section in Administering Oracle WebCenter Portal.

9.6 Working with Round-Trip Development

If you enable runtime administration of your portal resources, you may want to be
able to pull those resources back into JDeveloper. For example, if a user adds a new
navigation model to a portal at runtime, you may want to further refine it in
JDeveloper. To do this, you must download the portal resource from the deployed
application, using the Assets page. You can then import the portal 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 portal resources in any deployed Portal Framework application that includes the Assets page, including WebCenter Portal.

---

**Note:** Oracle WebCenter Portal provides a JDeveloper project (WebCenterSpacesResources) to help you develop portal resources for WebCenter Portal. Section 55, "Developing Components for WebCenter Portal Using JDeveloper" explains how to use the WebCenterSpacesResources project to build resources for use in WebCenter Portal.

---

The process for round-trip development works as follows:

1. Build Portal Framework application in JDeveloper
2. Deploy application
3. Add or edit portal resources and add content to the application at runtime
4. Download portal resource to EAR file using the Assets page
   See Section 9.6.1, "How to Download a Portal Resource Using the Assets Page"
5. Import EAR file into JDeveloper
   See Section 9.6.2, "How to Import a Portal Resource into JDeveloper"
6. Edit portal resource in JDeveloper
7. Export portal resource to EAR file
   See Section 9.6.3, "How to Export a Portal Resource from JDeveloper"
8. Upload portal resource to deployed application using the Assets page
   See Section 9.6.4, "How to Upload a Resource Using the Assets Page"
9. Repeat from step 3.

In addition, after the deployment of the application, developers can continue to create and edit portal resources in JDeveloper and upload those resources to the deployed application periodically.

Figure 9–4 illustrates the round-trip development process described above.
This section includes the following topics:

- Section 9.6.1, "How to Download a Portal Resource Using the Assets Page"
- Section 9.6.2, "How to Import a Portal Resource into JDeveloper"
- Section 9.6.3, "How to Export a Portal Resource from JDeveloper"
- Section 9.6.4, "How to Upload a Resource Using the Assets Page"
To edit a portal resource in JDeveloper, you first need to use the Assets page in the deployed application to download the portal resource to an EAR file that contains all the metadata for the resource.

For information about how to do this, see the "Downloading an Asset" section in Administering Oracle WebCenter Portal.

### 9.6.2 How to Import a Portal Resource into JDeveloper

When you have the EAR file, you can import it into JDeveloper, where you can then edit the portal resource.

**Note:** You must import the portal resource into a Portal Framework application or an application 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 9–5), in the Import Archive File Name field, enter or browse for the archive file that contains the portal resource that you want to import.
9.6.3 How to Export a Portal Resource from JDeveloper

After editing the portal resource, you must export it to create an EAR file that can be uploaded to the deployed application.

To export 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 9.5.1.2, "How to Include a Portal Resource on the Assets Page."

2. In the Export Portal Resource dialog (Figure 9–6), 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.
3. Click OK.

The export archive contains the files that make up the resource, for example, the JSPX file and the page definition. The archive also contains the `generic-site-resources.xml` file and the registry file with just the information about the exported resource.

---

**Note:** Artifacts used or referenced by portal resources, such as icons and images, are not included in the archive. When you upload the portal resource, you must manage and move dependent artifacts manually. We recommend that you use a folder structure on your content server specifically for portal resource artifacts so that content is easy to identify and move, if required.

If you are managing legacy portal resources that store artifacts in MDS, we recommend that you relocate dependent artifacts to your content server. However, if you do need to move artifacts stored in MDS, you can use the MDS WLST commands `exportMetadata` and `importMetadata`.

---

**9.6.4 How to Upload a Resource Using the Assets Page**

The final step is to take the EAR file of the updated resource and use the Assets page to upload the resource back into the deployed application.

For information about how to do this, see the "Uploading an Asset" section in *Administering Oracle WebCenter Portal*.

**9.7 Working with WebCenter Portal Resources**

WebCenter Portal is a web-based application that offers the very latest technology for building portals, social networking, communication, collaboration, and personal productivity. Out of the box, WebCenter Portal provides various built-in portal resources that users can leverage in their portals. If these built-in portal resources do not meet users' requirements, application administrators can build more within WebCenter Portal. For more information, see the "Creating Assets" section in *Building Portals with Oracle WebCenter Portal*.

**Tip:** In WebCenter Portal, portal resources are referred to as assets.

In some cases, WebCenter Portal may not provide the necessary controls to create a portal resource that provides all the required functionality. In such cases, you can
bring portal resources originally created in WebCenter Portal into JDeveloper for editing, and then upload them back into WebCenter Portal.

In addition, some portal resources used by WebCenter Portal, such as page styles and task flow styles, cannot be created in WebCenter Portal. Instead, these resources must be created in JDeveloper and then exported and uploaded to WebCenter Portal.

Oracle WebCenter Portal provides a special JDeveloper project (WebCenterSpacesResources) to help you develop portal resources for use in WebCenter Portal. For more information, see Section 55.1, "Developing Assets for WebCenter Portal."

When working with WebCenter Portal resources in JDeveloper, you should consider the following:

**Navigation Models**
- All the resources required for a WebCenter Portal navigation model exist within WebCenter Portal. Therefore, you should really only use JDeveloper to restructure WebCenter Portal navigation models and perform minor edits. First, you should create as much of the navigation model in WebCenter Portal 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 navigation model, you should obtain the path for the resource from WebCenter Portal by accessing the Show Properties dialog for the resource. For more information, see the "Viewing Information About an Asset" section in Building Portals with Oracle WebCenter Portal.
- When adding resources to a WebCenter Portal 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 and then enter that value manually as `page://pathToXmlFile`.

**Page Templates**
- 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.
This chapter describes how to use Oracle JDeveloper to create and manage the navigation for your WebCenter Portal Framework applications.

This chapter includes the following topics:

- Section 10.1, "Introduction to Navigation Models"
- Section 10.2, "Creating a Navigation Model"
- Section 10.3, "Editing a Navigation Model"
- Section 10.4, "Selecting the Default Navigation Model"
- Section 10.5, "Filtering the Contents of a Navigation Model"
- Section 10.6, "Deleting a Navigation Model"
- Section 10.7, "Editing the Navigation Registry File"
- Section 10.8, "Editing the Navigation Renderer"
- Section 10.9, "Navigation URL Reserved Keywords"
- Section 10.10, "Tips for Developing Navigation Models"
- Section 10.11, "Visualizing Portal Navigation"
- Section 10.12, "Setting Context Parameters"
- Section 10.13, "Using a Navigation Model to Create a Sitemap"
- Section 10.14, "Switching the Locale Dynamically"

10.1 Introduction to Navigation Models

Typically, a portal is made up of many pages that provide information from various different sources. Users need a way to easily move through all these pages to quickly access the specific information that they need. You provide this path through your portal's pages with your portal navigation.

You can also use your portal navigation to provide access to other resources, such as external pages, content repositories, portlets, and task flows.

A 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)
- External links
Creating a Navigation Model

- Content (individual content items or the results of a content query)
- Other navigation models
- Portlets, task flows, and external applications

For more information, see Section 10.3, "Editing a Navigation Model."

You can also add resources that are dynamically generated:
- Custom Resources
- Components
- Custom Folders
- Custom Content

For more information, see Section 10.3.6, "How to Add Dynamically Generated Resources to a Navigation Model."

**Seeded Navigation Model**

When you create a new Portal 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 10.4, "Selecting the Default Navigation Model."

---

After you have created your navigation model, you can expose it on any page or page template in your portal. For more information, see Section 10.11, "Visualizing Portal Navigation."

**Runtime Management**

Portal Framework applications support the runtime administration of navigation models to help users continue developing a portal even after it has been deployed. With runtime administration, authorized users can manage an application's navigation models, and create new ones, in a browser-based environment, with no requirement to install or understand JDeveloper. For more information, see Section 9.5, "Working with Portal Resources at Runtime."

**Round-Trip Development**

You can bring navigation models that have been created or modified at runtime back into JDeveloper to ensure that any changes are not lost if the portal is redeployed. You can also edit these navigation models to further enhance them if necessary, and upload them back into the deployed portal. For more information, see Section 9.6, "Working with Round-Trip Development."

### 10.2 Creating a Navigation Model

If the seeded navigation model created for your application 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 (typically, /oracle/webcenter/portalapp/navigations) and choose New.

2. In the New Gallery, expand Web Tier, select Portal Framework and then Navigation, and click OK.

3. In the Create Navigation dialog (Figure 10–1), in the File Name field, enter a name for the XML file that represents the navigation model, for example mainNavigation.xml.

4. In the Directory field, enter the full directory path of the location under which to create the navigation model definition XML file.

   Note: In a Portal Framework application, to expose the new navigation model in the Assets page of the runtime administration console, you must create the navigation model under the Application_Root/Portal/public_html/oracle/webcenter/portalapp/navigations directory. By default Portal Framework application includes a separate navigations directory.

5. Select Create as a Portal Resource to enable users with the appropriate permissions to manage the navigation model at runtime in the Assets page of the runtime administration console.

   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 runtime administration console later by right clicking it and choosing Create Portal Resource.

6. Click OK.

   When you create a navigation model in a Portal Framework application, a navigation model definition file (an XML file) is created for the navigation model in the specified directory. The navigation model definition file opens in Design view (Figure 10–2).
The navigation model definition file is initially empty, so you must edit it to design the structure and content of your navigation model. For information, see Section 10.3, "Editing a Navigation Model."

**Tip:** If you do not want to add resources from scratch, as a starting point, you could copy the source code of the default-navigation-model.xml file into the Source view of your new navigation model definition.

If you created the navigation model under the appropriate directory and selected the Create as a Portal Resource option, the navigation model is automatically available for runtime administration. For more information, see Section 9.5, "Working with Portal Resources at Runtime."

If you create your own navigation model in a Portal Framework application, rather than extend the seeded navigation model, it is recommended that you set your navigation model as the default navigation model for the application. This makes it much easier to consume the navigation model in your pages and page templates. For more information, see Section 10.4, "Selecting the Default Navigation Model."

### 10.3 Editing a Navigation Model

To define the content of your navigation model you can edit the seeded navigation model, default-navigation-model.xml, to add nodes for the resources that you want users to be able to access within your application. The seeded navigation model is created automatically when you create an application using the WebCenter Portal Framework Application template.

You can also create your own navigation model and edit that. For more information, see Section 10.2, "Creating a Navigation Model."

You can add nodes 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 10–3).

![The Navigation Model Add New Node Menu](image)

In the Design view for the navigation model you can also modify and remove existing navigation model nodes.

This section includes the following topics:

- Section 10.3.1, "How to Drag and Drop a Resource onto a Navigation Model"
- Section 10.3.2, "How to Add a Page to a Navigation Model"
- Section 10.3.3, "How to Add a Folder to a Navigation Model"
- Section 10.3.4, "How to Add a Separator to a Navigation Model"
- Section 10.3.5, "How to Add Other Resources to a Navigation Model"
- Section 10.3.6, "How to Add Dynamically Generated Resources to a Navigation Model"
- Section 10.3.7, "How to Set Display Options for a Navigation Model or Navigation Model Node"
- Section 10.3.8, "How to Rearrange Nodes in a Navigation Model"
- Section 10.3.9, "How to Show or Hide a Navigation Model or Navigation Model Node"
- Section 10.3.10, "How to Edit a Navigation Model Node"
- Section 10.3.11, "How to Delete a Navigation Model Node"

### 10.3.1 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 of the navigation model definition file:

- **Pages**, from the Application Navigator. You are prompted to choose whether to add just the page itself or to add a page query that includes the page and its subpages.
Editing a Navigation Model

- **URL connections**, from the Application Resources pane or Resource Palette.
- **Content items**, from a WebCenter Content connection in the Application Resources pane or Resource Palette.
- **Portlets**, from the Application Resources pane or Resource Palette.
- **Task flows**, from the Application Resources pane or Resource Palette.
- **External applications**, from the Application Resources pane or Resource Palette.
- **Navigation models**, from the Application Navigator. This embeds one navigation model inside another.

### 10.3.2 How to Add a Page to a Navigation Model

A navigation model is mostly made up of links to the various pages within the application. You can add links to any JSPX page in the application.

To add a page to a navigation model:

1. In the Application Navigator, right-click the navigation model to which you want to add the page, 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 page.
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 more descriptive. The ID must be unique within the navigation model.
   
   The ID is used to provide direct access to a node in the navigation model. For example, if you have a top level node with the ID `home`, and a child node below it with the ID `sales`, then the child node can be accessed using the following URL:
   
   `http://host:port/application-context-root/servlet/home/sales`

   **Note:** If the node is at the top level of the navigation model, the ID must not be one of the navigation URL reserved keywords. For more information, see Section 10.9, "Navigation URL Reserved Keywords."

   **Tip:** If the ID is not unique, or you enter any invalid characters, the border of the field becomes red.

5. From the **Type** drop-down list, select **Page**.
6. The **Factory Class** field is automatically populated with the appropriate value for pages and cannot be edited.
7. In the **URL** field, enter or browse for the URL to access the page. The URL should use the following format:

    `page://pathToPage/pageName`

    For example:

    `page://oracle/webcenter/portalapp/pages/myPage.jspx`
Tip: If you are not sure of the path to the page, click Browse Files to locate the page in the application.

8. Select Render URL in Page Template, to render the page in the content area of the application using the specified page template.

Select Redirect to URL, to cause the page to issue a client-side redirect to the defined URL.

9. In the Visible field, specify whether the page is listed in the navigation model when it is rendered at runtime. The default value is the EL expression #{true}, which means the page is listed for all users at all times. Enter false or #{false} to hide the page from all users at all times, or enter an EL expression to specify the conditions under which the page is listed. If the field is empty, the value defaults to true.

---

Note: Nodes in a navigation model are automatically filtered based on security. If a user does not have access to a page, it is not displayed in the rendered navigation model.

10. In the URL Attributes section, specify any desired display options for the page. For more information, see Section 10.3.7, "How to Set Display Options for a Navigation Model or Navigation Model Node."

11. In the URL Parameters section, enter values, as desired, for any parameters supported by the selected page.

Click the Add icon to create a row in the Parameters table.

For example, you can specify the size of the inline frame used to display the page by adding the url_renderer_width and url_renderer_height parameters and specifying a value in pixels.

12. Save the navigation model definition file.

10.3.3 How to Add a Folder to a Navigation Model

Navigation models can be organized hierarchically to create subsets of nodes in your navigation model or to group similar nodes together. This hierarchy can be collapsed at runtime to save space if you have a lot of nodes in your navigation model.

You can place a node under any other node in a navigation model to create this hierarchy. If you have a collection of nodes that do not have a node that can act as their parent, you can create a folder. A folder does not point to an underlying resource; it serves only to contain other navigation model nodes.

To add a folder to a navigation model:

1. In the Application Navigator, right-click the navigation model to which you want to add the folder, 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 folder.

3. Click the Add new node icon and choose 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 more descriptive. The ID must be unique within the navigation model.
The ID is used to provide direct access to a node in the navigation model. For example, if you have a top level node with the ID home, and a child node below it with the ID sales, then the child node can be accessed using the following URL:

http://host:port/application-context-root/servlet/home/sales

**Note:** If the node is at the top level of the navigation model, the ID must not be one of the navigation URL reserved keywords. For more information, see Section 10.9, "Navigation URL Reserved Keywords."

**Tip:** If the ID is not unique, or you enter any invalid characters in the ID, the border of the field becomes red.

5. In the **Visible** field, specify whether the folder is displayed when the navigation model is rendered at runtime. The default value is the EL expression #{true}, which means the folder is displayed to all users at all times. Enter false or #{false} to hide the folder from all users at all times, or enter an EL expression to specify the conditions under which the folder is displayed. If the field is empty, the value defaults to true.

6. In the **Folder Attributes** section, specify any desired display options for the folder. For more information, see Section 10.3.7, "How to Set Display Options for a Navigation Model or Navigation Model Node."

7. In the **Folder Parameters** section, enter values, as desired, for any parameters supported by the folder.

   Click the **Add** icon to create a row in the Parameters table.

8. Save the navigation model definition file.

### 10.3.4 How to Add a Separator to a Navigation Model

Separators visually divide a navigation model into distinct groupings of nodes. Separators enable you to visually break up the nodes in the navigation model when it is rendered at runtime, making it easier to locate nodes.

To add a separator to a navigation model:

1. In the Application Navigator, right-click the navigation model to which you want to add the separator, 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 separator.

3. Click the **Add new node** icon and choose **Separator**.

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 more descriptive. The ID must be unique within the navigation model.

   The ID is used to provide direct access to a node in the navigation model. For example, if you have a top level node with the ID home, and a child node below it with the ID sales, then the child node can be accessed using the following URL:

   http://host:port/application-context-root/servlet/home/sales
5. In the **Visible** field, specify whether the separator is displayed when the navigation model is rendered at runtime. The default value is the EL expression `#{true}`, which means the separator is displayed to all users at all times. Enter `false` or `#{false}` to hide the separator from all users at all times, or enter an EL expression to specify the conditions under which the separator is displayed. If the field is empty, the value defaults to `true`.

6. Save the navigation model definition file.

### 10.3.5 How to Add Other Resources to a Navigation Model

You can add other resources to your navigation model, including documents, external web pages, portlets, and task flows.

To add other resources to a 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, then choose the type of resource that you want to add:
   - **Content Item** to add a document
   - **Content Query** to add a collection of documents that meet specific query criteria
   - **Link** to add a portlet, task flow, external application, or web page
   - **Pages Query** to add a list of pages
   - **Navigation Reference** to embed another navigation model

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 more descriptive. The ID must be unique within the navigation model.

   The ID is used to provide direct access to a node in the navigation model. For example, if you have a top level node with the ID `home`, and a child node below it with the ID `sales`, then the child node can be accessed using the following URL:

   ```
   http://host:port/application-context-root/servlet/home/sales
   ```

   **Note:** If the node is at the top level of the navigation model, the ID must not be one of the navigation URL reserved keywords. For more information, see Section 10.9, "Navigation URL Reserved Keywords."
Tip: If the ID is not unique, or you enter any invalid characters in the ID, the border of the field becomes red.

5. In the Visible field, specify whether the node is displayed when the navigation model is rendered at runtime. The default value is the EL expression #{true}, which means the node is displayed to all users at all times. Enter false or #{false} to hide the node from all users at all times, or enter an EL expression to specify the conditions under which the node is displayed. If the field is empty, the value defaults to true.

6. The remaining fields are specific to the resource type. See Table 10–1 for details.

<table>
<thead>
<tr>
<th>Table 10–1</th>
<th>Fields for Navigation Model Resources</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field</td>
<td>Applies to</td>
</tr>
<tr>
<td>Factory Class</td>
<td>Link</td>
</tr>
<tr>
<td></td>
<td>Content Item</td>
</tr>
<tr>
<td>Insert Folder Contents</td>
<td>Content Query</td>
</tr>
<tr>
<td></td>
<td>Page Query</td>
</tr>
<tr>
<td>Page Style</td>
<td>Page Query</td>
</tr>
<tr>
<td>Page Template</td>
<td>Page Query</td>
</tr>
<tr>
<td>Page Visibility</td>
<td>Page Query</td>
</tr>
<tr>
<td>Path</td>
<td>Page Query</td>
</tr>
<tr>
<td></td>
<td>Navigation Reference</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Query</td>
<td>Content Query</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Redirect to URL</td>
<td>Link</td>
</tr>
<tr>
<td></td>
<td>Content Item</td>
</tr>
<tr>
<td>Render URL in Page Template</td>
<td>Link</td>
</tr>
<tr>
<td></td>
<td>Content Item</td>
</tr>
</tbody>
</table>
Table 10–1  (Cont.) Fields for Navigation Model Resources

<table>
<thead>
<tr>
<th>Field</th>
<th>Applies to</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Repository</td>
<td>Content Query</td>
<td>Enter or browse for the content repository whose content you want to search. Note: If you enter the repository name manually (that is, without using drag and drop or the Choose a Resource dialog), and if the connection for the repository is available only in the Resource Palette, then ensure that you first add the connection to the application (in the Application Resources pane).</td>
</tr>
<tr>
<td>Scope</td>
<td>Page Query</td>
<td>This field is not applicable to Portal Framework applications (the scope is always the current application).</td>
</tr>
<tr>
<td>Type</td>
<td>Link, Content Item</td>
<td>Select the type of link that you want to add to the navigation model: External Link, External Application, Page, Taskflow, Portlet, Content.</td>
</tr>
<tr>
<td>URL</td>
<td>Link, Content Item</td>
<td>Enter the location of the resource. If you do not know the location, click the Browse Files icon to browse for available resources. Table 10–2 provides the URL format to use for the different resource types. Note: If you enter the URL for a Portlet, External Application, or Content resource manually (that is, without using drag and drop or the Choose a Resource dialog), and if the connection for the resource is available only in the Resource Palette, then ensure that you first add the connection to the application (in the Application Resources pane).</td>
</tr>
</tbody>
</table>

Table 10–2 provides the URL format to use for the different resource types.

Table 10–2  Format for URL for Different Resource Types

<table>
<thead>
<tr>
<th>Resource Type</th>
<th>Location in Application</th>
<th>URL Formats</th>
</tr>
</thead>
<tbody>
<tr>
<td>External Link</td>
<td>None</td>
<td>Absolute or relative path to the web page to which you want to link. 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 10.11, &quot;Visualizing Portal Navigation.&quot;</td>
</tr>
<tr>
<td>External Application</td>
<td>External application connection in the Application Resources pane</td>
<td>extapp://externalApplicationId</td>
</tr>
<tr>
<td>Page</td>
<td>Application Navigator</td>
<td>page://pathToPage/pageName</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example: page://oracle/webcenter/portalapp/pages/myPage.jspx</td>
</tr>
</tbody>
</table>
7. In the resource's Attributes section, specify any desired display options for the resource. For more information, see Section 10.3.7, "How to Set Display Options for a Navigation Model or Navigation Model Node."

8. In the resource's Parameters section, enter values, as desired, for any parameters supported by the resource.

   Click the Add icon to create a row in the Parameters table.

   For links you can specify the size of the inline frame used to display the target of the link by adding the `url_renderer_width` and `url_renderer_height` parameters and specifying a value in pixels.

   If you want to use a Content Presenter display template to display a content item or the results of a content query, in the Parameters section, click the Add icon, and choose `templateView`.

   In the Value field for the `templateView` parameter, enter the display template view ID to use for rendering the content, for example:

   - For a single content item:
     `oracle.webcenter.content.templates.default.detail`

   - For a content query:
     `oracle.webcenter.content.templates.default.list.simple`

Table 10–2 (Cont.) Format for URL for Different Resource Types

<table>
<thead>
<tr>
<th>Resource Type</th>
<th>Location in Application</th>
<th>URL Formats</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task Flow from an ADF Library</td>
<td>Resource Palette</td>
<td><code>taskflow://pathToTaskFlow/taskFlowDefinitionFileName#taskFlowId</code></td>
</tr>
<tr>
<td>Note: Navigating directly to a task flow in this way enables you to set the parameters of the task flow. If you need additional control over the task flow, add it to a page and then add the page to the navigation model.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Task Flow Within the Application</td>
<td>Application Navigator</td>
<td><code>taskflow://PathToTaskFlow/taskFlowDefinitionFileName#taskFlowId</code></td>
</tr>
<tr>
<td>Note: Navigating directly to a task flow in this way enables you to set the parameters of the task flow. If you need additional control over the task flow, add it to a page and then add the page to the navigation model.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Portlet</td>
<td>Portlet producer connection in the Application Resources pane or Resource Palette</td>
<td><code>portlet://producerId/portletId</code></td>
</tr>
<tr>
<td>Note: If you provide navigation directly to a portlet in this way, then when users navigate to the portlet they will not be able to customize it. If you want users to be able to customize the portlet, then add it to a page to create a portlet instance, which can be customized, and then add that page to the navigation model.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Content</td>
<td>Content repository connection in the Application Resources pane or Resource Palette</td>
<td><code>content://contentConnectionId/documentId</code></td>
</tr>
</tbody>
</table>
10.3.6 How to Add Dynamically Generated Resources to a Navigation Model

As well as adding known resources to your navigation model, you can also add resources that are dynamically generated at runtime.

- **Custom resources** are links to third party resources.
- **Custom components** are resources, such as simple ADF Faces components, compound objects containing two or more components, or JSF Verbatim tags that let you add arbitrary HTML content inside them.
- **Custom folders** are configured to display their 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.
- **Custom content providers** dynamically generate zero or more resources at runtime. They are very similar to custom folders, except that, when you add a custom folder, 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 dynamically generated resources to a 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, then choose the type of resource that you want to add:
   - **Link** to add a custom resource. You must also select **Other** from the **Type** drop-down list.
   - **Component** to add a custom component
   - **Custom Folder** to add a custom folder
   - **Custom Content** to add a custom content provider
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 more descriptive. The ID must be unique within the navigation model.

The ID is used to provide direct access to a node in the navigation model. For example, if you have a top level node with the ID `home`, and a child node below it with the ID `sales`, then the child node can be accessed using the following URL:

http://host:port/application-context-root/servlet/home/sales
5. In the **Visible** field, specify whether the node is displayed when the navigation model is rendered at runtime. The default value is the EL expression `#{true}`, which means the node is displayed to all users at all times. Enter `false` or `#{false}` to hide the node from all users at all times, or enter an EL expression to specify the conditions under which the node is displayed. If the field is empty, the value defaults to `true`.

6. The remaining fields are specific to the resource type. See **Table 10–3** for details.

**Note:** If the node is at the top level of the navigation model, the ID must not be one of the navigation URL reserved keywords. For more information, see Section 10.9, "Navigation URL Reserved Keywords."

**Tip:** If the ID is not unique, or you enter any invalid characters in the ID, the border of the field becomes red.

**Table 10–3** Fields for Custom Navigation Model Resources

<table>
<thead>
<tr>
<th>Field</th>
<th>Applies to</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Component Factory</td>
<td>Custom Component</td>
<td>Enter or browse for the Java class that implements the <code>oracle.adf.rc.component.ComponentFactory</code> interface and creates an instance of the component based on a set of parameters, for example, <code>oracle.adf.rc.component.XmlComponentFactory</code>. As you type the name, the field autocompletes with valid factory classes from the classpath. For more information, see Section C.4, &quot;Factory Classes Available for Adding Dynamic Resources to the Catalog.&quot;</td>
</tr>
<tr>
<td>Content Provider</td>
<td>Custom Content</td>
<td>Enter or browse for the Java class that implements the <code>oracle.adf.rc.spi.plugin.catalog.CustomContentProviderV2</code> interface. For more information, see Section C.4, &quot;Factory Classes Available for Adding Dynamic Resources to the Catalog.&quot;</td>
</tr>
<tr>
<td>Factory Class</td>
<td>Custom Resource</td>
<td>Enter or browse for the <code>URLResourceFactory</code> implementation class that can resolve the URL for the custom resource.</td>
</tr>
<tr>
<td>Initial Context Factory</td>
<td>Custom Folder</td>
<td>Enter or browse for the Java class that implements the <code>javax.naming.spi.InitialContextFactory</code> interface and generates the folder based on a set of parameters. For more information, see Section C.4, &quot;Factory Classes Available for Adding Dynamic Resources to the Catalog.&quot;</td>
</tr>
<tr>
<td>Insert Folder Contents</td>
<td>Custom Folder</td>
<td>Select to display the contents of the custom folder directly in the rendered navigation model rather than displaying them under the folder.</td>
</tr>
<tr>
<td>Path</td>
<td>Custom Folder</td>
<td>Enter the MDS path for the custom folder.</td>
</tr>
</tbody>
</table>
7. In the resource’s Attributes section, specify any desired display options for the resource. For more information, see Section 10.3.7, “How to Set Display Options for a Navigation Model or Navigation Model Node.”

<table>
<thead>
<tr>
<th>Field</th>
<th>Applies to</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Redirect to URL</td>
<td>Custom Resource</td>
<td>Select to cause the page to issue a client-side redirect to the defined URL.</td>
</tr>
<tr>
<td>Render URL in Page Template</td>
<td>Custom Resource</td>
<td>Select to render the resource in the current page, using the specified page template.</td>
</tr>
<tr>
<td>Type</td>
<td>Custom Resource</td>
<td>Select Other.</td>
</tr>
<tr>
<td>URL</td>
<td>Custom Resource</td>
<td>Enter the URL to access the resource.</td>
</tr>
</tbody>
</table>

8. In the resource’s Parameters section:

   - For custom resources, enter values, as desired, for any parameters supported by the custom resource.
   - For custom folders, add the parameters that were defined while implementing the factory class. You can bind these parameters to other artifacts in the application.
   - For custom content providers, define the parameters on the custom content provider. The parameters you define here are passed to and interpreted by the CustomContentProviderV2 implementation.

   Click the Add icon to create a row in the Parameters table.

9. Save the navigation model definition file.

10.3.7 How to Set Display Options for a Navigation Model or Navigation Model Node

You can specify various display options for a navigation model or any of its nodes to determine their appearance and behavior when the navigation model is rendered at runtime. The display options available depend on the type of the resource.

To set display options for a navigation model node:

1. In the Application Navigator, right-click the navigation model and choose Open.

2. In the Design view for the navigation model, in the Navigation column on the left side, select the node for which you want to set display options.

   To set display options for the navigation model, select the root node.

3. In the resource’s Attributes panel, click the Add icon and choose the attribute that you want to set. Table 10–4 lists the available attributes.
From the Value Type drop-down 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.

### 10.3.8 How to Rearrange Nodes in a Navigation Model

You can move nodes around within a navigation model after you have added them.

You can rearrange nodes in the following ways:
- Drag and drop a node from one location to another.
  
  To move a node within the navigation model, select the node and drag it to the desired position. To indent a node (child) under another (parent), select the child node and drag it over the top of the parent node.
Use the **Change Parent** context menu option to move a node to a different location.

To rearrange a node using the Change Parent option:

1. In the Application Navigator, right-click the navigation model and choose **Open**.
2. In the Design view for the navigation model, in the Navigation column on the left side, right-click the node and choose **Change Parent** from the context menu.
3. In the Choose Parent dialog (**Figure 10–4**), select the parent node in which to include the node and click **OK**.

**Figure 10–4  The Change Parent Dialog**

![Choose Parent Dialog](image)

**Note:** Any node within a navigation model can be the parent of another node.

4. Save the navigation model definition file.

### 10.3.9 How to Show or Hide a Navigation Model or Navigation Model Node

You can specify whether or not a navigation model or one of its nodes is visible at runtime.
To show or hide a navigation model or navigation model node:

1. In the Application Navigator, right-click the navigation model and choose Open.
2. In the Design view for the navigation model, in the Navigation column on the left side, select the node for which you want to set the visibility.

To set the visibility of the navigation model, select the root node.

3. In the Visible field, specify whether the navigation model or node is available for use in the application. This field takes an EL value. The default is #{true}, which means the navigation model or node is visible. Enter false or #{false} to hide the navigation model or node, or enter an EL expression to specify the conditions under which the navigation model or node is available. If the field is empty, the value defaults to true.
4. Save the navigation model definition file.

10.3.10 How to Edit a Navigation Model Node

To edit a navigation model node, select it in the Navigation section of the navigation model 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 navigation model definition file. For details about a resource’s attributes and parameters, see the appropriate section earlier in this chapter.

10.3.11 How to Delete a Navigation Model Node

To delete a resource, select it in the Navigation section of the navigation model definition file and click the Remove selected node icon (Figure 10–5).

10.4 Selecting the Default Navigation Model

Every Portal Framework application defines a default navigation model that provides a convenient way to identify the navigation model that should be used by default by the application. You can then reference this default navigation model without having to specify 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 Portal Framework application, the seeded navigation model, default-navigation-model.xml, is set as the application’s default navigation model. If you subsequently create your own navigation model to use for the main navigation path through your application, rather than having to explicitly reference this
Filtering the Contents of a Navigation Model

You can set the default navigation model for a Portal Framework 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 application, for example:
   
   ```xml
   value="/oracle/webcenter/portalapp/navigations/myNavigationModel.xml"
   ```

   Example 10–1 shows an example of the complete preference element.

**Example 10–1  The Default Navigation Model ADF Preference**

```xml
<preference id="oracle.webcenter.portalapp.navigation.model"
desc="Default Navigation Model"
value="/oracle/webcenter/portalapp/navigations/myNavigationModel.xml"
resourceType="navigation" display="true"/>
```

5. Save the adf-config.xml file.

### 10.5 Filtering the Contents of a Navigation Model

You can use a filter to determine which nodes to include in the navigation model, based on specific criteria, when it is rendered at runtime.

**Tip:** You can conditionally hide individual nodes in the navigation model by specifying an EL expression in the Visible field for the node. For more information, see Section 10.3.9, "How to Show or Hide a Navigation Model or Navigation Model Node."

To filter the contents of a navigation model:

1. In the Application view, right-click the navigation model and choose Open.

2. In the Design view for the navigation model, in the Navigation panel on the left side, select the node that represents the navigation model (the root node).

3. In the Navigation Filter field, enter the name of a Java class to use to filter out selected nodes in the navigation model based on specific criteria.

   The Java class should implement the oracle.adf.rc.spi.plugin.catalog.CatalogDefinitionFilter interface.
Nodes are automatically filtered based on security. If a user does not have access to a particular resource, the node for that resource is not displayed when the navigation model is rendered at runtime.

4.  Save your changes.

### 10.6 Deleting a Navigation Model

If a navigation model is no longer required within your application, you can delete it.

---

**Note:** If you delete an application’s default navigation model, you must make sure to change the default navigation model to a valid navigation model. For more information, see Section 10.4, "Selecting the Default Navigation Model."

---

To delete a navigation model, right-click the navigation model in the Application Navigator and choose **Delete** from the context menu. For more information, see Section 9.4, "Deleting Portal Resources."

### 10.7 Editing 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 Portal Framework application contains one navigation registry file, `navigation-registry.xml`, which is created when you create the application. You can edit this file, for example, to remove the Navigation Reference resources so that users cannot embed other navigation models at runtime.

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 14.3, "Editing a 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

   **Note:** If you delete an application’s default navigation model, you must make sure to change the default navigation model to a valid navigation model. For more information, see Section 10.4, "Selecting the Default Navigation Model."
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.

10.8 Editing the Navigation Renderer

The navigation renderer is a JSPX page that renders non-page resources when accessed through a navigation model, including:

- Portlets
- Content
- External links
- External applications
- Task flows

You can edit the navigation renderer page to change how these types of resources are rendered. 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.

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 10.4, "Selecting the Default Navigation Model."

The navigation renderer page is hidden by default, so you must first show it in your application before you can edit it.

To edit the navigation renderer:

1. In the Application Navigator, right-click the application project and choose Project Properties.
2. In the Project Properties dialog, 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:
6. In the Application Navigator, 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.

10.9 Navigation URL Reserved Keywords
You can use specific reserved keywords to provide direct access to nodes within a navigation model. These reserved keywords are:

- `wcnav_defaultSelection`—Use to access the first navigable node of a navigation model
- `wcnav_currentSelection`—Use to access the currently selected node of a navigation model
- `wcnav_externalId`—Use to access the node within a navigation model with the specified external ID
- `wcnav_title`—Use to access the node within a navigation model with the specified title

You can use these keywords in an `af:goLink` component in a page or page template (Example 10–2) or in the URL of an External Link navigation item in a navigation model (Example 10–3).

**Example 10–2  Providing Direct Access to a Navigation Model Node from a goLink Component**

```xml
<af:goLink id="pt_glink1" text="Navigation Item"
            destination="/faces/wcnav_defaultSelection"/>
```

**Example 10–3  Providing Direct Access to Navigation Model Node from an External Link Navigation Item**

```xml
<url visible="#{true}" factoryClass="oracle.webcenter.portalframework.sitestructure.rc.UrlResourceFactory"
      id="externalLink"
      url="/faces/wcnav_externalId/myNavigationItem">
</url>
```

By default these URLs provide access to nodes within the default navigation model. 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/
s7f466cab_f622_4b68_a83e_b7eaf28b52ec/navigation/gsr0271c712_721a_4565_9f0e_75578a7093b/
myProjectNavigationModel
```

Depending on where you are using the link, you may need to encode the URL parameters, for example, if the external ID contains spaces or special characters you would specify the URL as `/faces/wcnav_externalId/my%2Fnav%2Fitem`. 

---

10-22   Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper
10.10 Tips for Developing Navigation Models

When creating a navigation model for your Portal Framework application, consider the following:

- Try to use only one main navigation model. Using multiple navigation models limits the control of friendly URLs.
- Ensure that each section of your Portal Framework application has a friendly subroot name. For example, place all footer links under a Footer folder.
- Include all links that take users to another part of the Portal Framework application, including header and footer links, in the navigation model.
- Use external IDs to surface persistent links in the application so that external applications can reference them without breaking when navigation model nodes are moved.
- If a page defines parameters, you can set these parameters in the navigation model node to dynamically control how the page is rendered.
- Use page links as often as possible. For example, if you want to include a contact link, create a page to display the link and then include that page in the navigation model.
- Include all pages used within the portal as page links in the navigation model.

For more information, see the "Oracle WebCenter & ADF Architecture Team" blog.

10.11 Visualizing Portal Navigation

The navigation visualization determines how the navigation appears in the application. 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.

Oracle WebCenter Portal provides several built-in navigation task flows in the WebCenter Portal - Services Catalog of the resource catalog that you can add to pages or page templates to visualize the navigation. These built-in navigation task flows provide a quick way to test your navigation model, but they provide a fairly simple navigation visualization. For more information about these built-in navigation task flows, see the "Adding Navigation to a Page Template" section in Building Portals with Oracle WebCenter Portal.

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 Oracle WebCenter Portal navigation APIs.
Oracle 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 nodes.
- **REST APIs**—Use standard HTTP methods to point to the navigation model and navigation nodes, 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.

---

**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:

```xml
<parent-action id="wcnav_parentAction">
  <root-outcome>wcnav_outcome</root-outcome>
</parent-action>
```

This section includes the following topics:

- Section 10.11.1, "Using the Navigation Expression Language APIs"
- Section 10.11.2, "Using the Navigation REST APIs"
- Section 10.11.3, "Tips for Visualizing Portal Navigation"

### 10.11.1 Using the Navigation Expression Language APIs

Oracle 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.7, "ELs Related to Navigation."

This section includes the following topics:

- Section 10.11.1.1, "Introduction to the Navigation Context"
Section 10.11.2, "Introduction to the Navigation Runtime Model"

Section 10.11.3, "Introduction to the Navigation Resource"

Section 10.11.4, "How to Render the Navigation Model as a List of Links"

Section 10.11.5, "How to Render the Navigation Model as a Tree"

Section 10.11.6, "How to Render the Navigation Model as a Three-Level Menu"

Section 10.11.7, "How to Render the Navigation Model as Breadcrumbs"

Section 10.11.8, "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 a Portal Framework application.

---

10.11.1.1 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 nodes.

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 10.4, "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 or 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 10–4** creates a simple tree UI using the Navigation Context to access the current navigation model and render the tree using `<af:outputText/>`.

**Example 10–5** 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`.

### 10.11.1.2 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.
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 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 10–6 renders the current navigation model’s menu model as a breadcrumb.
Example 10–6  Rendering the Navigation Model as a Menu Model

Example 10–7 produces a Sitemap for the application based on the default navigation model.

Example 10–7  Producing a Sitemap

For more information about Sitemaps, see Section 10.13, "Using a Navigation Model to Create a Sitemap."

10.11.1.3 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 at runtime.
  - **AccessKey**—A key mnemonic (single character) that users can enter to access the node without using the mouse.
  - **Description**—A description of the node.
  - **Subject**—Keywords to facilitate searching of the node.
  - **IconURI**—An icon to visually represent the node. This is displayed next to the Title when the navigation model is rendered at runtime.
  - **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 node 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.7, "ELs Related to Navigation."

**Examples**

**Example 10–8** renders an ADF commandImageLink component with various attributes.

**Example 10–8  Rendering a commandImageLink with Attributes**

```xml
<af:commandImageLink text="#{node.title}" id="cil1"
  actionListener="#{navigationContext.processAction}"
  shortDesc="#{node.attributes['ToolTip']}
  accessKey="#{node.attributes['AccessKey']}"
  inlineStyle="#{node.selected ? 'font-weight:bold;' : ''}"
</af:commandImageLink>
```

**Example 10–9** accesses values passed from a node to the corresponding page through parameters.

**Example 10–9  Passing Node Values to Page Parameters**

```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 10–10** 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 10–10  Conditionally Displaying Navigation Nodes**

```af:breadCrumbs id="bc1"
  var="node"
  value="#{navigationContext.currentNavigationModel.defaultMenuModel}"
</af:breadCrumbs>
```

```af:switcher facetName="#{node.navigable}" id="swnl"
  <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>
```

**10.11.1.4 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 10–11** 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 10–11  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="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:commandImageLink>
      <af:iterator id="i2" value="#{node.children}" var="node2">
        <af:panelList id="pl1">
          <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>
        </af:panelList>
      </af:iterator>
    </af:panelGroupLayout>
  </af:iterator>
</af:panelGroupLayout>
```
10.11.1.5 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 or page template, you can bind the ADF Faces tree navigation component to your navigation model. **Example 10–12** shows how to do this. In the example, the second level of the navigation hierarchy is collapsed by default, but an icon enables users to expand a node to show its children.

**Example 10–12  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}"/>
  </f:facet>
</af:tree>
<af:spacer id="sp3" height="20px"/>
</af:panelGroupLayout>
```

10.11.1.6 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 10–13** 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 10–13  Navigation as a Menu**

```af:panelGroupLayout id="pgl1" layout="vertical">
<af:spacer id="sp1" height="20px"/>
<af:menuBar id="mb1">
```
10.11.1.7 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 or page template, you can bind the ADF Faces breadCrumbs navigation component to your navigation model. **Example 10–14** shows how to do this.

**Example 10–14 Navigation as Breadcrumbs**

```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}"
      >
      </af:commandNavigationItem>
  </f:facet>
</af:breadCrumbs>
<af:Spacer id="sp2" height="20px"/>
<af:panelGroupLayout>```

10.11.1.8 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 10–15** 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.

**Tip:** For the c:set tag to work correctly in the following examples, you must include the JSTL library by adding the following to the jsp:root tag:

```xmlns:c="http://java.sun.com/jsp/jstl/core"```  

**Example 10–15 Master-Detail Navigation Using a Single Navigation Model**

```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"]}
  .defaultMenuModel["includeStartNode=false"]">
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 10–16 shows how.

**Example 10–16 Master-Detail Navigation Using Multiple Navigation Models**

```xml
<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']
      .defaultMenuModel[1]}">
    <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"/>
  <!-- Detail -->
  <af:tree id="tree1" var="node2" initiallyExpanded="true"
    value="#{navigationContext.navigationModel[ 'modelPath=/oracle/webcenter/portalapp/navigations/master-detail']
      .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>
```
10.11.2 Using the Navigation REST APIs

Oracle WebCenter Portal provides REST APIs to access the navigation model of your application. Use the navigation REST APIs to access the interface used to visualize 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 Oracle WebCenter Portal iPhone App uses REST APIs to interact with a Portal Framework application.

For an introduction to the REST APIs, see Chapter 53, "Using Oracle WebCenter Portal REST APIs."

To access the navigation REST APIs, construct a URL as follows:

http://host:port/<ApplicationContextRoot>/api/navigations/default-navigation-model

where host and port are the hostname and port number for the server where the portal is running, and <ApplicationContextRoot> is the context root for the portal application. For example:


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_4cb4_4bb8_bbee_296c916a23ed/navigations/default-navigation
Using this information, you can start to retrieve information about specific navigation models and the nodes within them, using the URIs listed in Table 10–5. The URIs shown in Table 10–5 can be appended to the http://host:port/<ApplicationContextRoot> part of the URL to access the desired REST endpoint.

<table>
<thead>
<tr>
<th>REST URI</th>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>.../api/navigations</td>
<td>All navigation models in the application.</td>
</tr>
<tr>
<td>.../api/navigations/default-navigation</td>
<td>The properties of the default navigation model.</td>
</tr>
<tr>
<td>.../api/navigations/pathToModel/modelId</td>
<td>The properties of the specified navigation model (modelId).</td>
</tr>
<tr>
<td>.../api/navigations/pathToModel/modelId/resources</td>
<td>All root resources under the specified navigation model (modelId).</td>
</tr>
<tr>
<td>.../api/navigations/pathToModel/modelId/resources/pathToResource/resourceId</td>
<td>The specified resource (resourceId) from the specified navigation model (modelId).</td>
</tr>
<tr>
<td>.../api/navigations/pathToModel/modelId/resources/pathToResource/resourceId/children</td>
<td>A navigation tree listing all the children of the specified resource (resourceId) from the specified navigation model (modelId).</td>
</tr>
<tr>
<td>.../api/navigations/pathToModel/modelId/resources/pathToResource/resourceId/children?depth=dep</td>
<td>A navigation tree listing the children of the specified resource (resourceId) from the specified navigation model (modelId) up to a specified level (dep).</td>
</tr>
</tbody>
</table>

Each navigable node has a prettyURL element that you can use to drill down further in the navigation model hierarchy.

10.11.3 Tips for Visualizing Portal Navigation

When visualizing portal navigation model, consider the following:

- Use ADF Faces layout components to organize your page or page template, rather than HTML div and span tags. For more information, see the "Organizing Content on Web Pages" chapter in the Web User Interface Developer’s Guide for Oracle Application Development Framework.

- If your Portal Framework application is public facing, or you want to enable search engine optimization, add navigation visualization using the af:goLink component in conjunction with the goLinkPrettyUrl method. You can also use other variations, such as goButton, goImageLink, and gMenuItem.

For example:
Manage styling, decorative images, and geometry using skins.

Select the page template using the default page template setting in adf-config.xml or using the Render URL in Page Template option for individual navigation links.

If you use the af:commandLink component, or one of its variations such as commandButton; commandMenuItem; and so on, to visualize your portal navigation, you must use the following web.xml parameters to force the ADF framework to redirect instead of using PPR navigation:

oracle.adf.view.rich.pprNavigation.OPTIONS
oracle.webcenter.navigationframework.REDIRECT_OPTIONS

For more information, see Section 10.12, "Setting Context Parameters."

For more information, see the "Oracle WebCenter & ADF Architecture Team" blog.

10.12 Setting Context Parameters

If you want to use the af:CommandLink component to visualize your portal navigation, you must set the oracle.adf.view.rich.pprNavigation.OPTIONS context parameter in the web.xml file to force the ADF framework to redirect instead of using partial page refresh (PPR).

If your application does not use PPR, the URL in the browser changes whenever a user navigates to a node 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 context parameters:

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.adf.view.rich.pprNavigation.OPTIONS.
5. In the Value field, enter off.

This adds the following to the web.xml file:

```xml
<context-param>
    <param-name>oracle.adf.view.rich.pprNavigation.OPTIONS</param-name>
    <param-value>off</param-value>
</context-param>
```
6. In the Name field, enter `oracle.webcenter.navigationframework.REDIRECT_OPTIONS`.

7. In the Value field, enter `toPrettyURL`.

   This adds the following to the `web.xml` file:
   ```xml
   <context-param>
     <param-name>oracle.webcenter.navigationframework.REDIRECT_OPTIONS</param-name>
     <param-value>toPrettyURL</param-value>
   </context-param>
   ```

8. Save the `web.xml` file.

### 10.13 Using a Navigation Model 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/

Portal Framework 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, such as Google's googlebot or Microsoft's bingbot, the HTML is returned for the request instead of the JavaScript.

You can create a Sitemap for your application based on a navigation model, as shown in Example 10–17.

**Example 10–17 Creating a Sitemap**

```xml
<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>
```

### 10.14 Switching the Locale Dynamically

The WebCenter Navigation Model supports localization of certain textual strings such as Title. The setting of the resource bundle based strings is configured through the Navigation Model editor. For more information see, Section 14.3.9, "How to Set Display Options for a Resource Catalog or Resource."

Once the strings have been defined and the application configured with the supporting Locales, you can view the changes when switching the browser language.
Switching the Locale Dynamically

settings. This is easy to implement, however, if the requirement is to be able to dynamically switch the Locale through a user interaction such as a drop-down list, then you must create custom code to support this requirement. In this section, one possible solution for this requirement is presented. This section assumes that the face-config.xml file has already been modified to support the different Locales for the application. For more information see, Chapter 75, "Building Multilanguage Portals."

Example Solution

This example solution contains an ADF af:selectOneChoice (SOC) component, which will be responsible for enabling the user interaction for the dynamic switching of the Locale. The SOC will be wired to a managed bean that will serve two important functions: one function is to store the users Locale selection, and the other function is to provide the valueChangeEvent listener method to enable the setting of the new Locale. In this example, the new Locale is stored in a variable and accessed by the valueChangeEvent.

Locale preferredLocale

public void listChanged(ValueChangeEvent valueChangeEvent) {
    String locale = valueChangeEvent.getNewValue().toString();
    changeLocale(locale);
    setCurrentLocale(locale);
    setPreferredLocale(FacesContext.getCurrentInstance().getViewRoot().getLocale());
}

This method also calls the changeLocale method, which is responsible for updating the current view:

private void changeLocale(String lang) {
    Locale newLocale = new Locale(lang);
    FacesContext fctx = FacesContext.getCurrentInstance();
    fctx.getViewRoot().setLocale(newLocale);
    resetPage();
}

This method then finishes the Locale change by calling resetPage():

public void resetPage() {
    FacesContext context = FacesContext.getCurrentInstance();
    UIViewRoot root = context.getViewRoot();
    _returnViewId = root.getViewId();
    if (_returnViewId == null)
        return;
    ExternalContext external = context.getExternalContext();
    Application application = context.getApplication();
    ViewHandler viewHandler = application.getViewHandler();
    String actionURL = viewHandler.getActionURL(context, _returnViewId);
    actionURL = external.encodeActionURL(actionURL);
    try {
        external.redirect(actionURL);
    } catch (IOException e) {
        String message = e.getMessage();
        // to do ...
    }
}

Now that the code is in place to handle the switching, the next step is to enable the dynamic switching. This behavior can be achieved by implementing a custom Phase
Listener to get the new Locale from the managed bean, then enable UIViewRoot to be set with this Locale. The code for handling this will be in the overridden beforePhase() method.

**Note:** Example code has not been optimized and can contain performance contentions.

The code using SiteStructureContext is the focus in this example, and is necessary to ensure that the navigation model is updated. Otherwise, the model (Locale) can only be updated by the browser settings.

```java
public void beforePhase(PagePhaseEvent pagePhaseEvent) {
    Integer phase = pagePhaseEvent.getPhaseId();
    if (phase.equals(ADFLifecycle.PREPARE_MODEL_ID)) {
        FacesContext fctx = FacesContext.getCurrentInstance();
        ChangeLocale changeLocale =
            (ChangeLocale)fctx.getApplication().evaluateExpressionGet
            (fctx, "#{ChangeLocale}", Object.class);
        Locale preferredLocale = changeLocale.getPreferredLocale();
        UIViewRoot uiViewRoot = fctx.getCurrentInstance().getViewRoot();
        if (preferredLocale == null) {
            changeLocale.setPreferredLocale(uiViewRoot.getLocale());
        } else {
            uiViewRoot.setLocale(preferredLocale);
        }
        try {
            SiteStructureContext ctx = SiteStructureContext.getInstance();
            SiteStructure model = ctx.getDefaultSiteStructure();
            model.invalidateCache();
        } catch (ResourceNotFoundException rnfe) {
            rnfe.printStackTrace();
        }
    }
}
```
This chapter describes how to use Oracle JDeveloper to design and develop page templates for portal pages displayed in WebCenter Portal Framework applications and portals built with WebCenter Portal (Portal Builder).

This chapter includes the following topics:

- Section 11.1, "Introduction to Developing Page Templates"
- Section 11.2, "Best Practices for Developing Page Templates"
- Section 11.3, "Creating a Page Template"
- Section 11.4, "Editing a Page Template"
- Section 11.5, "Applying a Page Template to a Page"
- Section 11.6, "Selecting the Default Page Template"
- Section 11.7, "Deleting a Page Template"
- Section 11.8, "Page Templates Tutorials and Examples"

11.1 Introduction to Developing Page Templates

Page templates define how individual pages and groups of pages display on a user’s screen, and help to ensure that the pages in a portal are consistent in structure and layout. If you change a page template, then all pages that reference that template automatically inherit the changes.

See Also: For general information about page templates, see the "Using Page Templates" section in Web User Interface Developer’s Guide for Oracle Application Development Framework.

Oracle recommends that you use JDeveloper to develop page templates for both Portal Framework applications and portals built with WebCenter Portal (Portal Builder). In JDeveloper, you can build fully functional page templates from scratch or you can refine and further develop existing page templates to suit your particular requirements. You can also develop page templates in Portal Builder, but the editing capabilities are limited.

When fully developed, you can upload page templates directly to WebCenter Portal (the portal server) for immediate use or for testing. Alternatively, you can export the page template to a file and upload the page template to WebCenter Portal later on through Portal Builder.

You can develop page templates in the following ways:
Build page templates from scratch, as described in Section 11.3, "Creating a Page Template."

Download out-of-the-box page templates provided with WebCenter Portal (Portal Builder), and modify them, as described in Section 11.4, "Editing a Page Template."

Leverage the out-of-the-box page templates provided with JDeveloper, as described in Section 11.1.4, "Understanding the Built-In Page Templates in a WebCenter Portal Framework Application” and Section 11.4, “Editing a Page Template."

This section includes the following topics:

- Section 11.1.1, "Understanding Page Template Structure"
- Section 11.1.2, "Understanding Page Template Layout"
- Section 11.1.3, "Understanding Page Template Layout Components"
- Section 11.1.4, "Understanding the Built-In Page Templates in a WebCenter Portal Framework Application”

11.1.1 Understanding Page Template Structure

Typical elements of a page template include:

- Header, content area (different in each page), and footer. The header and footer commonly include brand-specific elements. For example, a header usually includes a logo and possibly a slogan, and a footer usually includes contact and copyright information.

- Navigation. A page template can expose the navigation for a portal in many ways. For example, on a mobile device, portal navigation may be shown as a popup or slide in/out. On a desktop browser, portal navigation is typically shown along the top of the page, or down the side of the page.

- Branding elements. For example, a page template may include a company logo, slogan, or copyright message.

- Links and actions. For example, a log in/log out link, drop-down menus, or global links (such as links to send a mail message to the web administrator or to display a privacy statement).

- Conditional elements. For example, some elements on the page might differ depending on whether the user is public or authenticated or depending on the user’s role and privileges.

Figure 11–1 shows a sample application based on a page template that illustrates these elements.
One of the most important aspects of page template design is the layout of components, both elements of the template and page content. There are two basic strategies:

- **A flowing layout** is the most typical layout. Components are arranged side by side or one below the other, displayed using their natural size. When the content of the page extends beyond the size of the browser window, the browser displays scroll bars on the page.

- **A stretching layout** may be a suitable choice when your page content fills a large area, or you want the page content to grow and shrink depending on the size of the browser window. Components are stretched to occupy the available space on the page. For example, a stretching layout may be suitable when a page contains a table or graph that you want to fill up the whole content area, no matter what size it is. Another example is a page that contains an editing area, where you want the editor to be exactly as tall and wide as the space given to the content area. This layout has a region for the page content, and adds scroll bars to the region on the page when the content cannot be contained within the size of the browser window. In other words, individual components display scroll bars (which might mean that you have multiple scroll bars on one page).

Stretching enables you to maximize the usage of the viewable area. You can use tabs, accordions, menus, and popups to expand your viewable area. When scroll bars are added to the page, the navigation area, page header, and page footer remain in view while the content area scrolls.

Because most web sites use a flowing layout, you will probably also want to use a flowing layout as it will likely feel more familiar to your users. However, stretching layouts are good for dashboards and applications that are rich in nature or when you want to mimic a desktop experience. You can also combine flowing and stretching layout on the same page.
Vertical Behavior
Depending on whether your layout is flowing or stretching, the vertical behavior of the page differs as follows:

- **Flowing layout:**
  - The header and/or footer might not always be visible
  - The height of the page is calculated based on the page contents
  - The content is never stretched vertically
  - The browser might display a scroll bar

- **Stretching layout:**
  - The header and footer are always visible
  - The height of the page is determined by the browser window
  - The content is stretched vertically
  - The content area might display a scroll bar

Horizontal Behavior
Depending on whether your layout is flowing or stretching, the horizontal behavior of the page differs as follows:

- **Flowing layout:**
  - If your page includes a side bar (for example, left-side navigation), the side bar might not always be visible
  - The width of the page is calculated based on the components
  - Some components might be stretched to fill up existing space
  - The browser might display a scroll bar

- **Stretching layout:**
  - If your page includes a side bar (for example, left-side navigation), the side bar is always visible
  - The width of the page is determined by the browser window
  - The content is stretched horizontally
  - The content might display a scroll bar

11.1.3 Understanding Page Template Layout Components
The underlying structure of a page template is provided by Oracle Application Development Framework (ADF) layout components. After you decide on the appropriate layout to use for your page template (see Section 11.1.2, “Understanding Page Template Layout”), you will use ADF layout components to create your page template. This is a complex task, and requires knowledge of the ADF components that will achieve the structure and layout you require, and best practices to employ (see Section 11.2, “Best Practices for Developing Page Templates”).
See Also:

- For details on what different layout components look like, see the ADF Faces Rich Client demo online tool:
  

  In the demo tool, you can select Page or Page Template from the View Source menu to see what layout components and attributes are used to achieve the page structure.
  
- Section 11.8, "Page Templates Tutorials and Examples"

Figure 11–2 and Figure 11–3 illustrate common ADF components used in a page template layout, showing the page template code followed by the generated layout:

- af:panelGridLayout—a versatile layout component that uses rows (gridRow) and cells (gridCell) to define a structured layout. This component is the preferred general layout component as it offers a small client side footprint while being very flexible in layout capabilities. It allows you to more naturally define areas of the page to match your desired layout in the form of rows and columns. With panelGridLayout, you can easily specify fixed or variable column widths (% or pixels), which cannot be done as easily with other layout components. See Figure 11–2.

- af:panelGroupLayout—a flowing series of components arranged horizontally, vertically, or in scrollable structures. Typically, panelGroupLayout is used with flowing layouts, and with flowing content inside a stretching layout. It provides vertical and horizontal scroll bars if the content does not fit into the available space. See Figure 11–2 and Figure 11–3.

- af:panelSplitter—a stretched box divided into two user-modifiable sections. See Figure 11–3.
**Figure 11–2  Page Template Code and Generated Stretching Layout: Example 1**

```xml
<af:gridRow height="50px" id="gr1">
    <af:gridCell halign="stretch" valign="stretch" colspan="3" id="gc4">
        <af:panelGroupLayout id="pg1"/>
    </af:gridCell>
</af:gridRow>

<af:gridRow height="100%" id="gr1">
    <af:gridCell width="20%" halign="stretch" valign="stretch" id="gc2">
        <af:panelGroupLayout id="pg2"/>
    </af:gridCell>
    <af:gridCell width="70%" halign="stretch" valign="stretch" id="gc5">
        <af:panelGroupLayout id="pg3"/>
    </af:gridCell>
    <af:gridCell width="10%" halign="stretch" valign="stretch" id="gc3">
        <af:panelGroupLayout id="pg4"/>
    </af:gridCell>
</af:gridRow>

<af:gridRow height="50px" id="gr3">
    <af:gridCell halign="stretch" valign="stretch" colspan="3" id="gc1">
        <af:panelGroupLayout id="pg5"/>
    </af:gridCell>
</af:gridRow>
</af:panelGridLayout>
```
11.1.4 Understanding the Built-In Page Templates in a WebCenter Portal Framework Application

Note: For information about the built-in page templates, see the "About Built-in Page Templates" section in Building Portals with Oracle WebCenter Portal. You can download these page templates and modify them in JDeveloper, as described in Section 11.4, "Editing a Page Template."

When you create a Portal Framework application, and select the Configure the application with standard Portal features check box (see Section 6.1, "Creating a New WebCenter Portal Framework Application"), as shown in Figure 11–4, two built-in templates are added to your application: pageTemplate_globe.jspx and pageTemplate_swooshy.jspx.
These built-in page templates both offer essentially the same functionality but with different graphics. Figure 11–5 shows the pageTemplate_globe.jspx page template.
1. Link to the portal home page
2. Tag line
3. Welcome message
4. Link to the runtime administration console
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

These page templates use the following attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>contentWidth</td>
<td>String</td>
<td>960px</td>
</tr>
<tr>
<td>showNavigation</td>
<td>Boolean</td>
<td>true</td>
</tr>
<tr>
<td>showGreetings</td>
<td>Boolean</td>
<td>$(securityContext.authenticated)</td>
</tr>
<tr>
<td>showLogin</td>
<td>Boolean</td>
<td>true</td>
</tr>
<tr>
<td>showAdmin</td>
<td>Boolean</td>
<td>$(securityContext.authenticated)</td>
</tr>
<tr>
<td>isAdminPage</td>
<td>Boolean</td>
<td>false</td>
</tr>
<tr>
<td>isEditingTemplate</td>
<td>Boolean</td>
<td>false</td>
</tr>
</tbody>
</table>

You can edit these built-in page templates to meet your particular requirements, or you can create your own from scratch. If you choose to create your own page templates, Oracle recommends that you refer to the latest built-in page templates, which provide examples of recommended layout components that include fewer task flows and faster performance. See the “About Built-in Page Templates” section in *Building Portals with Oracle WebCenter Portal*.

### 11.2 Best Practices for Developing Page Templates

Because page templates are present in every page of a portal, the design of your page templates should be carefully planned to optimize their performance and conform to best practices. This section provides tips for developing page templates for a portal.
Note about Page Templates in WebCenter Portal (Portal Builder):
The development of a page template is a complex task. Oracle recommends that you use JDeveloper to develop page templates for both Portal Framework applications and portals built with WebCenter Portal (Portal Builder). You can also develop page templates in Portal Builder, but the editing capabilities are limited.

When fully developed, you can upload page templates directly to WebCenter Portal (the portal server) for immediate use or for testing. Alternatively, you can export the page template to a file and upload the page template to WebCenter Portal later on through Portal Builder.

For more information, see Section 11.3, "Creating a Page Template," Section 11.4, "Editing a Page Template," and Section 55.1, "Developing Assets for WebCenter Portal."

Table 11–2 provides a quick reference summary of considerations and guidance for achieving the best results out of your page templates.

Table 11–2  Best Practices Summary for Page Templates

<table>
<thead>
<tr>
<th>Considerations</th>
<th>Best Practices</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance</td>
<td>While all of this section, &quot;Best Practices for Developing Page Templates&quot;, is aimed at improving the performance of your page templates, there are a few general tips to keep in mind as overall best practices:</td>
</tr>
<tr>
<td></td>
<td>■ Minimize the use of embedded task flows. For examples, refer to the latest built-in page templates included with WebCenter Portal, described in the &quot;About Built-in Page Templates&quot; section in Building Portals with Oracle WebCenter Portal.</td>
</tr>
<tr>
<td></td>
<td>■ Minimize the use of ADF layout components, using panelGridLayout to implement your page template structure. The fewer ADF layout components, the easier it is to apply skins.</td>
</tr>
<tr>
<td></td>
<td>■ Avoid using images for decorative purposes (such as rounded corners). Consider using CSS instead of images.</td>
</tr>
<tr>
<td></td>
<td>■ Defer loading of ADF components, such as menus and dialogs, where possible. This can be controlled through ADF such as contentDelivery, childCreation, and so on.</td>
</tr>
<tr>
<td></td>
<td>■ Avoid elements that require extra time in rendering page templates and try to substitute for elements that require less processing time.</td>
</tr>
</tbody>
</table>
Best Practices for Developing Page Templates

Table 11–2 (Cont.) Best Practices Summary for Page Templates

<table>
<thead>
<tr>
<th>Considerations</th>
<th>Best Practices</th>
</tr>
</thead>
</table>
| Layout         | One of the most important aspects of page template design is how to lay out components, both elements of the template and page content. Page templates can have a flowing layout or a stretching layout. For more details about these two strategies, see Section 11.1.2, "Understanding Page Template Layout."

As a page template developer, you can control whether the content facet is in a stretching or flowing region of the page. Page content must be created taking the layout strategy into consideration.

Once you decide on the best strategy for your page templates, see the following sections for tips on creating the layout you choose:

- Section 11.2.1, "Best Practices for Creating Stretching Layouts"
- Section 11.2.2, "Best Practices for Creating Flowing Layouts"

| Navigation     | A page template can expose the navigation for a portal in many ways. For example, in a desktop browser, portal navigation is typically shown along the top of the page, or down the side of the page:

- A top navigation page template exposes the portal navigation in header area. Top navigation makes effective use of the horizontal space on the page, and is recommended when there are seven or fewer top level pages in the portal navigation. This page template design generally has a header, page and footer sections, and is an ideal starting point for sites that require a flowing layout.

- A side navigation page template exposes the portal navigation in a sidebar on the left side of the page. The vertical nature of side navigation allows for a more lengthy list of navigation items, and is recommended when there are more than seven top level pages in the portal navigation. Choose a side navigation template for more complex navigation models.

On different devices, other navigation models can be used. For example, in page templates optimized for mobile devices, navigation can be provided as either a popup or slide in/out.

The navigation model provided out-of-the-box with WebCenter Portal (Portal Default Navigation Model) is a shared asset that can be embedded in a page template to provide ready-built navigation that can be customized and extended as desired. For more information, see the "Editing a Navigation Model" and "About Built-in Page Templates" sections in Building Portals with Oracle WebCenter Portal.

| Skin           | Each page template works together with a skin to determine the overall look and feel of the pages in your portal. While the page template controls the structure of components on the page, the skin controls the visual appearance of those components, such as the colors, fonts, and other aspects such as the position, height, and width of components on the page.

Each page template can define a preferred skin to identify the skin that works best with that page template. When the page template is selected as the default page template for a portal or as the system default, the default skin automatically updates to the page template’s preferred skin.

For more information, see Chapter 13, "Developing Skins."
Table 11–2 (Cont.) Best Practices Summary for Page Templates

<table>
<thead>
<tr>
<th>Considerations</th>
<th>Best Practices</th>
</tr>
</thead>
</table>
| Storage location     | In a Portal Framework application, to expose the new page template in the **Assets** page of the runtime administration console, you must create the page template under the `Application_Root/Portal/public_html/oracle/webcenter/portalapp/pagetemplates` directory, which contains the out-of-the-box templates. Creating your page templates in this directory provides the following advantages:  
  - Page templates can be packaged into resource archives, which can be imported and used in other WebCenter Portal Framework applications, or portals built with Portal Builder. You can create general purpose, reusable page templates.  
  - Page templates can be managed at run time using the Administration Console. After importing a page template, you need to edit `pagetemplate-metadata.xml` to register the newly imported template for JDeveloper. For more information, see Section 9.5.1, "Enabling Runtime Administration of Portal Resources." |
| Runtime behavior     | If you develop a page template in JDeveloper that you plan to allow authorized users to edit in Composer at runtime, follow the tips provided in Section 11.2.3, "Best Practices for Developing Page Templates That Can Be Edited at Runtime." Select the **Create Associated ADFm Page Definition** option when developing a page template so that the template can include portlets and task flows and to enable runtime switching. For more information, see Section 11.3, "Creating a Page Template," and Section 11.5.3, "How to Enable Dynamic Switching of Page Templates at Runtime in a WebCenter Portal Framework Application." |
| Components           | For details on what different ADF layout components look like, see the ADF Faces Rich Client demo online tool ([http://jdevadf.oracle.com/adf-richclient-demo/faces/visualDesigns/index.jsp](http://jdevadf.oracle.com/adf-richclient-demo/faces/visualDesigns/index.jsp)). Select **Page** or **Page Template** from the View Source menu for a component to see what tags and attributes are used as well as what the component structure looks like for the page. Because a page template layout can be changed, create pages and design custom components that display properly in flowing and stretching context. For tips on customizing the appearance of components, see Section 11.2.4, "Best Practices for Customizing the Appearance of Components." |
| Scrolling            | Add scroll bars only around flowing island content. For tips on implementing scroll bars, see Section 11.2.5, "Best Practices for Defining Scrolling in a Page Template." |
| Margins, borders, and padding | Due to browser CSS Box Model Rules, defining margins, borders and padding on components might be complex. For tips on resolving the complexities of defining margins, borders and padding on components, see Section 11.2.6, "Best Practices for Defining Margins, Borders, and Padding." |
Best Practices for Developing Page Templates

This section includes the following topics:

- Section 11.2.1, "Best Practices for Creating Stretching Layouts"
- Section 11.2.2, "Best Practices for Creating Flowing Layouts"
- Section 11.2.3, "Best Practices for Developing Page Templates That Can Be Edited at Runtime"
- Section 11.2.4, "Best Practices for Customizing the Appearance of Components"
- Section 11.2.5, "Best Practices for Defining Scrolling in a Page Template"
- Section 11.2.6, "Best Practices for Defining Margins, Borders, and Padding"
- Section 11.2.7, "Best Practices for Hiding Content in Page Templates in a Portal Framework Application"

### 11.2.1 Best Practices for Creating Stretching Layouts

If your page template is best suited to a stretching layout (see Section 11.1.2, "Understanding Page Template Layout"), follow these tips as you develop the layout:

- Build the outer structure with containers that can be stretched and can stretch their children. Inside your Document component, use containers such as panelGridLayout (Figure 11–2) with rows (gridRow) and cells (gridCell), and panelSplitter (Figure 11–3).
Note: Each layout or panel component's tag documentation identifies whether it is stretchable and how to achieve it in its "Geometry Management" documentation. Some components have attributes to determine whether children will be stretched or not. For example: document has a maximized attribute, showDetailItem has a stretchChildren attribute.

- Create flowing islands. Inside of the stretchable outer structure, create islands of flowing (non-stretched) components. To make this transition from stretching to flowing, use panelGroupLayout with layout="scroll" or layout="vertical" since it supports being stretched but will not stretch its children.
- On stretching components, set dimensionsFrom="auto" so that the stretching component (for example, panelStretchLayout) will only try to stretch its child if it itself is being stretched. If it is not being stretched, then it will flow (not stretch) its children.
- Do not stretch something vertically (by using a height with a percent value) when inside a flowing container.
- Do not use the position CSS property in an inlineStyle. Doing so will impede the ability to override this property with a style specified in the skin.

Note: The following components are just some of the components that cannot be reliably stretched:
- Most input components
- panelBorderLayout
- panelFormLayout
- panelGroupLayout (with layout="default")
- panelGroupLayout (with layout="horizontal")
- panelHeader (with type="flow")
- panelLabelAndMessage
- panelList
- panelGrid
- tableLayout (Apache MyFaces Trinidad HTML Component)

11.2.2 Best Practices for Creating Flowing Layouts

If your page template is best suited to a flowing layout (see Section 11.1.2, "Understanding Page Template Layout"), follow these tips as you develop the layout:

- Use panelGridLayout with rows (gridRow) and cells (gridCell) to define a structured layout that will flow.
- To avoid multiple scroll bars, do not nest scrolling panelGroupLayout components, instead use layout="vertical".
- Most stretchable ADF components also work in flowing context with dimensionsFrom="auto".
To stretch a component horizontally, use `styleClass="AFStretchWidth"` (instead of `inlineStyle="width:100.0%"`).

Working with customizable components:

- In `panelCustomizable`, use `layout="auto"` to detect whether to stretch its children.
- To support flowing and stretching layouts, use `showDetailFrame` with `stretchChildren="auto"`.

### 11.2.3 Best Practices for Developing Page Templates That Can Be Edited at Runtime

To create a page template at design time (in JDeveloper) that is suited to being editable at runtime (in Composer), follow these tips:

- Add components from the Composer group in the Component Palette.
- Do not add `pageCustomizable` to a page template.
- Add at least one `panelCustomizable` component, which provides a container with horizontal or vertical layout that holds other components.
- Inside a `panelCustomizable` component, add ADF components (such as `outputText`, `richTextEditor`, or `goLink`) surrounded by a `showDetailFrame`, which provides a chrome for a component that enables you to add actions like show, hide, and move. You can edit the frame or the embedded component properties.

**Caution:** Use this technique sparingly, as `showDetailFrame` components impact processing time. If the end user is not expected to move components around, do not wrap them in a `showDetailFrame`.

Many of the components available to add to a page at design time are also available at runtime in the resource catalog. **Table 11–3** maps design time components to runtime components.

<table>
<thead>
<tr>
<th>Design Time (JDeveloper)</th>
<th>Runtime (Composer)</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>panelCustomizable</code></td>
<td>Box</td>
</tr>
<tr>
<td><code>outputText</code></td>
<td>HTML Markup</td>
</tr>
<tr>
<td><code>goLink</code></td>
<td>Hyperlink</td>
</tr>
<tr>
<td><code>goImageLink</code></td>
<td>Image</td>
</tr>
<tr>
<td><code>showDetailFrame</code></td>
<td>Movable Box</td>
</tr>
<tr>
<td><code>richTextEditor</code></td>
<td>Text</td>
</tr>
<tr>
<td><code>inlineFrame</code></td>
<td>Web Page</td>
</tr>
</tbody>
</table>

### 11.2.4 Best Practices for Customizing the Appearance of Components

To customize the appearance of components, follow these tips:

- For custom styling, use Cascading Style Sheets (CSS), which is easy to maintain and can be changed without changing the page template source. For example,
make the background color of a page blue using the CSS code `background-color: blue`.

- Use a custom skin for consistently modified appearances if the existing skin doesn’t provide all that you need.
- For instance-specific alternative styling, use the `styleClass` attribute. Keep the corresponding style definitions in an easy-to-maintain location such as in a custom skin (recommended) or in a style provided by the `af:resource` tag.
- As a last resort, use component attributes such as `inlineStyle`, `contentStyle`, and `labelStyle`. These are harder to maintain as they are scattered throughout components rather than collected in a single style sheet, contribute more to the page’s raw HTML size, and may not even be needed if one or more of the above mechanisms are used. Styles are directly processed by the web browser, which gives you a great deal of power but at the cost of being error-prone.
- Browsers do not support all styles on all elements and certain combinations of styles produce non-obvious results. Here is some guidance on style configurations to avoid:
  - An `inlineStyle` with a height value with % units
  - An `inlineStyle` with a width value between 90% and 100% (use `styleClass="AFStretchWidth"` or `styleClass="AFAuxiliaryStretchWidth"` instead)
  - An `inlineStyle` with height, top, and bottom values
  - An `inlineStyle` with width, left, and right values
  - An `inlineStyle` with a position value
  - In a child being stretched by a parent component, an `inlineStyle` with width or height values

### 11.2.5 Best Practices for Defining Scrolling in a Page Template

To define scrolling in a page template, follow these tips:

- Try to avoid the need for end users to scroll horizontally by designing the page content to occupy the available screen size.
- Add scroll bars only around flowing island content. The recommended transition component for switching from a stretching outer frame into a flowing island is the `panelGroupLayout` with `layout="scroll"`. If the contents of this `panelGroupLayout` cannot fit in the space allocated, the browser will determine whether scroll bars are needed and will add them automatically.
- Do not nest scrolling `panelGroupLayout` components because this will make the user see multiple scroll bars. Also, this should only be used at transitions from stretching to flowing areas and since you should not have stretching areas inside of flowing areas, you would generally never end up with nested scroll bars. It is best to minimize the number of areas that users must scroll in order to see what they are looking to find. Take time to consider what scrolling users will need. In cases where undesired scroll bars exist, you may want to change the `layout` attribute of the `panelGroupLayout` to `vertical`.
11.2.6 Best Practices for Defining Margins, Borders, and Padding

Due to browser CSS Box Model Rules, defining margins, borders and padding on components can be complex. In many cases, to apply these kinds of styles, you need to use multiple components together:

- In a scrolling area, you can add an extra panelGroupLayout with layout="vertical" with the padding defined on it, inside of the outer panelGroupLayout with layout="scroll".

- In a stretching area, you may need to wrap a panelGridLayout component inside a panelGridLayout with spacers inside gridCell components. See Figure 11–2.

See Also: Refer to the Navigation-Master-Detail, Tiled Flowing, and Tiled Stretching layout pattern examples for various mechanisms to apply padding:


11.2.7 Best Practices for Hiding Content in Page Templates in a Portal Framework Application

In a Portal Framework application, there are several special pages based on a page template that hide parts of the page template. These pages use the attributes specified in the page template to accomplish this:

- In out-of-the-box login.jspx and error.jspx: showNavigation=false and showLogin=false.

- In built-in administration pages: isAdminPage=true.

- When editing a template, Assets Manager sets isEditingTemplate=true.

Use these attributes in your custom page template if the template will be used as the default template or if you use the out-of-the-box login page.

You may allow page templates with editable components to be edited using Composer. Oracle recommends you hide the template content, except the content facet reference, when editing the page so that only the page content, not the page template, can be edited. For example, use the rendered attribute:

rendered = "#{!composerContext.inEditMode or attrs.isEditingTemplate}"

11.3 Creating a Page Template

Before you create a page template, read through Section 11.2, "Best Practices for Developing Page Templates."

This section describes how to start from scratch with a new page template of your own design. Alternatively, you can start with the two built-in page templates provided in Portal Framework applications, or the latest page templates provided with WebCenter Portal (recommended), and modify them according to your requirements. To use an existing page template as a starting place, see Section 11.4, "Editing a Page Template."
Creating a Page Template

To create a page template:

1. In the Application Navigator, right-click the node where you want to create and store page templates (Figure 11–6) and choose **New**:  

   ![Figure 11–6  Contents of the pagetemplates Folder](image)

   - For Portal Framework applications, the recommended directory is `/oracle/webcenter/portalapp/pagetemplates`

   **Note:** In a Portal Framework application, to expose the new page template in the **Assets** page of the runtime administration console, you must create the page template under the `Application_Root/Portal/public_html/oracle/webcenter/portalapp` directory. By default, Portal Framework applications include a separate `pagetemplates` directory, which contains the built-in page templates.

   - For page templates to be used in WebCenter Portal (Portal Builder), use the `WebCenterSpacesResource` project to select a directory for shared assets or portal assets, as described in Section 55.1, "Developing Assets for WebCenter Portal."

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 11–7), 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 11–7 The Create JSF Page Template Dialog**

4. In the **Directory** field, verify or enter the full directory path of the location under which to create the page template.

5. In the **Page Template Name** field, enter a display name for the page template that will help users quickly identify which type of pages this template should be used for.

6. If you want to use one of the predefined layouts for your page template, select **Use a Quick Start Layout**.

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. For runtime 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 built-in page templates provided with Portal Framework applications have associated page definitions, for page template switching to work in Portal Framework applications, any other page templates created within the application must also have associated page definitions.
8. Click the **Facet Definitions** tab to define the areas, or *facets*, in the page template where content can be inserted.

Facets are placeholders in the template that will be filled in by the pages using the template. An ADF page template can have any number of facets, but a Portal Framework application requires that each template have one facet called *content*. Typically Portal Framework applications do not use any other facets, since WebCenter Portal's runtime tools recognize only this facet. Other facets could be used only at design time, in JDeveloper.

A page template that is intended for use in a WebCenter Portal application must contain at least one facet definition that must be called *content*. To create this facet definition:

   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 Section 11.4, "Editing a Page Template."

Example 11–1 shows the *content* facet definition in the `<af:xmlContent>` section of the pageTemplate_globe.jspx built-in page template.

**Example 11–1  Content Facet in the Globe Built-In Page Template**

```xml
<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.

A page template without attributes is syntactically correct. However, page template attributes are useful when you want to use one page template for several pages where the template should render slightly differently.

For example, the pageTemplate_globe.jspx built-in 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 11–2 shows the *isAdminPage* attribute definition in the page template.

**Example 11–2  Attribute Definition in the Globe Built-In Page Template**

```xml
<af:xmlContent>

```
...<attribute>
  <attribute-name>isAdminPage</attribute-name>
  <attribute-class>java.lang.Boolean</attribute-class>
  <default-value>#{false}</default-value>
</attribute>
...

Example 11–3 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 11–3 Conditional Value for Rendering the Administration Page Link

Example 11–4 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 11–4 Setting the Value of a Page Template Attribute

Example 11–5 shows how the page template uses the value of the contentWidth attribute to set the page width. The default value is 960 pixels.

Example 11–5 contentWidth Attribute

Example 11–6 shows how the page template uses the value of the showNavigation and showLogin attributes to show or hide the login link and the navigation bar.

10. Click OK.

The page template is created and opened in the visual editor. The next step is to edit the page template to define its layout and content.

11.4 Editing a Page Template

You can edit your page template after its initial creation. You can also edit one of the built-in page templates to alter it to meet your specific requirements. Any changes you make are automatically rolled out to any pages that use the page template.
Before you edit a page template, read through Section 11.2, "Best Practices for Developing Page Templates."

For examples of features that you can include in your own page templates, refer to:

- The built-in page templates provided with WebCenter Portal, specifically the latest templates that include fewer task flows and the performance-optimizing panelGridLayout component, which uses rows (gridRow) and cells (gridCell) to define a structured layout. See the "About Built-in Page Templates" section in Building Portals with Oracle WebCenter Portal.

Even if you do not intend to base your page templates on the built-in ones at all, it is still worth studying them for ideas. For example, the built-in page templates include a login form, and a good example of navigation visualization, which you can modify according to your requirements.

If iterative development is enabled, any changes that you make to page templates can be seen immediately in a running Portal Framework application. For more information about iterative development, see Section 2.3.1, "Preparing for Iterative Development in a Portal Framework Application."

To edit a page template:

1. To add WebCenter Portal-specific components to your page template, you must first add the JSP library to your project in JDeveloper. For example:

   JDev_HOME/jdeveloper/webcenter/modules/oracle.webcenter.framework_11.1.1/spaces-components.jar

   After adding the library to your project, you see a new component palette named SpacesDeclarativeComponents. You can drop components from the palette into your page template. These same built-in components are available at run time through the resource catalog, under Template Development.

2. In the Application Navigator, right-click the page template that you want to edit, and choose Open.

   - For Portal Framework applications, the recommended directory for page templates is /oracle/webcenter/portalapp/pagetemplates
Note: In a Portal Framework application, to expose the new page template in the Assets page of the runtime administration console, the page template must be stored under the Application_Root/Portal/public_html/oracle/webcenter/portalapp directory. By default, Portal Framework applications include a separate pagetemplates directory.

For page templates to be used in WebCenter Portal (Portal Builder), use the WebCenterSpacesResource project to locate page templates for shared assets or portal assets, as described in Section 55.1, "Developing Assets for WebCenter Portal."

3. In the visual editor, make required changes to the layout and content of the page template.
   a. Drag components from the Component Palette and drop them onto the page template in the visual editor.

   Navigation visualization is a very important aspect of a page template. For information about adding navigation visualization to your page template, see Section 10.11, "Visualizing Portal Navigation."

   For information about adding other resources to your page template, such as portlets, task flows, or content, see Chapter 15, "Creating Pages and Adding Resources."

   Other features you might want to provide in your page template include a log in/log out area or a search field. The built-in 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."

   b. 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 drop-down 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 11–6 shows how the content facet is included in the pageTemplate_globe.jspx built-in page template.

      Example 11–6 The content Facet in the Globe Built-In Page Template

<pre>
<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:panelGroupLayout>
</af:pageTemplateDef>
</pre>
c. To add new facet definitions or attributes, select the `<af:pageTemplateDef>` tag in the Structure window and use the Property Inspector. For examples, see the steps in Section 11.3, "Creating a Page Template."

d. For general information about JSF page templates, see the "Using Page Templates" section in the Web User Interface Developer’s Guide for Oracle Application Development Framework.

4. Save the page template definition file.

11.5 Applying a Page Template to a Page

This section describes how to apply a page template to a page in a Portal Framework application.

---

**Note:** If you developed a page template in JDeveloper for use in WebCenter Portal (Portal Builder), this task is performed using Portal Builder, where the page template is selected at either the application level or the portal level to define the structure and layout (header, body, navigation, and footer) of all pages in the portal. For Portal Builder, see the "Choosing a Default Page Template" section in Administering Oracle WebCenter Portal to configure global defaults across portals, and the "Changing the Page Template for a Portal" section in Building Portals with Oracle WebCenter Portal to select the default page template for the portal.

---

This section includes the following topics:

- Section 11.5.1, "How to Apply a Page Template During Page Creation in a WebCenter Portal Framework Application"
- Section 11.5.2, "How to Change the Page Template Applied to a Page in a WebCenter Portal Framework Application"
- Section 11.5.3, "How to Enable Dynamic Switching of Page Templates at Runtime in a WebCenter Portal Framework Application"
- Section 11.5.3.1, "What Happens at Runtime"

11.5.1 How to Apply a Page Template During Page Creation in a WebCenter Portal Framework Application

When you create a JSF page in a Portal Framework application, you can choose the page template to apply to the page in the Create JSF Page dialog, as shown in Figure 11–8.
Applying a Page Template to a Page

11.5.2 How to Change the Page Template Applied to a Page in a WebCenter Portal Framework Application

When you create a page, you select a specific page template to use for that page (see Section 11.5.1, "How to Apply a Page Template During Page Creation in a WebCenter Portal Framework Application"). This adds a static reference to the page template to the JSPX file for the page. Example 11–7 shows such a reference.

Example 11–7  Referencing A Page Template in a Page

```xml
<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. Example 11–8 shows such a binding.

Example 11–8  Binding a Page Template to a Page

```xml
<executables
    <variableIterator id="variables"/>
    <page path="/oracle.webcenter.portalapp.pagetemplates.pageTemplate_globePageDef"
        id="pageTemplateBinding" Refresh="ifNeeded"/>
</executables>
```

To apply a different page template to a page, you must change both these references.
11.5.3 How to Enable Dynamic Switching of Page Templates at Runtime in a WebCenter Portal Framework Application

You can change the references in a page's JSPX file and page definition file to use an EL expression that identifies the page template to use from the default page template setting. Users with the appropriate permissions can use the runtime administration console to switch to a different default 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 dynamic switching of page templates at runtime:

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:

```xml
<af:pageTemplate
    viewId="pathToPageTemplate"
    value="#{bindings.pageTemplateBinding}"
    id="ptId">
    <f:facet name="content"/>
</af:pageTemplate>
```

with:

```xml
<af:pageTemplate
    value="#{bindings.pageTemplateBinding.templateModel}"
    id="ptId">
    <f:facet name="content"/>
</af:pageTemplate>
```

This indicates that the page template to be used by the page is defined in the page template's page definition.

4. Right-click the JSPX file and choose **Go to Page Definition**.
5. Click the **Source** tab.
6. Replace the following code:

```xml
<page
    path="pathToPageTemplatePageDef"
    id="pageTemplateBinding" Refresh="ifNeeded"/>
```

with:

```xml
<page
    viewId="${preferenceBean.defaultPageTemplate}"
    id="pageTemplateBinding" Refresh="ifNeeded"/>
```

The `${preferenceBean.defaultPageTemplate}` EL expression retrieves the name of the page template from the default page template setting.

7. You can set the default page template, as described in Section 11.6, "Selecting the Default Page Template."

**11.5.3.1 What Happens at Runtime**

When users with appropriate permission select a page template in the runtime administration console, it will be used by all the pages because the expression
11.6 Selecting the Default Page Template

This section describes how to set the default page template that should be used by default for all pages in a Portal Framework application.

When you first create a Portal Framework application, the pageTemplate_globe.jspx built-in page template is set as the application's default page template. If you subsequently create your own page template to use for the structure and layout of your application pages, you can set it as the default page template.

You can set the default page template for a Portal Framework application by editing the oracle.webcenter.portalapp.pagetemplate.pageTemplate preference in the adf-config.xml file. The preference bean #{preferenceBean} allows runtime access to this section in adf-config.xml and the configuration specified there.

To select the default page template:

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.pagetemplate.pageTemplate

4. Set the value attribute to the path of the page template that you want to use as the default for the application, for example:

   value="/oracle/webcenter/portalapp/pagetemplates/myPageTemplate.jspx"

   **Example 11–9** shows an example of the complete preference element.

   **Example 11–9  The Default Page Template ADF Preference**

   <preference id="oracle.webcenter.portalapp.pagetemplate.pageTemplate"
   desc="Default Page Template"
   value="/oracle/webcenter/portalapp/pagetemplates/myPageTemplate.jspx"
5. **Save the adf-config.xml file.**

You can now reference the default page template without having to specify its actual name. **Example 11–10** shows the EL expression to include in a page’s page definition file to indicate that a page should use the application’s default page template, rather than an explicitly selected page template.

**Example 11–10  EL Expression to Use the Application Default Page Template**

```
<page viewId="${preferenceBean.defaultPageTemplate}" id="pageTemplateBinding" Refresh="ifNeeded"/>
```

### 11.7 Deleting a Page Template

This section describes how to delete a page template in a Portal Framework application.

---

**Note:** If you developed a page template for use in WebCenter Portal (Portal Builder), this task is performed using Portal Builder, where the default page template is deleted at either the application level or the portal level. For Portal Builder, see the “Deleting an Asset” section in Building Portals with Oracle WebCenter Portal.

---

**Caution:** Before you delete a page template, ensure that there are no pages that use the page template. When you delete a page template, attempting to run any page that has a direct reference to the page template will result in a runtime error.

To delete a page template in JDeveloper:

- Right-click the page template in the Application Navigator and choose **Delete** from the context menu.

For more information, see **Section 9.4, "Deleting Portal Resources."**

### 11.8 Page Templates Tutorials and Examples

The following supplementary tutorials and examples provide additional information about page templates:

- **Dissecting a Page Template.** Analyzes the details of a simple page template for an Oracle WebCenter Portal Framework application, including recommended practices.
  

- **Oracle WebCenter Portal Online Training: Creating and Using Page Templates in Oracle WebCenter Portal Applications.** Provides a recorded presentation and slides to create and use page templates for a portal in an earlier release.
  
- *Optimizing WebCenter Portal Mobile Delivery.* Identifies and analyzes some common WebCenter Portal performance bottlenecks related to page weight and describes a generic approach that can streamline a portal while improving the performance and response times. Of particular interest in the context of developing page templates is the section "Page Design and Component Choices." A sample application is available for download.

http://www.ateam-oracle.com/webcenter-mobile-delivery/

- *Working with Links in WebCenter Application.* Describes considerations for using links. Pertinent to page templates are links in menus, breadcrumbs, buttons, and images.

This chapter describes how to use Oracle JDeveloper to create and manage page styles and task flow styles for your WebCenter Portal Framework applications.

This chapter includes the following topics:

- Section 12.1, "Introduction to Styles"
- Section 12.2, "Developing Page Styles"
- Section 12.3, "Developing Task Flow Styles"

12.1 Introduction to Styles

Oracle WebCenter Portal provides several different styles to determine the layout of content within a portal:

- **Page styles** determine the layout of content in a new page that is created at runtime. For more information, see Section 12.1.1, "Introduction to Page Styles."

- **Task flow styles** determine the layout of content in a task flow that is created at runtime. Task flow styles can also include objects that can be bound to any data visualization that is added to the task flow. For more information, see Section 12.1.2, "Introduction to Task Flow Styles."

- **Content Presenter display templates** define the layout of Content Presenter content items on a page. These templates are not discussed in this chapter. For more information, see Chapter 27, "Creating Content Presenter Display Templates."

Runtime Management

Portal Framework applications support the runtime administration of styles to help users continue developing a portal even after it has been deployed. With runtime administration, authorized users can manage styles in a browser-based environment, with no requirement to install or understand JDeveloper. For more information, see Section 9.5, "Working with Portal Resources at Runtime."

Round-Trip Development

You can bring runtime-created or modified styles back into JDeveloper, to ensure that any changes are not lost when the portal is redeployed. You can also edit these styles to further enhance them if necessary, and upload them back into the deployed portal. For more information, see Section 9.6, "Working with Round-Trip Development."
12.1.1 Introduction to 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 12–1 shows a sample portal page that is based on a page style.

Figure 12–1  Sample Page that Uses a Page Style

You create page styles in JDeveloper, but they are only used at runtime to create pages. A user can create pages at runtime based on the available page styles. The Create Page option is available in the Assets page of the runtime administration console. It is also available in other application pages that contain the Page - Create New task flow. When a user clicks Create Page, the Create Page dialog displays a set of predefined styles, as shown in Figure 12–2. 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.
12.1.2 Introduction to Task Flow Styles

Task flow styles are very similar to page styles, except that they are used at runtime to create task flows rather than pages. Task flow styles are especially useful when creating task flows for use with Data Presenter, as they can include objects that can then be bound to any data visualization that is added to the task flow.

Task flow styles are used only when creating task flows at runtime, but they are created and published in JDeveloper.

Task flow styles are displayed in the Create New Task Flow dialog box (Figure 12–3). Users can choose a style and create a task flow based on that style. For more information, see the “Working with Task Flows” section in Building Portals with Oracle WebCenter Portal.
12.2 Developing Page Styles

Page styles are JSPX pages that can be used as templates to create new pages at runtime.

When users create a new page at runtime, the Create Page dialog, shown in Figure 12–2, displays the available page styles.

This section includes the following topics:

- Section 12.2.1, "How to Create a Page Style"
- Section 12.2.2, "What You May Need to Know About Developing Page Styles"
- Section 12.2.3, "How to Delete a Page Style"

---

**Note:** You can develop page styles in JDeveloper for use in WebCenter Portal. For more information, see Section 9.7, "Working with WebCenter Portal Resources."

---

12.2.1 How to Create a Page Style

The recommended method for creating page styles is to download one of the existing page styles and import it into JDeveloper. You can then use this page style as the starting point for your new page style and then export it for uploading back into your application.

Creating a page style this way also ensures that it is available in the Create Page dialog.

To create a page style:

1. In the deployed application, download one of the existing page styles. For more information, see the "Downloading an Asset" section in Building Portals with Oracle WebCenter Portal.

2. Import the downloaded page style into JDeveloper. For more information, see Section 9.6.2, "How to Import a Portal Resource into JDeveloper."

3. In the Application Navigator, right-click the imported page style, and choose Open.
4. In the visual editor, make any changes to the layout of content of the page style, in the same way as you would edit any page.

For information about how to define the content of the page style, see Section 17.6.1, "How to Create Templates for Pages Created at Runtime" and Section 12.2.2, "What You May Need to Know About Developing Page Styles."

Example 12–1 shows the code for a sample page style, /oracle/webcenter/portalapp/pages/pageStyles/TemplateFlowingTwoColumn.jspx, which creates a basic page designed to flow and provide two columns that are 35 percent and 65 percent in proportion. Example 12–2 shows the associated page definition file, /oracle/webcenter/portalapp/pages/pageStyles/TemplateFlowingTwoColumnPageDef.jspx.

**Example 12–1 Source Code of a Custom Page Style**

```xml
<?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="pcl1">
              <af:panelGroupLayout id="pgl1" layout="scroll"
                styleClass="replace_with_scheme_name"
                inlineStyles="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 id="tl2"/>
                </af:panelGroupLayout>
              </f:facet>
              <f:facet name="editor">
                <pe:pageEditorPanel id="pep1"/>
              </f:facet>
            </pe:pageCustomizable>
          </f:facet>
        </af:pageTemplate>
      </af:form>
    </af:document>
  </f:view>
</jsp:root>
```
5. Save your changes.
6. Export the page style. For more information, see Section 9.6.3, "How to Export a Portal Resource from JDeveloper."
7. Upload the exported page style into the deployed application. For more information, see the "Uploading an Asset" section in Building Portals with Oracle WebCenter Portal.

12.2.2 What You May Need to Know About Developing Page Styles

There are a few prerequisites for developing page styles to be exposed as portal resources.

- If a page style is based on a page template, then while designing the page style JSPX file, 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 page template to use for the page style, in the page style JSPX file, specify that the page template is defined in the page style's page definition, as follows:
  
  ```xml
<af:pageTemplate value="#{bindings.pageTemplateBinding.templateModel}" id="T"/>
  
  You can then specify an EL value in the page style page definition to retrieve the name of the page template from the default page template setting, as follows:

  ```xml
<page viewId="#{preferenceBean.defaultPageTemplateViewId}" id="pageTemplateBinding" Refresh="ifNeeded"/>
  
  At runtime, users with the appropriate permissions can switch to a different page template by selecting a new default page template in the runtime administration console. For more information, see the "Choosing a Default Page Template" section in Administering Oracle WebCenter Portal.
- You can create a managed bean that controls which page template to use, as shown in the following example:

  ```java
public class siteTemplatesManager {
    final private String templateA = "/templateA.jspx";
    final private String templateB = "/templateB.jspx";
    private String currentSiteTemplateViewId;

    public siteTemplatesManager() {
      
  ```
12.3 Developing Task Flow Styles

Task flow styles are ADF task flows that can be used as templates to create new task flows at runtime.

You may choose to create task flow styles for the following purposes:

- To provide a specific, consistent layout for task flows.
- To seed a UI picker that can in turn be used in the task flows created from the task flow style, for example, a task flow style with a portal 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 portal.
- To provide a predefined UI that can be bound separately to data, for example, you can create a flowing data layout in the task flow style, create a task flow from it, drop a data control, then bind the flowing data layout to the data control.

When users create a new task flow at runtime, the Create New Task Flow dialog, shown in Figure 12–3, displays the available task flow styles.

This section includes the following topics:

- Section 12.3.1, "How to Create a Task Flow Style"
- Section 12.3.2, "What You May Need to Know About Developing Task Flow Styles"
- Section 12.3.3, "How to Delete a Task Flow Style"

Note: You can develop task flow styles in JDeveloper for use in WebCenter Portal. For more information, see Section 9.7, “Working with WebCenter Portal Resources.”
12.3.1 How to Create a Task Flow Style

Creating a task flow style is the same as creating a task flow, but you create task flow styles only so that they can be used as templates to create task flows at runtime.

The recommended method for creating task flow styles is to download one of the existing task flow styles and import it into JDeveloper. You can then use this task flow style as the starting point for your new task flow style and then export it for uploading back into your application.

Creating a task flow style this way also ensures that it is available in the Create New Task Flow dialog.

To create a task flow style:

1. In the deployed application, download one of the existing task flow styles. For more information, see the "Downloading an Asset" section in Building Portals with Oracle WebCenter Portal.

2. Import the downloaded task flow style into JDeveloper. For more information, see Section 9.6.2, "How to Import a Portal Resource into JDeveloper."

3. In the Application Navigator, right-click the imported task flow style, and choose Open.

4. Make any changes to the code of the task flow style.

5. Save your changes.

6. Export the task flow style. For more information, see Section 9.6.3, "How to Export a Portal Resource from JDeveloper."

7. Upload the exported task flow style into the deployed application. For more information, see the "Uploading an Asset" section in Building Portals with Oracle WebCenter Portal.

12.3.2 What You May Need to Know About Developing Task Flow Styles

To expose a task flow style for use when creating task flows at runtime, you must consider the following requirements:

- Create only one view fragment for a task flow that will be used as a task flow style. The view fragment JSFF file must be created in the same location as the task flow definition XML file.

- Add a Panel Customizable component to the view fragment from the Composer tag library. This ensures that task flows based on this task flow style can be populated at runtime.

- Create a page definition for the view fragment, by right-clicking the JSFF file and choosing Go to Page Definition. The page definition for the view fragment must also be located in the same directory as the task flow definition XML file.

- If the task flow style refers to code, for example, using an EL value, then that code must be present in the deployed application.

12.3.3 How to Delete a Task Flow Style

If a task flow style is no longer required within your application, you can delete it.

When you delete a task flow style, it is no longer listed in the Create New Task Flow dialog at runtime. Deleting a task flow style has no impact on task flows that have already been created using the task flow style.
To delete a task flow style, right-click the task flow style in the Application Navigator and choose Delete from the context menu. For more information, see Section 9.4, "Deleting Portal Resources."
This chapter describes how to use Oracle JDeveloper to create and manage skins for your WebCenter Portal Framework applications.

This chapter includes the following topics:

- Section 13.1, "Introduction to Developing Skins"
- Section 13.2, "Best Practices for Developing Skins"
- Section 13.3, "Creating a Skin"
- Section 13.4, "Editing a Skin"
- Section 13.5, "Applying a Skin to Your Application"
- Section 13.6, "Conditionally Changing Skins for Users"
- Section 13.7, "Deploying a Skin as a Separate Shared Library"
- Section 13.8, "Troubleshooting Problems with Skins"

13.1 Introduction to Developing 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 Portal Framework applications. For more information about ADF Faces skins, see the "ADF Faces Skins" section in the Web User Interface Developer’s Guide for Oracle Application Development Framework.

Seeded Skin

When you create a Portal Framework application, a seeded skin, portal-skin.css, is added to your application.

You can edit this seeded skin to meet your particular requirements, or you can create your own. If you choose to create your own skin, the seeded skin is still a useful tool for you to discover the different things you can achieve in your own skin.
Best Practices for Developing Skins

Runtime Management
Portal Framework applications support the runtime administration of skins to help users continue developing a portal even after it has been deployed. With runtime administration, authorized users can manage an application’s skins, and create new ones, in a browser-based environment, with no requirement to install or understand JDeveloper. For more information, see Section 9.5, "Working with Portal Resources at Runtime."

Round-Trip Development
You can bring skins that have been created or modified at runtime back into JDeveloper, to ensure that any changes are not lost if the portal is redeployed. You can also edit these skins to further enhance them if necessary, and upload them back into the deployed portal. For more information, see Section 9.6, "Working with Round-Trip Development."

13.2 Best Practices for Developing Skins
When skinning WebCenter Portal, you can look at it from coarse-grain and fine-grain standpoint. At the coarse-grain level many large elements on the page, such as the background and the center of the page, can use very basic styling techniques to impart a look and feel to particular corporate brand with very little effort. At the fine-grain level you can apply the styling to specific components and controls within the page. The most efficient way to develop your skin is to start by defining the coarse-grain elements and then use fine-grain styling to tune your overall look and feel to be inline with your corporate brand.

In many cases a hybrid model of styling works very well. Taking the coarse-grained elements (page background, main portion of the body, and so on) and using traditional CSS approaches with those, but then getting specific using the ADF skinning, will work together to generate the overall appearance for WebCenter Portal.

An example of a hybrid approach would be using technology in WebCenter Portal to generate a menu that uses unordered lists and list items, then applying traditional CSS to them. Even though you are benefiting from WebCenter Portal's navigation models, you are not doing traditional ADF skinning – but instead using standard CSS.

Expression Language (EL) allows you to access the various objects for navigation within your template design. Looping in EL is simple and coding is done inline with the page template. You can mix regular HTML directly into the looping markup.

13.2.1 Externalizing Static Assets
When using coarse-grained techniques in addition to ADF styling, it is often helpful to hold various styling assets outside of WebCenter Portal. To do this you can use WebCenter Content to manage all of the unstructured assets for WebCenter Portal. They can include things like CSS and images that you want manage within your environment and provides revision control and workflow. This is a best practice if you want to allow design teams to access and work with WebCenter Portal without involving the development team for each and every change.

13.3 Creating a Skin
By default, Portal 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 topics:

- Section 13.3.1, "How to Create a CSS File"
- Section 13.3.2, "How to Define Skin Style Selectors"
- Section 13.3.3, "What You May Need to Know About Skin-Related Artifacts"

13.3.1 How to Create a CSS File

For information about creating an ADF Faces skin, see the "How to Add a Custom Skin to an Application" section in the Web User Interface Developer's Guide for Oracle Application Development Framework.

**Note:** When you create a CSS file, by default, JDeveloper places it under the `Application_Root/Portal/public_html/css` folder.

However, skins created at this default location are not included on the Assets page of the runtime administration console, and thus cannot be managed at runtime.

To enable authorized users to manage a skin at runtime, you must create it in the following directory:

`Application_Root/Portal/public_html/oracle/webcenter/portalapp`

By default, Portal Framework applications include a separate `skins` directory.

For more information, see Section 9.5.1.2, "How to Include a Portal Resource on the Assets Page."

13.3.2 How to Define Skin Style Selectors

After creating the CSS file, you must define the required ADF Faces skin selectors for the 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;
}
```

For information about:

- ADF Faces skin selectors in general, refer to the "Skin Style Selectors" section in the 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 the "Defining Skin Style Properties" and "Changing the Style Properties of a Component” sections in the Web User Interface Developer’s Guide for Oracle Application Development Framework.

13.3.3 What You May Need to Know About Skin-Related Artifacts

Oracle recommends that you store asset-related artifacts, such as images and icons, on your content server and that you create a folder structure on your content server specifically for asset artifacts so that content is easy to identify and move, if required.
13.4 Editing a Skin

You can make changes to your skin after its initial creation by editing the CSS file to edit or add style selectors. You can also edit the seeded portal skin by editing the portal-skin.css file.

To edit a skin:

1. In the Application Navigator, right-click the skin that you want to edit, for example portal-skin.css, and choose Open.

2. In the source code for the skin, define the required ADF Faces skin selectors for the components in your application. For example, to set a font size of 16px for your application content, you can use the .AFDefaultFont:alias skin selector as follows:

   For more information, see Section 13.3.2, "How to Define Skin Style Selectors."

3. Save the CSS file.

13.5 Applying a Skin to Your Application

Every Portal Framework application defines a default skin that determines the appearance of all the pages within the application.

When you first create a Portal Framework application, the seeded skin, portal, is set as the application's default skin. If you subsequently create your own skin, you can set it as the default instead.

You can set the default skin for a Portal Framework application by editing the oracle.webcenter.portalapp.skin preference in the adf-config.xml file.

---

**Note:** If runtime administration is enabled, authorized users can change the skin at runtime. For information, see the "Changing the Skin for a Portal" section in Building Portals with Oracle WebCenter Portal.

---

To apply a skin to your application:

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.skin

4. Set the value attribute to the family name of the skin that you want to apply to your application, for example:

   value="portal"

   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 skin.
5. Optionally, in the `desc` property, specify a description of the skin you want to apply. At runtime, this value shows up as the display name of the skin.

6. The `display` property specifies whether the skin is listed in the skin picker at runtime. Example 13–1 shows an example of the complete preference element.

Example 13–1  The Default Skin ADF Preference

```xml
<portal:preference id="oracle.webcenter.portalapp.skin"
    desc="Default Portal Skin" values="portal"
    resourceType="Skin" display="true"/>
```

7. Save the `adf-config.xml` file.

Note: If you want to apply a skin created at a 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 "How to Register a Custom Skin" section in the Web User Interface Developer's Guide for Oracle Application Development Framework.

13.6 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:

```xml
<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 the 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

13.7 Deploying a Skin as a Separate Shared Library

Oracle WebCenter Portal provides various default skins. All Oracle WebCenter Portal skin artifacts are available in the shared library, `oracle.webcenter.skin`. The shared
Deploying a Skin as a Separate Shared Library

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 Portal Framework applications and Portlet Producer applications. Therefore, all skins are available through the classpath. When you use a portlet in your Portal 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 Portal Framework application and the Portlet Producer application.

However, there may be a case where you modified portal-skin.css in your Portal Framework application. So, the skin available in the shared library is not the same as the updated version available in the Portal Framework application. If the skin’s version in the Portal Framework application is not identical to the skin in the shared library, skin sharing between Portal 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 Portal Framework application uses the modified skin. The Portal 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 Portal 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 Portal Framework application.

Therefore, whenever you modify a skin or create a custom skin for your Portal Framework applications, you must deploy the new or modified skin as a 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 Portal 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 check box 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 Portal Framework application and the Portlet Producer application to reference the custom skin shared library.

```xml
<library-ref>
  <library-name>Custom_Library_Name</library-name>
  <specification-version>Library_Version</specification-version>
</library-ref>
```

10. Redeploy both the Portal 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.

---

13.8 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:

```xml
<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:

```xml
<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:

```xml
<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>
```

**Problem**

The weblogic log output includes the following message:

```
<SkinFactoryImpl> <getSkin> Cannot find a skin that matches family portal and version 1.1. We will use the skin portal.desktop
```

**Solution**

Edit your application's trinidad-config.xml file and remove the <skin-version> element.

```xml
<?xml version='1.0' encoding='UTF-8'?>
<trinidad-config xmlns="http://myfaces.apache.org/trinidad/config">
  <skin-family>#{preferenceBean.defaultTrinidadSkin}</skin-family>
  <skin-version>v1.1</skin-version>
</trinidad-config>
```
Developing Resource Catalogs

This chapter describes how to use Oracle JDeveloper to create and manage resource catalogs for your WebCenter Portal Framework applications.

This chapter includes the following topics:

- Section 14.1, "Introduction to Resource Catalogs"
- Section 14.2, "Creating a Resource Catalog"
- Section 14.3, "Editing a Resource Catalog"
- Section 14.4, "Selecting the Default Resource Catalog"
- Section 14.5, "Filtering Items in a Resource Catalog"
- Section 14.6, "Deleting a Resource Catalog"
- Section 14.7, "Defining Task Flow Parameters and Attributes of the Enclosing Show Detail Frames for Task Flows"
- Section 14.8, "Troubleshooting Problems with Resource Catalogs"

14.1 Introduction to Resource Catalogs

Resource catalogs provide a consolidated view of the contents of one or more otherwise unrelated repositories in a unified search and browse user interface. They provide a mechanism for defining and organizing all the resources available for inclusion in a page, page template, or task flow. Resources originate in their source repository and are then exposed through the resource catalog as shown in Figure 14–1.
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 Portal 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 2.2.2, "Installing the WebCenter Portal Extension for JDeveloper"
- Oracle JDeveloper online Help for more information about the Resource Palette and catalogs.

To locate this information in JDeveloper, from the Help menu, select Table of Contents. In Help Center, expand JDeveloper Basics and select Working with the Resource Palette.

In WebCenter Portal and deployed Portal Framework applications, runtime resource catalogs determine the resources available when using Composer to populate pages, task flows, and page templates.
Seeded Resource Catalog Configuration
When you add a Page Customizable component to a page in your application, a seeded 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 Portal 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 14–2.

![Default Resource Catalog](image)

*Figure 14–2  Default Resource Catalog*
In a non-Portal Framework 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 producers, the Portlets folder is hidden.

You can create your own resource catalogs or modify the seeded catalog and include components relevant to your business.

**Runtime Management**

Portal Framework applications support the runtime administration of resource catalogs to help users continue developing a portal even after it has been deployed. With runtime administration, authorized users can create and manage resource catalogs in a browser-based environment, with no requirement to install or understand JDeveloper. For more information, see Section 9.5, "Working with Portal Resources at Runtime."

**Round-Trip Development**

You can bring runtime-created or modified resource catalogs back into JDeveloper, to ensure that any changes are not lost when the portal is redeployed. You can also edit these resource catalogs to further enhance them if necessary, and upload them back into the deployed portal. For more information, see Section 9.6, "Working with Round-Trip Development."

### 14.2 Creating a Resource Catalog

If the seeded resource catalog created for your application does not meet your requirements, or if you want to configure multiple resource catalogs, you can create your own resource catalogs with all the required resources.

---

**Note:** 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.

---

To create a resource catalog:

1. In the Application Navigator, right-click the node where you want to create the resource catalog (typically, `/oracle/webcenter/portalapp/catalogs`) and choose **New**.
2. In the New Gallery, expand **Web Tier**, select **Portal Framework** and then **Resource Catalog**, and click **OK**.
3. In Create Application Resource Catalog dialog, in the **File Name** field, enter a name for the XML file that represents the resource catalog, for example, `users-catalog.xml`. 
4. In the **Directory** field, enter the full directory path of the location under which to create the resource catalog definition XML file.

**Note:** In a Portal Framework application, to expose the new resource catalog in the Assets page of the runtime administration console, you must create the resource catalog under the `Application_Root/Portal/public_html/oracle/webcenter/portalapp` directory. By default, Portal Framework applications include a separate `catalogs` directory.

In a non-Portal Framework application, create the resource catalog in the `Application_Root/Portal/src/portal` directory.

**Tip:** If you do not select this option now, you can add the resource catalog to the runtime administration console later by right clicking it and choosing **Create Portal Resource**.

5. Select **Create as Portal Resource** to enable users with the appropriate permissions to manage the resource catalog at runtime in the Assets page of the runtime administration console.

**Note:** This option is available in a Portal Framework application only if the directory you entered is the `Application_Root/Portal/public_html/oracle/webcenter/portalapp` directory or one of its subdirectories.

6. Click **OK**.

When you create a resource catalog in a Portal Framework application, a catalog definition file (an XML file) is created for the resource catalog in the specified directory. The catalog definition file opens in Design view.
The catalog definition file is initially empty, so you must edit it to design the structure and content of your resource catalog. For more information, see Section 14.3, "Editing a Resource Catalog."

**Tip:** If you do not want to add resources from scratch, as a starting point, you could copy the source code of the default-catalog.xml file into the Source view of your new catalog definition.

If you created the resource catalog under the appropriate directory and selected the Create as a Portal Resource option, the resource catalog is automatically available for runtime management. For more information, see Section 9.5, "Working with Portal Resources at Runtime."

### 14.3 Editing a Resource Catalog

To define the content of your resource catalog you can edit the seeded resource catalog, default-catalog.xml, to add the resources that you want users to be able to add to the pages and task flows in your application. The seeded resource catalog is created automatically when you create an application using the WebCenter Portal Framework Application template.

You can also create your own resource catalog and edit that. For more information, see Section 14.2, "Creating a Resource Catalog."

You can add resources to a resource catalog in two ways:

- Drag and drop resources from different areas in JDeveloper onto the resource catalog in Design view.
- Use the options on the Add new node menu in the Design view for the resource catalog (Figure 14–5).
In the Design view for a resource catalog you can also rearrange, modify, and delete existing resources.

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.

**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.

**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 includes the following topics:
- Section 14.3.1, "How to Define Connections to Resources"
- Section 14.3.2, "How to Drag and Drop a Resource into a Resource Catalog"
- Section 14.3.3, "How to Add a Link to a Resource Catalog"
- Section 14.3.4, "How to Add a Folder to a Resource Catalog"
- Section 14.3.5, "How to Add a Custom Component to a Resource Catalog"
- Section 14.3.6, "How to Add a Custom Folder to a Resource Catalog"
- Section 14.3.7, "How to Add a Custom Content Provider to a Resource Catalog"
- Section 14.3.8, "How to Add a Resource Catalog to Another Resource Catalog"
- Section 14.3.9, "How to Set Display Options for a Resource Catalog or Resource"
14.3.1 How to Define Connections to Resources

Many resources, such as WebCenter Portal tools and 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 4.2, "Preparing Your Framework Application for Tools and Services."

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 the 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.

14.3.2 How to Drag and Drop a Resource into a Resource Catalog

You can add the following resources to a catalog by dragging and dropping them onto the desired location in the Design view of the catalog definition file:

- **Portlets**, from the Application Resources pane or the Resource Palette.
- **Content**, from a WebCenter Content connection in the Application Resources pane or 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 15.4, "Adding Custom Task Flows to a Page."
- **Resource catalogs**, from the Application Navigator. This embeds one resource catalog inside another.

### 14.3.3 How to Add a Link to a Resource Catalog

You can add links to the following resources to a catalog:

- **Task Flow**: Link to a WebCenter Portal tools and services task flow (in the Resource Palette) or a custom task flow created in JDeveloper (in the Application Navigator).
- **Portlet**: Link to any registered portlet producer. See Chapter 63, "Consuming Portlets" for detailed information about registering portlet producers and adding portlets to a page.
- **Content**: Link to a file or directory from an existing WebCenter Content connection.
- **Other**: Link to a custom component that you create by providing the XML code for that component.

To add a link:

1. In the Application Navigator, right-click the resource catalog to which you want to add the link, and choose **Open**.
2. In the Design view for the resource catalog, in the Catalog section on the left side, select the node in the resource catalog under which you want to add the link.
3. Click the **Add new node** icon and choose **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.
4. The **Id** field is automatically populated with generated ID. You can keep this ID, but you will probably want to change it to something more descriptive. The ID must be unique within the catalog definition file.
5. From the **Type** list, select the type of resource that you want to create. You can choose **Taskflow**, **Portlet**, **Content**, or **Other**.

---

**Notes**: 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 15.4, "Adding Custom Task Flows to a Page."

---

6. If you are adding a link to a custom resource (that is, you selected **Other** from the **Type** list), then specify the factory class of the component. For all other resource types, the **Factory Class** value is automatically populated with the appropriate value for the selected link 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."

7. In the URL field, enter the URL to access the resource. If you do not know the location, click the Browse Files icon to browse for available resources. Table 14–1 provides the URL format to used for each resource type.

<table>
<thead>
<tr>
<th>Resource Type</th>
<th>Location in the Application</th>
<th>URL Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task Flow from an ADF Library</td>
<td>Resource Palette</td>
<td>taskflow://pathToTaskFlow/taskFlowDefinitionFileName#taskFlowId</td>
</tr>
<tr>
<td>Task Flow Within the Application</td>
<td>Application Navigator</td>
<td>taskflow://pathToTaskFlow/taskFlowDefinitionFileName#taskFlowId</td>
</tr>
<tr>
<td>Portlet</td>
<td>Portlet producer connection in the Application Resources pane or in the Resource Palette</td>
<td>portlet://producerId/portletId</td>
</tr>
<tr>
<td>Content</td>
<td>Content Repository connection in the Application Resources pane</td>
<td>content://contentConnectionId/documentId</td>
</tr>
</tbody>
</table>

Note: If you enter 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.

8. In the Visible field, specify whether the link is listed in the resource catalog at runtime. The default value is the EL expression #{true}, which means the link is listed for all users at all times. Enter false or #{false} to hide the link from all users at all times, or enter an EL expression to specify the conditions under which the link is listed. If the field is empty, the value defaults to true.

9. Optionally, in the URL Attributes section, specify any desired display options for the link. For more information, see Section 14.3.9, "How to Set Display Options for a Resource Catalog or Resource."

10. In the URL Parameters section, enter values, as desired, for any parameters supported by the resource. You can bind these parameters to other artifacts in the application.

Click the Add icon to create a row in the Parameters table.

11. Save the catalog definition file.

14.3.4 How to Add a Folder to a Resource Catalog

You can create folders to group similar resources, thereby organizing the catalog better. To add a folder:

1. In the Application Navigator, right-click the resource catalog to which you want to add the folder, and choose Open.
2. In the Design view for the resource catalog, in the Catalog section on the left side, select the node in the resource catalog under which you want to add the folder.

3. Click the Add new node icon and choose 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.

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 more descriptive. The ID must be unique within the resource catalog.

5. In the Visible field, specify whether the folder is listed in the resource catalog at runtime. The default value is the EL expression #{true}, which means the folder is listed for all users at all times. Enter false or #{false} to hide the folder from all users at all times, or enter an EL expression to specify the conditions under which the folder is listed. If the field is empty, the value defaults to true.

6. Optionally, in the Folder Attributes section, specify any desired display options for the folder. For more information, see Section 14.3.9, “How to Set Display Options for a Resource Catalog or Resource.”

7. In the Folder Parameters section, enter values, as desired, for any parameters supported by the folder. You can bind these parameters to other artifacts in the application.

Click the Add icon to create a row in the Parameters table.

8. Save the catalog definition file.

### 14.3.5 How to Add a Custom Component to a Resource Catalog

You can add custom component to a resource catalog, 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 resource catalog to which you want to add the custom component, and choose Open.

2. In the Design view for the resource catalog, in the Catalog section on the left side, select the node in the resource catalog under which you want to add the custom component.

3. Click the Add new node icon and choose Component.

A component element is added to the catalog. The right side of the page displays fields to define the new component's attributes and parameters.

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 more descriptive. The ID must be unique within the resource catalog.

5. In the Component Factory field, enter 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."

6. In the Visible field, specify whether the component is listed in the resource catalog at runtime. The default value is the EL expression #{true}, which means the component is listed for all users at all times. Enter false or #{false} to hide the component from all users at all times, or enter an EL expression to specify the conditions under which the component is listed. If the field is empty, the value defaults to true.

7. Optionally, in the Component Attributes section, specify any desired display options for the custom component. For more information, see Section 14.3.9, "How to Set Display Options for a Resource Catalog or Resource."

8. 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="#"

The # value ensures that a unique ID is generated dynamically for the component.

9. Save the catalog definition file.

14.3.6 How to Add a Custom Folder to a Resource Catalog

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. In the Application Navigator, right-click the resource catalog to which you want to add the custom folder, and choose Open.

2. In the Design view for the resource catalog, in the Catalog section on the left side, select the node in the resource catalog under which you want to add the custom folder.

3. Click the Add new node icon and choose Custom Folder.

   A customFolder element is added to the catalog. The right side of the page displays fields to define the new folder’s attributes and parameters.

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 more descriptive. The ID must be unique within the resource catalog.
5. In the **Initial Context Factory** field, enter 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."

6. In the **Visible** field, specify whether the folder is listed in the resource catalog at runtime. The default value is the EL expression #{true}, which means the folder is listed for all users at all times. Enter `false` or #{false} to hide the folder from all users at all times, or enter an EL expression to specify the conditions under which the folder is listed. If the field is empty, the value defaults to `true`.

7. Select **Insert Folder Contents** to display the contents of the custom folder directly in the catalog rather than displaying them under the folder.

8. Optionally, in the Custom Folder Attributes section, specify any desired display options for the folder. For more information, see Section 14.3.9, "How to Set Display Options for a Resource Catalog or Resource."

9. 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 drop-down menu on the **Add** icon if the factory class exposes parameters, for example, Content Presenter-specific factory classes provide a `templateView` parameter.

10. Save the catalog definition file.

### 14.3.7 How to Add a Custom Content Provider to a Resource Catalog

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. In the Application Navigator, right-click the resource catalog to which you want to add the custom content provider, and choose **Open**.

2. In the Design view for the resource catalog, in the Catalog section on the left side, select the node in the resource catalog under which you want to add the custom content provider.

3. Click the **Add new node** icon and choose **Custom Content**.

   A `customContent` element is added to the catalog. The right side of the page displays fields to define the new folder’s attributes and parameters.

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 more descriptive. The ID must be unique within the resource catalog.
5. In the Content Provider field, enter 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."

6. In the Visible field, specify whether the folder is listed in the resource catalog at runtime. The default value is the EL expression #true, which means the folder is listed for all users at all times. Enter false or #(false) to hide the folder from all users at all times, or enter an EL expression to specify the conditions under which the folder is listed. If the field is empty, the value defaults to true.

7. Optionally, in the Custom Content Attributes section, specify any desired display options for the folder. For more information, see Section 14.3.9, "How to Set Display Options for a Resource Catalog or Resource."

8. 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.

9. Save the catalog definition file.

14.3.8 How to Add a Resource Catalog to Another 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 to another resource catalog:

1. In the Application Navigator, right-click the resource catalog to which you want to add the resource catalog, and choose Open.

2. In the Design view for the resource catalog, in the Catalog section on the left side, select the node in the resource catalog under which you want to add the resource catalog.

3. Click the Add new node icon and choose Catalog Reference.

A catalog element is added to the catalog. The right side of the page displays fields to define the resource catalog's attributes and parameters.

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 more descriptive. The ID must be unique within the resource catalog.

5. In the Path field, enter 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.

6. In the Visible field, specify whether the folder is listed in the resource catalog at runtime. The default value is the EL expression #true, which means the folder is listed for all users at all times. Enter false or #(false) to hide the folder from all users at all times, or enter an EL expression to specify the conditions under which the folder is listed. If the field is empty, the value defaults to true.
7. Optionally, in the Catalog Attributes section, specify any desired display options for the resource catalog. For more information, see Section 14.3.9, "How to Set Display Options for a Resource Catalog or Resource."

8. Save the catalog definition file.

14.3.9 How to Set Display Options for a Resource Catalog or Resource

You can specify various display options for a resource catalog or any of its resources to determine their appearance and behavior. The display options available depend on the type of the resource.

■ **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 location of the icon to display next to the resource name in the catalog.

To set display options for a resource catalog or resource:

1. In the Application Navigator, right-click the resource catalog and choose **Open**.

2. In the Design view for the resource catalog, in the Catalog column on the left side, select the resource for which you want to set display options.

   To set display options for the resource catalog, select the root node.

3. In the resource's Attributes panel, click the **Add** icon and choose the attribute that you want to set.

4. From the **Value Type** drop-down 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 catalog definition file.

<table>
<thead>
<tr>
<th><strong>Table 14–2 Resource Catalog Display Options</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Attribute</strong></td>
</tr>
<tr>
<td>---------------</td>
</tr>
<tr>
<td>Title</td>
</tr>
<tr>
<td>Description</td>
</tr>
<tr>
<td>Subject</td>
</tr>
<tr>
<td>ToolTip</td>
</tr>
<tr>
<td>IconURI</td>
</tr>
</tbody>
</table>
14.3.10 How to Rearrange Resources in a Catalog

You can move resources to different folders within a resource catalog after you have added them. You can rearrange resources in the following ways:

- Drag and drop a resource 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. In the Application Navigator, right-click the resource catalog and choose Open.
2. In the Design view for the resource catalog, in the Catalog column on the left side, right-click the resource and choose Change Parent from the context menu.
3. In the Choose Parent dialog (Figure 14–6), select the folder in which to include the resource and click OK.

Figure 14–6 The Change Parent Dialog

![Choose Parent Dialog](image)

Note: The OK button is enabled only when you select a valid location for the resource.

4. Save the navigation model definition file.
14.3.11 How to Show or Hide a Resource Catalog or Resource

You can specify whether or not a resource catalog or one of its resources is visible at runtime.

**Tip:** As well as specifying an absolute true or false value, you can also dynamically determine whether or not the resource catalog or resource is available by using an EL expression to specify the conditions under which it is visible.

To show or hide a resource catalog or resource:

1. In the Application Navigator, right-click the resource catalog and choose **Open**.

2. In the Design view for the resource catalog, in the Catalog column on the left side, select the resource for which you want to set the visibility.

To set the visibility of the navigation model resource catalog, select the root node.

3. In the **Visible** field, specify whether the resource catalog or resource is available for use in the application. This field takes an EL value. The default is `#{true}`, which means the resource catalog or resource is visible. Enter `false` or `#{false}` to hide the resource catalog or resource, or enter an EL expression to specify the conditions under which it is available. If the field is empty, the value defaults to `true`.

4. Save the navigation model definition file.

14.3.12 How to Edit a Resource in a Resource Catalog

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 appropriate section earlier in this chapter.

14.3.13 How to Delete a Resource from a Resource Catalog

To delete a resource, select it in the Catalog section of the catalog definition file and click the **Remove selected node** icon (Figure 14–7).

*Figure 14–7 Delete Option in the Resource Catalog Design View*
14.3.14 How to Expose Data Controls Created at Design Time in a Resource Catalog

Oracle WebCenter Portal exposes a large number of service data controls that can be added to application pages and task flows. In addition, you may have created your own 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 Example 14–1 and Figure 14–8:

**Example 14–1 Custom Folder for Design Time Created Data Controls**

```xml
<customFolder id="dtDataControls"
   factoryClass="oracle.webcenter.datacomposer.internal.adapter.datacontrol.DTDataControlContextFactory">
   <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 14–8 Design Time Data Controls Folder**

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:
The `isKey` attribute defined on the attributes of a translatable resource:

```xml
<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.DTDataControlContextFactory">
    <attributes>
      <attribute value="DATACONTROL_FOLDER.TITLE" isKey="true" attributeId="Title"/>
      <attribute value="DATACONTROL_FOLDER.DESCRIPTION" isKey="true" attributeId="Description"/>
      <attribute value="DATACONTROL_FOLDER.KEYWORDS" isKey="true" attributeId="Subject"/>
      <attribute value="/adf/webcenter/folderlists_qualifier.png" isKey="false" attributeId="IconURI"/>
    </attributes>
  </customFolder>

  For more information about using resource bundles, see Section 20.12, "Configuring Runtime Resource String Editing."

  At runtime, the resource 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 Assets page of the runtime administration console 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 the "Adding a Resource to a Resource Catalog" section in Building Portals with Oracle WebCenter Portal.

Exposing Custom Data Controls in an Already Deployed Application

If you created a SQL or web service data control 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 55.2.5, "Deploying Extensions to the WebCenter Portal Shared Library (extend.spaces.webapp.war)."

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.

14.3.15 How to Expose Data Controls Created at Runtime in a Resource Catalog

Users can create data controls at runtime in the Assets page of the runtime administration console. 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 Example 14–2.
Example 14–2  Custom Folder for Runtime Created Data Controls

```xml
<customFolder id='rtDataControls'

factoryClass="oracle.webcenter.datacomposer.internal.adapter.datacontrol.DataContr
olContextFactory">

<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 14.3.14, "How to Expose Data Controls Created at Design Time in a Resource Catalog" 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.

14.3.16 How to Control Visibility of Portlets in a 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 visibility of portlets in a resource catalog:

1. Implement the PortletItemSecurityHelper interface in your own item security helper class, for example, CustomPortletSecurityHelper.java.

   Example 14–3 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.

   ```java
   Example 14–3  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});
}

try{
  canView = true;
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:

```xml
<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.

### 14.4 Selecting the Default Resource Catalog

You can configure the default catalog at the application and page levels. You can also use different resource catalogs depending on different criteria.

This section includes the following subsections:

- Section 14.4.1, "How to Select the Default Resource Catalog for an Application"
- Section 14.4.2, "How to Configure Multiple Resource Catalogs"
- Section 14.4.3, "How to Select the Default Resource Catalog for a Page"

#### 14.4.1 How to Select the Default Resource Catalog for an Application

Every Portal Framework application defines a default resource catalog that is used when users add content to a page, task flow, or page template.

When you first create a Portal Framework application, the seeded resource catalog, default-catalog.xml, is set as the application’s default resource catalog. If you subsequently create your own resource catalog, you can set that as the default.

You can set the default resource catalog for a Portal Framework application in the application’s adf-config.xml file. For more

To select the default resource catalog for an 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\RCASampleApp\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.

### 14.4.2 How to Configure Multiple Resource Catalogs

Most businesses 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. 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:

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 configure multiple resource catalogs:

1. Create your custom catalogs by performing the steps outlined in Section 14.2, "Creating a Resource Catalog" and Section 14.3, "Editing a Resource Catalog."

2. Implement the `oracle.adf.rc.model.config.ResourceCatalogSelector` interface in your own resource catalog selector class, for example, `CatalogSelector.java`.

Example 14–4 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 14–4  Sample ResourceCatalogSelector Showing Entries for Multiple Resource Catalogs**

```java
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 userahunold
            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 shown in the following example:

```xml
<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."

---

### 14.4.3 How to Select the Default Resource Catalog for a Page

If your application page contains Composer components, then you can specify 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 Component."

### 14.5 Filtering Items in a 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.

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.

**Tip:** You can conditionally hide individual resources in a catalog by specifying an EL expression in the Visible field for the resource. For more information, see Section 14.3.11, "How to Show or Hide a Resource Catalog or Resource."

To filter items in a resource catalog:

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, enter 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 14–5** shows a sample `CatalogDefinitionFilter` implementation. It is configured to hide the Discussions folder (forumsFolder) for user jdoe from the custom catalog, users-catalog.xml. The entire catalog, with the ADF Faces Components and Discussions folders, is shown to all other users.

**Example 14–5  Sample Showing CatalogDefinitionFilter Implementation**

```java
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. In the Application Navigator, right-click the resource catalog and choose **Open**.

8. In the Design view for the resource catalog, in the Catalog panel on the left side, select the root node of the resource catalog.

9. In the **Catalog Filter** field, enter the name of the Java class you created, for example, `webcenter.CatalogFilter`.

   This adds a `definitionFilter` attribute in Source view of the file.

10. Save the catalog definition file.

### 14.6 Deleting a Resource Catalog

If a resource catalog is no longer required within your application, you can delete it.
14.7 Defining 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:

   ```xml
   <url visible="#{true}"
       factoryClass="oracle.webcenter.portalframework.sitestructure.rc.TaskFlowResourceFactory"
       id="wallMiniView"
       url="taskflow://oracle/webcenter/peopleconnections/wall/controller/taskflows/WallViewer.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>
   ```
### 14.8 Troubleshooting Problems with Resource Catalogs

This section provides information to assist you in troubleshooting problems you may encounter while using resource catalogs.

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
```

For example:

```
taskflow://oracle/webcenter/peopleconnections/wall/controller/taskflows/WallViewer.xml#WallViewer
```
Creating Pages and Adding Resources

This chapter describes how to create pages in your Portal Framework applications. It also discusses the different resources you can add to those pages to build up the content of your application.

This chapter includes the following topics:
- Section 15.1, "Introduction to Pages"
- Section 15.2, "Creating Pages in a WebCenter Portal Framework Application"
- Section 15.3, "Adding WebCenter Portal Tools and Services Task Flows to a Page"
- Section 15.4, "Adding Custom Task Flows to a Page"
- Section 15.5, "Adding Content from WebCenter Content to a Page"
- Section 15.6, "Adding Portlets to a Page"
- Section 15.7, "Adding Pagelets to a Page"
- Section 15.8, "Adding Data Controls to a Page"
- Section 15.9, "Using External Applications"

15.1 Introduction to Pages

Pages are the fundamental building blocks of a Portal Framework application. You use the pages in a Portal Framework application to expose the different kinds of content you want to provide to your users. For more information, see Section 5.3.2, "Understanding Pages, Page Templates, and the Portal Page Hierarchy."

Pages can include content from various resources, such as:
- Portlets and pagelets
- WebCenter Portal task flows and custom task flows
- Content from UCM
- External applications
- Data controls

Adding content to the pages in your application is no different from adding content to any other web application.
15.2 Creating Pages in a WebCenter Portal Framework Application

The WebCenter Portal Framework Application template populates a new Portal Framework application with several default pages. You must add your own pages to develop your application beyond this initial framework.

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

1. In the Application Navigator, right-click the pages folder (/oracle/webcenter/portalapp/pages) in your application project and choose New.

   Tip: For pages to be used in the page hierarchy, they must be located under oracle/webcenter/portalapp. For more information, see Section 5.3.2, "Understanding Pages, Page Templates, and the Portal Page Hierarchy."

2. In the New Gallery, expand Web Tier, select JSF and then JSF Page, and click OK.

3. In the Create JSF Page dialog (Figure 15–1), in the File Name field, enter a file name for the page.

   Figure 15–1 The Create JSF Page Dialog


5. In the Initial Page Layout and Content section, select Page Template and select the page template that you want to use to define the layout and structure of the page. For more information on page templates, see Chapter 11, "Developing Page Templates."
After you have created the page, you can edit the code to enable dynamic selection of the page template. For more information, see Section 11.5.3, "How to Enable Dynamic Switching of Page Templates at Runtime in a WebCenter Portal Framework Application."

6. Click OK.

For information about the other settings in this dialog that are not specific to Portal Framework application pages, see the "Creating a View Page" section in Web User Interface Developer’s Guide for Oracle Application Development Framework.

For information about how to add general components, such as layout components, to the page, see the "How to Add ADF Faces Components to JSF Pages" section in Web User Interface Developer’s Guide for Oracle Application Development Framework.

To enable users to edit your application’s pages at runtime, you must add Composer components to it. For more information, see Chapter 18, "Enabling Runtime Editing of Pages Using Composer."

15.3 Adding WebCenter Portal Tools and Services Task Flows to a Page

WebCenter Portal tools and services enrich applications with enterprise 2.0 capabilities, including social computing services, personal productivity services, online awareness and communications, content integration, and web analytics. WebCenter Portal tools and services expose their functionality through task flows. For more information about WebCenter Portal tools and services, see Chapter 4, "Preparing Your Application for WebCenter Portal Tools and Services."

If you create your application using the WebCenter Portal Framework Application template, all the appropriate WebCenter Portal tools and services connection wizards and tag libraries are readily visible and available in the New Gallery and Component Palette. When you consume a WebCenter Portal tools and services task flow or component, the necessary libraries are automatically added to the project. Depending on the tool or 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 4, "Preparing Your Application for WebCenter Portal Tools and Services."

For more information about the individual WebCenter Portal tools and services and their task flows, see Table 15–1.

<table>
<thead>
<tr>
<th>WebCenter Portal Tools and Services</th>
<th>More Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activity Graph</td>
<td>Chapter 46, &quot;Integrating the Activity Graph&quot;</td>
</tr>
<tr>
<td>Analytics</td>
<td>Chapter 47, &quot;Integrating Analytics&quot;</td>
</tr>
<tr>
<td>Announcements</td>
<td>Chapter 32, &quot;Integrating Announcements&quot;</td>
</tr>
<tr>
<td>Blogs</td>
<td>Section 30.4, &quot;Integrating Blogs&quot;</td>
</tr>
<tr>
<td>Discussions</td>
<td>Chapter 33, &quot;Integrating Discussions&quot;</td>
</tr>
<tr>
<td>Documents</td>
<td>Chapter 28, &quot;Integrating Documents&quot;</td>
</tr>
<tr>
<td>Events</td>
<td>Chapter 48, &quot;Integrating Events&quot;</td>
</tr>
<tr>
<td>Instant Messaging and Presence (IMP)</td>
<td>Chapter 34, &quot;Integrating Instant Messaging and Presence&quot;</td>
</tr>
</tbody>
</table>
Adding Custom Task Flows to a Page

Table 15–1  (Cont.) WebCenter Portal Tools and Services - More Information

<table>
<thead>
<tr>
<th>WebCenter Portal Tools and Services</th>
<th>More Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Links</td>
<td>Chapter 43, &quot;Integrating Links&quot;</td>
</tr>
<tr>
<td>Lists</td>
<td>Chapter 49, &quot;Integrating Lists&quot;</td>
</tr>
<tr>
<td>Mail</td>
<td>Chapter 35, &quot;Integrating Mail&quot;</td>
</tr>
<tr>
<td>Notifications</td>
<td>Chapter 50, &quot;Integrating Notifications&quot;</td>
</tr>
<tr>
<td>People Connections</td>
<td>Part VI, &quot;Working with People Connections&quot;</td>
</tr>
<tr>
<td>Polls</td>
<td>Chapter 36, &quot;Integrating Polls&quot;</td>
</tr>
<tr>
<td>Recent Activities</td>
<td>Chapter 51, &quot;Integrating Recent Activities&quot;</td>
</tr>
<tr>
<td>RSS</td>
<td>Chapter 52, &quot;Integrating RSS&quot;</td>
</tr>
<tr>
<td>Search</td>
<td>Chapter 45, &quot;Integrating Search&quot;</td>
</tr>
<tr>
<td>Tags</td>
<td>Chapter 44, &quot;Integrating Tags&quot;</td>
</tr>
<tr>
<td>Wikis</td>
<td>Section 30.3, &quot;Integrating Wikis&quot;</td>
</tr>
<tr>
<td>Worklists</td>
<td>Chapter 41, &quot;Integrating Worklists&quot;</td>
</tr>
</tbody>
</table>

**Note:** If the WebCenter Portal tools and services task flows do not quite meet your requirements, you can customize them to change their look and feel or functionality. For example, you can add text, change labels, remove regions or components, or show additional attributes. For more information, see Chapter 23, "Customizing WebCenter Portal Tools and Services Task Flows."

### 15.4 Adding Custom Task Flows to a Page

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 for users to add to pages or task flows at runtime. For more information, see Section 14.3.2, "How to Drag and Drop a Resource into a Resource 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 the "Creating ADF Task Flows" part in *Fusion Developer’s Guide for Oracle Application Development Framework.*

You cannot directly upload a task flow created in and exported from JDeveloper into an already deployed application, but you can import a task flow created at runtime into JDeveloper, modify it, and export it back to the deployed application.

**Exposing Task Flows Created in JDeveloper to a Deployed Application**

If you create 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 included in the runtime Resource Registry and can be added to custom resource catalogs from there.
For the steps to deploy an application to a Portal Framework application as a shared library, see the "Deploying the Application to a WebLogic Managed Server" section in *Administering 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 instance, see Section 55.2.5, "Deploying Extensions to the WebCenter Portal Shared Library (extend.spaces.webapp.war)."

### 15.5 Adding Content from WebCenter Content to a Page

To provide access to the content in your WebCenter Content content repository, you must create a connection to the WebCenter Content. For more information, see Section 25.2.1, "Creating a Content Repository Connection Based on the Oracle Content Server Adapter."

With a connection to WebCenter Content established, you can provider users with a user-friendly interface to manage, display, and search documents at runtime. For more information, see Chapter 28, "Integrating Documents."

Next, add one of the content task flows or document components to your page to expose the content from WebCenter Content. Content task flows include Content Presenter and the documents 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
- AutoVue

Document components include:

- Document Link

**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 22.6.1, "Granting Permissions on Task Flows."
15.6 Adding Portlets to a Page

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 57, "Introduction to Portlets."

To add a portlet to your application, you must first register the portlet's producer with the Portal Framework application. For more information, see Section 63.2, "Registering Portlet Producers with a WebCenter Portal Framework Application."

After you have registered the portlet producer, you can then drag and drop any portlets that you want to include in your application onto the appropriate page. For more information, see Section 63.5, "Adding Portlets to a Page."

15.7 Adding Pagelets to a Page

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 pages, see Section 62.7, "Using Pagelets in Web Applications."

15.8 Adding Data Controls to a Page

Data controls are used to retrieve information from different data sources and create user interface (UI) components within your Portal 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 a data source to a JSF page, you must first create a connection to the data source, then use the connection to create a data control based on the repository.

You can create custom data controls to connect to different data sources and add them to your application pages or task flows. For detailed information about data controls, see the Fusion Developer’s Guide for Oracle Application Development Framework.

You cannot directly upload a data control created in and exported from JDeveloper into an already deployed application. Neither can you import a data control created at runtime into JDeveloper.

Some WebCenter Portal tools and services provide data controls for creating customized visualizations of the tool or service. For information about consuming
these data controls in your application, see Section 4.1.3, "Using WebCenter Portal Data Controls."

**Exposing Data Controls for Consumption in an Already Deployed Application**

The runtime Resource Registry in a Portal Framework application contains a Design Time Data Controls folder by default. This folder exposes the seeded WebCenter Portal tools and services 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 Portal Framework application, 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 Portal Framework application as a shared library, see the "Deploying Applications Using the WLS Administration Console" section in *Administering 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 instance, see Section 55.2.5, "Deploying Extensions to the WebCenter Portal Shared Library (extend.spaces.webapp.war)."

### 15.9 Using External Applications

If the resources you add to your pages 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 page might include portlets that point to applications that require a separate login. Or, if your page includes the Mail task flow, you must identify an external application to map the mail server user to the application user so that the user does not have to log in separately every time.

The first time a user accesses the external application, he or she is prompted for a user name and password for the external application. These credentials are then mapped to the 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 74.13, "Working with External Applications."
Part III

Customizing Your Application and Extending Customization Options

Part III contains the following chapters:

- Chapter 16, "Using WebCenter Portal Composer"
- Chapter 17, "Enabling Runtime Creation and Management of Pages"
- Chapter 18, "Enabling Runtime Editing of Pages Using Composer"
- Chapter 19, "Extending Runtime Editing Capabilities Using Composer"
- Chapter 20, "Performing Advanced Composer Configurations"
- Chapter 21, "Performing Composer-Specific MDS Configurations"
- Chapter 22, "Modifying Default Security Behavior of Composer Components"
- Chapter 23, "Customizing WebCenter Portal Tools and Services Task Flows"
Using WebCenter Portal Composer

This chapter introduces Oracle WebCenter Portal’s Composer and describes the various tasks it supports at runtime.

This chapter contains the following topics:

- **Section 16.1, "Introduction to Composer"**
- **Section 16.2, "Understanding Application and User Customizations"**
- **Section 16.3, "Understanding the Edit Mode and View Mode of a Page"**
- **Section 16.4, "Customizing Capabilities in Page View Mode"**
- **Section 16.5, "Editing Capabilities in Design View and Add Content View in Page Edit Mode"**
- **Section 16.6, "Editing Capabilities in Structure View in Page Edit Mode"**
- **Section 16.7, "Composer Components"**
- **Section 16.8, "Security and Composer"**

### 16.1 Introduction to Composer

Composer is a fully-integrated page editor for adding and editing page content at application 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 integrated development environment (IDE) for creating or editing applications. For Oracle 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.

Line of business (LOB) administrators want to add an "Expense Incurred by Self/Directs" task flow.

Line of business capabilities are administration capabilities that can be done by a business analyst or equivalent position. LOB administrators are able to perform administrative tasks that do not require top-level administration capabilities to complete.

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 redeploy 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.

16.2 Understanding 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 LOB members. 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 changes as either user customizations 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 21.1, "Introduction to MDS."

16.3 Understanding the Edit Mode and View Mode of a Page

When a user opens an application page in a browser, the page opens in View mode. Composer provides access to another mode called Edit mode to enable users with appropriate privileges to edit application pages. These modes are described in detail:

- **Edit mode**: The mode typically available to authenticated users with access privileges to edit page content. In addition to the standard 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.

- **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 that user alone and are saved as user customizations.

16.3.1 About 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 chapter 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 16.7, "Composer Components."

When you add the Change Mode Link or Change Mode Button component to a page that can be customized, an Edit link appears in View mode of the page in Composer. When users click the Edit link or button, the page opens in Edit mode of Composer. Figure 16–1 shows an Edit link on a page in View mode.
In Edit mode, the default Composer toolbar displays the title of the page being edited, Page Properties icon, Reset Page icon, and the Close button (Figure 16–2).

By default, the page Edit mode provides two views: Add Content and Structure. However, you can configure your pages to display any or all of the following views: Design, Add Content, Structure, Select, and Preview. You use the DesignViews attribute to specify the views to display in Edit mode. If you specify the value as all, all the views become available in Edit mode except Add Content view.

For enhanced functionality, it is recommended that you configure your pages to display all the views in Composer. For information about the DesignViews attribute used for specifying the views to display in Edit mode, see Table B–1, "Attributes of a Page Customizable Component".

This section contains the following subsections:

- Section 16.3.1.1, "About Add Content View in Edit Mode"
- Section 16.3.1.2, "About Structure View in Edit Mode"
- Section 16.3.1.3, "About Design View in Edit Mode"
- Section 16.3.1.4, "About Select View in Edit Mode"
- Section 16.3.1.5, "About Preview View in Edit Mode"

16.3.1.1 About Add Content View in Edit Mode

The Add Content view is the default view when you open a page in Composer. It provides a WYSIWYG rendering of the page and its content, where controls are directly selectable on each component.
16.3.1.2 About Structure View in Edit Mode

The **Structure view** provides a combined WYSIWYG and hierarchical rendering of page components, where controls are available in the toolbar of the page structure pane to add, edit, delete, hide, and rearrange page components (Figure 16–4).

---

**Note:** The `childCreation` attribute determines whether the children of popup dialogs can be viewed in Composer. By default, this attribute is set to `deferred` and users are unable to see the children of popup dialogs. Setting the value to `immediate` allows users to view the children of popup dialogs.

There is a performance impact by setting the `childrenCreation` attribute to `immediate`. If you do not need to use the popup on the page, using the default setting of `deferred` will avoid the penalty of CPU and memory usage when it is not needed. Note that the popup will be shown non-modally, thus, the application must be able to handle that fact without error.

For more information, see Section G.13, "EL Expressions Relevant to Composer" in Chapter G, "Expression Language Expressions."

---

**Figure 16–4  Structure View of Page Edit Mode**

![Structure View of Page Edit Mode](image)
16.3.1.3 About Design View in Edit Mode

The Design view provides a WYSIWYG rendering of the page and its content, where controls are directly selectable on each component. The resource catalog displays inline on the right side of the page, where you can select components to add to the page (Figure 16–5).

You can choose to hide the inline resource catalog, and instead show each area of the page with an Add Content button to help you identify where you can add components.

Design view is not available in Edit mode by default. You can enable it by setting the value of the DesignViews attribute of the Page Customizable component to design or all. For information, see Appendix B.1.1, “Page Customizable Component.”

If you choose to display Design view, it is rendered as the default view, and Add Content view is not displayed in Edit mode.

Figure 16–5  Design View of Composer

16.3.1.4 About Select View in Edit Mode

The Select view provides a WYSIWYG rendering of the page and its content, where you can select a component for quick access to its properties or the properties of its parent component. You cannot delete a component in Select view.

Just like Design view, the resource catalog displays inline on the right side of the page, where you can select components to add to the page (Figure 16–6).

In Edit mode, Select view is not available by default. You can enable it by setting the value of the DesignViews attribute of the Page Customizable component to select or all. For information, see Appendix B.1.1, "Page Customizable Component."
Customizing Capabilities in Page View Mode

Using WebCenter Portal Composer

16.3.1.5 About Preview View in Edit Mode

The Preview view displays a preview of the published page, as it will appear in your application.

In Edit mode, Preview view is not available by default. You can enable it by setting the value of the DesignViews attribute of the Page Customizable component to preview or all. For information, see Appendix B.1.1, "Page Customizable Component."

16.4 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 16.7, "Composer Components." Changes made to a page in View mode are available only to the user making these changes.

This section contains the following subsections:

- Section 16.4.1, "Rearrange Components"
- Section 16.4.2, "Change the Layout"
- Section 16.4.3, "Expand and Collapse Components"

Note: For a detailed description of the editing tasks that you can perform on a page, see the "Creating and Editing a Portal Page" chapter in Building Portals with Oracle WebCenter Portal.

16.4.1 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 16–7 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 16–7  Actions Menu on a Show Detail Frame

![Actions Menu on a Show Detail Frame](image)

16.4.2 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 16–8.

Figure 16–8  Change Layout Icon

![Change Layout Icon](image)

Figure 16–9 shows the layout options available with the Layout Customizable component. A gray colored border highlights the layout currently applied to the page or area.
16.4.3 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 16–10** shows the **Collapse** icons on Show Detail Frame components.

The **Expand** icon on a collapsed component enables users to disclose the component so that it displays the header and content. **Figure 16–11** shows the **Expand** icons on two collapsed Show Detail Frame components.
16.5 Editing Capabilities in Design View and Add Content View in Page Edit Mode

This section describes the editing capabilities that Composer provides in page Edit mode. It contains the following subsections:

- Section 16.5.1, "Add Content"
- Section 16.5.2, "Rearrange Page Content"
- Section 16.5.3, "Edit Component Properties and Parameters"
- Section 16.5.4, "Reset and Override Component Properties in the Current Layer"
- Section 16.5.5, "Edit Resource Strings"
- Section 16.5.6, "Show or Hide Components"
- Section 16.5.7, "Delete Components"
- Section 16.5.8, "Change the Layout"
- Section 16.5.9, "Edit Page Properties"
- Section 16.5.10, "Wire Components to Page Parameters"
- Section 16.5.11, "Reset Page"
- Section 16.5.12, "Reset Application Customizations on Detecting Errors"
- Section 16.5.13, "Customize the Display Name of Child Components"
- Section 16.5.14, "Create Labels On Saving Application Customizations"
- Section 16.5.15, "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 16.8, "Security and Composer."

For a detailed description of the runtime editing tasks that users can perform on a page, see the "Editing a Page" section in *Building Portals with Oracle WebCenter Portal*. 

---

**Figure 16–11 Expand Icon on a Show Detail Frame**

![Expand Icon on a Show Detail Frame](image1.png)

Support Home Page

- Latest News
- Press Release
16.5.1 Add Content

Authorized 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 Edit mode. Oracle 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 Edit mode of a page:

- In Add Content and Design views, by clicking the Add Content button on any container component on the page.

  In Design view, by default an inline resource catalog is available, but you can hide it to show each area of the page with an Add Content button to help you identify where you can add components.

- In Structure view, by selecting a container component and clicking the Add content into the selected component icon in the Structure view toolbar.

Notes: If you restrict customization on a container component, then users cannot add components inside it at runtime. See Section 22.1, "Applying Component-Level Restrictions by Defining Customization Policies" for more information.

Figure 16–12 shows the Add Content dialog in a Portal Framework application.
The Add Content dialog contains folders and components. An **Open** link next to an item in the catalog indicates that the item is a folder. Authorized 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 dialog 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. For more information, see Section 19.6, "Configuring Event Handlers for Composer UI Events."

Depending on its configuration, the dialog exposes all or a subset of the following components to authorized users:

**Note:** For information, see Chapter 14, "Developing Resource Catalogs."
ADF Faces Components

The Web Development 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 HTML Markup, Hyperlink, Image, Text, and Web Page components that are analogous to the JDeveloper design time components Output Text, Go Link, Go Image Link, Rich Text Editor, and Inline Frame respectively. In JDeveloper, these components are available in the ADF Faces tag library.

The Web Development folder in the catalog contains these components, as shown in Figure 16–13.

**Figure 16–13  ADF Faces Components You Can Add to a Page**

- Task Flows
Authorized users can add custom task flows and out-of-the-box WebCenter Portal tools and 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. Authorized 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 63, “Consuming Portlets.”

- **Documents**

  The Content Management folder lists documents and folders that users can add to the page. For more information, see the “Working with Web Development Components on a Page” section in Building Portals with Oracle WebCenter Portal.

### 16.5.1.1 Add a Tab Set or a Tab to a Box Component

The toolbar on the Box component provides an option (Figure 16–14) 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.

The Add a tab set or a tab icon is available when the showTabAction attribute of a Panel Customizable component is set to true.

**Figure 16–14  Add Tab Set or a Tab Icon on a Box Component**

![Add Tab Set or a Tab Icon on a Box Component](image)

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 Structure 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 16–15. Users can add any number of tabs in this way.
The first tab that a user adds will surround the selected Box component. Each subsequent tab will contain a new Box component inside it.

Authorized users can delete a tab by clicking the Delete icon on a tab in Add Content or 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 Structure 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.

### 16.5.1.2 Add Box Components Adjacent to Existing Ones

A toolbar on the Box component in Add Content and 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 22, "Modifying Default Security Behavior of Composer Components."

---

Users can delete Box components that they added using the Add icons, in Add Content, Design, and Structure views.

**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

16.5.1.2.1 Add Box Above  Clicking the Add box above icon (Figure 16–16) adds a Box component above the selected Box component. The layout attribute on the Panel Group Layout is set to vertical.

*Figure 16–16  Add Box Above Icon on a Box Component*

*Figure 16–17  Box Component Before Adding Another One Adjacent to It*

*Figure 16–18  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.

16.5.1.2.2 Add Box Below Clicking the Add box below icon (Figure 16–19) adds a Box component below the selected Box component. The layout attribute on the Panel Group Layout is set to vertical.

Figure 16–19 Add Box Below Icon on a Box Component

![Add Box Below Icon on a Box Component](image)

Users can click the Add box below icon repeatedly to add many contiguous Box components vertically within the same parent Panel Group Layout.

16.5.1.2.3 Add Box Left Clicking the Add box left icon (Figure 16–20) 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 16–20 Add Box Left Icon on a Box Component

![Add Box Left Icon on a Box Component](image)

Figure 16–21 shows the two Box components horizontally aligned in the component navigator.

Figure 16–21 Box Component Added to the Left

![Box Component Added to the Left](image)

Users can click the Add Box Left icon repeatedly to add many contiguous Box components horizontally within the same parent Panel Group Layout.

16.5.1.2.4 Add Box Right Clicking the Add box right icon (Figure 16–22) adds another Box component to the right of the selected Box component. The layout attribute on the Panel Group Layout is set to horizontal.
Users can click the **Add box right** icon repeatedly to add many contiguous Box components horizontally within the same parent Panel Group Layout.

### 16.5.2 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 Add Content and Design views, 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 16–23).

**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.
16.5.3 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.

---

**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 22, "Modifying Default Security Behavior of Composer Components."

- It is a component from the ADF Faces library

---

In Add Content and Design views, clicking the **Edit** icon on a component displays the Component Properties dialog, shown in Figure 16–24.
The Display Options tab enables users to edit visual properties, such as background and title. The Parameters tab, if available, enables users to edit parameters for components such as portlets and task flows.

**Graphs, Pie Graphs, and Gauges**
If the component is a Graph, Pie Graph, or Gauge component, additional panels are available to edit the properties of the graph.

**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 16–26 shows the two different icons used to indicate that the property was updated.
Editing Capabilities in Design View and Add Content View in Page Edit Mode

**Note:** Visual indication of property customization is relevant only if you have configured multiple customization layers in your application. For information, see Section 21.3, "Adding Customization Layers to View and Edit Modes: Example."

### Figure 16–26 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 16.5.4, "Reset and Override Component Properties in the Current Layer."

#### 16.5.4 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 16–27) 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.
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.

### 16.5.5 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) Portal Framework applications, see Section 20.12, "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 16–27. Users can select this option to edit resource strings for properties.

The Select Text Resource dialog (Figure 16–28) provides options to search existing resources, edit or delete resource keys, and add new resource key/value pairs.

**Figure 16–28  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 16–29), 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 key/value**: Create key and value pairs in the application override bundle.
Users can edit only strings that were created in the same MDS layer. Other strings appear as read-only.

**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.

- 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.
- 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 (Figure 16–30) and create a new string by specifying values for the Key, Display Value, and Description, and then click OK.
Users can just create inactive strings that get saved to the override bundle. Clicking the Use button in the last column selects a string for use as the current property value.

- Delete button: For deleting a selected resource from the search results table.
- 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.
- Refresh: For refreshing the search results table.
- 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 or identifies a string as the active resource (Figure 16–31).
**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:

```xml
<c:set var="portalBundle"
       value="${compBundle['test.resource.ComposerBundle']}"/>
```

---

**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.
16.5.6 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, deselect the Show Component check box (Figure 16–32).

**Figure 16–32  Show Component Option in the Component Properties Dialog**

Deselecting 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, deselect the check box shown against the component name to hide a component. (Figure 16–33)
16.5.7 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 22.1.1, "How to Define Type-Level Customization Policies."

16.5.8 Change the Layout

The Change Layout icon enables users to select a layout from a set of eight predefined layouts. The default page layout is threeColumn.

In Structure 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 16.7.5, "Layout Customizable."

16.5.9 Edit Page Properties

In the Page Properties dialog, users can create or edit a page's parameters and display properties. Figure 16–34 shows the Security tab in the Page Properties dialog.
On the Parameters tab, 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.

16.5.10 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 the "Wiring Pages, Task Flows, Portlets, and ADF Components" chapter in Building Portals with Oracle WebCenter Portal.

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 the "Passing Parameter Values Through the Page URL" section in Building Portals with Oracle WebCenter Portal.

16.5.11 Reset Page

The Reset Page icon is available on the page in Edit mode. This icon invokes the Reset Page dialog that provides options to remove edits made to a page in the current layer or top layer and reset it to a previously saved version or its original, out-of-the-box state.
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 21.2.1, "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 16–1 shows the sample code to create a label with a `SAVE_LABEL_` prefix.

**Example 16–1  Code to Create a Label Prefixed with SAVE_LABEL_**

```java
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);

16.5.12 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 16–36, provides the option to roll back only the last change or all changes that users made in the current context.

![Figure 16–36 Error Dialog Prompting Users to Discard Changes](image)

- If the change is not yet saved to MDS, an Error dialog, shown in Figure 16–37, simply informs users that an error occurred and the page will be reloaded.

![Figure 16–37 Error Dialog Informing Users About Page Reload](image)

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.

16.5.13 Customize the Display Name of Child Components

On a page, a component may be a parent to a number of child components. When you view the properties of the parent component, you can find its child components listed on the Child Components tab of the Component Properties dialog. When multiple child components are of the same type, they may be indistinguishable from each other.
in the list of child components. For example, a panelGroupLayout component uses the value of its layout attribute (for example, vertical) as the display name, so multiple panelGroupLayout components may all have the same name in the list of child components, as illustrated in Figure 16–38 in Structure view.

Figure 16–38  Child Component Names: Structure View and Component Properties Dialog

If you need to identify and work with a specific child component, you can edit the source code of a page to specify a unique display name for any component to make it easily distinguishable. This display name is shown in hint text and the Component Properties dialog. The custom attribute that implements this functionality is composer_name_hint, which you can specify using <f:attribute>, as follows:

Syntax:

```xml
<f:attribute name="composer_name_hint" value="display_name"/>
```

Example:

```xml
<af:panelGroupLayout id='pgl1' layout='scroll' inlineStyle=""/>
```
Editing Capabilities in Design View and Add Content View in Page Edit Mode

![Diagram showing Structure View and Component Properties Dialog](image)

Figure 16–39 shows the Child Components tab of the Component Properties dialog for a parent panelCustomizable (Box) component that includes four panelCustomizable (Box) child components.

Notice that the display name Box is used for the parent component and the four child components, making them indistinguishable.

To customize the display name of these components, you can edit the source code to add a composer_name_hint attribute to each component. The composer_name_hint value is exposed in the following locations:

- In Structure view, the composer_name_hint value shows as hint text for the component (Figure 16–40).
In Select view, right-clicking on a selected component displays a context menu showing the `composer_name_hint` value for the component and/or its parent component (Figure 16–41).

The Component Properties dialog for the component includes the `composer_name_hint` value in its title (Figure 16–42).

In the parent component's Component Properties dialog, the **Child Components** tab lists each child component using its `composer_name_hint` value as the display name (Figure 16–42).
The page source code for the parent Box (panelCustomizable) component and its four child Box components shown in Figure 16–42 looks like this:

```xml
<af:showDetailItem id="82127" text="Project XYZ"
xmlns:af="http://xmlns.server.com/adf/ faces/rich" disclosed="true"
inflexibleHeight="100">
  <cust:panelCustomizable id="45399" layout="scroll"
xmlns:cust="http://xmlns.server.com/adf/faces/customizable">
    <f:attribute name="composer_name_hint" value="Project XYZ content area"/>
  </cust:panelCustomizable>
  <cust:panelCustomizable id="273"
xmlns:cust="http://xmlns.server.com/adf/faces/customizable">
    <f:attribute name="composer_name_hint" value="header area"/>
  </cust:panelCustomizable>
  <cust:panelCustomizable id="959"
xmlns:cust="http://xmlns.server.com/adf/faces/customizable">
    <f:attribute name="composer_name_hint" value="dept area"/>
  </cust:panelCustomizable>
  <cust:panelCustomizable id="393"
xmlns:cust="http://xmlns.server.com/adf/faces/customizable">
    <f:attribute name="composer_name_hint" value="employee area"/>
  </cust:panelCustomizable>
  <cust:panelCustomizable id="941"
xmlns:cust="http://xmlns.server.com/adf/faces/customizable">
    <f:attribute name="composer_name_hint" value="project area"/>
  </cust:panelCustomizable>
</cust:panelCustomizable>
<af:showDetailItem>
```

### 16.5.14 Create Labels On Saving Application Customizations

A Save and Label button on the Composer toolbar, shown in Figure 16–43, 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 Component."

![Figure 16–43  Save and Label Button on the Composer Toolbar](image)

16.5.15 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 19.2.4, "How to Display the Customization Manager Add-On." If you configure your application to display the Customization Manager add-on, the Composer toolbar displays the Customization Manager icon, as shown in Figure 16–44.

![Figure 16–44  Customization Manager Button](image)

Clicking the Customization Manager button invokes the Customization Manager dialog, shown in Figure 16–45.
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 drop-down 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.
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 16–46 shows the Upload Customization dialog that lets users select the file they want uploaded.

---

**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 19.8, "Defining Property Filters."
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 16.5.14, "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 16–47.
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.

### 16.6 Editing Capabilities in Structure View in Page Edit Mode

Structure view (Figure 16–48) provides a WYSIWYG and a hierarchical rendering of page components in a component navigator. Add Content, Switch, Delete, Cut, and Paste controls are available on Structure view toolbar provide specific operations in Composer. In Structure view, users can access and modify properties of components that are not otherwise selectable in Design or Add Content view. For example, many ADF Faces components can be edited only in Structure view. Users can also edit components within a task flow.
By default, the component navigator displays on the right side of the page in Composer. Users can choose to display it at the top, bottom, or left, using the Dock menu (Figure 16–49). Users can also drag the border on the edge of the component navigator to alter its height or width.

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 at the bottom of the page displays a bread crumb trail for the selected component (Figure 16–50). 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 16–50  Bread Crumbs for a Component Selected in Page Structure 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 16.6.2, "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 and Add Content view, in Structure 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 Structure 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 Structure view:

- Section 16.6.1, "Rearrange Page Content Using Cut and Paste Options"
- Section 16.6.2, "Edit Content Inside a Task Flow"  
- Section 16.6.3, "Edit Content Inside a Declarative Component"
- Section 16.6.4, "Show or Hide Components"

Notes:

- The Save button is provided on the Composer toolbar only if the application is configured to use a sandbox. For more information about sandboxes, see Section 21.2, "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 16.8, "Security and Composer."
16.6.1 Rearrange Page Content Using Cut and Paste Options

Users can rearrange components on a page using the Cut and Paste options on the Structure view toolbar (Figure 16–51). A Paste menu enables users to paste a component into, before, or after any other component in the component navigator.

Figure 16–51 Cut and Paste Options on Structure 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 Structure 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 three columns. The Hard disk column in turn contains the Magnetic disks and Solid State columns inside a column group, as shown in Figure 16–52.
Users can switch to Structure 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).

Users can select a column and click Cut, then select another column and click Paste Before or Paste After.

### 16.6.2 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.

Structure 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 16–53.
Clicking the **Edit Task Flow** link results in Structure view zooming in to display the task flow’s page and its components, as shown in **Figure 16–54**.

**Figure 16–54  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 **Close** link next to an open task flow results in the component navigator displaying the page containing the task flow.

### 16.6.2.1 Reset Task Flow

The **Reset Task Flow** button on the Structure view toolbar (**Figure 16–55**) 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 16.5.11, "Reset Page."

Figure 16–55 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 21.2.1, "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 16–2 shows the sample code to create a label with a SAVE_LABEL_ prefix.
Example 16–2  Code to Create a Label Prefixed with SAVE_LABEL_

```java
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);
```

16.6.3 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 16.6.2, "Edit Content Inside a Task Flow."

Figure 16–56 and Figure 16–57 show the Edit Component and Close options available on declarative components.

Figure 16–56  Edit Option On a Declarative Component
16.6.4 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 16–58).

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.
16.7 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 18.1, "Designing Editable Pages Using Composer Components."

The Composer tag library in JDeveloper (Figure 16–59) is available from the Component Palette and provides components that you can add to make a page editable.

**Figure 16–59  Composer Tag Library in the Component Palette**

This section provides an overview of the Composer components that are used to enable page editing. It contains the following subsections:

- Section 16.7.1, "Page Customizable"
- Section 16.7.2, "Change Mode Link and Change Mode Button"
- Section 16.7.3, "Panel Customizable"
- Section 16.7.4, "Show Detail Frame"
- Section 16.7.5, "Layout Customizable"

For more information about these components and other Composer components such as Custom Action, see Appendix B, "Composer Component Properties and Files."

16.7.1 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 (Figure 16–60).
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 18.1.2, "How to Enable Runtime Customization Using a Page Customizable Component."

### 16.7.2 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 16–61 shows a Change Mode Link component on the page.

For more information about using Change Mode Link or Change Mode Button, see Section 18.1.3, "How to Enable Switching Between Page Modes Using a Change Mode Link or Change Mode Button."

### 16.7.3 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 16–62. You can use this button to open the resource catalog Viewer and add components within the Panel Customizable.

Figure 16–62  Panel Customizable Component

For more information, see Section 18.1.4, "How to Define Editable Areas of a Page Using Panel Customizable Components."

16.7.4 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 16–63). 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 16–63  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 20.1, "Enabling Custom Actions on a Show Detail Frame Component by Using Facets."

For information about adding this component to a page, see Section 18.1.6, "How to Enable Component Customization Using Show Detail Frame Components."
16.7.5 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 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 16–64 shows a **Layout Customizable** component and the predefined layouts that display when users click the layout changer.

![Figure 16–64  Layout Customizable Component](image)

For more information, see Section 18.1.5, "How to Enable Layout Customization for a Page Using a Layout Customizable Component."

16.8 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 honoring 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 16–1 describes Composer behavior to reflect application customization restrictions.

<table>
<thead>
<tr>
<th>Restriction</th>
<th>Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>The page (and its contents) is open for all customizations</td>
<td>Composer allows editing of components without restrictions.</td>
</tr>
</tbody>
</table>
Using WebCenter Portal Composer

Security and Composer

16.8 Security and Composer

Depending on the privileges granted while implementing security, users can perform different user and application customization tasks when they log in to the application.

<table>
<thead>
<tr>
<th>Restriction</th>
<th>Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>The page is restricted</td>
<td>Composer displays but none of the options, for example Page Properties and Edit icons, are accessible.</td>
</tr>
<tr>
<td>Some operations are restricted</td>
<td>Composer displays but the options corresponding to the restricted operations are disabled.</td>
</tr>
<tr>
<td>A component is restricted</td>
<td>The Edit icon is not rendered on the component and its properties cannot be edited.</td>
</tr>
<tr>
<td>A component’s attributes are restricted</td>
<td>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.</td>
</tr>
</tbody>
</table>

For information about overriding default security behavior, see Chapter 22, "Modifying Default Security Behavior of Composer Components."

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 16.8.1, "Page and Task Flow Security"
- Section 16.8.2, "MDS Customization Restrictions"
- Section 16.8.3, "Component Action-Level Security"

For information about overriding default security behavior, see Chapter 22, "Modifying Default Security Behavior of Composer Components."

16.8.1 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 16.4, "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 Oracle WebCenter Portal Customizable Components do not differentiate between Edit and Customize privileges.
Notes: Composer and Oracle 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 16–2 explains Composer behavior based on page- and task flow-level privileges. Only those privileges that are relevant to Composer and Oracle 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 16–2 Mapping of Page or Task Flow Privileges to Composer Behavior

<table>
<thead>
<tr>
<th>Privilege</th>
<th>Composer Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edit or Customize</td>
<td>Users can switch to Edit mode of the page, where Composer is invoked, and edit the page.</td>
</tr>
<tr>
<td></td>
<td>Users with either the Edit or Customize privilege can perform all runtime editing tasks.</td>
</tr>
<tr>
<td></td>
<td>With the Edit or Customize privilege on a page or task flow, users can:</td>
</tr>
<tr>
<td></td>
<td>■ Add content</td>
</tr>
<tr>
<td></td>
<td>■ Edit component properties</td>
</tr>
<tr>
<td></td>
<td>■ Rearrange content</td>
</tr>
<tr>
<td></td>
<td>■ Delete components in Composer</td>
</tr>
<tr>
<td></td>
<td>■ Personalize or customize a portlet</td>
</tr>
<tr>
<td></td>
<td>■ Move a portlet or task flow in View mode. This is persisted as a user customization.</td>
</tr>
<tr>
<td></td>
<td>■ Expand and collapse a task flow or portlet in View mode. This is persisted as a user customization.</td>
</tr>
<tr>
<td></td>
<td>■ Resize a column of a table in a task flow. This is persisted as a user customization.</td>
</tr>
<tr>
<td></td>
<td>■ Reset the page to its original state</td>
</tr>
<tr>
<td></td>
<td>■ Delete components in View mode</td>
</tr>
<tr>
<td></td>
<td>Note: You can perform all or some of these tasks depending on whether you have page-level or task flow-level privileges.</td>
</tr>
<tr>
<td></td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td>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 Structure view.</td>
</tr>
</tbody>
</table>
16.8.1.1 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 16–2.

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 16–65.
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 18.3.1.1, "Considerations for Adding Task Flows" and Section 22.6, "Implementing Task Flow Security."

For more information about task flows and their security behavior, see Fusion Developer’s Guide for Oracle Application Development Framework.

### 16.8.2 MDS Customization Restrictions

Portal 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 22.1, "Applying Component-Level Restrictions by Defining Customization Policies."

16.8.3 Component Action-Level Security

The 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 22.5, "Applying Action-Level Restrictions on Panel Customizable and Show Detail Component Actions."
Enabling Runtime Creation and Management of Pages

This chapter describes how to create and manage pages at runtime and also provides advanced information on managing pages.

This chapter contains the following topics:

- Section 17.1, "Introduction to Page Creation and Management"
- Section 17.2, "Creating Pages and Task Flows"
- Section 17.3, "Defining Values for the Page - Create New Task Flow Parameters"
- Section 17.4, "How to Create Task Flow View Pages"
- Section 17.5, "Managing Pages"
- Section 17.6, "Introduction to Custom Styles and Templates"
- Section 17.7, "Customizing Page Views"
- Section 17.8, "Advanced Information for Managing Pages"

17.1 Introduction to Page Creation and Management

You can 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 APIs.

After you create pages or task flows, users can view and manage them with either the page data control or the page 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 portal in WebCenter Portal).

Table 17–1 describes the page developer tools.

<table>
<thead>
<tr>
<th>Tool</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Page - Create New</td>
<td>A task flow for creating pages or task flows at runtime.</td>
</tr>
</tbody>
</table>
Creating Pages and Task Flows

Pages are integrated with many WebCenter Portal tools and services, such as links, search, and tags. You can track the most recent changes in pages with Recent Activities. Because it is preconfigured to work with Recent Activities, pages automatically produce the information that Recent Activities uses to display the most recent additions, changes, or deletions in pages.

### 17.2 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 APIs.

<table>
<thead>
<tr>
<th>Tool</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Page Data Control</td>
<td>A data control for viewing or deleting information about listed pages and task flows at runtime. Page Data Control is available at design time after you add the page libraries to your application. By default, the page libraries are added to the application when you use the WebCenter Portal Application template or when you add the Page - Create New task flow.</td>
</tr>
<tr>
<td>Page APIs</td>
<td>APIs for creating and managing pages and task flows at runtime.</td>
</tr>
</tbody>
</table>

Pages are integrated with many WebCenter Portal tools and services, such as links, search, and tags. You can track the most recent changes in pages with Recent Activities. Because it is preconfigured to work with Recent Activities, pages automatically produce the information that Recent Activities uses to display the most recent additions, changes, or deletions in pages.

**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 APIs along with the Page - Create New task flow. The page APIs provide the flexibility of customization within the pages.

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 and templates.

Page styles 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’s look and feel. You can select the page style 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 17–1* shows the Create Page dialog at runtime.
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 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 "Editing a Page" section in Building Portals with Oracle WebCenter Portal.

17.2.1 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 17.2.1.1, "How to Add the Page - Create New Task Flow"
- Section 17.2.1.2, "Setting Security for Pages in JDeveloper"
- Section 17.2.1.3, "How to Create Pages at Runtime"
- Section 17.2.1.4, "Structure of Pages Created at Runtime"
- Section 17.2.1.5, "How to Access Pages Created at Runtime"

17.2.1.1 How to Add the Page - Create New Task Flow

To add the Page - Create New task flow to your Portal Framework application:

1. Create a page. Follow the steps described in Section 15.2, "Creating Pages in a WebCenter Portal Framework Application."

2. Open the customizable page to which you want to add the task flow.


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 the Create Page button appears on the page.

9. Click Create Page.

The Create Page dialog opens (see Figure 17–1).

10. Enter a page name, then select the page template and the page style.

11. Click Create.

A page is created based on the selected information. For more details on where the page is created, see Section 17.2.1.4, "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 17.3, "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 17.5, "Managing Pages."

---

**17.2.1.2 Setting Security for Pages in JDeveloper**

Pages do not require their own, specific configuration of ADF security. They follow the security set up for the Portal 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 4.2.1.1, "Implementing Security for Tools and Services."

2. After you add the Page - Create New task flow to a page, locate the jazn-data.xml file in Application Resources, and double-click its name.

3. Click Resource Grants, then select Task Flow from the Resource Type drop-down list.

4. Select the Show task flows imported from ADF libraries check box.

5. In the Resources column, select the page-create-page task flow.

6. In the Granted To column, click the Add icon to add a role to which to grant task flow access privileges.

7. Select the required role, and click OK.

   The newly added role is displayed in the Granted To list.

8. In the Actions column, select the privileges to grant to the selected role (Figure 17–2).
Figure 17–2  Adding Privileges for the Page - Create New Task Flow

9. From Resource Type, select Web Page to apply security to the page on which you added the Page - Create New task flow.

Follow the same procedure outlined in steps 5 through 8: select a page, add roles, and apply privileges.

Note: The Page Data Control data control does not require any security setting for a secure or non-secure application.

17.2.1.3 How to Create Pages at Runtime

For detailed information, see the “Creating and Editing a Portal Page” chapter in Building Portals with Oracle WebCenter Portal.

17.2.1.4 Structure of Pages Created at Runtime

The pages you create at runtime using the Page - Create New task flow or the page APIs are stored in the mds folder, located under your application root. Table 17–2 provides examples of the default formats used for page and task flow file names.

| Table 17–2  Default File Name Formats |
|------------|--------------------------------------|
| File Type  | Format                              |
| Page       | Page<N>.jspx (value of N starts from 1) |
|            | For example: Pagel.jspx, Page2.jspx, Page<N>.jspx |
| Page Definition | PagelPageDef.xml, Page2PageDef.xml, Page<N>PageDef.xml |
| Task Flow View pages | Pagel.jsff, Page2.jsff, Page<N>.jsff |
| Task Flow View Page Definitions | PagelPageDef.xml, Page2PageDef.xml, Page<N>PageDef.xml |
| Task Flow Definitions | Pagel.xml, Page2.xml, Page<N>.xml |

When you use the page 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 (Pagel.jspx) and its associated page definition (PagelPageDef.xml) created at runtime by the user user1.
Creating Pages and Task Flows

Example 17–1  Page Created Under a Default Scope
mds\oracle\webcenter\page\scopedMD\s8bba98ff_4cbb_40b8_beee_296c916a23ed\user\user1\Page1.jspx

Example 17–2  Page Definition Created Under a Default Scope
mds\pageDefs\oracle\webcenter\page\scopedMD\s8bba98ff_4cbb_40b8_beee_296c916a23ed\user\user1\Page1PageDef.xml

Example 17–3  Page Created Under a Custom Scope
mds\oracle\webcenter\page\scopedMD\<scopeGUID>\Page1.jspx

Example 17–4  Page Definition Created Under a Custom Scope
mds\pageDefs\oracle\webcenter\page\scopedMD\<scopeGUID>\Page1PageDef.xml

In Example 17–4, the scopeGUID is a unique ID assigned to a specific scope by pages.

Example 17–5  Task Flow View Page Created Under a Default Scope
mds\oracle\webcenter\page\scopedMD\s8bba98ff_4cbb_40b8_beee_296c916a23ed\user\user1\Page1.jspx

Example 17–6  Task Flow Definition Created Under a Default Scope
mds\oracle\webcenter\page\scopedMD\s8bba98ff_4cbb_40b8_beee_296c916a23ed\user\user1\Page1.xml

Example 17–7  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 17–8  Task Flow View Page Created Under a Custom Scope
mds\oracle\webcenter\page\scopedMD\<scopeGUID>\Page1.jspx

Example 17–9  Task Flow Definition Created Under a Custom Scope
mds\oracle\webcenter\page\scopedMD\<scopeGUID>\Page1.xml

Example 17–10  Task Flow Page Definition Created Under a Custom Scope
mds\pageDefs\oracle\webcenter\page\scopedMD\<scopeGUID>\Page1PageDef.xml

In Example 17–10, the scopeGUID is a unique ID assigned to a specific scope by the pages.

17.2.1.5 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
17.3 Defining Values for the Page - Create New Task Flow Parameters

This section describes how to access and define values for the Page - Create New task flow parameters. It contains the following subsections:

- Section 17.3.1, "How to Access Page - Create New Task Flow Parameters"
- Section 17.3.2, "Setting Scope in a Page - Create New Task Flow"
- Section 17.3.3, "Setting an Outcome Parameter"
- Section 17.3.4, "Specifying Styles"
- Section 17.3.5, "Specifying an ADF Template"
- Section 17.3.6, "Showing a Command Link"
- Section 17.3.7, "Displaying an Image"
- Section 17.3.8, "Customizing the Label"

17.3.1 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, then select Go to Page Definition.
2. In the Executables section, scroll to select the pagecreatepage 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 17–3.

**Figure 17–3 Page Data Binding Definition**

The Edit Task Flow Binding dialog opens, as shown in Figure 17–4.
The following optional Page - Create New parameters are available:

- **showtemplate**
  
  By default, the Page Template drop-down list appears in the Create Page task flow at application runtime. You can choose to hide the page templates option by setting `showtemplate` to `false`.

- **scopename**
  
  For information, see Section 17.3.2, "Setting Scope in a Page - Create New Task Flow."

- **outcome**
  
  For information, see Section 17.3.3, "Setting an Outcome Parameter."

- **templatefile**
  
  For information, see Section 17.3.4, "Specifying Styles."

- **adftemplate**
  
  For information, see Section 17.3.5, "Specifying an ADF Template."

- **uietype**
  
  For information, see Section 17.3.6, "Showing a Command Link."

- **icon**
  
  For information, see Section 17.3.7, "Displaying an Image."

- **label**
  
  For information, see Section 17.3.8, "Customizing the Label."

### 17.3.2 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 17-5.
Defining Values for the Page - Create New Task Flow Parameters

In Figure 17–5, PageViewBean is a custom managed bean that is invoking the `getScopeName()` method to fetch the value of `scopeName`. It uses the page API. By doing this, the Page - Create New task flow creates all pages and task flows under the specified scope, rather than the 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 17–11  Creating a Scope for Application Pages**

```java
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;
    }
}
```

17.3.3 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 pages.

```java
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());
        }
    }
}
```
17.3.4 Specifying Styles

Pages provide out-of-the-box styles that are packaged within the page 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 17.3.4.1, "Out-of-the-Box Styles"
- Section 17.3.4.2, "Custom Styles"

Note: For more information about styles, see Section 17.6, "Introduction to Custom Styles and Templates."

17.3.4.1 Out-of-the-Box Styles

The following out-of-the-box-styles are provided (see Figure 17–1):

- Blank
- Left Narrow
- Right Narrow
- Stretch
- Text Page
- Three Column
- 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 Portal 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 libraries. For example:

  name="/oracle/webcenter/page/pstemplates/TemplateThreeColumn.jspx"

  TemplateThreeColumn.jspx is the page within the page library that defines the Three Column Template style.

- **title**: The value you enter for title is used as the name 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, the default icons assigned to each out-of-the-box style are used.

Example 17–12 shows the content of a sample `default_pageservice_styles.xml` file that lists the other out-of-the-box styles:

**Example 17–12  Sample default_pageservice_styles.xml File**

```xml
<?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 17.3.1, "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:

```java
oracle_webcenter_page_createpage_templatefile = ${'./mystyles/default_pageservice_style.xml'}
```

6. Click OK.

**17.3.4.2 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 17.6, "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
<?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"
</templateDef>
```

17-12  Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper
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` parameter, 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 17.3.5, "Specifying an ADF Template."

5. Open the Edit Task Flow Binding dialog as described in Section 17.3.1, "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}'`

### 17.3.5 Specifying 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 17.3.1, "How to Access Page - Create New Task Flow Parameters."

2. In the Value column next to the `oracle_webcenter_page_createpage_adftemplate` parameter, enter the path to the ADF template.

   For example:
   
   `$(templates/portalTemplate.jspx)'`
In Figure 17–6, an ADF template called portalTemplate.jspx is used.

**Figure 17–6  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.

### 17.3.6 Showing a Command Link

By default, the Page - Create New task flow is rendered as a command *button* (Figure 17–7). You can instead specify that it is rendered as a command *link* (Figure 17–8).

**Figure 17–7  Command Button**

![Create Page](Create Page)

**Figure 17–8  Command Link**

Create Page

To render the Page - Create New task flow as a command link:

1. Open the Edit Task Flow Binding dialog as described in Section 17.3.1, "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:

   ```javascript
   oracle_webcenter_page_createpage_uitype = `${'link'}`
   ```

3. Click OK.

### 17.3.7 Displaying an Image

You can associate an icon with the Page - Create New task flow button or link.

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 17.3.1, "How to Access Page - Create New Task Flow Parameters."

2. Revise the oracle_webcenter_page_createpage_icon parameter to the required value.
   For example:
   
   oracle_webcenter_page_createpage_icon = ${'/images/page.jpg'}

3. Click OK.

17.3.8 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 17–9 shows the task flow rendered as a command button with a custom label.

Figure 17–9 Custom Label

To change the command button or link label:

1. Open the Edit Task Flow Binding dialog as described in Section 17.3.1, "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.

17.4 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 17.3.1, "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
<?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 17.3.4.1, "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, for 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 (that you created in Section 17.3.4.2, "Custom Styles"), and then click Create.

    The new task flow view page is rendered.

    For information about where the task flows are created, see Section 17.2.1.4, "Structure of Pages Created at Runtime."

---

**17.5 Managing Pages**

You can manage pages using either the Page data control or the page APIs. This section contains the following subsections:
17.5.1 Using the Page Data Control to Manage Pages
You can customize page views with the Page data control. The data control enables you to view information about existing pages at runtime and delete any of the listed pages. The Page data control, Page Data Control, is included in page libraries and is available at design time after you add the libraries to the application.

For more information about the Page data control, see Section 17.8.2, "Using the Page Data Control.”

17.5.2 Using the Page APIs to Manage Pages and Task Flows
You can use 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 APIs, see Section 17.8.1, "Using the Page Java APIs.”

17.6 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 17–1). 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 17.6.1, "How to Create Templates for Pages Created at Runtime”
- Section 17.6.2, "Using ADF Templates”
- Section 17.6.3, "Creating Styles for the Create Page Dialog”

Note: For more information about the templates.xml file, see Section 17.3.4, "Specifying Styles.”
17.6.1 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 17.6.1.1, "How to Create a Page or Page Fragment Style"
- Section 17.6.1.2, "How to Edit Create Page Dialog Styles"

17.6.1.1 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 template is copied and a new page is created 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 17–13 shows the sample code of a page fragment style, pstempletview.jsff.

**Example 17–13  Sample Code of a New Page Style**

```xml
<?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'>
```
Introduction to Custom Styles and Templates

Enabling Runtime Creation and Management of Pages

<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>
                    <f:facet name="editor">
                        <pe:pageEditorPanel id="pep1"/>
                    </f:facet>
                </af:panelStretchLayout>
            </pe:pageCustomizable>
        </f:facet>
    </af:pageTemplate>
</af:form>
</af:document>

Alternatively, the JSPX or JSFF template file that you are creating can be based on an ADF template.

Example 17–14 shows the sample code of an ADF template.

Note: The page definition file of the JSPX page, which defines the style, must contain the following syntax for the id and Package attributes. These values are replaced with the relevant values when a page is created based on a custom style.

```xml
<?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 17–14 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 17–14  Sample Code of a Page Style that uses an ADF Template

```xml
<?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.jspx"
        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>
              <f:facet name="editor">
                <pe:pageEditorPanel id="pep1"/>
              </f:facet>
            </af:panelStretchLayout>
            <f:facet name="editor">
              <pe:pageEditorPanel id="pep1"/>
            </f:facet>
            <f:facet name="editor">
              <pe:pageEditorPanel id="pep1"/>
          </f:facet>
        </f:facet>
      </af:pageTemplate>
    </af:form>
  </af:document>
</f:view>
```

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.
Introduction to Custom Styles and Templates

Enabling Runtime Creation and Management of Pages

Example 17–15 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 Web User Interface Developer’s Guide for Oracle Application Development Framework.

Example 17–15 Sample Code of a Page Style Based 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']}"
                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>
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 17.3.4, "Specifying Styles."

Example 17–16 shows a sample templates.xml file.

**Example 17–16 templates.xml File**

```xml
<?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>
```

17.6.1.2 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:
Introduction to Custom Styles and Templates

Enabling Runtime Creation and Management of Pages

17-23

<templateDef name="pstempletview.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:

```xml
<templateDef name="pstempletview3.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.

17.6.2 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 17.3.4, "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 17–15 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 `af:pageTemplate` tag in the page template is searched and the `viewId` attribute is updated 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 17.6.2.1, "How to Specify the ADF Template Name Using the Task Flow Parameter"
- Section 17.6.2.2, "How to Specify the ADF Template Name Using API Parameters"
17.6.2.1 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**.

17.6.2.2 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:
   ```java
   PageService.createPage(
       String pageType, String nameFormat, String title,
       String pageTemplate, String pageTemplatePath,
       String ADFTemplateViewID,  
       String cssStyle, String schemeBGImage, String schemeBGCColor)
   ```

   where **ADFTemplateViewID** is the ID of the ADF template view activity.

   To create a task flow:
   ```java
   PageService.createTaskflow(
       String nameFormat, String title,  
       String pageTemplate, String pageTemplatePath,  
       String ADFTemplateViewID,  
       String cssStyle, String schemeBGImage, String schemeBGCColor)
   ```

   where **ADFTemplateViewID** is the ID of the ADF page template view activity.

17.6.3 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, 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 17.6.3.1, "How to Create a Style for the Create Page Dialog"
- Section 17.6.3.2, "How to Reference a New Style from the Page - Create New Task Flow"

17.6.3.1 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 17–17 shows the code entered into sample XML file templates2.xml that references the ptemplateview.jsff fragment you created in Section 17.6.1.1, "How to Create a Page or Page Fragment Style."

   
   ```xml
   <?xml version="1.0" encoding="UTF-8" ?>
   <templatesDef xmlns="http://xmlns.oracle.com/webcenter/page">
     <templateDef name="ptemplateview.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.

17.6.3.2 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.

5. Click OK.

6. Click Save All to save your work.

17.7 Customizing Page Views

This section describes various ways to customize page views. It contains the following subsections:

- Section 17.7.1, "Rendering Pages with ADF Faces Components"
- Section 17.7.2, "Managing User Security on Pages and Task Flows"

17.7.1 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 17.7.1.1, "Rendering Pages as Tabs Using ADF Faces Components"
- Section 17.7.1.2, "Rendering Pages as Links Using ADF Faces Components"
- Section 17.7.1.3, "Rendering Pages as Image Links Using ADF Faces Components"

17.7.1.1 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 of pages as tabs. It contains the following subsections:

- Section 17.7.1.1.1, "Rendering Pages as Tabs Using af:navigationPane"
- Section 17.7.1.1.2, "Rendering Pages as Tabs Using af:panelTabbed"

17.7.1.1.1 Rendering Pages as Tabs Using af:navigationPane  Example 17–18 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 API, see the getPages() method in page API section.
Example 17–18  Rendering Pages as Tabs Using af:navigationPane

```xml
<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 17–18 renders as a series of tabs (Figure 17–10).

Figure 17–10  Render Pages as Tabs

![Tabs](image)

17.7.1.1.2  Rendering Pages as Tabs Using af:panelTabbed  Example 17–19 illustrates how pages can be rendered as tabs using panelTabbed components.

Here a method called pages from a custom bean (MyPageServiceBean) is used to get the list of pages.

Example 17–19  Rendering Pages as Tabs Using af:panelTabbed

```xml
<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 17–19 renders as a series of tabs (Figure 17–11).

Figure 17–11  Render Pages as af:panelTabbed Component

![Tabs](image)

17.7.1.2  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 contains the following subsections:

- Section 17.7.1.2.1, "Rendering Pages as Links Using af:commandLink"
- Section 17.7.1.2.2, "Rendering Pages as Links Using af:goLink"

17.7.1.2.1  Rendering Pages as Links Using af:commandLink  Example 17–20 illustrates how pages can be rendered as links using af:commandLink.
Example 17–20  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 17–20 renders as a list of links (Figure 17–12).

Figure 17–12  Render Pages as Links

List Of Pages
My Page
Wiki Page
Documents
Blogs
Announcements
Discussions

17.7.1.2.2  Rendering Pages as Links Using af:goLink  Example 17–21 illustrates how pages can be rendered as links using af:goLink.

Example 17–21  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 17–21 renders as a list of links (Figure 17–12).

17.7.1.3  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 17–22 illustrates how pages can be rendered as image links.

Example 17–22  Rendering Thumbnail Views of Pages

<af:panelGroupLayout id="p11" layout="scroll">
  <af:panelGroupLayout id="pgl1" 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="#{$MyPageServiceBean.pageCount}"
        end="#{$MyPageServiceBean.pageCount}"
      >
        <af:panelGroupLayout id="sp11" 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>

</af:forEach>
</af:panelGroupLayout>
In Example 17–22, 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 APIs.

In Example 17–22, <image_name> is the name of the image with which you associate your page.

The output of the code in Example 17–22 is rendered as a linked series of thumbnail views of a page (Figure 17–13).

Figure 17–13  Render Pages as Image Links

17.7.2 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 APIs.

For more information about runtime security, see the “Setting Page Security” section in Building Portals with Oracle WebCenter Portal.

For more information about setting page access with the page APIs, see Java API Reference for Oracle WebCenter Portal.

17.8 Advanced Information for Managing Pages

This section describes the APIs and data controls available for working with pages. It contains the following sections:

- Section 17.8.1, "Using the Page Java APIs"
- Section 17.8.2, "Using the Page Data Control"
17.8.1 Using the Page Java APIs

This section provides information about page 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 17.8.1.1, "Configuration Settings Required for Using Page APIs"
- Section 17.8.1.2, "Introduction to the Page APIs"
- Section 17.8.1.3, "How to Set Up Your Application to Use the Page APIs"
- Section 17.8.1.4, "Example: How to Create a Page"

17.8.1.1 Configuration Settings Required for Using Page APIs

To use the page APIs, you must perform the following steps:

1. Confirm that you have the following entries in your `adf-config.xml` file:

   ```xml
   <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:

   ```xml
   <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.

17.8.1.2 Introduction to the Page APIs

The page APIs are located in the `oracle.webcenter.page.model.PageService` class. The following APIs are available for pages:
Advanced Information for Managing Pages

- createPage - see Section 17.8.1.4, "Example: How to Create a Page"
- deletePage
- createTaskflow - see Example 17–23, "Using Page APIs to Create a Task Flow View Page"
- deleteTaskflow
- createScope
- deleteScope
- copyPage
- getPageList - see Example 17–24, "Using Page APIs to Get a List of Pages"
- setPageTitle - see Example 17–25, "Using Page APIs to Set the Page Title"
- changeHiddenStatus
- changePageScheme - see Example 17–26, "Using Page APIs to Set the Page Scheme"
- setPageAccess

For more information about the page APIs, see Java API Reference for Oracle WebCenter Portal.

**Example 17–23  Using Page APIs to Create a Task Flow View Page**

```java
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 17–24  Using Page APIs to Get a List of Pages**

```java
PageDef getPageList(String path)
```

where
path is 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 17–25  Using Page APIs to Set the Page Title**

```java
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 17–26  Using Page APIs to Set the Page Scheme**

```java
setPageScheme(String pagepath, String schemeName, String schemeBGImage, String schemeBGColor, String otherCSS)
```

where
- **pagepath** is the path of the page for which you want to change the scheme.
- **schemeName** is the name of the CSS for the page.
- **schemeBGImage** is the background image for the custom scheme.
- **schemeBGColor** is the background color for the custom scheme.
- **otherCSS** refers to any other CSS attributes you want to specify.

It updates the page's scheme properties in the JSPX page.

**17.8.1.3 How to Set Up Your Application to Use the Page APIs**

To use the page APIs, Oracle WebCenter Portal's page 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 Portal - Page Service, as shown in **Figure 17–14**.
After confirming that WebCenter Portal - Page Service is included with Oracle WebCenter Portal's extensions, you must add the page libraries to the project.

To add page libraries to your project:

1. Right-click the Portal project and select Project Properties and then Libraries and Classpath.
2. Click Add Library.
3. Select the WebCenter Page Service and WebCenter Page Service View libraries, as shown in Figure 17–15.

**Figure 17–14  JDeveloper Extensions**

![JDeveloper Extensions](image)

**Note:** For information about adding Oracle WebCenter Portal’s extensions, see Chapter 2, “Setting Up Your Development Environment.”
4. Click OK.

If you do not want to use JDeveloper as the code editor, then you can set 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`

### 17.8.1.4 Example: How to Create a Page

This section provides an example of how to create a page using the page 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 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 17–27 shows how to use the `createPage()` API to create a scope named `MyScope`, create an MDS session instance, instantiate the page 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 17–27  Sample Showing How to Create a Page

```java
... 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
    // 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 jsp file.
    PageDef newPage = mPageService.createPage(
        "personaluserpage", pageNameFormat, pageTitle,
        "MyPageTemplate1.jspx", 
        "*/mytemplates/",
        null, null, null);
}

...
The Page Data Control 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 17.2.1.1, “How to Add the Page - Create New Task Flow”).
- Added the Page Data Control as described in Section 4.1.3, "Using WebCenter Portal Data Controls.”.

2. In the Data Controls panel of the Application Navigator, expand Page Data Control, as shown in Figure 17–16.

*Figure 17–16  Page 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 Edit Table Columns dialog, click OK.

6. In the Edit Action Binding dialog, enter the `scopeName` parameter to view all the pages listed under the specified scope. Click OK.

   If you do not enter any value, the data control takes the default scope. In Figure 17–17, the scope value is fetched from a custom managed bean.
7. Add a new column to the table you created in step 4.

8. To add a column for deleting pages at runtime, drag the `deletePage(string)` method and drop it in the new column.

9. From the Create menu, select Method, and then select ADF Link.

10. In the Parameters section of the Edit Action Binding dialog, shown in Figure 17–18, specify the following value for the `pagePath` parameter:

    ```
    #{bindings.PageTreeNode.treeModel.rowData.pagePath}
    ```
11. Click OK.

12. From the Data Controls panel, expand Page Data Control, getPageTree(), PageTreeNode, and then Operations.

13. Drag Execute and drop it on the page as an ADF Button.

14. In the Property Inspector, 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.

15. 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"/>

16. Save your page, and run it to your browser.

17. The created data control table should look something like Figure 17–19.
### 17.8.2.2 How to View, Edit, and Delete Pages at Runtime

To view a page at runtime, follow the link generated from step 16 in Section 17.2.1.1, "How to Add the Page - Create New Task Flow." 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 step 8 in Section 17.2.1.1, "How to Add the Page - Create New Task Flow." 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 "Working with Portal Pages" part in *Building Portals with Oracle WebCenter Portal*.

---

**Figure 17–19 Sample Data Control Table**

<table>
<thead>
<tr>
<th>Title</th>
<th>pageName</th>
<th>pagePath</th>
<th>creator</th>
<th>lastModified</th>
<th>hidden</th>
<th>deletePage</th>
</tr>
</thead>
<tbody>
<tr>
<td>myPersonalPage</td>
<td>Page1.jsp</td>
<td>/oracle/webcenter/px</td>
<td>steve</td>
<td>10/7/2008</td>
<td>false</td>
<td>deletePage</td>
</tr>
<tr>
<td>FinancePage</td>
<td>Page2.jsp</td>
<td>/oracle/webcenter/px</td>
<td>steve</td>
<td>10/7/2008</td>
<td>false</td>
<td>deletePage</td>
</tr>
<tr>
<td>HRPage</td>
<td>Page3.jsp</td>
<td>/oracle/webcenter/px</td>
<td>steve</td>
<td>10/7/2008</td>
<td>false</td>
<td>deletePage</td>
</tr>
<tr>
<td>MarketingPage</td>
<td>Page4.jsp</td>
<td>/oracle/webcenter/px</td>
<td>steve</td>
<td>10/7/2008</td>
<td>false</td>
<td>deletePage</td>
</tr>
<tr>
<td>OperationsPage</td>
<td>Page5.jsp</td>
<td>/oracle/webcenter/px</td>
<td>steve</td>
<td>10/7/2008</td>
<td>false</td>
<td>deletePage</td>
</tr>
</tbody>
</table>
This chapter describes how to add Composer components to your Portal Framework application pages to enable runtime editing of the pages.

This chapter contains the following topics:

- Section 18.1, "Designing Editable Pages Using Composer Components"
- Section 18.2, "Designing Editable Pages Using Composer Components: Example"
- Section 18.3, "Populating Pages with Content"
- Section 18.4, "Troubleshooting Composer Problems"

18.1 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 Portal Framework applications and non-Portal 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 16.7, "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 18.1.1, "How to Create a Customizable Page"
- Section 18.1.2, "How to Enable Runtime Customization Using a Page Customizable Component"
- Section 18.1.3, "How to Enable Switching Between Page Modes Using a Change Mode Link or Change Mode Button"
- Section 18.1.4, "How to Define Editable Areas of a Page Using Panel Customizable Components"
18.1.1 How to Create a Customizable Page

For information about creating a customizable JSPX page in your Portal Framework application, see Section 15.2, "Creating Pages in a WebCenter Portal Framework Application."

When you create a new page in JDeveloper, it is listed in the Application Navigator under Web Content as shown in Figure 18–1. Additionally, the page is opened in the editor and becomes the active editor panel.

Security Considerations

By default, a Portal 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 74.4, "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 74.3, "Configuring ADF Security."

### 18.1.2 How to Enable Runtime Customization Using a Page Customizable Component

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 18.1.9, "What Happens When You Add Composer Components."

---

**Note:** For considerations you must make before adding Page Customizable to a page, see Section 18.1.11, "What You May Need to Know When Designing Editable Pages."

---

To add a Page Customizable component to a page:

1. Open a customizable JSPX page.
2. In the Component Palette, select ADF Faces from the drop-down list and drag the 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 from the drop-down list.
4. Add a Page Customizable component to the center facet of Panel Stretch Layout.

   You must ensure that Page Customizable is nested inside an af:form element. 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.

   For information about defining or modifying attribute values, see Section B.1.1, "Page Customizable Component."

6. Save the page.

   Under the Page Customizable component, by default, a Panel Customizable component is added as a child component and Page Editor Panel and various other facets are added, as shown in Figure 18–2.
18.1.3 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 *Java API Reference for Oracle WebCenter Portal*.

For things you should consider before adding a Page Customizable component to a page, see Section 18.1.11, "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 from the Component Palette.

You must ensure that 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.

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."

Notes:

- If you have not used 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 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 Change Mode Link or Change Mode Button but no Page Customizable component.

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 18–3.
18.1.4 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 **Page Customizable** to a page, see Section 18.1.11, “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** from the drop-down list.
2. Drag a **Panel Customizable** component from the Component Palette to the Structure window and drop it at any suitable location within the form. You must ensure that 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 the 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.

---

### 18.1.5 How to Enable Layout Customization for a Page Using a Layout Customizable Component

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 from the drop-down list.
2. Drag a **Layout Customizable** component to the Structure window and drop it inside the Panel Customizable component.

**Notes:** If you select the 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.
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 18–5. 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 component contain child components.
18.1.6 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** component 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 18–6**.

   **Figure 18–6  Show Detail Frame Component**

   ![Show Detail Frame Component Diagram](image)

   For a better understanding of this type of task, see **Section 18.2, "Designing Editable Pages Using Composer Components: Example."**

### 18.1.7 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, from the Composer tag library, 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 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 18–7.

Figure 18–7  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.

18.1.8 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.
Designing Editable Pages Using Composer Components

18.1.9 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 14, "Developing Resource Catalogs."

In a Portal 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 20.11, "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 component 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 20.11, "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 Oracle WebCenter Portal extension JAR file. Example 18–2 shows the code in the page definition file after a Page Customizable component is added to an application page.

Example 18–2  Page Definition File After Adding Page Customizable Component

```xml
<?xml version="1.0" encoding="UTF-8" ?>
```
18.1.10 What Happens at Runtime

At runtime, users can perform all the tasks described in Section 16.4, "Customizing Capabilities in Page View Mode" and Section 16.5, "Editing Capabilities in Design View and Add Content View in Page Edit Mode."

Each Composer component provides runtime capabilities that are described in Section 16.7, "Composer Components."

---

**Note:** Runtime customizations that you perform on a page are not carried over when you deploy the application to a target server.

---

18.1.11 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 74.4, "Using the Role Manager Task Flow."
- To enable runtime customization of components, add only one Page Customizable component 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 \textit{ID} attribute is defined on all components on the page. Runtime editing of components that have no \textit{ID} value is not supported in Composer.

If your page includes components with no \textit{ID} value, then you may encounter problems while editing the page in Composer.

To enable View mode user customization, place a \texttt{Show Detail Frame} component within a \texttt{Panel Customizable} component.

Use a \texttt{Show Detail Frame} to enclose a single child only. If you must enclose multiple components in a \texttt{Show Detail Frame}, then place a grouping component, such as a \texttt{Panel Group Layout} or \texttt{Panel Customizable}, within the \texttt{Show Detail Frame} component and then place the ADF Faces components or other content within this grouping component.

Portlets need not be placed within \texttt{Show Detail Frame} components. Portlets come equipped with a header and display options that are similar to \texttt{Show Detail Frame} components.

### 18.2 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 \texttt{MyPage.jspx} in a Portal Framework application named \texttt{MyWebCenterApp}.

To create a customizable page:

1. Create a Portal Framework application named \texttt{MyWebCenterApp} by performing the steps in Chapter 6, "Creating WebCenter Portal Framework Applications."

2. Create a JSPX page named \texttt{MyPage.jspx} by performing the steps described in Section 15.2, "Creating Pages in a WebCenter Portal Framework Application."

3. Add a \texttt{Panel Stretch Layout} from the ADF Faces tag library to \texttt{MyPage.jspx}.

   \begin{quote}
   \textbf{Note:} The \texttt{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.
   \end{quote}

4. Add \texttt{Page Customizable} from the Composer tag library to the center facet by following the steps in Section 18.1.2, "How to Enable Runtime Customization Using a Page Customizable Component."

5. Set the border color of \texttt{Page Customizable} to blue to differentiate the editable area from other noneditable areas.

   Under the Style category in the Property Inspector for \texttt{Page Customizable}, click the Box tab and set the \texttt{Border Color} attribute to Blue.

6. Set the border color of child \texttt{Panel Customizable} to red.

   Under the Style category in the Property Inspector for \texttt{Panel Customizable}, click the Box tab and set the \texttt{Border Color} attribute to Red.

7. Add a \texttt{Change Mode Link} component to the top facet of the \texttt{Panel Stretch Layout} by performing the steps in Section 18.1.3, "How to Enable Switching Between Page Modes Using a Change Mode Link or Change Mode Button."
8. In the center facet, in the Panel Customizable that is a direct child of the Page Customizable, add a Layout Customizable component by following the steps in Section 18.1.5, "How to Enable Layout Customization for a Page Using a Layout Customizable Component."

9. Add Show Detail Frame components inside each Panel Customizable on the page by performing the steps in Section 18.1.6, "How to Enable Component Customization Using Show Detail Frame Components."

   Figure 18–8 shows Show Detail Frame components inside Panel Customizable that is a direct child of Layout Customizable, and inside Panel Customizable components that are direct children of the ContentA and ContentB facets.

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 18–8.
11. Grant Edit or Customize permission on the page to the required users and roles. For more information about granting permissions, see Section 74.4, "Using the Role Manager Task Flow."


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 16.5, "Editing Capabilities in Design View and Add Content View in Page Edit Mode."

Figure 18–9 shows how the page appears in Edit mode at runtime.
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:

   For more information, see Section 74.3, "Configuring ADF Security."

2. Create three users ahunold, sking, and jdoe.
   For the detailed procedure, see Section 74.2, "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.

### 18.3 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.

#### 18.3.1 What You May Need to Know When Adding Content to a Page

Consider the following when adding content to your editable page:
18.3.1 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 Chapter 22.6, "Implementing Task Flow Security"

18.4 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 “Testing and Debugging ADF Components” chapter in 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/controller/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:

```java
import oracle.adf.model.dvt.objects.DvtElementObjectFactory;
import oracle.adf.model.managers.FactoryManager;
...
static
{  
  FactoryManager.getInstance().registerFactory(new DvtElementObjectFactory());
}
```
This chapter describes how to use Composer's declarative and programmable extensibility mechanism to customize runtime capabilities to suit your business needs.

This chapter contains the following topics:

- Section 19.1, "Overview of Extensibility Options"
- Section 19.2, "Creating Composer Add-Ons"
- Section 19.3, "Creating Custom Property Panels"
- Section 19.4, "Extending the Expression Builder"
- Section 19.5, "Configuring Custom LOVs or Pickers for Task Flow Parameters"
- Section 19.6, "Configuring Event Handlers for Composer UI Events"
- Section 19.7, "Configuring Drop Handlers in the Resource Catalog"
- Section 19.8, "Defining Property Filters"
- Section 19.9, "Enabling Parameter Support on the Customization Manager Task Flow"
- Section 19.10, "Customizing the Composer Toolbar"
- Section 19.11, "Enabling Direct Select in Design or Add Content View"
- Section 19.12, "Troubleshooting Problems with Composer Extensibility Features"

### 19.1 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 19.1.8, "Configuration Files."

This section describes the options available for extending Composer runtime capabilities declaratively. It contains the following subsections:

- Section 19.1.1, "Composer Add-Ons"
- Section 19.1.2, "Composer Custom Property Panels"
- Section 19.1.3, "Composer Events Handlers"
- Section 19.1.4, "Drop Handlers."
19.1.1 Composer Add-Ons

The default Composer toolbar (Figure 19–1) includes the following:

- **Page title**: Shows the title of the page opened in Edit mode.
- **Page Properties** icon: Opens the Page Properties dialog, allowing the user to edit metadata for the host page.
- **Reset Page** icon: Reverts any changes made to the page in the current edit session.
- **Close** button: Exits Edit mode and returns the user to the standard UI.

![Figure 19–1 Default Add-Ons on the Composer Toolbar](image)

The Page Properties and Reset Page icons are examples of add-ons. Click these icons to display panels for editing page properties and resetting page customizations.

Typically, add-ons are custom task flows that are rendered as icons 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 icon on the Composer toolbar. Clicking this icon 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 19.2, "Creating Composer Add-Ons."

19.1.2 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 19.3, "Creating Custom Property Panels."

19.1.3 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,
Overview of Extensibility Options

Extending Runtime Editing Capabilities Using Composer

19.1.4 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 handlers 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 19.7, "Configuring Drop Handlers in the Resource Catalog."

19.1.5 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 19.8, "Defining Property Filters."

19.1.6 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 19.9, "Enabling Parameter Support on the Customization Manager Task Flow."

19.1.7 Composer Toolbar Customization

The default Composer toolbar displays the following elements: page name, Page Properties and Reset Page icons, and the Close button, as shown in Figure 19–2.

Figure 19–2 Composer Toolbar with Default Elements

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 19.6, "Configuring Event Handlers for Composer UI Events."
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 19.10, "Customizing the Composer Toolbar."

### 19.1.8 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."

### 19.2 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 icon 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 icon 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 16.5.11, "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 icon on the Composer toolbar. Clicking this icon would display the page’s revision history.

This section contains the following subsections:

- Section 19.2.1, "How to Create and Register Add-Ons"
- Section 19.2.2, "What Happens at Runtime"
- Section 19.2.3, "How to Exclude Composer Default Add-Ons"
- Section 19.2.4, "How to Display the Customization Manager Add-On"
- Section 19.2.5, "How to Selectively Display Add-Ons"

### 19.2.1 How to Create and Register Add-Ons

You can create and register custom task flows that can be invoked from icons on the Composer toolbar. All registered add-ons have an associated icon 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 icon on the Composer toolbar (Figure 19–3) that users can click to invoke a task flow that contains information about the application.

**Figure 19–3  About Icon on the Composer Toolbar**

This section contains the following subsections:

- Section 19.2.1.1, "Creating an Add-On Task Flow"
- Section 19.2.1.2, "Registering Add-Ons with Composer"
- Section 19.2.1.3, "Registering Add-Ons in adf-config.xml"

#### 19.2.1.1 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 JSF 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 19–1.
Example 19–1  Sample code in the JSFF Fragment

```xml
<?xml version='1.0' encoding='UTF-8'?>
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version='2.1'
   xmlns:jc="http://xmlns.oracle.com/af/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 *Fusion Developer’s Guide for Oracle Application Development Framework*

5. Save the task flow definition file.

Optional, 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`. 

---

19-6  Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper
3. In the Deploy dialog, choose **Deploy to ADF Library JAR file**.

4. Click **Finish**.

   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.

### 19.2.1.2 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 19–2** shows the code of the extension file with a `<panel>` entry.

   **Example 19–2  Composer Extension File**

   ```xml
   <?xml version="1.0" encoding="US-ASCII" ?>
   <pe-extension xmlns="http://xmlns.oracle.com/adf/pageeditor/extension">
      <addon-config>
         <panels>
                  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:

   ```xml
   <event-handlers>
      <event-handler event="close"> oracle.custom.TaskFlowEventHandler
   ```
<event-handler/>
</event-handlers>

For more information, see Section 19.6, "Configuring Event Handlers for Composer UI Events."

5. Save the pe_ext.xml file.

19.2.1.3 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 19–3.

The name attribute must contain the name you used to register the panel in the Composer extension file.

Example 19–3 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.

4. Save the adf-config.xml file.

For information about the Composer-specific configurations you can make in adf-config.xml, see Section B.2.2, “adf-config.xml.”

19.2.2 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 icon is rendered on the Composer toolbar (Figure 19–4).

**Figure 19–4  About Icon on the Composer Toolbar**

Clicking this icon displays the About the Application task flow (Figure 19–5).

**Note:** The add-on panel is automatically sized to fit the content inside the task flow.

**Figure 19–5  About the Application Task Flow**

### 19.2.3 How to Exclude Composer Default Add-Ons

You can choose to show or hide the Page Properties, Reset Page, and Customization Manager icons 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 19–4** shows the code to hide the Page Properties add-on.

**Example 19–4  The adf-config.xml File with the Reset Page Option Excluded**

```xml
<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.
19.2.4 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 16.5.15, "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 icon as shown in Figure 19–6.

*Figure 19–6  Customization Manager Icon*

19.2.5 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 19–5.

*Example 19–5  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 icon on the Composer toolbar. If the returned value is `true`, Composer displays the Reset Page icon.
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 icon, 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 19.3.1, "How to Create and Register Custom Property Panels"
- Section 19.3.2, "What Happens at Runtime"
- Section 19.3.3, "How to Override Default Property Panels"
- Section 19.3.4, "How to Exclude Default Property Panels"
- Section 19.3.5, "How to Selectively Render Property Panels"
- Section 19.3.6, "How to Display Properties and Parameters in a Custom Property Panel"

19.3.1 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 19.3.1.1, "Creating a Custom Property Panel"
Creating Custom Property Panels

19.3.1.1 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 19.2.1.1, "Creating an Add-On Task Flow" and Section 19.2.1.2, "Registering Add-Ons with Composer."

Example 19–6 shows a sample property panel declaration in the pe_ext.xml file.

Example 19–6  Custom Property Panel Declaration in pe_ext.xml

```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 19–7. This ensures that the panel displays automatically in the Composer Component Properties and Page Properties dialog.

Example 19–7  Custom Property Panel Configuration in pe_ext.xml

```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.

19.3.1.2 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 19.2.1.2, "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 19–8 shows a custom property panel that is associated with a CommandButton component by specifying the component's fully qualified class name. For information about Oracle ADF components and their runtime classes, see Fusion Developer's Guide for Oracle Application Development Framework.

**Example 19–8  Code to Register a Property Panel for a Component**

```xml
<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.

**19.3.1.3 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 19–9 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.
Creating Custom Property Panels

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 19.2.1.2, "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 19–9 shows a custom property panel associated with a dashboard task flow.

   **Example 19–9  Code to Register a Property Panel for a Task Flow Instance**

   ```xml
   <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>
   ```
Creating Custom Property Panels

19.3.2 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.

19.3.3 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 19–10 shows the code to override the Display Options panel with a panel defined using a task flow called custom-panel-task-flow.

Example 19–10  Sample Code to Override a Default Panel

```xml
<addon-config>
  <panels>
    <panel name="oracle.adf.pageeditor.pane.generic-property-inspector" title="Sample Panel"
  </panel>
```

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 19.3.6, "How to Display Properties and Parameters in a Custom Property Panel."

Note: 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.
Creating Custom Property Panels

19.3.4 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 set the rendered attribute to false.

Example 19–11 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 19–11 rendered Attribute for a Property Panel

Example 19–11 rendered Attribute for a Property Panel

19.3.5 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}"

19.3.6 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 19.3.1.1, "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 19–7.

Figure 19–7  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:

```xml
<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:

  ```xml
  <!-- 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:

  ```xml
  <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:
Creating Custom Property Panels

Extending Runtime Editing Capabilities Using Composer

- 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 is selected, and it displays a drop-down 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:

```java
/** Set the refresh interval attribute value on the component
 * @param refreshInterval the refresh interval value
 */
public void setRefreshInterval(String refreshInterval)
{
    this._refreshInterval = 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;
}
```

```java
public List<SelectItem> getSelectItems()
{
    return _selectItems;
}
```

```java
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);
}````
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:

```java
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);
    }
}
```

19.4 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 19–8. 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 19–8 Expression Language Editor](image)

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 19.4.1, "How to Extend the Expression Builder"
19.4.1 How to Extend the Expression Builder

You can extend the expression builder by adding custom options to the expression value drop-down 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 19–12 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 19–12 Sample Code to Add a Custom Value in the Expression Dialog

```java
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 19.2.1.2, "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">
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.

### 19.4.2 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 as described in Section 19.11, "Enabling Direct Select in Design or Add Content 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 if an EL can be entered for component properties. The default is true. If an EL is already defined for a property, it cannot be changed.
- `<pe:protect-el>` defines whether EL in form elements is read-only. The default is false. If an EL is already defined and the component is being edited in a different level, you cannot change the EL.

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. If an EL is already defined for a property, it cannot be changed.
- `protectEL` (boolean) defines whether or not EL in form elements is read-only. The default is false. If an EL is already defined and the component is being edited in a different level, you cannot change the EL.

The settings defined in the .jspx page override any settings defined in adf-config.xml.

### 19.4.3 What Happens at Runtime

The selector in the runtime expression builder displays the custom value along with the default values.

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.)
19.5 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. 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.

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 19.5.1, "How to Configure an LOV"
- Section 19.5.2, "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 19–9.

![Weather Task Flow](image.png)

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 19–10.

**Figure 19–10  Static LOV for the tempUnits Parameter**

- A picker with two fields that enable users to specify a city name or ZIP code (Figure 19–11) to view corresponding details in the weather task flow.

**Figure 19–11  City Picker for the zipCode Parameter**

- Alternatively, a dynamic LOV for the zipCode parameter to display values (Figure 19–12) based on an EL value that is evaluated at runtime.
The weather task flow includes a task flow definition file, `weather-task-flow-definition.xml`, with the code shown in Example 19–13.

### Example 19–13  The `weather-task-flow-definition.xml` File

```xml
<?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 options.
19.5.1 How to Configure an LOV

This section describes how to configure different types of LOVs for task flow parameters. It contains the following subsections:

- Section 19.5.1.1, “Configuring a Static LOV”
- Section 19.5.1.2, “Configuring a Dynamic LOV”
- Section 19.5.1.3, “Configuring a Picker”
- Section 19.5.1.4, “Configuring an LOV from a Global List”

19.5.1.1 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:

```xml
<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-definition">
    <!-- 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.

### 19.5.1.2 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:

   ```java
   package portal.weather;
   ...
   
   public List<SelectItem> getCitiesList() {
       return citiesList;
   }

   private static final String F = "°F";
   private static final String C = "°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]));
       }
   }
   
   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\adfsmsrc\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-definition">
    <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.

19.5.1.3 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 contains the following subsections:

■ Section 19.5.1.3.1, "Creating the Picker Task Flow"
■ Section 19.5.1.3.2, "Registering the Picker for a Task Flow Parameter"

19.5.1.3.1 Creating the Picker Task Flow

Create a picker task flow with a JSF 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:** For information about creating task flows, see the "Getting Started with ADF Task Flows" chapter in *Fusion Developer's Guide for Oracle Application Development Framework*.

4. Design the fragment with components of your choice. **Example 19–14** 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 19–14  Sample code in the Picker JSFF Fragment**

```xml
<?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>
            </af:panelGroupLayout>
        </f:facet>
        <f:facet name="bottom">
            <!-- 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"/>
        </f:facet>
    </af:panelStretchLayout>
</jsp:root>
```

See Also: For information about creating task flows, see the "Getting Started with ADF Task Flows" chapter in *Fusion Developer’s Guide for Oracle Application Development Framework*. 

---

**Configuring Custom LOVs or Pickers for Task Flow Parameters**

**Extending Runtime Editing Capabilities Using Composer**  19-29
Configuring Custom LOVs or Pickers for Task Flow Parameters

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:

```java
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:

```xml
<managed-bean id="pickerbean">
    <managed-bean-name id="pickerbeannname">weatherPicker</managed-bean-name>
    <managed-bean-class id="pickerbeanclass">portal.weather.picker.WeatherPicker</managed-bean-class>
</managed-bean>
```

7. Define two input parameters, `inParamVal` and `outParam`, on the picker task flow, as shown in the following example:

```xml
<!-- Parameter used to store the value that must be passed to the main task flow -->
<input-parameter-definition id="pickeripl"
    <name id="pickeriplname">outParam</name>
    <value>#{pageFlowScope.outParam}</value>
Extending Runtime Editing Capabilities Using Composer

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:

```xml
<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>
```

The task flow definition will now have code similar to the following example:

```xml
<?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>
  </task-flow-definition>
</adfc-config>
```
Configuring Custom LOVs or Pickers for Task Flow Parameters

19-32

9. Save the task flow definition file.

19.5.1.3.2 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
<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>
</task-flow-definition>
</adfc-config>
```
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.

### 19.5.1.4 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:

```xml
<?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-definition">
      <!-- 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-flow-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.

You can configure pickers for any number of parameters within the `<lov-config>` section.

You can configure pickers for any number of parameters within the `<lov-config>` section.

You can configure pickers for any number of parameters within the `<lov-config>` section.

You can configure pickers for any number of parameters within the `<lov-config>` section.
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:

   ```xml
   <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.

19.5.2 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 will list values from the static LOV that you defined in the `pe_ext.xml` file.

If you configured a picker for the Zipcode field, the field shows a magnifier icon to its right. This icon invokes a picker that allows users to select a city by providing a name or ZIP code.

19.6 Configuring Event Handlers for Composer UI Events

Event handlers are Java classes registered with Composer and called when a user 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 event. In addition to the event handlers that Composer provides by default, you can configure additional handlers for application-specific events, for example, saving changes in a custom property panel.

This section describes how to create and register event handlers for UI events in Composer. It contains the following subsections:

- Section 19.6.1, "How to Create and Register Handlers for Composer UI Events"
- Section 19.6.2, "What Happens When You Create and Register Event Handlers"
- Section 19.6.3, "Additional Composer Event Handler Configurations"

19.6.1 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 19.6.1.1, "UI Events that Support Event Handler Registration"

Section 19.6.1.2, "Creating a Save Event Handler: Example"

Section 19.6.1.3, "Registering an Event Handler with Composer"

19.6.1.1 UI Events that Support Event Handler Registration

Table 19–1 lists the UI events for which Composer currently supports registering of handlers.

<table>
<thead>
<tr>
<th>Event</th>
<th>Cause</th>
<th>Event Type</th>
<th>Listener Interface (oracle.adf.view.page.editor.event)</th>
<th>Method</th>
<th>Event Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>Save</td>
<td>Invoked when a user clicks the <strong>Save</strong> or <strong>Save and Label</strong> button on the Composer toolbar, or the <strong>Apply</strong> (save) or <strong>OK</strong> (save and close) button in the Component Properties dialog or Page Properties dialog.</td>
<td>save</td>
<td>SaveListener</td>
<td>processSave</td>
<td>SaveEvent</td>
</tr>
<tr>
<td>Close</td>
<td>Invoked when a user clicks the <strong>Close</strong> button on the Composer toolbar.</td>
<td>close</td>
<td>CloseListener</td>
<td>processClose</td>
<td>CloseEvent</td>
</tr>
<tr>
<td>Deletion</td>
<td>Invoked when a user deletes the component.</td>
<td>delete</td>
<td>DeletionListener</td>
<td>processDeletion</td>
<td>DeletionEvent</td>
</tr>
</tbody>
</table>

Get the deleted component using `getComponent`.
19.6.1.2 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 19–1.

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.

```java
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 SaveHandler() {
        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.
19.6.1.3 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 19.2.1.2, "Registering Add-Ons with Composer."

To register an event handler:

1. In the application's `pe_ext.xml` file, add the following entries:

   ```xml
   <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.

19.6.2 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 `processEventName` 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.

19.6.3 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 are executed and configure a listener to terminate a process or notify Composer that the event has been handled.

19.6.3.1 Specifying 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:

   ```xml
   <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.**

### 19.6.3.2 Terminating Event Processing

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 19–15**.

**Example 19–15  DeletionListener Implementation**

```java
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);
            }
        }
    }
    ...
} 
```
19.7 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 19.7.1, "How to Create and Register Java Drop Handlers"
- Section 19.7.2, "What Happens at Runtime"

19.7.1 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 contains the following subsections:

- Section 19.7.1.1, "Creating a Drop Handler"
- Section 19.7.1.2, "Registering a Drop Handler with Composer"
- Section 19.7.1.3, "Adding an XML Component to the Resource Catalog"

19.7.1.1 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:

```java
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:
Configuring Drop Handlers in the Resource Catalog

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

19-42 Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper


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.

19.7.1.2 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 19.2.1.2, "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.

### 19.7.1.3 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 Section 14.4, "Selecting the Default Resource Catalog."

2. Add the following code within the `<contents>` section of the file:

   ```xml
   <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_qualified.png"/>
     </attributes>
     <parameters>
       <parameter id="xml">
       </parameter>
     </parameters>
   </component>
   

   At runtime, the resource catalog shows the XML component. For more information, see Section 14.3.5, "How to Add a Custom Component to a Resource Catalog."

### 19.7.2 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 19–13 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.
19.8 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.

You can define additional filters to hide more properties in the Component Properties dialog. This section describes how. It contains the following subsections:

- **Section 19.8.1, "How to Define Property Filters"
- **Section 19.8.2, "What Happens at Runtime"
- **Section 19.8.3, "How to Remove Property Filters"

19.8.1 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 19.2.1.2, "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:

```xml
<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.

### 19.8.2 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.

### 19.8.3 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 in 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.

19.9 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 19–2 describes the parameters supported by the Customization Manager task flow.

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>defaultPage</td>
<td>Page URI</td>
<td>Used to specify the default page to be managed. If this value is null, the current page will be used.</td>
</tr>
<tr>
<td>dynamicNodeMap</td>
<td>Map&lt;String,List&lt;CustomizationManagerNode&gt;&gt;</td>
<td>Used to return a list of pages, XML documents, or task flows on a given page or fragment, using the following format: Page1.jspx -&gt; 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.</td>
</tr>
<tr>
<td>showBase</td>
<td>true/false</td>
<td>Used to show or hide the column for the base document.</td>
</tr>
<tr>
<td>showCurrentLayers</td>
<td>true/false</td>
<td>Used to show or hide the column showing details about the currently used layer.</td>
</tr>
<tr>
<td>Parameter Name</td>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>showAllLayers</td>
<td>true/false</td>
<td>Used to show or hide the column showing details about layers other than the current one. Default: true</td>
</tr>
<tr>
<td>showLayersForCurrent</td>
<td>Map&lt;String,List&lt;String&gt;&gt;</td>
<td>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 19–16 for more information.</td>
</tr>
<tr>
<td>showLayersForAll</td>
<td>Map&lt;String,List&lt;String&gt;&gt;</td>
<td>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 19–16 for more information.</td>
</tr>
<tr>
<td>labelPrefixes</td>
<td>List&lt;String&gt;</td>
<td>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.</td>
</tr>
<tr>
<td>maxDepth</td>
<td>String</td>
<td>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 a value higher than 10.</td>
</tr>
<tr>
<td>showDownload</td>
<td>true/false</td>
<td>Used to show or hide the Download link. Default: true</td>
</tr>
<tr>
<td>showUpload</td>
<td>true/false</td>
<td>Used to show or hide the Upload link. Default: true</td>
</tr>
<tr>
<td>showDelete</td>
<td>true/false</td>
<td>Used to show or hide the Delete link. Default: true</td>
</tr>
<tr>
<td>showPromote</td>
<td>true/false</td>
<td>Used to show or hide the Promote link. Default: true</td>
</tr>
</tbody>
</table>
Enabling Parameter Support on the Customization Manager Task Flow

This section describes the steps to enable parameter editing on the Customization Manager task flow. It contains the following subsections:

- **Section 19.9.1, "How to Set the Customization Manager Task Flow Parameters"**
- **Section 19.9.2, "What Happens When You Enable Parameter Support"**

### 19.9.1 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:

   ```xml
   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 19–16.

**Example 19–16  Customization Manager Add-On Referenced in adf-config.xml**

```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:

```java
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;
}
```

**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`.

Extending Runtime Editing Capabilities Using Composer 19-49
4. Save the adf-config.xml file.

19.9.2 What Happens When You Enable Parameter Support

Customization Manager displays options to users based on the new parameter settings.

19.10 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.

Notes: The Save button is displayed on the toolbar only if the application is configured to use a sandbox. For more information, see Section 21.2, "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.

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 19–3 describes the toolbar areas.

<table>
<thead>
<tr>
<th>Section or Area</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>Area containing the message with the page name.</td>
</tr>
<tr>
<td>statusindicator</td>
<td>Area containing the status indicator.</td>
</tr>
<tr>
<td>menu</td>
<td>Area containing the View menu that allows users to switch between Design view and Structure view.</td>
</tr>
<tr>
<td>addonpanels</td>
<td>Area containing Composer add-ons. The add-ons are all displayed in a sequence within this area. For more information, see Section 19.2, &quot;Creating Composer Add-Ons.&quot;</td>
</tr>
<tr>
<td>help</td>
<td>Area containing the Help icon.</td>
</tr>
<tr>
<td>button</td>
<td>Area containing the Save and Close buttons.</td>
</tr>
<tr>
<td>stretch</td>
<td>Wide space between two sections.</td>
</tr>
<tr>
<td>newline</td>
<td>A line break between sections. All sections after a line break are pushed to a new line on the toolbar.</td>
</tr>
</tbody>
</table>

Using toolbar sections, you can customize the Composer toolbar in the following ways:

- Rearrange and hide built-in elements
Customizing the Composer Toolbar

- 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 19.10.1, "How to Rearrange or Hide Toolbar Elements"
- Section 19.10.2, "What Happens at Runtime"
- Section 19.10.3, "How to Add New Sections to the Toolbar"
- Section 19.10.4, "How to Override a Toolbar Section to Display Custom Content"

19.10.1 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 addonpanels stretch button help, 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 under Appearance.

   The section names in Table 19–3 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:

   ```xml
   <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.

19.10.2 What Happens at Runtime

At runtime, the toolbar displays the sections you specified in the order you specified them (Figure 19–14).
19.10.3 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 icon 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 icon, 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 will display a Report a Bug icon. Clicking this icon will display the File a Bug dialog to enable users to report a bug.
19.10.4 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:

```xml
<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>
```

19.11 Enabling Direct Select in Design or Add Content View

Direct select allows you to extend specific customization capabilities from Composer Structure view to Design or Add Content 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 Structure view, but business users do not have the knowledge or experience to use Structure 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 about direct select functionality, see Section 19.11.2, "What Happens at Runtime."

19.11.1 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 the Select view. 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 tabs to switch between views.
   - design: Makes the Design view the only view available.
   - add-content: Makes the legacy Add Content view the only available view.
   - select: Makes the Select view the only available view.
Enabling Direct Select in Design or Add Content View

- source: Makes the Structure view the only available view.
- preview: Makes Preview the only available view.

For example, if you want only Select and Add Content views, you can add the following tag:

```xml
<pe:enable-design-views> select add content</pe:enable-design-views>
```

You can also use EL to determine the appropriate view, as shown in the example below:

```xml
<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.

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 in which the direct select is enabled.
- `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 19.3, "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.

```xml
<selection-config>
  <global-filter>
    <selection view="design" enabled="false"/>
  </global-filter>

  <selection-taglib-filter
```
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 19.8.1, "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 `<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.)

4. To disallow the use of Expression Language in form fields and protect existing EL from overwrite, see Section 19.4.2, "How to Protect Expression Language."

For more information about 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"
19.11.2 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 (Figure 19–15).

![Select View](image1.png)

To edit the properties for a component, the user can select an enabled element and click Edit Component or Edit Parent Component, when applicable (Figure 19–16).

![Direct Select Enabled for an Element](image2.png)

The Component Properties dialog 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 dialog. If EL is not allowed or is protected from overwrite, the Expression Builder and Override options are not available, as shown in Figure 19–17. This image also includes an example of a field that is protected from edit (Show Component).

![Component Properties Dialog](image3.png)
Multiple operations are shown in tabs in the dialog. 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 19.3, "Creating Custom Property Panels."

19.12 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 Chapter 18, "Enabling Runtime Editing of Pages Using 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 19.2, "Creating Composer Add-Ons."

Problem
You have registered a custom property panel. However, it does not appear in the Component Properties dialog when you 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 19.3, "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 19.8.1, "How to Define Property Filters" and Section 22.4, "Applying Attribute-Level Security."
This chapter describes how to perform certain advanced Composer configurations to enhance the end user experience.

This chapter contains the following topics:

- Section 20.1, "Enabling Custom Actions on a Show Detail Frame Component by Using Facets"
- Section 20.2, "Enabling Custom Actions on a Show Detail Frame Component by Using Facets: Example"
- Section 20.3, "Enabling Custom Actions on a Task Flow"
- Section 20.4, "Enabling Custom Actions that Display on Task Flows in the Component Navigator"
- Section 20.5, "Configuring a Keyboard Shortcut to Access Composer"
- Section 20.6, "Creating Event-Enabled Task Flows"
- Section 20.7, "Configuring an Application Page to Display in Structure View by Default"
- Section 20.8, "Disabling Structure View for the Application"
- Section 20.9, "Disabling Task Flow Zoom Capability"
- Section 20.10, "Applying Styles to Components"
- Section 20.11, "Configuring the Persistence Change Manager"
- Section 20.12, "Configuring Runtime Resource String Editing"
- Section 20.13, "Troubleshooting Problems with Advanced Composer Configurations"

20.1 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, and click the arrow displayed to the right of this option.

2. 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.

3. 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 20.2, "Enabling Custom Actions on a Show Detail Frame Component by Using Facets: Example."

20.2 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 20.2.1, "How to Create an ADF Task Flow"
- Section 20.2.2, "How to Include an Additional Actions Facet"
- Section 20.2.3, "How to Create a Redirection Page"
- Section 20.2.4, "How to Create Navigation Rules Between Pages"
- Section 20.2.5, "What Happens at Runtime"

20.2.1 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 shown in the following example:

   ```java
   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.

20.2.2 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 20–1.

8. Save the page.

20.2.3 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.

### 20.2.4 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** icon in the Managed Beans 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:

```xml
<managed-bean>
  <managed-bean-name>backingBean</managed-bean-name>
  <managed-bean-class>viewBackingBean</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.

20.2.5 What Happens at Runtime

In JDeveloper, run Page1.jspx. The Actions menu on the showDetailFrame1 component displays the Customize action, as shown in Figure 20–2.

Clicking Customize takes you to Page2.jspx (Figure 20–3), where you can update the values for Label1 and Label2.

Clicking OK takes you back to Page1.jspx, which reflects the recent text changes.

20.3 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 contains the following subsections:

- Section 20.3.1, "How to Enable Custom Actions Directly on a Task Flow"
Enabling Custom Actions on a Task Flow

20.3.1 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 20–1 shows the code of the `adf-settings.xml` file with a `<customAction>` entry.

   **Example 20–1 The custComps-config Section in an adf-settings.xml File**

   ```xml
   <customActions>
   </customActions>
   ```

   ```xml
   <customAction action="next" locations="chrome"
   rendered="true"
   icon="/adf/webcenter/editheader_ena.png"
   text="Next"
   taskFlowId="/WEB-INF/task-flow-definition.xml#task-flow-definition"
   shortDesc="next"/>
   ```

   ```xml
   <customAction action="prev" location="chrome"
   rendered="true"
   icon="/adf/webcenter/editheader_ena.png"
   text="Previous"
   ```
Enabling Custom Actions on a Task Flow

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.

20.3.2 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 contains the following subsections:

- Section 20.3.2.1, "Defining Custom Actions at the Instance Level"
- Section 20.3.2.2, "Defining Custom Actions at the Global Level"
- Section 20.3.2.3, "Configuring Custom Actions that Display Task Flow Views in a Separate Browser Window"

20.3.2.1 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 20–2.

**Example 20–2  Custom Action Code**

```xml
<cust:showDetailFrame text="showDetailFrame 1" id="sdf1">
  <af:region values="#{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 *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.

**20.3.2.2 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 20–3.

   **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 20–3 Custom Actions Definitions in the adf-config.xml File**

   ```xml
   <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 22.5, "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 instead of the cust:adf-config-child section shown in Example 20–3.

- 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 20.3.4, "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:

- 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 7, "Deploying and Testing Your Portal Framework Application."

20.3.2.3 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`
Enabling Custom Actions on a Task Flow

attribute of the custom action corresponding to that view. The following example shows an action attribute definition:

```xml
<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:

```xml
<af:commandButton text='dialog:simpleEditPoup'
    id='SDFCustomActionCmd_simpleEditPoup'
    action='dialog:simpleEditPoup' useWindow='true'
    windowEmbedStyle='inlineDocument' windowHeight='200'
    windowWidth='300'
    windowModalityType='applicationModal'
    returnListener='#{pageFlowScope.recentPagesBean.refreshMainView}"
    visible='false'/>
```

Example 20–4 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 20–4 Global Custom Action Defined in the adf-config.xml File**

```xml
<customizableComponentsSecurity
    xmlns="http://xmlns.oracle.com/adf/faces/customizable/config">
    <enableSecurity value='true'/>
    <customActions>
        <customAction action='dialog:simpleEditPoup'
            text='Simple Edit'
            shortDesc='Simple Edit'
```
Enabling Custom Actions on a Task Flow

Performing Advanced Composer Configurations

Example 20–5 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 20–5  Instance-Level Custom Action Defined in the JSPX Page

```xml
<cust:showDetailFrame id="adf_for_RecentPagesTF1"
  text="Recent Pages" stretchContent="false"
  showResizer="never">
  <af:region id="RecentPagesTF1"
    value="#{bindings.regionBinding1.regionModel}"/>
  <cust:customAction action="dialog:simpleEditPopup"
    text="My Simple Edit"
    shortDesc="Simple Edit"
    location="menu"
    rendered="#{!changeModeBean.inEditMode}"
    icon="/adf/pe/images/editproperties_ena.png"
    actionComponent="SDFCustomActionCmd_simpleEditPopup"/>
</cust:showDetailFrame>
```

**Note:** A custom action that is launched in a popup dialog is sent as a new request to the 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.

20.3.3 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.

20.3.4 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 18.1.1, "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.


   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 20–4).

Figure 20–4  Custom Actions on a Show Detail Frame Enclosing a Task Flow

To see this in action, look at the Composer Custom Actions sample application, ComposerCustomActions.jws, on the Oracle WebCenter Suite 11g Demonstrations and Samples page on Oracle Technology Network (OTN).
20.4 Enabling Custom Actions that Display on Task Flows in the Component Navigator

The component navigator in Composer Structure view provides an option to zoom into a task flow and display the components on its page or fragment, as shown in Figure 20–5.

Figure 20–5 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 20–6.

Figure 20–6 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 20.4.1, "How to Configure Custom Actions in the Component Navigator"
- Section 20.4.2, "What Happens at Runtime"

20.4.1 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
Enabling Custom Actions that Display on Task Flows in the Component Navigator

application page containing the task flow. This section describes the steps to do so in detail. It contains the following subsections:

- Section 20.4.1.1, "Defining the Logic for the Custom Action"
- Section 20.4.1.2, "Creating a JSPX Page Containing the Custom Action"
- Section 20.4.1.3, "Calling the JSPX Page from the Application Page Containing the Task Flow"

20.4.1.1 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.
4. The BackingBean.java file displays in Source view.
5. Import the required libraries as follows:

   ```java
   import javax.faces.application.FacesMessage;
   import javax.faces.context.FacesContext;
   import javax.faces.event.ActionEvent;
   import oracle.adf.view.page.editor.sourceview.ComponentInfo;
   ```

6. Add the following code:

   ```java
   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.

20.4.1.2 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
   <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:

   ```xml
   <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:

   ```xml
   <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
   <?xml version='1.0' encoding='UTF-8'?>
   ```
Enabling Custom Actions that Display on Task Flows in the Component Navigator

20.4.1.3 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
<af:document id="d1">
  <af:form id="f1">
    <af:panelStretchLayout topHeight="50px" id="ps1">
      <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}">
              <af:region value="#{bindings.taskflowdefinition1.regionModel1}" id="r1"/>
            </af:region>
          </cust:panelCustomizable>
        </pe:pageCustomizable>
      </f:facet>
    </af:panelStretchLayout>
  </af:form>
</af:document>
```

5. Save the JSPX page.
The task flow, taskflowdefinition1, contains the following view.jsff fragment:

```xml
<?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:

```xml
<pe:pageCustomizable id="pageCustomizable1" sourceViewNodeAction="/customLink.jspx">
  
</pe:pageCustomizable>
```

The sourceViewNodeAction attribute can take the name of a JSPX file or an EL value that evaluates to a JSPX file name.

### 20.4.2 What Happens at Runtime

When you run MyPage.jspx to the browser and open the page in Composer Structure view, the component navigator displays a Test link and an Edit Task Flow link next to each task flow instance on the page.

### 20.5 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 contains the following subsections:

- Section 20.5.1, "How to Enable Linkless Entry Into Edit Mode"
- Section 20.5.2, "How to Configure an Event Handler for the Shortcut Key"
- Section 20.5.3, "What Happens at Runtime"

#### 20.5.1 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 20–6. The example shows the `<pe:key-sequence>` property with the shortcut value `ctrl+E`.

Example 20–6  Enabling Linkless Edit Mode Entry

```xml
<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.

20.5.2 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:

```xml
<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:

```java
public void handleModeSwitch(ModeChangeEvent editMode)
{
    // Your implementation to handle mode change goes here
}
```

20.5.3 What Happens at Runtime

Users can access the application page and enter Edit mode by using the configured shortcut.

20.6 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 the "Creating Contextual Events" section in Fusion Developer’s Guide for Oracle Application Development Framework.
20.7 Configuring an Application Page to Display in Structure View by Default

When a user invokes Composer by clicking the Edit button or link on a page, the page opens in Add Content or Design view by default. If your business so requires, you can configure your application pages to display in Composer Structure view by default.

20.7.1 How to Configure an Application Page to Display in Structure View by Default

The Edit Mode view attribute on the Page Customizable component enables you to specify the default page view in Edit mode.

To open a page in Structure 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 drop-down list and select source.
4. Save the page.

Setting the Structure View Position and Size
By default, the component navigator in Structure view is displayed on the right side of the page, and its default width 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 Component."

20.7.2 What Happens at Runtime

When a user clicks the Edit link or button on the page, Composer opens the page in Structure view.

20.8 Disabling Structure View for the Application

By default, Structure view is enabled in Composer-enabled pages. You can disable Structure view if you want to prevent users from being able to edit any page components other than task flows, portlets, and layout components. This section describes how. It contains the following subsections:

- Section 20.8.1, "How to Disable Structure View"
- Section 20.8.2, "What Happens at Runtime"

Note: For information about the editing capabilities in Structure view, see Section 16.6, "Editing Capabilities in Structure View in Page Edit Mode."
20.8.1 How to Disable Structure View

You can disable Structure view 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 Structure 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 20–7.

Example 20–7  Disabling Structure View in `adf-config.xml`

```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.

20.8.2 What Happens at Runtime

When a user switches to page Edit mode, the page is rendered in Add Content or Design view and Structure view is not displayed to the user, as shown in Figure 20–7.

Figure 20–7  Add Content View of a Page without the Structure Tab

---

20.9 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.
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 20–8.

   **Example 20–8 Disabling Task Flow Zoom Capability in adf-config.xml**

   ```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.

### 20.10 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 Portal Framework application. For more information about style selectors and skins, see the "Customizing the Appearance Using Styles and Skins" chapter in *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.
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 20–8 Defining Styles for contentStyle and inlineStyle in the Property Inspector

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.

20.11 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. This section describes the default change manager configuration in new Portal Framework applications containing Composer-enabled pages.

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:**

  ```xml
  <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>
  ```

- **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:**

  ```xml
  <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:

```xml
<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-config>
    <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 22.4.1, "How to Define Change Persistence at the Component Level."

If you want to persist changes to operations as well as attributes, include the `persist-operations` attribute and set the value to `all`. When the `persist-operations` attribute is included in a tag, the tag persists changes such as adding, moving, reordering, and removing children, and adding and removing facets.

For a list of ADF Faces components and attributes that are implicitly persisted, see the "Allowing User Customizations at Runtime" chapter in 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 user's 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.

### 20.12 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.
This section describes the steps to enable resource string editing. It contains the following sections:

- **Section 20.12.1, “Overview of the Resource String Editor”**
- **Section 20.12.2, “How to Enable the Resource String Editor in Your Application”**
- **Section 20.12.3, “How to Configure the Override Bundle”**
- **Section 20.12.4, “What Happens at Runtime”**

### 20.12.1 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 16.5.5, “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.

20.12.2 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:

```xml
<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.

20.12.3 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.resourceComposerOverrideBundle.xlf`, if it is not already being used. This override bundle is available in the `pageeditor.jar` file, and 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. From the **Application** menu, select **Application Properties**.
4. If the Bundle section does not display `oracle.adf.view.page.editor.resourceComposerOverrideBundle.xlf`, click the **Add** button to add the bundle.
5. In the File Name field in the Select Resource bundle dialog, enter `oracle.adf.view.page.editor.resourceComposerOverrideBundle.xlf`, and click **Open**.
6. In the Application Properties dialog, select the **Overridden** check box (Figure 20–9).

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.

### 20.12.4 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 16.5.5, "Edit Resource Strings."

### 20.13 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 nor 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 20.3.2, "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 Oracle 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:

```xml
<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.
This chapter describes how to create customization layers for saving changes made to a 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.

This chapter contains the following topics:

- Section 21.1, "Introduction to MDS"
- Section 21.2, "Using Composer Sandbox"
- Section 21.3, "Adding Customization Layers to View and Edit Modes: Example"
- Section 21.4, "Troubleshooting MDS-Related Problems with Composer"

### 21.1 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 application-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 21.3, "Adding Customization Layers to View and Edit Modes: Example."

**Composer Sandbox**

Typically, in a Portal 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 21.2, "Using Composer Sandbox."

## 21.2 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 page Edit mode.

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 21.2.1, "How to Enable Composer Sandbox Creation"
- Section 21.2.2, "What Happens at Runtime"
- Section 21.2.3, "How to Disable Sandbox for an Application"
- Section 21.2.4, "How to Destroy Stale Sandboxes"
21.2.1 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 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 21.2.1.1, "Updating Your Application's adf-config.xml File"
- Section 21.2.1.2, "Updating Your Application's web.xml File"
- Section 21.2.1.3, "Selecting a Database Store for Your Application"
- Section 21.2.1.4, "Enabling a Full Page Refresh at Runtime"

21.2.1.1 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 21–1.

**Example 21–1  Namespace Definitions in the adf-config.xml File**

```xml
<!-- 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 21–2.

**Example 21–2  Configuring Namespaces for Sandbox Creation**

```xml
<pe:sandbox-namespaces>
    <pe:namespace path="/pages"/>
    <pe:namespace path="/pageDefs"/>
</pe:sandbox-namespaces>
```
4. Save the adf-config.xml file.

21.2.1.2 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.

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:

2. Add WebCenterComposerFilter before the adfBindings filter:
   
   ```xml
   <filter>
     <filter-name>composerFilter</filter-name>
     <filter-class>oracle.adf.view.page.editor.webapp.WebCenterComposerFilter</filter-class>
   </filter>
   ```

3. Add corresponding <filter-mapping> elements before the filter mapping for AdfBindingFilter as shown in Example 21–3.

   **Example 21–3  Filter Mappings for Composer-Specific Filter**

   ```xml
   <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.

21.2.1.3 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 7.3.4.1, "Deploying to a Managed Server Using Local Data Sources."

21.2.1.4 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 21.3.7, "How to Redirect the Servlet to Enable Switch Between MDS Customization Layers."

21.2.2 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 21–1.

Figure 21–1  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 21–2**.

![Figure 21–2 Page Concurrency Message](image)

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 21–3**.

![Figure 21–3 Task Flow Concurrency Message](image)

**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.

---

**21.2.3 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`.

### 21.2.4 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`:


2. Define a new listener, `WebCenterComposerSessionListener`, as shown in the following example:

   ```xml
   <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>
   ```


For information about the Composer-specific configurations you can make in `web.xml`, see Section B.2.4, "web.xml."

### 21.3 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 21.3.3, "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 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 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 21.3.1, "How to Add Composer to a JSF Page"
- Section 21.3.2, "How to Create a Custom SiteCC Tip Layer"
- Section 21.3.3, "How to Create a Custom UserCC Tip Layer"
- Section 21.3.4, "How to Implement the ComposerSessionOptionsFactory Class"
- Section 21.3.5, "How to Register the Implementation with Composer"
- Section 21.3.6, "How to Configure WebCenterComposerFilter"
- Section 21.3.7, "How to Redirect the Servlet to Enable Switch Between MDS Customization Layers"
- Section 21.3.8, "What Happens at Runtime"

### 21.3.1 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:

```xml
<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:

```xml
  <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.

### 21.3.2 How to Create a Custom SiteCC Tip Layer

This section describes how to create a custom SiteCC tip layer, site, in which all application-level customizations performed in Edit mode are stored. In this sample application, application-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. **In the Source view, press Enter after** `public class SiteCC extends CustomizationClass, and add the following string:**

```java
{
    private static final String DEFAULT_LAYER_NAME = "site";
    private String mLayerName = DEFAULT_LAYER_NAME;
    private String mLayerValue = "webcenter";
}
```

**The following libraries are imported:**

```java
import oracle.mds.core.MetadataObject;
import oracle.mds.core.RestrictedSession;
import oracle.mds.cust.CacheHint;
```

7. **Update the code as shown in bold here:**

```java
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 }
    }
}
```

8. **Save the SiteCC.java file.**

### 21.3.3 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.

Ensure the Implement Abstract Methods check box is selected.

5. Click OK.

The UserCC.java file displays in the Source view.

6. In the Source view, press Enter after public class UserCC extends CustomizationClass, and add the following string:

```java
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:

```java
import oracle.mds.core.MetadataObject;
import oracle.mds.core.RestrictedSession;
import oracle.mds.cust.CacheHint;
```

7. Update the code as shown in bold here:

```java
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};
    }
}
```

8. Save the UserCC.java file.

### 21.3.4 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:

   ```java
   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:

   ```java
   public class AppsSessionOptionsFactoryImpl
       implements ComposerSessionOptionsFactory
   {

       public SessionOptions createSessionOptions(SessionOptions defaultSessionOptions, String mode)
       {

           CustomizationClass[] custLayer;
           CustConfig custConfig = null;

           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() };

7. Save the AppsSessionOptionsFactoryImpl.java file.

21.3.5 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>

21.3.6 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 21.2.1, "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:
Adding Customization Layers to View and Edit Modes: Example

....

<filter>
    <filter-name>composerFilter</filter-name>
    <filter-class>oracle.adf.view.page.editor.webapp.WebCenterComposerFilter</filter-class>
</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."

21.3.7 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 21.3.7.1, "How to Create the AppNavigationUtils Class (start here)"
- Section 21.3.7.2, "How to Create AppCloseHandler"
- Section 21.3.7.3, "How to Register the AppCloseHandler"
Section 21.3.7.4, "How to Create AppModeBean"

21.3.7.1 How to Create the AppNavigationUtils Class (start here)

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:

   ```java
   import javax.faces.context.FacesContext;
   import javax.servlet.http.HttpServletRequest;
   ```

5. Add the following code:

   ```java
   public class AppNavigationUtils
   {
   
   public static void redirectToSamePage()
   {
   HttpServletRequest request = (HttpServletRequest)FacesContext.getCurrentInstance().getExternalContext().getRequest();
   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();
   }
   
   }
   }
   ```


21.3.7.2 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();
    }
}


21.3.7.3 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 19.6, "Configuring Event Handlers for Composer UI Events."

21.3.7.4 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();
       }
   }
Troubleshooting MDS-Related Problems with Composer

Performing Composer-Specific MDS Configurations

AppNavigationUtils.redirectToSamePage();
}
}

6. Save the file.

21.3.8 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 21–4.

Figure 21–4 Customized Page

Go to /mds/mdssys/cust/site/webcenter in your application directory, and open the pagename.jspx.xml file. This is where the application-level 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.

21.4 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 21.2, "Using Composer Sandbox."
Modifying Default Security Behavior of Composer Components

This chapter describes how to override the default security behavior of Composer components.

This chapter contains the following topics:

- Section 22.1, "Applying Component-Level Restrictions by Defining Customization Policies"
- Section 22.2, "Applying a Component Instance-Level Customization Restriction By Using Security Roles: Example"
- Section 22.3, "Applying Tag-Level Security Using the customizationAllowed Attribute"
- Section 22.4, "Applying Attribute-Level Security"
- Section 22.5, "Applying Action-Level Restrictions on Panel Customizable and Show Detail Component Actions"
- Section 22.6, "Implementing Task Flow Security"
- Section 22.7, "Overriding Composer’s Default Security Policies"
- Section 22.8, "Troubleshooting Problems with Composer Components Security"

22.1 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.

This section contains the following subsections:

- Section 22.1.1, "How to Define Type-Level Customization Policies"
- Section 22.1.2, "How to Define Instance-Level Customization Policies"
Applying Component-Level Restrictions by Defining Customization Policies

22.1.1 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
<?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 22.2.3, "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:

```xml
<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
<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.

### 22.1.2 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 22.1.1, "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:

\[ \text{tag_id}(@\text{xmlns(tag_prefix=tag_namespace)}/@\text{tag_prefix:attribute_name}) \]

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
<?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>
    </rdf:Description>
</rdf:RDF>
```

At runtime, these policies are read and the artifacts are enabled for customization accordingly.

## 22.2 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 22.2.1, "How to Configure ADF Security"
Section 22.2.2, "How to Define Roles and Grant Privileges in the jazn-data.xml File"

Section 22.2.3, "How to Customize the SessionOptions Object to Include Customization Policy"

Section 22.2.4, "How to Register the Implementation with Composer"

Section 22.2.5, "How to Configure WebCenterComposerFilter"

Section 22.2.6, "How to Apply an Instance-Level Customization Restriction"

Section 22.2.7, "What Happens at Runtime"

22.2.1 How to Configure ADF Security

Before applying an instance-level customization restriction, you must secure your application using ADF security. For information, see Section 74.3, "Configuring ADF Security."

22.2.2 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 Application Roles tab.
3. In the Application Roles page, click the Add icon in the Roles section, and select Add New Role, shown in Figure 22-1.
4. In the Name field, specify the role name, admin.

5. Optionally, in the Display Name and Description fields, specify a display name and description for the new role.

6. Repeat steps 3 through 6 to add another role, customer.

7. Click the Users tab.

8. In the Users page, create new users by clicking the New User icon and specifying credentials. Assign roles to these users by clicking the Assign Roles icon in the Assigned Roles section and selecting Assign Application Role, as shown in Figure 22–2.

9. Click the Resource Grants tab on the left on the page.

10. From Resource Type, select Web Page.
11. From the list of page definitions, select the page on which you want to grant privileges.

12. In the Granted To column, click the Add Grantee icon, and select **Add Application Role** from the list, as shown in Figure 22–3.

**Figure 22–3 Resources in the jazn-data.xml File**

13. In the Select Roles dialog (Figure 22–4), select the **admin** and **customer** roles and click OK.

**Figure 22–4 Select Roles Dialog**

14. Select each role and grant all the permissions (customize, grant, personalize, and view), as shown in Figure 22–5.
22.2.3 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:

```java
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.
3. In the Name field in the Create Java Class dialog, enter AppsSessionOptionsFactoryImpl.
4. Click OK.

   The AppsSessionOptionsFactoryImpl.java file is rendered in the Source view.
5. Import the following libraries:

   ```java
   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.CustizationClass;
   import oracle.mds.cust.CustimizationPolicy;
   
   public class AppsSessionOptionsFactoryImpl
   ```

   Add the following code to implement the ComposerSessionOptionsFactory class and provide SessionOptions:

   ```java
   implements ComposerSessionOptionsFactory
   {
   public AppsSessionOptionsFactoryImpl()
   ```

   For more information about creating users and roles, see the "Enabling ADF Security in a Fusion Web Application" chapter in Fusion Developer’s Guide for Oracle Application Development Framework.
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);
    }
}

private static final CustomizationClass[] EDIT_LAYER =
        { new SiteCC() };
private static final CustomizationClass[] VIEW_LAYER =
        { new SiteCC(), new UserCC() };}
22.2.4 How to Register the Implementation with Composer

Perform the steps described in Section 21.3.5, "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.

22.2.5 How to Configure WebCenterComposerFilter

To configure WebCenterComposerFilter, perform the steps described in Section 21.3.6, "How to Configure WebCenterComposerFilter."

22.2.6 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:

```xml
<pe:changeModeLink id="cml1"/>
<pe:pageCustomizable id="pageCustomizable1">
  <cust:panelCustomizable id="panelCustomizable1" layout="scroll">
    <f:facet name="editor">
      <pe:pageEditorPanel id="pep1"/>
    </f:facet>
  </cust:panelCustomizable>
</pe:pageCustomizable>
```

The design view should look like Figure 22–6.
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 15.2, “Creating Pages in a WebCenter Portal Framework Application.”

### 22.2.7 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 access the properties of the outer Panel Customizable, as shown in Figure 22–7. That is, users with admin role can edit this component.
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 Structure view, the Add Content and Edit icons are grayed out for the component.

22.3 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 22.3.1, "How to Enable Application Customization on an Image Component"
- Section 22.3.2, "How to Restrict Customization on an Image Component"
Applying Attribute-Level Security

For more information, see the "Extended Metadata Properties" section in Fusion Developer’s Guide for Oracle Application Development Framework.

22.3.1 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 18.2, "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. Select Structure to switch to Structure view of the page.

5. Select the image in the component hierarchy in the Structure View panel and click the Show the properties of 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.

22.3.2 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. Select Structure to switch to Structure view of the page.

5. Select the image in the component hierarchy in the Structure view panel, and click the Show the properties of icon.

   In the Component Properties dialog, the properties are grayed out and cannot be edited.

22.4 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
Applying Attribute-Level Security

For information about applying property filters, see Section 19.8.1, "How to Define Property Filters."

This section describes how to persist changes at the component level. It contains the following subsections:

■ Section 22.4.1, "How to Define Change Persistence at the Component Level"
■ Section 22.4.2, "What Happens at Runtime"

### 22.4.1 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 20.11, "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 22–1 shows a sample adf-config.xml file with persist-changes attributes defined for some attributes.

**Example 22–1 Using the persist-changes Attribute in the adf-config.xml File**

```xml
<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>
    </adf-faces-config>
</adf-config>
```

---

**Example 22–1 Using the persist-changes Attribute in the adf-config.xml File**

```xml
<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>
    </adf-faces-config>
</adf-config>
```

---

### 22.4.2 What Happens at Runtime

If you include the persist-changes attribute and set the value to true, then a change made to the component attribute is persisted in any instance of that component.
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 22.5.1, "How to Add an enableSecurity Section to adf-config.xml"
- Section 22.5.2, "Defining Security at the Actions Category Level"
- Section 22.5.3, "Defining Security at the Actions Level"

22.5.1 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 22–2, and set the enableSecurity element in that section to true.

Example 22–2 enableSecurity Element in the Customizable Components Security Section in adf-config.xml

```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 Oracle 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

---

**Note:** The persist-changes attribute applies only to implicit customization made in View mode.
Applying Action-Level Restrictions on Panel Customizable and Show Detail Component Actions

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.

### 22.5.2 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 22-1 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 22–1  Actions Categories and Actions Mapping**

<table>
<thead>
<tr>
<th>actionsCategory</th>
<th>Actions Supported</th>
</tr>
</thead>
<tbody>
<tr>
<td>personalizeActionsCategory</td>
<td>showMoveAction, showRemoveAction, showMinimizeAction, showResizer, allowAction</td>
</tr>
<tr>
<td>customizeActionsCategory</td>
<td>showEditAction, showAddContentAction, showSplitAction</td>
</tr>
</tbody>
</table>

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 22.5.3, “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 22–3 shows the actionsCategory entry that you can add to the customizable components actions security section in the adf-config.xml file.
### Example 22–3  **actionsCategory Element in the Customizable Components Security Section**

```xml
<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 22–4.

### Example 22–4  **EL Used in Customizable Components an actionCategory Entry**

```xml
<cust:actionsCategory>
  <cust:actionCategory name="personalizeActionsCategory" value="#{appBusinessRules.InsideCorpNetwork}"/>
</cust:actionsCategory>
```

For reference, the managed bean, `appBusinessRules`, is defined as shown in Example 74–7 in Section 74.15, "Overriding Inherited Security on Portlets and Customizable Components."

#### 22.5.3 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 22–2 explains the ability to add content depending on the `allowAction` and `showAddContentAction` settings.
Table 22–2  Actions Settings for Adding Content to a Page

<table>
<thead>
<tr>
<th>allowAction Setting</th>
<th>showAddContentAction Setting</th>
<th>Add Content Button Displayed?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>Yes</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>No</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>No</td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>No</td>
</tr>
</tbody>
</table>

Table 22–3 explains the ability to add content using the Add Box icons depending on the allowAction and showSplitAction settings.

Table 22–3  Actions Settings for Rearranging Content on a Page

<table>
<thead>
<tr>
<th>allowAction Setting</th>
<th>showSplitAction Setting</th>
<th>Add Box Icons Displayed?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>Yes</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>No</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>No</td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>No</td>
</tr>
</tbody>
</table>

Table 22–4 explains the ability to move content on the page depending on the allowAction and showMoveAction settings.

Table 22–4  Actions Settings for Moving Content on a Page

<table>
<thead>
<tr>
<th>allowAction Setting</th>
<th>showMoveAction Setting</th>
<th>Move Content on the Page?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>Yes</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>No</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>No</td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>No</td>
</tr>
</tbody>
</table>

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 Add Content or 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 Add Content or Design view of the page. However, users can still select such components in Structure view and edit or delete them.
Example 22–5 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 22–5  action Elements in the Customizable Components Security Section**

```xml
<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>
```

22.6 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 contains the following subsections:

- Section 22.6.1, "Granting Permissions on Task Flows"
- Section 22.6.2, "Enabling Task Flow Permission Check"

22.6.1 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 Resource Grants.
2. In the jazn-data.xml file, from Resource Type, select Task Flow.
3. From Resources list, select the task flow and click the Add Grantee icon in the Granted To column, then select the required role category, such as Add Application Role.
4. Select the roles to which you want to grant permissions or create a new role by clicking the Add icon.
5. Click OK.
6. Select one or more permissions in the Actions column (Figure 22–8).

**Figure 22–8 Task Flow Security Region in the jazn-data.xml File**

7. Save the jazn-data.xml file.

**Note:** The permission grant would appear as follows in the source view of the file if customize and view permissions are granted:

```xml
<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 16.8.1, "Page and Task Flow Security."

For more information about defining permission grants on task flows, see the "How to Define Security Policies for ADF Bounded Task Flows" section in *Fusion Developer’s Guide for Oracle Application Development Framework*.

### 22.6.2 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:
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.

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 contains the following sections:

- Section 22.7.1, "How to Create a Custom Security Policy"
- Section 22.7.2, "How to Register a Custom Policy with Composer"
- Section 22.7.3, "What Happens at Runtime"

22.7.1 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:

- 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 22–6.

**Example 22–6 Custom Policy Extending DefaultPageSecurityPolicy**

```java
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();
    }
}
```

### 22.7.2 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:

```xml
<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 22.6, "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.
22.7.3 What Happens at Runtime
Composer performs the security checks defined in your custom policy and enables customization on components accordingly.

22.8 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 22.6, "Implementing Task Flow Security."
This chapter describes how to extend or alter the look and feel or functionality of Oracle WebCenter Portal tools and services task flows using the Oracle JDeveloper Customization Developer role.

This chapter includes the following topics:

- Section 23.1, "Introduction to Task Flow Customization"
- Section 23.2, "Preparing for Task Flow Customization"
- Section 23.3, "Configuring the JDeveloper Customization Developer Role"
- Section 23.4, "Customizing Oracle WebCenter Portal Tools and Services Task Flows"
- Section 23.5, "Applying Task Flow Customizations to WebCenter Portal or Deployed WebCenter Portal Applications"
- Section 23.6, "Removing Customizations from Deployed WebCenter Portal Applications"
- Section 23.7, "Catalog of Customizable Oracle WebCenter Portal Tools and Services Task Flows"

23.1 Introduction to Task Flow Customization

When you apply task flow customizations to WebCenter Portal or a deployed Portal 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 JDeveloper, task flow customizations are deployed at application-level, so any customizations that you make will apply to all portals. To customize a task flow for a specific portal only, you must use WebCenter Portal's administration tools for task flow customization as explained in the "Customizing Task Flows for a Portal" chapter in Using Oracle WebCenter Portal. 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 portal, use the development-based customization approach explained in this chapter.
23.2 Preparing for Task Flow Customization

This section includes the following topics:

- **Section 23.2.1, "WebCenter Portal: Install the WebCenter Portal Customization JDeveloper Extension"**
- **Section 23.2.2, "WebCenter Portal: Create a Task Flow Customization Application"**
- **Section 23.2.3, "WebCenter Portal Framework Applications: Enable Customization"**

### 23.2.1 WebCenter Portal: Install the WebCenter Portal Customization JDeveloper Extension

To customize task flows for WebCenter Portal, you must first install the WebCenter Portal Customization JDeveloper extension.

The WebCenter Portal Customization JDeveloper extension includes the WebCenter Portal Task Flow Customization Application template and some Portal Server specific task flows that are not included in the Oracle WebCenter Portal Framework and Services Design Time JDeveloper extension bundle.

**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.

To install the WebCenter Portal Customization JDeveloper extension:

1. In JDeveloper, choose Help then Check for Updates.
2. In the Check for Updates wizard, if the Welcome page displays, click Next.
3. On the Source page, from the Search Update Centers list, select Oracle Fusion Middleware Products, and click Next.
5. Exit and restart JDeveloper.

### 23.2.2 WebCenter Portal: Create a Task Flow Customization Application

If you are customizing task flows for WebCenter Portal, you must also create a WebCenter Portal Task Flow Customization application.

To create a task flow customization application:

1. Access the Create WebCenter Portal Task Flow Customization Application wizard in any of the following ways:
From the File menu, choose New. In the New Gallery dialog, expand General, select Applications then WebCenter Portal Task Flow Customization Application, and then click OK.

From the Application menu, choose New. In the Create Generic Application wizard, in the Application Template list select WebCenter Portal Task Flow Customization Application.

If you have an existing application open, in the Application Navigator, click the application name and choose New Application. In the Create Generic Application wizard, in the Application Template list select WebCenter Portal Task Flow Customization Application.

If you have an existing application open, then in the Application Navigator, right-click the application name and choose New. In the New Gallery dialog, expand General, select Applications then WebCenter Portal Task Flow Customization Application, and then click OK.

2. In the Name your application page of the application creation wizard (Figure 23–1), in the Application Name field, enter a name for the application.

![Figure 23–1 The Create WebCenter Portal Task Flow Customization Application Wizard](image)

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\myCustomizationApplication

   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 the application.

5. Click Finish to create the task flow customization application with default project configurations.
23.2.3 WebCenter Portal Framework Applications: Enable Customization

To enable task flow customization in a Portal Framework application, first ensure that you have created your application using the WebCenter Portal Framework Application template (see Section 6.1, "Creating a New WebCenter Portal Framework Application"). Then follow the steps below.

To enable customization in a Portal Framework application:

1. In the Application Navigator, right-click the Portal project, and choose Project Properties.

2. In the Project Properties dialog, select ADF View, then select the Enable Seeded Customizations check box (Figure 23–2).

![Enable Seeded Customizations Option](image)

3. Click OK and save your files.

23.3 Configuring the JDeveloper Customization Developer Role

After you have created your customization application, or enabled customization for your application as described in Section 23.2, "Preparing for Task Flow Customization," you must configure the customization layer values to use the JDeveloper Customization Developer role.

1. In JDeveloper, choose Tools and then Preferences.

2. In the Preferences dialog, select Roles and then select the Customization Developer option (Figure 23–3).
3. Click OK.

JDeveloper prompts you to exit. Click Yes and then when you restart JDeveloper it will be using the Customization Developer role.

When started using the Customization Developer role, JDeveloper displays an icon that includes a small figure next to your application name in the Application Navigator (Figure 23–4).

**Figure 23–4  Icon Indicating Customization Developer Role**

You will also see a Customization Context window (Figure 23–5).

**Figure 23–5  Customization Context Window**
4. For Portal Framework applications, in the Customization Context window, click **Override global layer values**.

5. In the Confirm Override dialog, click **Yes**.

6. In the `CustomizationLayerValues.xml` file, enter the configuration for your application's customization class. For applications using the default ADF site customization class, use the following code:

   ```xml
   <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>
   ```

7. Save your files.

   The layer value in the Customization Context window is now set to the value that you defined in `CustomizationLayerValues.xml`. This indicates that you have successfully configured your application to enable customization of Oracle WebCenter Portal tools and services task flows.

8. In the Application Navigator, ensure that **Show Libraries** is enabled in the Navigator Display Options drop-down menu (Figure 23–6).

   **Tip:** The Navigator Display Options icon is on the far right of the Projects toolbar.

   ![Figure 23–6 Navigator Display Options - Show Libraries](image)

23.4 Customizing Oracle WebCenter Portal Tools and Services Task Flows

JDeveloper's Customization Developer role is a powerful mechanism that allows you to customize the ADF Library without changing the code in the base library JAR. Since both Oracle WebCenter Portal and Oracle ADF leverage MDS, you can use the Customization Developer role to extend WebCenter Portal tools and services task flows. All WebCenter Portal tools and services 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 23.3, "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 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 23.4.1, "Examples: Customizing Task Flows for WebCenter Portal" and Section 23.4.2, "Examples: Customizing Task Flows for WebCenter Portal Framework Applications."

- For a list of customizable task flows, see Section 23.7, "Catalog of Customizable Oracle WebCenter Portal Tools and Services Task Flows."

- After you have completed your customizations, deploy them as described in Section 23.5, "Applying Task Flow Customizations to WebCenter Portal or Deployed WebCenter Portal Applications."

### 23.4.1 Examples: Customizing Task Flows for WebCenter Portal

To customize task flows for WebCenter Portal, open the customization application you created in Section 23.2.2, "WebCenter Portal: Create a Task Flow Customization Application" and make your customizations.

This section provides examples that explain two common customizations. It includes the following topics:

- Section 23.4.1.1, "Example: Customizing the Worklist Task Flow"
- Section 23.4.1.2, "Example: Customizing the Discussion Forums Task Flow"
- Section 23.4.1.3, "Example: Customizing the Document Explorer Task Flow"

#### 23.4.1.1 Example: Customizing the Worklist Task Flow

This example shows how to customize the 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. Figure 23–7 shows a sample Worklist with the default two-row entry configuration.
To customize the Worklist task flow:

1. If you are not already using the Customization Developer role, switch to this role as described at the beginning of Section 23.3, "Configuring the JDeveloper Customization Developer Role."


3. In the Customization Context window, ensure that **Edit with following Customization Context** is selected and that **WebCenter (webcenter)** is selected as the customization layer (Figure 23–8).

4. In the Application Navigator, expand the **ViewController** project to view all the libraries available in the project.

   **Tip:** If you cannot see the libraries, click the Navigator Display Options icon in the Projects bar, and choose Show Libraries.

5. Expand the **WebCenter Worklist Service View** library and then **worklist-service-ui-component-view.jar**, **oracle.webcenter.worklist**, **view**, **jsf**, and **fragments**.

6. Right-click **worklist.jsff** and choose **Open**.

7. Switch to the list view facet of the view. Click the empty facet on the page and choose **Switch To - list** (Figure 23–9).
8. In the Structure window, expand the structure tree to locate the `af:outputText - #{row.title}` component. Right-click the component and choose **Insert Before af:outputText - #{row.title} > Link**.

9. In the Structure window, click on the newly created `af:commandLink` to open the Property Inspector dialog.

10. In the Property Inspector, click the down arrow next to the **Text** field and choose **Expression Builder** (Figure 23–10). Enter `#{row.title}` as the expression.

11. 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.
12. In the **ActionListener** field, enter `openTaskDetailsApp()`

13. Find the newly added **commandLink** and delete the following components:
   
   ```
   af:outputText - #{row.title}
   af:panelGroupLayout
   ```

14. The `worklist.jsff.xml` file that is generated (under the ADF Library Customizations node in the Application Navigator) should include code similar to that shown in **Example 23–1**.

**Example 23–1 Updated worklist.jsff file**

```xml
<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>
```

15. Deploy the metadata deployment profile to WebCenter Portal as explained in **Section 23.5, "Applying Task Flow Customizations to WebCenter Portal or Deployed WebCenter Portal Applications."**

**Figure 23–11** shows a sample Worklist after customization, with a single row and a popup.

**Figure 23–11 Worklist - After Customization**

![Worklist - After Customization](image)

23.4.1.2 **Example: Customizing the Discussion Forums Task Flow**

This example shows how you can customize the Discussion Forums task flow to display a profile image for the user who initiated the discussion. **Figure 23–12** shows a sample Discussion Forum without the profile image.
To customize the Discussion Forums task flow:

1. If you are not already using the Customization Developer role, switch to this role as described at the beginning of Section 23.3, "Configuring the JDeveloper Customization Developer Role."


3. In the Application Navigator, expand the ViewController project to view all the libraries available in the project.

   **Tip:** If you cannot see the libraries, click the Navigator Display Options icon in the Projects bar, and choose Show Libraries.

4. Expand the WebCenter Discussion Services View library and then forum-view.jar, oracle.webcenter.collab, view, forum, jsf, and fragments.

5. Right-click ListTopics.jsff and choose Open.

6. In the Structure window, search for the <rtc:presence> tag on the page. This tag renders the user name.

7. Right-click the rtc.:presence tag, then choose Insert After > ADF Faces and select Panel Group Layout from the dialog.

8. In the Structure window, right-click the new Panel Group Layout and choose Insert > Image Component.

9. 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.

10. Save your files and rebuild the project.

    The ListTopics.jsff.xml file that is generated will contain the following code:

    **Example 23–2  Updated ListTopics.jsff.xml File**

    ```xml
    <mds:insert customization version="11.1.1.60.46" xmlns:mds="http://xmlns.oracle.com/mds">
      <mds:insert after="pl" 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="swcii1" />
        </af:panelGroupLayout>
      </mds:insert>
    </mds:insert>
    </mds:customization>
    ```
11. Deploy the metadata deployment profile to the WebCenter Portal application as explained in Section 23.5, "Applying Task Flow Customizations to WebCenter Portal or Deployed WebCenter Portal Applications."

Figure 23–13 shows a sample Discussion Forum after customization, with a profile image.

![Discussion Forum - After Customization (With User Profile Image)](image)

23.4.1.3 Example: Customizing the Document Explorer Task Flow

The example that follows shows how to customize the Document Explorer task flow to display additional metadata for content items. Figure 23–14 shows the Document Explorer task flow, with its default set of fields.

![Document Explorer Task Flow - Before Customization](image)

To customize the Document Explorer task flow:

1. Configure the Documents service to expose additional custom metadata fields:
   a. Log in to the WebLogic Server Administration Console.
   b. Navigate to Environment > Servers > WC_Spaces.
   c. Open the Configuration > Server Start tab and in the Arguments field enter the server startup argument `oracle.webcenter.doclib.attributenames` and specify the metadata field that you want to add to the Document Explorer (this is the name as defined in Oracle WebCenter Content). For example:

   ```
   -Doracle.webcenter.doclib.attributenames=xState
   ```
d. Restart the WC_Spaces managed server.

2. In JDeveloper, if you are not already using the Customization Developer role, switch to this role as described at the beginning of Section 23.3, "Configuring the JDeveloper Customization Developer Role."


4. In the Application Navigator, expand the ViewController project to view all the libraries available in the project.

Tip: If you cannot see the libraries, click the Navigator Display Options icon in the Projects bar, and choose Show Libraries.

5. Expand the WebCenter Portal Document Library Service View library and then oracle.webcenter.doclib, view, jsf, taskflows, and folderViewer.

6. Right-click folderViewer.jsff and choose Open.

7. In the Source view, locate the table that lists the content items and add a column for the new attribute, for example:

```xml
<af:column id="stateColumn"
    headerText="State"
    nowrap="false"
    width="250"
    align="start"
    sortable="true"
    sortProperty="attrs.xState">
    <af:outputText id="state" value="#{item.attrs.xState}" />
</af:column>
```

8. Save your files and rebuild the project.

9. Deploy the metadata deployment profile to the WebCenter Portal application as explained in Section 23.5, "Applying Task Flow Customizations to WebCenter Portal or Deployed WebCenter Portal Applications."

Figure 23–15 shows a sample Document Explorer task flow after customization, with an additional State column.

Figure 23–15  Document Explorer Task Flow - After Customization

Tip: If you want to add more than one field, separate the names with commas.

Tip: If you cannot see the libraries, click the Navigator Display Options icon in the Projects bar, and choose Show Libraries.
23.4.2 Examples: Customizing Task Flows for WebCenter Portal Framework Applications

To customize task flows in a Portal Framework application, first follow the instructions in Section 23.2.3, "WebCenter Portal Framework Applications: Enable Customization."

This section provides examples for two possible customizations. It includes the following topics:

- Section 23.4.2.1, "Example: Customizing the Document Library - List View Task Flow"
- Section 23.4.2.2, "Example: Customizing the Content Presenter Task Flow"

23.4.2.1 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 Portal Framework application as described in Section 23.2.3, "WebCenter Portal Framework Applications: Enable Customization."

To customize the Document Library - List View task flow:

1. Open your Portal 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 **Portal** project to view all the libraries available in the project.
5. Expand the **WebCenter Document Library Service View** library and then **doclib-service-view.jar**, oracle.webcenter.doclib, view, jsf, and fragments.
6. Right-click **docListViewerTableTemplate.jsff** and choose **Open**.
7. Add a new column after **ITEM_NAME_COLUMN_HEADER**.
   a. In the Structure window, right-click the column with title `#{diBndi.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)"
      ```
8. Save your files.

### 23.4.2.2 Example: Customizing the Content Presenter Task Flow

This example shows how to customize the Content Presenter task flow to control the display of the border and toolbar in Composer Edit mode according to the permission of the current user. Before you perform these steps, ensure that you have prepared a customizable Portal Framework application as described in Section 23.2.3, "WebCenter Portal Framework Applications: Enable Customization."

To customize the Content Presenter task flow:

1. Open your Portal 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 **Documents - Content Presenter** and choose **Add to project**. If a confirmation dialog displays, click **Add Library**.

4. In the Application Navigator, expand the **Portal** project to view all the libraries available in the project.

   **Tip:** If you cannot see the libraries, click the **Navigator Display Options** icon in the Projects bar, and choose **Show Libraries**.

5. Expand the **WebCenter Document Library Service View** library and then **doclib-service-view.jar, oracle.webcenter.doclib, view, jsf, taskflows, and presenter**.

6. Right-click **presenterSingleView.jsff** and choose **Open**.

7. In Source view, edit the code for the main **af:panelGroupLayout** component as follows:

   ```xml
   <af:panelGroupLayout id="psvPgl1"
     partialTriggers="psvCtb3 psvPopup2 psvPopup3 psvPopup4 psvCbNp1 psvCbNp0"
     layout="vertical"
     inlineStyle="#{composerContext.subView == 'contentContribution'
       and pageServiceBean.canEdit
       ? 'border:2px dashed blue' : ''}">
   ``

   **Tip:** You can also change the border for **2px dashed blue** to a different style.

8. Edit the code for the **af:panelBox** component as follows:

   ```xml
   <af:panelBox id="psvPbNp0" showHeader="never" background="light"
     ramp="highlight" inlineStyle="width:350px"
     rendered="#{composerContext.subView == 'layout'
       and backingBeanScope.presenterBacking.nodePending
       and pageServiceBean.canEdit}"
   ``

9. Edit the code for the **af:toolbar - 1** component under the **contentContribution** facet as follows:

   ```xml
   <af:toolbar id="psvTb1" flex="1"
     stretchId="psvSp1" rendered="#{pageServiceBean.canEdit}"
   ```
10. Save your files.

23.5 Applying Task Flow Customizations to WebCenter Portal or Deployed WebCenter Portal Applications

The previous sections explained how to customize WebCenter Portal tools and services task flows for different scenarios. The output of these exercises is the generated MDS customization. The customizations show up as .xml or .jsff.xml files in the Portal or ViewController 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 application. 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 best practice to test your customizations in a development or stage environment first.

---

This section includes the following topics:

- **Section 23.5.1**, "WebCenter Portal Framework Applications: Creating a Metadata Deployment Profile"
- **Section 23.5.2**, "Deploying Task Flow Customizations Directly from JDeveloper"
- **Section 23.5.3**, "Deploying Task Flow Customizations Using WLST"

### 23.5.1 WebCenter Portal Framework Applications: Creating a Metadata Deployment Profile

For Portal Framework applications, you must first create a metadata deployment profile.

---

**Note:** These steps are not necessary if you are using the WebCenter Portal Task Flow Customization Application to customize task flow for WebCenter Portal.

---

To create a metadata deployment profile:

1. In the Application Navigator, right-click the application name and choose Application Properties (Figure 23–16).
2. Click Deployment and then click New to create a new metadata deployment profile.

3. In the Create Deployment Profile dialog (Figure 23–17), from the Archive Type drop-down list, select MAR File.

4. Click OK to close the Create Deployment Profile dialog.

5. Click OK, then click OK again to finish creating the deployment profile.

23.5.2 Deploying Task Flow Customizations Directly from JDeveloper
You can deploy your task flow customizations directly from JDeveloper.
To deploy task flow customizations directly from JDeveloper:
1. In the Application Navigator, right-click the application name and choose **Deploy** and then the metadata deployment profile (**Figure 23–18**).

**Figure 23–18  Deploying the Metadata Deployment Profile**

![Deploying the Metadata Deployment Profile](image)

2. In the **Deploy metadata** dialog (**Figure 23–19**), select **Export to a Deployed Application** and click **Next**.

**Figure 23–19  Deploy Metadata Dialog**

![Deploy Metadata Dialog](image)
3. On the **Application Server** page (Figure 23–20), 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 23–20  Deploy Metadata: Application Server*

4. On the **Server Instance** page (Figure 23–21), 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 23–21  Deploy Metadata Dialog: Server Instance*
5. On the **Deployed Application** page ([Figure 23–22](#)), select the application to which the customizations should be deployed then click **Next**.

![Figure 23–22 Deploy Metadata: Deployed Applications](image)

6. Click **Finish** to deploy the customization.

### 23.5.3 Deploying Task Flow Customizations Using WLST

You can also use the WLST command `importMetadata` to deploy task flow customizations.

To deploy task flow customizations using WLST:

1. In the Application Navigator, right-click the application name and choose **Deploy** and then the name of the metadata deployment profile ([Figure 23–23](#)).
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 target 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 command `importMetadata` command to import the task flow customizations to the application's MDS repository.

**Example 23–3** shows an example of how to run the `importMetadata` WLST command. For command syntax and detailed examples, see the "importMetadata" section in the *WebLogic Scripting Tool Command Reference*.

**Example 23–3  Command Line for Importing Customizations**

```bash
wls:/weblogic/serverConfig>importMetadata(application='webcenter',
server='WC_Spaces', fromLocation='/path/metadata.mar',
docs='/**')
```

For information on how to run WLST commands, see the "Running Oracle WebLogic Scripting Tool (WLST) Commands" section in *Administering Oracle WebCenter Portal*. 
23.6 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 command deleteMetadata to remove the applied customizations.

---

**Note:** Use the `deleteMetadata` command with caution as incorrect use of this command may cause the unintended loss of metadata documents. The sample command below removes the customization created in Section 23.4.1.2, "Example: Customizing the Discussion Forums Task Flow."

---

**Example 23-4** shows an example of how to run the `deleteMetadata` WLST command. For command syntax and detailed examples, see the "deleteMetadata" section in the WebLogic Scripting Tool Command Reference.

**Example 23–4  Command to Delete the Customization Metadata from a Deployed WebCenter Portal Application**

```python
deleteMetadata(application='webcenter', server='WC_Spaces',
    docs='/oracle/webcenter/webcenterapp/view/taskflows/discussionServices/**',
    excludeBaseDocs='true')
```

For information on how to run WLST commands, see the "Running Oracle WebLogic Scripting Tool (WLST) Commands" section in Administering Oracle WebCenter Portal.

23.7 Catalog of Customizable Oracle WebCenter Portal Tools and Services Task Flows

The following Oracle WebCenter Portal tools and services 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 23–1.

---

**Note:** To customize analytics task flows, you must add the following JARs to the project:

```text
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>
<thead>
<tr>
<th>Description</th>
<th>Library/JAR Files</th>
<th>Definition Path</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activity Stream - Mini View</td>
<td>WebCenter Portal Activity Streaming Service View</td>
<td>oracle/webcenter/activitystreaming/controller/taskflows/activity-streaming-miniview.xml</td>
<td>Activity stream table is a custom tag and cannot be customized.</td>
</tr>
<tr>
<td>Activity Stream - Main View</td>
<td>WebCenter Portal Activity Streaming Service View</td>
<td>oracle/webcenter/activitystreaming/controller/taskflows/activity-streaming-mainview.xml</td>
<td>Activity stream table is a custom tag and cannot be customized.</td>
</tr>
<tr>
<td>Analytics - Console</td>
<td>Task flows included in analytics-reporting-service-view.jar and</td>
<td>oracle/webcenter/analytics/view/js f/taskflows/seeded/analytics-console.xml</td>
<td></td>
</tr>
<tr>
<td>Analytics - WebCenter Portal Traffic</td>
<td>Task flows included in analytics-reporting-service-view.jar and</td>
<td>oracle/webcenter/analytics/view/js f/taskflows/seeded/analytics-report-service-summary.xml</td>
<td></td>
</tr>
<tr>
<td>Analytics - Page Traffic</td>
<td>Task flows included in analytics-reporting-service-view.jar and</td>
<td>oracle/webcenter/analytics/view/js f/taskflows/seeded/analytics-report-service-page-traffic.xml</td>
<td></td>
</tr>
<tr>
<td>Analytics - Login Metrics</td>
<td>Task flows included in analytics-reporting-service-view.jar and</td>
<td>oracle/webcenter/analytics/view/js f/taskflows/seeded/analytics-report-service-login.xml</td>
<td></td>
</tr>
<tr>
<td>Analytics - Portlet Traffic</td>
<td>Task flows included in analytics-reporting-service-view.jar and</td>
<td>oracle/webcenter/analytics/view/js f/taskflows/seeded/analytics-report-service-portlet-traffic.xml</td>
<td></td>
</tr>
<tr>
<td>Analytics - Portlet Response Time</td>
<td>Task flows included in analytics-reporting-service-view.jar and</td>
<td>oracle/webcenter/analytics/view/js f/taskflows/seeded/analytics-report-service-portlet-instance-resptime.xml</td>
<td></td>
</tr>
<tr>
<td>Analytics - Portlet Instance Traffic</td>
<td>Task flows included in analytics-reporting-service-view.jar and</td>
<td>oracle/webcenter/analytics/view/js f/taskflows/seeded/analytics-report-service-portlet-instance-traffic.xml</td>
<td></td>
</tr>
<tr>
<td>Analytics - Portlet Instance Response Time</td>
<td>Task flows included in analytics-reporting-service-view.jar and</td>
<td>oracle/webcenter/analytics/view/js f/taskflows/seeded/analytics-report-service-portlet-instance-resptime.xml</td>
<td></td>
</tr>
<tr>
<td>Analytics - Search Metrics</td>
<td>Task flows included in analytics-reporting-service-view.jar and</td>
<td>oracle/webcenter/analytics/view/js f/taskflows/seeded/analytics-report-service-search-phrases.xml</td>
<td></td>
</tr>
<tr>
<td>Analytics - Document Metrics</td>
<td>Task flows included in analytics-reporting-service-view.jar and</td>
<td>oracle/webcenter/analytics/view/js f/taskflows/seeded/analytics-report-service-document-view.xml</td>
<td></td>
</tr>
<tr>
<td>Analytics - Wiki Metrics</td>
<td>Task flows included in analytics-reporting-service-view.jar and</td>
<td>oracle/webcenter/analytics/view/js f/taskflows/seeded/analytics-report-service-wiki-view.xml</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Library/JAR Files</td>
<td>Definition Path</td>
<td>Notes</td>
</tr>
<tr>
<td>-------------</td>
<td>------------------</td>
<td>----------------</td>
<td>-------</td>
</tr>
<tr>
<td>Analytics - Blog Metrics</td>
<td>Task flows included in analytics-reporting-service-view.jar and analytics-reporting-service-model.jar</td>
<td>oracle/webcenter/analytics/view/jsf/taskflows/seeded/analytics-reports-view.xml</td>
<td></td>
</tr>
<tr>
<td>Analytics - Discussion Metrics</td>
<td>Task flows included in analytics-reporting-service-view.jar and analytics-reporting-service-model.jar</td>
<td>oracle/webcenter/analytics/view/jsf/taskflows/seeded/analytics-reports-discussion-view.xml</td>
<td></td>
</tr>
<tr>
<td>Announcements</td>
<td>WebCenter Portal Announcement Service View</td>
<td>oracle/webcenter/collab/announcement-service-view/taskflows/main-view-definition.xml</td>
<td></td>
</tr>
<tr>
<td>Announcements - Quick View</td>
<td>WebCenter Portal Announcement Service View</td>
<td>oracle/webcenter/collab/announcement-service-view/taskflows/main-view-definition.xml</td>
<td></td>
</tr>
<tr>
<td>Application Navigator</td>
<td>WebCenter Portal View</td>
<td>oracle/webcenter/webcenterapp/view/taskflows/applinks/AppNavigatorRegion.xml</td>
<td></td>
</tr>
<tr>
<td>Blogs</td>
<td></td>
<td>oracle/webcenter/blog/view/jsf/taskflows/blogDigestViewer/blog-main-view.xml</td>
<td></td>
</tr>
<tr>
<td>Calendar Main View</td>
<td>WebCenter Portal Events Service View</td>
<td>oracle/webcenter/collab/events/view/taskflows/calendar-main-view.xml</td>
<td></td>
</tr>
<tr>
<td>Calendar Mini View</td>
<td>WebCenter Portal Events Service View</td>
<td>oracle/webcenter/collab/events/view/taskflows/calendar-mini-view.xml</td>
<td></td>
</tr>
<tr>
<td>ChooseLanguageTaskflow</td>
<td>WebCenter Portal View</td>
<td>oracle/webcenter/webcenterapp/view/taskflows/translations/ChooseLanguageTaskflow.xml</td>
<td></td>
</tr>
<tr>
<td>CommunityBrowserRegion</td>
<td>WebCenter Portal View</td>
<td>oracle/webcenter/community/view/taskflows/browser/CommunityBrowserRegion.xml</td>
<td></td>
</tr>
<tr>
<td>Community-contacts-taskflow</td>
<td>WebCenter Portal View</td>
<td>oracle/webcenter/people/view/jsf/regions/community-contacts-taskflow.xml</td>
<td></td>
</tr>
<tr>
<td>Connections - Detailed View</td>
<td>WebCenter Portal PeopleConnections View</td>
<td>oracle/webcenter/peopleconnections/connections/controller/taskflows/connections-mini-view-advanced-taskflow.xml</td>
<td></td>
</tr>
<tr>
<td>Connections</td>
<td>WebCenter Portal PeopleConnections View</td>
<td>oracle/webcenter/peopleconnections/connections/controller/taskflows/connections-main-view-taskflow.xml</td>
<td></td>
</tr>
<tr>
<td>Connections - Quick View</td>
<td>WebCenter Portal PeopleConnections View</td>
<td>oracle/webcenter/peopleconnections/connections/controller/taskflows/connections-main-view-untabbed-taskflow.xml</td>
<td></td>
</tr>
<tr>
<td>Connections - Mini View</td>
<td>WebCenter Portal PeopleConnections View</td>
<td>oracle/webcenter/peopleconnections/connections/controller/taskflows/connections-mini-view-taskflow.xml</td>
<td></td>
</tr>
<tr>
<td>customization-manager-taskflow</td>
<td>Oracle Composer</td>
<td>oracle/adfinternal/pageeditor/manager/taskflows/customization-manager-taskflow.xml</td>
<td></td>
</tr>
<tr>
<td>Discussion Forums</td>
<td>WebCenter Portal Discussions Service View</td>
<td>oracle/webcenter/collab/forum/view/taskflows/main-task-flow.xml</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Library/JAR Files</td>
<td>Definition Path</td>
<td>Notes</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>---------------------------------------------</td>
<td>--------------------------------------------------------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>Discussions - Recent Topics</td>
<td>WebCenter Portal Discussions Service View</td>
<td>oracle/webcenter/collab/forum/view/taskflows/recentTopic-task-flow.xml</td>
<td></td>
</tr>
<tr>
<td>Discussions - Quick View</td>
<td>WebCenter Portal Discussions Service View</td>
<td>oracle/webcenter/collab/forum/view/taskflows/miniview-task-flow.xml</td>
<td></td>
</tr>
<tr>
<td>Documents - Content Presenter</td>
<td>WebCenter Portal Document Library Service View</td>
<td>oracle/webcenter/doclib/view/jsf/taskflows/presenter/contentPresenter.xml</td>
<td></td>
</tr>
<tr>
<td>Documents - Main View</td>
<td>WebCenter Portal Document Library Service View</td>
<td>oracle/webcenter/doclib/view/jsf/taskflows/mainView.xml</td>
<td></td>
</tr>
<tr>
<td>Documents - Recent Documents</td>
<td>WebCenter Portal Document Library Service View</td>
<td>oracle/webcenter/doclib/view/jsf/taskflows/recentDocuments.xml</td>
<td></td>
</tr>
<tr>
<td>Documents - Rich Text Editor</td>
<td>WebCenter Portal Document Library Service View</td>
<td>oracle/webcenter/doclib/view/jsf/taskflows/richTextEditor/editor.xml</td>
<td></td>
</tr>
</tbody>
</table>
### Table 23–1 (Cont.) Oracle WebCenter Portal Tools and Services Customizable Task Flows

<table>
<thead>
<tr>
<th>Description</th>
<th>Library/JAR Files</th>
<th>Definition Path</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Events</td>
<td>WebCenter Portal Events Service View</td>
<td>oracle/webcenter/collab/events/view/taskflows/calendar-main-view.xml</td>
<td></td>
</tr>
<tr>
<td>Exportregion</td>
<td>Internal - invoked by WebCenter Portal administrator for export of portals or portal templates.</td>
<td>oracle/webcenter/webcenterapp/view/taskflows/lifecycle/Exportregion.xml</td>
<td></td>
</tr>
<tr>
<td>Feedback</td>
<td>WebCenter Portal PeopleConnections View</td>
<td>oracle/webcenter/peopleconnections/kudos/controller/taskflows/KudosDetailViewer.xml</td>
<td></td>
</tr>
<tr>
<td>Feedback - Quick View</td>
<td>WebCenter Portal PeopleConnections View</td>
<td>oracle/webcenter/peopleconnections/kudos/controller/taskflows/KudosMiniViewer.xml</td>
<td></td>
</tr>
<tr>
<td>Links</td>
<td>WebCenter Portal Links Service View</td>
<td>oracle/webcenter/relationship/view/jsf/resources/links-detail.xml</td>
<td></td>
</tr>
<tr>
<td>Links Dialog</td>
<td>WebCenter Portal Links Service View</td>
<td>oracle/webcenter/relationship/view/jsf/resources/links-detail-popup.xml</td>
<td></td>
</tr>
<tr>
<td>List - Main View</td>
<td>WebCenter Portal List Service View</td>
<td>oracle/webcenter/list/view/jsf/regions/main-view-task-flow.xml</td>
<td></td>
</tr>
<tr>
<td>list-instance-view-task-flow</td>
<td>WebCenter Portal Lists Service View</td>
<td>oracle/webcenter/list/view/jsf/regions/list-instance-view-task-flow.xml</td>
<td>Table columns in this task flow are set up dynamically at runtime and should not be customized.</td>
</tr>
<tr>
<td>Message Board - Quick View</td>
<td>WebCenter Portal PeopleConnections View</td>
<td>oracle/webcenter/peopleconnections/wall/controller/taskflows/WallViewer.xml</td>
<td></td>
</tr>
<tr>
<td>Message Board</td>
<td>WebCenter Portal PeopleConnections View</td>
<td>oracle/webcenter/peopleconnections/wall/controller/taskflows/WallDetailViewer.xml</td>
<td></td>
</tr>
<tr>
<td>Organization View</td>
<td>WebCenter Portal PeopleConnections View</td>
<td>oracle/webcenter/peopleconnections/profile/view/jsf/regions/orgview/organization-view.xml</td>
<td></td>
</tr>
<tr>
<td>Page - Create New</td>
<td>WebCenter Portal Page Service View</td>
<td>oracle/webcenter/page/view/jsf/fragments/page-create-page.xml</td>
<td></td>
</tr>
<tr>
<td>Polls - Polls Manager</td>
<td>WebCenter Portal Polls and Surveys Service View</td>
<td>oracle/webcenter/collab/survey/view/jsf/taskflows/list-surveys-definition.xml</td>
<td></td>
</tr>
<tr>
<td>Polls - Quick View</td>
<td>WebCenter Portal Polls and Surveys Service View</td>
<td>oracle/webcenter/collab/survey/view/jsf/taskflows/quick-poll-definition.xml</td>
<td></td>
</tr>
<tr>
<td>Polls - Take Poll</td>
<td>WebCenter Portal Polls and Surveys Service View</td>
<td>oracle/webcenter/collab/survey/view/jsf/taskflows/take-polls-definition.xml</td>
<td></td>
</tr>
<tr>
<td>Polls - View Poll Results</td>
<td>WebCenter Portal Polls and Surveys Service View</td>
<td>oracle/webcenter/collab/survey/view/jsf/taskflows/view-results-definition.xml</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Library/JAR Files</td>
<td>Definition Path</td>
<td>Notes</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------------------------------------</td>
<td>---------------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>Profile Gallery</td>
<td>WebCenter Portal PeopleConnections View</td>
<td>oracle/webcenter/peopleconnections/personalweb/view/jsf/regions/profile-gallery.xml</td>
<td></td>
</tr>
<tr>
<td>Profile</td>
<td>WebCenter Portal PeopleConnections View</td>
<td>oracle/webcenter/peopleconnections/view/jsf/regions/extended/extended-profile.xml</td>
<td></td>
</tr>
<tr>
<td>Profile - Snapshot</td>
<td>WebCenter Portal PeopleConnections View</td>
<td>oracle/webcenter/peopleconnections/view/jsf/regions/profile-snapshot.xml</td>
<td></td>
</tr>
<tr>
<td>Publisher</td>
<td>WebCenter Portal PeopleConnections View</td>
<td>oracle/webcenter/peopleconnections/wall/controller/taskflows/Publisher.xml</td>
<td></td>
</tr>
<tr>
<td>Recommended Connections</td>
<td>WebCenter Portal Activity Graph Service View</td>
<td>view/oracle/webcenter/activitygraph/controller/taskflows/recommended-connections.xml</td>
<td>This task flow is mostly generated by code; the content is only customizable through resource files and metadata mappings.</td>
</tr>
<tr>
<td>- Resource Viewer</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Recent Activities</td>
<td>WebCenter Portal Recent Activity Service View</td>
<td>oracle/webcenter/recentactivity/controller/taskflows/recent-activities.xml</td>
<td></td>
</tr>
<tr>
<td>RSS Viewer</td>
<td>WebCenter Portal RSS Service View</td>
<td>oracle/webcenter/rssviewer/view/jsf/fragments/RSSViewerTaskFlow.xml</td>
<td></td>
</tr>
<tr>
<td>Search</td>
<td>WebCenter Portal Search Services View</td>
<td>oracle/webcenter/search/controller/taskflows/searchResults.xml</td>
<td></td>
</tr>
<tr>
<td>Search Preferences</td>
<td>WebCenter Portal Search Services View</td>
<td>oracle/webcenter/search/controller/taskflows/preferences.xml</td>
<td></td>
</tr>
<tr>
<td>Search - Saved Searches</td>
<td>WebCenter Portal Search Services View</td>
<td>oracle/webcenter/search/controller/taskflows/allSavedSearches.xml</td>
<td></td>
</tr>
<tr>
<td>Search Toolbar</td>
<td>WebCenter Portal Search Services View</td>
<td>oracle/webcenter/search/controller/taskflows/localToolbarSearch.xml</td>
<td></td>
</tr>
<tr>
<td>- My Impersonators</td>
<td>WebCenter Common View</td>
<td>oracle/webcenter/security/view/impersonation/jsf/taskflows/impersonation-myimpersonators-task-flow-definition.xml</td>
<td></td>
</tr>
<tr>
<td>Security - Self Registration</td>
<td>WebCenter Common View</td>
<td>oracle/webcenter/security/view/selfregistration/jsf/taskflows/selfregistration-task-flow-definition.xml</td>
<td>Public user registration used in WebCenter Portal and custom applications; can be customized in both.</td>
</tr>
<tr>
<td>Security - Enterprise Role - Members</td>
<td>WebCenter Common View</td>
<td>oracle/webcenter/security/view/rolemembers/jsf/taskflows/rolemembers-task-flow-definition.xml</td>
<td>Lists the members of an enterprise group. Used in WebCenter Portal and custom applications; can be customized in both.</td>
</tr>
<tr>
<td>Description</td>
<td>Library/JAR Files</td>
<td>Definition Path</td>
<td>Notes</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>-------------------------------------------</td>
<td>---------------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>Security - Enterprise Role - Members Search</td>
<td>WebCenter Common View</td>
<td>oracle/webcenter/security/view/rolemembers/jsf/taskflows/rolemembers-psearch-task-flow-definition.xml</td>
<td>Returns the members of an enterprise group of a particular pattern. Used in WebCenter Portal and custom applications; can be customized in both.</td>
</tr>
<tr>
<td>Security - Enterprise Role - Members Viewer</td>
<td>WebCenter Common View</td>
<td>oracle/webcenter/security/view/rolemembers/jsf/taskflows/rolemembers-browser-task-flow-definition.xml</td>
<td>A tabbed page; one tab shows the members of a group and the other provides a search form. Used in WebCenter Portal and custom applications; can be customized in both.</td>
</tr>
<tr>
<td>Security - Role Manager</td>
<td>WebCenter Common View</td>
<td>oracle/webcenter/security/view/rolemanager/jsf/taskflows/global-role-manager-task-flow-definition.xml</td>
<td>Interface to create, modify and delete application roles. Used in custom applications only.</td>
</tr>
<tr>
<td>Security - External Application Change Password</td>
<td>WebCenter Portal External Application Service View</td>
<td>oracle/adfinternal/extapp/view/fragments/extapp-change-password-taskflow.xml</td>
<td>Screen to change the password for all external applications created. Used in WebCenter Portal and custom applications; can be customized in both.</td>
</tr>
<tr>
<td>Similar Items</td>
<td>WebCenter Portal Activity Graph Service View</td>
<td>view/oracle/webcenter/activitygraph/controller/taskflows/similar-items.xml</td>
<td>This task flow is mostly generated by code; the content is only customizable through resource files and metadata mappings.</td>
</tr>
<tr>
<td>Similar Portals</td>
<td>WebCenter Portal Activity Graph Service View</td>
<td>view/oracle/webcenter/activitygraph/controller/taskflows/similar-group-space.xml</td>
<td>This task flow is mostly generated by code; the content is only customizable through resource files and metadata mappings.</td>
</tr>
<tr>
<td>Portals</td>
<td>WebCenter Portal View</td>
<td>oracle/webcenter/community/view/taskflows/browsers/CommunityBrowserRegion.xml</td>
<td></td>
</tr>
<tr>
<td>Portal Members</td>
<td>WebCenter Portal View</td>
<td>oracle/webcenter/webcenterapp/view/taskflows/admin/table-of-members-taskflow.xml</td>
<td></td>
</tr>
<tr>
<td>Subscription Preferences</td>
<td>WebCenter Portal Notification Service View</td>
<td>oracle/webcenter/notification/view/jsf/regions/SubscriptionPreferences.xml</td>
<td>The subscription preferences table in this task flow cannot be customized (use af:iterator).</td>
</tr>
<tr>
<td>Subscription Viewer</td>
<td>WebCenter Portal Notification Service View</td>
<td>oracle/webcenter/notification/view/jsf/regions/SubscriptionsViewer.xml</td>
<td></td>
</tr>
<tr>
<td>Tag Cloud</td>
<td>WebCenter Portal Tagging Service View</td>
<td>oracle/webcenter/tagging/controller/taskflows/tag-selection.xml</td>
<td></td>
</tr>
<tr>
<td>Tagging Dialog</td>
<td>WebCenter Portal Tagging Service View</td>
<td>oracle/webcenter/tagging/controller/taskflows/launch-dialog.xml</td>
<td></td>
</tr>
<tr>
<td>Tagged Items</td>
<td>WebCenter Portal Tagging Service View</td>
<td>oracle/webcenter/tagging/controller/taskflows/related-resources.xml</td>
<td></td>
</tr>
<tr>
<td>Tagging - Personal View</td>
<td>WebCenter Portal Tagging Service View</td>
<td>oracle/webcenter/tagging/controller/taskflows/tagging-personal-view.xml</td>
<td></td>
</tr>
<tr>
<td>Tagging - Related Links</td>
<td>WebCenter Portal Tagging Service View</td>
<td>oracle/webcenter/tagging/controller/taskflows/related-links.xml</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Library/JAR Files</td>
<td>Definition Path</td>
<td>Notes</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------</td>
<td>------------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>Tagging - Similar Items</td>
<td>WebCenter Portal Tagging Service View</td>
<td>oracle/webcenter/tagging/controller/taskflows/related-links.xml</td>
<td></td>
</tr>
<tr>
<td>Top Items</td>
<td>WebCenter Portal Activity Graph Service View</td>
<td>oracle/webcenter/activitygraph/controller/taskflows/top-items.xml</td>
<td>This task flow is mostly generated by code; the content is only customizable through resource files and metadata mappings.</td>
</tr>
<tr>
<td>Worklist</td>
<td>WebCenter Portal Worklist Service View</td>
<td>oracle/webcenter/worklist/view/jsf/taskFlowDefs/worklist.xml</td>
<td></td>
</tr>
</tbody>
</table>
Part IV contains the following chapters:

- Chapter 24, "Introduction to Integrating and Publishing Content"
- Chapter 25, "Configuring Content Repository Connections"
- Chapter 26, "Working with Content Data Controls"
- Chapter 27, "Creating Content Presenter Display Templates"
- Chapter 28, "Integrating Documents"
- Chapter 29, "Adding Content Task Flows and Document Components to a Portal Page"
- Chapter 30, "Integrating Wikis and Blogs"
- Chapter 31, "Content Management REST API"
This chapter describes the various ways to integrate content into a WebCenter Portal Framework application: content data controls, Content Management Interoperability Services (CMIS) REST APIs, Content Presenter, and the Documents tool. 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 Portal Framework application using any of the following methods (for more information, see the references at the end of this chapter):

- **Documents** task flows and components offer a variety of formats to display folders and files, including wikis and blogs, on a page in a Portal 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 WebCenter Content, 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 25.2.3.1, "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 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 task flows built into the application to manage, display, and search documents at runtime.

- **Content Presenter** enables you to precisely customize the selection and presentation of content in a Portal Framework application. The Content Presenter task flow is available only when the connected content repository is WebCenter Content and your WebCenter Portal administrator has completed the requisite 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 Portal Framework application. Content Presenter has no dependency on the Documents tool for adding or managing the content it displays.

- **CMIS REST APIs** surface and manage content in WebCenter Content.
Content data controls use JCR adapters to enable read-only access to content in a WebCenter Content, Oracle Portal, or SharePoint content repository, and maintain tight control over the way the content displays in a Portal 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 tool and its task flows.

Table 24–1 provides a comparative overview of these methods to help you select the most appropriate method for your needs.

| Table 24–1  Methods of Integrating Content into a Framework Application |
|---|---|---|---|---|
| **Content Data Controls** | **Content Management REST APIs** | **Content Presenter** | **Documents Tool** |
| **Repository** | WebCenter Content | WebCenter Content | WebCenter Content |
| | Oracle Portal | | Oracle Portal |
| | SharePoint | | 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 | | Supports Oracle Site Studio region definitions-based content |
| **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. | Task flow-based component intended primarily for rendering content. |
| | Surface content through single item selection, by folder, and by query results. | Surface content through single item selection, by folder, and by query results. | Task flow-based component intended primarily for collaborating and managing content |
| | | | WebCenter Content only: |
| | | | Document preview |
| | | | View links |
| | | | View images |
| | | | View file information (properties, version history) |
| | | | View metadata |

24-2 Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper
The following chapters provide information that you will need for any method you use:

- "Managing Content Repositories" in *Administering Oracle WebCenter Portal* to configure and manage content repositories used by WebCenter Portal applications.

- Chapter 25, "Configuring 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 26, "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 27, "Creating Content Presenter Display Templates"

- Chapter 29, "Adding Content Task Flows and Document Components to a Portal Page"

- "Publishing Content Using Content Presenter" in *Building Portals with Oracle WebCenter Portal* to work with content at runtime in WebCenter Portal applications.

---

### Table 24–1 (Cont.) Methods of Integrating Content into a Framework Application

<table>
<thead>
<tr>
<th>Content Management</th>
<th>Content Data Controls</th>
<th>Content Management REST APIs</th>
<th>Content Presenter</th>
<th>Documents Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>Manage content using REST APIs and custom client or server side application code</td>
<td>In-context contribution editing for HTML and Site Studio content</td>
<td>Manage content through graphical user interface</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Create, update, delete folders and content files and associated metadata fields</td>
<td></td>
<td>Create, update, delete folders and content files</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>WebCenter Content only:</td>
<td>WebCenter Content only:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Manage associated metadata properties</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Item-level security</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Oracle Workflow (WebCenter Portal only)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- View metadata</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Benefits</td>
<td>Standard JCR API integrates with many different content repositories.</td>
<td>Flexible REST-based APIs useful for client-side style development</td>
<td>Flexible display using display templates Oracle Site Studio support</td>
<td>Choice of task flows to provide easy UI access to managing content</td>
</tr>
<tr>
<td>Limitations</td>
<td>Read-only content No Oracle Site Studio support</td>
<td>Content must reside in Oracle Content Server repository No Oracle Site Studio support</td>
<td>Content must reside in WebCenter Content repository No Oracle Site Studio support</td>
<td></td>
</tr>
</tbody>
</table>
The following chapters provide information about using the \textit{Documents tool}, which includes the documents task flows, document components (links, inline frames, and images), wikis, and blogs:

- Chapter 28, "Integrating Documents"
- Chapter 29, "Adding Content Task Flows and Document Components to a Portal Page"
- Chapter 30, "Integrating Wikis and Blogs"
- "Working with Content in a Portal" in \textit{Building Portals with Oracle WebCenter Portal} to work with content at runtime in WebCenter Portal applications.
This chapter describes how to configure connections to content repositories to provide access to the content, and how to add content to pages in a Portal Framework application.

For information about other ways in which to integrate content into a Portal Framework application, see Chapter 24, "Introduction to Integrating and Publishing Content."

This chapter includes the following sections:
- Section 25.1, "Overview of Content Adapters"
- Section 25.2, "Configuring Content Repository Connections"
- Section 25.3, "Editing Content Repository Connections"
- Section 25.4, "Using an Existing Repository Connection for a New Portal Framework Application"

25.1 Overview of Content Adapters

In your Portal 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 25.2.3.1, "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.
25.2 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 25.2.1, "Creating a Content Repository Connection Based on the Oracle Content Server Adapter"
- Section 25.2.2, "Creating a Content Repository Connection Based on the Oracle Portal Adapter"
- Section 25.2.3, "Creating a Content Repository Connection Based on the Oracle WebCenter Adapter for SharePoint"
- Section 25.2.4, "Creating a Content Repository Connection Based on the File System Adapter"
- Section 25.2.5, "What Happens When You Create a Repository Connection"
- Section 25.2.6, "About Creating a Repository Connection"

25.2.1 Creating 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 11.1.1.8.0 or later. For information about how to configure Content Server for Oracle WebCenter Portal, see the "Configuring an Oracle WebCenter Content Server Repository" section in Administering Oracle WebCenter Portal.

For an overview of the prerequisites and tasks required to get Content Server working in a portal or Framework application, see the "Configuration Roadmap for Content Server" section in Administering Oracle WebCenter Portal.

To create a Content Server-based repository connection:

1. In Oracle JDeveloper, open your Portal 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.
Configuring Content Repository Connections

**Note:** If you plan to deploy your portal to a managed WebLogic Server instance for which a connection is already established (for example, by an administrator using WLST), then you must use the same **Connection Name** here as was used for the managed server connection. Consult with your portal system administrator if you are unsure about the correct connection name.

**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**, to make this connection the primary (default) connection for all content management operations through the documents tool 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 task flow, that value overrides this setting.

**Note:** To be able to set a connection as the primary (default) connection for documents, 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 25–1 and Figure 25–1.

### Table 25–1 Configuration Parameters for Content Server

<table>
<thead>
<tr>
<th>Configuration Parameters</th>
<th>Values</th>
</tr>
</thead>
</table>
| 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.  
  `jaxws`: Uses an HTTP(S) connection to connect to Content Server.  
  *Table 25–2* 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.

Note: Although the intradocport value for a particular Content Server defaults to 4444, that value may change depending on the Content Server configuration.

URL of the Web Server Plugin

If the RIDC socket type is web, then the URL must be:

For Content Server release 11.1.1.8.0: http://host_name:port_number/_dav/web_root/plugin_root. For example: http://mycontentserver:4444/_dav/cms/idcplg

Web Server Context Root

The Web server context root for Content Server to integrate advanced metadata and Site Studio capabilities in the Portal 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 connections 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 the "Getting Started with Integrating WebCenter Content into Your Environment" section in Developing with Oracle WebCenter Content.

For information about predefined security policies, see the "Security Policies" section in Security and Administrator’s Guide for Web Services.

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.

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 25-1 (Cont.) Configuration Parameters for Content Server

<table>
<thead>
<tr>
<th>Configuration Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content Server Listener Port</td>
<td>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. Note: Although the intradocport value for a particular Content Server defaults to 4444, that value may change depending on the Content Server configuration.</td>
</tr>
<tr>
<td>URL of the Web Server Plugin</td>
<td>If the RIDC socket type is web, then the URL must be: http://host_name:port_number/_dav/web_root/plugin_root. For example: <a href="http://mycontentserver:4444/_dav/cms/idcplg">http://mycontentserver:4444/_dav/cms/idcplg</a></td>
</tr>
<tr>
<td>Web Server Context Root</td>
<td>The Web server context root for Content Server to integrate advanced metadata and Site Studio capabilities in the Portal Framework application. The format of the Web server context root is /context_root. For example, /cs.</td>
</tr>
<tr>
<td>Admin Username</td>
<td>The user name with administrative rights for the Content Server instance. It defaults to sysadmin.</td>
</tr>
<tr>
<td>Admin Password</td>
<td>The password for the Content Server admin user. For example: password. This is needed for web connections only.</td>
</tr>
<tr>
<td>KeyStore File Location</td>
<td>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.</td>
</tr>
<tr>
<td>KeyStore Password</td>
<td>The password required to access the keystore.</td>
</tr>
<tr>
<td>Private Key Alias</td>
<td>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.</td>
</tr>
<tr>
<td>Private Key Password</td>
<td>The client private key password required to retrieve the key from the keystore.</td>
</tr>
<tr>
<td>JAX-WS Client Security Policy</td>
<td>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 the “Getting Started with Integrating WebCenter Content into Your Environment” section in Developing with Oracle WebCenter Content. For information about predefined security policies, see the “Security Policies” section in Security and Administrator’s Guide for Web Services.</td>
</tr>
<tr>
<td>Cache Invalidation Interval</td>
<td>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.</td>
</tr>
<tr>
<td>Binary Cache Maximum Entry Size</td>
<td>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).</td>
</tr>
</tbody>
</table>
### Table 25–2  Content Server Connection Parameters for Each RIDC Socket Type

<table>
<thead>
<tr>
<th>Connection Parameters</th>
<th>RIDC Socket Type: web</th>
<th>RIDC Socket Type: socket</th>
<th>RIDC Socket Type: socketssl</th>
<th>RIDC Socket Type: jaxws</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server Host Name</td>
<td>Not Applicable</td>
<td>Mandatory for Content Presenter, Defaults to local host.</td>
<td>Mandatory for Content Presenter, Defaults to local host.</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Content Server Listener Port</td>
<td>Not Applicable</td>
<td>Port specified for the incoming provider in the server. Defaults to 4444. *Note: Although the intradocport value for a particular Content Server defaults to 4444, that value may change depending on the Content Server configuration.</td>
<td>Port specified for the ssincomings provider in the server. Defaults to 4444.</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>URL of the Web Server Plugin</td>
<td>Mandatory</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>Mandatory Use format: [<a href="http://hostname">http://hostname</a> :port/web_root] For example: [http(s)://myhost.com:9044/idcnativeviews]</td>
</tr>
<tr>
<td>Web Server Context Root</td>
<td>Optional</td>
<td>Optional</td>
<td>Optional</td>
<td>Optional</td>
</tr>
<tr>
<td>Connection Parameters</td>
<td>RIDC Socket Type: web</td>
<td>RIDC Socket Type: socket</td>
<td>RIDC Socket Type: socketssl</td>
<td>RIDC Socket Type: jaxws</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------</td>
<td>------------------------</td>
<td>---------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>JAX-WS Client Security Policy</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>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 “Security Policies” section in the Security and Administrator’s Guide for Web Services.</td>
</tr>
<tr>
<td>Admin Username</td>
<td>Mandatory</td>
<td>Optional</td>
<td>Optional</td>
<td>Optional</td>
</tr>
<tr>
<td></td>
<td>Defaults to sysadmin</td>
<td>Defaults to sysadmin</td>
<td>Defaults to weblogic</td>
<td></td>
</tr>
<tr>
<td>Admin Password</td>
<td>Mandatory</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>Whether the password is used or not depends on the selected JAX-WS security policy.</td>
</tr>
<tr>
<td>Key Store and Private Key</td>
<td>Not Applicable</td>
<td>Not Applicable</td>
<td>Mandatory</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Identity Propagation</td>
<td>Not supported at runtime</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported, provided that the JAX-WS security policy supports identity propagation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For testing purpose, connects as guest if no/invalid user name is specified.</td>
<td>For testing purpose, connects as guest if no/invalid user name is specified.</td>
<td></td>
</tr>
<tr>
<td>External Application</td>
<td>Mandatory</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Password is not used.</td>
<td>Password is not used.</td>
<td>Whether the password is used or not depends on the selected JAX-WS security policy.</td>
</tr>
</tbody>
</table>
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 25.2.6.1, "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 "Prerequisites to Configuring Content Server" section in Administering 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.
10. Click Test Connection to check whether you have entered the connection details correctly. You should see a Success! status.

During testing, the 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.

13. Enter the User Name and Password for the Content Server connection and click OK. The connection displays under the Application Resources panel.

---

**25.2.2 Creating 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 25.2.6, "About 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 Portal 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 25–2.
5. Select **Set as primary connection for Documents** to make this connection the primary (default) connection for all content management operations through the documents tool 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 task flow, that value overrides this setting.

**Note:** To be able to set a connection as primary (default) connection for documents, 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 25–3 and Figure 25–3.

   **Table 25–3** Parameters for Creating Oracle Portal-based Content Data Control

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Driver</td>
<td>There are two types of drivers: thin and oci8. The thin driver can be used to connect to Oracle Database release 8i 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.</td>
</tr>
</tbody>
</table>
Configuring Content Repository Connections

25-10 Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper

**Table 25–3 (Cont.) Parameters for Creating Oracle Portal-based Content Data Control**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host Name</td>
<td>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.</td>
</tr>
<tr>
<td>JDBC Port</td>
<td>Value to identify the TCP/IP port.</td>
</tr>
<tr>
<td>SID</td>
<td>Unique system identifier of an Oracle database instance.</td>
</tr>
<tr>
<td>Service Name</td>
<td>The service name for an Oracle database instance.</td>
</tr>
</tbody>
</table>

**Figure 25–3 Create Database Connection**

- Click **Test Connection** to check whether you have entered the connection details correctly. You should see a Success! status.
- 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.


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 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 26.2, "Configuring Content Data Controls for JCR Adapters".

### 25.2.3 Creating 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 25.2.6.3, "About Oracle WebCenter Adapter for SharePoint" before configuring and using Oracle WebCenter Adapter for SharePoint.

This section includes the following subsections:

- Section 25.2.3.1, "Installing the Oracle WebCenter Adapter for SharePoint"
- Section 25.2.3.2, "Creating a Content Repository Connection Based on the Oracle WebCenter Adapter for SharePoint"
- Section 25.2.3.3, "Mapping Microsoft SharePoint Content and Services"

#### 25.2.3.1 Installing the Oracle WebCenter Adapter for SharePoint

Before you configure the Oracle WebCenter Portal 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 the Oracle WebCenter Portal Companion DVD.
5. Click Finish. When prompted, restart JDeveloper.

25.2.3.2 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 Portal 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.
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, if you intend to make it the default connection for documents. Selecting this option writes a configuration to the adf-config.xml file that identifies this connection as the default connection for documents. If a Document Library task flow is used without any connectionName input parameter, then this connection is used. For information about documents, see Chapter 28, "Integrating Documents."

Note: To be able to set a connection as primary connection for documents, 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 25–4**.

**Figure 25–4  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 25.2.6.3, "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 25.2.6.1, "About Using Identity Propagation and External Application Authentication Methods."
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 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.

### 25.2.3.3 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 25–5.
For the Microsoft Office SharePoint Server (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 25–6. This path reflects the structure where the Documents Center is a sub-site 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.

### 25.2.4 Creating 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 development environments only.

To create a File System repository connection:

1. In Oracle JDeveloper, open your Portal 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** to make this connection the primary (default) connection for all content management operations through the documents tool 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 task flow, that value overrides this setting.

---

**Note:** To be able to set a connection as primary (default) connection for documents, 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 26.5, "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, for example, `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 25–7.
9. Click OK.

10. In the Application Resources panel, expand the repository connection you just created.

25.2.5 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 25–8. You can double-click folders and files to view them.

Figure 25–8 Application Resources
You can use repository connections to create JCR data controls that enable integration of the repository content with JSF pages. See Section 26.2, "Configuring Content Data Controls for JCR Adapters" and Section 26.5, "Integrating Content Using Content Data Controls" for information. These connections can also be consumed through the documents tool and Content Presenter task flows. See Chapter 28, "Integrating Documents" for information.

25.2.6 About Creating a Repository Connection

This section includes:

- Section 25.2.6.1, "About Using Identity Propagation and External Application Authentication Methods"
- Section 25.2.6.2, "About Oracle Portal"
- Section 25.2.6.3, "About Oracle WebCenter Adapter for SharePoint"

25.2.6.1 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 26.4, "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 74.13, "Working with External Applications" and Section 28.3, "Setting Security for the Documents Tool."

External applications allow 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 external applications, 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/
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.

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 documentation for more information on how to configure click through login.

g. Click Finish.

25.2.6.2 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 11.1.1.0.0 provided that the appropriate patches have been applied. Consult Oracle Application Server Release Notes for Microsoft Windows (for the required platform) for release 11.1.1.0.0 for the exact patches to be applied.

- 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.

25.2.6.3 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 fail 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 Portal 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 in 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 combination of these. Additional SharePoint-specific characteristics are in some instances added as mix-ins (that is, sp:Folder, sp:File, and so on). The SharePoint-specific namespace, 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 namespace. 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 that demonstrate this hierarchical mapping are described in the following list item.

- **Site Structure:** The structure mapping of a SharePoint site is as follows:
  - /sp:Site: The root element of the site.
  - /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 sub-site 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.

- **Search by Creator**: The value provided for the task flow should have values in the form `GBR10021\spmember` rather than `GBR10021\spmember`, as the latter will result in an IllegalState exception and the page will not load.

- **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.
25.3 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 25.3.1, "How to Edit a Common Repository Connection"
- Section 25.3.2, "Editing a Framework Application-Specific Content Repository Connection"

25.3.1 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 25–9. The Edit Repository Connection dialog displays, as shown in Figure 25–10.

* Properties for comparison: Size, Author, Modified, URL, Title, IsDocument, ContentClass, SiteName, Description, FileName, jcr:data, jcr:mimeType, jcr:created, jcr:lastModified.
2. Change the appropriate parameters.

3. Test the connection and click OK.

25.3.2 Editing a Framework Application-Specific Content Repository Connection

Application-specific content repository connections exist under the Application Resources panel.

To edit a Portal 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 25–11. The Edit Content Repository Connection dialog displays, as shown in Figure 25–12.
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 25.2.1, "Creating a Content Repository Connection Based on the Oracle Content Server Adapter."

3. Test the connection and click OK.

25.4 Using an Existing Repository Connection for a New Portal Framework Application

You can use an existing repository connection for any Portal 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 Portal 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 25–13.

   **Tip:** Alternatively, right-click the connection and choose Add to Application. The connection is displayed under the Application Resources panel.

   ![Figure 25–13 Dragging an Existing Common Repository Connection (IDE Connection) to an Application](image)

You can now configure a data control for your new application from this repository connection, as described in Section 26.2, "Configuring Content Data Controls for JCR Adapters".
This chapter describes how to use JCR data controls to access content such as documents and relational content.

The JCR API enables independent access to content, regardless of the underlying repository or the type of the content. 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 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. For more information, see Chapter 24, "Introduction to Integrating and Publishing Content."

This chapter includes the following sections:

- Section 26.1, "Overview of Content Data Controls"
- Section 26.2, "Configuring Content Data Controls for JCR Adapters"
- Section 26.3, "Editing Content Repository Data Controls"
- Section 26.4, "Securing a Content Repository Data Control"
- Section 26.5, "Integrating Content Using Content Data Controls"

### 26.1 Overview of Content Data Controls

A content data control is a container for all the data objects, collections, methods, and operations used to create UI components within your Portal 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, Content Server, and Oracle WebCenter Adapter for SharePoint:
26.1.1 **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 26–1 describes the parameters of the `getItems` method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>path</td>
<td>Defines the starting point for <code>getItems</code>.</td>
</tr>
<tr>
<td>type</td>
<td>Specifies what should be returned: only files, only folders, or any object.</td>
</tr>
</tbody>
</table>

The `getItems` method returns the attributes described in Table 26–2.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>icon16</td>
<td>Provides a URI to the icon that documents use for an item.</td>
</tr>
<tr>
<td>icon32</td>
<td>Provides a URI to the icon that documents use for an item.</td>
</tr>
<tr>
<td>ID</td>
<td>Parameter to document task flows that require an ID.</td>
</tr>
<tr>
<td>lastModified</td>
<td>Describes the last modified date of an item.</td>
</tr>
<tr>
<td>name</td>
<td>Describes the name of the returned file or folder.</td>
</tr>
<tr>
<td>path</td>
<td>Describes the location of the returned file or folder within the content repository.</td>
</tr>
<tr>
<td>URI</td>
<td>The direct access URL of a file or folder.</td>
</tr>
<tr>
<td>primaryType</td>
<td>Describes whether the returned object is a file or folder or some other type.</td>
</tr>
</tbody>
</table>

26.1.2 **search Method**

The `search` method enables you to create a simple search by name pattern or keyword. Table 26–3 describes the parameters of the `search` method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>path</td>
<td>Starting path of the search.</td>
</tr>
</tbody>
</table>
Overview of Content Data Controls

Working with Content Data Controls

The search method returns the attributes described in Table 26–4.

Table 26–4  Return Attributes of the search Method

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>icon16</td>
<td>Provides a URI to the icon that documents use for an item.</td>
</tr>
<tr>
<td>icon32</td>
<td>Provides a URI to the icon that documents use for an item.</td>
</tr>
<tr>
<td>ID</td>
<td>Parameter to the document task flows that require an ID.</td>
</tr>
<tr>
<td>lastModified</td>
<td>Describes the last modified date of an item.</td>
</tr>
<tr>
<td>name</td>
<td>Describes the name of the returned file or folder.</td>
</tr>
<tr>
<td>path</td>
<td>Describes the location of the returned file or folder within the content repository.</td>
</tr>
<tr>
<td>URI</td>
<td>The direct access URL of a file or folder.</td>
</tr>
<tr>
<td>primaryType</td>
<td>Describes whether the returned object is a file or folder or some other type.</td>
</tr>
</tbody>
</table>

Table 26–3  Parameters of the search Method (Cont.)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>isRecursive</td>
<td>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.</td>
</tr>
<tr>
<td>keyword</td>
<td>Search keyword for full text search.</td>
</tr>
<tr>
<td>namePattern</td>
<td>Pattern search on name. Use the % wildcard to match any number of characters and the _ wildcard to match one character.</td>
</tr>
</tbody>
</table>

26.1.3 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 26–5 describes the parameters of the advancedSearch method.

Table 26–5  Parameters of the advancedSearch Method

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>path</td>
<td>Starting path of the search.</td>
</tr>
<tr>
<td>isRecursive</td>
<td>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.</td>
</tr>
<tr>
<td>keyword</td>
<td>Search keyword for full text search.</td>
</tr>
<tr>
<td>namePattern</td>
<td>Pattern search on name. Use the % wildcard to match any number of characters and the _ wildcard to match one character.</td>
</tr>
<tr>
<td>matchAny</td>
<td>Specifies whether all predicates (=false) or any predicate (=true) should be matched.</td>
</tr>
<tr>
<td>predicates</td>
<td>A collection of SimplePredicate parameters that consist of attributes, comparators, and values.</td>
</tr>
</tbody>
</table>
### 26.1.4 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.

### 26.1.5 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 26–7 describes the attributes that the `getAttributes` method returns.

#### Table 26–7  Return Attributes of the getAttributes Method

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the attribute.</td>
</tr>
<tr>
<td>value</td>
<td>Value of the attribute.</td>
</tr>
</tbody>
</table>

### 26.2 Configuring Content Data Controls for JCR Adapters

This section describes how to create a content data control based on a content repository connection:

- Section 26.2.1, "Configuring a Content Repository Data Control"
- Section 26.2.2, "What Happens When You Configure a Content Repository Data Control"
26.2.1 Configuring a Content Repository Data Control

The procedure to create content repository data controls is the same regardless of the types of connections used to create data controls. If you have not created a connection to your content repository, then see Section 25.2, "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 25.2, "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
```
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.

### 26.2.2 What Happens When You Configure a Content Repository Data Control

Once the data control is successfully configured, it appears as a node in the Data Controls panel. You can expand the node to see the hierarchical list of methods, parameters, and operations for the new data control as shown in Figure 26–1. For more information on using the Data Control panel, see the "Using ADF Model in a Fusion Web Application" section in Fusion Developer’s Guide for Oracle Application Development Framework.

*Figure 26–1  Data Controls Panel - Content Server*

Table 26–8 describes the icons and object hierarchy of a data control.
The following files (Figure 26–2) 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.
26.3 Editing Content Repository Data Controls

This section describes a generic procedure to edit content data controls that you configured as described in Section 26.2, "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 26–3. The Edit Content Repository Data Control dialog displays, as shown in Figure 26–4.

---

- **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 26–2 Projects Panel - Portal Project*

*Figure 26–3 Edit Data Controls*
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. Refer to Section 25.2, "Configuring Content Repository Connections" for information about the configuration parameters specific to the connection type you intend to create.

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 25.2, "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.

26.4 Securing a Content Repository Data Control

You can enable security for your content repository connections. For information, see Chapter 74, "Securing Your WebCenter Portal Framework Application."

26.5 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 "Using ADF Model in a Fusion Web Application" chapter in 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 26.5.1, "Publishing Content As Links"
- Section 26.5.2, "What Happens at Runtime"
- Section 26.5.3, "Publishing Content in a Table"
- Section 26.5.4, "What Happens at Runtime"
- Section 26.5.5, "Publishing Folder Content in a Tree"
- Section 26.5.6, "What Happens at Runtime"
- Section 26.5.7, "Adding Search Capabilities to Content Repositories"
- Section 26.5.8, "What Happens at Runtime"
- Section 26.5.9, "Using Search Capabilities"

### 26.5.1 Publishing 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 26.5.1.1, "Publishing Content As a Textual Link"
- Section 26.5.1.2, "Creating a Clickable Image to Link to a Document"
Before you begin:

1. Configure a repository connection as described in Section 25.2, "Configuring Content Repository Connections."

2. Configure a data control as described in Section 26.2, "Configuring Content Data Controls for JCR Adapters."

3. Create a page as described in Section 15.2, "Creating Pages in a WebCenter Portal Framework Application."

26.5.1.1 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.jsp, 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 26–5.

Figure 26–5 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 26–6.

If this is the first time you have dropped a node onto the page, then the Edit Action Binding dialog displays.

Figure 26–6 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 26–7. 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.
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 26–8.

Figure 26–8 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 26–9.

Note: To grant edit, personalize, customize, and view permissions at the attribute level, see Section 26.4, "Securing a Content Repository Data Control".
Integrating Content Using Content Data Controls

Figure 26–9  ADF Go Link in a Browser

10. Click the link to check that the correct file is displayed.

In Section 26.5.1.2, “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 26–10.

Figure 26–10  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:

```xml
<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="getURI1Iterator"/>
</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:

```xml
<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:

```xml
<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 26–11.

Figure 26–11 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 26–12 shows the new link, **FOD Logo**, in addition to the **Plasma News** link that was added in the first part of Section 26.5.1.1, "Publishing Content As a Textual Link".

---

**Figure 26–12  New Textual Link**

You can add as many links as required by following these steps.

### 26.5.1.2 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 26.5.1.1, "Publishing Content As a Textual Link", in the Structure window, select **Insert inside af:goLink** and choose **Browse**. The Insert Item dialog displays.

3. From the **Select the item to be created** list, select **Image**, as shown in Figure 26–13.
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 26–14 shows a sample output.

**Figure 26–14  ADF Object Image Link in a Browser**

---

### 26.5.2 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 "Understanding the Fusion Page Lifecycle" chapter in *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.

### 26.5.3 Publishing 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 26.5.3.1, "Displaying Files and Folders in Read-Only Format"
- Section 26.5.3.2, "Displaying the Name Attribute As a Go Link"
- Section 26.5.3.3, "Configuring a Table to Show Only Files"

**Before you begin:**

1. Configure a repository connection as described in Section 25.2, "Configuring Content Repository Connections."
2. Configure a data control as described in Section 26.2, "Configuring Content Data Controls for JCR Adapters."
3. Create a page as described in Section 15.2, "Creating Pages in a WebCenter Portal Framework Application."

#### 26.5.3.1 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 26–15.
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 26–16.

4. In the Edit Table Columns dialog, as shown in Figure 26–17, 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 26–17  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 26–18. 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.
6. Click OK. You should now see a table on the page that looks like Figure 26–19.

Figure 26–19  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 26–20 shows a read-only table displaying both files and folders at runtime.
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).

### 26.5.3.2 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 26.5.3.1, "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 26–21), expand the first column of the table (af:column - Name) to show the default display format af:outputText - #{row.name}.

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="#{bindings.Return.hints.name.name}"`
3. In the Structure window, right-click af:column - URI 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 26–22. 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.

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 26–23.

Figure 26–22  Folder Content Displayed as Hyperlinks

Figure 26–23  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 26–24.

![Figure 26–24  Table Properties - Common Tab](image)

4. Run the page to view the output. Figure 26–25 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.
26.5.3.3 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.

---

In the next section, the Name column will be configured to show only files.
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 26–26, and click OK.

*Figure 26–26  Display Files Only*

4. Now run the page. Figure 26–27 shows only files in the single-column table because the column type is `nt:file`. 
26.5.4 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.

### 26.5.5 Publishing 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 26.5.5.1, "Displaying Files and Folders in Read-Only Format"
- Section 26.5.5.2, "Displaying File Names As Hyperlinks"

#### Before you begin:

1. Configure a repository connection as described in Section 25.2, "Configuring Content Repository Connections."
2. Configure a data control as described in Section 26.2, "Configuring Content Data Controls for JCR Adapters."
3. Create a page as described in Section 15.2, "Creating Pages in a WebCenter Portal Framework Application."

#### 26.5.5.1 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 26–28.
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 26–29.

If this is the first time you have dropped a node onto the page, the Edit Action Binding dialog displays.

Figure 26–29  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 26–30. 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.
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 26–31, which enables the tree to find its child items.
8. Click OK. A tree displays in the page that looks like Figure 26–32.

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 26–33 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 (namePageDef.xml).
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.

### 26.5.5.2 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 26–34.

### Figure 26–33  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.

#### 26.5.5.2 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 26–34.

### Figure 26–34  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 26–35.

*Figure 26–35  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 26–36 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.
26.5.6 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.

26.5.7 Adding 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
Integrating Content Using Content Data Controls

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 26.5.7.1, "Adding Simple Search Capabilities"
- **Section 26.5.7.2, "Adding Advanced Search Capabilities"

**Before you begin:**

1. Configure a repository connection as described in **Section 25.2, "Configuring Content Repository Connections."**
2. Configure a data control as described in **Section 26.2, "Configuring Content Data Controls for JCR Adapters."**
3. Create a page as described in **Section 15.2, "Creating Pages in a WebCenter Portal Framework Application."**

**26.5.7.1 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 26–37**.

**Figure 26–37  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 26–38** displays.
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 26–39.

26.5.7.2 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.

```java
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:

```java
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 26–40, and click OK.

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 26–41.

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 26–42.
26.5.8 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')]
```

### 26.5.9 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.
Creating Content Presenter Display Templates

This chapter describes the out-of-the-box Content Presenter display templates, how to create new display templates, and how to make display templates available to users.

Note: Content Presenter can only be used with Content Server-based content. No other content repository connection types are supported.

This chapter includes the following topics:

- Section 27.1, "About Content Presenter Display Templates"
- Section 27.2, "Using the Out-of-the-Box Display Templates"
- Section 27.3, "Creating Content Presenter Display Templates"
- Section 27.4, "Adding the Content Presenter Task Flow to a Page"
- Section 27.5, "Making Content Presenter Display Templates Available"
- Section 27.6, "What Happens at Runtime?"
- Section 27.7, "Configuring Coherence as the Caching Mechanism"
- Section 27.8, "Optimizing Performance for Content Presenter Display Templates"
- Section 27.9, "Content Presenter Tips, Tutorials and Examples"

27.1 About Content Presenter Display Templates

A Content Presenter display template is a JSFF file (JSF page fragment) that defines how Content Presenter renders content items (including images and text) on a portal or Framework application page. WebCenter Portal provides several out-of-the-box display templates to get you started, but you can also create your own templates to solve specific display requirements.

Some typical situations where Content Presenter display templates provide value are:

- Providing different layouts for different parts of a page. For example, you may have articles from different sources on the same page, each requiring its own layout (for more information, see Section 27.3.3, "Defining Multiple-Item Display Templates").

- Presenting content based on the capabilities of the viewing device. You can provide display templates that respond to whether the viewing device is a standard monitor, tablet or smart phone (for more information, see Section 27.3.4, "Using Responsive Templates").
Making content available through Content Presenter display templates lets you solve complex display issues through a standardized template rather than doing each individual layout by hand. By making display templates available to users you enable them to solve layout requirements based on predefined department or company standards without developer involvement.

**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 WebCenter Content repository. A content type defines the properties of the content item. Content types can map to WebCenter Content profile definitions and Site Studio region definitions.

**Tip:** Oracle recommends that you use Content Presenter ADF templates to integrate Site Studio and WebCenter Portal instead of Site Studio region templates. The recommended flow is:

- Develop region definitions in Site Studio
- Develop ADF templates referencing region definitions using JDeveloper
- Publish the templates and import them into Portal Server
- Use Content Presenter to render the content and to enable users to contribute content

Content types are created on the content repository (WebCenter Portal 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 WebCenter Content is to use the Content Presenter Configuration dialog in a portal. For a detailed description of this technique, see Section 27.3.8, "Discovering Content Type Property Names."

At runtime, an authorized end user can choose a display template in the Content Presenter Configuration dialog. For more information about best practices and determining when and how to use Content Presenter templates, see the blog entry at: http://www.ateam-oracle.com/portal-and-content-content-integration-best-practices
27.2 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, and path. Figure 27.4 shows a content item displayed at runtime with this template.

![Figure 27–1 Default Document Details View Display Template](image1)

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 27–2. (Note that one item will always be open in an accordion view. That is, you cannot close all of the items, you can only change which single item is open at any time.)

![Figure 27–2 Accordion View Display Template](image2)

For a complete list of out-of-the-box display templates, see Section 29.7.1, “Content Presenter Task Flow Parameters and Out-of-the-Box Display Templates.”
27.3 Creating Content Presenter Display Templates

If the out-of-the-box display templates (see Section 27.2, "Using the Out-of-the-Box Display Templates") do not meet your needs, you can define custom Content Presenter templates.

**Tip:** As well as creating new Content Presenter display templates, you can also copy one of the out-of-the-box templates and modify it.

This section discusses these topics:

- Section 27.3.1, "About Content Presenter Display Templates"
- Section 27.3.2, "Creating Single-Item Display Templates"
- Section 27.3.3, "Defining Multiple-Item Display Templates"
- Section 27.3.4, "Using Responsive Templates"
- Section 27.3.5, "Extending Responsive Templates"
- Section 27.3.6, "Using Image Renditions in Content Presenter Display Templates"
- Section 27.3.7, "Using EL Expressions to Retrieve Content Item Information"
- Section 27.3.8, "Discovering Content Type Property Names"
- Section 27.3.9, "Referencing External Files in Display Templates"
- Section 27.3.10, "Referencing Site Studio Region Elements in a Custom View"
- Section 27.3.11, "Using a Content Presenter Display Template"

27.3.1 About Content Presenter Display Templates

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 27.3.2, "Creating Single-Item Display Templates" and Section 27.3.3, "Defining Multiple-Item Display Templates."
- Deciding what kind of information you want to display with content items. Consider also, for example, using WebCenter Portal’s EL expressions in your template to retrieve and display this information. See Section 27.3.7, "Using EL Expressions to Retrieve Content Item Information," and Appendix G, "Expression Language Expressions" for more information about using EL expressions.
Creating Content Presenter Display Templates

- Designing the look and feel of the template. In addition to standard JSFF constructs, display templates can use ADF components. See Web User Interface Developer’s Guide for Oracle Application Development Framework.

- Exporting the template as a portal resource. See Section 27.5.1, "Export a Content Presenter Display Template as a Portal Resource."

- Configuring Content Presenter to use Coherence as the caching mechanism for production (optional, but recommended) and HA environments (required) as described in Section 27.7, "Configuring Coherence as the Caching Mechanism."

27.3.2 Creating 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 and using a different set of properties).

The definition of a single content item display template uses the JSP tags listed in Table 27–1.
<table>
<thead>
<tr>
<th>JSP Tag</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>contentTemplateDef</td>
<td>Required. Defines a single content item template.</td>
<td><code>&lt;dt:contentTemplateDef var=&quot;node&quot;&gt;&lt;af:outputText value=&quot;#{node.name}&quot; /&gt;&lt;/dt:contentTemplateDef&gt;</code></td>
</tr>
<tr>
<td></td>
<td><strong>Attributes:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>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 27.3.7.1, &quot;Retrieving Basic Information About a Content Item&quot; to retrieve required information about the node.</td>
<td></td>
</tr>
<tr>
<td>renderProperty</td>
<td>Optional. Retrieves and renders the string value(s) of the specified node property inline.</td>
<td><code>&lt;dt:contentTemplateDef var=&quot;node&quot;&gt;&lt;cmf:renderProperty id=&quot;rp1&quot; name=&quot;#{node.primaryProperty.name}&quot; node=&quot;#{node}&quot; /&gt;&lt;/dt:contentTemplateDef&gt;</code></td>
</tr>
<tr>
<td></td>
<td><strong>Attributes:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>id - Identifies this component. This value must be unique within the closest parent component that is a naming container.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>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.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>node - Specifies the node to use. This value can either be a bound attribute to a managed bean, or a request attribute variable.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>blockSize - Specifies the size of the blocks in bytes, to read the bytes of a binary property. Default is 2048 bytes.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>startIndex - Specifies the index (from 0) in the document's bytes at which to start printing. Defaults to 0.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>endIndex - Specifies the index (from 0) in the document's bytes at which to stop printing. Defaults to value of blockSize.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>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.</td>
<td></td>
</tr>
</tbody>
</table>
To create 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 27–3):
   - Select WebCenter Content Display Templates for a list of display template tags.
   - Select WebCenter Content Management Faces for the renderProperty tag.
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 27–1 for information about each tag and required parameter values.

Example 27–1 and Example 27–2 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 27–1 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 retrieve these property names for a given content type, see Section 27.3.8, "Discovering Content Type Property Names."

Example 27–1 Sample Content Presenter Display Template for a Press Release Document

```xml
<?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 27–2 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 27–2 Sample Content Presenter Template for Another Press Release Document

```xml
<?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">
```
Creating Content Presenter Display Templates

27.3.3 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 WebCenter Content folder.

A Content Presenter display template can also combine both single and multiple items. 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. For an example of how to do this, see the A-Team blog entry "Enable Content Editing of Iterative Components."

The definition of a multiple content item display template uses the JSP tags listed in Table 27–2.
### Table 27–2  Content Display Template Tags for Multiple Content Items

<table>
<thead>
<tr>
<th>JSP Tag</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>contentListTemplateDef</td>
<td>Required. Defines the multiple content item template.</td>
<td>&lt;dt:contentListTemplateDef var=&quot;nodes&quot;&gt;</td>
</tr>
<tr>
<td></td>
<td>Attributes:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>var - Specifies the content node that will be rendered by this display template.</td>
<td>&lt;af:iterator value=&quot;#{nodes}&quot; var=&quot;node&quot;&gt;</td>
</tr>
<tr>
<td></td>
<td>In the template definition code, you can use the EL expressions described in Section 27.3.7.1, &quot;Retrieving Basic Information About a Content Item&quot; to retrieve required information about the node.</td>
<td>&lt;af:outputText values=&quot;#{node.name}&quot; /&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td><a href="">dt:contentListTemplateDef</a></td>
</tr>
<tr>
<td>contentListTemplate</td>
<td>Optional. Nested under contentListTemplateDef.</td>
<td>&lt;dt:contentListTemplateDef var=&quot;nodes&quot;&gt;</td>
</tr>
<tr>
<td></td>
<td>Calls another multiple item template.</td>
<td>&lt;f:verbatim&gt;</td>
</tr>
<tr>
<td></td>
<td>Attributes:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>nodes - Required. Provides the list of VCR nodes that should be displayed</td>
<td>&lt;blockquote&gt;</td>
</tr>
<tr>
<td></td>
<td>category - Required. Specifies the target template category.</td>
<td>&lt;/blockquote&gt;</td>
</tr>
<tr>
<td></td>
<td>view - Required. Specifies the target view name.</td>
<td>&lt;f:verbatim&gt;</td>
</tr>
<tr>
<td></td>
<td>id - Optional. Specifies the JSF component ID.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>rendered - Optional. Specifies whether the component should be rendered.</td>
<td></td>
</tr>
<tr>
<td>contentTemplate</td>
<td>Optional. Nested under contentListTemplateDef.</td>
<td>&lt;dt:contentListTemplateDef var=&quot;nodes&quot;&gt;</td>
</tr>
<tr>
<td></td>
<td>Calls a single item template.</td>
<td>&lt;f:verbatim&gt;</td>
</tr>
<tr>
<td></td>
<td>Attributes:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>node - Required. Specifies the content repository node that should be displayed</td>
<td>&lt;dt:contentTemplate node=&quot;#{node}&quot;</td>
</tr>
<tr>
<td></td>
<td>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.</td>
<td>view='templates.pressrelease.listitem'</td>
</tr>
<tr>
<td></td>
<td>view - Optional. Specifies the target view name.</td>
<td>nodesHint='#{nodes}'/&gt;</td>
</tr>
<tr>
<td></td>
<td>id - Optional. Specifies the JSF component ID.</td>
<td>&lt;af:iterator rows=&quot;0&quot; var='node'</td>
</tr>
<tr>
<td></td>
<td>rendered - Optional. Specifies whether the component should be rendered.</td>
<td>varStatus='iterator' values='#{nodes}'/&gt;</td>
</tr>
</tbody>
</table>

Creating Content Presenter Display Templates  27-11
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 27.3.2, "Creating 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 27–2 for information about each tag and required parameter values.

Example 27–3 shows a sample multiple content item display template definition.

**Example 27–3 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
<?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}" nodesHint="#{nodes}" view="mycorp.content.templates.pressrelease.listitem"/>
      </af:iterator>
    </af:panelGroupLayout>
  </dt:contentListTemplateDef>
</jsp:root>
```

### 27.3.4 Using Responsive Templates

Using CSS3 media queries, you can render content to adapt to conditions such as screen resolution. The out-of-the-box Content Presenter templates "Articles View" and "Full Articles View" are examples of how CSS3 media queries can be used in a Content Presenter template for a responsive layout. For information about how these responsive templates can be extended, see Section 27.3.5, "Extending Responsive Templates." For information about Content Presenter's out-of-the-box display template parameters, see Section 29.7.1, "Content Presenter Task Flow Parameters and Out-of-the-Box Display Templates." For information about optimizing viewport settings for mobile devices such as smart phones and tablets, see the "Optimizing Portals for Mobile Devices" section in *Building Portals with Oracle WebCenter Portal*.

- Section 27.3.4.1, "Prerequisites"
- Section 27.3.4.2, "Displaying Multiple Articles"
- Section 27.3.4.3, "Using CSS3 Media Queries"

#### 27.3.4.1 Prerequisites

The Articles View and Full Article View templates described in Section 27.3.4.2, "Displaying Multiple Articles," rely on the Site Studio RD_ARTICLE region definition. Consequently, Site Studio must be enabled in WebCenter Content to allow the Articles View and Full Article View Content Presenter templates to be used.

Follow the steps below to enable Site Studio and seed Content Server with the RD_ARTICLE region definition:
1. Enable Site Studio (see the "Understanding Site Studio Integration" section in *Building Portals with Oracle WebCenter Portal*).

2. Start (or restart) WebCenter Portal after Site Studio has been enabled (this will seed the RD_ARTICLE region definition).

### 27.3.4.2 Displaying Multiple Articles

The template definition in Example 27–4 iterates over the data items selected for display, and uses three style classes:

- **article** - applied to each content item
- **article-3** - applied to each content item that will be in the first column if the page is split into rows of three columns
- **article-2** - applied to each content item that will be in the first column if the page is split into rows of two columns

The style classes are defined in the articles.css style sheet.

#### Example 27–4 Sample display template definition for displaying multiple articles (articles.jsff)

```xml
xmlns:dt="http://xmlns.oracle.com/webcenter/content/templates"
xmlns:f="http://java.sun.com/jsf/core"
xmlns:h="http://java.sun.com/jsf/html"
xmlns:rah="http://xmlns.oracle.com/webcenter/resourcehandler"
xmlns:tr="http://myfaces.apache.org/trinidad">
<dt:contentListTemplateDef var="nodes">
<af:iterator rows="0" var="node" varStatus="iterator" value="#{nodes}" id="it0">
<h:panelGroup styleClass="#{(iterator.index % 3 == 0) ? 'article-3' : ''}#{(iterator.index % 2 == 0) ? 'article-2' : ''}article" id="pg1">
<af:commandLink immediate="true" partialSubmit="true" id="cl2">
<rah:resourceActionBehavior id="rah1" serviceId="oracle.webcenter.content.presenter" resourceId="#{node.id}" resourceTitle="#{node.propertyMap['RD_ARTICLE:TITLE'].asTextHtml}" useResourcePopup="never" />
<f:attribute name="taskFlowInstId" value="a5fafa8-90e6-4972-997d-314401b6c98b"/>
<f:attribute name="datasourceType" value="dsTypeSingleNode"/>
<f:attribute name="datasource" value="#{node.id.repositoryName}#dDocName:#{node.propertyMap['dDocName'].value}"/>
<f:attribute name="templateView" value="oracle.webcenter.content.templates.sitestudio.fullarticle"/>
<af:outputText value="#{node.propertyMap['RD_ARTICLE:IMAGE'].asTextHtml}" escape="false" id="ot4"/>
<tr:panelHeader text="#{node.propertyMap['RD_"```
27.3.4.3 Using CSS3 Media Queries

The style sheet in Example 27–5 uses media queries to render the content in different ways depending on the width of the browser.

Example 27–5  Style sheet using CSS3 Media Queries (articles.css)

```css
/* Default styles */
.article {
  display: block;
  float: left;
  width: 31.623931623931625%;
  margin-left: 2.564102564102564%;
  margin-bottom: 1em;
}
.article-3 {
  margin-left: 0;
  clear: both;
}
.article img {
  width: 100%;
  margin: 0 0 .25em;
  float: none;
  padding-top: 0;
  display: block;
  border: 0;
}
.article h1 {
  font-size: 1.5em;
}

@media only screen and (max-width : 480px) {
  /* up to width of iPhone (excluding iPhone 5 in landscape) */
  .article {
    margin-top: .5em;
    float: left;
    display: inline;
    width: 100%;
    margin-left: 0;
  }
  .article-3 {
    width: 100%;
    clear: none;
  }
  .article img {
    margin-right: 4%;
    width: 48%;
}
```
27.3.5 Extending Responsive Templates

These section describes the steps required to modify the out-of-the-box templates if they don’t meet your local requirements. For information about optimizing viewport settings for mobile devices such as smart phones and tablets, see the "Optimizing Portals for Mobile Devices" section in Building Portals with Oracle WebCenter Portal.

This section includes the following subsections:

- Section 27.3.5.1, "Extending the Articles View Template"
- Section 27.3.5.2, "Extending the Full Article View Template"
- Section 27.3.5.3, "Adapting the Out-of-the-Box Templates"

27.3.5.1 Extending the Articles View Template

To extend the Articles View template follow the steps below:

1. Create an empty template for displaying multiple items (see Section 27.3.3, "Defining Multiple-Item Display Templates") and copy the contents of
articles.jsff (located in the JDEVELOPER/webcenter/samples/contentpresenter/ directory) into the empty template.

2. Embed the CSS used by articles.jsff into the new template by changing:

```xml
<af:resource type="css" source="/oracle/webcenter/content/templates/seeded/articles.css"/>
```

To:

```xml
<af:resource type="css">
/* Default styles */
.article {  
display: block;
float: left;
width: 31.623931623931625%;
margin-left: 2.564102564102564%;
margin-bottom: 1em;
}
.article-3 {  
marg
3. Edit the new template to adapt it to your requirements (see Section 27.3.5.3, "Adapting the Out-of-the-Box Templates").

4. Export the new template as a Portal Resource (see Section 27.5.1, "Export a Content Presenter Display Template as a Portal Resource").

5. Upload the new template to your WebCenter Portal or Framework application (see Section 27.5.2, "Upload the New Content Presenter Display Template").

27.3.5.2 Extending the Full Article View Template

Follow the steps below to extend the Full Article View template:

1. Create an empty template for displaying a single content item (see Section 27.3.2, "Creating Single-Item Display Templates") and copy the contents of full-article.jsff (located in the JDEVELOPER/webcenter/samples/contentpresenter/ directory) into the empty template.

2. Embed the CSS used by full-article.jsff into the new template by changing:

   `<af:resource type="css" source="/oracle/webcenter/content/templates/seeded/articles.css"/>`

   To:

   `<af:resource type="css" />
   /* Default styles */
   .full-article h1 {
       font-size: 1.5em;
3. Edit the new template to adapt it your requirements (see Section 27.3.5.3, "Adapting the Out-of-the-Box Templates").

4. Export the new template as a Portal Resource (see Section 27.5.1, "Export a Content Presenter Display Template as a Portal Resource").

5. Upload the new template (see Section 27.5.2, "Upload the New Content Presenter Display Template").

### 27.3.5.3 Adapting the Out-of-the-Box Templates

This section describes changes you may want to make to the Articles View and Full Article View templates, and outlines how you could go about making those changes. For example:

- You may want to update the templates to support a responsive layout on older browsers like Internet Explorer 8.
- You may want to update the templates to work with a different region definition.

These extensions are described in the following sections:

- Section 27.3.5.3.1, "Supporting Responsive Layouts for Older Browsers"
- Section 27.3.5.3.2, "Using Different Region Definitions"
- Section 27.3.5.3.3, "Updating the Layout in a Template"

#### 27.3.5.3.1 Supporting Responsive Layouts for Older Browsers

The current responsive templates rely on CSS3 media queries, which are not supported by older browsers, so you may want to use a third-party JavaScript library that can take the media queries and use JavaScript to enable a responsive design.
To do this:

- Create and deploy a custom Shared Library for WebCenter Portal containing the third-party JavaScript library (see Section 55.2, "Developing Task Flows, Data Controls, and Managed Beans for WebCenter Portal").
- Add a reference to a third-party JavaScript library to the template (where path/to/3rd/party/library.js is the path to the JavaScript file within the Custom Shared Library):

```xml
<af:resource type="javascript" source="path/to/3rd/party/library.js"/>
```

### 27.3.5.3.2 Using Different Region Definitions

This section shows you how you can modify the region definitions in your responsive template. For example, you might have a region definition called RD_NEWS with four elements: TITLE, LEAD, IMAGE, and BODY (see Section 27.3.10, "Referencing Site Studio Region Elements in a Custom View"). You could then update the templates to reference this new region definition by changing references of RD_ARTICLES to RD_NEWS and changing the references of SUMMARY to LEAD.

**News View**

This template displays a list of news items (of type RD_NEWS) based on the Articles View template:

```xml
<?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"
   xmlns:f="http://java.sun.com/jsf/core"
   xmlns:h="http://java.sun.com/jsf/html"
   xmlns:rah="http://xmlns.oracle.com/webcenter/resourcehandler"
   xmlns:tr="http://myfaces.apache.org/trinidad">
   <dt:contentListTemplateDef var="nodes">
       <af:resource type="css" source="/oracle/webcenter/content/templates/seeded/articles.css"/>
       <af:iterator rows="0" var="node" varStatus="iterator" value="#{nodes}" id="it0">
           <h:panelGroup styleClass="#{(iterator.index % 3 == 0) ? 'article-3 ' : ''}#{(iterator.index % 2 == 0) ? 'article-2 ' : ''}article" id="pg1">
               <af:commandLink immediate="true" partialSubmit="true" id="cl2">
                   <rah:resourceActionBehavior id="rah1" serviceId="oracle.webcenter.content.presenter" resourceId="#{node.id}" resourceTitle="#{node.propertyMap['RD_NEWS:TITLE'].asTextHtml}" useResourcePopup="never"/>
                   <f:attribute name="taskFlowInstId" value="a5fafea8-90e6-4972-997d-314401b6c98b"/>
                   <f:attribute name="datasourceType" value="dsTypeSingleNode"/>
                   <f:attribute name="datasource" value="#{node.id.repositoryName}#dDocName:#{node.propertyMap['dDocName'].value}"/>
                   <f:attribute name="templateView" value="oracle.webcenter.content.templates.newsitem"/>
                   <f:attribute name="regionTemplate" value="#{false}"/>
                   <af:outputText value="#{node.propertyMap['RD_NEWS:IMAGE'].asTextHtml}"/>
               </af:commandLink>
           </h:panelGroup>
       </af:iterator>
   </dt:contentListTemplateDef>
</jsp:root>
```
News Item View
This template displays a full news item (View ID: oracle.webcenter.content.templates.newsitem) based on the Full Articles View template:

```xml
<?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"
    xmlns:f="http://java.sun.com/jsf/core"
    xmlns:h="http://java.sun.com/jsf/html"
    xmlns:tr="http://myfaces.apache.org/trinidad">
    <dt:contentTemplateDef var="node">
        <af:resource type="css" source="/oracle/webcenter/content/templates/seeded/articles.css"/>
        <af:panelGroupLayout id="ps11" layout="vertical">
            <tr:panelGroupLayout>
                <af:outputText value="#{node.propertyMap['RD_NEWS:IMAGE'].asTextHtml}" escape="false" id="ot4"/>
            </tr:panelGroupLayout>
            <tr:panelHeader text="#{node.propertyMap['RD_NEWS:TITLE'].asTextHtml}" id="ph1">
                <af:outputText value="#{node.propertyMap['RD_NEWS:LEAD'].asTextHtml}" escape="false" id="ot3"/>
            </tr:panelHeader>
            <af:commandLink>
                <h:panelGroup>
                    <af:iterator>
                        <dt:contentListTemplateDef>
                    </jsp:root>
```
Creating Content Presenter Display Templates

Figure 27–4  Article with Wide Layout

You could change this so that the text doesn’t flow under the image:

Figure 27–5  Article with Narrow Layout
To do this, you could create a new template based on the Articles View template, add a style class to the block of text (`article-text`), and set the display CSS property for this style class to `table-cell` for the narrow layout:

```xml
<?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"
  xmlns:f="http://java.sun.com/jsf/core"
  xmlns:h="http://java.sun.com/jsf/html"
  xmlns:rah="http://xmlns.oracle.com/webcenter/resourcehandler"
  xmlns:tr="http://myfaces.apache.org/trinidad">
  <dt:contentListTemplateDef var="nodes">
    <af:resource type="css">
      /* Default styles */
      .article {
        display: block;
        float: left;
        width: 31.623931623931625%;
        margin-left: 2.564102564102564%;
        margin-bottom: 1em;
      }
      .article-3 {
        margin-left: 0;
        clear: both;
      }
      .article img {
        width: 100%;
        margin: 0 0 .25em;
        float: none;
        padding-top: 0;
        display: block;
        border: 0;
      }
      .article h1 {
        font-size: 1.5em;
      }
    </af:resource>
    @media only screen and (max-width : 480px) {
      /* up to width of iPhone */
      .article {
        margin-top: .5em;
        float: left;
        display: inline;
        width: 100%;
        margin-left: 0;
      }
      .article-3 {
        width: 100%;
        clear: none;
      }
      .article img {
        margin-right: 4%;
        width: 48%;
        float: left;
      }
      .article {
        font-size: 1em;
      }
      .article h1 {
```
Creating Content Presenter Display Templates

/* Set the display CSS property to table-cell for the article-text style class in the narrow layout */
.article-text {
  display: table-cell;
}

@media only screen and (min-width : 481px) and (max-width : 780px) {
  /* up to the width of iPad in portrait */
  .article {
    display: block;
    float: left;
    width: 48.71794871794871%;
    margin-left: 2.564102564102564%;
    clear: none;
  }
  .article-3 {
    margin-left: 2.564102564102564%;
    clear: none;
  }
  .article-2 {
    margin-left: 0;
    clear: both;
  }
  .article h1 {
    font-size: 1.25em;
  }
}

@media only screen and (min-width : 769px) and (max-width : 1024px) {
  /* up to the width of iPad in landscape */
}

@media only screen and (min-width : 1025px) {
  /* desktop */
}
</af:resource><af:iterator rows="0" var="node" varStatus="iterator" value="#{nodes}" id="it0">
<h:panelGroup styleClass="#{(iterator.index % 3 == 0) ? 'article-3 ' :
''}#{(iterator.index % 2 == 0) ? 'article-2 ' : ''}article" id="pg1">
<af:commandLink immediate="true" partialSubmit="true" id="cl2">
<rah:resourceActionBehavior id="rah1"
    serviceId="oracle.webcenter.content.presenter"
    resourceId="#{node.id}" resourceTitle="#{node.propertyMap['RD_ARTICLE:TITLE'].asTextHtml}"
    useResourcePopup="never"/>
<attribute name="taskFlowInstId" value="a5fafe8-90e6-4972-997d-314401b6c98b"/>
<attribute name="datasourceType" value="dsTypeSingleNode"/>
<attribute name="datasource" value="#{node.id.repositoryName}#dDocName:#{node.propertyMap['dDocName'].value}"/>
</af:commandLink>
</h:panelGroup>
</af:iterator>
27.3.6 Using Image Renditions in Content Presenter Display Templates

In some cases you may want to use different renditions of an image in different circumstances. For example, you might want to use a large, high resolution image when the page containing the image is displayed using a desktop browser. However, if the same page is displayed on a mobile device, a large, high resolution image might not be appropriate given the smaller screen size and possible slower download speed. For mobile devices it would be better to display a smaller, lower resolution version, or rendition, of the image. Content Presenter display templates provide the capability of specifying which rendition of an image to display under which circumstances.

To enable the use of different image renditions, you can use EL expressions in your Content Presenter display templates to determine which image renditions to use when.

---

**Note:** Image renditions are not supported through CMIS.

---

This section includes the following topics:

- Section 27.3.6.1, "Prerequisites"
- Section 27.3.6.2, "Retrieving Image Rendition Information for Image Documents"
- Section 27.3.6.3, "Retrieving Image Renditions for Site Studio Region Elements"

### 27.3.6.1 Prerequisites

For full image rendition support, the WebCenter Content where your images are checked in must have Digital Asset Management (DAM) enabled. If DAM is not enabled, there is limited support for separate web and thumbnail renditions only. For information about enabling DAM, see the "Enabling Digital Asset Manager" section in *Administering Oracle WebCenter Portal*.

When DAM is enabled, different renditions are automatically created when an image is checked in, determined by the rendition set specified during check in. DAM provides some built-in rendition sets but the WebCenter Content administrator can also create new rendition sets. The individual renditions can then be referenced by name in Content Presenter display templates by using the appropriate EL expression.
For more information about DAM and rendition sets, see the "Working with Image and Video Conversions" chapter of Managing Oracle WebCenter Content.

Note: WebCenter Portal supports multiple renditions for images only, not video.

In addition, the following other prerequisites must also be met:

- You must be using WebCenter Content Release 11gR1 (11.1.1.9.0) or later.
- WebCenter Portal and WebCenter Content must be using the same OHS front-end.
- The webContextRoot parameter must be set in line with the common OHS front-end so that WebCenter Portal can find WebCenter Content objects, for example, /cs.
- Single sign-on must be enabled across WebCenter Portal and WebCenter Content.

27.3.6.2 Retrieving Image Rendition Information for Image Documents

To retrieve image rendition information for image documents in your Content Presenter display template, use the following EL expression:

node.renditionsMap['renditionName:renditionProperty']

Where:

renditionName is:

- for DAM implementations, the name of the rendition from the appropriate rendition set. For example, web, thumbnail, preview, or lowres.
- for non-DAM implementations, either web or thumbnail.

Note: The renditionName is not case sensitive, therefore preview refers to the same rendition as Preview.

renditionProperty is the required rendition information.

In most cases, you will want to retrieve the URL of the image rendition so that the rendition can be displayed on a page. To do this use the url property.

You can also retrieve other information about the rendition, such as name, type, width, height, resolution, size, or contentType.

Note: For web and thumbnail renditions, only size, contentType, and url are applicable; name returns the name of the native file, not of the web or thumbnail rendition.

For example to display the preview rendition of an image, add the following to your template:

<af:image source="${node.renditionsMap['preview:url']}" shortDesc="Preview rendition" id="imageContent3"/>
Example 27–6 shows a Content Presenter display template that displays a carousel of images. If the carousel is displayed on a desktop device (rendered="#{DeviceAgent.desktop}"), then the a200 rendition is used for the images (node.renditionsMap['a200:url']). If the carousel is displayed on a mobile device (rendered="#{DeviceAgent.mobile}"), then the smaller thumbnail rendition is used (node.renditionsMap['thumbnail:url']).

**Example 27–6  Content Presenter Template Using Image Renditions**

```xml
<?xml version='1.0' encoding='utf-8'?>
<af:carousel id="ci1"
  value="#{nodes}'
  var='node'
  emptyText='#{templateBundle.EMPTY_NODES}'
  inlineStyle='#{DeviceAgent.desktop ? '' : 'height:200px;'}'>
  <f:facet name='nodeStamp'>
    <af:carouselItem id="ci1"
      text='#{node.isFolder ? node.name : (empty
        node.propertyMap['dDocTitle'] ?
      node.name :
      node.propertyMap['dDocTitle'].value.stringValue)}'
      shortDesc='#{not empty
      node.propertyMap['xComments'].value.stringValue ?
      node.propertyMap['xComments'].value.stringValue :
      node.primaryProperty.value.binaryValue.name}'>
      <af:image id='cimg1'
        source='#{node.primaryProperty.isImage ?
        node.icon.largeIcon}'
        shortDesc='#{node.primaryProperty.value.binaryValue.name}'
        rendered='#{DeviceAgent.desktop}'/> 
      <af:image id='cimg2'
        source='#{node.primaryProperty.isImage ?
        node.icon.largeIcon}'
        shortDesc='#{node.primaryProperty.value.binaryValue.name}'
        rendered='#{DeviceAgent.mobile}'/> 
    </af:carouselItem>
  </f:facet>
</af:carousel>
<dt:contentListTemplateDef var='nodes'>
</jsp:root>
```

**Note:** Wherever possible, url points to a static rendition URL for the image rendition on the WebCenter Content. If the static URL cannot be determined, the url uses the WebCenter Portal showProperty servlet.
27.3.6.3 Retrieving Image Renditions for Site Studio Region Elements

The handling of image renditions for images in Site Studio region definitions is determined by how the region definitions are defined in Site Studio. You can either have a region definition that includes a single region element that defines the original image or a region definition that includes separate region elements for each image rendition. Your Content Presenter display template must use the appropriate EL expression according to how the region definition is set up.

For example, your region definition could have a single region element (myimage) for the original image, therefore to display a particular image rendition, you must use an EL expression where you can specify which rendition to retrieve from the WebCenter Content, for example, highres or lowres. Alternatively, if your region definition has separate region elements for the different image renditions, say one element (desktop) for a rendition suitable for desktop browsers and one (mobile) for a rendition more suited to mobile devices, you can use an EL expression that refers directly to the region element that defines the rendition that you want to use.

Tip: If you anticipate adding further image renditions in the future, for example for new devices, it makes sense to create region definitions with single image region elements so that as new renditions are added, they can be included in your Content Presenter display templates without having to create new region elements.

- If the region definition includes a single region element that defines the original image, you can also specify which rendition of the image you want to use:

```text
node.propertyMap['regionDefinitionName:elementName/Rendition:renditionName'].asTextHtml
```

- To reference a region element within a region definition that defines a specific image rendition, use the following EL expression:

```text
node.propertyMap['regionDefinitionName:elementName'].asTextHtml
```

Where:

- `regionDefinitionName` is the name of the region definition.
- `elementName` is the name of the region element.
- `renditionName` is:
  - for DAM implementations, the name of the rendition from the appropriate rendition set. For example, web, thumbnail, highres or lowres.
  - for non-DAM implementations, either web or thumbnail.

Note: The `renditionName` is not case sensitive, therefore preview refers to the same rendition as Preview.

The EL expressions return an HTML `img` tag with a `src` element that points to the URL of the appropriate image rendition.

Note: Wherever possible, the `src` URL points to a static rendition URL for the image rendition on the WebCenter Content. If the static URL cannot be determined, the `url` uses the WebCenter Portal `showProperty servlet`.
Figure 27–6 shows a Site Studio region definition (RD_REUSEIMG) for an article. The definition includes region elements for the title, summary, body and image of the article. The single image region element defines the original image (Image).

**Figure 27–6  Region Definition with Single Image Region Element**

To use a lower resolution rendition (lowres) of the image on a mobile device, use the following code:

```
<af:outputText value='#(node.propertyMap['RD_REUSEIMG:Image/Rendition:lowres'].asTextHtml)' escape='false' id='olmaget' rendered='#{DeviceAgent.mobile}'/>
```

**Note:** If the specified renditionName does not exist for the image, the image's primary rendition is used instead.

Figure 27–7 shows a different region definition (RD_NAMEDREN) for a different article that includes separate region elements for the different possible renditions of the article image: BaseImage, ThumbnailImage, A100Image, and A200Image.
Creating Content Presenter Display Templates

To use the ThumbnailImage region element to render the image on a mobile device, use the following code:

```xml
<af:outputText value="#{node.propertyMap['RD_NAMEDREND:ThumbnailImage'].asTextHtml}" escape="false" id="olmaget" rendered="#{DeviceAgent.mobile}"/>
```

Within a WYSIWYG or static list region element you can reference a specific rendition of an image using the `node.propertyMap['regionDefinitionName:elementName']` EL expression. Alternatively, you can reference original images in the WYSIWYG or static list region elements and then use the `node.propertyMap['regionDefinitionName:elementName/Rendition:renditionName']` EL expression to use a specific rendition for all images within the element.

Example 27–7 shows how a particular rendition (Preview) is used for images within a static list region element (paragraph).
Example 27–7  Using Image Renditions within a Static List Region Element

```af:iterator value="#{node.propertyMap['RD_MYREGION:paragraph/Rendition:Preview'].values}" var="row" 
    id="il"
    rendered="#{DeviceAgent.desktop}"
>   <af:showDetailItem text="#{row.nestedValue[0].value}" id="sdi1" stretchChildren="first">
>       <af:panelGroupLayout layout="horizontal" id='tt' inlineStyle="padding:5px;">
>           <af:outputText escape="false" value="#{row.nestedValue[1].value}" id="ld1"/>
>           <af:outputText escape="false" value="#{row.nestedValue[2].value}" id="ld2"/>
>       </af:panelGroupLayout>
>   </af:showDetailItem>
</af:iterator>
```

In Example 27–8 RD_MYREGION:body corresponds to a WYSIWYG region element. All images within that WYSIWYG element will be rendered using the A100 rendition.

Example 27–8  Using Image Renditions within a WYSIWYG Region Element

```af:outputText value="#{node.propertyMap['RD_MYREGION:body/Rendition:A100'].asTextHtml}" escape="false" id='ot5'/>
```

**Note:** If the WYSIWYG or static list element references an element that defines a particular image rendition, that rendition supersedes any rendition specified using the Rendition:renditionName syntax.

### 27.3.7  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 27.3.2, "Creating Single-Item Display Templates" and Section 27.3.3, "Defining Multiple-Item Display Templates.” These expressions are used with the JSP tags described in Table 27–1 and Table 27–2.

This section contains the following subsections:

- Section 27.3.7.1, "Retrieving Basic Information About a Content Item"
- Section 27.3.7.2, "Working with Content Item Properties and Values"
- Section 27.3.7.3, "Working with Content Item Icons and URLs"
- Section 27.3.7.4, "Working with Image Renditions"
- Section 27.3.7.5, "Working with Group Portal Information"

#### 27.3.7.1  Retrieving Basic Information About a Content Item

The EL expressions listed in Table 27–3 let you display basic information about a content item in a display template.
Creating Content Presenter Display Templates

### 27.3.7.2 Working with Content Item Properties and Values

Use the EL expression described in Table 27–4 and Table 27–5 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 27.3.8, "Discovering Content Type Property Names."

#### Table 27–3  EL Expressions for Retrieving Basic Content Information

<table>
<thead>
<tr>
<th>EL Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{node.createdBy}</td>
<td>Returns the user name of the node’s creator.</td>
</tr>
<tr>
<td>#{node.createdDate}</td>
<td>Returns the node’s creation date.</td>
</tr>
<tr>
<td>#{node.hasParentNode}</td>
<td>Returns true if the current node has a valid parent node ID, or a node that has no parent.</td>
</tr>
<tr>
<td>#{node.icon}</td>
<td>Returns the icon service defined in the current web application.</td>
</tr>
<tr>
<td>#{node.id}</td>
<td>Returns the node’s identifier.</td>
</tr>
<tr>
<td>#{node.isFolder}</td>
<td>Returns true if this node is associated with a folder or container.</td>
</tr>
<tr>
<td>#{node.isInherited}</td>
<td>Returns true if this node is inherited by another object class definition.</td>
</tr>
<tr>
<td>#{node.modifiedBy}</td>
<td>Returns the user name of the node’s last modifier.</td>
</tr>
<tr>
<td>#{node.modifiedDate}</td>
<td>Returns the node’s last modification date.</td>
</tr>
<tr>
<td>#{node.name}</td>
<td>Returns the node’s name.</td>
</tr>
<tr>
<td>#{node.parentId}</td>
<td>Returns the parent node’s identifier.</td>
</tr>
<tr>
<td>#{node.path}</td>
<td>Returns the node’s path.</td>
</tr>
<tr>
<td>#{node.primaryProperty}</td>
<td>Returns the node’s primary property, if available.</td>
</tr>
<tr>
<td>#{node.propertyMap}</td>
<td>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}.</td>
</tr>
<tr>
<td>#{node.url}</td>
<td>Returns an instance of the node property URL service for the primary property of this node (if any). By default, it resolves to #{node.url.renderUrl}. This is a shortcut for #{node.primaryProperty.url}.</td>
</tr>
</tbody>
</table>

#### Table 27–4  EL Expressions for Content Item Node Properties

<table>
<thead>
<tr>
<th>EL Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{node.propertyMap['myProp'].asTextHtml}</td>
<td>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.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].hasValue}</td>
<td>Returns true if a value is associated with this property.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].icon}</td>
<td>Returns the icon service defined in the current web application.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].indexedName}</td>
<td>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}</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].isAudio}</td>
<td>Returns true if the property’s mime type is ’audio/’.</td>
</tr>
<tr>
<td>EL Expression</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].isBinary}</td>
<td>Returns true if the current property is of type Property.BINARY.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].isBoolean}</td>
<td>Returns true if the current property is of type Property.BOOLEAN.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].isCalendar}</td>
<td>Returns true if the current property is of type Property.CALENDAR.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].isDouble}</td>
<td>Returns true if the current property is of type Property.DOUBLE.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].isExcel}</td>
<td>Returns true if the property's mime type is 'application/vnd.ms-excel', 'application/excel', 'application/x-excel', or 'application/x-msexcel'.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].isGIF}</td>
<td>Returns true if the property's mime type is 'image/gif'.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].isHTML}</td>
<td>Returns true if the property's mime type is 'text/html'.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].isImage}</td>
<td>Returns true if the property's mime type is 'image/'.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].isJPEG}</td>
<td>Returns true if the property's mime type is 'image/jpeg'.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].isLink}</td>
<td>Returns true if the current property is of type Property.LINK.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].isLong}</td>
<td>Returns true if the current property is of type Property.LONG.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].isMSWord}</td>
<td>Returns true if the property's mime type is 'application/vnd.ms-word' or 'application/msword'.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].isMultiValued}</td>
<td>Returns true if this property is multi-valued.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].isNested}</td>
<td>Returns true if the current property is of type Property.NESTED.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].isPDF}</td>
<td>Returns true if the property's mime type is 'application/pdf'.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].isPlainText}</td>
<td>Returns true if the property's mime type is 'text/plain'.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].isPNG}</td>
<td>Returns true if the property's mime type is 'image/png'.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].isPowerPoint}</td>
<td>Returns true if the property's mime type is 'application/vnd.ms-powerpoint', 'application/mspowerpoint', or 'application/x-mspowerpoint'.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].isPrimaryProperty}</td>
<td>Returns true if this property is the primary property.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].isRequired}</td>
<td>Returns true if this property is required/mandatory.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].isRestricted}</td>
<td>Returns true if this property is restricted.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].isRichText}</td>
<td>Returns true if the property's mime type is 'text/richtext'.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].isString}</td>
<td>Returns true if the current property is of type Property.STRING.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].isTextBased}</td>
<td>Returns true if this property is text-based (isHTML, isPlainText, isString, isCalendar, isBoolean, isDouble, isLong).</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].isVideo}</td>
<td>Returns true if the property's mime type is 'video/'.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].isXML}</td>
<td>Returns true if the property's mime type is 'text/xml'.</td>
</tr>
</tbody>
</table>
### Table 27-4 (Cont.) EL Expressions for Content Item Node Properties

<table>
<thead>
<tr>
<th>EL Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{node.propertyMap['myProp'].isZip}</td>
<td>Returns true if the property's mime type is 'application/zip'.</td>
</tr>
<tr>
<td>#{node.propertyMap['myRefNode'].linkAsNode}</td>
<td>Returns the myRefNode property as a node, where myRefNode is a link property type. Properties can be referenced in EL expressions.</td>
</tr>
<tr>
<td><strong>Example:</strong> #node.propertyMap['myRefNode'].linkAsNode.primaryProperty.url.renderUrl</td>
<td></td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].name}</td>
<td>Returns the property's name.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].nestedProperty}</td>
<td>Retrieves nested properties for this single-valued property, and returns a list of properties.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].type}</td>
<td>Returns the data type of this property value. For example: String, Integer, Long, and so on.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].url}</td>
<td>Returns a URL service for this property.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].value}</td>
<td>Returns the value service for this property.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].nestedProperties}</td>
<td>Returns elements from a static list. For example:</td>
</tr>
<tr>
<td></td>
<td>&lt;af:iterator var=&quot;listItem&quot; value=&quot;#{node.propertyMap['ARTICLE_RGD:Paragraphs'].nestedProperties}&quot; varStatus=&quot;vs&quot;&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;af:outputText id=&quot;ot1&quot; value=&quot;#{listItem[0].value}&quot;/&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;af:outputText id=&quot;ot3&quot; value=&quot;#{listItem[1].value}&quot;/&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/af:iterator&gt;</td>
</tr>
<tr>
<td>#node.propertyMap['regionDefName:elementName'].asTextHtml</td>
<td>Returns Site Studio data as HTML text. For example:</td>
</tr>
<tr>
<td></td>
<td>#node.propertyMap['RD_NEWS:LEAD'].asTextHtml</td>
</tr>
<tr>
<td></td>
<td>The element name (LEAD) is prefixed by the Region Definition name (RD_NEWS). See also Section 27.3.10, &quot;Referencing Site Studio Region Elements in a Custom View.&quot;</td>
</tr>
</tbody>
</table>

### Table 27-5 EL Expressions for Content Item Node Property Values

<table>
<thead>
<tr>
<th>EL Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{node.propertyMap['myProp'].value.binaryValue}</td>
<td>Returns custom attributes for a binary property type or attachment. Attributes available on binaryValue are:</td>
</tr>
<tr>
<td></td>
<td>■ contentType – Returns the mime type of the binary content (for example, text or html).</td>
</tr>
<tr>
<td></td>
<td>■ name – Returns the filename of the binary content (for example, index.html).</td>
</tr>
<tr>
<td></td>
<td>■ size – Returns the size of the binary content, or -1 if the size is unavailable.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].value.booleanValue}</td>
<td>Returns the value of this property as java.lang.Boolean.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].value.calendarValue}</td>
<td>Returns the value of this property as java.util.Calendar.</td>
</tr>
<tr>
<td>#{node.propertyMap['myProp'].value.doubleValue}</td>
<td>Returns the value of this property as java.lang.Double.</td>
</tr>
</tbody>
</table>
### 27.3.7.3 Working with Content Item Icons and URLs

Table 27–6 and Table 27–7 describe EL expressions for working with icons and URLs associated with content items and properties.

#### Table 27–6  EL Expressions for Content Item Node or Property Icons

<table>
<thead>
<tr>
<th>EL Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{node.icon.largeIcon}</td>
<td>Returns a URL to an image resource for a large icon.</td>
</tr>
<tr>
<td>#{node.propertyMap['myDoc'].icon.largeIcon}</td>
<td>Example: &lt;af:image source=&quot;#{node.propertyMap['projectFolder'].largeIcon}&quot;/&gt;</td>
</tr>
<tr>
<td>#{node.icon.smallIcon}</td>
<td>Returns a URL to an image resource for a small icon.</td>
</tr>
<tr>
<td>#{node.propertyMap['myDoc'].icon.smallIcon}</td>
<td>Example: &lt;af:image source=&quot;#{node.icon.smallIcon}&quot;/&gt;</td>
</tr>
</tbody>
</table>

#### Table 27–7  EL Expressions for Content Item Node URLs

<table>
<thead>
<tr>
<th>EL Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{node.url.downloadUrl}</td>
<td>Creates a URL to the binary content. Forces a download, and the underlying operating system renders the content based on the content type.</td>
</tr>
<tr>
<td>Example: &lt;af:goLink destination=&quot;#{node.url.downloadUrl}&quot; targetFrame=&quot;_blank&quot;/&gt;</td>
<td></td>
</tr>
<tr>
<td>#{node.url.renderUrl}</td>
<td>Creates a URL to the binary content. Allows the browser to render the content based on the content type.</td>
</tr>
<tr>
<td>By default, #{node.url} resolves to #{node.url.renderUrl}.</td>
<td></td>
</tr>
<tr>
<td>Example: &lt;af:goLink destination=&quot;#{node.url.renderUrl}&quot; targetFrame=&quot;_blank&quot;/&gt;</td>
<td></td>
</tr>
</tbody>
</table>

### 27.3.7.4 Working with Image Renditions

Table 27–8 lists the EL expressions available to retrieve information about a particular rendition of an image, including the URL to render the image using that rendition.

Table 27–9 lists the EL expressions available to retrieve different image renditions for Site Studio region elements.
For more information about image renditions, see Section 27.3.6, "Using Image Renditions in Content Presenter Display Templates."

**Table 27–8  EL Expressions for Image Renditions of Image Documents**

<table>
<thead>
<tr>
<th>EL Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{node.renditionsMap['renditionName:url']}</td>
<td>Returns the URL of the image rendition. For example: <a href="http://mymachine.example.com:8888/cs/groups/personalspaces/@pewebcenter/0799c4d7d-255c-46c8-80f5-0d06c848dd65/documents/document/swrf/mdax/~disp/~extract/ID_001642~3~staticrendition/preview.gif">http://mymachine.example.com:8888/cs/groups/personalspaces/@pewebcenter/0799c4d7d-255c-46c8-80f5-0d06c848dd65/documents/document/swrf/mdax/~disp/~extract/ID_001642~3~staticrendition/preview.gif</a>. Wherever possible, url points to a static rendition URL for the image rendition on the WebCenter Content. If the static URL cannot be determined, the url uses the WebCenter Portal showProperty servlet.</td>
</tr>
<tr>
<td>#{node.renditionsMap['renditionName:name']}</td>
<td>Returns the name of the rendition, for example web, thumbnail, or highres. For web and thumbnail renditions, name returns the name of the native file, not of the web or thumbnail rendition.</td>
</tr>
<tr>
<td>#{node.renditionsMap['renditionName:type']}</td>
<td>Returns the file type of the image rendition, for example, preview. The type property is not valid for web and thumbnail renditions.</td>
</tr>
<tr>
<td>#{node.renditionsMap['renditionName:width']}</td>
<td>Returns the width of the image rendition in pixels, for example 250. The width property is not valid for web and thumbnail renditions.</td>
</tr>
<tr>
<td>#{node.renditionsMap['renditionName:height']}</td>
<td>Returns the height of the image rendition in pixels, for example 34. The height property is not valid for web and thumbnail renditions.</td>
</tr>
<tr>
<td>#{node.renditionsMap['renditionName:resolution']}</td>
<td>Returns the resolution (DPI) of the image rendition, for example 96 dpi. The resolution property is not valid for web and thumbnail renditions.</td>
</tr>
<tr>
<td>#{node.renditionsMap['renditionName:size']}</td>
<td>Returns the file size of the image rendition, for example, 3133.</td>
</tr>
<tr>
<td>#{node.renditionsMap['renditionName:contentType']}</td>
<td>Returns the MIME type of the image rendition, for example, image/gif.</td>
</tr>
</tbody>
</table>
Table 27–9  EL Expressions for Image Renditions of Site Studio Region Elements

<table>
<thead>
<tr>
<th>EL Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{node.propertyMap['regionDefinitionName:elementName'].asTextHtml}</td>
<td>Returns an HTML img tag with the src element set to the URL of the image defined in the specified element. For example:</td>
</tr>
<tr>
<td></td>
<td><code>&lt;img src=&quot;http://mymachine.example.com:9400/cs/idcplg?IdcService=GET_FILE&amp;amp;dDocName=id_038497&amp;amp;RevisionSelectionMethod=LatestReleased&quot; alt=&quot;S3-1&quot; class=&quot;&quot;&gt;</code></td>
</tr>
<tr>
<td></td>
<td>Use this EL to reference a Site Studio region element that defines a specific rendition of an image.</td>
</tr>
<tr>
<td></td>
<td>Wherever possible, the URL points to a static rendition URL for the image rendition on the WebCenter Content. If the static URL cannot be determined, the url uses the WebCenter Portal showProperty servlet.</td>
</tr>
<tr>
<td>#{node.propertyMap['regionDefinitionName:elementName/Rendition:renditionName'].asTextHtml}</td>
<td>Returns an HTML img tag with the src element set to the URL of the specified image rendition of the image defined in the specified element. For example:</td>
</tr>
<tr>
<td></td>
<td><code>&lt;img src=&quot;/cs/groups/wc081512c/8a5cdcaaa4610341a8bb88387effdf21790/documents/document/awrf/mdm4/~edisp/~extract/ID_038497~1~staticrendition/preview.gif&quot; alt=&quot;S3-1&quot; class=&quot;&quot;&gt;</code></td>
</tr>
<tr>
<td></td>
<td>If the specified image rendition does not exist for the image, then the URL of the original image is returned.</td>
</tr>
<tr>
<td></td>
<td>Use this EL to reference a Site Studio region element that defines the original image for which you want to display the specified renditionName.</td>
</tr>
<tr>
<td></td>
<td>Wherever possible, the URL points to a static rendition URL for the image rendition on the WebCenter Content. If the static URL cannot be determined, the url uses the WebCenter Portal showProperty servlet.</td>
</tr>
</tbody>
</table>

27.3.7.5 Working with Group Portal Information

Table 27–10 and Table 27–11 provide EL expressions you can use to work with group portal information in your content presenter template.

Table 27–10  EL Expressions for Basic Group Portal Information

<table>
<thead>
<tr>
<th>EL Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{spaceContext.currentSpace.metadata.createdBy}</td>
<td>Returns the user who created the current or specified group portal.</td>
</tr>
<tr>
<td>#{spaceContext.space[portalName].metadata.createdBy}</td>
<td></td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata.creationDate}</td>
<td>Returns a java.util.Calendar object representing the date and time on which the current or specified portal was created.</td>
</tr>
<tr>
<td>#{spaceContext.space[portalName].metadata.creationDate}</td>
<td></td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata.customAttributes[attributeName]}</td>
<td>Returns the value of a specific custom attribute of the name attributeName for the portal.</td>
</tr>
<tr>
<td>#{spaceContext.space[portalName].metadata.customAttributes[attributeName]}</td>
<td></td>
</tr>
</tbody>
</table>
### Table 27–10  (Cont.) EL Expressions for Basic Group Portal Information

<table>
<thead>
<tr>
<th>EL Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{spaceContext.currentSpace.metadata.defaultLanguage}</td>
<td>Returns the default language for the current or specified portal.</td>
</tr>
<tr>
<td>#{spaceContext.space[portalName].metadata.defaultLanguage}</td>
<td></td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata.description}</td>
<td>Returns the description associated with the current portal with the display name in the language in which the portal was created. If the portal name has been translated, the translated name is not shown.</td>
</tr>
<tr>
<td>#{spaceContext.space[portalName].metadata.description}</td>
<td>Example:</td>
</tr>
<tr>
<td>#{spaceContext.space['FinanceProject'].metadata.description}</td>
<td>Evaluates to conglomerate of all teams involved in financial activities.</td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata.displayName}</td>
<td>Returns the display name associated with the current or specified portal in the language in which the portal was created. If the portal name has been translated, the name in which the portal was created will be shown.</td>
</tr>
<tr>
<td>#{spaceContext.space[portalName].metadata.displayName}</td>
<td>Example:</td>
</tr>
<tr>
<td>#{spaceContext.space['Web20Portal'].metadata.displayName}</td>
<td>If a group portal called &quot;Web20Portal&quot; has the display name &quot;Web 2.0 Portal&quot;, then:</td>
</tr>
<tr>
<td></td>
<td>#{spaceContext.space['Web20Portal'].metadata.displayName} will evaluate to &quot;Web 2.0 Portal&quot;.</td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata.guid}</td>
<td>Returns the unique ID associated with the current or specified portal.</td>
</tr>
<tr>
<td>#{spaceContext.space[portalName].metadata.guid}</td>
<td></td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata.icon}</td>
<td>Returns a URL to the icon associated with the current or specified portal.</td>
</tr>
<tr>
<td>#{spaceContext.space[portalName].metadata.icon}</td>
<td></td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata.keywords}</td>
<td>Returns a comma-separated list of searchable keywords associated with the current or specified portal.</td>
</tr>
<tr>
<td>#{spaceContext.space[portalName].metadata.keywords}</td>
<td></td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata.lastUpdatedDate}</td>
<td>Returns the java.util.Calendar object representing the date-time on which the current or specified portal was last updated.</td>
</tr>
<tr>
<td>#{spaceContext.space[portalName].metadata.lastUpdatedDate}</td>
<td></td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata.logo}</td>
<td>Returns the path to the image associated with the logo of the current or specified portal.</td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata.logo}</td>
<td></td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata.name}</td>
<td>Returns the name of the current or specified portal used generally at the back end of the portal and is used with the pretty URL.</td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata.name}</td>
<td></td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata.groupSpaceURI}</td>
<td>Returns the pretty URL of the current or specified portal.</td>
</tr>
<tr>
<td>#{spaceContext.space[portalName].metadata.groupSpaceURI}</td>
<td></td>
</tr>
</tbody>
</table>
Table 27–10  (Cont.) EL Expressions for Basic Group Portal Information

<table>
<thead>
<tr>
<th>EL Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{spaceContext.currentSpace.metadata.closed}</td>
<td>Returns Boolean value indicating whether the portal has been kept closed.</td>
</tr>
<tr>
<td>#{spaceContext.space[portalName].metadata.closed}</td>
<td>Returns Boolean value indicating whether the portal has been kept closed.</td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata.discoverable}</td>
<td>Returns Boolean value indicating whether users will be able to discover the existence of the portal by searching for it or getting it listed in My Portals.</td>
</tr>
<tr>
<td>#{spaceContext.space[portalName].metadata.discoverable}</td>
<td>Returns Boolean value indicating whether users will be able to discover the existence of the portal by searching for it or getting it listed in My Portals.</td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata.offline}</td>
<td>Returns Boolean value indicating whether the portal has been taken offline.</td>
</tr>
<tr>
<td>#{spaceContext.space[portalName].metadata.offline}</td>
<td>Returns Boolean value indicating whether the portal has been taken offline.</td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata.publishRSS}</td>
<td>Returns Boolean value, indicating whether the portal publishes RSS feeds.</td>
</tr>
<tr>
<td>#{spaceContext.space[portalName].metadata.publishRSS}</td>
<td>Returns Boolean value, indicating whether the portal publishes RSS feeds.</td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata.selfRegistration}</td>
<td>Returns a Boolean value indicating whether users are allowed to subscribe themselves to the portal.</td>
</tr>
<tr>
<td>#{spaceContext.space[portalName].metadata.selfRegistration}</td>
<td>Returns a Boolean value indicating whether users are allowed to subscribe themselves to the portal.</td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata.unsubscriptionApprovalRequired}</td>
<td>Returns a Boolean value representing whether approval is required to unsubscribe from the current or specified portal.</td>
</tr>
<tr>
<td>#{spaceContext.space[portalName].metadata.unsubscriptionApprovalRequired}</td>
<td>Returns a Boolean value representing whether approval is required to unsubscribe from the current or specified portal.</td>
</tr>
</tbody>
</table>

Table 27–11  EL Expressions for Portal UI Information

<table>
<thead>
<tr>
<th>EL Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{spaceContext.space[portalName].metadata.gsSiteTemplateId}</td>
<td>Returns the ID of the page template associated with the current or specified portal.</td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata.gsSiteTemplateId}</td>
<td>Returns the ID of the page template associated with the current or specified portal.</td>
</tr>
<tr>
<td>#{spaceContext.space[portalName].metadata.rcForGSPages}</td>
<td>Returns the ID of the resource catalog associated with the current or specified portal.</td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata.rcForGSPages}</td>
<td>Returns the ID of the resource catalog associated with the current or specified portal.</td>
</tr>
<tr>
<td>#{spaceContext.space[portalName].metadata.rcForGSSiteTemplates}</td>
<td>Returns the ID of the resource catalog associated with the page template of the current portal.</td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata.rcForGSSiteTemplates}</td>
<td>Returns the ID of the resource catalog associated with the page template of the current portal.</td>
</tr>
<tr>
<td>#{spaceContext.space[portalName].metadata.gsSiteStructureId}</td>
<td>Returns the ID of the navigation model associated with the current portal.</td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata.gsSiteStructureId}</td>
<td>Returns the ID of the navigation model associated with the current portal.</td>
</tr>
<tr>
<td>#{spaceContext.space[portalName].metadata.skin}</td>
<td>Returns the ADF Faces skin family associated with the current portal.</td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata.skin}</td>
<td>Returns the ADF Faces skin family associated with the current portal.</td>
</tr>
</tbody>
</table>
27.3.8 Discovering Content Type Property Names

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.

Each content item is associated with a specific content type defined in the WebCenter Content repository. Content types can map to WebCenter Content profile definitions and Site Studio region definitions. Types are created on the WebCenter Content and define the properties of the content item. The property names of a content item’s content type, however, are different than the display names, and need to be obtained from the Content Server. For more information, see "Defining Content Types" in Managing Oracle WebCenter Content.

**Note:** You can also use REST services to obtain content type property names. For more information see Chapter 31, "Content Management REST API.”

27.3.9 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 application, as explained in Section 27.5, “Making Content Presenter Display Templates Available.”

27.3.10 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:

```#
ode.propertyMap['RD_NEWS:LEAD'].asTextHtml```

Table 27–11 (Cont.) EL Expressions for Portal UI Information

<table>
<thead>
<tr>
<th>EL Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{spaceContext.space[portalName].metadata.uiMetadata.footerHidden}</td>
<td>Returns the Boolean value representing whether the footer of the portal is hidden.</td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata.uiMetadata.footerHidden}</td>
<td></td>
</tr>
<tr>
<td>#{spaceContext.space[portalName].metadata.uiMetadata.copyrightMessage}</td>
<td>Returns the copyright message used by the portal.</td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata.uiMetadata.copyrightMessage}</td>
<td></td>
</tr>
<tr>
<td>#{spaceContext.space[portalName].metadata.uiMetadata.privacyPolicyUrl}</td>
<td>Returns the URL to the privacy policy document followed.</td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata.uiMetadata.privacyPolicyUrl}</td>
<td></td>
</tr>
</tbody>
</table>
Example 27–9 illustrates how these Site Studio Region elements can be included in a contentTemplateDef definition:

**Example 27–9  Referencing Site Studio Region Elements in a Template**

```xml
<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>
        <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>
</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://www.ateam-oracle.com/content-presenter-cmis-complete

**27.3.11 Using a Content Presenter Display Template**

For information on using a display template in a Content task flow, see Section 29.5, "Adding a Content Task Flow to a Page." See also Section 29.7.1, "Content Presenter Task Flow Parameters and Out-of-the-Box Display Templates." For information on integrating a display template into a Portal Framework application, see the following section Section 27.5, "Making Content Presenter Display Templates Available."

**Note:** If you display a wiki page in a Portal 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 30.3.5, "Displaying Wiki Page Links Within Content Presenter."

**27.4 Adding the Content Presenter Task Flow to a Page**

You can add a Content Presenter task flow at design time or runtime. For information about how to add Content Presenter at design time (using JDeveloper), see Chapter 29,
"Adding Content Task Flows and Document Components to a Portal Page." For information about how to add Content Presenter at runtime, see the "Publishing Content Using Content Presenter" chapter in *Building Portals with Oracle WebCenter Portal*.

## 27.5 Making Content Presenter Display Templates Available

This section explains how to make a Content Presenter display template available to Content Presenter in a portal or Framework application.

- Section 27.5.1, "Export a Content Presenter Display Template as a Portal Resource"
- Section 27.5.2, "Upload the New Content Presenter Display Template"
- Section 27.5.3, "Test the New Content Presenter Display Template"

**Note:** If you are using a locally configured JDev environment to develop a Portal 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 27.5.1, "Export a Content Presenter Display Template as a Portal Resource." You do not need to perform the upload task as explained in Section 27.5.2, "Upload the New Content Presenter Display Template."

### 27.5.1 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 portal and to Content Presenter at runtime.

1. Create a Content Presenter display template, as explained previously in Section 27.3, "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 27–8. The dialog for a multiple-item template is shown in Figure 27–9.
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.

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 this, see the WebCenter Architecture Team blog entry at: http://www.ateam-oracle.com/content-presenter-cmis-complete. See also Section 27.3.10, "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 portal, you must pick the repository that matches the one used by the portal 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 9.6.3, "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 (JSPX) file. You can place this file anywhere on your file system.


Now that your Content Presenter display template is exported as a portal resource, the next step is to upload it to your portal. 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.

27.5.2 Upload the New Content Presenter Display Template

You can upload a new display template to any deployed portal or Framework application that includes the Resource Manager. To upload a Content Presenter display template to a portal or Framework application, an application administrator performs the following steps:

**Note:** The content repository connection names and details must be the same between the design time portal or Framework application in which you developed the Content Presenter display template and the destination application. Also, any references to external files in the display template must either be absolute or relative path references. See Section 27.3.9, "Referencing External Files in Display Templates."

**Note:** Uploading from the scope of a specific portal within WebCenter Portal instance is different than uploading from the application scope of the portal's administration tools. For more information, see the section on application-level versus portal-level resources in Section 55.1.5, "Editing WebCenter Portal Assets in JDeveloper."

1. Open your portal or Framework application.

2. Click Administration.

   Note that you must have administrator privileges.


4. Under Look and Layout, click Content Presenter.

5. Select Upload, as shown in Figure 27–10.
6. In the Upload New Content Presenter dialog, navigate to the portal resource file you saved on your file system.

7. After you've selected the template file, click OK. If the operation is successful, an Information dialog appears with the message "Resource uploaded successfully."

8. Locate your Content Presenter display template in the list of Content Presenter templates and select it as shown in Figure 27–11.

9. To make the template visible to users and available for use in Content Presenter, check the Available check box.

The new Content Presenter display template is now ready to be used with Content Presenter in a portal or Framework application.

27.5.3 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 portal or Framework application to which you added a Content Presenter display template. This application must have a
WebCenter Content connection that Content Presenter can use. For portals, if you just completed the setup procedures described in the previous sections, click **Back to return to the Portal** link in the upper right corner of the Administration page.

2. Navigate to the portal or Framework application in which you want to use the new Content Presenter display template. Note that you’ll need to be able to edit it.

3. Optionally, create a new page to test the display template. Select **Create a New Page** from the Page Actions menu. Then 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 Content Presenter.

6. In the Add Content dialog, open the **Content Management** folder, as shown in **Figure 27–12**.

**Figure 27–12  Opening the Content 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 27–13**).
10. If this is a new page, select one or more Content items on the Content tab.
11. In the Content Presenter Configuration window, select the Template tab.
12. From the Template menu, select your Content Presenter display template, as shown in Figure 27–14.

**Figure 27–13  Selecting the Edit Icon**

![Selecting the Edit Icon](image)

**Figure 27–14  Selecting the Content Presenter Display Template**

![Selecting the Content Presenter Display Template](image)

The Content Presenter template will be applied to the Content Presenter instance. For more information on using Content Presenter and display templates, see Section 29.2, "Understanding the Documents Task Flows" and Section 27.2, "Using the Out-of-the-Box Display Templates."

### 27.6 What Happens at Runtime?

This section includes these topics:

- Section 27.6.1, "Runtime Overview"
- Section 27.6.2, "Identifying Display Templates for Selected Content Items"

#### 27.6.1 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 29.6, "Modifying Content Task Flow Parameters" and Section 27.2, "Using the Out-of-the-Box Display Templates").

If you integrate the Documents Service in your application (see Chapter 28, "Integrating Documents"), 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 the "Publishing Content Using Content Presenter" chapter in Building Portals with Oracle WebCenter Portal. To understand how Content Presenter identifies which display templates to expose in the Content Presenter Configuration dialog, see Section 27.6.2, "Identifying Display Templates for Selected Content Items."

### 27.6.2 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 9, "Introduction to Portal Resource Management." Note that uploaded templates must also be "available" in the list of resources for them to be selectable. See Section 27.5.2, "Upload the New Content Presenter Display Template" for more information.

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.

### 27.7 Configuring Coherence as the Caching Mechanism

Content Presenter, by default, is configured to use a local (in-memory) cache. However, using Coherence for any type of production environment is strongly recommended, and is a requirement for High Availability (HA) environments. This section describes how to enable Coherence as the caching mechanism for Framework applications.
You can enable content caching with Coherence for both WebCenter Portal and Framework applications by modifying the Coherence configuration file. For WebCenter Portal, only the steps described in the "Modifying Cache Settings for Content Presenter" section in *Administering Oracle WebCenter Portal* are required. For Framework applications, however, an additional step is required to add the configuration file to the application (EAR) classpath or server's system classpath.

This section contains the following subsections:

- Section 27.7.1, "Adding the Coherence Configuration File to the EAR Classpath"
- Section 27.7.2, "Verifying the Coherence Configuration"

### 27.7.1 Adding the Coherence Configuration File to the EAR Classpath

The following steps show how to add the XML file in the EAR classpath.

1. Complete the steps described in the "Modifying Cache Settings for Content Presenter" section in *Administering Oracle WebCenter Portal*.
2. Under the application's src directory (for example, WebCenterPortalCoherence/src), create a directory named APP-INF/classes and place the content-coherence-cache-config.xml in it.
3. Open the application's Application Properties and select Deployment.
4. Edit the profile contained in the EAR file (see Figure 27–15).

![Figure 27–15 Adding the XML File to the EAR Classpath](image)

5. Add a new File Group named Packaging. Leave the Type as Packaging.

Note: Your Coherence license may or may not support multi-node environments depending on the license option you have purchased. To check what your license agreement provides, see the "Oracle Coherence" section in *Licensing Information*. 
6. Make sure classes and content-coherence-cache-config.xml file are selected in the Filters pane.

7. Deploy the application and restart the managed server on which it was deployed. After deploying your application, Coherence should be enabled.

### 27.7.2 Verifying the Coherence Configuration

After restarting the managed server where your application is deployed, connect to it by entering `jconsole` from the command line and choosing the process corresponding to `WC_Spaces` to open JConsole. In JConsole, check for Coherence in the MBeans tab. You should also see something like the following output in the `WC_Spaces-diagnostic.log` file:

```plaintext
```

### 27.8 Optimizing Performance 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 27–12 shows whether or not node properties are loaded immediately upon node retrieval, by retrieval mechanism.

For information about the node properties listed in Table 27–12, see Configuration Reference for Oracle WebCenter Content.

#### Table 27–12 Loading of Node Properties By Node Retrieval Mechanism

<table>
<thead>
<tr>
<th>OCS Global Profile Properties</th>
<th>GET BY PARENT ID (&quot;Contents Under a Folder&quot;)</th>
<th>SEARCH (&quot;Results of a Query&quot;)</th>
<th>GET BY UUID (&quot;Single Content Item&quot; and &quot;List of Items&quot;)</th>
</tr>
</thead>
<tbody>
<tr>
<td>VaultFileSize</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dCheckoutUser</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>dCreateDate</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dDocAccount</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dDocAuthor</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dDocName</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
</tbody>
</table>
The following supplementary tutorials and examples further illustrate how Content Presenter can be used:

- **Using Content Presenter Templates** - describes how to build a region definition in Site Builder, and then use that region definition as the basis for a Content Presenter template (rather than a Site Studio region template), and finally, add the template to a Portal Framework application page.

  http://yonaweb.be/webcenter_tutorial/using_content_presenter_templates

- **UCM, Site Studio and Templates** - describes the components (Content Server, Site Studio, and Content Presenter) that drive content interactions in WebCenter Portal.

### Table 27–12 (Cont.) Loading of Node Properties By Node Retrieval Mechanism

<table>
<thead>
<tr>
<th>OCS Global Profile Properties</th>
<th>GET BY PARENT ID (&quot;Contents Under a Folder&quot;)</th>
<th>SEARCH (&quot;Results of a Query&quot;)</th>
<th>GET BY UUID (&quot;Single Content Item&quot; and &quot;List of Items&quot;)</th>
</tr>
</thead>
<tbody>
<tr>
<td>dDocTitle</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dDocType</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dFormat</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dID</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dInDate</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dIsCheckedOut</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>dOutDate</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dReleaseDate</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>dReleaseState</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>dRevClassID</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>dRevLabel</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dRevRank</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>dRevisionID</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dSecurityGroup</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dStatus</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>dWebExtension</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dWorkflowState</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>idcPrimaryFile</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>idcRenditions</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>xCollectionID</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>xComments</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>xForceFolderSecurity</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>xHidden</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>xInhibitUpdate</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>xReadOnly</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
</tbody>
</table>

- *Content Presenter-CMIS-Complete* - steps you through using Content Presenter-based pages and complex CMIS and custom templates to create a filterable content list viewer.
  
  http://www.ateam-oracle.com/content-presenter-cmis-complete

- *Portal and Content - Content Integration - Best Practices* - highlights some of the best practices to consider when integrating content into your portal.
  
This chapter describes how to integrate documents into a Portal Framework application to both add content to the application, and to provide end users with content and documents task flows built into the application to manage, display, and search documents at runtime.

Note: For a description of the other ways to integrate content into a WebCenter Portal Framework application, see Chapter 24, "Introduction to Integrating and Publishing Content."

This chapter includes the following topics:

- Section 28.1, "Introduction to Documents in WebCenter Portal Framework Applications"
- Section 28.2, "Setting Up Connections"
- Section 28.3, "Setting Security for the Documents Tool"
- Section 28.4, "Using the Documents Tool with Other WebCenter Services"
- Section 28.5, "Setting Parameters to Upload Files to Content Repositories"

28.1 Introduction to Documents in WebCenter Portal Framework Applications

The Documents tool provides features for accessing, adding, and managing folders and files; configuring and viewing file and folder properties; and searching file and folder content in WebCenter Content, Oracle Portal, SharePoint, or, in development environments, File System content repositories. It provides these features through documents 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 25.2.3.1, "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 documents into a Portal 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 28.2, “Setting Up Connections.”

28.2 Setting Up Connections

Before you can use the Documents tool and task flows, you must first set up the connection to the WebCenter Content, Oracle Portal, or SharePoint content repository that contains the documents you want to manage. You can reuse connections you have 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 also later add, delete, or modify connections in your deployed environment using Enterprise Manager Fusion Middleware Control. For more information, refer to the Administering 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 25.2, "Configuring Content Repository Connections."

28.3 Setting Security for the Documents Tool

The documents 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 WebCenter Content or Oracle Portal. For more information about using security with content repositories, see Section 26.4, "Securing a Content Repository Data Control." For information about using external applications, see Section 74.13, "Working with External Applications."

If you are using a content repository that handles its own authentication, such as Oracle Portal or Oracle WebCenter Content, 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 28–1).
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 28–2) then click **Next**.

5. On the General page, under **Authentication Details**, choose **BASIC** from the **Authentication Method** list (Figure 28–3), then click **Next**.

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 28–4).

9. Click **Finish**. Notice that your connection now uses the external application for authentication (Figure 28–5).
10. In the Edit Content Repository Connection dialog box, click OK. In the Application Resources panel, notice that the external application now appears (Figure 28–6).

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.

28.4 Using the Documents Tool with Other WebCenter Services

You can use the Documents tool 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 tool 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 Chapter 4, "Preparing Your Application for WebCenter Portal Tools and Services."

**Note:** When you integrate the Documents tool with the Search service, the Search service returns results from all content repository connections.

28.5 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 28–1) or simply update the code in the Source view. After you have made your changes, save the web.xml file, then restart Oracle JDeveloper.

Table 28–1   Apache MyFaces Trinidad Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>org.apache.myfaces.trinidad.UPLOAD_MAX_MEMORY</td>
<td>The maximum amount of memory in bytes that a single file can use when uploaded</td>
</tr>
<tr>
<td>org.apache.myfaces.trinidad.UPLOAD_MAX_DISK_SPACE</td>
<td>The maximum amount of disk space in bytes that a single file can use when uploaded</td>
</tr>
<tr>
<td>org.apache.myfaces.trinidad.UPLOAD_TEMP_DIR</td>
<td>The directory in which the file being uploaded is temporarily stored</td>
</tr>
</tbody>
</table>

For more information, see the Apache MyFaces Trinidad documentation at http://myfaces.apache.org/trinidad/devguide/fileUpload.html.
This chapter describes how to add content to a page in a Portal Framework application using the content task flows and document components. In a Portal Framework application, you can add content from one or more connected content repositories to the application, using Content Presenter or the documents 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.

---

**Note:**

- For a description of the other ways to integrate content into a WebCenter Portal Framework Application, see Chapter 24, "Introduction to Integrating and Publishing Content."
- For information about viewing, editing, and managing content in content task flows, see the "Working with Documents" part in *Using Oracle WebCenter Portal*.

---

This chapter includes the following topics:

- Section 29.1, "Understanding the Content Presenter Task Flow"
- Section 29.2, "Understanding the Documents Task Flows"
- Section 29.3, "Understanding Document Components"
- Section 29.4, "Adding a Selected Folder or File to a Page"
- Section 29.5, "Adding a Content Task Flow to a Page"
- Section 29.6, "Modifying Content Task Flow Parameters"
- Section 29.7, "Content Task Flow Parameters"

### 29.1 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 WebCenter Content and your WebCenter Portal administrator has completed the prerequisite configuration.
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 tool for adding or managing the content it displays.

To add the Content Presenter task flow to a page, see Section 29.4, "Adding a Selected Folder or File to a Page" and Section 29.5, "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 29–4, "Content Presenter Task Flow Parameters"

For information about creating a page template, see Section 11.3, "Creating a Page Template."

---

### 29.2 Understanding the Documents Task Flows

The documents task flows provide a variety of formats to display folders and files, including wikis and blogs, on a page in a Portal 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 WebCenter Content, 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 25.2.3.1, "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 task flow to a page in a Portal Framework application, your WebCenter Portal administrator must first integrate the Documents tool into the application by establishing a content repository connection, as described in Chapter 28, "Integrating Documents" and Chapter 25, "Configuring Content Repository Connections."
Using documents 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 task flows built into the application to manage, display, and search documents at runtime.

Table 29–1 provides an overview of the documents task flows.

### Table 29–1  Documents Task Flows

<table>
<thead>
<tr>
<th>Task Flow</th>
<th>Folder and File Listings</th>
<th>Individual Folders</th>
<th>Individual Files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Document Explorer task flow</strong>&lt;br&gt;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 29–1.&lt;br&gt;**</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td><strong>Document List Viewer task flow</strong>&lt;br&gt;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 29–2.</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td><strong>Document Manager task flow</strong>&lt;br&gt;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 29–3, Figure 29–4, and Figure 29–5.</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td><strong>Document Navigator task flow</strong>&lt;br&gt;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 29–6.</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td><strong>Folder Viewer task flow</strong>&lt;br&gt;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 29–7.</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td><strong>Recent Documents task flow</strong>&lt;br&gt;Displays a list of the files most recently created or modified files in the current folder. See Figure 29–8.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Document Viewer task flow</strong>&lt;br&gt;Displays a preview of a file, or file properties for files that do not support a preview. See Figure 29–9 and Figure 29–10.</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td><strong>Document Mini Properties task flow</strong>&lt;br&gt;Displays the Basic properties of a file in a read-only view. See Figure 29–11.</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td><strong>Document Properties task flow</strong>&lt;br&gt;Displays both Basic and Advanced properties of a file, along with an Edit button to allow you to modify property values. See Figure 29–12.</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>
To add a documents task flow to a page, see Section 29.4, "Adding a Selected Folder or File to a Page" and Section 29.5, "Adding a Content Task Flow to a Page."

For detailed information about each documents task flow, click the links below to display the pertinent sections in Building Portals with Oracle WebCenter Portal.

- "Understanding the Document Explorer Task Flow"

  Figure 29–1 shows an example of the Document Explorer task flow. For information about setting properties for this task flow, see Section 29.7.2, "Document Explorer Task Flow Parameters."

- "Understanding the Document List Viewer Task Flow"

  Figure 29–2 shows an example of the Document List Viewer task flow. For information about setting properties for this task flow, see Section 29.7.3, "Document List Viewer Task Flow Parameters."
Figure 29–2  Document List Viewer Task Flow

Figure 29–3, Figure 29–4, and Figure 29–5 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 29.7.4, "Document Manager Task Flow Parameters."

Figure 29–3  Document Manager Task Flow: Explorer Layout

Figure 29–4  Document Manager Task Flow: Table Layout
"Understanding the Document Navigator Task Flow"

Figure 29–6 shows an example of the Document Navigator task flow. For information about setting properties for this task flow, see Section 29.7.5, "Document Navigator Task Flow Parameters."

"Understanding the Folder Viewer Task Flow"

Figure 29–7 shows an example of the Folder Viewer task flow. For information about setting properties for this task flow, see Section 29.7.6, "Folder Viewer Task Flow Parameters."
Understanding the Recent Documents Task Flow

Figure 29–7 shows an example of the Recent Documents task flow. For information about setting properties for this task flow, see Section 29.7.7, "Recent Documents Task Flow Parameters."

Figure 29–8 Recent Documents Task Flow

"Understanding the Document Viewer Task Flow"

Figure 29–9 and Figure 29–10 show examples of the Document Viewer task flow for different file types. For information about setting properties for this task flow, see Section 29.7.8, "Document Viewer Task Flow Parameters."
Figure 29–9  Document Viewer Task Flow for Word File
"Understanding the Document Mini Properties Task Flow"

Figure 29–11 shows an example of the Document Mini Properties task flow. For information about setting properties for this task flow, see Section 29.7.9, “Document Mini Properties Task Flow Parameters.”

"Understanding the Document Properties Task Flow"

Figure 29–12 shows an example of the Document Properties task flow. For information about setting properties for this task flow, see Section 29.7.10,
"Document Properties Task Flow Parameters."

*Figure 29–12  Document Properties Task Flow*

![Document Properties Task Flow Diagram](image)

**Basic**

- **Edit**
  - **Name**: Lincoln stamps.doc
  - **Identifier**: dev-ucm#dDocName:SLC04RNTUSORAC010406

**Description**

- **Type**: Document
- **Size**: 133 KB

- **Created**: 4/26/13 11:42 AM
- **Created By**: @weblogic
- **Last Modified**: 4/26/13 11:42 AM
- **Last Modified By**: @weblogic

**Advanced**

- **Content ID**: SLC04RNTUSORAC010406
- **User Access List**
- **Group Access List**
- **Role Access List**
- **Web Site Object Type**
- **Web Sites**
- **Exclude From Lists**
- **Web Site Section**
- **Region Definition**
- **Tags**
- **Page ID**
- **Workflow Assignment**
- **Workflow Approval User List**
  - **CustomText**
  - **CustomText**
  - **CustomDate**
  - **CustomMemo**

- "Understanding the Document Version History Task Flow"

*Figure 29–13 and Figure 29–14* show examples of the Document Version History task flow, where the `horizontalLayout` property is set to format vertically or horizontally, respectively. For information about setting properties for this task flow, see *Section 29.7.13, "Document Version History Task Flow Parameters."*
29.3 Understanding Document Components

The Documents tool provides features for adding document components to a portal page. As described in Section 29.4, "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 29–15).

Figure 29–15 Adding a 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 29–16).

Figure 29–16 Link Component (for Microsoft Word file)
- An **Inline Frame** displays the content of a selected file as a preview on the page (Figure 29–17).

*Figure 29–17  Inline Frame Component (for HTML file)*

**Lincoln Portraits**

**United States**

- An **Image** displays a selected file as an image on the page (Figure 29–18).

*Figure 29–18  Image Component (for JPEG file)*

To add a document component to a page in a Portal Framework application, you must first integrate the Documents tool into the application by establishing a connection to a content repository, as described in Chapter 28, "Integrating Documents."

Table 29–2 shows the document components available for the different content types.

<table>
<thead>
<tr>
<th>Content Type</th>
<th>Document Components</th>
</tr>
</thead>
<tbody>
<tr>
<td>Folder</td>
<td>Link (ADF Go Link)</td>
</tr>
<tr>
<td>Documents of various types (XML, PDF, JAVA, TXT, DOC, XLS, HTM)</td>
<td>Link (ADF Go Link)</td>
</tr>
<tr>
<td>Documents that can be rendered in a browser (HTML, flash, PDF and image)</td>
<td>Inline frame (ADF Inline Frame)</td>
</tr>
<tr>
<td>Images (PNG, JPG, GIF)</td>
<td>Link (ADF Go Link)</td>
</tr>
<tr>
<td></td>
<td>Inline frame (ADF Inline Frame)</td>
</tr>
<tr>
<td></td>
<td>Image (ADF Image)</td>
</tr>
</tbody>
</table>

To display a folder or file using one of these document components, see Section 29.4, "Adding a Selected Folder or File to a Page."
29.4 Adding a Selected Folder or File to a Page

This section describes how to display an individual folder or file from a connected content repository on a page, as shown in Figure 29–19 and Figure 29–20, by choosing the required container for the selected folder or file. If you want to add a content task flow to a page, independent of a specific folder or file, refer to Section 29.5, "Adding a Content Task Flow to a Page."

Figure 29–19 Adding a Folder to Portal Page

Table 29–3 lists all containers available for displaying folders and files on a portal page:

Table 29–3

<table>
<thead>
<tr>
<th>Container Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADF Co Link</td>
<td></td>
</tr>
<tr>
<td>Documents</td>
<td></td>
</tr>
<tr>
<td>Document Viewer</td>
<td></td>
</tr>
<tr>
<td>Document Manager</td>
<td></td>
</tr>
<tr>
<td>Folder Viewer</td>
<td></td>
</tr>
<tr>
<td>Documents</td>
<td></td>
</tr>
<tr>
<td>Document Navigator</td>
<td></td>
</tr>
<tr>
<td>Documents</td>
<td></td>
</tr>
<tr>
<td>List Viewer</td>
<td></td>
</tr>
<tr>
<td>Documents</td>
<td></td>
</tr>
<tr>
<td>Document Explorer</td>
<td></td>
</tr>
<tr>
<td>Documents</td>
<td></td>
</tr>
<tr>
<td>Upload</td>
<td></td>
</tr>
<tr>
<td>Documents</td>
<td></td>
</tr>
<tr>
<td>Rich Text Editor</td>
<td></td>
</tr>
<tr>
<td>Documents</td>
<td></td>
</tr>
<tr>
<td>Content Presenter</td>
<td></td>
</tr>
<tr>
<td>Blogs</td>
<td></td>
</tr>
</tbody>
</table>

Figure 29–20 Adding a File to Portal Page
Adding a Selected Folder or File to a Page

To add a selected folder or file to a page in a Portal Framework application (see Section 6.1, "Creating a New WebCenter Portal Framework Application"):

Table 29–3  Containers for Adding Individual Folders and Files to a Page

<table>
<thead>
<tr>
<th>Root Folder</th>
<th>Folders</th>
<th>Individual Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Document Explorer</td>
<td>• Content Presenter</td>
<td>• Content Presenter</td>
</tr>
<tr>
<td>• Document List Viewer</td>
<td>• Document Explorer</td>
<td>• Document Viewer</td>
</tr>
<tr>
<td>• Document Manager</td>
<td>• Document List Viewer</td>
<td>• Document Mini</td>
</tr>
<tr>
<td>• Document Navigator</td>
<td>• Document Manager</td>
<td>• Properties</td>
</tr>
<tr>
<td>• Folder Viewer</td>
<td>• Document Navigator</td>
<td>• Rich Text Editor</td>
</tr>
<tr>
<td>• Recent Documents</td>
<td>• Folder Viewer</td>
<td>• Document Upload</td>
</tr>
<tr>
<td>• Document Viewer(^1)</td>
<td>• Document Viewer(^1)</td>
<td>• Document Version</td>
</tr>
<tr>
<td>• Document Mini Properties(^1)</td>
<td>• Document Mini Properties(^1)</td>
<td>• History</td>
</tr>
<tr>
<td>• Document Properties(^1)</td>
<td>• Document Properties(^1)</td>
<td>• AutoVue</td>
</tr>
<tr>
<td>• Document Upload</td>
<td>• Document Upload</td>
<td>• Link component</td>
</tr>
<tr>
<td>• Rich Text Editor</td>
<td>• Rich Text Editor</td>
<td>• Inline Frame component</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Image component</td>
</tr>
</tbody>
</table>

\(^1\) To show folder information in a Document Viewer, Document Mini Properties, or Document Properties task flow, you must set the task flow's **resourceID** parameter to the ID of the target folder. See Section 29.7, "Content Task Flow Parameters."

For more information about any of the containers listed in Table 29–3, see Section 29.1, "Understanding the Content Presenter Task Flow,” Section 29.2, ”Understanding the Documents Task Flows,” and Section 29.3, ”Understanding Document Components.”

**Note:** When an item that is stored in a content repository is dropped onto a page, the Portal 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 WebCenter Content. 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/owcs-ps3-sharepoint-wcs-wp-335282.pdf](http://www.oracle.com/technetwork/middleware/webcenter/owcs-ps3-sharepoint-wcs-wp-335282.pdf).
1. Follow the steps in Section 4.2.1, "How to Prepare Your Application to Consume Tools and Services" to implement security.

2. If you want to add a documents task flow, create a content repository connection, which adds the Documents tool to your application (see Chapter 28, "Integrating Documents").

3. Create or open a page in your application where you want to add the task flow (see Section 15.2, "Creating Pages in a WebCenter Portal Framework Application").

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 18, "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 18.1.6, "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 29–21 (root folder), Figure 29–22 (subfolder), and Figure 29–23 (individual file).

---

Figure 29–21  Adding a Content Repository Root Folder to Portal Page
8. Choose the content task flow or document component that you want 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 WebCenter Content. When you select the Content Presenter task flow, the folder or file displays in a default display template for its type.

The Document Viewer, Document Mini Properties, and Document Properties task flows are not available in the Create menu for a folder. To show folder information in one of these task flows, you must first add an empty task flow to the page and then set the task flow’s resource ID parameter to the ID of the target folder.

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 29.7, “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.
29.5 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 24, "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 Portal Framework application:

1. Follow the steps in Section 4.2.1, "How to Prepare Your Application to Consume Tools and Services" to implement security.

2. If you want to add a documents task flow, see Chapter 28, "Integrating Documents".

3. Create or open a page in your application where you want to add the task flow (see Section 15.2, "Creating Pages in a WebCenter Portal Framework Application").

4. Before you add a task flow to your page, consider whether you want to allow end users of your application to edit the task flow:
   - To enable runtime editing of the page, see Chapter 18, "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 18.1.6, "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 29–24).

---

Figure 29–24  Adding a Standalone Documents Task Flow to JSF Page

For information about viewing, editing, and managing content in content task flows, see the "Working with Content in a Portal" part in Building Portals with Oracle WebCenter Portal.
6. From the menu that displays, choose Region to open the Edit Task Flow Binding dialog (Figure 29–25).

**Figure 29–25  Specifying Parameters for Document Explorer Task Flow**

![Edit Task Flow Binding](image)

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 29.7, "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 29.6, "Modifying Content Task Flow Parameters."

For information about viewing, editing, and managing content in content task flows, see the "Working with Content in a Portal" part in *Building Portals with Oracle WebCenter Portal*.

**Note:** For information about adding custom actions to task flows, see Section 20.3.2, "How to Enable Custom Actions on a Show Detail Frame Enclosing a Task Flow."
29.6 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**, select the task flow you added. Figure 29–26 shows an example of a Document List Viewer task flow in the **Executables** section.

   ![Figure 29–26  Document List View Task Flow in the Bindings View](image)

   Executables
   - Taskflow - pageeditorpanel
   - Taskflow - docIndocDocumentListviewer
   - Taskflow - forummainl
   - Taskflow - announcementmainview1
   - Variables
   - OmniPortlet1_1
   - OmniPortlet2_1

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 29.7, "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.

---

**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 *Web User Interface Developer’s Guide for Oracle Application Development Framework*. 

---
5. Click OK to add the parameters.

29.7 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 29.7.1, "Content Presenter Task Flow Parameters and Out-of-the-Box Display Templates"
- Section 29.7.2, "Document Explorer Task Flow Parameters"
- Section 29.7.3, "Document List Viewer Task Flow Parameters"
- Section 29.7.4, "Document Manager Task Flow Parameters"
- Section 29.7.5, "Document Navigator Task Flow Parameters"
- Section 29.7.6, "Folder Viewer Task Flow Parameters"
- Section 29.7.7, "Recent Documents Task Flow Parameters"
- Section 29.7.8, "Document Viewer Task Flow Parameters"
- Section 29.7.9, "Document Mini Properties Task Flow Parameters"
- Section 29.7.10, "Document Properties Task Flow Parameters"
- Section 29.7.11, "Rich Text Editor Task Flow Parameters"
- Section 29.7.12, "Document Upload Task Flow Parameters"
- Section 29.7.13, "Document Version History Task Flow Parameters"

29.7.1 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 WebCenter Content. This task flow displays selected documents in out-of-the-box or custom display templates.
See Also:

- Section 29.1, "Understanding the Content Presenter Task Flow"
- Section 29.5, "Adding a Content Task Flow to a Page"

Table 29–4 describes the Content Presenter task flow parameters. Table 29–5 and Table 29–6 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 information about the Content Presenter Configuration dialog at runtime, see the "Publishing Content Using Content Presenter" chapter in *Building Portals with Oracle WebCenter Portal*.

- 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 Portal 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 29–4  Content Presenter Task Flow Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>datasource</td>
<td>The data source of the content. The value depends on the value of datasourceType:</td>
</tr>
<tr>
<td></td>
<td>■ When datasourceType=dsTypeSingleNode, set datasource to a</td>
</tr>
<tr>
<td></td>
<td>single document node identifier in the format:</td>
</tr>
<tr>
<td></td>
<td>connection_name#dDocName:content_id</td>
</tr>
<tr>
<td></td>
<td>Example: xmlns.oracle.com#dDocName:STAN_IDC-007619</td>
</tr>
<tr>
<td></td>
<td>■ When datasourceType=dsTypeFolderContents, set datasource to a</td>
</tr>
<tr>
<td></td>
<td>single folder node identifier in the format:</td>
</tr>
<tr>
<td></td>
<td>connection_name#dCollectionID:collection_id</td>
</tr>
<tr>
<td></td>
<td>Example: xmlns.oracle.com#dCollectionID:45535</td>
</tr>
<tr>
<td></td>
<td>■ When datasourceType=dsTypeQueryExpression, set datasource to a</td>
</tr>
<tr>
<td></td>
<td>CMIS search expression.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Using a CMIS query as the datasource requires a valid</td>
</tr>
<tr>
<td></td>
<td>taskFlowInstId.</td>
</tr>
<tr>
<td></td>
<td>Examples:</td>
</tr>
<tr>
<td></td>
<td>select * from cmis:document where cmis:name like 'test%'</td>
</tr>
<tr>
<td></td>
<td>connectionName=connection_name#select * from cmis:document where cmis:createdBy = 'weblogic'</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> If connectionName is not specified, then the primary</td>
</tr>
<tr>
<td></td>
<td>connection will be used.</td>
</tr>
<tr>
<td></td>
<td>For more information about how to format the query and for</td>
</tr>
<tr>
<td></td>
<td>more examples, see Chapter 31, &quot;Content Management REST API.&quot;</td>
</tr>
<tr>
<td></td>
<td>■ When datasourceType=dsTypeMultiNode, set datasource to a</td>
</tr>
<tr>
<td></td>
<td>set of comma-delimited document node identifiers in the format:</td>
</tr>
<tr>
<td></td>
<td>connection_name#dDocName:content_id, connection_name#dDocName:content_id,...</td>
</tr>
<tr>
<td></td>
<td>Example: myconn#dDocName:DOCUMENT_ID_12345,myconn#dDocName:DOCUMENT_ID_56789</td>
</tr>
</tbody>
</table>
### Table 29–4 (Cont.) Content Presenter Task Flow Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| datasource (cont'd) | 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  
  
  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 WebCenter Content.  
  `collection_id` = the dCollectionID found in the URL for the folder information page in WebCenter Content, 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 the "Configuring Content Presenter" chapter in Administering Oracle WebCenter Portal and Appendix A, "WebCenter Portal Files" (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.  
  
  For information about the Personalization Conductor and scenarios, see Chapter 66, "Personalizing Oracle WebCenter Portal Applications." |
| 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 66, "Personalizing Oracle WebCenter Portal Applications." |
| maxResults         | The maximum number of results to display when `datasourceType` is `dsTypeQueryExpression`.  
  
  Default: 100 |
The out-of-the-box display templates provided by Oracle WebCenter are listed in Table 29–5 and Table 29–6.

### Table 29–4  (Cont.) Content Presenter Task Flow Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>regionTemplate</td>
<td>Specifies whether the display template is a Site Studio region definition template. This value is valid only with WebCenter Content 11g or higher:</td>
</tr>
<tr>
<td></td>
<td>■ Selected: Display template is a Site Studio region definition template. The templateView value is set to the Content ID of the region template.</td>
</tr>
<tr>
<td></td>
<td>■ Cleared (default): Display template is not a Site Studio region definition template.</td>
</tr>
<tr>
<td></td>
<td>For information about creating and using Site Studio region templates, see the “Publishing Content Using Content Presenter” chapter in <em>Building Portals with Oracle WebCenter Portal</em> and the Oracle Site Studio documentation library.</td>
</tr>
<tr>
<td>taskFlowInstId</td>
<td>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. Note: Using a CMIS query as the datasource requires a valid taskFlowInstId.</td>
</tr>
<tr>
<td>templateCategory</td>
<td>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 29–6) or a custom category created for a display template for multiple content items that has been defined as described in Section 27.3, &quot;Creating Content Presenter Display Templates&quot;.</td>
</tr>
<tr>
<td>templateView</td>
<td>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 29–5), a custom display template that has been defined as described in Section 27.3, &quot;Creating Content Presenter Display Templates&quot;, or set to the contentID of a region template if the content is a region.</td>
</tr>
</tbody>
</table>

The out-of-the-box display templates provided by Oracle WebCenter are listed in Table 29–5 and Table 29–6.

### Table 29–5  Out-of-the-Box Templates for Displaying Single Content Items

<table>
<thead>
<tr>
<th>Single Content Item Display Templates</th>
<th>View ID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Document Details View</td>
<td>oracle.webcenter.content.templates.default.document.details</td>
<td>Displays detailed information about any single content item including creation date, modification date, created by username, modified by username, path and any comments.</td>
</tr>
</tbody>
</table>
### Table 29–5  Out-of-the-Box Templates for Displaying Single Content Items

<table>
<thead>
<tr>
<th>Single Content Item Display Templates</th>
<th>View ID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default List Item View</td>
<td><code>oracle.webcenter.content.templates.default.list.item</code></td>
<td>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.</td>
</tr>
<tr>
<td>Default View (default when no template is selected)</td>
<td><code>oracle.webcenter.content.templates.default.detail</code></td>
<td>Displays any single content item, either directly in the browser (images, HTML, text) or as a link that downloads the associated file when clicked.</td>
</tr>
<tr>
<td>Full Article View</td>
<td><code>oracle.webcenter.content.templates.sitestudio.fullarticle</code></td>
<td>Displays a full article, including the Title, Image, and Body of an article. This is a responsive template, meaning that it uses CSS3 media queries to produce a responsive layout—a layout that adjusts to the width of the browser. For more information, see Section 27.3.4, &quot;Using Responsive Templates.&quot; This template requires Site Studio to be enabled on the WebCenter Content, as it uses the <code>RD_ARTICLE</code> Site Studio region definition.</td>
</tr>
</tbody>
</table>

### Table 29–6  Out-of-the-Box Templates for Displaying Multiple Content Items

<table>
<thead>
<tr>
<th>Multiple Content Item Display Templates</th>
<th>View ID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accordion View</td>
<td><code>oracle.webcenter.content.templates.default.list.panelaccordion</code></td>
<td>Displays multiple content items in an accordion format, where each item can be expanded to display its details.</td>
</tr>
<tr>
<td>Articles View (under the Site Studio Template category)</td>
<td><code>oracle.webcenter.content.templates.sitestudio.articles</code></td>
<td>Displays the summaries of multiple articles, including the Title, Image, and Summary of articles. This is a responsive template, meaning that it uses CSS3 media queries to produce a responsive layout—a layout that adjusts to the width of the browser. For more information, see Section 27.3.4, &quot;Using Responsive Templates.&quot; This template requires Site Studio to be enabled on the WebCenter Content, as it uses the <code>RD_ARTICLE</code> Site Studio region definition.</td>
</tr>
<tr>
<td>Bulleted View</td>
<td><code>oracle.webcenter.content.templates.default.list.bulleted</code></td>
<td>Displays multiple content items in a bulleted list format. Only content items will be displayed; folder items will be omitted.</td>
</tr>
</tbody>
</table>
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 Building Portals with Oracle WebCenter Portal
- Section 29.5, "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 29–7 describes the Document Explorer task flow parameters.

#### Table 29–7 Out-of-the-Box Templates for Displaying Multiple Content Items

<table>
<thead>
<tr>
<th>Multiple Content Item Display Templates</th>
<th>View ID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bulleted with Folder Label View</td>
<td>oracle.webcenter.content.templates.default.list.bulleted.label</td>
<td>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.</td>
</tr>
<tr>
<td>Carousel View</td>
<td>oracle.webcenter.content.templates.default.list.carousel</td>
<td>Displays multiple content items in a carousel format, where items can be browsed by moving a slider left or right.</td>
</tr>
<tr>
<td>Icon View</td>
<td>oracle.webcenter.content.templates.default.list.tile</td>
<td>Displays multiple content items in a tiled format with large icons and file names.</td>
</tr>
<tr>
<td>List View (default when no template is selected)</td>
<td>oracle.webcenter.content.templates.default.list.simple</td>
<td>Displays multiple content items in a simple list.</td>
</tr>
<tr>
<td>List with Details Panel View</td>
<td>oracle.webcenter.content.templates.default.list.details.panel</td>
<td>Displays multiple content items in a list on the left, with a panel to the right displaying the details of a selected item.</td>
</tr>
<tr>
<td>Sortable Table View</td>
<td>oracle.webcenter.content.templates.default.list.tabular</td>
<td>Displays multiple content items in a sortable table that includes the document name, date created, and date modified.</td>
</tr>
<tr>
<td>Tabbed View</td>
<td>oracle.webcenter.content.templates.default.list.panel.tabbed</td>
<td>Displays multiple content items as tabs that can be selected to display item details. Content items in a simple list.</td>
</tr>
</tbody>
</table>

### 29.7.2 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.
### Table 29–7 Document Explorer Task Flow Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>connectionName</td>
<td>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 the &quot;Registering Content Repositories&quot; chapter in <em>Administering Oracle WebCenter Portal</em>. 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.</td>
</tr>
<tr>
<td>featuresOff</td>
<td>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, multfile-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'}</td>
</tr>
</tbody>
</table>
| 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 want 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}: 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. |
29.7.3 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.

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 29–8 describes the Document List Viewer task flow parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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 portal's default root folder. But it is useful, for example, when you want the start folder to be other than a portal's default root folder and when you want to display content from another portal. 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 portal. |
| treeNavCollapsed | Specifies whether to collapse or expand the panel containing the tree navigation:  
  ■ $(true) (default): Collapse panel.  
  ■ $(false): Expand panel. |

---

### Table 29–7 (Cont.) Document Explorer Task Flow Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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 portal’s default root folder. But it is useful, for example, when you want the start folder to be other than a portal’s default root folder and when you want to display content from another portal. 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 portal. |
| treeNavCollapsed | Specifies whether to collapse or expand the panel containing the tree navigation:  
  ■ $(true) (default): Collapse panel.  
  ■ $(false): Expand panel. |
### Table 29–8 (Cont.) Document List Viewer Task Flow Parameters

<table>
<thead>
<tr>
<th>Task Flow Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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’)`

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’)`

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’)`

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’)`

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’)`

| 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: 15

Typical scenarios where you may want 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.
### Document Manager Task Flow Parameters

The Document Manager task flow provides comprehensive document management functionality, such as copying, moving, pasting, and deleting folders and files, as well as the ability to configure the task flow layout. The Document Manager task flow displays folders and files as specified by its `layout` parameter: it can 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`.

#### See Also:
- "Understanding the Document Manager Task Flow" in *Building Portals with Oracle WebCenter Portal*
- Section 29.5, "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 29–9 describes the Document Manager task flow parameters.

### Table 29–9  Document Manager Task Flow Parameters

<table>
<thead>
<tr>
<th>Task Flow Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>connectionName</td>
<td>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 the &quot;Registering Content Repositories&quot; chapter in <em>Administering Oracle WebCenter Portal</em>. 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.</td>
</tr>
<tr>
<td>featuresOff</td>
<td>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: <code>${search, advancedSearch, clipboard, dnd, rename, newfolder, upload, newwiki, checkin, checkout, editoffice, edithtml, delete, sidebars, history}</code></td>
</tr>
<tr>
<td>layout</td>
<td>A target layout for the task flow. Select from:</td>
</tr>
<tr>
<td></td>
<td>- <code>${explorer}</code> (default): Displays folders and files in two panes; the left pane shows folders, and the right pane shows 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 29–3.</td>
</tr>
<tr>
<td></td>
<td>- <code>${table}</code>: Displays only the contents of the current folder in a single pane, with the capability to click a folder to drill down, refreshing the pane with the folder contents. See Figure 29–4.</td>
</tr>
<tr>
<td></td>
<td>- <code>${treeTable}</code>: Displays the folder hierarchy in a single pane, beginning with the root folder, with the capability to expand and collapse folders. See Figure 29–5.</td>
</tr>
<tr>
<td>pageSize</td>
<td>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</td>
</tr>
<tr>
<td>readOnly</td>
<td>Specifies whether to disable and hide all content management operations:</td>
</tr>
<tr>
<td></td>
<td>- <code>${true}</code>: Disable content management.</td>
</tr>
<tr>
<td></td>
<td>- <code>${false}</code> (default): Expose content management to users.</td>
</tr>
</tbody>
</table>

Typical scenarios where you may want 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.
Table 29–9  (Cont.) Document Manager Task Flow Parameters

<table>
<thead>
<tr>
<th>Task Flow Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>connectionName</td>
<td>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 the “Registering Content Repositories” chapter in Administering 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.</td>
</tr>
<tr>
<td>resourceId</td>
<td>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.</td>
</tr>
</tbody>
</table>
| 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 portal's default root folder. But it is useful, for example, when you want the start folder to be other than a portal's default root folder and when you want to display content from another portal. 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 portal. |

29.7.5 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 menus options available to the end user for this task flow.

See Also:

- "Understanding the Document Navigator Task Flow" in Building Portals with Oracle WebCenter Portal
- Section 29.5, "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 29–10 describes the Document Navigator task flow parameters.

Table 29–10  Document Navigator Task Flow Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>connectionName</td>
<td>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 the “Registering Content Repositories” chapter in Administering 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.</td>
</tr>
</tbody>
</table>
Table 29–10 (Cont.) Document Navigator Task Flow Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>featuresOff</td>
<td>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, mult file-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'}'</td>
</tr>
<tr>
<td>pageSize</td>
<td>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 want 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.</td>
</tr>
<tr>
<td>readOnly</td>
<td>Specifies whether to disable and hide all content management operations: ■ ${true}: Disable content management. ■ ${false} (default): Expose content management to users.</td>
</tr>
<tr>
<td>resourceId</td>
<td>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.</td>
</tr>
<tr>
<td>showDocuments</td>
<td>Specifies whether the navigation tree shows documents and folders, or folders only: ■ ${true} (default): Show documents and folders. ■ ${false}: Show folders and hide documents.</td>
</tr>
</tbody>
</table>
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 *Building Portals with Oracle WebCenter Portal*
- Section 29.5, "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 29–11 describes the Folder Viewer task flow parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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 portal’s default root folder. But it is useful, for example, when you want the start folder to be other than a portal’s default root folder and when you want to display content from another portal. 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 portal. |

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>connectionName</td>
<td>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 the “Registering Content Repositories” chapter in <em>Administering Oracle WebCenter Portal</em>. 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.</td>
</tr>
</tbody>
</table>
Adding Content Task Flows and Document Components to a Portal Page

### Table 29–11 (Cont.) Folder Viewer Task Flow Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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: 

```javascript
${'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 want 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.
29.7.7 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 Building Portals with Oracle WebCenter Portal
- Section 29.5, "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 29–12 describes the Recent Documents task flow parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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 portal’s default root folder. But it is useful, for example, when you want the start folder to be other than a portal’s default root folder and when you want to display content from another portal. Example:  
- `${'/PersonalSpaces/monty/Public'}`  
- `${'/WebCenterB5/Proj_X/Specs'}`  
You can specify an EL expression to set this value. See Section G.9, "ELs Related to Documents." Default: The root folder of the content repository configured with the specified connection for the current portal. |

Table 29–12 Recent Documents Task Flow Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>connectionName</td>
<td>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 the &quot;Registering Content Repositories&quot; chapter in Administering 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.</td>
</tr>
</tbody>
</table>
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.

### Table 29–12  (Cont.) Recent Documents Task Flow Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>groupSpace</code></td>
<td>(Used in WebCenter Portal only) The name of a portal that is the source of the recently created or modified documents listed in the task flow. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>- no value or <code>${null}</code> (default): Displays documents for the entire connected content repository.</td>
</tr>
<tr>
<td></td>
<td>- Display name or ID of a specific portal: Displays documents in that portal.</td>
</tr>
<tr>
<td></td>
<td>- <code>${'all'}</code>: Displays documents from any portal, excluding non-portal documents.</td>
</tr>
</tbody>
</table>
| `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: `${'yyyy-mm-ddThh:mm:ss.sssTZD'}`
|               | Example: `${'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: `${'yyyy-mm-ddThh:mm:ss.sssTZD'}`
|               | Example: `${'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 "+0800".

### 29.7.8 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 Building Portals with Oracle WebCenter Portal
- Section 29.5, "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 29–13 describes the Document Viewer task flow parameters.

<table>
<thead>
<tr>
<th>Table 29–13</th>
<th>Document Viewer Task Flow Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Parameter</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>featuresOff</td>
<td>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'}</td>
</tr>
</tbody>
</table>
| initialSidebar | Specifies the tabbed pane to have initial focus in the Document Viewer. Valid values are:  
- ${'comments'} (default). Displays the Comments pane (see the "Commenting on Items" section in Using Oracle WebCenter Portal).  
- ${'tags'}. Displays the Tags pane (see the "Managing Tags on Files" section in Using Oracle WebCenter Portal).  
- ${'history'}. Displays the History pane (see the "Viewing and Deleting File Version History" section in Using Oracle WebCenter Portal).  
- ${'docInfo'}. Displays the Info pane (see the "Working with Folder and File Properties" section in Using Oracle WebCenter Portal).  
- ${'tags'}. Displays the Links pane (see the "Managing Document Links" section in Using Oracle WebCenter Portal).  
- ${'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 Oracle AutoVue" section in Using Oracle WebCenter Portal). |
| 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. |
29.7.9 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 Building Portals with Oracle WebCenter Portal
- Section 29.5, "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 29–14 describes the Document Mini Properties task flow parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>resourceID</td>
<td>The ID of the document for which to display basic properties.</td>
</tr>
</tbody>
</table>

29.7.10 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 Building Portals with Oracle WebCenter Portal
- Section 29.5, "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 29–15 describes the Document Properties task flow parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>resourceID</td>
<td>The ID of the document for which to display basic and advanced properties.</td>
</tr>
<tr>
<td>readOnly</td>
<td>Specifies whether or not to allow end user to edit document properties:</td>
</tr>
<tr>
<td></td>
<td>${true}: Do not display Edit button in Basic and Advanced properties panes,</td>
</tr>
<tr>
<td></td>
<td>thus disabling the ability for the end user to edit document properties.</td>
</tr>
<tr>
<td></td>
<td>${false} (default): Display Edit button in Basic and Advanced properties</td>
</tr>
<tr>
<td></td>
<td>panes, allowing end users to edit document properties.</td>
</tr>
</tbody>
</table>

29.7.11 Rich Text Editor Task Flow Parameters

The Rich Text Editor task flow displays an HTML or wiki document in the Rich Text Editor.
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 29–18 describes the Rich Text Editor task flow parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>destinationID</td>
<td>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.</td>
</tr>
<tr>
<td>featuresOff</td>
<td>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, 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, lls. Example: ${'search, advancedSearch, clipboard, dnd, rename, newfolder, upload, newwiki, checkin, checkout, editoffice, edithtml, delete, sidebars, history'}</td>
</tr>
<tr>
<td>initialSidebar</td>
<td>Specifies a pane to display in the sidebar of the Document Viewer. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>- ${'comments'} (default). Displays a Comments pane (see the &quot;Commenting on Items&quot; section in Using Oracle WebCenter Portal)</td>
</tr>
<tr>
<td></td>
<td>- ${'tags'}. Displays a Tags pane (see the &quot;Managing Tags on Files&quot; section in Using Oracle WebCenter Portal)</td>
</tr>
<tr>
<td></td>
<td>- ${'history'}. Displays a History pane (see the &quot;Viewing and Deleting File Version History&quot; section in Using Oracle WebCenter Portal)</td>
</tr>
<tr>
<td></td>
<td>- ${'docInfo'}. Displays a Properties pane (see the &quot;Working with Folder and File Properties&quot; section in Using Oracle WebCenter Portal)</td>
</tr>
<tr>
<td></td>
<td>- ${'links'}. Displays a Links pane (see the &quot;Managing Document Links&quot; section in Using Oracle WebCenter Portal)</td>
</tr>
<tr>
<td></td>
<td>- ${'recommendations'}. Displays a Recommendations pane (see the &quot;Viewing Document Recommendations&quot; section in Using Oracle WebCenter Portal)</td>
</tr>
</tbody>
</table>

**Note:** Values are case-sensitive.

### 29.7.12 Document Upload Task Flow Parameters

The Document Upload task flow displays the Upload dialog to allow users to upload new documents to the page.
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 29–17 describes the Document Upload task flow parameters.

Table 29–17  Document Upload Task Flow Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>destinationID</td>
<td>The currently focused resource. This value specifies a folder ID to upload a document into that folder.</td>
</tr>
<tr>
<td>destinationSelectionEnabled</td>
<td>Specifies whether or not the destination folder can be changed:</td>
</tr>
<tr>
<td></td>
<td> ${true}: Display destination folder as an input text box along with a Browse button.</td>
</tr>
<tr>
<td></td>
<td> ${false} (default): Display destination folder as read only text.</td>
</tr>
<tr>
<td>featuresOff</td>
<td>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'}</td>
</tr>
</tbody>
</table>

29.7.13  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:

- Section 29.5, "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 29–18 describes the Document Version History task flow parameters.

Table 29–18  Version History Task Flow Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>horizontalLayout</td>
<td>Specifies the layout orientation for the version history information:</td>
</tr>
<tr>
<td></td>
<td> ${vertical} (default): Vertical orientation (see Figure 29–13).</td>
</tr>
<tr>
<td></td>
<td> ${horizontal}: Horizontal orientation (see Figure 29–13).</td>
</tr>
<tr>
<td>resourceID</td>
<td>The ID of the document for which to display version history.</td>
</tr>
</tbody>
</table>
The AutoVue task flow integrates the Oracle AutoVue functionality to allow users to review and collaborate on a document.

**Table 29–19  AutoVue Task Flow Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>readOnly</td>
<td>Specifies whether to disable and hide all content management operations:</td>
</tr>
<tr>
<td></td>
<td>${true}: Disable content management.</td>
</tr>
<tr>
<td></td>
<td>${false} (default): Expose content management to users.</td>
</tr>
</tbody>
</table>

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 29–19** describes the AutoVue task flow parameters.
This chapter describes how to integrate the wiki and blog functionality into your WebCenter Portal Framework applications to enable your application end users to collaborate and share content and ideas.

This chapter includes the following topics:

- Section 30.1, "Introduction to Wikis and Blogs"
- Section 30.2, "Requirements for Wikis and Blogs"
- Section 30.3, "Integrating Wikis"
- Section 30.4, "Integrating Blogs"

30.1 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. They 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 Portal Framework applications through the Documents tool. There is no limitation on the number of wiki pages or blogs that you can expose in an application.

30.2 Requirements for Wikis and Blogs

To support the wiki and blog functionality, the Documents tool relies on WebCenter Content. For wiki and blog functionality to be available in your Portal Framework applications, the following prerequisites must be met:

- WebCenter Content 11g must be installed and configured as the default content repository for WebCenter Portal. Wiki and blog functionality is not available with WebCenter Content 10g. For information about installing and configuring WebCenter Content 11g, see the "Managing Content Repositories" chapter in *Administering Oracle WebCenter Portal*. 
A connection to WebCenter Content 11g must be established. For information, see Section 25.2.1, "Creating a Content Repository Connection Based on the Oracle Content Server Adapter."

Complete the steps covered in Section 4.2, "Preparing Your Framework Application for Tools and 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. End 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 tool to add wikis and blogs to your Portal Framework applications:

- To add wikis to your application, you will need to add a documents task flow that provides the Add Wiki Document action (such as the Document Explorer task flow). See Section 30.3, "Integrating Wikis."
- To add blogs to your application, you use the blog tasks flows. See Section 30.4, "Integrating Blogs."

If you want to convert wikis and blogs in your Portal Framework applications into PDF, your application administrator must configure the WebCenter Conversion component, as described in the "Enabling the Conversion of Wikis and Blogs into PDFs" section in Administering Oracle WebCenter Portal.

### 30.3 Integrating Wikis

This following sections provide the information you need to add wiki documents to your Portal Framework applications:

- Section 30.3.1, "About Integrating Wikis"
- Section 30.3.2, "Adding the Document Explorer Task Flow to Provide the Wiki Functionality"
- Section 30.3.3, "Modifying Document Explorer Task Flow Parameters"
- Section 30.3.4, "What Happens at Runtime"
- Section 30.3.5, "Displaying Wiki Page Links Within Content Presenter"

#### 30.3.1 About Integrating Wikis

To provide your end users with the ability to add and manage wiki documents at runtime, you need to add the Document Explorer task flow to your Portal 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 29.7.2, "Document Explorer Task Flow Parameters."

#### 30.3.2 Adding the Document Explorer Task Flow to Provide the Wiki Functionality

To prepare your Portal 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 30.2, "Requirements for Wikis and Blogs."
2. Create or open the JSF page on which you want to provide the wiki functionality (see Section 15.2, "Creating Pages in a WebCenter Portal Framework Application").

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 29.4, "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 29.5, "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 29.7.2, "Document Explorer Task Flow Parameters."

When you run the page, the Document Explorer task flow displays in your default browser, as shown in Figure 30–1. End users can create a wiki document by clicking **New Wiki Document**. For more information, see Section 30.3.4, "What Happens at Runtime."

![Figure 30–1  Creating a New Wiki Document](image)

### 30.3.3 Modifying Document Explorer Task Flow Parameters

The way you modify parameters of a Document Explorer task flow is the same as any other documents task flow. For information, see Section 29.6, "Modifying Content Task Flow Parameters." For information about the specific parameters that you can modify, see Section 29.7.2, "Document Explorer Task Flow Parameters."

### 30.3.4 What Happens at Runtime

At runtime, authorized users can create wiki documents in a Portal 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. End users can create a wiki document by clicking **New Wiki Document** (Figure 30–2), which opens the Rich Text Editor (RTE) (Figure 30–3), displaying a blank wiki document.

**Figure 30–2 Creating a New Wiki Document**

![Figure 30–2 Creating a New Wiki Document](image)

Users can add and edit wiki content using the Rich Text Editor (RTE), which is a fully-integrated HTML text editor (Figure 30–3).

**Figure 30–3 Rich Text Editor Displaying New Wiki Document**

![Figure 30–3 Rich Text Editor Displaying New Wiki Document](image)

**Note:** By default, the **Wiki Markup** tab is hidden in the RTE. To show and hide the **Wiki Markup** tab, your system administrator can edit a configuration file, as described in the "Showing and Hiding the Wiki Markup Tab in the Rich Text Editor" section in *Administering Oracle WebCenter Portal*.

By default, all wiki documents are created directly under the root folder (Figure 30–4).
The runtime procedures for adding and managing wiki documents in a Portal Framework application are the same as those in WebCenter Portal. For generic information, refer to the "Working with Wikis" chapter in *Using Oracle WebCenter Portal*.

### 30.3.5 Displaying Wiki Page Links Within Content Presenter

At runtime, users can choose to display an existing wiki document using Content Presenter, as described in the "Adding a Selected Folder or File to a Page" section in *Building Portals with Oracle WebCenter Portal*.

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 link targets within Content Presenter.

To open the targets of links present in a wiki document within Content Presenter rather than the Document Viewer:

1. Open the `adf-config.xml` file. This file is located under `Descriptors/ADF META-INF` in the *Application Resources* section of 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:
   
   ```xml
   <wpsC:adf-service-config
   xmlns="http://xmlns.oracle.com/webcenter/framework/service">
   <resource-handler
   </wpsC:adf-service-config>
   ```

4. Save `adf-config.xml`.

### 30.4 Integrating Blogs

When you integrate the blogs functionality into your Portal Framework application, authorized end users can create and manage blog posts. To enable the blog functionality, you need to add the blog task flows.

This following sections provide the information you need to add blogs to a Portal Framework application:

- Section 30.4.1, "Understanding the Blog Task Flows"
- Section 30.4.2, "Adding a Blog to a Page"
30.4.1 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 30–1 lists and describes the blog task flows:

Table 30–1 Blog Task Flows

<table>
<thead>
<tr>
<th>Blog Task Flow</th>
<th>Description</th>
<th>Example as Exposed on a Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blog Archives</td>
<td>Displays a composite list of blogs based on dates.</td>
<td><img src="image" alt="Archives" /></td>
</tr>
<tr>
<td></td>
<td></td>
<td>![2013 (2)]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>![June (2)]</td>
</tr>
<tr>
<td>Blog Banner</td>
<td>Displays a banner for the blog.</td>
<td><img src="image" alt="MyBlog" /></td>
</tr>
<tr>
<td>Blog Digest</td>
<td>Displays a blog or blog post.</td>
<td><img src="image" alt="New Post" /></td>
</tr>
<tr>
<td>Recent Blog Posts</td>
<td>Displays a list of most recent blog posts.</td>
<td><img src="image" alt="Recent Posts" /></td>
</tr>
<tr>
<td></td>
<td></td>
<td><img src="image" alt="Philatelists Club?" /></td>
</tr>
<tr>
<td></td>
<td>If you are interested in joining the online Philatelists Club, check out my website: <a href="http://www.arisite/philatclub">www.arisite/philatclub</a></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Created on 6/17/13 8:04 PM by Ari Spie - 0 comments - 0</td>
<td></td>
</tr>
</tbody>
</table>
To add a blog to a page in your Portal Framework application:

1. Ensure that you have completed the requirements listed in Section 30.2, "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 WebCenter Content.

   If you choose external application, you must specify the external application you want to use for authenticating users to WebCenter Content.

   For information, see Chapter 4.2.1.1, "Implementing Security for Tools and Services."

2. Create or open a page in your application where you want to expose the blog functionality (see Section 15.2, "Creating Pages in a WebCenter Portal Framework Application").

3. Before you add a blog task flow to your page, consider whether or not you want to allow your end users to edit the task flow. To enable runtime editing of the page, see Chapter 18, "Enabling Runtime Editing of Pages Using Composer."

4. Add a blog to the page using an existing folder to create a default blog page, or by creating a custom blog by selecting component blog task flows:

   - Section 30.4.2.1, "Adding a Blog Using an Existing Folder"
   - Section 30.4.2.2, "Adding a Custom Blog to a Page Using the Blog Task Flows"

### 30.4.2.1 Adding a Blog Using an Existing Folder

If you have an existing folder, perhaps already containing blog posts, you can establish that folder as the source of blog posts for a blog, instead of creating a new "empty" blog (see Section 30.4.2.2, "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 WebCenter Content 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 a documents task flow or the WebCenter Content console.

   Files stored in the blog folder are identified as blog posts in their metadata Type field, which is set to Blog Post. Uploading an HTML file into a blog folder does not expose it as a blog post, unless you set its Type to Blog Post.

3. Drag the folder onto the page, and release the mouse button to display a menu listing all available task flows (Figure 30–5).

   **Figure 30–5 Menu of Available Task Flows for Selected Folder**

4. From the menu, choose Blogs to open the Edit Task Flow Binding dialog (Figure 30–6).
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 30.4.5.5, "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 Post. As posts are added to the blog at runtime, the blog posts are stored in this folder. See Section 30.4.3, "What Happens at Runtime."

30.4.2.2 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 30.4.2.1, "Adding a Blog Using an Existing 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 release the mouse button.

   For information about each of the five blog task flows, see Section 30.4.1, "Understanding the Blog Task Flows."

2. From the menu that displays, choose Region (Figure 30–7).
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 30.4.5, "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 Posts. As posts are added to the blog at runtime, the blog posts are stored in this folder. See Section 30.4.3, "What Happens at Runtime."

### 30.4.3 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 30.4.2.1, "Adding a Blog Using an Existing 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 30.4.2.2, "Adding a Custom Blog to a Page Using the Blog Task Flows," then the name of the folder that you specified 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 that you specified 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 30–8), 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.
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 blog page (Figure 30–9), providing controls to manage the blog post.

When public users (not logged in) view a blog post in a Portal Framework application, they will not see the profile photos associated with each blog post because the anonymous-role that is assigned to public users is not granted permission to view profiles by default (see Table 74–2, "Automated Security Grants for Portal Framework Applications"). Your system administrator can use the grantPermission WLST command (see the "grantPermission" section in the WebLogic Scripting Tool Command Reference) to allow public users to view profile photos for blog posters. For example:

```bash
grantPermission(appStripe=<app-name>,
    principalClass="oracle.security.jps.internal.core.principals.JpsAnonymousRoleImpl",
    principalName="anonymous-role",
)```
The procedures for managing blogs at runtime in a Portal Framework application are the same as those for managing blogs in WebCenter Portal. For information, see the "Working with Blogs" chapter in Using Oracle WebCenter Portal.

30.4.4 Modifying Blog Task Flow Parameters

The procedure for modifying the parameters of a Blogs task flow is the same as any other documents task flow. For information, see Section 29.6, "Modifying Content Task Flow Parameters." For information about the specific parameters that you can modify, see Section 30.4.5, "Blog Task Flow Parameters."

30.4.5 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 30.4.5.1, "Blog Archives Task Flow Parameters"
- Section 30.4.5.2, "Blog Banner Task Flow Parameters"
- Section 30.4.5.3, "Blog Digest Task Flow Parameters"
- Section 30.4.5.4, "Blog Recent Posts Task Flow Parameters"
- Section 30.4.5.5, "Blogs Task Flow Parameters"

30.4.5.1 Blog Archives Task Flow Parameters

The Blog Archives task flow displays a composite list of blogs based on dates. See Also: Section 30.4.1, "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 30–2 describes the Blog Archives task flow parameters.

<table>
<thead>
<tr>
<th>Table 30–2 Blog Archives Task Flow Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>resourceID</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
| | connection_name#dCollectionId:
| | where connection_name is the name of the WebCenter Content connection, and dCollectionId is the collection ID of the folder on WebCenter Content that you want to expose as a blog. |
30.4.5.2 Blog Banner Task Flow Parameters

The Blog Banner task flow task flow displays a banner for the blog.

See Also: Section 30.4.1, "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 30–3 describes the Blog Banner task flow parameters.

Table 30–3 Blog Banner Task Flow Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| resourceId | The resource ID of the blog folder (see Figure 30–6), which can be specified in the following formats:  
  - connection_name/path_to_folder  
    where connection_name is the name of the WebCenter Content connection, and path_to_folder is the path to the folder on WebCenter Content that you want to expose as a blog.  
  - connection_name#dCollectionID:dCollectionId  
    where connection_name is the name of the WebCenter Content connection, and dCollectionId is the collection ID of the folder on WebCenter Content that you want to expose as a blog. |
| 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. |

30.4.5.3 Blog Digest Task Flow Parameters

The Blog Digest task flow displays a blog or blog post.

See Also: Section 30.4.1, "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 30–4 describes the Blog Digest task flow parameters.
Table 30–4  Blog Digest Task Flow Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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. A folder can be specified in the following formats:  
  ■ connection_name/path_to_folder  
  where connection_name is the name of the WebCenter Content connection, and path_to_folder is the path to the folder on WebCenter Content that you want to expose as a blog.  
  ■ connection_name#dCollectionID:dCollectionId  
  where connection_name is the name of the WebCenter Content connection, and dCollectionId is the collection ID of the folder on WebCenter Content that you want to expose as a blog.  
  Note: To allow users to add new blog posts (by clicking New Post in the task flow), the specified folder must have a security group assigned in WebCenter Content. |
| year        | A four-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.  
  ■ Deselected (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 |

30.4.5.4 Blog Recent Posts Task Flow Parameters

The Blog Recent Posts task flow displays a list of most recent blog posts.

See Also: Section 30.4.1, "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 30–5 describes the Blog Recent Posts task flow parameters.
### Table 30–5 Blog Recent Posts Task Flow Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>resourceId</td>
<td>The resource ID of the blog folder (see Figure 30–6), which can be specified in the following formats:</td>
</tr>
<tr>
<td></td>
<td>- connection_name/path_to_folder</td>
</tr>
<tr>
<td></td>
<td>where connection_name is the name of the WebCenter Content connection, and path_to_folder is the path to the folder on WebCenter Content that you want to expose as a blog.</td>
</tr>
<tr>
<td></td>
<td>- connection_name#dCollectionId:dCollectionId</td>
</tr>
<tr>
<td></td>
<td>where connection_name is the name of the WebCenter Content connection, and dCollectionId is the collection ID of the folder on WebCenter Content that you want to expose as a blog.</td>
</tr>
<tr>
<td>recentPostListSize</td>
<td>The number of recent posts to display.</td>
</tr>
<tr>
<td></td>
<td>Default: 10</td>
</tr>
</tbody>
</table>

### 30.4.5.5 Blogs Task Flow Parameters

The Blogs task flow displays a blog or blog post with a default design.

**See Also:** Section 30.4.1, "Understanding the Blog Task Flows"

Parameters that are unique to the Blogs task flow are shown in the Edit Task Flow Binding dialog box when you add the task flow to a page. Table 30–6 describes the Blogs task flow parameters.

### Table 30–6 Blogs Task Flow Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>resourceId</td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td>A folder can be specified in the following formats:</td>
</tr>
<tr>
<td></td>
<td>- connection_name/path_to_folder</td>
</tr>
<tr>
<td></td>
<td>where connection_name is the name of the WebCenter Content connection, and path_to_folder is the path to the folder on WebCenter Content that you want to expose as a blog.</td>
</tr>
<tr>
<td></td>
<td>- connection_name#dCollectionId:dCollectionId</td>
</tr>
<tr>
<td></td>
<td>where connection_name is the name of the WebCenter Content connection, and dCollectionId is the collection ID of the folder on WebCenter Content that you want to expose as a blog.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> To allow users to add new blog posts (by clicking New Post in the task flow), the specified folder must have a security group assigned in WebCenter Content.</td>
</tr>
<tr>
<td>hideComments</td>
<td>Specifies whether the Comments feature is exposed:</td>
</tr>
<tr>
<td></td>
<td>- Selected: Hide the Comments link and pane.</td>
</tr>
<tr>
<td></td>
<td>- Deselected (default): Show the Comments link and pane.</td>
</tr>
<tr>
<td>pageSize</td>
<td>The number of blog posts displayed in the Blogs task flow before the Next and Previous icons are enabled.</td>
</tr>
<tr>
<td></td>
<td>Default: 10</td>
</tr>
</tbody>
</table>
This chapter supplements the OASIS CMIS specification, and provides details on the specific implementation of the Content Management REST Service.

The OASIS CMIS (Content Management Interoperability Services) Technical Committee works to standardize a Web services interface specification that will enable greater interoperability of Enterprise Content Management (ECM) systems. For more information, see the OASIS CMIS site:

http://www.oasis-open.org/committees/cmis/

Before continuing, all users should review the OASIS CMIS specification. This chapter references the Content Management Interoperability Services (CMIS) Version 1.0, which can be viewed at the following URL:

http://docs.oasis-open.org/cmis/CMIS/v1.0/cmis-spec-v1.0.html.

The Content Management REST Service provides a server that uses the CMIS RESTful AtomPub server binding to provide access to Oracle Content Server repositories configured in your application.

The specification includes the domain model and two server bindings. As mentioned above, only the RESTful AtomPub binding is currently implemented by the Content Management REST Service. Users should be familiar with Atom and AtomPub, which are the default formats for responses.

---

**Note:** CMIS provides a lowest common denominator for a wide range of different content systems; it is not aligned directly with Oracle WebCenter Content functionality. Refer to the Content Management REST Service service document to identify available functionality.

---

This chapter includes the following sections:

- Section 31.1, "CMIS Domain Model"
- Section 31.2, "CMIS Part II: RESTful AtomPub Binding"
- Section 31.3, "Content Management REST Service Best Practices and Examples"

### 31.1 CMIS Domain Model

*Section 2: Domain Model* in the CMIS specification defines a domain model that can be used by applications to work with one or more Content Management repositories.

- Section 31.1.1, "Data Model"
Section 31.2, "Services"

31.1 Data Model

The Content Management REST Service service document consists of AtomPub workspaces. Each workspace maps to a content connection (only Oracle WebCenter Content repositories are supported by the Content Management REST Service). For details on the service document, see the next section, Section 31.2, "CMIS Part II: RESTful AtomPub Binding".

31.1.1 Repository

For this release, some of the optional capabilities listed in section 2.1.1 have not been implemented. Versioning, ACL, Policies, Relationships, Change Log, Folder Descendants/Tree, and Renditions will be considered for future releases.

Specifically, the Content Management REST Service implementation has the following optional capabilities:

capabilityGetDescendants = true
capabilityGetFolderTree = false
capabilityContentStreamUpdatability = anytime
capabilityChanges = none
capabilityRenditions = none
capabilityMultifiling = false
capabilityUnfiling = false
capabilityVersionSpecificFiling = false
capabilityPWCUpdateable = false
capabilityPWCSearchable = false
capabilityAllVersionsSearchable = false
capabilityJoin = none
capabilityACL = none
capabilityQuery = none, metadataonly, or both combined

31.1.2 Object

The Content Management REST Service supports document and folder objects. In CMIS the cmis:baseTypeId for a Node will be cmis:folder or cmis:document. Also, the cmis:baseId for a Type will be cmis:folder or cmis:document.

31.1.3 Object-Type

A CMIS Object-Type contains fields mapped from the Oracle WebCenter Content: WebCenter Content metadata field definitions and Oracle WebCenter Content's Site Studio region definitions.

The mapping from Oracle WebCenter Content metadata fields to CMIS property definitions is as follows:

<table>
<thead>
<tr>
<th>Oracle WebCenter Content Metadata</th>
<th>CMIS Property Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>TEXT metadata field with option list configured with select list validated and YesNoView or TrueFalseView view</td>
<td>cmis:propertyBoolean</td>
</tr>
<tr>
<td>All other TEXT metadata fields</td>
<td>cmis:propertyString</td>
</tr>
<tr>
<td>LONG TEXT metadata field</td>
<td>cmis:propertyString</td>
</tr>
<tr>
<td>MEMO metadata field</td>
<td>cmis:propertyString</td>
</tr>
<tr>
<td>INTEGER metadata field</td>
<td>cmis:propertyInteger</td>
</tr>
</tbody>
</table>
The mapping from Site Studio region definition fields to CMIS property definitions is as follows:

### Table 31–2  Site Studio Region Definition Mappings

<table>
<thead>
<tr>
<th>Site Studio Region Definition</th>
<th>CMIS Property Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image Element Definition fields</td>
<td>cmis:propertyString</td>
</tr>
<tr>
<td>WYSIWYG Element Definition fields</td>
<td>cmis:propertyString</td>
</tr>
<tr>
<td>Plain Text Element Definition fields</td>
<td>cmis:propertyString</td>
</tr>
<tr>
<td>Static List Element Definition fields</td>
<td>cmis:propertyString</td>
</tr>
</tbody>
</table>

### 31.1.4 Document Object

Document Objects are the elementary information entities managed by the repository. As defined by the CMIS specification, Document Objects may be version-able, file-able, query-able, control-able and ACLControl-able. As stated earlier, the Content Management REST Service does not support versioning, multi-filing, Policies or ACL for this release.

If a Node is determined to be a Document (not a Folder) then any children it has will not be exposed through CMIS. In CMIS, each Document Object is associated with a single content stream, and for WebCenter CMIS REST, this stream is the Oracle WebCenter Content: WebCenter Content binary associated with the document.

### 31.1.5 Folder Object

The CMIS specification states that Folder Objects do not have a content-stream and are not version-able. If a Node is determined to be a Folder, then the Content Management REST Service exposes it in this manner. (In Oracle WebCenter Content, folders do not have a content stream and are not versionable).

### 31.1.6 Relationship Object

Section 2.1.6: Relationship Object in the CMIS specification is not applicable; the Content Management REST Service does not support Relationships for this release.

### 31.1.7 Policy Object

Section 2.1.7: Policy Object in the CMIS specification is not applicable; the Content Management REST Service does not support Policies for this release.

### 31.1.8 Access Control

Most of Section 2.1.8: Access Control in the CMIS specification is not applicable; the Content Management REST Service does not support ACL for this release. See the next section for details on allowable actions.

### 31.1.8.1 AllowableActions Mapping

This section lists allowable actions defined for Objects. Because of how this release is implemented, some of these are hard-coded for all objects. Other allowable actions will be set based on the repository configuration.
canGetObjectRelationships = false
canCreateRelationship = false
canGetDescendants = false
canGetFolderTree = false
canCheckOut = false (versioning)
canCancelCheckOut = false (versioning)
canCheckIn = false (versioning)
canAddObjectToFolder = false (multi-filing)
canRemoveObjectFromFolder = false (unfiling/multi-filing)
canApplyPolicy = false
canGetAppliedPolicies = false
canRemovePolicy = false
canCreatePolicy = false
canApplyACL = false
canGetACL = false
canGetRenditions = false
canDeleteTree = true
canGetAllVersions = false (versioning)

31.1.1.9 Versioning

Section 2.1.9: Versioning in the CMIS specification is not applicable; the Content Management REST Service does not support versioning for this release.

31.1.1.10 Query

CMIS queries return a Result Set where each Entry object will contain only the properties that were specified in the query. The Content Management REST Service does not support JOINs in queries, so each result entry will represent properties from a single node. Common searches use a query like "SELECT * FROM ...".

- The FROM clause specifies a content-type to be searched.
  - FROM cmis:document ==> any Oracle WebCenter Content document (for example, IDC:GlobalProfile)
  - FROM cmis:folder ==> any Oracle WebCenter Content folder (for example, IDC:Folder)
  - FROM typeQueryName ==> type's cmis queryName, as long as the type is queryable (for example, ora:t:IDC!;GlobalProfile)

- The cmis:document and cmis:folder types are always queryable. Other types will be queryable if they are searchable in the repository.

- The IN_FOLDER predicate is implemented as the folder ID specified, being the parent of the results.

- The IN_TREE predicate is implemented as the folder ID specified, being a parent in the folder structure of the results.

- The CONTAINS() predicate is a full-text query expression operator.
Properties of cmis:document and cmis:folder will be queryable and orderable if their corresponding Oracle WebCenter Content system property is searchable and sortable. The system property mappings are:

- cmis:createdBy ==> dDocAuthor
- cmis:lastModifiedBy ==> dDocCreator
- cmis:creationDate ==> dCreateDate
- cmis:lastModificationDate ==> dLastModifiedDate (for 10g, folders map to dLastModifiedDate and documents map to dCreateDate)
- cmis:name ==> dOriginalName (for a document) or dCollectionName (for a folder)
- cmis:contentStreamFileName ==> dOriginalName
- cmis:contentStreamLength ==> VaultFileSize
- cmis:contentStreamMimeType ==> dFormat
- cmis:objectId ==> dDocName
- cmis:objectTypeId ==> Oracle WebCenter Content profile name or SiteStudio Region Definition name

Note: cmis:objectTypeId is never orderable.

- cmis:path ==> use IN_FOLDER or IN_TREE predicate

Note: Some repositories may have capabilities that are not representable in a CMIS query, and some repositories may have restrictions which will limit the CMIS-query predicates (or combinations of predicates) that can be used in a query. Use the mappings above to translate repository capabilities and restrictions into corresponding considerations for CMIS queries.

Nested properties are not queryable or orderable.

The Content Management REST Service implementation reports as orderable any properties which Oracle WebCenter Content specifies as sortable. This list can include properties which Oracle WebCenter Content cannot actually sort on. To allow ordering on a field for which Oracle WebCenter Content reports a sort error, follow the steps below to make the specified Oracle WebCenter Content field sortable:

1. Go to Administration and open Admin Applets.
2. Open the Configuration Manager applet and click Advanced Search Design.
3. Edit the field you wish to make orderable and select Is sortable.
4. Save your changes and exit Administration.

The table that follows lists specific search considerations and recommendations. For example queries, see Section 31.3, "Content Management REST Service Best Practices and Examples."
## Table 31–3  Search Considerations and Recommendations

<table>
<thead>
<tr>
<th>Consideration</th>
<th>Recommendation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle WebCenter Content provides limited support for querying on null or non-null values.</td>
<td>Be aware of the differences in search behavior and do not write search expressions that depend on unsupported criteria.</td>
</tr>
<tr>
<td>Recursive search for folders is not supported by Oracle WebCenter Content, but is supported for documents if configured on the Oracle WebCenter Content server as described to the right.</td>
<td>Scope the search to only include documents (add a select clause like select * from cmis:document).</td>
</tr>
<tr>
<td>Recursive search for folders is not supported by Oracle WebCenter Content, but is supported for documents if configured on the Oracle WebCenter Content server as described to the right.</td>
<td>Set the search path on the Search object (add a where clause like &quot;where IN_TREE('/StellentRepository/IDC:Folder/2').”</td>
</tr>
<tr>
<td>Recursive search for folders is not supported by Oracle WebCenter Content, but is supported for documents if configured on the Oracle WebCenter Content server as described to the right.</td>
<td>Configure the folders_g CollectiveSearchRecursiveContent and related settings like CollectionMaxBranch in the Oracle WebCenter Content config.cfg file.</td>
</tr>
<tr>
<td>Multivalued property operators perform substring matches. This is true for ANY &lt;multiValuedQueryName&gt; IN (&lt;literal&gt;, ...) or &lt;literal&gt; = ANY &lt;multiValuedQueryName&gt;. In Oracle WebCenter Content, a field with an option list stores values in a comma-delimited manner. For example, if you have values &quot;A&quot;, &quot;B&quot;, and &quot;C&quot;, these will be represented as &quot;A, B, C&quot;. Using an ANY OR ANY IN search for 'A, B' will find this item.</td>
<td>Be aware of the differences in search behavior and consider changing the Oracle WebCenter Content option list delimiter character in the Configuration Manager applet to reduce the potential for finding extra matches.</td>
</tr>
<tr>
<td>When searching folders (FROM cmis:folder), at most one value can be specified per criteria. Each criteria is logically ANDed with the others to make a more selective query. There is no support for OR or NOT operators when searching folders.</td>
<td>Do not use OR and NOT in Oracle WebCenter Content folder search.</td>
</tr>
<tr>
<td>Not all properties are searchable, and if the search encounters a property that is not searchable, it will return a ParseException (400 error).</td>
<td>Understand which properties are searchable for a given content type by examining the Oracle WebCenter Content Configuration Manager information fields section, or by reviewing the ContentType definition. Examine the URL for the isSearchable field setting.</td>
</tr>
<tr>
<td>Not all ContentTypes are searchable. An attempt to search for a non-searchable ContentType will throw an exception. For example, the IDC:FileReference ContentType is not searchable.</td>
<td>Be aware that not all ContentTypes are searchable.</td>
</tr>
<tr>
<td>Only String multi-valued properties can be searched.</td>
<td>Do not specify search for multi-valued property types other than String.</td>
</tr>
<tr>
<td>The NOT operator cannot be used for LONG properties.</td>
<td>Try to restructure the query using supported syntax.</td>
</tr>
<tr>
<td>Only one sort criteria is supported.</td>
<td>Do not write search expressions that use more than a single sort criteria.</td>
</tr>
<tr>
<td>Consideration</td>
<td>Recommendation</td>
</tr>
<tr>
<td>------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Sorting on non-indexed fields results in an exception.</td>
<td>Understand which fields have been indexed before using them as sort criteria. Examine the URL for the isSortable field definition.</td>
</tr>
<tr>
<td>Searching on a non-indexed field throws an exception, with the embedded exception code of, for example, &quot;DRG-10837: section dStatus does not exist&quot;.</td>
<td>Example URL: <a href="http://myContentServer/idc/idcplg?IdcService=GET_ADVANCED_SEARCH_OPTIONS&amp;IsSoap=1">http://myContentServer/idc/idcplg?IdcService=GET_ADVANCED_SEARCH_OPTIONS&amp;IsSoap=1</a>, Or examine the type through CMIS to see if the property definition is queryable.</td>
</tr>
<tr>
<td>Empty values are not allowed in a search query.</td>
<td>Do not use a criteria such as cmis:name != ''.</td>
</tr>
<tr>
<td>The Notequals operator is not supported for non-String properties</td>
<td>Be aware of the differences in search behavior and do not write search expressions that depend on unsupported criteria.</td>
</tr>
<tr>
<td>Multiple search paths on the same Oracle WebCenter Content repository are not supported.</td>
<td>Be aware of the differences in search behavior and do not write search expressions that depend on unsupported criteria.</td>
</tr>
<tr>
<td>When searching for documents, recursive search (folder tree search) is supported if Oracle WebCenter Content is configured properly. If the search path is not set, then all documents in the repository will be searched (both filed and unfiled).</td>
<td>Configure the content server folders_g CollectionSearchRecursiveContent and related settings like CollectionMaxBranch in the Oracle WebCenter Content config.cfg file. These settings are described in the Oracle WebCenter Content documentation. To perform a document search scoped to a folder tree, use the IN_TREE predicate.</td>
</tr>
<tr>
<td>When searching for documents and using the LIKE operator, wildcards (%) are only supported in the last path element.</td>
<td>Be aware of the differences in search behavior and do not write search expressions that depend upon unsupported criteria.</td>
</tr>
<tr>
<td>When searching documents (select * from cmis:document), it is not possible to limit the search to more than just a single content type; for example, this is not supported: select * from IDC:MyProfile, IDC:AnotherProfile because it has multiple explicit content types and JOINS are not supported.</td>
<td>If you need to limit the search to more than just a single content type, issue multiple queries to achieve the same behavior. If you want to search across all types, use cmis:document in the select statement.</td>
</tr>
<tr>
<td>Oracle WebCenter Content search does not support OR when cmis:objectTypeId is specified in a query; other parameters can be ANDed with this criteria, but OR is not supported.</td>
<td>If this functionality is necessary, issue multiple queries to achieve the same behavior.</td>
</tr>
<tr>
<td>The cmis:objectTypeId criteria only supports ==, !=, and like. Use of != is restricted to the case of excluding folders (which behaves the same as adding select * from cmis:document).</td>
<td>Be aware of the valid operators when using cmis:objectTypeId criteria.</td>
</tr>
<tr>
<td>Queries may be case-sensitive depending on the selected Oracle WebCenter Content search engine.</td>
<td>Be aware that the Oracle WebCenter Content search engine selection can affect case-sensitivity. Metadata searches using OracleTextSearch as the search engine are generally case-insensitive. Metadata searches using DATABASE.FULLTEXT as the search engine are generally case-sensitive. The exact behavior sometimes varies by metadata field.</td>
</tr>
</tbody>
</table>
31.2 CMIS Part II: RESTful AtomPub Binding

Section 3: RESTful AtomPub Binding in the CMIS specification defines a specification based on AtomPub that can be used by applications to work with one or more Content Management Repositories. REST services are available through a portal instance.

31.2.1 Service Document

All navigation of a repository begins with the AtomPub Service Document. From this document, all accessible content in a repository can be discovered through the collections, links, and templates.

The URI to the service document, relative to the CMIS web application's context-root, is /rest/cmis/repository.

Therefore, if an application is deployed with a library-context-root-override as in the example above, the service document would be accessed through the following URL:

http://hostname:port/rest/cmis/repository

---

**Note:** The REST application is only available with portals and not with Portal Framework applications. However, you can make HTTP requests from your Framework application to access the REST and CMIS resources that are available in a portal. You must make sure that the portal instance is configured with the relevant content connections that you wish to access.

By default, this document will contain a workspace for each configured Oracle WebCenter Content repository (only Oracle WebCenter Content repositories are supported by CMIS REST in Oracle WebCenter Portal). A service document for a single repository can be obtained by using the repositoryId query parameter, as described in section 3.6.2.1 of the CMIS AtomPub binding specification.

As noted in the previous section, the service document consists of AtomPub workspaces. Each workspace maps to a Oracle WebCenter Portal WebCenter Content connection.

Specific URIs beyond the service document are not published; it is assumed that users will start at the service document and navigate the collections and links down expected paths. The relationships of the links and the titles and types of the collections are all defined in the CMIS specification, and thus can be commonly navigated by a client implementation. There are also templates defined for each repository, for easier access of objects by path, object by ID, type by ID, and queries. The format of the variables for the path and ID templates can be discovered by viewing the Entries of Folders and Documents.

31.2.2 Response Formats

Section 3.1.3: Response Formats in the CMIS specification indicates that Atom/AtomPub style formats will be returned by default unless overridden by a supported media type expressed in the Accept header.
A generic AtomPub feed reader can walk through any of the feeds returned by the CMIS REST server. It will not see all the CMIS specifics, but will be able to navigate through links. In general, to set up a feed reader, you need to know the URI of a particular feed, which can be discovered by navigating through the service document, for example, the workspace link for "typesdescendants".

For details on query syntax, see the CMIS specification. For Content Management REST Service best practices and examples, see Section 31.3, "Content Management REST Service Best Practices and Examples."

31.2.3 Additional Functionality

The Content Management REST Service provides the following additional functionality beyond the CMIS specification.

- Section 31.2.3.1, "Folder Children Collection"
- Section 31.2.3.2, "Document Entry"
- Section 31.2.3.3, "Content Stream"

31.2.3.1 Folder Children Collection

The specification defines the following CMIS services:

- GET: getChildren
- POST: createDocument
  or createFolder
  or createPolicy
  or moveObject
  or addObjectToFolder

The Content Management REST Service provides the following additional services:

- POST: create
  This service has five query parameters: uid, fileName, contentId, comments, simpleResponse, and one header parameter: Slug. This service is meant to be used as a simple binary request upload. A new document is created with this service. Slug and fileName are used to name the binary that is attached to the request (using both parameters is optional; only one needs to be defined and fileName is checked first). The comments parameter is optional and contentId is optional if Oracle WebCenter Content is set up to auto-generate the dDocName.

- POST: create Content-Type: multipart/form-data
  This service has a single query parameter: uid, which is the uid of the folder in which the document is to be created. The boolean query parameter simpleResponse will return a response of media type: application/atom+xml;type=entry, if set to false. If set to true, a response of media type: text/html will be returned with a URI pointing to the newly created document. The comments and simpleResponse parameters are both optional, contentId is optional if Oracle WebCenter Content is set up to auto-generate the dDocName, and the name "fileUpload" is required.

```html
<html>
<head>
  
  <title>simple post</title>

  </head>

  <body>

  <form action="http://<host>:<port>/rest/api/cmis/children/StellentRepository?uid=IDC:
```
31.2.3.2 Document Entry

The specification defines the following CMIS services:

- **GET**: `getObject`, `getObjectOfLatestVersion (getObject)`
- **PUT**: `updateProperties`
- **DELETE**: `deleteObject`

The Content Management REST Service provides the following additional services:

- **POST**: `postToDelete`
  
  This service has two query parameters: `uid` and `_method`, and allows a document to be deleted through POST.

  ```
  http://<host>:<port>/rest/api/cmis/document/repoName?uid=ABC&_method="delete"
  ```

31.2.3.3 Content Stream

The specification defines the following CMIS services:

- **GET**: `getContentStream`
- **PUT**: `setContentStream`
- **DELETE**: `deleteContentStream`

The Content Management REST Service provides the following additional services:

- **POST**: `postTunnelContentStream`
  
  This service has five query parameters: `uid`, `overwriteFlag`, `fileName`, `comments`, `_method`, and one header parameter: Slug. This service is meant to be used as a simple binary request upload or delete through POST. A document must already exist for this service. Slug and `fileName` are used to name the binary that is attached to the request (using both parameters is optional; only one needs to be defined and `fileName` is checked first). The `overwriteFlag` parameter defaults to true, the `comments` parameter is optional and valid values for `_method` are "delete" or "put" (not case sensitive).

  ```
  http://<host>:<port>/rest/api/cmis/stream/repoName?uid=ABC&_method="delete"
  ```

- **POST**: `postTunnelContentStream`
  
  **Content-Type**: `multipart/form-data`

  This service has a single query parameter: `uid` and is meant to be used as a simple HTML multipart/form-data upload or delete through POST. A document must already exist for this service. The attribute name="fileUpload" is required, "comments" is optional, and valid values for "_method" are "delete" or "put" (not case sensitive).

  ```
  action="http://<host>:<port>/rest/api/cmis/stream/repoName?uid=WDOC019113"
  ```
31.3 Content Management REST Service Best Practices and Examples

This section provides best practices and examples using the Content Management REST Service. For details on query syntax, see the CMIS specification.

This section includes the following subsections:

- Section 31.3.1, "Best Practices"
- Section 31.3.2, "Content Management REST Service Examples"

31.3.1 Best Practices

The following list provides suggested best practices for repositories that use the Content Management REST Service.

- To determine the types that can be used in the "FROM" portion of a query, see the types collection from the AtomPub service document. The type must be queryable and the query name of that type must be used.

  For example, IDC:GlobalProfile might have type information similar to the following:

  ```xml
  <cmis:localName>IDC:GlobalProfile</cmis:localName>
  <cmis:displayName>IDC:GlobalProfile</cmis:displayName>
  <cmis:queryName>ora:t:IDC!;GlobalProfile</cmis:queryName>
  <cmis:queryable>true</cmis:queryable>
  
  An example query for the type information above could be: "SELECT * FROM ora:t:IDC!;GlobalProfile".
  
  To determine the properties that can be used in the "SELECT" and "WHERE" portions of a query, see the entry for the associated type. Each property definition of that type will be listed and will have a setting for queryable and orderable. The cmis:queryname will be the value to be used in the query.

  For example, IDC:GlobalProfile might have property definition information similar to the following:

  ```xml
  <cmis:propertyStringDefinition>
    <cmis:id>/stanl18-ucm11g/IDC:GlobalProfile.ora:p:dDocName</cmis:id>
    <cmis:localName>dDocName</cmis:localName>
    <cmis:displayName>dDocName</cmis:displayName>
    <cmis:queryName>ora:p:dDocName</cmis:queryName>
    <cmis:description>Content ID</cmis:description>
    <cmis:propertyType>string</cmis:propertyType>
    <cmis:cardinality>single</cmis:cardinality>
    <cmis:updatability>readwrite</cmis:updatability>
    <cmis:inherited>false</cmis:inherited>
    <cmis:required>false</cmis:required>
    <cmis:queryable>true</cmis:queryable>
    <cmis:orderable>true</cmis:orderable>
  </cmis:propertyStringDefinition>
  ```
An example query for the property definition information above could be:
"SELECT ora:p:dDocName FROM ora:t:IDC!;GlobalProfile"

- To keep queries more readable, avoid non-alphanumeric characters in ContentType and PropertyDefinition names.

### 31.3.2 Content Management REST Service Examples

This section provides some example queries. For details on query syntax, see the CMIS specification. (To get the full URI for a query, see the query URI template in the service document.)

- SELECT * from cmis:folder
- SELECT cmis:name, cmis:contentStreamFileName, cmis:contentStreamMimeType, cmis:contentStreamLength FROM cmis:document WHERE cmis:contentStreamFileName = 'BinaryName' AND cmis:contentStreamMimeType = 'text/html' AND cmis:contentStreamLength > 1
- SELECT cmis:name, cmis:creationDate, cmis:lastModificationDate FROM cmis:folder WHERE cmis:name = 'Trash' AND cmis:lastModificationDate > TIMESTAMP '2008-05-18T10:32:44.703-06:00'
- SELECT * FROM cmis:document WHERE cmis:name LIKE 'baker%'
- SELECT * FROM cmis:document WHERE cmis:name NOT IN ('nodeBoolean', 'nodeLong')
- SELECT cmis:name from cmis:document where IN_TREE('/StellentRepository')
- SELECT * FROM ora:t:IDC:GlobalProfile WHERE ora:p:xBooleanTestField = FALSE ORDER BY ora:p:dDocTitle ASC
- SELECT ora:p:xMultiValuedDelimiter Test FROM ora:t:IDC:GlobalProfile WHERE ANY ora:p:xMultiValuedDelimiter Test NOT IN ('four')
- SELECT cmis:name FROM ora:t:IDC:GlobalProfile WHERE CONTAINS('test') ORDER BY ora:p:dInDate DESC
- SELECT * FROM cmis:document where IN_TREE('/StellentRepository/IDC:Folder/2')
Part V

Enabling Communication and Collaboration

Part V contains the following appendixes:

- Chapter 32, "Integrating Announcements"
- Chapter 33, "Integrating Discussions"
- Chapter 34, "Integrating Instant Messaging and Presence"
- Chapter 35, "Integrating Mail"
- Chapter 36, "Integrating Polls"
This chapter explains how to integrate announcements into a Portal Framework application at design time.

This chapter includes the following sections:

- Section 32.1, "Introduction to Announcements"
- Section 32.2, "Basic Configuration for Announcements"
- Section 32.3, "Advanced Information for Using Announcements"

For more information about managing and including announcements, see:

- the "Managing Announcements and Discussions" chapter in *Administering Oracle WebCenter Portal*
- the "Working with Announcements" chapter in *Using Oracle WebCenter Portal*

### 32.1 Introduction to Announcements

With portals and Framework applications, you can create and expose announcements on your application pages. Access to announcements boosts community participation, problem resolution, and knowledge sharing.

Note that while for portals, announcements can show either forum-level announcements or global announcements (that is, system announcements not specific to a forum), for Framework applications only global announcements are shown.

This section includes the following subsections:

- Section 32.1.1, "Understanding Announcements"
- Section 32.1.2, "Requirements for Announcements"
- Section 32.1.3, "What Happens at Runtime"

### 32.1.1 Understanding Announcements

With WebCenter Portal, you can do the following:

- Create announcements
- Edit and delete existing announcements
- Mail announcements
- Personalize the way in which announcements are viewed. 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.

Announcements are integrated with many other WebCenter Portal features, such as activities, RSS feeds, and instant messaging and mail, and you can link announcements to other WebCenter Portal features 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 messaging tool to speak directly with a customer service representative to find out about or purchase the product.

### 32.1.2 Requirements for Announcements

To use announcements in a Framework application you must have installed and configured the discussions server that comes with Oracle Fusion Middleware.

**See Also:** *Installation Guide for Oracle WebCenter Portal*

### 32.1.3 What Happens at Runtime

At runtime, users who have been granted the Create Announcements permission can post announcements, and these announcements are visible to all users who have View Announcements permission. For example, an application specialist can use announcements to showcase the availability of a new feature or the plan to shut down the application temporarily for maintenance.

*Figure 32–1* shows a sample announcement at runtime.

*Figure 32–1*  **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 32–2*).

*Figure 32–2*  **Open Announcement Manager Icon**

**Note:** The Open Announcement Manager icon does not display if the user does not have the required privileges.

*Figure 32–3* shows the 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 what has been made available in WebCenter Portal and is 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 announcements at runtime, see the "Working with Announcements" chapter in Using Oracle WebCenter Portal.

### 32.2 Basic Configuration for Announcements

This section describes required steps for adding announcements to your Framework application. It includes the following subsections:

- Section 32.2.1, "Setting Up a Connection for Announcements"
- Section 32.2.2, "Adding Announcements at Design Time"
- Section 32.2.3, "Setting Security for Announcements"

#### 32.2.1 Setting Up a Connection for Announcements

To take advantage of announcements, 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.

#### 32.2.1.1 Announcements Connections

Announcements 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 announcements and discussions use this same connection. If you have an existing connection, then you can skip this section and see Section 32.2.2, "Adding Announcements at Design Time."

If you do not have an existing connection, then you must create a new Discussion Forum connection.
32.2.1.2 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 announcements, then that is used by default for announcements. 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 check box. You can have multiple connections, but only one can be active (default). If you have different discussions servers (for example, for each page in your application), then do not select the check box, but one connection must be marked as the active connection (Figure 32–4).

   **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.
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 discussions and announcements to perform administrative operations on behalf of WebCenter Portal users.

   In WebCenter Portal, this account is mostly used for managing portal-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 WebCenter Portal, that is, the category (on the discussions server) under which all portal-related discussions and announcements are stored.

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 a default value of 10 seconds is used.

   ■ **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 "Configuring WS-Security" chapter in Administering Oracle WebCenter Portal.

Figure 32–5  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 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.

### 32.2.2 Adding Announcements at Design Time

This section explains how announcements can be incorporated at design time.

#### 32.2.2.1 Announcements 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 32–1  Announcements 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 32.3.1, "How to Add the Announcements - Quick View Task Flow." |
32.2.2.2 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 2, "Setting Up 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 an announcement.


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.

32.2.2.3 How to Modify Announcement Task Flow Parameters

The announcements 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 32–6 shows an example of a Search task flow in the Executables section.

Figure 32–6 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 32–7**, revise the binding parameter values as required.

**Figure 32–7  Edit Task Flow Binding Dialog for Announcements - Quick View**

![Edit Task Flow Binding Dialog](image)

5. When you are finished, click **OK**.

6. Save and run your page to see the results.

**Table 32–2** describes the properties that are unique to announcements task flows.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Task Flow</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>parentld</strong></td>
<td>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, <code>${2}</code>. If this parameter is not specified, then announcements default to global announcements.</td>
<td>Announcements</td>
</tr>
<tr>
<td></td>
<td><strong>Announcements - Quick View</strong></td>
<td></td>
</tr>
<tr>
<td><strong>freeFlowView</strong></td>
<td>A Boolean value representing whether to remove the announcement title (subject) and show the announcement body as is. The default value is <code>${false}</code>, which means that the announcement list displays the title and body in plain text and the truncateAt and expandedAnnouncements parameters control it. Enter <code>${true}</code> to remove the announcement title and ignore the values for truncateAt and expandedAnnouncements.</td>
<td>Announcements - Quick View</td>
</tr>
<tr>
<td></td>
<td><strong>Announcements - Quick View</strong></td>
<td></td>
</tr>
<tr>
<td><strong>expandedAnnouncements</strong></td>
<td>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.</td>
<td>Announcements - Quick View</td>
</tr>
</tbody>
</table>
Basic Configuration for Announcements

You can change the look and feel of the Announcements - Quick View task flow using parameter values. For example, Figure 32–12 shows the Announcements - Quick View task flow at runtime with the freeFlowView parameter set to false (or empty).

Figure 32–8  Announcements - Quick View with Optional Parameters

![Image of the Announcements - Quick View task flow with optional parameters]

Figure 32–13 shows the Announcements - Quick View task flow at runtime with the freeFlowView parameter set to $\{true\}$.

Figure 32–9  Announcements - Quick View with freeFlowView Parameter

![Image of the Announcements - Quick View task flow with freeFlowView parameter set to $\{true\}$]
32.2.3 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 on 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:** Announcements requires an external LDAP-based identity store; that is, you cannot use the out-of-the-box embedded LDAP with announcements.

---

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 74.3, "Configuring ADF Security."

32.3 Advanced Information for Using Announcements

This section describes optional features available with announcements. It includes the following subsections:

- Section 32.3.1, "How to Add the Announcements - Quick View Task Flow"
- Section 32.3.2, "Customizing Announcements Views"
- Section 32.3.3, "Obtaining a Portal RSS News Feed URL for Announcements"

32.3.1 How to Add the Announcements - Quick View Task Flow

The Announcements - Quick View task flow provides a snapshot (dashboard) view of the announcements (Figure 32–10).

**Figure 32–10  Announcements - Quick View**

<table>
<thead>
<tr>
<th>Release 2.0 Ready to Ship</th>
</tr>
</thead>
<tbody>
<tr>
<td>Office Closing at 3:00</td>
</tr>
</tbody>
</table>
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 32–10).

**Figure 32–11 Announcement Popup**

```
Announcement

Release 2.0 Ready to Ship

Start celebrating!
```

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 32.2.2, "Adding Announcements at Design Time," but drag and drop Announcements - Quick View onto the page.

Table 32–2 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 32.2.2.3, "How to Modify Announcement Task Flow Parameters."
32.3.2 Customizing Announcements Views

You can change the look and feel of the Announcements - Quick View with the parameter values. For example, Figure 32–12 shows the Announcements - Quick View at runtime with the `freeFlowView` parameter set to false (or empty).

![Figure 32–12  Announcements - Quick View with Optional Parameters](image)

Figure 32–12 shows the Announcements - Quick View task flow at runtime with the `freeFlowView` parameter set to `${true}`.

![Figure 32–13  Announcements - Quick View task flow with freeFlowView Parameter](image)

32.3.3 Obtaining a Portal RSS News Feed URL for Announcements

You can expose WebCenter Portal functionality in a Framework application so that your Framework application users can find out what is happening in a specific portal through RSS news feeds.

Configure RSS news feeds for the announcements to enable users to view portal announcements from within a Framework application. To obtain the portal RSS news feed URL for the announcements, use either of the following WebCenter Portal API methods:

- `getServiceRSSFeedURL`
- `getServiceRSSFeedURLbyGuid`

To obtain an RSS feed URL, you must identify the portal (by name or GUID) and specify the service required (by service ID). The service ID for the announcements is `GroupSpaceWSClient.ANNOUNCEMENT_SERVICE_ID`.

For information about how to use these APIs, see Section 56.1.5.3.9, "Retrieving RSS Feed URLs for WebCenter Portal Tools and Services."
This chapter explains how to integrate discussions in a Portal Framework application at design time.

This chapter includes the following topics:

- Section 33.1, "Introduction to Discussions"
- Section 33.2, "Basic Configuration for Discussions"
- Section 33.3, "Advanced Information for Discussions"

For more information about managing and including discussions, see:

- the "Managing Announcements and Discussions" chapter in *Administering Oracle WebCenter Portal*
- the "Working with Discussions" chapter in *Using Oracle WebCenter Portal*

### 33.1 Introduction to Discussions

You can 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.

#### 33.1.1 Understanding Discussions

With discussions, users 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 33.2.3, "Setting Security for Discussions."

Discussions is integrated with many WebCenter Portal components, such as instant messaging and presence, RSS, and search (to search within forums). Mail can be archived into discussions as threads. You can also link to a discussion from any WebCenter Portal object.

### 33.1.2 Requirements for Discussions

Discussions requires a discussions server. Install and configure the discussions server that comes with Oracle Fusion Middleware.

**See Also:** *Installation Guide for Oracle WebCenter Portal*

### 33.1.3 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 the "Granting Administrator Role on the Discussions Server" section in *Administering Oracle WebCenter Portal*.

Figure 33–1 shows a discussion forum at runtime.

**Figure 33–1 Discussion Forums Task Flow at Runtime**

<table>
<thead>
<tr>
<th>Topic</th>
<th>Recent Topics</th>
<th>Replies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development Issues</td>
<td>9/9/11 2:05 PM</td>
<td>0</td>
</tr>
<tr>
<td>QA Issues</td>
<td>9/9/11 2:05 PM</td>
<td>0</td>
</tr>
<tr>
<td>Sales Issues</td>
<td>9/9/11 2:04 PM</td>
<td>0</td>
</tr>
</tbody>
</table>

For more information about discussions at runtime, see the "Viewing and Participating in Discussions" chapter in *Using Oracle WebCenter Portal*. 
33.2 Basic Configuration for Discussions

This section describes required steps for adding discussions to your application. It includes the following subsections:

- Section 33.2.1, "Setting up Connections for Discussions"
- Section 33.2.2, "Adding Discussions Functionality at Design Time"
- Section 33.2.3, "Setting Security for Discussions"

33.2.1 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 announcements and discussions use this same connection. If you have an existing connection, then you can skip this section and see Section 33.2.2, "Adding Discussions Functionality at Design Time."

If you do not have an existing connection, then you must create a new Discussion Forum connection.

33.2.1.1 Discussions Connections

Discussions 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 Administering Oracle WebCenter Portal.

33.2.1.2 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 announcements, then that is used by default for discussions. 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 check box. 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 portal), then do not select the checkbox, but one connection must be marked as the default connection (Figure 33–2).

---

**Note:** After you create a connection and set it as the default connection, you cannot reset it so that it is not the default. That is, the **Set as default connection** option cannot be unchecked. To use a different default connection, you must create a new connection and set that as the default connection.

---

**Figure 33–2  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 discussions and announcements to perform administrative operations on behalf of WebCenter Portal users.

   In Portal, this account is mostly used for managing portal-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 Portal, that is, the category (on the discussions server) under which all portal-related discussions and announcements are stored.

---

**Note:** If your Framework application does not include portal-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 default is used. The 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 "Configuring WS-Security" chapter in *Administering Oracle WebCenter Portal*. 
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 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 Discussions Functionality at Design Time

This section explains a basic incorporation of discussions.

#### 33.2.2.1 Discussions Task Flows

There are several task flows (Table 33–1) to provide discussions in a form that best suits your needs.
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 4.2.1, "How to Prepare Your Application to Consume Tools and Services" to implement security and create a new customizable page in your application.

2. Open the page on which you want to add discussions.


Table 33–1 Discussions Task Flows

<table>
<thead>
<tr>
<th>Task Flow</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discussion Forums</td>
<td>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 focused views of watched forums or watched topics. The parameters alter the way the view appears. For more information, see Section 33.2.2.3, &quot;How to Modify the Discussions Task Flow Parameters.&quot;</td>
</tr>
<tr>
<td>Discussions - Popular Topics</td>
<td>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 33.3.1, &quot;Adding the Discussions - Popular Topics Task Flow.&quot;</td>
</tr>
<tr>
<td>Discussions - Recent Topics</td>
<td>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 33.3.2, &quot;Adding the Discussions - Recent Topics Task Flow.&quot;</td>
</tr>
<tr>
<td>Discussions - Watched Forums</td>
<td>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 33.3.3, &quot;Adding the Discussions - Watched Forums Task Flow.&quot;</td>
</tr>
<tr>
<td>Discussions - Watched Topics</td>
<td>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 33.3.4, &quot;Adding the Discussions - Watched Topics Task Flow.&quot;</td>
</tr>
<tr>
<td>Discussions - Quick View</td>
<td>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 33.3.5, &quot;Adding the Discussions - Quick View Task Flow.&quot;</td>
</tr>
</tbody>
</table>
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 33–2** 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 task flows that require connections to back-end servers, such as task flows from announcements or mails.

---

The Discussion Forums main view contains some features that require other components 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 Links."

- For users’ presence indicators to work, you must have configured the WebCenter Portal Presence service. For more information, see Chapter 34, "Integrating Instant Messaging and Presence."

- 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 discussions respects those credentials in the discussions server.

---

### 33.2.2.3 How to Modify the Discussions Task Flow Parameters

Discussions 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 33–4** shows an example of a Search task flow in the Executables section.
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 33–5, revise the binding parameter values as required.

Figure 33–5  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 33–2 describes the properties that are unique to the discussions task flows.
### Table 33–2  Discussions Task Flow Parameters

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Task Flow</th>
</tr>
</thead>
<tbody>
<tr>
<td>categoryId</td>
<td>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 <code>application.root.category.id</code> 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.</td>
<td>■ Discussion Forums&lt;br&gt;■ Discussions - Quick View&lt;br&gt;■ Popular Topics&lt;br&gt;■ Recent Topics&lt;br&gt;■ Watched Forums&lt;br&gt;■ Watched Topics</td>
</tr>
<tr>
<td>forumId</td>
<td>This parameter is an identifier for an existing forum in your discussions server for which popular topics should be fetched. If both <code>categoryId</code> and <code>forumId</code> are given, then only <code>categoryId</code> is honored.</td>
<td>■ Discussion Forums&lt;br&gt;■ Popular Topics&lt;br&gt;■ Recent Topics&lt;br&gt;■ Watched Forums&lt;br&gt;■ Watched Topics</td>
</tr>
<tr>
<td>disableToolbar</td>
<td>This parameter indicates whether the toolbar, including the Refresh icon, should be displayed. It can be <code>${true}</code>, <code>${false}</code>, or undefined. If you leave this parameter undefined, then the default behavior (false) is to show the toolbar.</td>
<td>■ Popular Topics&lt;br&gt;■ Recent Topics&lt;br&gt;■ Watched Forums&lt;br&gt;■ Watched Topics</td>
</tr>
<tr>
<td>showRecursiveForums</td>
<td>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.</td>
<td>■ Discussion Forums</td>
</tr>
<tr>
<td>isCategoryView</td>
<td>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 <code>categoryId</code>; false, the default value, means you want the task flow to display the topics associated with the specified <code>forumId</code>. This parameter value works in combination with other parameters.</td>
<td>■ Discussion Forums</td>
</tr>
<tr>
<td>defaultTopicFetchSize</td>
<td>Sets the number of visible topics.</td>
<td>■ Discussion Forums</td>
</tr>
<tr>
<td>doNotAllowSelectingPageSize</td>
<td>If set to true, users are not allowed to change the number of visible topics.</td>
<td>■ Discussion Forums</td>
</tr>
<tr>
<td>visibleTopicsFetchSize</td>
<td>Sets the number of visible watched topics.</td>
<td>■ Recent Topics&lt;br&gt;■ Watched Topics</td>
</tr>
<tr>
<td>visibleForumsFetchSize</td>
<td>Sets the number of visible watched forums.</td>
<td>■ Watched Forums</td>
</tr>
<tr>
<td>doNotShowMoreLink</td>
<td>If set to true, the More link is not visible.</td>
<td>■ Discussions - Quick View&lt;br&gt;■ Recent Topics&lt;br&gt;■ Watched Forums&lt;br&gt;■ Watched Topics</td>
</tr>
</tbody>
</table>
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.

### 33.2.3 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.

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:** Discussions 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 74.3, "Configuring ADF Security."

### 33.3 Advanced Information for Discussions

This section describes optional features available with discussions. It includes the following subsections:

- Section 33.3.1, "Adding the Discussions - Popular Topics Task Flow"
- Section 33.3.2, "Adding the Discussions - Recent Topics Task Flow"
- Section 33.3.3, "Adding the Discussions - Watched Forums Task Flow"
- Section 33.3.4, "Adding the Discussions - Watched Topics Task Flow"
- Section 33.3.5, "Adding the Discussions - Quick View Task Flow"
- Section 33.3.6, "Obtaining Portal RSS News Feed URL for Discussions"
- Section 33.3.7, "Using Custom Discussions APIs"
- Section 33.3.8, "Using the Discussions REST API"
- Section 33.3.9, "Troubleshooting Discussions"

#### 33.3.1 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 33.2.2, "How to Add Discussions to a Page," but drag and drop Discussions - Popular Topics onto the page.
Table 33–2 describes the available parameters for this task flow.

Figure 33–6 shows how the Discussions - Popular Topics task flow looks at runtime.

Figure 33–6  Discussions - Popular Topics View at Runtime

33.3.2 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 33.2.2.2, "How to Add Discussions to a Page," but drag and drop Discussions - Recent Topics onto the page.

Table 33–2 describes the available parameters for this task flow.

Figure 33–7 shows how the Discussions - Recent Topics task flow looks at runtime. To view a topic post, simply click the relevant topic.

Figure 33–7  Discussions - Recent Topics View at Runtime

33.3.3 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, in addition to it appearing in their watched forums or topics list, the user receives an email notification whenever other users add to that forum or topic.

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 33.2.2.2, "How to Add Discussions to a Page," but drag and drop Discussions -
Watched Forums onto the page.

Table 33–2 describes the available parameters for this task flow.

Figure 33–8 shows how the Discussions - Watched Forums task flow looks at runtime.

Figure 33–8 Discussions - Watched Forums View at Runtime

For information on how to watch a forum or topic at runtime, see the “Working with Discussions” chapter in Using Oracle WebCenter Portal.

33.3.4 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 33.2.2.2, "How to Add Discussions to a Page," but drag and drop Discussions - Watched Topics onto the page.

Table 33–2 describes the available parameters for this task flow.

Figure 33–9 shows how the Discussions - Watched Topics task flow looks at runtime.

Figure 33–9 Discussions - Watched Topics View at Runtime

For information on how to watch a forum or topic at runtime, see the “Working with Discussions” chapter in Using Oracle WebCenter Portal.

33.3.5 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 33.2.2.2, "How to Add Discussions to a Page," but drag and drop Discussions - Quick View onto the page.

Table 33–2 describes the available parameters for this task flow.

Figure 33–10 shows how the Discussions - Quick View task flow looks at runtime.

33.3.6 Obtaining Portal RSS News Feed URL for Discussions

You can expose portal functionality in a Framework application. Your Framework application users can find out what is happening in a specific portal through RSS news feeds.

Configure RSS news feeds for discussions to enable users to view portal discussions from within a Framework application. To obtain the portal RSS news feed URL for discussions, use either of the following WebCenter Portal API methods:

- getServiceRSSFeedURL
- getServiceRSSFeedURLbyGuid

To obtain an RSS feed URL, you must identify the portal (by name or GUID) and specify the component required (by service ID). The service ID for discussions is GroupSpaceWSClient.DISCUSSION_SERVICE_ID.

For information about how to use these methods, see Section 56.1.5.3.9, "Retrieving RSS Feed URLs for WebCenter Portal Tools and Services."

33.3.7 Using Custom Discussions APIs

The back-end discussions 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).

33.3.8 Using the Discussions REST API

WebCenter Portal provides a REST API to support discussions. Use the discussions REST API to post, read, update, and delete discussion forums, topics, and messages.

This section describes the REST API methods associated with discussions. It includes the following subsections:

- Section 33.3.8.1, "Discussions Entry Point"
- Section 33.3.8.2, "Discussions Resource Type Taxonomy"
- Section 33.3.8.3, "Security Considerations"
- Section 33.3.8.4, "Discussions Resource Types"
For an introduction to the REST APIs, see Chapter 53, "Using Oracle WebCenter Portal REST APIs."

### 33.3.8.1 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 discussions, 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 discussions.

For more information about the Resource Index, see Section 53.5.1, "Using the Resource Index."

For more information about resource types, see Section 53.5.2.1, "Resource Type."

### 33.3.8.2 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 53.5.2.1, "Resource Type."

The taxonomy for discussions 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`).

### 33.3.8.3 Security Considerations

There are no specific security considerations for discussions. For general security considerations, see Section 53.8, "Security Considerations for WebCenter Portal REST APIs."

### 33.3.8.4 Discussions Resource Types

This section provides the information about each resource type. It includes the following subsections:

- Section 33.3.8.4.1, "urn:oracle:webcenter:discussions:forums"
- Section 33.3.8.4.2, "urn:oracle:webcenter:discussions:forum"
- Section 33.3.8.4.3, "urn:oracle:webcenter:discussions:forum:topics"
- Section 33.3.8.4.4, "urn:oracle:webcenter:discussions:forum:topic"
- Section 33.3.8.4.5, "urn:oracle:webcenter:discussions:forum:topic:messages"
- Section 33.3.8.4.6, "urn:oracle:webcenter:discussions:forum:topic:message"
33.3.8.4.1 **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

Note that the number of forums displayed per page defaults to 10. For more information, see Section 53.5.2.5, "Templates."

**Resource Types Linked to From forums**
Table 33–3 lists the resource types that the client can link to from this resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td><strong>urn:oracle:webcenter:discussions:forums</strong></td>
</tr>
<tr>
<td></td>
<td><strong>urn:oracle:webcenter:discussions:forum</strong></td>
</tr>
</tbody>
</table>

33.3.8.4.2 **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
```
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 33–4 lists the writable elements for this resource.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Required</th>
<th>Constraints</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Yes</td>
<td>1 or more characters</td>
<td>Name of the forum</td>
</tr>
<tr>
<td>displayName</td>
<td>String</td>
<td>No</td>
<td>1 or more characters</td>
<td>Name used in presentation</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>No</td>
<td>1 or more characters</td>
<td>Description of the forum</td>
</tr>
</tbody>
</table>

Read-only Elements for forum
Table 33–5 lists the read-only elements for this resource.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Integer</td>
<td>ID of the forum</td>
</tr>
<tr>
<td>parentId</td>
<td>Integer</td>
<td>ID of the parent category</td>
</tr>
<tr>
<td>createdBy</td>
<td>String</td>
<td>ID of the user that created the forum</td>
</tr>
<tr>
<td>author</td>
<td>personReference</td>
<td>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)</td>
</tr>
<tr>
<td>createdOn</td>
<td>Date</td>
<td>Date on which the forum was created</td>
</tr>
</tbody>
</table>
### Resource Types Linked to From forum

Table 33–6 lists the resource types that the client can link to from this resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:discussions:forum</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:discussions:forum:topics</td>
</tr>
</tbody>
</table>

#### 33.3.8.4.3 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

---

**Table 33–5 (Cont.) Read-only Elements for forum**

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>updatedBy</td>
<td>String</td>
<td>ID of the user that performed the last modification</td>
</tr>
<tr>
<td>modifiedBy</td>
<td>personReference</td>
<td>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)</td>
</tr>
<tr>
<td>updatedOn</td>
<td>Date</td>
<td>Date on which the forum was last modified</td>
</tr>
<tr>
<td>webUrl</td>
<td>String</td>
<td>URL for direct access to the discussions server</td>
</tr>
<tr>
<td>topicCount</td>
<td>Integer</td>
<td>Number of topics</td>
</tr>
<tr>
<td>messageCount</td>
<td>Integer</td>
<td>Number of messages</td>
</tr>
<tr>
<td>locked</td>
<td>Boolean</td>
<td>True if this forum is locked</td>
</tr>
<tr>
<td>favorite</td>
<td>Boolean</td>
<td>True if this forum is marked as a favorite</td>
</tr>
</tbody>
</table>

---

**Table 33–6 Related Resource Types for forum**

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:discussions:forum</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:discussions:forum:topics</td>
</tr>
</tbody>
</table>
POST

- request - body: topic
- response - body: topic

For more information, see Section 53.5.2.5, "Templates."

Resource Types Linked to From topics
Table 33–7 lists the resource types that the client can link to from this resource.

**Table 33–7 Related Resource Types for topics**

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:discussions:forum:topics</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:discussions:forum:topic</td>
</tr>
</tbody>
</table>

33.3.8.4.4 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 33–8 lists the writable elements for this resource type.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Required</th>
<th>Constraints</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>subject</td>
<td>String</td>
<td>Yes</td>
<td>1 or more characters</td>
<td>Subject of the topic</td>
</tr>
<tr>
<td>body</td>
<td>String</td>
<td>No</td>
<td>0 or more characters</td>
<td>Contents of the topic</td>
</tr>
</tbody>
</table>

### Read-only Elements for topic

Table 33–9 lists the read-only elements for this resource type.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Integer</td>
<td>Identifier for the topic</td>
</tr>
<tr>
<td>parentId</td>
<td>Integer</td>
<td>Identifier for the parent message</td>
</tr>
<tr>
<td>forumId</td>
<td>Integer</td>
<td>Identifier for the forum under which this topic is posted</td>
</tr>
<tr>
<td>topicId</td>
<td>Integer</td>
<td>Identifier for the topic under which this topic is posted</td>
</tr>
<tr>
<td>createdBy</td>
<td>String</td>
<td>ID of the user who created the topic</td>
</tr>
<tr>
<td>author</td>
<td>personReference</td>
<td>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)</td>
</tr>
<tr>
<td>createdOn</td>
<td>Date</td>
<td>Date on which the topic was created</td>
</tr>
<tr>
<td>updatedBy</td>
<td>String</td>
<td>User who lasted updated the topic</td>
</tr>
<tr>
<td>updatedOn</td>
<td>Date</td>
<td>Date on which the topic was last updated</td>
</tr>
<tr>
<td>webUrl</td>
<td>String</td>
<td>URL for direct access to the discussions server</td>
</tr>
<tr>
<td>depth</td>
<td>Integer</td>
<td>Depth in the hierarchy of messages</td>
</tr>
<tr>
<td>messageCount</td>
<td>Integer</td>
<td>Number of child messages under this topic</td>
</tr>
<tr>
<td>numberOfReplies</td>
<td>Integer</td>
<td>Number of replies to this topic</td>
</tr>
<tr>
<td>favorite</td>
<td>Boolean</td>
<td>Whether the topic is marked as a favorite for the user</td>
</tr>
<tr>
<td>locked</td>
<td>Boolean</td>
<td>Whether the topic is locked</td>
</tr>
<tr>
<td>hidden</td>
<td>Boolean</td>
<td>Whether the topic is hidden</td>
</tr>
<tr>
<td>hasAttachment</td>
<td>Boolean</td>
<td>Whether the topic has attachments</td>
</tr>
</tbody>
</table>

### Resource Types Linked to From topic

Table 33–10 lists the resource types that the client can link to from this resource.
Table 33–10 Related Resource Types for topic

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:discussions:forum:topic</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:discussions:forum:topics</td>
</tr>
<tr>
<td></td>
<td>messages</td>
</tr>
</tbody>
</table>

33.3.8.4.5 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 53.5.2.5, "Templates."

Resource Types Linked to From messages
Table 33–11 lists the resource types that the client can link to from this resource.

Table 33–11 Related Resource Types for messages

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:discussions:forum:topic:messages</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:discussions:forum:topic:message</td>
</tr>
</tbody>
</table>

33.3.8.4.6 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:

```plaintext
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 33–12 lists the writable elements for this resource.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Required</th>
<th>Constraints</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>subject</td>
<td>String</td>
<td>Yes</td>
<td>1 or more characters</td>
<td>Subject of this message</td>
</tr>
<tr>
<td>body</td>
<td>String</td>
<td>No</td>
<td>0 or more characters</td>
<td>Content of this message</td>
</tr>
</tbody>
</table>

**Read-only Elements for message**

Table 33–13 lists the read-only elements for this resource.
### Table 33–13 Read-only Elements for message

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Integer</td>
<td>ID of the message</td>
</tr>
<tr>
<td>parentId</td>
<td>Integer</td>
<td>ID of the parent message</td>
</tr>
<tr>
<td>forumId</td>
<td>Integer</td>
<td>ID of the forum under which the message is posted</td>
</tr>
<tr>
<td>topicId</td>
<td>Integer</td>
<td>ID of the topic under which the message is posted</td>
</tr>
<tr>
<td>createdBy</td>
<td>String</td>
<td>ID of user that created the message</td>
</tr>
<tr>
<td>author</td>
<td>personReference</td>
<td>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)</td>
</tr>
<tr>
<td>createdOn</td>
<td>Date</td>
<td>Date on which the message was created</td>
</tr>
<tr>
<td>updatedBy</td>
<td>String</td>
<td>User that performed the last modification</td>
</tr>
<tr>
<td>updatedOn</td>
<td>Date</td>
<td>Date on which the message was last modified</td>
</tr>
<tr>
<td>webUrl</td>
<td>String</td>
<td>URL for direct access to the discussions server</td>
</tr>
<tr>
<td>depth</td>
<td>Integer</td>
<td>Depth in the hierarchy of messages</td>
</tr>
<tr>
<td>messageCount</td>
<td>Integer</td>
<td>Number of messages</td>
</tr>
<tr>
<td>numberOfReplies</td>
<td>Integer</td>
<td>Number of replies to this message</td>
</tr>
<tr>
<td>hidden</td>
<td>Boolean</td>
<td>True if this message is hidden</td>
</tr>
<tr>
<td>hasAttachment</td>
<td>Boolean</td>
<td>True if this message has attachments</td>
</tr>
</tbody>
</table>

### Resource Types Linked to From message

Table 33–14 lists the resource types that the client can link to from this resource.

### Table 33–14 Related Resource Types for message

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:discussions:forum:topic:message</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:discussions:forum:topic:messages</td>
</tr>
</tbody>
</table>

### 33.3.9 Troubleshooting Discussions

This section describes common problems and solutions for discussions.

**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 the Framework application but cannot log on to the discussions server.

**Solution**
Make sure that discussions server is configured to use the same identity store as the Framework application. Contact your administrator for details. For information about configuring the identity store, see the "Configuring the Identity Store" section in *Administering Oracle WebCenter Portal*.

**Problem**
When you access a discussions forum from your Framework application, the following error message is displayed:

```plaintext
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.
This chapter explains how to integrate instant messaging and presence (IMP) in a Portal Framework application at design time.

This chapter includes the following topics:

- Section 34.1, "Introduction to Instant Messaging and Presence"
- Section 34.2, "Basic Configuration for IMP"
- Section 34.3, "Advanced Information for IMP"

For more information about managing and including instant messaging and presence, see:

- the "Managing Instant Messaging and Presence" chapter in Administering Oracle WebCenter Portal
- the "Using Instant Messaging and Presence Viewer" chapter in Using Oracle WebCenter Portal

34.1 Introduction to Instant Messaging and Presence

Instant messaging and presence (IMP) enables users 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 third-party network presence service, such as Yahoo! Messenger.

Tools and services within WebCenter Portal that have user names with the same identity can integrate with IMP; for example, discussions, documents, or mail.

This section provides an overview of IMP features and requirements. It includes the following subsections:

- Section 34.1.1, "Understanding Instant Messaging and Presence"
- Section 34.1.2, "Requirements for Instant Messaging and Presence"

34.1.1 Understanding Instant Messaging and Presence

Figure 34–1 shows the Presence icon indicating a user who is online.

*Figure 34–1 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 34–2).

![Figure 34–2 Presence Icon Context Menu](image)

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 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 IMP at runtime, including screen shots and descriptions of the presence status options, see the “Using Instant Messaging and Presence Viewer” chapter in *Using Oracle WebCenter Portal*.

### 34.1.2 Requirements for Instant Messaging and Presence


---

**Note:** Oracle Beehive Server connections are not supported in this release.

---

### 34.2 Basic Configuration for IMP

This section describes the steps required for adding IMP to your application. It includes the following subsections:

- **Section 34.2.1, "Setting up Connections for IMP"**
- **Section 34.2.2, "Adding IMP Functionality at Design Time"**
- **Section 34.2.3, "Setting Security for IMP"**

#### 34.2.1 Setting up Connections for IMP

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 34.2.1.1, "Instant Messaging and Presence Connections"**
- **Section 34.2.1.2, "How to Set Up Microsoft LCS Connections for IMP"**
34.2.1.1 Instant Messaging and Presence Connections

Instant Messaging and Presence requires an **Instant Messaging and Presence** connection to the presence server (Microsoft LCS, Microsoft OCS, or Microsoft Lync).

When 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 Fusion Middleware Control. For more information, see *Administering Oracle WebCenter Portal*.

---

34.2.1.1.1 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 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 property `im.address.resolver.class` from the default `oracle.webcenter.collab.rtc.IMPAddressResolverImpl` to your resolver implementation. **Example 34–1** 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 34–1  Resolver Implementation**

```java
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: The “setIMPServiceProperty” section in the WebLogic Scripting Tool Command Reference

34.2.1.2 How to Set Up Microsoft LCS Connections for IMP
To set up the connection to the LCS presence server:

1. In Oracle JDeveloper, open the application in which to consume instant messaging and presence.
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 34–3.
7. Click Next

8. On the General page (Figure 34–4), enter the information for your Microsoft LSC instance.
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) WebCenter Portal 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 IMP, see Section 34.2.3, "Setting Security for IMP."

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 34–5), you can optionally add parameters.

![Figure 34–5](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.

34.2.1.3 How to Set Up Microsoft OCS and Microsoft Lync Connections for IMP

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 instant messaging and presence.

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.
   One connection must be marked as the default connection, as shown in **Figure 34–6**.

   **Figure 34–6  Create Microsoft OCS or Microsoft Lync IMP Connection, Step 1**

7. Click **Next**.

8. On the **General** page (**Figure 34–7**), enter the parameters for your Microsoft OCS or Microsoft Lync instance.
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) WebCenter Portal 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 IMP, see Section 34.2.3, "Setting Security for IMP."

10. Click Test Connection to confirm that the connection is good.

11. Click Next.

12. On the Additional Properties page (Figure 34–5), you can optionally add parameters.

13. Click Finish.

You can see the new IM and presence connection under Application Resources - Connections.

34.2.2 Adding IMP Functionality at Design Time

This section explains a basic incorporation of IMP into your application. It includes the following subsections:

- Section 34.2.2.1, "IMP Task Flows"
- Section 34.2.2.2, "How to Add IMP Functionality to your Application"

34.2.2.1 IMP Task Flows

IMP does not include any task flows.
34.2.2.2 How to Add IMP Functionality to your Application

To add IMP to your Framework application:

1. Follow the steps described in Chapter 2, "Setting Up Your Development Environment" to create a customizable page in your application.

2. Open the page on which you want to add IMP.

3. Ensure that you have configured your application to connect to the presence server. See Section 34.2.1, "Setting up Connections for IMP."

4. In the Component Palette, drag and drop the Presence component to the page to add the Presence icon (Figure 34–9).

Figure 34–9  Component Palette - IMP 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 34.3.2, "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 34–10 shows the runtime Presence component with the display name of user Monty Montasaurus111.

![WebCenter Portal Presence Icon - Online](image)

### Figure 34–10  WebCenter Portal Presence Icon - Online

**Monty Montasaurus111**

#### 34.2.3 Setting Security for IMP

IMP 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 74.3, "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, IMP 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 IMP 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 34–11**.

---

**Note:** The external application for IMP requires this additional field. It must be displayed to users.
8. External applications allow 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 74.13, "Working with External Applications."

34.3 Advanced Information for IMP

This section describes optional features available with IMP. It includes the following subsections:

- Section 34.3.1, "Enabling Network Presence"
- Section 34.3.2, "Customizing IMP Views"
- Section 34.3.3, "Troubleshooting IMP"

34.3.1 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 third-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 34.3.1.1, "Setting Up Yahoo! Messenger Presence"
- Section 34.3.1.2, "Setting Up Other Network Presence Providers"

34.3.1.1 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 34–2 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 34–2   User Interface for Yahoo! Messenger Presence

   ```html
   <af:inputText label="Display Name" value="#{rtcPreferenceHandler.displayName}" id="it1"/>
   <af:inputText label="IM Address" value="#{rtcPreferenceHandler.imAddress}" id="it2"/>
   ```
2. Run the JPSX page to see that now users can enter their Yahoo! Messenger credentials (Figure 34–12).

Example 34–3 shows the full source code for the page, with the user interface included.

Example 34–3  Source Code for the Page with Yahoo Presence

```xml
<?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: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>
      </af:panelStretchLayout>
    </af:form>
  </af:document>
</f:view>
</jsp:root>
```
Presence tags for that user shows their Yahoo presence, either online (Figure 34–13) or offline (Figure 34–14).

Figure 34–13   Yahoo Presence Icon - Online

Figure 34–14   Yahoo Presence Icon - Offline

Figure 34–15 shows a sample page that includes the Discussion Forums task flow, where Yahoo! Messenger presence and WebCenter Portal presence are shown together.

34.3.1.2 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 34–3 shows this SamplePNA class file.

**Example 34–4  SamplePNA Class File**

```java
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();
        String 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

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"/>

### 34.3.2 Customizing IMP Views

Table 34–1 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>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>username</td>
<td>The user whose presence information you want to add to the application page. This attribute is required.</td>
</tr>
<tr>
<td>display</td>
<td>How the component should display. Takes one of the following values:</td>
</tr>
<tr>
<td></td>
<td>■ icon: Display only the Presence icon; do not render the name.</td>
</tr>
<tr>
<td></td>
<td>■ name: Display the user name; do not display the Presence icon.</td>
</tr>
<tr>
<td></td>
<td>■ both (default): Display both the icon and the user name.</td>
</tr>
<tr>
<td>displayName</td>
<td>By default, the Presence component displays the user name.</td>
</tr>
<tr>
<td></td>
<td>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.</td>
</tr>
<tr>
<td>iconPosition</td>
<td>The position for the icon. Possible values are begin and end. The default value is begin.</td>
</tr>
<tr>
<td>controlsEnabled</td>
<td>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.</td>
</tr>
<tr>
<td>id</td>
<td>A unique identifier for the component on the page. The identifier must follow a subset of the syntax allowed in HTML:</td>
</tr>
<tr>
<td></td>
<td>■ Must not be a zero-length String.</td>
</tr>
<tr>
<td></td>
<td>■ First character must be an ASCII letter (A-Z, a-z) or an underscore ('_').</td>
</tr>
<tr>
<td></td>
<td>■ Subsequent characters must be an ASCII letter or digit (A-Z, a-z, 0-9), an underscore ('_'), or a dash ('-').</td>
</tr>
<tr>
<td>smallIcon</td>
<td>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.</td>
</tr>
<tr>
<td>rendered</td>
<td>A Boolean value that defines whether to render this component. The default value is true.</td>
</tr>
<tr>
<td>binding</td>
<td>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.</td>
</tr>
<tr>
<td>inlineStyle</td>
<td>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.</td>
</tr>
</tbody>
</table>
### 34.3.3 Troubleshooting IMP

This section describes common problems and solutions for IMP.

**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, IMP fetches the presence information from the presence server and stores it in the presence cache. For presence requests, IMP 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` 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 34–5 shows an example.

**Example 34–5 Setting the rtc.cache.time Expiration Value in adf-config.xml**

```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 IMP context menu in your Framework application.

**Solution**
Ensure that IMP 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 IMP 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.
This chapter explains how to integrate email functionality in a Portal Framework application at design time.

This chapter includes the following sections:

- **Section 35.1, "Introduction to Mail"
- **Section 35.2, "Basic Configuration for Mail"
- **Section 35.3, "Advanced Information for Mail"

---

**Note:** WebCenter Portal Mail supports only the Inbox. No other folders (or moving of messages) are supported.

---

For more information about managing and including mail, see:

- the "Managing Mail" chapter in *Administering Oracle WebCenter Portal*
- the "Working with Mail" chapter in *Using Oracle WebCenter Portal*

### 35.1 Introduction to Mail

This section provides an overview of WebCenter Portal mail features and requirements. It includes the following subsections:

- **Section 35.1.1, "Understanding Mail"
- **Section 35.1.2, "Requirements for Mail"

#### 35.1.1 Understanding Mail

Mail functionality enables users to access the inbox of a mail server that supports Internet Message Access Protocol 4 (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.

This mail does not replace users’ mail clients; it simply enables users to access and compose mail in a single, collaborative environment.

With mail, 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 35–1 shows the Mail task flow at runtime. At the top of the view are three elements: a dropdown list that shows the number of mail messages 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.

![Mail at Runtime](image)

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 35–2).

![Message Filter](image)

Note: By default, All displays the 50 most recent messages. For information on how to increase this amount, see Section 35.3.2, “Configuring the Number of Mail Messages 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 35–3.
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 mail at runtime, see the "Working with Mail" section in *Using Oracle WebCenter Portal*.

### 35.1.2 Requirements for Mail

Mail requires a mail server that supports IMAP4 and SMTP. The Microsoft Exchange mail server is required for automatic creation of distribution lists when portals are created. In a Portal Framework application, this feature may not be desirable. To disable it, do not provide the LDAP (Active Directory) server details in the mail connection.

### 35.2 Basic Configuration for Mail

This section describes the steps required for adding mail functionality to your application. It includes the following subsections:

- **Section 35.2.1, "Configuration Roadmap - Mail"
- **Section 35.2.2, "Setting up Connections for Mail"
- **Section 35.2.3, "Adding Mail Functionality at Design Time"
- **Section 35.2.4, "Setting Security for Mail"
35.2.1 Configuration Roadmap - Mail

Use the roadmap in this section as a guide through the configuration process.

Figure 35–4 and Table 35–1 provide an overview of the prerequisites and tasks required to get mail working in Portal Framework applications.

**Figure 35–4 Configuring Mail for Portal Framework Applications**

**Table 35–1 Configuring Mail for Portal Framework Applications**

<table>
<thead>
<tr>
<th>Actor</th>
<th>Task</th>
<th>Sub-task</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrator</td>
<td>Install WebCenter Portal and the back-end components for mail</td>
<td>1.a For Microsoft Exchange 2007 only, follow additional configuration steps</td>
</tr>
<tr>
<td>Developer</td>
<td>Integrate mail in your Portal Framework application</td>
<td>2.a Configure a connection to your mail server in JDeveloper, associating the mail server with an external application</td>
</tr>
<tr>
<td></td>
<td>Add the Mail task flow to a page in JDeveloper</td>
<td>2.b Add the Mail task flow to a page in JDeveloper</td>
</tr>
</tbody>
</table>
35.2.2 Setting up Connections for Mail

Before you can use mail, you must first set up the connection to your mail server. This can be any mail server based on IMAP4 and SMTP protocol.

<table>
<thead>
<tr>
<th>Actor</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developer or Administrator</td>
<td>3. Deploy the Portal Framework application using one of the following tools:</td>
</tr>
<tr>
<td></td>
<td>• JDeveloper (Developer)</td>
</tr>
<tr>
<td></td>
<td>• Fusion Middleware Control (Administrator)</td>
</tr>
<tr>
<td></td>
<td>• WLST (Administrator)</td>
</tr>
<tr>
<td></td>
<td>• WLS Admin Control (Administrator)</td>
</tr>
<tr>
<td>Developer or Administrator</td>
<td>4. (Optional) Add/modify connection parameters using one of the following tools:</td>
</tr>
<tr>
<td></td>
<td>• JDeveloper, then redeploy the application (Developer)</td>
</tr>
<tr>
<td></td>
<td>• Fusion Middleware Control (Administrator)</td>
</tr>
<tr>
<td></td>
<td>• WLST (Administrator)</td>
</tr>
<tr>
<td>End User</td>
<td>5. Access mail by clicking <strong>Login to Mail</strong> on a Mail task flow, and entering your login credentials for the mail server</td>
</tr>
</tbody>
</table>

**Table 35–1 (Cont.) Configuring Mail for Portal Framework Applications**

**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 *Administering 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 mail.
2. In your application, under Application Resources, right-click **Connections**, then select **Mail** from the list.
3. Select to create the Mail 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 35–5**).
When configuring multiple mail accounts, you are not necessarily required to select this as the default connection. Keep in mind, however, that one connection must 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 35–6).
Basic Configuration for Mail

Integrating Mail

Figure 35–6  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 mail to create distribution lists reliably, the WebCenter Portal 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 74.13, "Working with External Applications."
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 mail 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 mail 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 35–7.
Figure 35–7  Email Address and Reply-To Additional Fields

- External applications allow 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.

- 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 portal distribution lists.

For detailed parameter information, see Table 35-2.

---

**Note:** For WebCenter Portal and mail to share an identity management system for setting up portal mailing lists, you must use Active Directory. For information about installing and configuring mail servers as the WebCenter Portal administrator, see *Administering Oracle WebCenter Portal*.

---

### Table 35–2 LDAP Directory Server Configuration Parameters

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LDAP Host</td>
<td>Enter the host name of the computer where the LDAP directory server is running.</td>
</tr>
<tr>
<td>LDAP Port</td>
<td>Enter the port on which the LDAP directory server listens.</td>
</tr>
<tr>
<td>LDAP Base DN</td>
<td>Enter the base distinguished name for the LDAP schema. For example, CN=Users,DC=oracle,DC=com.</td>
</tr>
<tr>
<td>LDAP Domain</td>
<td>Enter the domain to be appended to distribution list names. For example, if the domain value is set to oracle.com, then the Finance Project portal maintains a distribution list named <a href="mailto:FinanceProject@oracle.com">FinanceProject@oracle.com</a>.</td>
</tr>
<tr>
<td>LDAP Administrator User Name</td>
<td>Enter the user name of the LDAP directory server administrator. A valid user with privileges to make entries into the LDAP schema.</td>
</tr>
<tr>
<td>LDAP Administrator Password</td>
<td>Enter the password for the LDAP directory server administrator. The password is stored in a secured store.</td>
</tr>
<tr>
<td>LDAP Default User</td>
<td>Enter a comma-delimited list of user names to whom you want to grant moderator capabilities. These users become members of every portal distribution list that is created. The users specified must exist in the base LDAP schema (specified in the LDAP Base DN field).</td>
</tr>
<tr>
<td>ldap.secured</td>
<td>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.</td>
</tr>
</tbody>
</table>

11. Click Finish.

You can see the new mail connection under Application Resources - Connections.
35.2.3 Adding Mail Functionality at Design Time

This section explains a basic incorporation of mail functionality. It includes the following subsections:

- Section 35.2.3.1, "Mail Task Flows"
- Section 35.2.3.2, "How to Add Mail to your Application"

35.2.3.1 Mail Task Flows

The Mail task flow displays a mail inbox.

35.2.3.2 How to Add Mail to your Application

To add mail to your application:

1. Follow the steps described in Chapter 2, "Setting Up 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 74.3, "Configuring ADF Security."

4. Open the page on which to add mail functionality.


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.
### 35.2.3.3 How to Modify the Mail Task Flow Parameters

The Mail 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 35–8 shows an example of a Search task flow in the Executables section.

### Figure 35–8 Page Data Binding Definition

Page Data Binding Definition

This shows the Oracle ADF data bindings defined for your page. Select a binding to see its relation.

Page Definition File: portal/pageDefs/untitled3PageDef.xml

<table>
<thead>
<tr>
<th>Bindings and Executables</th>
<th>Contextual Events</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Model</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bindings</td>
<td><strong>Executables</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>variables</td>
<td></td>
</tr>
<tr>
<td></td>
<td>taskFlow-mailminview1</td>
<td></td>
</tr>
</tbody>
</table>

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 35–9, revise the binding parameter values as required.

---

**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 task flows that require connections to back-end servers, such as task flows from discussions or announcements.

- 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`.

```xml
<adf:adf-properties-child
 xmlns="http://xmlns.oracle.com/adf/config/properties">
  <adf-property name="adfAppUID" value="WCApp1-1234"/>
</adf:adf-properties-child>
```
5. When you are finished, click OK.

6. Save and run your page to see the results.

Table 35–3 describes the properties that are unique to the Mail task flow.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Task Flow</th>
</tr>
</thead>
<tbody>
<tr>
<td>tabularView</td>
<td>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.</td>
<td>Mail</td>
</tr>
</tbody>
</table>

Figure 35–10 depicts the Mail task flow where the region parameter tabularView is set to true.

### 35.2.4 Setting Security for Mail

Mail 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 74.3, “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 application point to the same identity store.

**Note:** Even if the application and the mail server share the same identity store, mail does not support identity propagation. Single sign-on functionality is enabled through the external application mechanism.

Mail works in a non-secured 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 74.13, "Working with External Applications."

### 35.3 Advanced Information for Mail

This section describes optional features available with mail. It includes the following subsections:

- Section 35.3.1, "Invoking the Mail Compose Page"
- Section 35.3.2, "Configuring the Number of Mail Messages Displayed"
- Section 35.3.3, "Troubleshooting Mail"

#### 35.3.1 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 35–1).

**Example 35–1 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 portal but want to provide the ability for portal members to send mail 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 portal members.
Example 35–2 shows some parameters for the Mail Compose page (ComposeView.jspx):

Example 35–2  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 35–3 shows how to hide the recipients fields (like To, Cc and Bcc) by setting the following action listener in ComposeView.jspx:

Example 35–3  Hiding Recipient Fields in the Mail Compose Page
<af:setActionListener from="#{'false'}" to="#{'pageFlowScope['collab.mail.message.showrecipients']'}"/>

35.3.2 Configuring the Number of Mail Messages Displayed

By default, the 50 most recent mail messages are displayed 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 35–4:

Example 35–4  Increasing the Number of Mail Messages 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.

This change applies to all users; that is, following Example 35–4, all users see 500 recent mail messages 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.

### 35.3.3 Troubleshooting Mail

This section describes common problems and solutions for mail.

**Problem**

Users are not able to retrieve their mail messages or send mail messages from within the Portal Framework application.

**Solution**

Ensure the following:

- Ensure that a Mail connection exists in your application.
- Ensure that the required Mail 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 within your Portal 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 mail, it throws an ExtApp Authorization Exception.

**Solution**

Mail works in a non-secured Portal 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 Portal Framework application to run in secure mode, then you must configure ADF security for your application.

**Problem**

When users receive mail in Portal 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](http://support.microsoft.com/kb/930468).
This chapter explains how to integrate polling functionality into a Portal Framework application at design time.

This chapter includes the following topics:

- Section 36.1, "Introduction to Polls"
- Section 36.2, "Basic Configuration for Polls"
- Section 36.3, "Advanced Information for Polls"

For more information about using polls, see the “Adding Polls to a Portal” chapter in *Building Portals with Oracle WebCenter Portal*.

### 36.1 Introduction to Polls

WebCenter Portal and Framework application users can 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 36.1.1, "Understanding Polls"
- Section 36.1.2, "Requirements for Polls"
- Section 36.1.3, "What Happens at Runtime"

### 36.1.1 Understanding Polls

In addition to taking available polls, users can 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

Polls is integrated with instant messaging and presence in the Polls Manager.
36.1.2 Requirements for Polls

In a Framework application, polls requires a connection to the WebCenter Portal database schema. For details about setting up a database connection, see Section 4.2.2, "Setting Up a Database Connection."

36.1.3 What Happens at Runtime

At runtime, users can quickly create polls with just a name and a question. Figure 36–1 shows a sample quick poll.

**Figure 36–1  Quick Poll**

[Quick Poll image]

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 polls, analyze results, clear results, and delete the poll (Figure 36–2).

**Figure 36–2  Polls Manager**

[Polls Manager image]

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 using polls at runtime, see the "Adding Polls to a Portal" chapter in Building Portals with Oracle WebCenter Portal.

36.2 Basic Configuration for Polls

This section describes required steps for adding this to your application. It includes the following subsections:

- Section 36.2.1, "Setting up Connections for Polls"
- Section 36.2.2, "Adding Polls Functionality at Design Time"
- Section 36.2.3, "Setting Security for Polls"
36.2.1 Setting up Connections for Polls

Polls requires a connection to the database where the WebCenter Portal schema is installed. For details about setting up a database connection, see Section 4.2.2, "Setting Up a Database Connection."

36.2.2 Adding Polls Functionality at Design Time

This section explains a basic incorporation of polls. It includes the following sections:

- Section 36.2.2.1, "Polls Task Flows"
- Section 36.2.2.2, "How to Add the Quick Poll Task Flow to Your Page"
- Section 36.2.2.3, "How to Modify the Polls Task Flow Parameters"

36.2.2.1 Polls Task Flows

Polls task flows are described in Table 36–1.

<table>
<thead>
<tr>
<th>Task Flow</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Poll - Quick Poll</td>
<td>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.</td>
</tr>
<tr>
<td>Polls - Polls Manager</td>
<td>This task flow allows users to perform administrative operations on polls. Any user who has access to this task flow can perform the administrative operations, such as edit or delete the poll.</td>
</tr>
<tr>
<td>Polls - Take Polls</td>
<td>This task flow displays the most recently-published available poll, unless it is set to a specific poll in the pollId parameter.</td>
</tr>
<tr>
<td>Polls - View Poll Results</td>
<td>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.</td>
</tr>
</tbody>
</table>

36.2.2.2 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 polls functionality.
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.

36.2.2.3 How to Modify the Polls Task Flow Parameters

Polls 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 36–4 shows an example of a Search task flow in the Executables section.

**Figure 36–3  Page Data Binding Definition**

![Page Data Binding Definition](image1)

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 36–4, revise the binding parameter values as required.

**Figure 36–4  Edit Task Flow Binding Dialog for Polls Manager Task Flow**

![Edit Task Flow Binding](image2)

5. When you are finished, click **OK**.

6. Save and run your page to see the results.
Table 36–2 describes the properties that are unique to the Polls task flows.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scope</td>
<td>The portal name from which polls are to be fetched. If this is supplied, then polls of that particular portal are shown. On the Home portal, when this parameter is not supplied, it fetches polls from all portals. This parameter appears in the properties for the Polls Manager task flow.</td>
</tr>
<tr>
<td>showUserDataOnly</td>
<td>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.</td>
</tr>
<tr>
<td>pollId</td>
<td>The poll to display. This parameter appears in the properties for the following task flows. ■ Take Polls ■ View Poll Results</td>
</tr>
<tr>
<td>showInEditMode</td>
<td>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.</td>
</tr>
</tbody>
</table>

### 36.2.3 Setting Security for Polls

Polls do 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 74.3, "Configuring ADF Security."

### 36.3 Advanced Information for Polls

This section describes optional features available with polls. It includes the following subsections:

- Section 36.3.1, "How to Add the Polls Manager Task Flow"
- Section 36.3.2, "How to Add the Take Polls Task Flow"
- Section 36.3.3, "How to Add the View Poll Results Task Flow"
- Section 36.3.4, "Using the Polls Data Controls"

### 36.3.1 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 Quick Poll task flow in Section 36.2.2, "Adding Polls Functionality at Design Time," but drag and drop Polls Manager onto the page.

### 36.3.2 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 Quick Poll task flow in Section 36.2.2, "Adding Polls Functionality at Design Time," but drag and drop Polls - Take Polls onto the page.

### 36.3.3 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 Quick Poll task flow in Section 36.2.2, "Adding Polls Functionality at Design Time," but drag and drop Polls - View Poll Results onto the page.

### 36.3.4 Using the Polls Data Controls

Use the Polls data controls to build a customized user interface for polls with a Framework application or a custom task flow.

Polls includes two data controls to customize the user interface:

- **Take Poll Data Control**
- **Polls Manager Data Control**

**See Also:** Section 4.1.3, "Using WebCenter Portal Data Controls" for information on how to consume data controls

#### 36.3.4.1 Take Poll Data Control

The Take Poll data control contains the following methods:

- **createResponse**
- **createResponseForLatestAvailablePoll**
- **getPublishedPolls**
- **getResults**
- **saveResponse**

**36.3.4.1.1 createResponse** This method lets users respond to a particular poll. Table 36–3 describes the required parameters for this method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pollId</td>
<td>String</td>
<td>The poll to display.</td>
</tr>
</tbody>
</table>
36.3.4.1.2 createResponseForLatestAvailablePoll This method lets users respond to the most recently-published poll. It has no parameters.

36.3.4.1.3 getPublishedPolls Table 36–4 describes the required parameters for this method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>startIndex</td>
<td>Integer</td>
<td>The index of the first matching result that should be included in the result set (0-n ... zero based). This is used for pagination.</td>
</tr>
<tr>
<td>numResults</td>
<td>Integer</td>
<td>The number or results to display.</td>
</tr>
</tbody>
</table>

36.3.4.1.4 getResults This method gets the poll results for a particular poll. Table 36–5 describes the required parameters for this method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pollId</td>
<td>String</td>
<td>The poll to display.</td>
</tr>
</tbody>
</table>

36.3.4.1.5 saveResponse Table 36–6 describes the required parameters for this method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>response</td>
<td>Response</td>
<td>The response object acquired from the data control calls to createResponse or createResponseForLatestAvailablePoll.</td>
</tr>
</tbody>
</table>

36.3.4.2 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
Advanced Information for Polls

- **publishPollOnDate**

### 36.3.4.2.1 addQuestion
This method creates polls with multiple choice questions. Table 36–7 describes the required parameters for this method.

<table>
<thead>
<tr>
<th>Table 36–7 addQuestion Input Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>pollId</td>
</tr>
<tr>
<td>question</td>
</tr>
<tr>
<td>rowOptions</td>
</tr>
<tr>
<td>allowMultipleSelection</td>
</tr>
</tbody>
</table>

### 36.3.4.2.2 addQuestion
This method creates complex polls with matrix-type questions. Table 36–8 describes the required parameters for this method.

<table>
<thead>
<tr>
<th>Table 36–8 addQuestion Input Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>pollId</td>
</tr>
<tr>
<td>questionType</td>
</tr>
<tr>
<td>question</td>
</tr>
<tr>
<td>rowOptions</td>
</tr>
<tr>
<td>columnOptions</td>
</tr>
<tr>
<td>isOptional</td>
</tr>
<tr>
<td>commentLabel</td>
</tr>
<tr>
<td>commentFieldHeight</td>
</tr>
</tbody>
</table>

### 36.3.4.2.3 closePoll
This method closes a particular poll. Table 36–9 describes the required parameters for this method.

<table>
<thead>
<tr>
<th>Table 36–9 closePoll Input Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>pollId</td>
</tr>
</tbody>
</table>

### 36.3.4.2.4 closePollAfterDate
This method closes a poll on a specific date. Table 36–10 describes the required parameters for this method.
36.3.4.2.5 closePollAfterDays This poll closes a poll at a set number of days after the poll is published. Table 36–11 describes the required parameters for this method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pollId</td>
<td>String</td>
<td>The poll to display.</td>
</tr>
<tr>
<td>days</td>
<td>Integer</td>
<td>Number of days, after the publish date, at which the poll should close.</td>
</tr>
</tbody>
</table>

36.3.4.2.6 closePollAfterResponses This method closes as poll after a set number of responses have been collected. Table 36–12 describes the required parameters for this method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pollId</td>
<td>String</td>
<td>The poll to display.</td>
</tr>
<tr>
<td>responses</td>
<td>Integer</td>
<td>Number of responses at which the poll should close.</td>
</tr>
</tbody>
</table>

36.3.4.2.7 createPoll This method creates a quick poll. Table 36–13 describes the required parameters for this method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the poll.</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>Description of the poll.</td>
</tr>
</tbody>
</table>

36.3.4.2.8 deletePoll This method deletes a particular poll. Table 36–14 describes the required parameters for this method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pollId</td>
<td>String</td>
<td>The poll to display.</td>
</tr>
</tbody>
</table>

36.3.4.2.9 editPoll Table 36–15 describes the required parameters for this method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pollInstance</td>
<td>Poll</td>
<td>If a poll instance is directly edited, then that instance can be supplied for editing.</td>
</tr>
</tbody>
</table>
36.3.4.2.10 **editPoll** This method edits a particular poll. Table 36–16 describes the required parameters for this method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pollId</td>
<td>String</td>
<td>The poll to display.</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the poll.</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>Description of the poll.</td>
</tr>
</tbody>
</table>

36.3.4.2.11 **findPoll** This method finds a particular poll. Table 36–17 describes the required parameters for this method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Poll instance ID</td>
</tr>
</tbody>
</table>

36.3.4.2.12 **getPolls** This method displays all polls, as defined in its parameters. Table 36–18 describes the required parameters for this method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>startIndex</td>
<td>Integer</td>
<td>The index of the first matching result that should be included in the result set (0-n ... zero based). This is used for pagination.</td>
</tr>
<tr>
<td>numResults</td>
<td>Integer</td>
<td>The number or results to display.</td>
</tr>
</tbody>
</table>

36.3.4.2.13 **getPollsByUser** This method displays polls created by a particular user. Table 36–19 describes the required parameters for this method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>createdBy</td>
<td>String</td>
<td>User name of the person who created the poll.</td>
</tr>
<tr>
<td>startIndex</td>
<td>Integer</td>
<td>The index of the first matching result that should be included in the result set (0-n ... zero based). This is used for pagination.</td>
</tr>
<tr>
<td>numResults</td>
<td>Integer</td>
<td>The number or results to display.</td>
</tr>
</tbody>
</table>

36.3.4.2.14 **getPollsByUserCount** Table 36–20 describes the required parameters for this method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>createdBy</td>
<td>String</td>
<td>User name of the person who created the poll.</td>
</tr>
</tbody>
</table>

36.3.4.2.15 **publishPoll** This method publishes a particular poll. Table 36–21 describes the required parameters for this method.
36.3.4.2.16 publishPollOnDate This method publishes a poll on a specific date. Table 36–22 describes the required parameters for this method.

Table 36–21 publishPoll Input Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pollId</td>
<td>String</td>
<td>The poll to display.</td>
</tr>
</tbody>
</table>

Table 36–22 publishPollOnDate Input Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pollId</td>
<td>String</td>
<td>The poll to display.</td>
</tr>
<tr>
<td>date</td>
<td>Date</td>
<td>Date at which the poll should be published.</td>
</tr>
</tbody>
</table>
Part VI contains the following chapters:

- Chapter 37, "Introducing the People Connections Service"
- Chapter 38, "Basic Configuration for the People Connections Service"
- Chapter 39, "People Connections Task Flow Binding Parameters"
- Chapter 40, "Using People Connections Data Controls and Java APIs"
- Chapter 41, "Integrating Worklists"
- Chapter 42, "Using the People Connections REST APIs"
This chapter describes the task flows associated with the People Connections service. The People Connections service provides social networking tools that enhance connection and communication in a project team and throughout an enterprise. The service provides its social networking features through Activity Stream, Connections, Feedback, Message Board, Profile, and Publisher task flows.

This chapter includes the following topics:

- Section 37.1, "Overview of the People Connections Service"
- Section 37.2, "Troubleshooting the People Connections Service"

### 37.1 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. The following 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:

- **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

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 "Configuring the Identity Store" chapter in *Administering Oracle WebCenter Portal*. For information about creating test users in `jazn-data.xml`, see Section 22.2.2, "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 37.1.1, "People Connections Service Task Flows"
- Section 37.1.2, "People Connections Service Requirements"

### 37.1.1 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. 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 37.1.1.1, "Activity Stream"
- Section 37.1.1.3, "Feedback"
- Section 37.1.1.4, "Message Board"
- Section 37.1.1.2, "Connections"
- Section 37.1.1.5, "Profile"
- Section 37.1.1.6, "The Publisher Task Flow"

**See Also:** For information about the People Connections service at runtime, see the associated chapters in *Using Oracle WebCenter Portal*.

For information about People Connections task flows, see Section 37.1.1, "People Connections Service Task Flows."

### 37.1.1.1 Activity Stream

The Activity Stream feature tracks the application activities of a user's connections. Table 37–1 lists the types of activities that can be tracked by the Activity Stream.

<table>
<thead>
<tr>
<th>Service</th>
<th>Tracked Activities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Announcements</td>
<td>■ Create announcement</td>
</tr>
<tr>
<td></td>
<td>■ Edit announcement</td>
</tr>
<tr>
<td>Blogs</td>
<td>■ Create blog</td>
</tr>
<tr>
<td></td>
<td>■ Update blog</td>
</tr>
<tr>
<td>Connections</td>
<td>■ Invitations to connect</td>
</tr>
<tr>
<td></td>
<td>■ People are connected</td>
</tr>
<tr>
<td>Discussions</td>
<td>■ Create forum</td>
</tr>
<tr>
<td></td>
<td>■ Create topic</td>
</tr>
<tr>
<td></td>
<td>■ Reply to topic</td>
</tr>
<tr>
<td>Documents</td>
<td>■ Create document</td>
</tr>
<tr>
<td></td>
<td>■ Edit document</td>
</tr>
<tr>
<td></td>
<td>■ Add tag</td>
</tr>
<tr>
<td></td>
<td>■ Remove tag</td>
</tr>
<tr>
<td>Events</td>
<td>■ Create an event</td>
</tr>
<tr>
<td></td>
<td>■ Edit an event</td>
</tr>
</tbody>
</table>

**Table 37–1 Activities Tracked by Activity Stream**
Overview of the People Connections Service

Introducing the People Connections Service

Activity Stream compares somewhat to the Recent Activities service, which also tracks and reports on application activities (for more information, see Chapter 51, "Integrating Recent Activities"). 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

<table>
<thead>
<tr>
<th>Table 37–1 (Cont.) Activities Tracked by Activity Stream</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Service</strong></td>
</tr>
<tr>
<td>-----------------</td>
</tr>
<tr>
<td>Feedback</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Lists</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Message Board</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Pages</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Profiles</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Tagging</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
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 application is accessed through an Oracle HTTP Server.

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. You can set a parameter associated with an Activity Stream task flow instance to omit file previews (for more information, see Section 39.3, "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 37–1).

**Figure 37–1 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 (Figure 37–2). Users who want to see all activities can click a More link to navigate to the main view (Activity Stream) of the task flow.
Introducing the People Connections Service

37.1.1.2 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 way your connections are displayed.

The following 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 37–3).

See Also: For information about Activity Stream task flows at runtime, see the "Tracking Portal Activities" chapter in Using Oracle WebCenter Portal.
Connections - Card—Provides a choice of the following views:

- vcard displays a connection’s photo, job title, and status message (Figure 37–4).
- 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 39, "People Connections Task Flow Binding Parameters").
Introducing the People Connections Service

Figure 37–4  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 37–5).

Figure 37–5  Connections - Quick View Task Flow

See Also: For information about Connections task flows at runtime, see the "Managing Your Contacts" chapter in Using Oracle WebCenter Portal.

37.1.1.3 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 when given the proper permissions.

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 37–6).
Feedback - Quick View—Provides users with a truncated view of the Feedback task flow, where only the most recent posts are shown (Figure 37–7). 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.

See Also: For information about Feedback task flows at runtime, see the "Working with Feedback and the Message Board" chapter in Using Oracle WebCenter Portal.

37.1.1.4 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—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 37–8).
Figure 37–8  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 (Figure 37–9). Users can click a More link to navigate to the Detailed View. |
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 37–10).
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 37–11).

Figure 37–11 Profile - Snapshot Task Flow
Profile Gallery—Provides a summary view of each People Connections task flow (Figure 37–12). For example, it includes Activity Stream - Quick View, Connections - Quick View, and so on.

**Figure 37–12  Profile Gallery 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 38.2, "How to Add People Connections Task Flows to a Page," and see Chapter 39, "People Connections Task Flow Binding Parameters."

---

See Also: For information about Profile task flows at runtime, see the "Managing Your Profile" chapter in Using Oracle WebCenter Portal.

### 37.1.1.6 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 37–13).

**Figure 37–13  Publisher Task Flow**
37.1.2 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 6.1, "Creating a New WebCenter Portal Framework Application").

2. Set up the WebCenter schema, and create a database connection to it (for more information, see Chapter 4, "Preparing Your Application for WebCenter Portal Tools and Services").

3. Create a JSF (.jspx) page (for more information, see Section 15.2, "Creating Pages in a WebCenter Portal Framework Application").

4. Secure your application and create some test users (for more information, see Chapter 74, "Securing Your WebCenter Portal Framework Application" and Section 22.2.2, "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 38.2, "How to Add People Connections Task Flows to a Page").

6. Provide required values for task flow bindings (for more information, see Chapter 39, "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.

37.2 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 might 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.
Basic Configuration for the People Connections Service

This chapter describes how to work with the People Connections service, including adding People Connections task flows to a page, setting People Connections security, and configuring site-level People Connections settings.

This chapter includes the following topics:

- Section 38.1, "How to Set Up a Database Connection for the People Connections Service"
- Section 38.2, "How to Add People Connections Task Flows to a Page"
- Section 38.3, "Setting Security for the People Connections Service"
- Section 38.4, "Establishing Site-Level Settings for People Connections Features"

38.1 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 tools and services, see Section 4.2.2, "Setting Up a Database Connection."

38.2 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 37.1.2, "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 39, "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 39, "People Connections Task Flow Binding Parameters."

---

### 38.3 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 74.3, "Configuring ADF Security."

- To create test users, follow the steps outlined in Section 22.2.2, "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:

```xml
<property name="[property-name]" value="[property-value]"/>
```

For example, to add `business_email` as `monty@example.com` for the user `monty`, add:

```xml
<user>
  <name>monty</name>
  <display-name>monty</display-name>
  <credentials>{903}EfEepsI6I/TrKthpcmjIJsr1P36Ysr/j</credentials>
  <property name="business_email" value="monty@example.com"/>
</user>
```

38.4 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 38.4.1, "Understanding the People Connections Site-Level Setting Scripts"
- **Section 38.4.2, "Supported Site-Level Settings for People Connections Features"
- **Section 38.4.3, "Example: Configuring Connections to Accept Invitations Automatically"**

### 38.4.1 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 38.4.1.1, "Preparing and Running settings-insert.sql"
- **Section 38.4.1.2, "Preparing and Running settings-update.sql"**

#### 38.4.1.1 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 38.4.3, "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.

38.4.1.2 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 38.4.3, "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 38.4.2, "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.

### 38.4.2 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 38.4.2.1, "Activity Stream Site-Level Settings"
- Section 38.4.2.2, "Connections Site-Level Settings"
- Section 38.4.2.3, "Feedback Site-Level Settings"
- Section 38.4.2.4, "Message Board Site-Level Settings"
- Section 38.4.2.5, "Profile Site-Level Settings"

#### 38.4.2.1 Activity Stream Site-Level Settings

Table 38–1 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>
<thead>
<tr>
<th>Setting Key</th>
<th>Description</th>
<th>Valid Site-Level Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>accesscontrol.value.accessControlLevel</td>
<td>Specifies who can view a user’s Activity Stream.</td>
<td>SELF, CONNECTIONS, USERS (that is, authenticated users), PUBLIC</td>
</tr>
<tr>
<td>accesscontrol.endUserConfigurable</td>
<td>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.</td>
<td>+, −, + = TRUE, − = FALSE</td>
</tr>
</tbody>
</table>
**Establishing Site-Level Settings for People Connections Features**

38.4.2.2 Connections Site-Level Settings

Table 38–1 (Cont.) Site-Level Settings for Activity Stream

<table>
<thead>
<tr>
<th>Setting Key</th>
<th>Description</th>
<th>Valid Site-Level Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>connectionsActivities.value</td>
<td>Specifies whether the Home portal activities of a user's connections are included in the user's Activity Stream. Users can override this setting on a task flow instance.</td>
<td>NONE, ALL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NONE=No activities of the user's connections are shown.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ALL=All activities of the user's connections are shown.</td>
</tr>
<tr>
<td>groupSpaceActivities.value</td>
<td>Specifies whether any portal activities of a user's connections are included in the user's Activity Stream. Users can override this setting on a task flow instance.</td>
<td>GSNONE, GSALL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GSNONE=No portal activities are shown.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GSALL=Activities are shown for all of the portals of which the user is a member and on which the user has view access.</td>
</tr>
<tr>
<td>servicePublishedSettings[serviceId]</td>
<td>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.</td>
<td>+, -</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+=TRUE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-=FALSE</td>
</tr>
</tbody>
</table>

**38.4.2.2 Connections Site-Level Settings**

Table 38–2 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.
38.4.2.3 Feedback Site-Level Settings

Table 38–3 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>
<thead>
<tr>
<th>Setting Key</th>
<th>Description</th>
<th>Valid Site-Level Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>visibility.accessControlSettings.accessControlLevel</td>
<td>Specifies who can view a user's received Feedback.</td>
<td>SELF, CONNECTIONS, USERS (that is, authenticated users), PUBLIC</td>
</tr>
<tr>
<td>visibility.endUserConfigurable</td>
<td>Specifies whether individual users can override 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.</td>
<td>+, - +=TRUE -=FALSE</td>
</tr>
<tr>
<td>addControl.accessControlSettings.accessControlLevel</td>
<td>Specifies who can give Feedback to a user.</td>
<td>SELF, CONNECTIONS, USERS (that is, authenticated users), PUBLIC</td>
</tr>
<tr>
<td>addControl.endUserConfigurable</td>
<td>Specifies whether individual users can override 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.</td>
<td>+, - +=TRUE -=FALSE</td>
</tr>
</tbody>
</table>
Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper

38.4.2.4 Message Board Site-Level Settings

Table 38–4 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 38–4 Site-Level Settings For Message Board

<table>
<thead>
<tr>
<th>Setting Key</th>
<th>Description</th>
<th>Valid Site-Level Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>miniViewRowCount.value</td>
<td>Specifies the number of messages to show in a Message Board - Quick View task flow.</td>
<td>Positive integer</td>
</tr>
<tr>
<td>miniViewRowCount.endUserConfigurable</td>
<td>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.</td>
<td>+, −</td>
</tr>
<tr>
<td>postedUserActions.delete</td>
<td>Specifies whether users are allowed to delete the Feedback they leave for other users.</td>
<td>+, −</td>
</tr>
<tr>
<td>postingUserActions.delete</td>
<td></td>
<td>+ = TRUE</td>
</tr>
<tr>
<td>postingUserActions.delete</td>
<td></td>
<td>− = FALSE</td>
</tr>
</tbody>
</table>

![Table 38–3 (Cont.) Site-Level Settings for Feedback](image-url)
Table 38–4 (Cont.) Site-Level Settings For Message Board

<table>
<thead>
<tr>
<th>Setting Key</th>
<th>Description</th>
<th>Valid Site-Level Values</th>
</tr>
</thead>
</table>
| 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  |

38.4.2.5 Profile Site-Level Settings

Table 38–5 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.

Note: If Oracle WebCenter Portal is connected to embedded LDAP (the default), you must disable profile editing or users will see an error when they try to edit their profiles.

Table 38–5 Site-Level Settings for Profile

<table>
<thead>
<tr>
<th>Setting Key</th>
<th>Description</th>
<th>Valid Site-Level Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>general-profile-settings.logActivity</td>
<td>Specifies whether profile updates, such as uploading a photo, updating personal status, changing profile attributes, should result in activities getting published in Activity Stream.</td>
<td>true, false</td>
</tr>
<tr>
<td>general-profile-settings.personalWebVisibility</td>
<td>Specifies who can view a user's Profile Gallery.</td>
<td>SELF, CONNECTIONS, USERS (that is, authenticated users), PUBLIC</td>
</tr>
<tr>
<td>general-profile-settings.pwVisibilityPersonalizable</td>
<td>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.</td>
<td>true, false</td>
</tr>
</tbody>
</table>
Establishing Site-Level Settings for People Connections Features

Table 38–5  (Cont.) Site-Level Settings for Profile

<table>
<thead>
<tr>
<th>Setting Key</th>
<th>Description</th>
<th>Valid Site-Level Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>general-profile-settings.view.name.profileSectionName.access-control-level</td>
<td>Specifies who can view the Profile section identified by profileSectionName. For example: general-profile-settings.view.name.personalInfo.access-control-level</td>
<td>SELF, USERS (that is, authenticated users), PUBLIC. <strong>Note:</strong> CONNECTIONS is not a valid value for this setting.</td>
</tr>
<tr>
<td>general-profile-settings.view.name.profileSectionName.allow-personalize-visibility</td>
<td>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 38–6, &quot;Profile Section Names&quot;.</td>
<td>true, false</td>
</tr>
<tr>
<td>general-profile-settings.view.name.profileSectionName.allow-user-edit</td>
<td>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</td>
<td>true, false</td>
</tr>
<tr>
<td>ootb-view-edit-settings.view.name.profileSectionName.profileFieldName.allow-edit</td>
<td>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</td>
<td>true, false</td>
</tr>
</tbody>
</table>

Table 38–6  Profile Section Names

<table>
<thead>
<tr>
<th>Section Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic</td>
<td>Profile summary These details are discoverable in an application search.</td>
</tr>
<tr>
<td>employee</td>
<td>Employee detail</td>
</tr>
<tr>
<td>businessContact</td>
<td>Business contact information</td>
</tr>
</tbody>
</table>
Table 38–6  (Cont.) Profile Section Names

<table>
<thead>
<tr>
<th>Section Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>personalInfo</td>
<td>Personal information</td>
</tr>
</tbody>
</table>

Table 38–7  Profile Field Names

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>email</td>
<td>Business email address</td>
</tr>
<tr>
<td>displayName</td>
<td>User display name</td>
</tr>
<tr>
<td>Dept</td>
<td>Office department</td>
</tr>
<tr>
<td>title</td>
<td>Job title</td>
</tr>
<tr>
<td>phone</td>
<td>Business phone number</td>
</tr>
<tr>
<td>timeZone</td>
<td>Time zone</td>
</tr>
<tr>
<td>photo</td>
<td>User photo</td>
</tr>
<tr>
<td>persStatNote</td>
<td>Personal status message</td>
</tr>
<tr>
<td>description</td>
<td>About me</td>
</tr>
<tr>
<td>empType</td>
<td>Employee type</td>
</tr>
<tr>
<td>empNo</td>
<td>Employee number</td>
</tr>
<tr>
<td>prefLang</td>
<td>User's preferred language</td>
</tr>
<tr>
<td>organization</td>
<td>Employee's organization</td>
</tr>
<tr>
<td>expertise</td>
<td>Employee's expertise</td>
</tr>
<tr>
<td>fax</td>
<td>Fax number</td>
</tr>
<tr>
<td>mobile</td>
<td>Mobile/cell phone number</td>
</tr>
<tr>
<td>pager</td>
<td>Page number</td>
</tr>
<tr>
<td>street</td>
<td>Office address: street</td>
</tr>
<tr>
<td>city</td>
<td>Office address: city</td>
</tr>
<tr>
<td>state</td>
<td>Office address: state</td>
</tr>
<tr>
<td>poBox</td>
<td>Office address: P.O. box</td>
</tr>
<tr>
<td>poCode</td>
<td>Office address: ZIP/PIN/P.O. code</td>
</tr>
<tr>
<td>country</td>
<td>Office address: country</td>
</tr>
<tr>
<td>homeAdd</td>
<td>Home address</td>
</tr>
<tr>
<td>homePhone</td>
<td>Home phone number</td>
</tr>
<tr>
<td>dob</td>
<td>Date of birth</td>
</tr>
<tr>
<td>maidenName</td>
<td>Maiden name/surname before marriage</td>
</tr>
<tr>
<td>doh</td>
<td>Date of hire</td>
</tr>
</tbody>
</table>

38.4.3 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,
To change the setting, the administrator performs the following steps:

1. **Open the settings-insert.sql script and uncomment the following lines:**

   ```sql
   -- 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', '-')
   -- ;
   
   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:**

   ```sql
   -- 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'
   -- ;
   
   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:

   ```sql
   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 38.4.2, “Supported Site-Level Settings for People Connections Features.”
This chapter describes how to revise task flow parameter values at design time and provides a table of binding parameters associated with People Connections task flows.

This chapter includes the following topics:

- Section 39.1, "About People Connections Task Flow Binding Parameters"
- Section 39.2, "How to Revise People Connections Task Flow Binding Parameters"
- Section 39.3, "People Connections Task Flow Binding Parameters"
- Section 39.4, "About the Activity Stream Advanced Query Option"

### 39.1 About People Connections Task Flow Binding Parameters

Each People Connections task flow has a set of required and optional task flow binding parameters that capture 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 38.2, "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.

### 39.2 How to Revise People Connections Task Flow Binding Parameters

After you have added a task flow to a page, you might 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 39–1) to open the Edit Task Flow Binding dialog (Figure 39–2).

**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 39–1  Bindings View of an Application Page**

**Page Data Binding Definition**

This shows the Oracle ADF data bindings defined for your page. Select a binding to see its relationship to the underlying page data.

**Page Definition File:** *wc_webcenter_second/view/pageDefs/people_consPageDef.xml*

**Figure 39–2  Edit Task Flow Binding Dialog**

**Edit Task Flow Binding**

**Task Flow:** */oracle/webcenter/peopleconnections/view/jsf/regions/profile-snapshot.xml#profile-snapshot*

**Input Parameters Map:**

**Input Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>userId</td>
<td></td>
</tr>
<tr>
<td>imageSize</td>
<td></td>
</tr>
<tr>
<td>photoUploadAllowed</td>
<td></td>
</tr>
<tr>
<td>statusUpdateAllowed</td>
<td></td>
</tr>
</tbody>
</table>

* = Required

[ oracle/webcenter/peopleconnections/view/jsf/regions/profile-snapshot.xml#profile-snapshot ]

3. Revise binding parameter values (for more information, see Section 39.3, "People Connections Task Flow Binding Parameters").

39-2  Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper
4. Click OK to save your changes and exit the dialog.
5. Save and run your page to see the results.

39.3 People Connections Task Flow Binding Parameters

Table 39–1 lists and describes task flow binding parameters applicable to the People Connections service.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>__followEnforced</td>
<td>A means of enforcing the follow logic when querying Activity Stream</td>
</tr>
<tr>
<td></td>
<td>• Enter #{true} to establish that the user must follow the business object</td>
</tr>
<tr>
<td></td>
<td>to access streamed activities, even when the object's activities are</td>
</tr>
<tr>
<td></td>
<td>queried in an advance query clause. The default is true.</td>
</tr>
<tr>
<td></td>
<td>• Enter #{false} to omit this requirement. The activities streamed for</td>
</tr>
<tr>
<td></td>
<td>Fusion business objects are returned as far as required by the advanced</td>
</tr>
<tr>
<td></td>
<td>query, no matter whether the user follows the objects.</td>
</tr>
<tr>
<td></td>
<td>This parameter (for Fusion application business objects) is used in</td>
</tr>
<tr>
<td></td>
<td>conjunction with the advancedQuery parameter.</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the Activity Stream task flow.</td>
</tr>
<tr>
<td>__objectExtHandlerEnabled</td>
<td>A means of using the object extension handler to process Fusion application</td>
</tr>
<tr>
<td></td>
<td>business objects</td>
</tr>
<tr>
<td></td>
<td>• Enter #{true} to use the object extension handler to process business</td>
</tr>
<tr>
<td></td>
<td>objects.</td>
</tr>
<tr>
<td></td>
<td>• Enter #{false} to omit using the object extension handler. False is the</td>
</tr>
<tr>
<td></td>
<td>default.</td>
</tr>
<tr>
<td></td>
<td>When you enter #{true}, then, before querying activities for the business</td>
</tr>
<tr>
<td></td>
<td>objects, the Activity Stream uses the object extension handler to process</td>
</tr>
<tr>
<td></td>
<td>the business objects. The handler checks whether the user has permission to</td>
</tr>
<tr>
<td></td>
<td>access the business object. If the user does not have permission, the object</td>
</tr>
<tr>
<td></td>
<td>is removed from the list, and, none activities are retrieved from this object.</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the following task flows:</td>
</tr>
<tr>
<td></td>
<td>• Activity Stream</td>
</tr>
<tr>
<td></td>
<td>• Activity Stream - Quick View</td>
</tr>
<tr>
<td>__objects</td>
<td>A comma separated list of objects for which to show activities</td>
</tr>
<tr>
<td></td>
<td>Enter objects in the format object-id;object-type;service-id. Typically</td>
</tr>
<tr>
<td></td>
<td>used only for followed objects.</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the Activity Stream task flow.</td>
</tr>
<tr>
<td>__objectSecured</td>
<td>A means of enforcing a security check on a Fusion application business</td>
</tr>
<tr>
<td></td>
<td>object.</td>
</tr>
<tr>
<td></td>
<td>• Enter #{true} to enforce the security check.</td>
</tr>
<tr>
<td></td>
<td>• Enter #{false} to omit the security check. False is the default.</td>
</tr>
<tr>
<td></td>
<td>The security check is performed by the resource authorizer that is</td>
</tr>
<tr>
<td></td>
<td>implemented by the Fusion application. If a user has no permission on a</td>
</tr>
<tr>
<td></td>
<td>business object, the activities related to that business object are filtered</td>
</tr>
<tr>
<td></td>
<td>out.</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the Activity Stream task flow.</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the following task flows:</td>
</tr>
<tr>
<td></td>
<td>• Activity Stream</td>
</tr>
<tr>
<td></td>
<td>• Activity Stream - Quick View</td>
</tr>
</tbody>
</table>
### Table 39–1 (Cont.) People Connections Service Task Flow Binding Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>advancedQuery</td>
<td>A field for specifying a custom query to filter streamed items. For more information, see Section 39.4, &quot;About the Activity Stream Advanced Query Option.&quot; This parameter is associated with the Activity Stream task flow.</td>
</tr>
<tr>
<td>connectionListName</td>
<td>The name of a connections list. This parameter is associated with the task flow Connections - Card.</td>
</tr>
<tr>
<td>currentView</td>
<td>The view to display by default. Valid values include:</td>
</tr>
<tr>
<td></td>
<td>- connections—(the default value) a list of connections</td>
</tr>
<tr>
<td></td>
<td>- receivedInvitations—a list of connections invitations you received</td>
</tr>
<tr>
<td></td>
<td>- sentInvitations—a list of connections invitations you sent</td>
</tr>
<tr>
<td></td>
<td>- people—a search field for finding people with whom to connect</td>
</tr>
<tr>
<td></td>
<td>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.</td>
</tr>
<tr>
<td>displayCount</td>
<td>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:</td>
</tr>
<tr>
<td></td>
<td>- Feedback</td>
</tr>
<tr>
<td></td>
<td>- Feedback - Quick View</td>
</tr>
<tr>
<td></td>
<td>- Message Board</td>
</tr>
<tr>
<td></td>
<td>- Message Board - Quick View</td>
</tr>
<tr>
<td>displayMessageSize</td>
<td>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.</td>
</tr>
<tr>
<td>display_maxConnections</td>
<td>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:</td>
</tr>
<tr>
<td></td>
<td>- Connections - Card</td>
</tr>
<tr>
<td></td>
<td>- Connections - Quick View</td>
</tr>
</tbody>
</table>
Table 39–1 (Cont.) People Connections Service Task Flow Binding Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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 if the task flow when there are more connections than can 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 |
| 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. |
| 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. |
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 sriraphy).

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 39–1  (Cont.) People Connections Service Task Flow Binding Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hideDocumentUploader</td>
<td>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.</td>
</tr>
</tbody>
</table>
| 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 |
| hideName            | A means of showing or hiding the user name for a connection Enter true to hide the name. Enter false to show it. This parameter is associated with the Connections - Card task flow |
### Table 39–1 (Cont.) People Connections Service Task Flow Binding Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hidePublisher</td>
<td>A means of showing or hiding the message entry field and the upload file and URL controls (the Publisher)</td>
</tr>
<tr>
<td></td>
<td>- Enter #{true} to disable the display of the message entry field in a given task flow instance.</td>
</tr>
<tr>
<td></td>
<td>- Enter #{false} to allow the display of the message entry field in a given task flow instance.</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the following task flows:</td>
</tr>
<tr>
<td></td>
<td>- Message Board</td>
</tr>
<tr>
<td></td>
<td>- Message Board - Quick View</td>
</tr>
<tr>
<td>hideScopePicker</td>
<td>Specifies whether the Share something with option is shown or hidden</td>
</tr>
<tr>
<td></td>
<td>- Check for #{true} to hide the Share something with option.</td>
</tr>
<tr>
<td></td>
<td>- Clear for #{false} to show the Share something with option.</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the Publisher task flow.</td>
</tr>
<tr>
<td>hideShare</td>
<td>A means of showing or hiding the Share link on a streamed activity</td>
</tr>
<tr>
<td></td>
<td>Enter true to hide the Share link. Enter false to show it.</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the Activity Stream task flow.</td>
</tr>
<tr>
<td>hideSpacesUI</td>
<td>Specifies whether to show the Portals UI in the Activity Stream and Activity Stream - Quick View Configuration dialogs. The Portals UI enables users to specify whether to show activities from all portals, the portals of which they are a member, no portals, or just the portal on which the task flow is placed.</td>
</tr>
<tr>
<td></td>
<td>- Enter #{true} to show the Portals UI. This is the default.</td>
</tr>
<tr>
<td></td>
<td>- Enter #{false} to hide the Portals UI.</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the following task flows:</td>
</tr>
<tr>
<td></td>
<td>- Activity Stream</td>
</tr>
<tr>
<td></td>
<td>- Activity Stream - Quick View</td>
</tr>
<tr>
<td>hintsTextKey</td>
<td>Specifies the resource bundle class and message key for hint text</td>
</tr>
<tr>
<td></td>
<td>Use the format key[,RBClass], <em>EMPTY</em> as the predefined key for no hint text. This is the default value.</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the Publisher task flow.</td>
</tr>
<tr>
<td>imageSize</td>
<td>The display size of the profile photo</td>
</tr>
<tr>
<td></td>
<td>Enter one of the following values:</td>
</tr>
<tr>
<td></td>
<td>- ORIGINAL</td>
</tr>
<tr>
<td></td>
<td>- LARGE</td>
</tr>
<tr>
<td></td>
<td>- MEDIUM</td>
</tr>
<tr>
<td></td>
<td>- SMALL</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the Profile - Snapshot task flow.</td>
</tr>
<tr>
<td>isUpdateStatus</td>
<td>A means of dedicating the task flow instance to updating the current user's Profile status message</td>
</tr>
<tr>
<td></td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the Publisher task flow.</td>
</tr>
</tbody>
</table>
### Table 39–1 (Cont.) People Connections Service Task Flow Binding Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>keepOpenAfterPublish</td>
<td>Specifies whether the text box remains active after a user clicks the Publish button</td>
</tr>
<tr>
<td></td>
<td>■ Enter #{true} to keep the text box open after a message is published.</td>
</tr>
<tr>
<td></td>
<td>■ Enter #{false} to close the text box after a message is published.</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the Publisher task flow.</td>
</tr>
<tr>
<td>launchStyle</td>
<td>Specifies the profile launch style. This parameter is considered only if Profile Launched On Selection is selected.</td>
</tr>
<tr>
<td></td>
<td>Enter one of the following:</td>
</tr>
<tr>
<td></td>
<td>■ snapshot—displays a user's profile snapshot when the user's name is clicked.</td>
</tr>
<tr>
<td></td>
<td>■ profile—displays user's Profile page when the user's name is clicked.</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the task flow Connections - Card.</td>
</tr>
<tr>
<td>messageType</td>
<td>Types of messages to display. Applicable only if viewing own's messageboard.</td>
</tr>
<tr>
<td></td>
<td>Specifies the types of messages to display.</td>
</tr>
<tr>
<td></td>
<td>Enter one of the following:</td>
</tr>
<tr>
<td></td>
<td>■ all—displays both public and private messages. This is the default.</td>
</tr>
<tr>
<td></td>
<td>■ private—displays only private messages.</td>
</tr>
<tr>
<td></td>
<td>■ public—displays only public messages.</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the following task flows:</td>
</tr>
<tr>
<td></td>
<td>■ Message Board</td>
</tr>
<tr>
<td></td>
<td>■ Message Board - Quick View</td>
</tr>
<tr>
<td>numberOfRows_list</td>
<td>The number of rows to show when the Connections task flow instance is set to list view</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the Connections task flow.</td>
</tr>
<tr>
<td>objectId</td>
<td>The ID of an object to associate with a published message</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the Publisher task flow.</td>
</tr>
<tr>
<td>objectType</td>
<td>The type of an object to associate with a published message</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the Publisher task flow.</td>
</tr>
<tr>
<td>orgBreadCrumbsShown</td>
<td>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</td>
</tr>
<tr>
<td></td>
<td>Enter #{true} to show organizational breadcrumbs. Enter #{false} to omit them.</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the Profile - Snapshot task flow.</td>
</tr>
<tr>
<td>pageSize</td>
<td>The number of items to stream in a given task flow instance</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the following task flows:</td>
</tr>
<tr>
<td></td>
<td>■ Activity Stream</td>
</tr>
<tr>
<td></td>
<td>■ Activity Stream - Quick View</td>
</tr>
</tbody>
</table>
### Table 39–1  (Cont.) People Connections Service Task Flow Binding Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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 Stream - Quick View task flow.  |
| 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, see Appendix G, “Expression Language Expressions.”  
  This parameter is associated with the Publisher task flow.  |
### People Connections Task Flow Binding Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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 and Tool 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 and Tool IDs".  
This parameter is associated with the Publisher task flow. |
| showViewListOfAll    | Specifies whether the footer includes a **See all your connections** link. If this parameter is enabled, the task flows shows no pagination.  
- Enter \#(true) to show the **See all your connections** link.  
- Enter \#(false) to hide the **See all your connections** link.  
This parameter is associated with the task flow Connections - Card. |
| spaceName            | The portal to which to publish messages  
If your application does not support portals, do not provide a value for this parameter.  
This parameter is associated with the following task flows:  
- Message Board  
- Message Board - Quick View  
- Publisher |
| spaces               | A comma-separated list of names or GUIDs of portals from which to stream activities  
Use this parameter to limit the display of streamed activities to only those associated with the specified portal or portals.  
If your application does not support portals, 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. |
39.4 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 39–2 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 39–2. For more information, see Appendix G, "Expression Language Expressions."

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>toDate</td>
<td>The end date of a date range within which to show feedback messages</td>
</tr>
<tr>
<td></td>
<td>Enter dates in the format YYYY-MM-DD.</td>
</tr>
<tr>
<td></td>
<td>See also fromDate.</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the following task flows:</td>
</tr>
<tr>
<td></td>
<td>■ Feedback</td>
</tr>
<tr>
<td></td>
<td>■ Feedback - Quick View</td>
</tr>
<tr>
<td></td>
<td>■ Message Board</td>
</tr>
<tr>
<td>uploadDocumentOnly</td>
<td>A means of limiting the Publisher task flow to its document upload feature</td>
</tr>
<tr>
<td></td>
<td>Enter #{true} to limit the Publisher task flow to be only a document</td>
</tr>
<tr>
<td></td>
<td>uploader. Enter #{false} to expose all Publisher features.</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the Publisher task flow.</td>
</tr>
<tr>
<td>userId</td>
<td>The current user ID</td>
</tr>
<tr>
<td></td>
<td>Enter #{securityContext.userName} to return the current user.</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the following task flows:</td>
</tr>
<tr>
<td></td>
<td>■ Connections - Card</td>
</tr>
<tr>
<td></td>
<td>■ Connections - Quick View</td>
</tr>
<tr>
<td></td>
<td>■ Feedback - Quick View</td>
</tr>
<tr>
<td></td>
<td>■ Message Board - Quick View</td>
</tr>
<tr>
<td></td>
<td>■ Profile - Snapshot</td>
</tr>
<tr>
<td>userName</td>
<td>The name of the user to whom to publish messages</td>
</tr>
<tr>
<td></td>
<td>This value is supplied by default. We recommend that you do not change</td>
</tr>
<tr>
<td></td>
<td>the default value, #{o_w_w_i_v_b_resourceViewerBean.username}.</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the Publisher task flow.</td>
</tr>
<tr>
<td>viaUser</td>
<td>The user name of the person who provided the object the current user is</td>
</tr>
<tr>
<td></td>
<td>sharing</td>
</tr>
<tr>
<td></td>
<td>For example, John is sharing a document with everyone that Jane originally</td>
</tr>
<tr>
<td></td>
<td>shared with him. In this case, Jane is the via user.</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the Publisher task flow.</td>
</tr>
</tbody>
</table>
Table 39–2  Supported Database Objects for Constructing a SQL WHERE Clause

<table>
<thead>
<tr>
<th>Database Object</th>
<th>Alias Prefix</th>
<th>Supported Fields/Columns</th>
</tr>
</thead>
</table>
| 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 39–3.  
| 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 39–3  Activity Type Names for Advanced Query

<table>
<thead>
<tr>
<th>Service</th>
<th>Activity Type Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Events</td>
<td>■ createEvent</td>
</tr>
<tr>
<td></td>
<td>■ updateEvent</td>
</tr>
<tr>
<td>Announcements</td>
<td>■ createAnnouncement</td>
</tr>
<tr>
<td></td>
<td>■ updateAnnouncement</td>
</tr>
<tr>
<td>Discussions</td>
<td>■ createForum</td>
</tr>
<tr>
<td></td>
<td>■ createTopic</td>
</tr>
<tr>
<td></td>
<td>■ replyTopic</td>
</tr>
<tr>
<td>Tags</td>
<td>■ updateBookmark</td>
</tr>
<tr>
<td>Lists</td>
<td>■ createList</td>
</tr>
<tr>
<td></td>
<td>■ editList</td>
</tr>
</tbody>
</table>
About the Activity Stream Advanced Query Option

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 39–4 for examples).

The advanced query WHERE clause is always ANDed to the internal query that is generated by Activity Stream based on the current user, portal 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 39–4 lists examples of advanced queries.

---

### Table 39–3 (Cont.) Activity Type Names for Advanced Query

<table>
<thead>
<tr>
<th>Service</th>
<th>Activity Type Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Page</td>
<td>createPage</td>
</tr>
<tr>
<td></td>
<td>editPage</td>
</tr>
<tr>
<td>Documents</td>
<td>create-blog</td>
</tr>
<tr>
<td></td>
<td>update-blog</td>
</tr>
<tr>
<td></td>
<td>create-document</td>
</tr>
<tr>
<td></td>
<td>create-wiki</td>
</tr>
<tr>
<td></td>
<td>update-document</td>
</tr>
<tr>
<td></td>
<td>update-wiki</td>
</tr>
<tr>
<td>People Connections (Profile)</td>
<td>updateStatus</td>
</tr>
<tr>
<td></td>
<td>updateProfile</td>
</tr>
<tr>
<td></td>
<td>updatePhoto</td>
</tr>
<tr>
<td>People Connections (Message Board)</td>
<td>postScope</td>
</tr>
<tr>
<td></td>
<td>postselself</td>
</tr>
<tr>
<td></td>
<td>post</td>
</tr>
<tr>
<td></td>
<td>sharescope</td>
</tr>
<tr>
<td></td>
<td>shareself</td>
</tr>
<tr>
<td></td>
<td>share</td>
</tr>
<tr>
<td></td>
<td>shareobjectscope</td>
</tr>
<tr>
<td></td>
<td>shareobjectself</td>
</tr>
<tr>
<td></td>
<td>shareobject</td>
</tr>
<tr>
<td></td>
<td>updatescope</td>
</tr>
<tr>
<td></td>
<td>updateself</td>
</tr>
<tr>
<td></td>
<td>update</td>
</tr>
<tr>
<td>People Connections (Connections)</td>
<td>connect</td>
</tr>
<tr>
<td></td>
<td>inviteForConnection</td>
</tr>
<tr>
<td>People Connections (Feedback)</td>
<td>post</td>
</tr>
</tbody>
</table>
### Table 39–4  Examples of Advanced Queries for Use with Activity Stream

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Query Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stream only document creation activities.</td>
<td>AE.SERVICE_ID = 'oracle.webcenter.doclib'</td>
</tr>
<tr>
<td>Stream activities only from an object or current portal.</td>
<td>OD.OBJECT_ID = 'objectA' OR AE.SCOPE_ID = '#{serviceCtx.scope.GUID}'</td>
</tr>
<tr>
<td>Stream activities only about wikis created by the current user.</td>
<td>OD.OBJECT_TYPE = 'Wiki' AND AD.ACTOR_NAME = '#{securityContext.userName}'</td>
</tr>
<tr>
<td>Stream activities for documents and discussions, but only create activities or all activities for the current user.</td>
<td>(AE.SERVICE_ID IN ('oracle.webcenter.doclib', 'oracle.webcenter.collab.forum') AND AE.ACTIVITY_TYPE IN ('createDocument', 'createTopic')) OR AD.ACTOR_NAME = '#{securityContext.userName}'</td>
</tr>
</tbody>
</table>
This chapter describes how to use People Connections service data controls and Java APIs. It provides details about each data control, including how to add a data control to a project and where to find additional information.

This chapter includes the following topics:

- Section 40.1, "Using People Connections Data Controls"
- Section 40.2, "People Connections Service Java APIs"

40.1 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 40.1.1, "Adding a Data Control to Your Project"
- Section 40.1.2, "Working with People Connections Management Data Control"
- Section 40.1.3, "Working with Profile Data Control"

40.1.1 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 40–1**.

![Figure 40–1 The Data Control Object Dragged and Dropped on to a .jspx Page With ADF Table Selected](image)

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 40–2**, and bind it to the data in your project to achieve the desired functionality.
40.1.2 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)

See Also: For more information about using data controls in a WebCenter Portal Framework application, see Section 4.1.3, "Using WebCenter Portal Data Controls."
The subsections in this section list and describe the methods associated with the People Connections Management Data Control.

40.1.2.1 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 40–1 lists and describes the attributes associated with this method.

Table 40–1  Attributes of the Method connectionLists

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The connections lists IDs</td>
</tr>
<tr>
<td>listName</td>
<td>The names of connections lists</td>
</tr>
<tr>
<td>localizedName</td>
<td>The localized names of connections lists</td>
</tr>
<tr>
<td>memberCount</td>
<td>The number of members on each named connections list</td>
</tr>
<tr>
<td>modifiable</td>
<td>Information about whether a named connections list can be modified</td>
</tr>
<tr>
<td>memberUserIds</td>
<td>The user IDs of members on each named connections list</td>
</tr>
</tbody>
</table>

40.1.2.2 groupedPeopleCounts
The number of connections contained in each connections group.

40.1.2.3 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 40–2 lists and describes the attributes associated with this method.

Table 40–2  Attributes of the Method receivedInvitations

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dateFormattedAsReceived</td>
<td>The date the invitation was received according to the recipient’s time stamp</td>
</tr>
<tr>
<td>dateFormattedAsSent</td>
<td>The date the invitation was sent according to the sender’s time stamp.</td>
</tr>
<tr>
<td>id</td>
<td>The GUID of the received invitation</td>
</tr>
</tbody>
</table>
40.1.2.4  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. Table 40–3 lists and describes
the attributes associated with this method.

Table 40–3  Attributes of the Method sentInvitations

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sentDate</td>
<td>The date the invitation was sent according to the system time stamp</td>
</tr>
<tr>
<td>invitorUserid</td>
<td>The user ID of the person who sent the invitation</td>
</tr>
<tr>
<td>inviteeUserid</td>
<td>The user ID of the person who received the invitation</td>
</tr>
<tr>
<td>invitationMessage</td>
<td>The invitation message</td>
</tr>
<tr>
<td>dateFormattedAsReceived</td>
<td>The date the invitation was received according to the recipient’s time stamp</td>
</tr>
<tr>
<td>dateFormattedAsSent</td>
<td>The date the invitation was sent according to the sender’s time stamp</td>
</tr>
</tbody>
</table>

40.1.2.5  acceptInvitation(String invitationId)
A method for accepting invitations to connect. Table 40–4 lists and describes the
parameters associated with this method.

Table 40–4  Parameters of the Method acceptInvitations

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>invitationId</td>
<td>The GUID of the invitation to connect to be accepted</td>
</tr>
</tbody>
</table>

40.1.2.6  addConnectionsToList(List userids, String listName)
A method for adding identified users to a named connections list. Table 40–5 lists and
describes the parameters associated with this method.

Table 40–5  Parameters of the Method addConnectionsToList

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userids</td>
<td>The user names of the users to add to the named connections list</td>
</tr>
<tr>
<td>listName</td>
<td>The name of the connections list to which to add identified users</td>
</tr>
</tbody>
</table>
40.1.2.7 createConnectionList(String listName)
A method for creating a connections list. Table 40–6 lists and describes the parameters associated with this method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>listName</td>
<td>The name to give the new connections list</td>
</tr>
</tbody>
</table>

40.1.2.8 declineInvitation(String invitationId)
A method for declining an invitation to connect. Table 40–7 lists and describes the parameters associated with this method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>invitationId</td>
<td>The unique GUID of the declined invitation</td>
</tr>
</tbody>
</table>

40.1.2.9 dropConnectionList(String listName)
Method for deleting a connections list. Table 40–8 lists and describes the parameters associated with this method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>listName</td>
<td>The name of the connections list to delete</td>
</tr>
</tbody>
</table>

40.1.2.10 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 40–9 lists and describes the parameters associated with this method. Table 40–10 lists and describes the attributes associated with this method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>connectionListName</td>
<td>The name of the connections list for which to return the following attributes (see Table 40–10):</td>
</tr>
<tr>
<td></td>
<td>■ id</td>
</tr>
<tr>
<td></td>
<td>■ listName</td>
</tr>
<tr>
<td></td>
<td>■ localizedName</td>
</tr>
<tr>
<td></td>
<td>■ memberCount</td>
</tr>
<tr>
<td></td>
<td>■ modifiable</td>
</tr>
<tr>
<td></td>
<td>■ memberUserIds</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The unique GUID of the connections list to return</td>
</tr>
<tr>
<td>listName</td>
<td>The name of the connections list to return</td>
</tr>
<tr>
<td>localizedName</td>
<td>The localized name of the connections list to return</td>
</tr>
</tbody>
</table>
Using People Connections Data Controls

Table 40–10 (Cont.) Attributes of the Method getConnectionList

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>memberCount</td>
<td>The number of members on the connections list to return</td>
</tr>
<tr>
<td>modifiable</td>
<td>An indicator of whether the returned connections list can be modified</td>
</tr>
<tr>
<td>memberUserIds</td>
<td>The user IDs of the members of a returned connections list</td>
</tr>
</tbody>
</table>

40.1.2.11 getConnection(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 40–11 lists and describes the parameters associated with this method. Table 40–12 lists and describes the attributes associated with this method.

Table 40–11 Parameters of the Method getConnections

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userid</td>
<td>The user ID for each returned connection</td>
</tr>
<tr>
<td>connectionListName</td>
<td>The name of the connection list from which to obtain connections</td>
</tr>
<tr>
<td>filterPattern</td>
<td>A filter to use against returned connections, such as &quot;co,&quot; which returns condoleezza as well as nicole</td>
</tr>
<tr>
<td>sortBy</td>
<td>The order by which to sort connections</td>
</tr>
<tr>
<td></td>
<td>Enter LAST_ACTIVITY_TIME to sort connections in descending date/time order. Leave blank to sort alphabetically by name.</td>
</tr>
<tr>
<td>startIndex</td>
<td>The point from which to start fetching results</td>
</tr>
<tr>
<td></td>
<td>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.</td>
</tr>
<tr>
<td>fetchSize</td>
<td>The number of results to return (see startIndex)</td>
</tr>
</tbody>
</table>

Table 40–12 Attributes of the Method getConnections

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>connecteeUserId</td>
<td>The user ID of a connection</td>
</tr>
<tr>
<td>connectionListsMembershipList</td>
<td>The names of connections lists of which the connection is a member</td>
</tr>
<tr>
<td>connecteeUserProfile</td>
<td>The connectee's Profile details</td>
</tr>
</tbody>
</table>

40.1.2.12 getNumberOfConnections(String userid, String connectionListName, String filterPattern)

Returns the number of users (int) connected to the identified user. Table 40–13 lists and describes the parameters associated with this method. Table 40–14 lists and describes the attributes associated with this method.
Using People Connections Data Controls

40.1.2.13 ignoreInvitation(String invitationId)
Specifies that the invitation with the specified ID should be ignored. Table 40–15 lists and describes the parameters associated with this method.

Table 40–15 Parameters of the Method ignoreInvitation

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>invitationId</td>
<td>The GUID of the connection invitation to ignore</td>
</tr>
</tbody>
</table>

40.1.2.14 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 40–16 lists and describes the parameters associated with this method.

Table 40–16 Parameters of the Method inviteUserForConnection

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userid</td>
<td>User ID of the invitee</td>
</tr>
<tr>
<td>invitationMessage</td>
<td>Connection message, such as, &quot;Let's connect!&quot;</td>
</tr>
<tr>
<td>connectionListNames</td>
<td>The names of the connections lists to add the invitee to once the invitation is accepted</td>
</tr>
</tbody>
</table>

40.1.2.15 isConnectedListModifiable(String listName)
Returns true or false (boolean) as to whether the named connections list can be modified. Table 40–17 lists and describes the parameters associated with this method. Table 40–18 lists and describes the attributes associated with this method.

Table 40–17 Parameters of the Method isConnectedListModifiable

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>listName</td>
<td>The name of the connections list for which to determine whether it can be modified</td>
</tr>
</tbody>
</table>
Table 40–18 Attributes of the Method isConnectionListModifiable

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean</td>
<td>True or false, depending on whether the named connections list can be modified</td>
</tr>
</tbody>
</table>

40.1.2.16 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 40–19 lists and describes the parameters associated with this method. Table 40–20 lists and describes the attributes associated with this method.

Table 40–19 Parameters of the Method isConnectionPartOfUnmodifiableList

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>connecteeGuid</td>
<td>The GUID of the connection for whom to determine whether this connection is a member of a connections list that cannot be modified</td>
</tr>
</tbody>
</table>

Table 40–20 Attributes of the Method isConnectionPartOfUnmodifiableList

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean</td>
<td>True or false, depending on whether the user is a member of a connections list that cannot be modified</td>
</tr>
</tbody>
</table>

40.1.2.17 removeConnection(String userid, boolean fromListOnly, String listName)

A method for removing the identified connection either as a connection or from a connections list. Table 40–21 lists and describes the parameters associated with this method.

Table 40–21 Parameters of the Method removeConnection

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userid</td>
<td>The user name of the connection to remove from the specified list</td>
</tr>
<tr>
<td>fromListOnly</td>
<td>A flag (true or false) indicating whether to remove the user as a connection or remove the user only from the specified connections list</td>
</tr>
<tr>
<td>listName</td>
<td>The name of the connections list from which to remove the specified user</td>
</tr>
</tbody>
</table>

40.1.2.18 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 40–22 lists and describes the parameters associated with this method. Table 40–23 lists and describes the attributes associated with this method.

Table 40–22 Parameters of the Method searchUsers

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filterPattern</td>
<td>A filter to use against returned search terms, such as &quot;co,&quot; which returns condoleezza as well as nicole</td>
</tr>
</tbody>
</table>

Table 40–23 Attributes of the Method searchUsers

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
</table>
### Table 40–22 (Cont.) Parameters of the Method searchUsers

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>startIndex</td>
<td>The point from which to start fetching results</td>
</tr>
<tr>
<td></td>
<td>This is used for pagination. For example, the search can return 50 matching</td>
</tr>
<tr>
<td></td>
<td>records, and only 10 are needed, starting from 1. Set startIndex to 1 and</td>
</tr>
<tr>
<td></td>
<td>fetchSize to 10.</td>
</tr>
<tr>
<td>fetchSize</td>
<td>The number of results to return (see startIndex)</td>
</tr>
</tbody>
</table>

### Table 40–23 Attributes of the Method searchUsers

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>relationalAttributes</td>
<td>—</td>
</tr>
<tr>
<td>userGuid</td>
<td>The user’s unique GUID</td>
</tr>
<tr>
<td>userProfile</td>
<td>The user’s Profile details</td>
</tr>
</tbody>
</table>

#### 40.1.2.19 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 40–24 lists and describes the parameters associated with this method.

### Table 40–24 Parameters of the updateConnectionListsMembership Method

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>member</td>
<td>The user name of the member whose connections list membership to manage</td>
</tr>
<tr>
<td>addListIds</td>
<td>The names of the connections lists to which to add the identified user</td>
</tr>
<tr>
<td>removeListIds</td>
<td>The names of the connections lists from which to remove the identified user</td>
</tr>
</tbody>
</table>

### 40.1.3 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.

#### 40.1.3.1 Method: getProfile(String userId)

A method for returning the WCUserProfile object, which is a collection of user Profile details. Table 40–25 lists and describes the parameters associated with this method.

### Table 40–25 Parameters for the getProfile Method

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userID</td>
<td>The ID of the user for whom to return Profile details</td>
</tr>
</tbody>
</table>

**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.
40.1.3.2 Method: getProfileForUpdate(String userId)
A method for updating the WCUserProfile object, which is a collection of user Profile details. Table 40–26 lists and describes the parameters associated with this method.

Table 40–26 Parameters for the getProfileForUpdate Method

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userID</td>
<td>The ID of the user for whom to update Profile details</td>
</tr>
</tbody>
</table>

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.

40.2 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

For more information, see the Java API Reference for Oracle WebCenter Portal.
This chapter describes how to integrate worklists 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 topics:

- **Section 41.1, "Introduction to Worklists"**
- **Section 41.2, "Roadmap - Configuring Worklists for Portal Framework Applications"**
- **Section 41.3, "Basic Configuration for Worklists"**
- **Section 41.4, "Advanced Information for Worklists"**

For information about managing and using worklists, see:

- the "Managing Worklists" chapter in *Administering Oracle WebCenter Portal*
- the "Adding Worklists to a Portal" chapter in *Building Portals with Oracle WebCenter Portal*
- the "Exploring Your Worklists" chapter in *Using Oracle WebCenter Portal*

### 41.1 Introduction to Worklists

Worklists enable 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 41.1.1, "Understanding Worklists"**
- **Section 41.1.2, "Requirements for Worklists"**
- **Section 41.1.3, "What Happens at Runtime"**

#### 41.1.1 Understanding Worklists

Worklists provide 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.

### 41.1.2 Requirements for Worklists

To use worklists, you must have a BPEL server installed. For information on installing BPEL for worklists, see the "Back-End Requirements for Worklists" in the Installation Guide for Oracle WebCenter Portal.

For a worklist to properly function, it must reside on a secured page, as it uses the currently authenticated user to access the BPEL server. If the page or application that contains the worklist is not secured, you will encounter exception errors.

### 41.1.3 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 "Exploring Your Worklists" chapter in Using Oracle WebCenter Portal.

To get a list of the items assigned to a user, worklists uses the WebService access function of the public BPM client API. Worklists uses SAML authentication to authorize the logged-in user to fetch the list of latest items from the BPEL repository. Worklists fetches the 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 Developer’s Guide for Oracle SOA Suite.

Figure 41–1 shows worklists at runtime.
Figure 41–1 Worklists at Runtime

Figure 41–2 and Table 41–1 in this section provide an overview of the prerequisites and tasks required to get worklists working in Portal Framework applications.
Table 41–1  Configuring the Worklist for Portal Framework Applications

<table>
<thead>
<tr>
<th>Actor</th>
<th>Task</th>
<th>Sub-Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrator</td>
<td>1. Install Oracle WebCenter Portal and the Oracle SOA Suite</td>
<td></td>
</tr>
</tbody>
</table>
| Developer        | 2. Integrate worklist in your Portal Framework application | 2.a Create a Worklist connection  
|                  |                                           | 2.b Add the Worklist task flow to a page in JDeveloper |

Figure 41–2  Configuring Worklists for Portal Framework Applications
This section describes how to set up a connection for worklists and add the task flow to your Portal Framework application.

This section contains the following subsections:

- **Section 41.3.1**, "Setting up Connections for Worklists"
- **Section 41.3.2**, "Adding Worklists Functionality at Design Time"
- **Section 41.3.3**, "Setting Security for Worklists"

### Table 41–1 (Cont.) Configuring the Worklist for Portal Framework Applications

<table>
<thead>
<tr>
<th>Actor</th>
<th>Task</th>
<th>Sub-Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developer/Administrator</td>
<td>3. Deploy the application using one of the following tools:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ JDeveloper (Developer)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ Fusion Middleware Control (Administrator)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ WLST (Administrator)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ WLS Admin Console (Administrator)</td>
<td></td>
</tr>
<tr>
<td>Administrator</td>
<td>5. (Optional): Configure BPEL server to use same identity store as the application</td>
<td></td>
</tr>
<tr>
<td>Administrator</td>
<td>6. (Optional): Secure the connection to the BPEL server</td>
<td>6.a Configure single sign-on</td>
</tr>
<tr>
<td></td>
<td>6.b Configure WS-Security</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6.c Configure SSL</td>
<td></td>
</tr>
<tr>
<td>Developer/Administrator</td>
<td>7. (Optional): Add/modify connection parameters using one of the following tools:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ JDeveloper, then redeploy the application (Developer)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ Fusion Middleware Control (Administrator)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ WLST (Administrator)</td>
<td></td>
</tr>
<tr>
<td>End User</td>
<td>8. Test that worklists functionality is working</td>
<td>8.a Log in to the Portal Framework application</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8.b Generate a worklist event</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8.c Verify event information displays in the task flow</td>
</tr>
</tbody>
</table>
41.3.1 Setting up Connections for Worklists

To use worklists, you must establish a connection to one or more BPEL servers. This section describes these connections and how to create them in a Portal Framework application.

41.3.1.1 Worklist Connections

Worklist rely on a BPEL server. After 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 Fusion Middleware Control. For more information, see the Administering Oracle WebCenter Portal.

41.3.1.2 How to Set Up Connections for Worklists

You can add the Worklist task flow to your application at design time or at runtime. Before you can run the task flow, you must create a connection from your application to a BPEL server.

See Also: Chapter 4, "Preparing Your Application for WebCenter Portal Tools and Services"

To set up the connection for worklists:

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 41–3).

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:
Basic Configuration for Worklists

- **SAML Token Client Policy** (<code>oracle/wss10_saml_token_client_policy</code>) - Select to verify your basic configuration without any additional security. This is the default setting.

- **SAML Token With Message Client Policy** (<code>oracle/wss10_saml_token_with_message_protection_client_policy</code>) - Select to increase the security using SAML-based BPEL Web Services. If selected, you must configure keys stores both in your Portal Framework application and in the BPEL application. For more information about this configuration, see the "Configuring Policies" section in the *Security and Administrator’s Guide for Web Services*.

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, <code>http://mySSO.host.com:7777</code>. This URL is used to build SSO- or HTTPS-enabled URL links to worklist items, whereas the URL property for worklists 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 is using multiple BPEL server connections. For information about working with recipient key alias and WS-security, see the chapter "Configuring WS-Security" in Administering 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.

41.3.2 Adding Worklists Functionality at Design Time

This section describes the Worklist task flow and how to add it to your application.

41.3.2.1 Worklist Task Flow

The Worklist task flow is found in the WebCenter Portal - Services Catalog in the Resource Palette.

41.3.2.2 How to Add the Worklists Functionality to your Application

You can add worklists functionality to your application before or after you have created your connection. However, you must set up your connection before you run the application.

To add the Worklist task flow to your application:

1. Follow the steps in Section 4.2.1, "How to Prepare Your Application to Consume Tools and 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 41–4.

Figure 41–4  Worklist Task Flow in the Page Source

```xml
<af region "first">
    <af region value="#{bindings:worklist.regionModel}"
        id="worklist"/>
</af:facet>
```

5. Save your project, then run your page to a browser. The worklist displays in your browser.

41.3.3 Setting Security for Worklists

Worklists display 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 4.2.1.1, "Implementing Security for Tools and Services"). The user names need to either exist in a shared user directory (LDAP), or set up similarly (same user name) on both the Portal Framework application and the BPEL Server. For example, if the user rsmith needs to use
worklists in your Portal Framework application to store and retrieve tasks from the BPEL server, then 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 Securing Applications with Oracle Platform Security Services.

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 is not secured, she will not see any worklist items stored on the BPEL server and exception errors may occur.

### 41.4 Advanced Information for Worklists

Worklists collate all the worklist items for the authenticated user from all connections in your application defined for the worklists. You can use the create connection method described in Section 41.3.1.2, "How to Set Up Connections for Worklists" to create additional connections, or right-click the connection name in the Application Resources list and choose **Delete** to remove a connection.
This chapter describes the REST APIs associated with People Connections.

This chapter includes the following topics:

- **Section 42.1, "Activity Stream REST API"
- **Section 42.2, "Connections and Profile REST API"
- **Section 42.3, "Feedback REST API"
- **Section 42.4, "Message Board REST API"
- **Section 42.5, "Creating an Invitation"

**See Also:** For an introduction to the REST APIs, see Chapter 53, "Using Oracle WebCenter Portal REST APIs."

### 42.1 Activity Stream REST API

Use the Activity Stream REST API to browse user application activities in an activity stream. This section provides information about the REST API methods you can use to perform this action.

This section 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 the REST API 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 Framework 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 People Connections, each feature has its own link element. For example, to find the entry point for the People Connections’ Activity Stream feature, find the link elements with a `resourceType` of:

```
urn:oracle:webcenter:activities:stream
```
The corresponding \texttt{href} or \texttt{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 People Connections' Activity Stream feature.

\textbf{See Also:} For more information about the Resource Index, see Section 53.5.1, "Using the Resource Index."

For more information about resource types, see Section 53.5.2.1, "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 People Connections' Activity Stream feature is:

\texttt{urn:oracle:webcenter:activities:stream}
\texttt{urn:oracle:webcenter:activities:activity}

### 42.1.3 Activity Stream Security Considerations

You must be logged into the REST service to access any of the People Connections REST APIs. After that, the underlying service handles permission checking and the like.

\textbf{See Also:} For general security considerations, see Section 53.8, "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:

- "\texttt{urn:oracle:webcenter:activities:stream}"
- "\texttt{urn:oracle:webcenter:activities:activity}"

\texttt{urn:oracle:webcenter:activities:stream}

The \texttt{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 portals, including the Home portal, 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 \texttt{stream} resource and the \texttt{rel} of the link that you use. For example, the activity stream query filter is available only from links with a \texttt{rel} attribute of

42-2 Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper
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>
<thead>
<tr>
<th>rel</th>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>urn:oracle:webcenter:activities:stream:person</td>
<td>Activities from the user’s stream (GUID/@self)¹</td>
</tr>
<tr>
<td>urn:oracle:webcenter:activities:stream:connections</td>
<td>Activities from the user's connections' streams (GUID/@connections)¹</td>
</tr>
<tr>
<td>urn:oracle:webcenter:activities:stream</td>
<td>Activities determined by the activity stream query filter</td>
</tr>
<tr>
<td>urn:oracle:webcenter:activities:stream:space</td>
<td>Activities from the portal activity stream</td>
</tr>
<tr>
<td>urn:oracle:webcenter:activities:stream:list</td>
<td>Activities from the portal activity list</td>
</tr>
</tbody>
</table>

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:

* fromDate – Specifies activities start date (yyyy-mm-dd).
* toDate – Specifies activities end date, (yyyy-mm-dd).
* 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 portal. **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 portals 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 portal GUIDs used to show activities.
* userGroupSpaceActivities – Specifies whether or not to show the user's activities in their portal. **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.
Using the People Connections REST APIs

Activity Stream REST API

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 39.4, "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 Home portal and from the streams of the user's connections:

http://host:port/rest/api/activities?personal=true&connections=true&personGuid=@me&token=utoken

The following URI returns only the user's personal profile and connections activities:

http://host:port/rest/api/activities?serviceIds=oracle.webcenter.peopleconnections.profile,oracle.webcenter.peopleconnections.connections&personal=true&personGuid=@me&token=utoken

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 portal, you can pass the GUID of the portal’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 portal. 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 39.4, "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>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:activities:stream:person</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:activities:stream</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:activities:activity</td>
</tr>
</tbody>
</table>
**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>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:activities:activity</td>
</tr>
<tr>
<td>icon</td>
<td>urn:oracle:webcenter:activities:activity:icon</td>
</tr>
</tbody>
</table>

**Read-only Elements for activity**
Table 42–4 lists the read-only elements for the activity resource.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>activityType</td>
<td>String</td>
<td>Activity type</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unique within the service. For example, Discussions can return the activityType createForum.</td>
</tr>
</tbody>
</table>

**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 (Cont.) Read-Only Elements for activity

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>commentsCount</td>
<td>String</td>
<td>If you specify commentsSummary in the data parameter, then the commentsCount parameter will be returned.</td>
</tr>
<tr>
<td>createdDate</td>
<td>Date (String)</td>
<td>Date the activity was created.</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>The description of the activity.</td>
</tr>
<tr>
<td>detail</td>
<td>String</td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Similar to detailURL. Both, either, or neither can be used. The detailURL will be available when creating wiki and blog in a portal.</td>
</tr>
<tr>
<td>detailURL</td>
<td>String</td>
<td>detailURL is available when creating wikis and blogs in a portal. In a Web-based WebCenter Portal, the user can click this URL and open a wiki or blog page in a portal.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Similar to detail. Both, either, or neither can be used.</td>
</tr>
<tr>
<td>displayDescription</td>
<td>String</td>
<td>A pre-formatted, internationalized description.</td>
</tr>
<tr>
<td>displayMessage</td>
<td>String</td>
<td>A pre-formatted, internationalized message (does not include template information).</td>
</tr>
<tr>
<td>groupSpace</td>
<td>groupSpaceReference</td>
<td>Information about the portal in which the activity was performed</td>
</tr>
<tr>
<td>id</td>
<td>String</td>
<td>Unique ID of the message</td>
</tr>
<tr>
<td>isSummary</td>
<td>true or false</td>
<td>Indicates whether or not this activity is a summary of other activities.</td>
</tr>
<tr>
<td>likesCount</td>
<td>String</td>
<td>If you specify likesSummary in the data parameter, then the likesCount parameter will be returned.</td>
</tr>
<tr>
<td>message</td>
<td>String</td>
<td>Localized string for this activity. This may contain replacement strings within curly braces ({}).</td>
</tr>
<tr>
<td>permission</td>
<td>PRIVATE, SHARED, PUBLIC</td>
<td>Permission level of this activity</td>
</tr>
</tbody>
</table>
Suppose you want to display the most recent messages for a user named Carl. You want to display information like this: "Carl created the portal Customer Feedback".

The `templateParams` element 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 portal, the `templateParams` element

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| `scope`       | String                         | Scope of the activity
|               |                                | This might return a portal, like the Home portal.                           |
|               |                                | For example, for a portal, 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 tools and services IDs, see Table G–7, "Service and Tool IDs". |
| `sharedLink_url` | String                      | Can be used to render a navigation link by the REST client as shown in the example below: |
|               |                                | `<param resourceType="urn:oracle:webcenter:activities:parameter">`        |
|               |                                | `<links>`                                                                   |
|               |                                | `<link href="http://www.google.com" />`                                    |
|               |                                | `</links>`                                                                  |
|               |                                | `<displayName>http://www.google.com</displayName>`                         |
|               |                                | `<key>_sharedLink_url</key>`                                               |
|               |                                | `<type>custom</type>`                                                       |
|               |                                | `</param> "`                                                               |
| `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.
for that activity captures information about the user and about the portal. 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 portal, the returned activity object contains a <message> item that is parameterized like this:

<message>{actor[0]} has created the portal {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 53.13.2, "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, portal, 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. Portal objects include an activity stream link for portals. 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**

Hyperlinks to comments, commentsSummary, likes and likesSummary are included by default (you can expand the commentsSummary and likesSummary by specifying data=data,commentsSummary,likesSummary). Comments and likes can be associated with the object referenced by the activity or the activity itself. For example, if you edit a document, then the comments will be associated with the document. If you add comments on an edit document activity on the activity stream page, then comments will be associated with the edit document activity.

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. For example, for body data you would use:

  ```json
  {
    "text" : "REST Comment"
  }
  ```
The following URLs show examples of GET requests for comments, comment Summary, likes and likes Summary for a document object:

```
"http://example.com:8892/rest/api/activities/services/oracle.webcenter.doclib/objectTypes/document/objects/(<document_object_id>)/comments?startIndex=0&itemsPerPage=10&utoken=FCvY3qQSBh0eAByLdxugV-1 UgF03_w***
```

```
"http://example.com:8892/rest/api/activities/services/oracle.webcenter.doclib/objectTypes/document/objects/(<document_object_id>)/commentsSummary?utoken=FCvY3qQSBh0eAByLdxugV-1 UgF03_w***
```

```
"http://example.com:8892/rest/api/activities/services/oracle.webcenter.doclib/objectTypes/document/objects/(<document_object_id>)/likes?startIndex=0&itemsPerPage=10&utoken=FCvY3qQSBh0eAByLdxugV-1 UgF03_3_w***
```

```
"http://example.com:8892/rest/api/activities/services/oracle.webcenter.doclib/objectTypes/document/objects/(<document_object_id>)/likesSummary?utoken=FCvY3qQSBh0eAByLdxugV-1 UgF03_3_w***
```

Where `<document_object_id>` is the document object ID.

The following URLs show examples of GET requests for comments, comment Summary, likes and likes Summary for a create document activity:

```
"http://example.com:8892/rest/api/activities/ffa9a9f0-d02f-4e30-8c58-8a41b7a6e8a3/comments?startIndex=0&itemsPerPage=10&utoken=FCvY3qQSBh0eAByLdxugV-1 UgF03_ w***
```

```
"http://example.com:8892/rest/api/activities/ffa9a9f0-d02f-4e30-8c58-8a41b7a6e8a3/commentsSummary?utoken=FCvY3qQSBh0eAByLdxugV-1 UgF03_w***
```

```
"http://example.com:8892/rest/api/activities/ffa9a9f0-d02f-4e30-8c58-8a41b7a6e8a3/likes?startIndex=0&itemsPerPage=10&utoken=FCvY3qQSBh0eAByLdxugV-1 UgF03_ w***
```

```
"http://example.com:8892/rest/api/activities/ffa9a9f0-d02f-4e30-8c58-8a41b7a6e8a3/likesSummary?utoken=FCvY3qQSBh0eAByLdxugV-1 UgF03_3_w***
```

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.
Connections and Profile REST API

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:
resource
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>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:activities:activity:icon</td>
</tr>
<tr>
<td>urn:oracle:webcenter:parent</td>
<td>urn:oracle:webcenter:activities:activity</td>
</tr>
</tbody>
</table>

42.2 Connections and Profile REST API
Use the Connections and Profile REST API 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 API methods to use to perform these actions.

This section 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 People Connections, each feature has its own link element. For example, to find the entry points for the People Connections’ Connections and Profile features, 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 People Connections’ Connections and Profile features.

See Also: For more information about the Resource Index, see Section 53.5.1, "Using the Resource Index.”

For more information about resource types, see Section 53.5.2.1, "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 People Connections’ Connections and Profile features 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 53.8, "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.
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>
<thead>
<tr>
<th>rel</th>
<th>ResourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:people:person:status</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:people:person:listName</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:activities:stream:list</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:activities:stream</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:feedback:all-received</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:feedback:all-given</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:people:person:status</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:people:person:list</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:people:person:listName</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:activities:stream:list</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:activities:stream</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:feedback</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:feedback</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:people:icon</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:people:person:status</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:people:person:list</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:people:person:listName</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:activities:stream:list</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:activities:stream</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:feedback:all-received</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:feedback:all-given</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:people:icon</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:people:person:status</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:people:person:list</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:people:person:listName</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:activities:stream:list</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:activities:stream</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:feedback</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:feedback</td>
</tr>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:people:icon</td>
</tr>
</tbody>
</table>
**Table 42–7 (Cont.) Related Resource Types for urn:oracle:webcenter:people:icon**

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>urn:oracle:webcenter:parent</td>
<td>urn:oracle:webcenter:people</td>
</tr>
</tbody>
</table>

**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]
    - q=[guid:equals:guid]
    - 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.
Connections and Profile REST API

**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>
<thead>
<tr>
<th><strong>Table 42–8  Read-Only Elements for person</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Element</strong></td>
</tr>
<tr>
<td>guid</td>
</tr>
<tr>
<td>id</td>
</tr>
<tr>
<td>displayName</td>
</tr>
<tr>
<td>birthday</td>
</tr>
<tr>
<td>connected</td>
</tr>
<tr>
<td>language</td>
</tr>
<tr>
<td>timeZone</td>
</tr>
<tr>
<td>name</td>
</tr>
<tr>
<td>address</td>
</tr>
<tr>
<td>emails</td>
</tr>
<tr>
<td>photos</td>
</tr>
<tr>
<td>phoneNumbers</td>
</tr>
<tr>
<td>organizations</td>
</tr>
<tr>
<td>manager</td>
</tr>
<tr>
<td>reportees</td>
</tr>
<tr>
<td>status</td>
</tr>
</tbody>
</table>

1 Data types, such as `DATE` and `BOOLEAN`, are stored in the API as `STRING`.

Using the People Connections REST APIs 42-17
**Resource Types Linked to from person**

Table 42–9 lists the resource types that the client can link to from the person resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:people:person</td>
</tr>
<tr>
<td>alternate</td>
<td>urn:oracle:webcenter:spaces:profile (HTML)</td>
</tr>
<tr>
<td>urn:oracle:webcenter:feedback:all-given</td>
<td>urn:oracle:webcenter:feedback</td>
</tr>
<tr>
<td>urn:oracle:webcenter:feedback:all-received</td>
<td>urn:oracle:webcenter:feedback</td>
</tr>
</tbody>
</table>

**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>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:people:person:list</td>
</tr>
</tbody>
</table>

1 self rel currently includes "urn:oracle:webcenter:people:person:list:1" 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>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:people:person:listNames</td>
</tr>
</tbody>
</table>
**urn:oracle:webcenter:people:person:listName**

The `listName` response contains the names of connections lists and URIs for use in accessing the connections lists. See also "urn:oracle:webcenter:people:person:listNames".

### Navigation Paths to listName

This section shows how the client can navigate through the hypermedia to access the `listName` resource:

```
resourceIndex
  person
    listNames
      listName
```

### Supported Methods for listName

The following method is supported by the `listName` resource:

- **DELETE**

  - request

### Writable Elements for listName

**Table 42–12** lists the writable elements for the `listName` resource.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Required</th>
<th>Constraints</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Yes</td>
<td>1 or more characters</td>
<td>A single list name</td>
</tr>
</tbody>
</table>

### Resource Types Linked to from listName

**Table 42–13** lists the resource types that the client can link to from the `listName` resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:people:person:list</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:activities:stream: list</td>
</tr>
<tr>
<td>urn:oracle:webcenter:people:person:lis...</td>
<td></td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:activities:stream</td>
</tr>
</tbody>
</table>

**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:
Connections and Profile REST API

**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>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Required</th>
<th>Constraints</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>guid</td>
<td>String</td>
<td>Yes</td>
<td>1 or more characters</td>
<td>GUID of the user</td>
</tr>
<tr>
<td>message</td>
<td>String</td>
<td>No</td>
<td>0 or more characters</td>
<td>Invitation message. Use this only when inviting users to be connections (that is POSTing to the @connections list, not to user-created connections lists).</td>
</tr>
</tbody>
</table>

**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>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Required</th>
<th>Constraints</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>note</td>
<td>String</td>
<td>Yes</td>
<td>1 or more characters</td>
<td>Content of status message</td>
</tr>
</tbody>
</table>
Resource Types Linked to from status
Table 42–16 lists the resource types that the client can link to from the status resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:people:person:status</td>
</tr>
</tbody>
</table>

**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>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Required</th>
<th>Constraints</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>No</td>
<td>1 or more characters</td>
<td>Message attached to the invitation</td>
</tr>
</tbody>
</table>
Resource Types Linked to from invitations

Table 42–18 lists the resource types that the client can link to from the invitations resource:

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:people:invitation</td>
</tr>
</tbody>
</table>

**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>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Required</th>
<th>Constraints</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>String</td>
<td>Yes (PUT)</td>
<td>ACCEPTED</td>
<td>The status of the invitation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>IGNORED</td>
<td>Note: When you accept or ignore an invitation, it is removed from your list of invitations.</td>
</tr>
</tbody>
</table>

Read-only Elements for invitation

Table 42–20 lists the read-only elements for the invitation resource.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Unique ID of the invitation</td>
</tr>
<tr>
<td>invitee</td>
<td>personReference</td>
<td>User to whom the invitation is sent</td>
</tr>
<tr>
<td>invitor</td>
<td>personReference</td>
<td>User from whom the invitation is sent</td>
</tr>
</tbody>
</table>
42.3 Feedback REST API

Use the Feedback REST API to read and delete feedback. This section provides information about the REST API methods to use to perform these actions.

This section 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 People Connections, each feature has its own link element. To find the entry points for the People Connections' Feedback feature, 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 People Connections' Feedback feature.

**See Also:** For more information about the Resource Index, see Section 53.5.1, "Using the Resource Index."

For more information about resource types, see Section 53.5.2.1, "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 People Connections' Feedback feature is:

```
urn:oracle:webcenter:feedback
urn:oracle:webcenter:feedback:message
```
42.3.3 Feedback Security Considerations

You must be logged into 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 53.8, "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>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self urn:oracle:webcenter:feedback:all-received</td>
<td>urn:oracle:webcenter:feedback</td>
</tr>
<tr>
<td>self urn:oracle:webcenter:feedback:all-given</td>
<td>urn:oracle:webcenter:feedback</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:feedback:message</td>
</tr>
</tbody>
</table>

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>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>body</td>
<td>String</td>
<td>Message content</td>
</tr>
<tr>
<td>id</td>
<td>String</td>
<td>Unique ID of the message</td>
</tr>
<tr>
<td>author</td>
<td>personReference</td>
<td>User who created the message</td>
</tr>
<tr>
<td>created</td>
<td>Date (String)</td>
<td>Date the message was created</td>
</tr>
<tr>
<td>receivedUser</td>
<td>personReference</td>
<td>A person reference to the user who received the feedback</td>
</tr>
</tbody>
</table>

1 Data types, such as DATE and BOOLEAN, are stored in the API as STRING.
42.4 Message Board REST API

Use the Message Board REST API to post, read, and delete messages to a user’s or a portal message board. This section provides information about the REST API methods to use to perform these actions.

This section 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"
- Section 42.4.5, "Filtering Messages Based on Visibility"

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 People Connections, each feature has its own link element. To find the entry points for the People Connections’ Message Board feature, 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 portal response for the portal 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 People Connections' Message Board feature.

See Also: For more information about the Resource Index, see Section 53.5.1, "Using the Resource Index."

For more information about resource types, see Section 53.5.2.1, "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 for People Connections is:

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:feedback:message</td>
</tr>
</tbody>
</table>
42.4.3 Message Board Security Considerations

You must be logged into 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 53.8, "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) portal 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
**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>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>messageType</td>
<td>String</td>
<td>Returns link if the message has a link. Otherwise, returns null.</td>
</tr>
<tr>
<td>link</td>
<td>String</td>
<td>Contains link data for messages with links: name, url, icon, description, mimeType, objectId, objectType, serviceId.</td>
</tr>
</tbody>
</table>

### Resource Types Linked to from messageBoard

Table 42–26 lists the resource types that the client can link to from the messageBoard resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:messageBoard</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:messageBoard:message</td>
</tr>
</tbody>
</table>

**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 portal 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
```
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>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Required</th>
<th>Constraints</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>body</td>
<td>String</td>
<td>Yes</td>
<td>1 or more characters</td>
<td>message content</td>
</tr>
</tbody>
</table>

Read-only Elements for message
Table 42–28 lists the read-only elements for the message resource.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Unique ID of the message</td>
</tr>
<tr>
<td>author</td>
<td>personReference</td>
<td>User who created the message</td>
</tr>
<tr>
<td>created</td>
<td>Date (String)1</td>
<td>Date the message was created</td>
</tr>
</tbody>
</table>

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>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:messageBoard:message</td>
</tr>
</tbody>
</table>
42.4.5 Filtering Messages Based on Visibility

Although the REST GET command for retrieving messages retrieves all shown messages by default, you can also retrieve messages that are private or hidden through the application user interface using a set of visibility filters.

Messages posted on message boards can have the following visibility criteria:

- Public
- Private
- Hidden
- Public and hidden
- Private and hidden

By default, messages are public.

Private Messages:
A person owning a message board can mark any message as private. When a message is marked as private, that message will not be visible to anyone other than the owner of the message unless the owner chooses to send a private message to someone else. Otherwise, other people viewing a message board with private messages will see only public messages.

Hidden Messages:
A person owning a message board can mark any message as hidden. When a message is marked as hidden, that message will not be visible to the owner of the message board, but will remain visible to other people viewing the message board.

REST URLs for Message Boards:
The context for message board URL filters is:

```
rest/api/messageBoards/<BOARD-TYPE>/<GUID>/<FILTER-TYPE>
```

Where:

- `<BOARD-TYPE>` is either person or space
- `<GUID>` is either `@me`, the person `<GUID>` (if `<BOARD-TYPE>` is person) or the space `<GUID>` (if `<BOARD-TYPE>` is portal)
- `<FILTER-TYPE>` is applicable only for the person `<BOARD-TYPE>` and can be one of:
  - `null` - (default) shows all messages
  - `private` - shows private messages
  - `public` - shows public messages
  - `hidden` - shows hidden and public messages
  - `private_hidden` - shows private and hidden messages

Available filters in the context of the applicable HTML methods are shown below.

**GET**

- `@me`
- `all`
  ```
  rest/api/messageBoards/person/@me
  ```
- private
  rest/api/messageBoards/person/@me/private
- public
  rest/api/messageBoards/person/@me/public
- hidden and public
  rest/api/messageBoards/person/@me/hidden
- private and hidden
  rest/api/messageBoards/person/@me/private_hidden

- person
  rest/api/messageBoards/person/<GUID>

  If the GUID matches that of the logged in user then the same filtering as for @me
  applies. If the GUID is different, then no filtering is available.

- space
  rest/api/messageBoards/space/<GUID>

  No filtering based on visibility is available.

**POST**

- @me
  - all
    rest/api/messageBoards/person/@me
  - private
    {"body" : "<BODY_CONTENT>",
     "visibilityType" : "private"}
  - public
    {"body" : "<BODY_CONTENT>",
     "visibilityType" : "public"}
  - hidden and public
    {"body" : "<BODY_CONTENT>",
     "visibilityType" : "hidden"}
  - private and hidden
    {"body" : "<BODY_CONTENT>",
     "visibilityType" : "private_hidden"}

- person
  rest/api/messageBoards/person/<GUID>

  If the GUID is the same as the logged in user, then the filters for GET apply. If the
  users are different, then the following filters apply:
  
  - public
    {"body" : "<BODY_CONTENT>",
     "visibilityType" : "public"}
  - private
    {"body" : "<BODY_CONTENT>",
     "visibilityType" : "private"}
Creating an Invitation

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"**
42.5.1 Creating an Invitation

1. The first step, as always with REST API methods, 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:
   a. In the resource index listing, scan for the link with the following resource type:
      ```
      urn:oracle:webcenter:people:person
      ```
   b. Execute a search for the user who you wish to invite. In this example, the user is named Monty.
      
      ```
      ```
   c. 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>1AE5AF102E261BF062B573E287934</guid>
   <id>monty</id>
   <name>
   <formatted>monty</formatted>
   </name>
   ```

d. 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

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:people:person</td>
</tr>
<tr>
<td>alternate</td>
<td>urn:oracle:webcenter:spaces:profile (HTML)</td>
</tr>
<tr>
<td>urn:oracle:webcenter:messageBoard</td>
<td>urn:oracle:webcenter:messageBoard</td>
</tr>
<tr>
<td>urn:oracle:webcenter:feedback:all:given</td>
<td>urn:oracle:webcenter:feedback</td>
</tr>
<tr>
<td>urn:oracle:webcenter:feedback:all:received</td>
<td>urn:oracle:webcenter:feedback</td>
</tr>
</tbody>
</table>

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:


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.
42.5.2 Accepting an Invitation

After an invitation has been sent, the next step is for the invitee (Monty) to accept the invitation.

1. Using a second REST client, retrieve the resource index:
   
   \[
   \text{GET http://<host:port>/rest/api/resourceIndex}
   \]

2. Login as Monty.

3. List all the invitations sent to Monty by making the following request:
   
   \[
   \text{GET http://<host:port>/rest/api/people/invitations?q=invitee:equals:@me&utoken=ASDF}\]

   Each invitation element listed in the response contains a link that supports the UPDATE operation and that looks something like this:

   \[
   <\text{links}>
   </\text{links}>
   <\text{id}><invitationid></id>
   \]

4. To accept the invitation, make the following request:
   
   \[
   \text{PUT http://<host:port>/rest/api/people/invitations/<invitationid>?utoken=ASDF}\]

   Headers
   
   \[
   \text{Accept -> application/xml}
   \text{Content-Type -> application/xml}\]

   Body
   
   \[
   <\text{invitation}>
   <\text{id}><invitationid></id>
   <\text{status}>accepted</\text{status}>
   </\text{invitation}>
   \]

42.5.3 Deleting an Invitation

After you initiate an invitation, you can view the invitation from your account by specifying invitor:equals:@me. For example:

\[
\text{GET http://<host:port>/rest/api/people/invitations?q=invitor:equals:@me&utoken=ASDF} \]

Each invitation element listed in the response contains a link that supports the DELETE operation and that looks something like this:

\[
<\text{links}>
\]
Creating an Invitation

Using the People Connections REST APIs

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 contains the following chapters:

- Chapter 43, "Integrating Links"
- Chapter 44, "Integrating Tags"
- Chapter 45, "Integrating Search"
- Chapter 46, "Integrating the Activity Graph"
- Chapter 47, "Integrating Analytics"
This chapter explains how to integrate links functionality in a Portal Framework application at design time.

This chapter includes the following topics:

- Section 43.1, "Introduction to Links"
- Section 43.2, "Basic Configuration for Links"
- Section 43.3, "Advanced Information for Links"

For more information about managing and including links, see:

- the "Setting up Database Connections" chapter in *Administering Oracle WebCenter Portal*
- the "Working with the Links Component" chapter in *Building Portals with Oracle WebCenter Portal*
- the "Linking Information in WebCenter Portal" chapter in *Using Oracle WebCenter Portal*

### 43.1 Introduction to Links

This section provides overview information about links functionality and requirements. It includes the following subsections:

- Section 43.1.1, "Understanding Links"
- Section 43.1.2, "Requirements for Links"
- Section 43.1.3, "What Happens at Runtime"

#### 43.1.1 Understanding Links

Links 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.

Links provides a means for the application developer to set up source objects (for example, the discussions tool) and target objects (for example, a document), thus enabling your users to create links between the two objects.

There are three actions associated with links: create, delete, and manage. The manage action includes the create and delete actions.

The following custom JSF components are included with links:
- **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.
  - The gold **Links** icon ([Figure 43–2](#)) indicates that links are present in the Links dialog.

*Figure 43–1  The Links Icon (No Links Present)*

![The Links Icon (No Links Present)](image1)

*Figure 43–2  The Links Icon (Links Present)*

![The Links Icon (Links Present)](image2)

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
Links are *not* available for the other WebCenter Portal components and tools, such as Mail and People Connections.

Links 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 Links

Links automatically recognize any WebCenter Portal tool or component in your application. After you have configured a tool or 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."

### 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.

Links 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**

![Links Dialog](image)

With links, 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 links at runtime, see the "Linking Information in WebCenter Portal" chapter in *Using Oracle WebCenter Portal*.

### 43.2 Basic Configuration for Links

This section describes the steps required for adding links functionality to your application. It includes the following subsections:
- Section 43.2.1, "Setting up Connections for Links"
- Section 43.2.2, "Adding Links Functionality at Design Time"
- Section 43.2.3, "Setting Security for the Links"
- Section 43.2.4, "Troubleshooting Links"

#### 43.2.1 Setting up Connections for Links

Links require 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 4.2.2, Setting Up a Database Connection.*

#### 43.2.2 Adding Links Functionality at Design Time

This section explains a basic incorporation of links functionality. It includes the following subsections:
- Section 43.2.2.1, "Links Task Flows."
- Section 43.2.2.2, "How to Add Links Functionality to your Application."

#### 43.2.2.1 Links Task Flows

WebCenter Portal includes the Links Dialog task flow.

#### 43.2.2.2 How to Add Links Functionality to your Application

To add the Links Dialog task flow to your Framework application:

1. Follow the steps described in *Chapter 2, Setting Up 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 links functionality.
4. In the Component Palette, click *WebCenter Portal - Links*.
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.
Basic Configuration for Links

**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 serviceId and objectId are combined 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**

```xml
<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>
</af:form>
```

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

Links functionality 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 74.3, "Configuring ADF Security."

43.2.4 Troubleshooting Links

This section describes common problems and solutions for links.

Problem
The Links icon does not appear.

Solution
Links functionality 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 administrators when Links components are consumed. Grant other users this permission through the ADF Security Policy Editor. For more information, see Section 74.3.1, "Configuring ADF Security Settings."

43.3 Advanced Information for Links

This section describes optional features available with links. It includes the following subsection:

- Section 43.3.1, "Using the Links REST API"

43.3.1 Using the Links REST API

WebCenter Portal provides a REST API to support links. Use the links REST API 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 API methods associated with links. 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 REST APIs, see Chapter 53, "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 links, 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 links.

For more information about the resource index, see Section 53.5.1, "Using the Resource Index."

For more information about resource types, see Section 53.5.2.1, "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 53.5.2.1, "Resource Type."

The taxonomy for links 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 53.8, "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`
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;;3b0c6edeb`
  
  - **resourceName={resourceName}**
    
    The resource name, such as New Event.

For more information, see Section 53.5.2.5, "Templates."

Resource Types Linked to From results

Table 43–1 lists the resource types that the client can link to from this resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td><code>urn:oracle:webcenter:links:results</code></td>
</tr>
</tbody>
</table>
This chapter describes how to integrate tags in a WebCenter Portal Framework application at design time.

This chapter includes the following topics:

- Section 44.1, "Introduction to Tags"
- Section 44.2, "Basic Configuration for Tags"
- Section 44.3, "Advanced Information for Tags"

For more information about managing and including tags, see the "Using Tags and Bookmarks" chapter in Using Oracle WebCenter Portal.

**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 4.3, "Customizing How Services Render Resources."

### 44.1 Introduction to Tags

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.

Tags can be put on pages, documents, and custom objects. Creating a tag automatically publishes the event to the activity stream.
44.1.1 Understanding Tags

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 Tags

Tags 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 4.2.2, "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 Search at Design Time."

---

Tags 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 28, "Integrating Documents."
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 tools and services exposing resources to be tagged (and therefore searched and viewed) WebCenter Portal provides a Resource Action Handling framework. Tags use this framework to allow acting on a search result. For more information, see Section 4.3, "Customizing How Services Render Resources."

For more information about tags at runtime, see the "Using Tags and Bookmarks" chapter in Using Oracle WebCenter Portal.

44.2 Basic Configuration for Tags

This section describes required steps for adding tags to your application. It includes the following topics:

- Section 44.2.1, "Setting up Connections for Tags"
- Section 44.2.2, "Adding Tags Functionality at Design Time"
- Section 44.2.3, "Setting Security for Tags"

44.2.1 Setting up Connections for Tags

Tags require a connection to the database with the WebCenter Portal schema installed.

For details about setting up a database connection, see Section 4.2.2, "Setting Up a Database Connection."

44.2.2 Adding Tags Functionality 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 topics:

- Section 44.2.2.1, "Tags Components"
- Section 44.2.2.2, "Tags Task Flows"
- Section 44.2.2.3, "How to Add Tags Functionality to your Page"
- Section 44.2.2.4, "How to Modify Tags Task Flow Parameters"

44.2.2.1 Tags Components

The following JSF components are available:

- 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.
In the Component Palette, drag and drop components to the page. In the resulting
dialog box, 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 Task Flows

Table 44–1 lists the tags task flows.

<table>
<thead>
<tr>
<th>Task Flow</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tagging - Dialog</td>
<td>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.</td>
</tr>
<tr>
<td>Tagging - Personal View</td>
<td>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.</td>
</tr>
<tr>
<td>Tagging - Related Links</td>
<td>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.</td>
</tr>
<tr>
<td>Tagging - Tag Cloud</td>
<td>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 links for all tags in the page.</td>
</tr>
<tr>
<td>Tagging - Tag Selection</td>
<td>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.)</td>
</tr>
<tr>
<td>Tagging - Tagged Items</td>
<td>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.</td>
</tr>
</tbody>
</table>

See Also: Section 44.2.2.4, "How to Modify Tags Task Flow Parameters"
44.2.2.3 How to Add Tags Functionality to your Page

To add the tags functionality to your WebCenter Portal Framework application:

1. Follow the steps described in Chapter 2, "Setting Up 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>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResourceId</td>
<td>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:</td>
</tr>
<tr>
<td>ResourceName</td>
<td>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.</td>
</tr>
<tr>
<td>ServiceId</td>
<td>An application-wide ID. (With custom components, you add this to service-definition.xml when adding the resource viewer.)</td>
</tr>
</tbody>
</table>

   ■ To tag pages, enter #{facesContext.viewRoot.viewId}
   ■ To tag custom components, enter customComponent123

   ■ To tag pages, enter #{facesContext.viewRoot.viewId}
   ■ To tag custom components, enter Custom Component 1

   ■ To tag pages, enter oracle.webcenter.page
   ■ To tag custom components, enter mycompany.myproduct.myapp.mycomponent

6. Click OK.


9. When prompted, select Region as the way to create the task flow.

10. For custom resources, follow the steps described in Section 4.3, "Customizing How Services Render Resources" to register a resource viewer. For tools and services exposing resources to be tagged (and therefore searched and viewed) WebCenter Portal provides a Resource Action Handling framework. Tags use this framework to allow acting on a search result.

In the Source view where you added tags, 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).
44.2.2.4 How to Modify Tags Task Flow Parameters

Some tags task flows have optional task flow binding parameters.

**Note:** You should not change tags 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**

This shows the Oracle ADF data bindings defined for your page. Select a binding to see its relationship

Page Definition File:  *portal/pageDefs/untitled1PageDef.xml*

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.
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 tags task flows.

**Table 44–3 Tags Task Flow Parameters**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Task Flow</th>
</tr>
</thead>
<tbody>
<tr>
<td>resourceId</td>
<td>Unique ID of the item or resource within a given tool/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.</td>
<td>Tagging - Related Links</td>
</tr>
</tbody>
</table>
| 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 tool/service.  
  ■ In Personal View, it limits the tags to this tool/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 Tags

Tags require 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 74.3, "Configuring ADF Security."
44.3 Advanced Information for Tags

This section describes optional features available with tags. It includes the following topics:

- Section 44.3.1, "Using the Tags Java APIs"
- Section 44.3.2, "Using the Tags 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 Tags"

**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 4.3, "Customizing How Services Render Resources."

44.3.1 Using the Tags Java APIs

Custom components can implement the tags APIs to create a tagging backend compatible with the WebCenter Portal REST APIs or tags task flows and components.

For more information on the Java APIs, see Java API Reference for Oracle WebCenter Portal.

44.3.2 Using the Tags REST APIs

WebCenter Portal provides REST APIs to support tags. Use the tags 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 users by tag

This section describes the REST APIs associated with tags. It includes the following topics:
Section 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 tags, 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 tags.

For more information about the Resource Index, see Section 53.5.1, "Using the Resource Index."

For more information about resource types, see Section 53.5.2.1, "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 53.5.2.1, "Resource Type."

The taxonomy for tags 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 REST API methods.

For general security considerations, see Section 53.8, "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 topics:

- 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:** startIndex, itemsPerPage, keyword, serviceId, relatedTag, userId, scopeGuid, shared, tags, users
  - **response - body:** tag

where:

- **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.
- **keyword**—Specifies a keyword for substring matches.
- **serviceId**—Specifies the service ID for which you want to find tags (null for all).
- **relatedTag**—Specifies a tag that is related to the list of tags. For example, if you have tagged item1 as "webcenter help" and item2 as "webcenter document" and you set relatedTag=webcenter, then you would get both tags (for help and document) returned.
- **userId**—Specifies a single user ID to find tags for (null for all). Uses logged-in user if null and not looking for shared tags.
- **scopeGuid**—Specifies the GUID of the portal. When set to the Home portal, it shows all (unfiltered) tags.
- **shared**—Specify true to return personal resources only (that is, resources tagged privately only by that same user); false to return system resources (that is, resources with at least one public tag).

- **tags**—Specifies to filter by this list of tags separated by URL-encoded spaces (+); for example, an item tagged with "webcenter" and "help" could be searched with tags=webcenter+help.

- **users**—Specifies to filter by this list of users separated by URL-encoded spaces (+); for example, users=monty+vicki+pat.

---

**Note:** Parameters can be used in certain combinations only.

---

**Table 44–4** shows the available parameter combinations for each operation. For example, the tags parameter cannot be specified with the serviceId parameter.

---

### Table 44–4 Parameter Combinations Available for Getting Tags

<table>
<thead>
<tr>
<th>API Method</th>
<th>keyword</th>
<th>serviceld</th>
<th>relatedTag</th>
<th>userId</th>
<th>scopeGuid</th>
<th>shared</th>
<th>tags</th>
<th>users</th>
</tr>
</thead>
<tbody>
<tr>
<td>findRelatedSystemTags</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>findPopularTagsCommon</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>findPopularTags (with GUID)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>findPopularTags</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

### Resource Types Linked to From tags

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.

**Table 44–5** lists the resource types that the client can link to from this resource. Tags have a reference link to themselves and to the next page or previous page of results.

---

### Table 44–5 Related Resource Types for tags

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:tagging:tags</td>
</tr>
<tr>
<td>Previous/Next</td>
<td>urn:oracle:webcenter:tagging:tags</td>
</tr>
</tbody>
</table>

---

**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
```
**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

**Example 44–2  Body of the PUT command for Renaming a Tag**

```
<tag>
  <name>newId</name>
</tag>
```

**Resource Types Linked to From tag**
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**.

Table 44–6 lists the resource types that the client can link to from this resource. A tag has a reference link to itself, to related taggedItems, to related users, and to related tags.

**Table 44–6  Related Resource Types for tag**

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:tagging:tag</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:tagging:taggedItems</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:tagging:users</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:tagging:tags</td>
</tr>
</tbody>
</table>

**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:
Supported Methods for taggedItems

The following methods are supported by this resource:

- **POST**
  - **request - body**: taggedItem
  - **response - body**: taggedItem

```xml
<taggeditem>
  <serviceId>oracle.webcenter.page</serviceId>
  <resourceId>/oracle/webcenter/page/somepath/someotherpath/a.jspx</resourceId>
  <tagWords>foo bar boo far</tagWords>
  <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, relatedRID, tags, users, dt, scopeGuid, shared, itemsPerPage
  - **response - body**: tagged items

where:

- **serviceId**—Specifies the service ID of the item.
- **keyword**—Specifies a keyword in the resource title.
- **relatedRID**—Specifies a related tag. For example, if you tagged item1 as "webcenter help" and item2 as "fusionapps documents" and item3 as "webcenter documents" and set relatedRID=item3, then you would get back item1 and item2, because they share tags in common with item3.
- **tags**—Specifies to filter by this list of tags separated by URL-encoded spaces (+); for example, an item tagged with "webcenter" and "help" could be searched with tags=webcenter+help.
- **users**—Specifies to filter by these users, separated by URL-encoded spaces (+); for example, {USER}+{USER}+...+{USER}.
- **dt**—Specifies to filter bookmarks by this date, which defaults to the most recent date on which bookmarks were saved {'DD-MM-CCYY'}.
- **scopeGuid**—Specifies the GUID of the portal. When set to the Home portal, it shows all (unfiltered) tags.
- **shared**—Specify true to return personal resources only (that is, resources tagged privately only by that same user); false to return system resources (that is, resources with at least one public tag).
- **itemsPerPage**—Specifies the maximum number of results to return in the response (1-n). This is used for pagination.
Table 44–7 shows the available parameter combinations for each operation. For example, the `tags` parameter cannot be specified with the `serviceId` parameter.

**Table 44–7 Parameter Combinations Available for Getting Tagged Items**

<table>
<thead>
<tr>
<th>API Method</th>
<th>serviceId</th>
<th>keyword</th>
<th>relatedRID</th>
<th>tags</th>
<th>users</th>
<th>dt</th>
<th>scopeGuid</th>
<th>shared</th>
</tr>
</thead>
<tbody>
<tr>
<td>findRelatedSystemResources</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>findUpdatedResources</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>findSystemResources</td>
<td></td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>findFilterPersonalResources</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>findPersonalResources</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>findFilteredPersonalResources</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(no keyword)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Resource Types Linked to From taggedItems**

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`.

Table 44–8 lists the resource types that the client can link to from this resource. Tagged items have a reference link to themselves and to the next page or previous page of results.

**Table 44–8 Related Resource Types for taggedItems**

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:tagging:taggedItems</td>
</tr>
<tr>
<td>Previous/Next</td>
<td>urn:oracle:webcenter:tagging:taggedItems</td>
</tr>
</tbody>
</table>

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:

```text
resourceindex
taggedItems
<taggedItem>
resourceindex
spaces
spaces:resourceindex
spaces:taggedItems
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

  where:
  - serviceId—Specifies the service ID of the item.
  - keyword—Specifies a keyword in the resource title.
  - tags—Specifies to filter by this list of tags separated by URL-encoded spaces (+); for example, an item tagged with "webcenter" and "help" could be searched with tags=webcenter+help.
  - users—Specifies to filter by this list of users separated by URL-encoded spaces (+); for example, users=monty+vicki+pat.
  - dt—Specifies to filter bookmarks by this date, which defaults to the most recent date on which bookmarks were saved [DD-MM-CCYY].
  - shared—Specify true to return personal resources only (that is, resources tagged privately only by that same user); false to return system resources (that is, resources with at least one public tag).
  - itemsPerPage—Specifies the maximum number of results to return in the response (1-n). This is used for pagination.

- **PUT**
  - request - body: taggedItem
  - response - body: taggedItem

```
<taggeditem>
<tagWords>foo bar boo far</tagWords>
</taggeditem>
```

- **DELETE**
  - request - body: none
  - response - body: none

**Resource Types Linked to From taggedItem**

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.

Table 44–9 lists the resource types that the client can link to from this resource. A tagged item has a reference link to itself, related tagged items, and an external link.

**Table 44–9  Related Resource Types for taggedItem**

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:tagging:taggedItem</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:tagging:taggedItems</td>
</tr>
</tbody>
</table>
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:

  - **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.
  - **tags**—Specifies to filter by this list of tags separated by URL-encoded spaces ('+'); for example, an item tagged with "webcenter" and "help" could be searched with `tags=webcenter+help`.
  - **users**—Specifies to filter by this list of users separated by URL-encoded spaces ('+'); for example, `users=monty+vicki+pat`.

**Resource Types Linked to From users**

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.

Table 44–10 lists the resource types that individual users can link to from this resource. Users have a reference link to themselves and to the next page or previous page of results.

**Table 44–9 (Cont.) Related Resource Types for taggedItem**

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>alternate</td>
<td>The resource type depends on the type of resource tagged; for example, a page would be <code>urn:oracle:webcenter:page:page</code></td>
</tr>
</tbody>
</table>

**Table 44–10** Related Resource Types for users

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td><code>urn:oracle:webcenter:tagging:users</code></td>
</tr>
<tr>
<td>Previous/Next</td>
<td><code>urn:oracle:webcenter:tagging:users</code></td>
</tr>
</tbody>
</table>
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 tags.
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 44–3.

Example 44–3 Sample JSF Page Template

```xml
<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">
```
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>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResourceId</td>
<td>Unique identifier of your page. In this case, because the resource ID is meant for a page, enter #{facesContext.viewRoot.viewId}.</td>
</tr>
<tr>
<td>ResourceName</td>
<td>Name of the page. In this case, enter #{facesContext.viewRoot.viewId}. (You can enter a more user-friendly name.)</td>
</tr>
<tr>
<td>ServiceId</td>
<td>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.</td>
</tr>
</tbody>
</table>

9. Click OK.

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 25.2, "Configuring Content Repository
Connections."

18. The source for your page template should now look similar to **Example 44–4**.

**Example 44–4  JSF Page Template with Tags and Search**

```xml
<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%;"/>
            <!-- facet start -->
            <af:panelSplitter orientation="vertical">
                <af:region value="#{bindings.taskservicetaskflowdefinition1.regionModel}"
                    id="taskservicetaskflowdefinition1"
                    inlineStyles="position:absolute; left:0px; width:100%;
                    height:100%"/>
            </af:panelSplitter>
            <!-- facet second -->
            <af:region value="#{bindings.doclibrecentdocuments1.regionModel}"
                id="doclibrecentdocuments1"/>
        </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 `lookseetemplate1` 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 25.2, "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](image)

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](image)

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 tools and 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 4.3, "Customizing How Services Render Resources."

### 44.3.5 Troubleshooting Tags

This section provides troubleshooting tips on implementing tags. It includes the following topics:

- **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**


**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 4.3, "Customizing How Services Render Resources"
This chapter describes how to integrate search at design time. This chapter includes the following topics:

- **Section 45.1, "Introduction to Search"**
- **Section 45.2, "Basic Configuration for Search"**
- **Section 45.3, "Advanced Information for Search"**

For more information about managing and including search, see:

- "Managing Oracle Secure Enterprise Search in WebCenter Portal" in *Administering Oracle WebCenter Portal*
- "Adding Search to a Portal" in *Building Portals with Oracle WebCenter Portal*
- "Searching for Information" in *Using Oracle WebCenter Portal*

### 45.1 Introduction to Search

WebCenter Portal 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.

WebCenter Portal provides two ways of searching your application:

- Oracle Secure Enterprise Search (SES) adapter
- Oracle WebCenter Portal live (delegated) search

**Note:** You can implement either Oracle SES or the live search solution in your application, but not both.

Oracle SES is set as the default and preferred search platform. In addition to providing better scalability and performance than WebCenter Portal's live search, Oracle SES is beneficial for the following reasons:

- Oracle SES provides unified ranking results, with the most relevant items appearing first.
- Oracle SES provides more thorough search. For example, when searching lists, WebCenter Portal live search only searches list names and descriptions, while Oracle SES also searches list column names and column contents.
- Oracle SES 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 supports the search REST APIs and data control for customizing your search interface.

In addition to the Oracle SES or live search functionality, the Documents tool 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.

45.1.1 Understanding Search

Search with Oracle SES requires additional configuration in Oracle SES Administration, Oracle WebCenter Content: Content Server, and Oracle WebCenter Portal's Discussion Server.

Oracle strongly recommends using Oracle SES release 11.2.2.2 (included with WebCenter Portal release 11.1.1.8).

Note: This chapter describes how to integrate Oracle SES release 11.2.2.2 in Portal Framework applications. If you choose to use Oracle SES release 11.1.2 or 11.1.2.2 instead, then see:

http://docs.oracle.com/cd/E28280_01/webcenter.1111/e10148/jpsdg_search.htm#BABBDFDC

This section contains the following topics:

- Section 45.1.1.1, "Understanding Search with WebCenter Portal Live Search"
- Section 45.1.1.2, "Understanding Search with the Oracle SES Adapter"
- Section 45.1.1.3, "Click Actions on Search Results"

45.1.1.1 Understanding Search with WebCenter Portal Live Search

Although Oracle SES is the preferred search platform for best performance, you can manually override search with Oracle SES and have applications search using WebCenter Portal's live (delegated) search. WebCenter Portal's live search adapters span all enabled and searchable components, such as documents, tags, people, and pages.

See Also: Section 45.3.2, "Configuring Search with WebCenter Portal's Live Search"

45.1.1.2 Understanding Search with the Oracle SES Adapter

Oracle SES provides unified ranking results for the following resources:

- Documents, including wikis and blogs
- Announcements and discussions

Note: WebCenter Portal applications include the Oracle SES crawler that indexes portals, lists, page metadata, and people resources. This crawler is not supported in Portal Framework applications.

Your search environment varies depending on the version of Oracle SES configured. For example:
Basic Configuration for Search

- If your system is configured with Oracle SES 11.2.2.2, then you can customize search on the Tools and Services - Search administration page. Oracle SES 11.2.2.2 supports faceted search, filtered search in the search box, and document thumbnails, while earlier releases of Oracle SES and implementations with live (delegated) search do not. (All customization with Oracle SES 11.2.2.2 is done on the Tools and Services - Search administration page, even though task flow parameters may display.)

- If your system is configured with Oracle SES 11.1.2.*, then you can customize search using Search - Non-Faceted Search task flow parameters. Oracle SES 11.1.2.* supports saving searches and setting user preferences with search, while the 11.2.2.2 adapter and implementations with live (delegated) search do not.

**Note:** Facets let users navigate indexed data without running a new search. Some search terms can provide massive results, but faceted navigation within search lets users clarify exactly what they are looking for, or even discover something new. Facets count the full corpus and have better response time that the search refiners used in earlier releases.

The Tools and Services - Search administration page shows what search is configured.

**See Also:** Section 45.3.1, "Configuring Search with the Oracle SES Adapter"

### 45.1.3 Click Actions on Search Results

If no Resource Action Handling is specified in the 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 components 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 4.3, "Customizing How Services Render Resources."

### 45.1.2 What Happens at Runtime

At runtime, users can perform global (application-wide) searches.

For more information about WebCenter Portal's search at runtime, see the "Searching for Information" chapter in Using Oracle WebCenter Portal.

### 45.2 Basic Configuration for Search

This section provides the steps for adding search to your application. It includes the following topics:

- Section 45.2.1, "Configuration Roadmap - Oracle SES"
- Section 45.2.2, "Setting up Connections for Oracle SES"
- Section 45.2.3, "Adding Search at Design Time"
- Section 45.2.4, "Setting Security for Search"

### 45.2.1 Configuration Roadmap - Oracle SES

Use the roadmap in this section as a guide through the configuration process.
Basic Configuration for Search

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 Search 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 Portal Framework applications.
Figure 45–1  Configuring Oracle SES for Portal Framework Applications

1. Install Portal Framework 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
   - Use Oracle SES Admin Tool
   - or
   - Use WLST
5. Create two crawl sources: Documents Crawler and Discussions Crawler
   - Use Oracle SES Admin Tool
   - or
   - Use WLST
6. Create a source group for the crawl sources
   - Use Oracle SES Admin Tool
   - or
   - Use WLST
7. Create facets and sorting attributes
   - Use Oracle SES Admin Tool with input files provided by WebCenter Portal
8. Integrate search in your application
9. Configure a connection to Oracle SES in JDeveloper
   - Use JDeveloper
   - or
   - Use Fusion Middleware Control
   - or
   - Use WLST
   - or
   - Use WLS Admin Console
10. Deploy the application
11. Optional: Secure the connection to Oracle SES with SSL
12. Configure additional search parameters
   - Use Fusion Middleware Control
   - or
   - Use WLST
   - or
   - Use Oracle SES Admin Tool
End
### Table 45–1 Configuring Oracle SES for Portal Framework Applications

<table>
<thead>
<tr>
<th>Actor</th>
<th>Task</th>
<th>Sub-task</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrator</td>
<td>1. Install WebCenter Portal and Oracle SES 11.2.2.2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2. Configure Oracle SES with an identity management system</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3. Set up a Document Service Manager in Oracle SES</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4. Create a Federation Trusted Entity using one of the following tools:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ Oracle SES Admin Tool</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ WLST</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5. Create two crawl sources: Documents Crawler and Discussions Crawler using one of the following tools:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ Oracle SES Admin Tool</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ WLST</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6. Create a source group for the crawl sources using one of the following tools:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ Oracle SES Admin Tool</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ WLST</td>
<td></td>
</tr>
<tr>
<td></td>
<td>7. Create facets and sorting attributes</td>
<td></td>
</tr>
<tr>
<td>Developer</td>
<td>8. Integrate search in your application</td>
<td>8.a Configure a connection to Oracle SES in JDeveloper</td>
</tr>
<tr>
<td></td>
<td>8.a Configure a connection to Oracle SES in JDeveloper</td>
<td><strong>Note:</strong> This step must include running the <code>setSESVersion</code> WLST command to enable the Tools and Services - Search administration page.</td>
</tr>
<tr>
<td></td>
<td>8.b Add a Search task flow to a page in JDeveloper</td>
<td></td>
</tr>
<tr>
<td>Developer or Administrator</td>
<td>9. Deploy the application using one of the following tools:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ JDeveloper (Developer)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ Fusion Middleware Control (Administrator)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ WLST (Administrator)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ WLS Admin Console (Administrator)</td>
<td></td>
</tr>
<tr>
<td>Administrator</td>
<td>10. (Optional) Secure the connection to Oracle SES with SSL</td>
<td></td>
</tr>
<tr>
<td>Administrator</td>
<td>11. Configure additional search parameters using one of the following tools:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ Fusion Middleware Control</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ WLST</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ WLS Admin Console</td>
<td></td>
</tr>
</tbody>
</table>
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 45.3.1, "Configuring Search with the Oracle SES Adapter."

1. Add the search toolbar to a page, following Section 45.2.3, "Adding Search 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 SES Connection.

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. Search 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**.
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

```xml
<ses-properties>
  <connection>SESconnection</connection>
  <data-group>MySources</data-group>
</ses-properties>
```

3. You must run the setSESVersion WLST command to obtain and store version information for the Oracle SES instance associated with your default connection. This enables the Tools and Services - Search administration page, which is necessary for customizing search settings with Oracle SES 11.2.2.2.

To confirm that the Oracle SES version is set correctly, run the listSESVersion WLST command.

For command syntax and examples, see the "setSESVersion" and "listSESVersion" sections in WebLogic Scripting Tool Command Reference.

**Note:** After your application has been deployed to an Oracle WebLogic managed server, you can configure it using WLST or Fusion Middleware Control. For more information, see Administering Oracle WebCenter Portal.

### 45.2.3 Adding Search at Design Time

This section describes the search task flows and explains how to integrate search into your application. It includes the following topics:

- Section 45.2.3.1, "Search Task Flows"
- Section 45.2.3.2, "How to Add Search to Your Page"
- Section 45.2.3.3, "How to Customize Search"

#### 45.2.3.1 Search Task Flows

Table 45–2 lists the search task flows included with Oracle SES 11.2.2.2.
Basic Configuration for Search

### 45.2.3.2 How to Add Search to Your Page

The Search - Toolbar task flow offers the easiest way to add search to your application. To add the Search - Toolbar task flow to your application:

1. Follow the steps described in Chapter 2, "Setting Up 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 74.3, "Configuring ADF Security."

2. Open the page on which to add search.


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.

7. The **Search-Faceted Search** task flow requires the **Search - Administration** task flow for access to the Search administration page (for customizing search settings). Add this task flow the same way you added the **Search - Toolbar** task flow.

### Table 45-2 Search Task Flows

<table>
<thead>
<tr>
<th>Task Flow</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search - Faceted Search</td>
<td>This task flow provides a rich search experience supporting faceted search, filtered search in the search box, and document thumbnails. Note: This Search task flow is provided in environments where Oracle SES 11.2.2.2 is configured.</td>
</tr>
<tr>
<td>Search - Administration</td>
<td>This task flow allows access to the Tools and Services - Search administration page for customizing search settings.</td>
</tr>
<tr>
<td>Search - Toolbar</td>
<td>This task flow enables users to enter simple search criteria and run the search from the application. Search results are rendered as links.</td>
</tr>
</tbody>
</table>

**Note:** The other Search task flows that display are not supported with the **Search-Faceted Search** task flow.
45.2.3.3 How to Customize Search

In previous releases configured with Oracle SES, application specialists could customize search (that is, alter the layout of search results as well as search behavior) using the Search (Search - Non-Faceted Search) task flow parameters. With these parameters, they could narrow the scope of searches to specific portals, tools/services, and document types. Also, they could add custom attributes to the list of standard attributes returned with each search result item, and they could hide standard refiners available to users with search results.

With WebCenter Portal release 11.1.1.8 and Oracle SES 11.2.2.2, search is customized completely on the Tools and Services - Search administration page (instead of using task flow parameters). This is a much easier way to configure search. This new search experience makes use of faceted search. This new Search - Faceted Search task flow is used only if Oracle SES 11.2.2.2 is configured.

45.2.4 Setting Security for Search

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 "Oracle SES - Configuration" section in Administering Oracle WebCenter Portal.

Note: WebCenter Portal's live (delegated) search relies on individual tools and 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 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 74.3, "Configuring ADF Security."

45.3 Advanced Information for Search

This section describes optional features available with search. It includes the following topics:

- Section 45.3.1, "Configuring Search with the Oracle SES Adapter"
- Section 45.3.2, "Configuring Search with WebCenter Portal's Live Search"
- Section 45.3.3, "Using the Search Java APIs"
- Section 45.3.4, "Using the Search REST APIs"
45.3.1 Configuring Search with the Oracle SES Adapter

WebCenter Portal applications automatically set Oracle 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 "Managing Oracle Secure Enterprise Search in WebCenter Portal" section in Administering 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 45–4 shows a sample results page for the search term webcenter, before Oracle SES has crawled anything.

   ![Search Results](image)

   **Figure 45–4** Search Results

   Figure 45–5 shows a sample results page after Oracle SES has been configured and crawled.

   ![Search Results](image)

   **Figure 45–5** Search Results After Oracle SES has Crawl
45.3.2 Configuring Search with WebCenter Portal's Live Search

You can manually override search with Oracle SES and have applications search using WebCenter Portal's live (delegated) search.

To enable search with WebCenter Portal's live search, 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

```xml
<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>
</searchC:adf-search-config>
```

See Also: Oracle SES documentation provided on the Oracle Fusion Middleware documentation library (in WebCenter Portal's product area)
Example 45–3  **Crawl Properties in adf-config.xml Commented Out**

```xml
<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"/>
</searchC:adf-search-config>
```

45.3.3  **Using the Search Java APIs**

Custom components can implement search APIs to expose their content to WebCenter Portal’s search.

For more information, see *Java API Reference for Oracle WebCenter Portal.*

45.3.4  **Using the Search REST APIs**

WebCenter Portal provides REST APIs that let you build a customized search user interface. With Oracle SES 11.2.2.2, you can use the search REST APIs to read (GET) searches and facets. Facets are returned with search results and can be used to filter the results. You can specify keywords and the scope of the search; for example, suppose a search for the term ‘webcenter’ produces thousands of results. A user could use facets (such as author or last modified date) in the URL to get the results for particular facet values, like author=Karen and last-modified-date=this week.

This section includes the following topics:

- [Section 45.3.4.1, "Search Entry Point"](#)
- [Section 45.3.4.2, "Search Resource Type Taxonomy"](#)
- [Section 45.3.4.3, "Security Considerations"](#)
- [Section 45.3.4.4, "Search Resource Types"](#)

For an introduction to the REST APIs, see Chapter 53, "Using Oracle WebCenter Portal REST APIs.”
**45.3.4.1 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 search, find the link element with a resourceType of:

```
urn:oracle:webcenter:searchcollection
```

The corresponding href or template element provides the URI entry point. The client sends HTTP requests to this entry point to work with search.

**See Also:** There are two entries in search in the resource index:

- `urn:oracle:webcenter:searchcollection` (This requires WebCenter Portal to be configured with Oracle SES 11.2.2.2.)
- `urn:oracle:webcenter:searchresults` (This requires WebCenter Portal to be configured with Oracle SES—any supported version.)

This chapter describes only `urn:oracle:webcenter:searchcollection`. For detailed information about `urn:oracle:webcenter:searchresults`, see:

```
http://docs.oracle.com/cd/E28280_01/webcenter.1111/e10148/jpsdg_search.htm
```

For more information about the resource index, see Section 53.5.1, "Using the Resource Index."

For more information about resource types, see Section 53.5.2.1, "Resource Type."

**45.3.4.2 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 53.5.2.1, "Resource Type."

The taxonomy for the search is:

```
urn:oracle:webcenter:searchcollection
  urn:oracle:webcenter:searchcollection: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). The entry point returns all the results in searchcollection element. Query various components by specifying appropriate values in dataType argument.

Support for searchTerms, data, and dataType where dataType has one or more of the following values:

- resultCount - number of results satisfying the query
- results - actual results
- facetCount - number of facets for the query result
- facets - facets for the current results

Specify more than one dataType with comma separated values. For example:

```
http://host:port//rest/api/searchcollection?&dataType=results,facets
&facetParams=author:weblogic&utoken=FDX7xKPzrrnsbWIRHNP-b-iUoWiJd4_w\*\\\\*\*\*\*\*\*\*\*\*\*\*\*
```
Navigation Paths to results
This section shows how the client can navigate through the hypermedia to access this resource:

resourceindex
   searchcollection

resourceindex
   spaces
      spaces:resourceindex
      spaces:searchcollection

Supported Methods for results
The following methods are supported by this resource:

- **GET**
  - **request - body**: none, **Parameters**: q, dataType, startIndex (pagination), itemsPerPage (pagination), data, facetParams
  - **response - body**: search results, and optionally facets

where:

- **q**={queryString}

  Searches may be specified using the following format:

  ```
  [[field1:operator][::value1][field2:operator: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 a keyword simple string with no ":" colon separator.

  The query string does not support anything other than a query keyword.

- **dataType** is the type of results

  For example, if you want only results, then dataType=results (or left empty). To include facets in the results, then dataType=results,facets.

- **data** is parameters for custom attributes

  For example:

  ```
  data=numberofreplies,wc_tagsWords
  ```

- **facetParams** is parameters for facet refinement

  Default set is Service ID, scope GUID, Tags, Author, Last Modified Date, Mimetype.

  For example, to refine by people, set the facet Service ID only to oracle.webcenter.people:

  ```
  facetParams=Service%20ID:oracle.webcenter.people
  ```

---

**Note**: If you want to use the facetParams parameter, then the dataType parameter must include facets.

---

**Example 45–4  Search REST Commands**

http://examplehost:8888/rest/api/searchcollection?q={keyword}
&data=dDocName,dOriginalName&dataType=results,facets
&facetParams=Service%20ID:oracle.webcenter.people

For more information, see Section 53.5.2.5, "Templates."

See Also: For detailed information on Oracle SES query syntax, see:
http://docs.oracle.com/cd/E21698_01/admin.1122/e21605/search001.htm#BABFDAGD.

Writable Elements for results
Table 45–3 lists the writable elements for this resource.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>author</td>
<td>String</td>
<td>Any user who has written to the result</td>
</tr>
<tr>
<td>resourceId</td>
<td>String</td>
<td>Unique identifier</td>
</tr>
<tr>
<td>serviceId</td>
<td>String</td>
<td>Service ID of the result; for example, oracle.webcenter.doclib</td>
</tr>
<tr>
<td>title</td>
<td>String</td>
<td>Title of result</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>Description of result</td>
</tr>
<tr>
<td>modified</td>
<td>Date</td>
<td>Last modified date</td>
</tr>
<tr>
<td>size</td>
<td>Number</td>
<td>Size of result</td>
</tr>
<tr>
<td>docid</td>
<td>String</td>
<td>Document identifier</td>
</tr>
<tr>
<td>created</td>
<td>Date</td>
<td>Original author of result</td>
</tr>
<tr>
<td>icon</td>
<td>String</td>
<td>Icon for result type</td>
</tr>
<tr>
<td>url</td>
<td>String</td>
<td>URL of result</td>
</tr>
<tr>
<td>mimetype</td>
<td>String</td>
<td>Mimetype of result</td>
</tr>
<tr>
<td>guid</td>
<td>String</td>
<td>GUID value of result</td>
</tr>
<tr>
<td>scope</td>
<td>String</td>
<td>GUID value of the portal</td>
</tr>
<tr>
<td>scopename</td>
<td>String</td>
<td>Portal name</td>
</tr>
<tr>
<td>resourceType</td>
<td>String</td>
<td>Type of result</td>
</tr>
<tr>
<td>language</td>
<td>String</td>
<td>Language of result</td>
</tr>
<tr>
<td>snippet</td>
<td>String</td>
<td>Snippet of result</td>
</tr>
<tr>
<td>modifier</td>
<td>String</td>
<td>Modifier of result</td>
</tr>
<tr>
<td>customattributes</td>
<td>any</td>
<td>List of any custom attributes specified in data</td>
</tr>
</tbody>
</table>

Resource Types Linked to From results
Table 45–4 lists the resource types that the client can link to from this resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td></td>
</tr>
</tbody>
</table>
45.3.4.3 Security Considerations
Authentication is required before using search REST API methods.

For general security considerations, see Section 53.8, "Security Considerations for WebCenter Portal REST APIs."

45.3.4.4 Search Resource Types
This section provides information about each resource type. It includes the following topics:

- Section 45.3.4.4.1, "urn:oracle:webcenter:searchcollection:links"
- Section 45.3.4.4.2, "urn:oracle:webcenter:searchcollection:results"
- Section 45.3.4.4.3, "urn:oracle:webcenter:searchcollection:facets"

45.3.4.4.1 urn:oracle:webcenter:searchcollection:links
Use this resource type for template links and pagination links.

45.3.4.4.2 urn:oracle:webcenter:searchcollection:results
Use this resource type to identify the URI to use to read (GET) a query containing keywords and facets.

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.

45.3.4.4.3 urn:oracle:webcenter:searchcollection:facets
Use this resource type to identify the URI to use to read (GET) a query containing facets. Facets are returned with the query results and can be used to filter the results. Users can use these facets in the URL to get the results for particular facet values, like author=weblogic and last-modified-date=this week.

The request is represented by the URL and the response is a list of facets, 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.

45.3.5 Using the Search Data Control
Use the Search data control to build a customized search user interface with an 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 application Resource Catalog to build site templates.

45.3.5.1 Search Data Control Methods, Attributes, and Classes
The Search data control contains the searchSes method. Table 45–5 describes its parameters for setting search refiners and limiting the scope of the search.
The `searchSes` method returns:
- Collection of `SearchServiceDCRows` objects
- Collection of `SearchServiceDCRefiner` objects

### 45.3.5.1.1 `SearchServiceDCRows`
- author
- comments
- createDate
- description
- iconPath
- language
- largeIconPath
- lastModifiedDate
- resourceId
- scopeGuid
- serviceId
- size
- tagWords
- title
- type
- `urlSearchServiceDCRefiner`

### 45.3.5.1.2 `SearchServiceDCRefiner` element

### 45.3.5.1.3 `SearchServiceDCRefinement`

---

**Table 45–5  `searchSes` Input Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>scopeGuid</code></td>
<td>No</td>
<td>String</td>
<td>GUID value to filter results by portal.</td>
</tr>
<tr>
<td><code>serviceId</code></td>
<td>No</td>
<td>String</td>
<td>The service ID of the item; for example, oracle.webcenter.doclib searches only documents. If omitted, this searches across all tools and services.</td>
</tr>
<tr>
<td><code>keywords</code></td>
<td>Yes</td>
<td>String</td>
<td>Search terms.</td>
</tr>
<tr>
<td><code>predicates</code></td>
<td>No</td>
<td>List</td>
<td>Collection of predicates.</td>
</tr>
<tr>
<td><code>fetchRefiners</code></td>
<td>No</td>
<td>Boolean</td>
<td>Whether to fetch refiners. If omitted, the default value is false.</td>
</tr>
<tr>
<td><code>startIndex</code></td>
<td>Yes</td>
<td>Integer</td>
<td>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.</td>
</tr>
<tr>
<td><code>itemsPerPage</code></td>
<td>Yes</td>
<td>Integer</td>
<td>Number of results to display.</td>
</tr>
</tbody>
</table>
■ keywords
■ scopeGuid
■ serviceId
■ Collection of predicates

45.3.5.2 Integrating the Search Data Control

To use the Search data control:

1. In JDeveloper, create an 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 4.1.2, "Understanding WebCenter Portal APIs."

45.3.6 Building Adapters for Search

You can use search adapters to add other sources to your search results. WebCenter Portal automatically discovers these search adapters and consolidates them in formulating search results in your application. Subsequently, when you expose your custom component to search 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 topics:

■ Section 45.3.6.1, "How to Add a Search Source"
■ Section 45.3.6.2, "How to Register a Custom Adapter"
■ Section 45.3.6.3, "Search Adapter Attributes"
■ Section 45.3.6.4, "What Happens at Runtime"

45.3.6.1 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 search.

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...

---

**Note:** Oracle SES is set as the default search platform. For this sample code to work with WebCenter Portal’s live search adapters instead, you must edit `crawl-properties` in `adf-config.xml`. For details, see Section 45.3.2, "Configuring Search with WebCenter Portal’s Live Search."

---

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 45–6**, then click **Add Library** and select it.
3. Create a class called SampleQueryManager.java (Example 45–5).

Example 45–5  SampleQueryManager.java

```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

```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_AUTHOR;
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.

```java
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**

```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**

```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_AUTHOR;
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 author, 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_AUTHOR, author);
        // 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();
    }
}

7. Create a class called CustomScopeAPIProvider.java.

This enables the custom search service, which must be provisioned by implementing the Service Provider Interface (SPI) - ServiceScopeAPI as shown in Example 45–9.
Example 45–9 CustomScopeAPIProvider.java

```java
package mycompany.myproduct.myapp.mycomponent.query;

import java.util.Collection;
import oracle.webcenter.framework.service.Scope;
import oracle.webcenter.command.CommandException;
import oracle.webcenter.command.spi.ServiceScopeAPI;
import oracle.webcenter.command.CopyOptions;

public class CustomScopeAPIProvider implements ServiceScopeAPI {

    /**
     * @inheritDoc
     */
    public void provision(Scope scope, String scopeDisplayName) throws CommandException {
    }

    /**
     * @inheritDoc
     */
    public void unprovision(Scope scope) throws CommandException {
    }

    /**
     * @inheritDoc
     */
    public boolean isProvisioned(Scope scope) throws CommandException {
        return true;
    }

    /**
     * @inheritDoc
     */
    public void copy(Scope sourceScope, Scope targetScope, CopyOptions options) throws CommandException {
    }

    /**
     * @inheritDoc
     */
    public void rename(Scope source, Scope target) throws CommandException {
    }

    /**
     * @inheritDoc
     */
    public void setReadOnly(Scope scope, boolean readOnly) throws CommandException {
    }

    /**
     * @inheritDoc
     */
    }
```
public boolean isReadOnly(Scope scope) throws CommandException {
    return false;
}

/**
 * @inheritDoc
 */
public boolean isConfigured() {
    return true;
}

/**
 * @inheritDoc
 */
public void updateSecurityHierarchy(Scope scope, Collection<Scope> children)
throws CommandException {
}

45.3.6.2 How to Register a Custom Adapter

To register your query manager with search, you must edit the service-definition.xml file. The SampleQueryManager line exposes the custom adapter's query manager implementation to search so that the custom adapter is included with any search user interface. To register the ServiceScopeAPI class (CustomScopeAPIProvider.java) implemented in Example 45–9, you must add a <command-definition> entry with the class name that implements it as the scope-class as shown in Example 45–10.

Example 45–10 Sample Query Manager Entry in service-definition.xml

<service-definition version="11.1.1.0.0" id="com.oracle.portal.sample">
    <resource-view taskFlowId="/WEB-INF/search-viewer-tf.xml#search-view"/>
    <name>Sample_Search</name>
    <description>Sample Search</description>
    <search-definition xmlns="http://xmlns.oracle.com/webcenter/search" version="11.1.1.0.0" id="com.oracle.portal.sample">
        <query-manager-class>com.oracle.portal.sample.search.SampleQueryManager</query-manager-class>
    </search-definition>
    <command-definition xmlns="http://xmlns.oracle.com/webcenter/command" version="11.1.1.0.0" id="com.oracle.portal.sample">
        <scope-class>com.oracle.portal.sample.search.CustomScopeAPIProvider</scope-class>
    </command-definition>
</service-definition>

For more information about the resource-view tag (and WebCenter Portal's Resource Action Handling framework), see Section 4.3, "Customizing How Services Render Resources."
45.3.6.3 Search Adapter Attributes
This section describes the attributes available for searching. It includes the following topics:

- Section 45.3.6.3.1, "Searchable Attributes"
- Section 45.3.6.3.2, "Keywords"
- Section 45.3.6.3.3, "Date Condition"
- Section 45.3.6.3.4, "Complex Date Condition"
- Section 45.3.6.3.5, "Person Condition"
- Section 45.3.6.3.6, "Complex Person Condition"
- Section 45.3.6.3.7, "Access"
- Section 45.3.6.3.8, "Selectable Attributes"

45.3.6.3.1 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 45.3.6.3.8, "Selectable Attributes."

45.3.6.3.2 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. Search 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.

45.3.6.3.3 Date Condition
For search 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`
- `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`

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`.

45.3.6.3.4 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.

45.3.6.3.5 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.

45.3.6.3.6 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.

45.3.6.3.7 Access Search (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.

45.3.6.3.8 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 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 search 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>.

  Search 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. Search 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 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 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 components, such as documents and pages, 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 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.

- **Author**

  For each resource returned, the search user interface can display the author 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_AUTHOR and AttributeConstants.WPSAPI_MODIFIED, respectively. The author is of type java.lang.String.

- **Icon**

  For each resource returned, the search 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 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.

**45.3.6.4 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 invokes a resource viewer using the Resource Action Handling framework. For information about registering a resource viewer to allow
45.3.7 Customizing the Search UI Without WebCenter Portal’s Search APIs

You can develop a custom search user interface in an application without WebCenter Portal’s search 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: “Setting Up Oracle SES to Search Documents” section in Administering 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.


   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 45–11 shows how to use PopUpResourceActionViewHandler to show the resource in a popup window when a search result is clicked.

Example 45–11  Specifying PopUp Behavior Handler

```xml
<wps:C:adf-service-config
xmlns:wpsC="http://xmlns.oracle.com/webcenter/framework/service"
xmlns="http://xmlns.oracle.com/webcenter/framework/service">
Example 45–12 shows how to use NavigationResourceActionHandler to show the resource in the navigation node of the resource when a search result is clicked.

### Example 45–12  Specifying the Navigation Handler

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

### 45.3.8 Troubleshooting Search

This section describes common problems and solutions for search.

**Problem**

There is a set timeout for querying each component. If a particular component does not return search results in the allotted time, then it times out. Announcements and discussions are susceptible to timeouts. Pages and portals also 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:

```xml
<execution-properties timeoutMs="3000" prepare TimeoutMs="1000"/>
```

At runtime, you can configure timeouts using WLST or Fusion Middleware Control.
This chapter describes how to integrate the activity graph with a Portal Framework application at design time.

This chapter includes the following topics:

- Section 46.1, "Introduction to the Activity Graph"
- Section 46.2, "Basic Configuration for the Activity Graph"
- Section 46.3, "Advanced Information for the Activity Graph"
- Section 46.4, "Extending the Activity Graph"
- Section 46.5, "Troubleshooting the Activity Graph"

For more information about managing and using the activity graph, see:

- "Managing Activity Graph" chapter in *Administering Oracle WebCenter Portal*
- "Adding Activity Graphs and Recommendations to a Portal" chapter in *Building Portals with Oracle WebCenter Portal*
- "Exploring Recommendations and Content" chapter in *Using Oracle WebCenter Portal*

### 46.1 Introduction to the Activity Graph

Users today contribute content at unprecedented rates. The activity graph provides seamlessly integrated information retrieval, leveraging collective intelligence to benefit search and social applications.

This section provides an overview of the activity graph, its task flows and the underlying architecture. It includes the following topics:

- Section 46.1.1, "Understanding the Activity Graph"
- Section 46.1.2, "Requirements for the Activity Graph"
- Section 46.1.3, "What Happens at Runtime"

### 46.1.1 Understanding the Activity Graph

The activity graph 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 portals or content that may be of interest, based on ranking calculations.

The activity graph 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).

![An Activity Graph](image)

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 "Setting Up Activity
Introduction to the Activity Graph

Node Classes
Activity graph nodes are grouped into classes. The default node classes for WebCenter Portal are:

- Users (WC.user)
- Portals (WC.group-space)
- Documents (WC.document)
- Wikis (WC.wiki-page)
- Blogs (WC.blog)
- Discussion Topics (WC.topic)

To extend the activity graph engine to integrate with other applications, 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 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>
<thead>
<tr>
<th>Action URN</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>connect</td>
<td>Connecting with another user</td>
</tr>
<tr>
<td>create</td>
<td>Creating a portal, document, wiki, blog, topic, or message</td>
</tr>
<tr>
<td>edit</td>
<td>For all items except portals:</td>
</tr>
<tr>
<td></td>
<td>- Checking in a new version of document</td>
</tr>
<tr>
<td></td>
<td>- Editing a wiki or blog</td>
</tr>
<tr>
<td></td>
<td>- Replying to a topic</td>
</tr>
<tr>
<td></td>
<td>- Editing a message</td>
</tr>
<tr>
<td></td>
<td>For portals:</td>
</tr>
<tr>
<td></td>
<td>- All of the above plus</td>
</tr>
<tr>
<td></td>
<td>- Creating, editing, or deleting a page</td>
</tr>
<tr>
<td></td>
<td>- Creating, editing, or deleting a list</td>
</tr>
<tr>
<td></td>
<td>- Creating, editing, or deleting an event</td>
</tr>
<tr>
<td>comment</td>
<td>Commenting on a document, wiki, or blog</td>
</tr>
<tr>
<td>like</td>
<td>Liking a document, wiki, blog, or message</td>
</tr>
<tr>
<td>tag</td>
<td>Tagging a document, wiki, or blog</td>
</tr>
<tr>
<td>view</td>
<td>Viewing a portal page, document, wiki, blog entry, or discussion topic</td>
</tr>
<tr>
<td>download</td>
<td>Downloading a document, wiki, or blog</td>
</tr>
</tbody>
</table>
When one of these WebCenter Portal actions occurs, for example, Monty viewing 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).

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.

<table>
<thead>
<tr>
<th>Action URN</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>edit-count</td>
<td>Checking in a new version of a document, editing a wiki or blog, replying to a topic, editing a message</td>
</tr>
<tr>
<td>view-count</td>
<td>Viewing a document, wiki, blog, or discussion topic</td>
</tr>
</tbody>
</table>

Table 46–1  (Cont.) WebCenter Portal Default Actions
To integrate the activity graph engine with other applications, 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 task flows, that queries the relation tables for recommendations using a recipe (Figure 46–4). A recipe is a weighted list of rank or 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.
Table 46–2 lists the calculations used by the activity graph task flows (for information about the task flows, see Section 46.2.3.1, "Activity Graph Task Flows").

<table>
<thead>
<tr>
<th>URN</th>
<th>Applies to</th>
<th>Description</th>
<th>Used by Task Flow</th>
</tr>
</thead>
<tbody>
<tr>
<td>item-edit</td>
<td>Documents</td>
<td>Users who edited the current item also edited the recommended item</td>
<td>Similar Items</td>
</tr>
<tr>
<td></td>
<td>Wikis</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Blogs</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Topics</td>
<td></td>
<td></td>
</tr>
<tr>
<td>item-like</td>
<td>Documents</td>
<td>Users who liked the current item also liked the recommended item</td>
<td>Similar Items</td>
</tr>
<tr>
<td></td>
<td>Wikis</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Blogs</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Topics</td>
<td></td>
<td></td>
</tr>
<tr>
<td>item-comment</td>
<td>Documents</td>
<td>Users who commented on the current item also commented on the recommended item</td>
<td>Similar Items</td>
</tr>
<tr>
<td></td>
<td>Wikis</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Blogs</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Topics</td>
<td></td>
<td></td>
</tr>
<tr>
<td>item-tag</td>
<td>Documents</td>
<td>Users who tagged the current item also tagged the recommended item</td>
<td>Similar Items</td>
</tr>
<tr>
<td></td>
<td>Wikis</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Blogs</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Topics</td>
<td></td>
<td></td>
</tr>
<tr>
<td>item-all</td>
<td>Documents</td>
<td>Users who interacted in any way with the current item, including viewing it,</td>
<td>Similar Items</td>
</tr>
<tr>
<td></td>
<td>Wikis</td>
<td>also interacted in some way with the recommended item</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Blogs</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Topics</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 46–2 (Cont.) Activity Graph Calculations

<table>
<thead>
<tr>
<th>URN</th>
<th>Applies to</th>
<th>Description</th>
<th>Used by Task Flow</th>
</tr>
</thead>
<tbody>
<tr>
<td>gs-edit</td>
<td>Portals</td>
<td>Users who participated in the current portal also participated in the recommended portal</td>
<td>Similar Portals</td>
</tr>
<tr>
<td>gs-all</td>
<td>Portals</td>
<td>Users who interacted in any way with the current portal, including viewing pages or content in the portal, also interacted in some way with the recommended portal</td>
<td>Similar Portals</td>
</tr>
<tr>
<td>user-connect</td>
<td>Users</td>
<td>The current user shares a number of connections with the recommended user</td>
<td>Recommended Connections</td>
</tr>
<tr>
<td>user-edit</td>
<td>Users</td>
<td>The current user edited some items that the recommended user also edited</td>
<td>Recommended Connections</td>
</tr>
<tr>
<td>user-like</td>
<td>Users</td>
<td>The current user likes some items that the recommended user also likes</td>
<td>Recommended Connections</td>
</tr>
<tr>
<td>user-comment</td>
<td>Users</td>
<td>The current user commented some items that the recommended user also commented on</td>
<td>Recommended Connections</td>
</tr>
<tr>
<td>user-tag</td>
<td>Users</td>
<td>The current user tagged some items that the recommended user also tagged</td>
<td>Recommended Connections</td>
</tr>
<tr>
<td>user-all</td>
<td>Users</td>
<td>The current user interacted in some way, including viewing, with some items with which the recommended user also interacted</td>
<td>Recommended Connections</td>
</tr>
<tr>
<td>like-rank</td>
<td>Documents, Wikis, Blogs</td>
<td>Users liked the recommended current item</td>
<td>Top Items</td>
</tr>
<tr>
<td>activity-rank</td>
<td>Documents, Wikis, Blogs</td>
<td>Users interacted in some way with the recommended item</td>
<td>Top Items</td>
</tr>
</tbody>
</table>

You can edit these calculations to change the weightings. You can create additional calculations to integrate the activity graph engine with other applications. 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. The filter uses the same mechanism used by the Search tool for 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 tool. See also Section 4.3, "Customizing How Services Render Resources."

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 Task Flows

Recommendations are presented to users via the activity graph task flows. The activity graph 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 Portals**—Suggests portals with which users have interacted in a similar way as the current portal.

- **Similar Items**—Suggests items with which users have interacted in a similar way as the current item.

- **Top Items**—Suggests items contributed by the current user that have had the most activity (including views) or with which users have shown the most "likes". In a Portal Framework application, this task flow works only in the Top Contributions mode.

For more information, see Section 46.2.3.1, "Activity Graph Task Flows."

### 46.1.2 Requirements for the Activity Graph

The activity graph requires that the activity graph engine has been installed and configured. For more information, see *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 4.2.2, "Setting Up a Database Connection."

The application must be configured to send usage events to the Analytics Event Collector. For more information, see the "Registering an Analytics Collector for Your Application" section in *Administering Oracle WebCenter Portal*. 
Before the activity graph 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 "Preparing Data for the Activity Graph" section in *Administering Oracle WebCenter Portal*.

The items suggested in the Similar Items task flow, Top Items task flow, and Recommendation data control depend on the tools available in your application. For example, documents are only recommended if the Documents tool is available. For information about making a tool available in your application, refer to the appropriate chapter for that tool. An item can also be filtered out of the recommendations by the Resource Authorizer of the tool that owns the item.

### 46.1.3 What Happens at Runtime

At runtime, the context is provided for the activity graph task flow:

- For the Recommended Connections task flow, the context is the current user.
- For the Similar Portals task flow, the context is the current portal.
- For the Similar Items task flow, the context is a WebCenter Portal resource, provided through task flow input parameters or through ADF UI events.
- For the Top Items task flow, the context is the profile owner.

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 tool at runtime, see the "Exploring Recommendations and Content" chapter in *Using Oracle WebCenter Portal*.

### 46.2 Basic Configuration for the Activity Graph

This section describes required steps for adding the activity graph to your application. It includes the following topics:

- Section 46.2.1, "Configuration Roadmap for the Activity Graph"
- Section 46.2.2, "Setting Up Connections for the Activity Graph"
- Section 46.2.3, "Adding the Activity Graph at Design Time"

#### 46.2.1 Configuration Roadmap for the Activity Graph

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 working in Framework applications.
Figure 46–5 Configuring the Activity Graph for Portal Framework Applications

1. Install WebCenter Portal and the back-end components for analytics and the activity graph.
2. Integrate the activity graph in your application.
3. Optional: Set up the activity graph engines schedule.
4. Deploy the application.
5. Optional: Add/modify connection parameters.
6. Test that activity graph data is available in the application. 
   a. Log in and interact with the application, adding content, and so on.
   b. Run the activity graph engines.
   c. View recommendations in the activity graph task flow.

Administrator

Developer

Optional

End User

Figure 46–5 Configuring the Activity Graph for Portal Framework Applications
Basic Configuration for the Activity Graph

Integrating the Activity Graph

Table 46–3  Configuring the Activity Graph for Portal Framework Applications

<table>
<thead>
<tr>
<th>Actor</th>
<th>Task</th>
<th>Sub-task</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrator</td>
<td>Install WebCenter Portal and the back-end components for analytics and the activity graph</td>
<td></td>
</tr>
<tr>
<td>Developer</td>
<td>Integrate the activity graph in your application</td>
<td>2.a Configure a connection to WebCenter Portal’s schema in JDeveloper</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.b Configure a connection to the Activities database in JDeveloper</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.c Configure a connection to the Analytics Collector in JDeveloper</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.d Add an activity graph task flow to a page in JDeveloper</td>
</tr>
<tr>
<td>Administrator</td>
<td>(Optional) Set up the activity graph engines schedule</td>
<td></td>
</tr>
<tr>
<td>Developer/</td>
<td>Deploy the application using one of the following tools:</td>
<td></td>
</tr>
<tr>
<td>Administrator</td>
<td>■ JDeveloper (Developer)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ Fusion Middleware Control</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ WLST (Administrator)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ WLS Admin Console (Administrator)</td>
<td></td>
</tr>
<tr>
<td>Administrator</td>
<td>(Optional) Add/modify connection parameters using one of the following tools:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ JDeveloper, then redeploy the application (Developer)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ Fusion Middleware Control</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ WLST (Administrator)</td>
<td></td>
</tr>
<tr>
<td>End User/</td>
<td>Test that activity graph data is available in the application</td>
<td></td>
</tr>
<tr>
<td>Administrator</td>
<td></td>
<td>6.a Log in and interact with the application, for example, by adding content (End User)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6.b Run the Activity Graph Engines (Administrator)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6.c View recommendations, for example, see the Recommended Connections on your Profile page (End User)</td>
</tr>
</tbody>
</table>

46.2.2  Setting Up Connections for the Activity Graph

The activity graph requires the following connections:

- **WebCenter Portal schema**—For information about how to create this connection, see Section 4.2.2, "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 4.2.2, “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 "Registering an Analytics Collector for Your Application" section in Administering Oracle WebCenter Portal.
Additionally, the items suggested in the Similar Items task flow, the Top Items task flow, and the Recommendation data control depend on the tools and services available in your application. For example, documents are only recommended if the Documents tool is enabled.

- A connection to a content repository is required to see documents, wikis, and blogs.
- A connection to a discussions server is required to see discussions.

**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 *Administering Oracle WebCenter Portal*.

### 46.2.3 Adding the Activity Graph at Design Time

This section provides descriptions of the activity graph task flows and steps you through the addition of the activity graph to your application. It includes the following topics:

- **Section 46.2.3.1, "Activity Graph Task Flows"**
- **Section 46.2.3.2, "How to Add Activity Graph Task Flows to a Page"**
- **Section 46.2.3.3, "How to Modify Activity Graph Task Flow Parameters"**
- **Section 46.2.3.4, "Activity Graph Task Flows and Task Flow Parameters"**

#### 46.2.3.1 Activity Graph Task Flows

This section describes and illustrates the task flows available through the activity graph.

Table 46–4 lists the task flows provided by the activity graph.

<table>
<thead>
<tr>
<th>Task Flow</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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.  
  See **Section 46.2.3.1.1, "The Recommended Connections Task Flow."**                                      |
| Similar Portals    | This task flow enables viewing and interacting with portals that the activity graph engine has calculated to be potentially of interest to the current user.  
  See **Section 46.2.3.1.2, "The Similar Portals 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.  
  See **Section 46.2.3.1.3, "The Similar Items Task Flow."**                                                 |
| Top Items          | This task flow enables viewing and interacting with the WebCenter Portal items (documents, wikis, or blogs) the activity graph has calculated to be most active in the current portal.  
  See **Section 46.2.3.1.4, "The Top 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.

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 Task Flow Parameters."

<table>
<thead>
<tr>
<th>Similarity Calculation</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>user-connect</td>
<td>100</td>
</tr>
<tr>
<td>user-edit</td>
<td>50</td>
</tr>
<tr>
<td>user-like</td>
<td>50</td>
</tr>
<tr>
<td>user-comment</td>
<td>10</td>
</tr>
<tr>
<td>user-tag</td>
<td>10</td>
</tr>
<tr>
<td>user-all</td>
<td>1</td>
</tr>
</tbody>
</table>

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 "Customizing Reason Strings for Similarity Calculations" section in Administering Oracle WebCenter Portal.

46.2.3.1.2 The Similar Portals Task Flow  The Similar Portals task flow enables viewing and interacting with portals that have been calculated to be potentially of interest to users of the current portal. The suggested portals are not specific to the current user, however the task flow lists only those portals that the current user has permission to see.

Table 46–6 shows the similarity calculations and their weightings used in the default recipe for the Similar Portals 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 Task Flow Parameters."

<table>
<thead>
<tr>
<th>Similarity Calculation</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>gs-edit</td>
<td>10</td>
</tr>
<tr>
<td>gs-all</td>
<td>1</td>
</tr>
</tbody>
</table>

The portal with the highest overall recommendation score is displayed first in the task flow. For each recommended portal, if the highest scoring related similarity calculation has an associated reason string, that string is displayed underneath the portal'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 "Customizing Reason Strings for Similarity Calculations" section in *Administering 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, 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 tool task flows:
  - Recent Documents
  - Document List Viewer
  - Document Navigator
  - Document Explorer
  - Folder Viewer
  - Document Manager
  - Document Browser

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 information about how to use this task flow at runtime, see the "Adding Activity Graphs and Recommendations to a Portal" section in *Administering Oracle WebCenter Portal*.

To see recommendations for a discussion topic, edit the Similar Items task flow `resourceID` parameter and enter the `resourceID` of the discussion topic.

*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 Task Flow Parameters."

<table>
<thead>
<tr>
<th>Similarity Calculation</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>item-edit</td>
<td>100</td>
</tr>
<tr>
<td>item-like</td>
<td>50</td>
</tr>
<tr>
<td>item-comment</td>
<td>20</td>
</tr>
<tr>
<td>item-tag</td>
<td>10</td>
</tr>
<tr>
<td>item-all</td>
<td>1</td>
</tr>
</tbody>
</table>

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 "Customizing Reason Strings for Similarity Calculations" section in Administering Oracle WebCenter Portal.

46.2.3.1.4 The Top Items Task Flow

The Top Items task flow enables viewing and interacting with the items the activity graph has calculated to be active in the current portal. Top items may include documents, wikis, and blogs. The items are not specific to the current user, however the task flow lists only those items that the current user has permission to see.

In a Portal Framework application, this task flow works only in the Top Contributions mode, which enables viewing and interacting with the WebCenter Portal items the activity graph has calculated to be the top contributions by the current user. This is available by default on each user’s Profile page.

Table 46–8 shows the rank calculations and their weightings used in the default recipe for the Top Items task flow. You can change this recipe by editing the recipe task flow binding parameter. For more information, see Section 46.2.3.3, "How to Modify Activity Graph Task Flow Parameters."

### Table 46–8 Default Recipe for Top Items

<table>
<thead>
<tr>
<th>Top Items Calculation</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>activity-rank</td>
<td>50</td>
</tr>
<tr>
<td>like-rank</td>
<td>50</td>
</tr>
</tbody>
</table>

46.2.3.2 How to Add Activity Graph 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 task flows.

To add the Recommended Connections task flow to your Portal Framework application:

1. Prepare your application as described in Section 46.1.2, "Requirements for the Activity Graph."
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 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 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 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. Right-click the page in Design view and choose **Edit Authorization**. Navigate to **Granted to > Select authenticated-users**, and under **Actions**, grant authenticated users **Customize**, **Grant**, **Personalize**, and **View** privileges.

9. Save and run your page.

### 46.2.3.3 How to Modify Activity Graph Task Flow Parameters

Each activity graph 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 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.

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 Task Flows and Task Flow Parameters").
5. When you are finished, click **OK**.

6. Save and run your page to see the results.

### 46.2.3.4 Activity Graph Task Flows and Task Flow Parameters

Table 46–9 lists and describes task flow binding parameters applicable to the activity graph.

**Table 46–9: Activity Graph Task Flow Binding Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Task Flows</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>classURN</td>
<td>Similar Items</td>
<td>The node class of the context. By default the information is derived from the selection event.</td>
</tr>
<tr>
<td>classURNRestrictions</td>
<td>Similar Items, Top Items</td>
<td>A comma-separated list of the node classes to include in the recommendations. If non-null, then only objects from the given node classes are included in the recommendations. By default this is empty, meaning that there are no restrictions on which classes of objects may be returned.</td>
</tr>
<tr>
<td>excludeObjectActions</td>
<td>Recommended Connections, Similar Portals, Similar Items</td>
<td>A comma-separated list of registered actions. The task flow will not show people, portals, 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 default actions are listed in Table 46–1.</td>
</tr>
</tbody>
</table>
This section describes advanced features available with the activity graph. It includes the following topics:

### Table 46–9 (Cont.) Activity Graph Task Flow Binding Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Task Flows</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>objectName</strong></td>
<td>Similar Portals, Similar Items</td>
<td>The name of the portal or item to use as a context for recommendations. For portals, the default value is the expression language token for the resource ID of the portal containing the task flow. For items, the value is derived from the selection event.</td>
</tr>
<tr>
<td><strong>recipe</strong></td>
<td>Recommended Connections, Similar Portals, Similar Items, Top Items</td>
<td>A comma-separated list of registered similarity calculation = weight pairs (or for Top Items, rank 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 Task Flows.”</td>
</tr>
<tr>
<td><strong>resourceId</strong></td>
<td>Recommended Connections, Similar Portals, Similar Items</td>
<td>The ID of the person, portal, 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 portals, the default value is the expression language token for the resource ID of the portal containing the task flow. For items, the value is derived from the selection event.</td>
</tr>
<tr>
<td><strong>resourceType</strong></td>
<td>Similar Items</td>
<td>Used in conjunction with resourceId and serviceId, as an alternative to classURN for identifying the node class of an object. For example resourceType is used by the Documents tool to identify whether an object is a wiki, blog, or document.</td>
</tr>
<tr>
<td><strong>serviceId</strong></td>
<td>Similar Items</td>
<td>Used in conjunction with resourceId, and optionally resourceType, as an alternative to classURN for identifying the node class of an object.</td>
</tr>
<tr>
<td><strong>suppressPopup</strong></td>
<td>Similar Items, Top Items</td>
<td>Determines whether links in the task flow launch content in inline popups (false) or in browser windows (true). The default value is false.</td>
</tr>
</tbody>
</table>
46.3.1 Using the Recommendation Data Control

The activity graph provides a data control that enables you to create your own user interface (task flows) for the tool.

The Recommendation data control exposes two methods to query the activity graph engine 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 Recommendation data control to your project:

1. Add the **Recommendation Data Control** 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 **Recommendation Data Control** 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 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:
For a Similar Portals task flow:
gs-edit=10, gs-all=1

For a Top Items task flow:
activity-rank=50, like-rank=50

- classURNRestrictionList—A comma-separated list of the node classes to include in the recommendations. If non-null, then only objects from the given node classes are included in the recommendations.
- 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 REST APIs

WebCenter Portal provides REST APIs to support the activity graph. Use the activity graph REST APIs to create your own interface for providing recommendations for connections, portals, and items.

This section describes the REST APIs associated with the activity graph. It includes the following topics:

- 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 53, "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, there are two entry points: one for recommendations and one for items. To find the entry points for the activity graph, 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.

For more information about the Resource Index, see Section 53.5.1, "Using the Resource Index."

For more information about resource types, see Section 53.5.2.1, "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 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

For general security considerations, see Section 53.8, "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 topics:

- 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, portals, 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, portals, 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 tool, 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-document`.

    * **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 portals, then the `gs-edit` action could be specified to exclude portals that the current user has edited (since the user already knows about those portals).

    * **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 53.5.2.5, "Templates."
Resource Types Linked to from recommendations

Table 46–10 lists the resource types that the client can link to from this resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:activitygraph:recommendations</td>
</tr>
</tbody>
</table>

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 46–11 lists the read-only elements for the recommendations resource.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>score</td>
<td>Float</td>
<td>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.</td>
</tr>
<tr>
<td>item</td>
<td>urn:oracle:webcenter:activitygraph:items:item</td>
<td>The recommended user, item, or portal.</td>
</tr>
<tr>
<td>componentScores</td>
<td>A list of componentScore elements</td>
<td>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 that can be used to retrieve the common items with which the user and the recommended object have interacted.</td>
</tr>
</tbody>
</table>

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-document.
    * **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 53.5.2.5, "Templates."

**Resource Types Linked to from items**
Table 46–12 lists the resource types that the client can link to from this resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:activitygraph:items</td>
</tr>
</tbody>
</table>

**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 46–13 lists the writable elements for this resource type.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Required</th>
<th>Constraints</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>String</td>
<td>Yes</td>
<td>interested</td>
<td>Whether the user is interested in the object</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>not interested</td>
<td></td>
</tr>
</tbody>
</table>

Read-only Elements for item
Table 46–14 lists the read-only elements for this resource type. Not all of these elements are available for all objects.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>classURN</td>
<td>String</td>
<td>Node class of the object</td>
</tr>
<tr>
<td>objectURN</td>
<td>String</td>
<td>Identifier for the object</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the object</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>Description of the object</td>
</tr>
<tr>
<td>modified</td>
<td>Date</td>
<td>Date on which the object was last updated</td>
</tr>
<tr>
<td>modifiedByUser</td>
<td>PersonReference</td>
<td>User information about the user who last updated the object, including GUID, ID, display name, and a link to the profile icon</td>
</tr>
<tr>
<td>author</td>
<td>PersonReference</td>
<td>User information about the user who created the object, including GUID, ID, display name, and a link to the profile icon</td>
</tr>
</tbody>
</table>

Resource Types Linked to from item
Table 46–15 lists the resource types that the client can link to from this resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:activitygraph:items:item</td>
</tr>
<tr>
<td>via</td>
<td>resourceType of the underlying object</td>
</tr>
</tbody>
</table>
46.4 Extending the Activity Graph

Out of the box, activity graph includes metadata definitions for mapping WebCenter Portal 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:

1. For information about exporting the activity graph metadata definitions, see the "Exporting Activity Graph Metadata" section in Administering Oracle WebCenter Portal.
2. For information about exporting analytics mapping metadata definitions, see the "Exporting Provider Configuration" section in Administering 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 "Importing Activity Graph Metadata" section in Administering Oracle WebCenter Portal.

This section includes the following topics:

- 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, portals, 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 "Exporting Activity Graph Metadata" section in Administering 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:

```xml
<properties>
    <property name="oracle.webcenter.resourceType" value="wiki" />
    <property name="oracle.webcenter.serviceID" value="oracle.webcenter.doclib" />
</properties>
```

**Example 46–1  Node Class Definition**

```xml
<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 Portal Framework application is deployed. For more information, see the "Importing Activity Graph Metadata" section in *Administering Oracle WebCenter Portal*.

### 46.4.2 Defining Custom Actions

Activity graph includes metadata definitions for the actions that occur in Portal 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 "Exporting Activity Graph Metadata" section in *Administering 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. 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 People Connections. 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

```xml
<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 Portal Framework application is deployed. For more information, see the "Importing Activity Graph Metadata" section in Administering 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 "Exporting Activity Graph Metadata" section in Administering 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**

```xml
<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 Portal Framework application is deployed. For more information, see the “Importing Activity Graph Metadata” section in *Administering 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.
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 "Exporting Activity Graph Metadata" section in *Administering 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**

```xml
<rankCalculations>
  <rankCalculation URN="activity-rank">
    <domainClasses>
      <URN>WC.user</URN>
    </domainClasses>
  </rankCalculation>
</rankCalculations>
```
3. Use the WLST command `importAGMetadata` to import the updated activity graph metadata file to the managed server where the Portal Framework application is deployed. For more information, see the "Importing Activity Graph Metadata" section in *Administering 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 "Exporting Activity Graph Metadata" section in *Administering 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.
Extending the Activity Graph

Integrating the Activity Graph

- **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**

```xml
<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 Portal Framework application is deployed. For more information, see the “Importing Activity Graph Metadata” section in *Administering 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 "Exporting Activity Graph Metadata" section in *Administering 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**

```xml
<QRPPs>
</QRPPs>
```
3. Use the WLST command `importAGMetadata` to import the updated activity graph metadata file to the managed server where the Portal Framework application is deployed. For more information, see the "Importing Activity Graph Metadata" section in *Administering Oracle WebCenter Portal*.

### 46.5 Troubleshooting the Activity Graph

This section provides information to assist you in troubleshooting problems you may encounter while using the activity graph.

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 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 "Validating Analytic Event Collection" section in *Administering 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 and the tools/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 WebCenter Portal and services should be wired to the same OID.

46.5.3 Troubleshooting the Similar Portals Task Flow

Problem
A portal is not suggested in the Similar Portals task flow even after a successful user interaction.

Solution
Check to see that the logged in user has permissions to access the portal by finding it in the Browse Portals 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 tool or service. If the tool is unavailable, then items from that tool 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 and the tools/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 and services should be wired to the same OID.
Analytics enable you to display usage and performance metrics for Portal Framework applications. This chapter describes how to set up and integrate analytics in Framework applications.

Note: For portals, connections to the Analytics database (ACTIVITIES) and the Analytics Collector are configured out-of-the-box so the Analytics service is available by default.

This chapter includes the following topics:

- Section 47.1, "Introduction to Analytics"
- Section 47.2, "Basic Configuration for Analytics"
- Section 47.3, "Building Analytics Reports"

For information about using any of the analytics task flows, see the "Adding Analytics to a Portal" chapter in Oracle Fusion Middleware Building Portals with Oracle WebCenter Portal. For a detailed list of database tables and parameters, see Appendix H, "WebCenter Portal Analytics Database Schema."

47.1 Introduction to Analytics

Analytics offers real-time usage and activity reporting for your portal or Framework application. This section provides an overview of analytics, its associated task flows, and how it can be used.

This section includes the following subsections:

- Section 47.1.1, "Understanding Analytics"
- Section 47.1.2, "Requirements for the Analytics Service"
- Section 47.1.3, "What Happens at Runtime"

47.1.1 Understanding Analytics

Analytics lets administrators and users track and analyze traffic and usage in portal and Framework applications. Analytics provides the following basic functionality:

- **Usage Tracking Metrics**: Analytics collects and reports metrics for common portal functions, including community, page, portlet, and document hits.
• **Behavior Tracking**: Analytics can be used to analyze portal and Framework application metrics to determine usage patterns, such as page visit duration and usage over time.

• **User Profile Correlation**: Analytics 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 events that WebCenter Portal collects for analytics.

<table>
<thead>
<tr>
<th>Component</th>
<th>Events</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pages</td>
<td>User creates, deletes, edits, tags, or views a page</td>
</tr>
</tbody>
</table>
| Discussions | User views, creates, edits, deletes, tags, likes, or replies to a discussion  
             | User views, creates, edits, or deletes an announcement  
             | User creates or deletes a discussion forum |
| Wikis/Blogs | User views a wiki or blog, downloads wiki or blog content, creates, edits, or deletes a wiki or blog, tags, likes or comments on a wiki or blog |
| Portal    | User views a portal, creates a portal, joins a portal, or deletes a portal |
| Lists     | User creates, deletes, or edits a list |
| Login     | User logs in |
| People    | User adds a connection, posts on a wall, edits their profile, or edits their status information |
| Portlet   | User views a portlet |
| Search    | User initiates a search |

Table 47–2 lists the analytics task flows available out-of-the-box with WebCenter Portal. These task flows work similarly for portals and Portal Framework applications. For detailed information about these task flows and how to use them, see the “Adding Analytics to a Portal” chapter in Oracle Fusion Middleware Building Portals with Oracle WebCenter Portal.

<table>
<thead>
<tr>
<th>Task Flows</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WebCenter Portal Traffic</td>
<td>A summarized view for common events within the portal.</td>
</tr>
<tr>
<td>Page Traffic</td>
<td>Displays the number of page hits and the number of unique users that visited any page within the portal.</td>
</tr>
<tr>
<td>Login Metrics</td>
<td>Reports portal logins.</td>
</tr>
<tr>
<td>Portlet Traffic</td>
<td>Displays usage data for a portlet.</td>
</tr>
<tr>
<td>Portlet Response Time</td>
<td>Displays performance data for a portlet.</td>
</tr>
<tr>
<td>Portlet Instance Traffic</td>
<td>Displays usage data for a portlet instance*.</td>
</tr>
<tr>
<td>Portlet Instance Response Time</td>
<td>Displays performance data for a portlet instance*.</td>
</tr>
</tbody>
</table>
* If the same portlet is displayed on several different pages, each placement is known as a portlet instance.

As well as events reported by WebCenter Portal’s out-of-the-box task flows, you can also create custom task flows based on a SQL query to report on additional events as described in Section 47.3, "Building Analytics Reports."

### 47.1.2 Requirements for the Analytics Service

The Analytics service requires that the analytics schema (`ACTIVITIES`) is configured and running on an appropriate database. In addition, Oracle WebCenter Portal’s Analytics Collector must be up and running on the `WC_Utilities` managed server. For detailed installation instructions, see *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, you may need to configure different values using WLST or Fusion Middleware Control. For more details, see the "Configuring Analytics Collector Settings" chapter in *Administering Oracle WebCenter Portal*.

To display usage and performance metrics through analytics task flows you must enable event generation for your Portal 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 Analytics."

### 47.1.3 What Happens at Runtime

At runtime, user activity in your Portal Framework application generates event data. For example, every time a user logs 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 Portal Framework application is configured with the location of the collector, and all events are transmitted to that location.

---

**Note:** In the current release, the Analytics Collector cannot be clustered.

Analytics task flows and custom analytics reports (based on SQL data controls) that are included in Portal 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 the "Adding Analytics to a Portal" in *Oracle Fusion Middleware Building Portals with Oracle WebCenter Portal*. 

---

### Table 47–2  (Cont.) Analytics Task Flows Available in JDeveloper

<table>
<thead>
<tr>
<th>Task Flows</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search Metrics</td>
<td>Tracks portal searches.</td>
</tr>
<tr>
<td>Document Metrics</td>
<td>Tracks document views.</td>
</tr>
<tr>
<td>Wiki Metrics</td>
<td>Tracks most popular/least popular wikis.</td>
</tr>
<tr>
<td>Blog Metrics</td>
<td>Tracks most popular/least popular blogs.</td>
</tr>
<tr>
<td>Discussion Metrics</td>
<td>Tracks most popular/least popular discussions.</td>
</tr>
</tbody>
</table>
47.2 Basic Configuration for Analytics

This section describes how to include analytics in your Portal Framework applications. This section contains the following subsections:

- Section 47.2.1, "Configuration Roadmap for Analytics"
- Section 47.2.2, "Setting up Connections for Analytics"
- 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 Analytics

The flow chart depicted in Figure 47–1, together with Table 47–3, provides an overview of the prerequisites and tasks required to get analytics working in Portal Framework applications. For WebCenter Portal, the steps are essentially the same, with the exception that connections are already configured for out-of-the-box task flows. Each of the subsequent sections identify which configurations apply only to Framework applications.
Basic Configuration for Analytics

Integrating Analytics

Figure 47–1 Configuring the Analytics for Framework Applications

Table 47–3 Configuring Analytics for Framework Applications

<table>
<thead>
<tr>
<th>Actor</th>
<th>Task</th>
<th>Sub-task</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrator</td>
<td>1. Install Oracle WebCenter Portal Framework and the back-end components for Analytics</td>
<td></td>
</tr>
<tr>
<td>Developer</td>
<td>2. Integrate the Analytics service in your Portal Framework application</td>
<td>2.a Configure a connection to the Activities database in JDeveloper</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.b Configure a connection to the Analytics Collector in JDeveloper</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.c Add an analytics task flow to a page in JDeveloper</td>
</tr>
</tbody>
</table>

Integrating Analytics

47-5
### 47.2.2 Setting up Connections for Analytics

Analytics require that the application is connected to the analytics database (ACTIVITIES). You must also configure a Portal Framework application to send event information to a specific Analytics Collector.

#### Table 47–3 (Cont.) Configuring Analytics for Framework Applications

<table>
<thead>
<tr>
<th>Actor</th>
<th>Task</th>
<th>Sub-task</th>
</tr>
</thead>
</table>
| Developer/Administrator       | 3. Deploy the Portal 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 Portal Framework application 6.a Log in to the Portal Framework application 6.b Display the analytics task flow |                                          |

#### 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 the "Managing the Analytics Service" chapter in *Administering 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

Analytics require 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](image)

This creates a data-source to the ACTIVITIES schema so that you can test analytics in your Portal 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:
Basic Configuration for Analytics

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 the "Configuring Analytics Collector Settings" chapter in Administering Oracle WebCenter Portal.

### 47.2.2.3 How to Set Up a Connection to the Analytics Collector

(Framework applications only) Analytics require a connection to the Analytics Collector that is collecting WebCenter Portal events and OpenUsage must be enabled in the Portal 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). Currently, clusters are not supported so specify only `true` here.
- **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 Portal Framework application. If the Integrated WebLogic Server is currently running you must restart it to pick up the new settings. See also, Section 7.2, "Deploying a Portal 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. The 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:

```xml
<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:

```java
public void sendPageEvent (ClientEvent event) {
    // Checks whether Analytics events must be sent
    if (!AnalyticsUtil.isSendingEvents())
```
Basic Configuration for Analytics

```java
return;

HttpServletRequest request = (HttpServletRequest)
FacesContext.getCurrentInstance().getExternalContext().getRequest();
String requestUser = 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(),
'PortalApp', 'PortalApp', requestUser, requestUser,
0, pageName, false, request);
```

### 47.2.4 Configuring a Namespace for Analytics Customizations in MDS

(Framework applications only) 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 Portal Framework application, you must configure a namespace for the analytics metadata in `adf-config.xml`.

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:

   ```xml
   <namespace path="/oracle/webcenter/analytics/scopedMD"
   metadata-store-usage="WebCenterFileMetadataStore"/>
   ```
3. Save the file.

See also, Chapter 21, "Performing Composer-Specific MDS Configurations."

### 47.2.5 Adding Analytics Task Flows at Design Time

This section describes the analytics task flows and how to add them 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"
47.2.5.1 Analytics Task Flows

Table 47–4 lists the analytics task flows available in JDeveloper. For detailed information about these task flows and the type of information that users can see at runtime, see the "Understanding Analytics Task Flows in WebCenter Portal" section in Oracle Fusion Middleware Building Portals with Oracle WebCenter Portal.

<table>
<thead>
<tr>
<th>Analytics Task Flows</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WebCenter Portal Traffic</td>
<td>A summarized view for common events within the portal.</td>
</tr>
<tr>
<td>Page Traffic</td>
<td>Displays the number of page hits and the number of unique users that visited any page within the portal.</td>
</tr>
<tr>
<td>Login Metrics</td>
<td>Reports portal logins.</td>
</tr>
<tr>
<td>Portlet Traffic</td>
<td>Displays usage data for a portlet.</td>
</tr>
<tr>
<td>Portlet Response Time</td>
<td>Displays performance data for a portlet.</td>
</tr>
<tr>
<td>Portlet Instance Traffic</td>
<td>Displays usage data for a portlet instance*.</td>
</tr>
<tr>
<td>Portlet Instance Response Time</td>
<td>Displays performance data for a portlet instance*.</td>
</tr>
<tr>
<td>Search Metrics</td>
<td>Tracks portal searches.</td>
</tr>
<tr>
<td>Document Metrics</td>
<td>Tracks document views.</td>
</tr>
<tr>
<td>Wiki Metrics</td>
<td>Tracks most popular/least popular wikis.</td>
</tr>
<tr>
<td>Blog Metrics</td>
<td>Tracks most popular/least popular blogs.</td>
</tr>
<tr>
<td>Discussion Metrics</td>
<td>Tracks most popular/least popular discussions.</td>
</tr>
</tbody>
</table>

* If the same portlet is displayed on several different pages, each placement is known as a portlet instance.

47.2.5.2 How to Add Analytics Task Flows to a Page

To add an analytics task flow to your 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).
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).

7. In the **Edit Task Flow Binding** dialog, enter values for analytics task flow parameters:

   - ```applicationName``` - Required
Basic Configuration for Analytics

Integrating Analytics

- 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 Portal 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 the "Setting Analytics Task Flow Properties" section in Oracle Fusion Middleware Building Portals with Oracle WebCenter Portal

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).
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."

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_<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 47–5 lists and describes task flow binding parameters applicable to the Analytics service.

<table>
<thead>
<tr>
<th>Parameter (* = required)</th>
<th>Task Flows</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>analyticsReportTitle</td>
<td>All</td>
<td>Specifies the display title that appears above the analytics data, that is, you can override the default report title. See also, Figure 47–7, &quot;analyticsReportTitle - Example&quot;.</td>
</tr>
<tr>
<td>analyticsResourceId*</td>
<td>All</td>
<td>Specifies the MDS document that will be generated to store user customizations/application customizations for the task flow instance in MDS. For example: mymainloginmetrics</td>
</tr>
<tr>
<td>applicationName*</td>
<td>All</td>
<td>Specifies the name of the Portal Framework application for which you want to display analytics data. For example: MyPortalApplication For portals, the application name is always webcenter. 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.</td>
</tr>
<tr>
<td>maxDataPointsPerSeries</td>
<td>All</td>
<td>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, &quot;maxDataPointsPerSeries - Example&quot;.</td>
</tr>
</tbody>
</table>
### 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 H, "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 Fusion Developer’s Guide for Oracle Application Development Framework.

You can also create data controls in a running Portal Framework application if runtime resource management features are enabled. To do this in WebCenter Portal, see the
"Creating a SQL Data Control" section in *Oracle Fusion Middleware Building Portals with Oracle WebCenter Portal*. To create a data control in other Portal Framework applications, see the "Managing Assets for a Portal Framework Application" chapter in *Administering Oracle WebCenter Portal*.

Use a SQL data control to fetch data from the analytics database and display analytics data in your Portal Framework application. Figure 47–9 shows a sample SQL statement in a Create/Edit SQL Data Control dialog in webCenter Portal. A database connection to the analytics database (*ActivitiesDS*) is required, and you must provide a valid database password.

**Sample SQL: Page Hits by Day**

```sql
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 WebCenter Portal**

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 WebCenter Portal application, the name of the application is set to `webcenter` and personal pages (pages in the Home portal) are
Building Analytics Reports

Included in the report shown in Figure 47–10.

**Figure 47–10  Setting Report Parameters**

![Enter bind parameter values](image)

**Table 47–6  Report Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>appname</td>
<td>webcenter</td>
<td>The corresponding application name. For portals, this will always be webcenter. For Framework applications, the value is unique to the application.</td>
</tr>
<tr>
<td>ispersonal</td>
<td>0 or 1</td>
<td>For portals, determines whether or not pages created in the Home portal are included in the report.</td>
</tr>
<tr>
<td>dateformat</td>
<td>mm/dd/yyyy hh24</td>
<td>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.</td>
</tr>
<tr>
<td>startdate</td>
<td>01/01/2010</td>
<td>Determines a start date for the data required.</td>
</tr>
<tr>
<td>enddate</td>
<td>01/01/2011</td>
<td>Determines an end date for the data required.</td>
</tr>
</tbody>
</table>

Figure 47–9 shows the custom report displayed as a graph.

**Figure 47–11  Sample Analytics Report - Page Hits by Day**

![Graph of Page Hits by Day](image)
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: Portal Traffic
- Analytics SQL: Portal 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 H, "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: Portal 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: Portal 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_instance_ is not null
and portlet.resourceid_ is not null
GROUP BY portlet.default_title_, portlet.id, fact.portlet_instance_, 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_instance_, 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_instance_ is not null
    and portlet.resourceid_ is not null
GROUP BY portlet.default_title_, portlet.id, fact.portlet_instance_, 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
```
47.3.2.10 Analytics SQL: Search Metrics

```sql
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

```sql
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

```sql
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

```sql
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
```
47.3.2.14 Analytics SQL: Discussion Metrics

```sql
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
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, see Section H.5, "Analytics User Properties" in Appendix H, "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.

```sql
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.

```sql
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'
```
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: Portal Activities
- Sample SQL: Portal 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 H, "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.

```sql
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: Portal Activities
The following query returns a list of all activities executed in a specific portal.

```sql
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

<table>
<thead>
<tr>
<th>Space</th>
<th>Activity</th>
<th>Events</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project1</td>
<td>PageViews</td>
<td>34</td>
</tr>
<tr>
<td>Project2</td>
<td>PageViews</td>
<td>42</td>
</tr>
<tr>
<td>Project1</td>
<td>PortletViews</td>
<td>98</td>
</tr>
<tr>
<td>Project2</td>
<td>PortletViews</td>
<td>74</td>
</tr>
</tbody>
</table>
47.3.4.3 Sample SQL: Portal Activities for a Specific Time Period

The following query returns a list of all activities executed in a specific portal during a specific time period.

```sql
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**

<table>
<thead>
<tr>
<th>Activity</th>
<th>Events</th>
</tr>
</thead>
<tbody>
<tr>
<td>Page views</td>
<td>15</td>
</tr>
<tr>
<td>Portlet views</td>
<td>200</td>
</tr>
</tbody>
</table>

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.

```sql
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
```

---

47-28  Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper
Building Analytics Reports

Integrating Analytics

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.

```sql
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

<table>
<thead>
<tr>
<th>Search Phrase</th>
<th>Times</th>
</tr>
</thead>
<tbody>
<tr>
<td>product prices</td>
<td>4</td>
</tr>
<tr>
<td>sales report 2011</td>
<td>1</td>
</tr>
</tbody>
</table>

47.3.4.6 Sample SQL: Page Views

The following query returns the pages viewed by a specific user during a specific time period.

```sql
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
```

Sample Report Output

<table>
<thead>
<tr>
<th>User</th>
<th>Views</th>
</tr>
</thead>
<tbody>
<tr>
<td>user3</td>
<td>245</td>
</tr>
<tr>
<td>user1</td>
<td>240</td>
</tr>
<tr>
<td>user2</td>
<td>193</td>
</tr>
</tbody>
</table>

GROUP BY u.userid
ORDER BY count(*) DESC
Building Analytics Reports

```
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**

<table>
<thead>
<tr>
<th>User</th>
<th>Views</th>
</tr>
</thead>
<tbody>
<tr>
<td>user3</td>
<td>245</td>
</tr>
<tr>
<td>user1</td>
<td>190</td>
</tr>
<tr>
<td>user2</td>
<td>65</td>
</tr>
</tbody>
</table>
Part VIII contains the following chapters:

- Chapter 48, "Integrating Events"
- Chapter 49, "Integrating Lists"
- Chapter 50, "Integrating Notifications"
- Chapter 51, "Integrating Recent Activities"
- Chapter 52, "Integrating RSS"
This chapter explains how to integrate events functionality into a WebCenter Framework application at design time.

This chapter includes the following sections:

- Section 48.1, "Introduction to Events"
- Section 48.2, "Basic Configuration for Events"
- Section 48.3, "Using the Events REST API"
- Section 48.4, "Troubleshooting Events"

For information about managing and including events, see:

- the "Managing Calendar Events" chapter in *Administering Oracle WebCenter Portal*.
- the "Working with Events" chapter in *Using Oracle WebCenter Portal*.

### 48.1 Introduction to Events

In Portal Framework applications, events provide access to your personal Microsoft Exchange calendar.

This section includes the following subsections:

- Section 48.1.1, "Understanding Events"
- Section 48.1.2, "Requirements for Events"
- Section 48.1.3, "What Happens at Runtime"

#### 48.1.1 Understanding Events

With events, users can:

- Display their Microsoft Exchange calendars in the 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.

Events is integrated with many WebCenter Portal tools and components, such as links, mail, and search.

*Figure 48–1* shows an example of a personal calendar in a Framework application.
48.1.2 Requirements for Events

Events 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 Prerequisites for Personal Events" in the Administering Oracle WebCenter Portal.

48.1.3 What Happens at Runtime

At runtime, users can view events from their Microsoft Exchange calendars.

For more information about events at runtime, see the chapter "Working with Events" in the Using Oracle WebCenter Portal.

---

Note: Using Oracle WebCenter Portal discusses both personal events and portal events. Only the information on personal events applies to Portal Framework applications.

48.2 Basic Configuration for Events

This section includes the following subsections:

- Section 48.2.1, "Configuration Roadmap for Events"
- Section 48.2.2, "Setting Up a Connection for Events"
- Section 48.2.3, "Adding Events Functionality at Design Time"
- Section 48.2.4, "Setting Security for Events"
48.2.1 Configuration Roadmap for Events

The flow chart (Figure 48–2) and table (Table 48–1) in this section provide an overview of the prerequisites and tasks required to get events working in Framework applications.

Figure 48–2 Configuring Events for Portal Framework Applications
### 48.2.2 Setting Up a Connection for Events

To take advantage of events functionality, 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 events, but only one connection is active at a time.

---

#### Table 48–1 Configuring Events for Portal Framework Applications

<table>
<thead>
<tr>
<th>Actor</th>
<th>Task</th>
<th>Sub-task</th>
<th>Notes</th>
</tr>
</thead>
</table>
| Administrator                | 1. Install WebCenter Portal and MS Exchange Server | 1.a Configure MS Exchange Server 2007  
1.b Edit security settings  
1.c (Optional) Enable SSL | MS Exchange Server is the back-end server for personal calendars          |
|                              | ■ Use MS Exchange Server 2007             |                                                                          |                                                 |
|                              | ■ 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 events 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 |

---

**Table 48–1** Configuring Events for Portal Framework Applications
To set up a connection for events:

1. In JDeveloper, open the application in which you plan to consume events.

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 48–3).

   ![Create Events Connection Dialog](image)

   **Figure 48–3 Create Events Connection Dialog**

   - **Create Connection In:** Application Resources
   - **Connection Name:** ExchangeServerForEvents
   - **Web Service URL:** http://abcde111.example.com/EWS/Services.wsdl
   - **Service Adapter:** WS Exchange 2007

4. In the **Web Service URL** field, enter the URL of the web service exposing the event application. Use the format:

   ```txt
   protocol://host:port/appWebServiceInterface/WSName
   ```

   For example:

   ```txt
   http://myexchange.com:80/ExchangeWS/PersonalEventsWebService.asmx
   http://myexchange.com:80/EWS/Services.wadl
   ```

5. From the **Service Adapter** dropdown list, select the version of the Microsoft Exchange server to connect to:
6. Select **Default Connection** to use this connection for events. 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 48–4). An external application connection is also created to store the user names and passwords of users who connect to the Exchange server.

![Application Resources - Personal Event Connection](image)

### 48.2.3 Adding Events Functionality at Design Time

This section includes the following subsections:

- **Section 48.2.3.1, “Events Task Flows”**
- **Section 48.2.3.2, “How to Add Events Functionality to Your Application”**

#### 48.2.3.1 Events Task Flows

Events task flows enable you to add a main view or quick view of events to a page.

<table>
<thead>
<tr>
<th>Task Flow</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calendar Main View</td>
<td>This task flow displays a calendar of events and provides users with options to view events in day, week, month, or list view.</td>
</tr>
<tr>
<td>Calendar Mini View</td>
<td>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).</td>
</tr>
</tbody>
</table>
48.2.3.2 How to Add Events Functionality to Your Application

To add an Events task flow to your application:

1. Follow the steps described in Section 4.2, "Preparing Your Framework Application for Tools and 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.


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 48–5 shows the Calendar Main View task flow at runtime.

Figure 48–5 Calendar Main View Task Flow

---

Figure 48–6 shows the Calendar Mini View task flow at runtime.

Figure 48–6 Calendar Mini View Task Flow

---

Figure 48–5 shows the Calendar Main View task flow at runtime.

Figure 48–6 shows the Calendar Mini View task flow at runtime.
48.2.4 Setting Security for Events

When you create the connection to the Microsoft Exchange Server for events, 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 48.2.2, "Setting Up a Connection for Events."

48.3 Using the Events REST API

WebCenter Portal provides a REST API to support events. Use the REST API to create your own interface for providing access to portal events.

---

**Note:** The Events REST API is available for portal events only. You cannot use the REST API to work with personal events.

This section describes the REST API methods associated with events. It includes the following subsections:

- Section 48.3.1, "Events Entry Point"
- Section 48.3.2, "Events Resource Type Taxonomy"
- Section 48.3.3, "Security Considerations"
- Section 48.3.4, "Events Resource Types"

48.3.1 Events Entry Point

The entry point for events can be reached only through a portal. First you need to navigate to the appropriate portal 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 events.

For more information about the Resource Index, see Section 53.5.1, "Using the Resource Index."

For more information about resource types, see Section 53.5.2.1, "Resource Type."

48.3.2 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 48.3.4, "Events Resource Types."

The taxonomy for events is:

```
urn:oracle:webcenter:events:gsEvents
urn:oracle:webcenter:events:gsEvent
urn:oracle:webcenter:events:gsCategories
```
48.3.3 Security Considerations

There are no specific security considerations for events. For general security considerations, see Section 53.8, "Security Considerations for WebCenter Portal REST APIs."

48.3.4 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 48.3.4.1, "urn:oracle:webcenter:events:gsEvents"
- Section 48.3.4.2, "urn:oracle:webcenter:events:gsEvent"
- Section 48.3.4.3, "urn:oracle:webcenter:events:gsCategories"

48.3.4.1 urn:oracle:webcenter:events:gsEvents

Use this resource to identify the URI to use to retrieve (GET) and create (POST) portal events. The response from a GET operation includes each portal 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: `<gsCategory>`, Parameters: startIndex, itemsPerPage, utoken
    
    For information about these common parameters, see "Common Request Query Parameters".
    
    The following additional parameters are available:
    *
    *
    
    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 48–3 lists the resource types that the client can link to from this resource.
Using the Events REST API

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:events:gsEvents</td>
</tr>
</tbody>
</table>

48.3.4.2 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 portal 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

<table>
<thead>
<tr>
<th>resourceIndex</th>
</tr>
</thead>
<tbody>
<tr>
<td>spaces</td>
</tr>
<tr>
<td>gsEvents</td>
</tr>
<tr>
<td>gsEvent</td>
</tr>
</tbody>
</table>

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 48–4 lists the writable elements for this resource.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Required</th>
<th>Constraints</th>
<th>Description</th>
</tr>
</thead>
</table>
| category  | String | No       | The name of an event category defined for the portal | The category to which the event belongs.  
Note: If the category does not exist, the default is used if the event is being created. However, an error will occur if the event is being modified. |
### Read-only Elements for gsEvent

Table 48–5 lists the read-only elements for this resource.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>author</td>
<td>personReference</td>
<td>User who created the event</td>
</tr>
<tr>
<td>created</td>
<td>Date</td>
<td>Date on which the event was created</td>
</tr>
<tr>
<td>duration</td>
<td>String</td>
<td>The length (in minutes) of the event</td>
</tr>
<tr>
<td>groupSpace</td>
<td>groupSpaceReference</td>
<td>The portal to which the event belongs</td>
</tr>
<tr>
<td>id</td>
<td>String</td>
<td>Unique ID of the event</td>
</tr>
<tr>
<td>isAllDayEvent</td>
<td>Boolean</td>
<td>Indicates whether the event takes place over an entire day</td>
</tr>
<tr>
<td>modified</td>
<td>Date</td>
<td>Date on which the event was last updated</td>
</tr>
<tr>
<td>modifiedByUser</td>
<td>personReference</td>
<td>User who last updated the event</td>
</tr>
</tbody>
</table>
Resource Types Linked to from gsEvent

Table 48–6 lists the resource types that the client can link to from this resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:events:gsEvent</td>
</tr>
</tbody>
</table>

48.3.4.3 urn:oracle:webcenter:events:gsCategories

Use this resource to identify the URI to use to retrieve (GET) and create (POST) portal 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: category
  - response - body: 0 or more categories
- **POST**
  - request - body: category
  - response - body: category

Resource Types Linked to From gsCategories

Table 48–7 lists the resource types that the client can link to from this resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:events:gsCategories</td>
</tr>
</tbody>
</table>

48.4 Troubleshooting Events

This section provides information to assist you in troubleshooting problems you may encounter while using events.

**Problem**

Error message displayed in calendar taskflow:

*Personal Events is not configured properly. Rectify the configuration and then click Try Again.*
**Solution**
Confirm that you have created a Personal Event Connection. For more information, see Section 48.2.2, "Setting Up a Connection for Events."

**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.
This chapter provides an overview of lists and describes how to integrate lists functionality in your WebCenter Portal application.

- Section 49.1, "Introduction to Lists"
- Section 49.2, "Basic Configuration for Lists"
- Section 49.3, "Adding Lists to a Resource Catalog"
- Section 49.4, "Using the Lists REST API"
- Section 49.5, "Troubleshooting Lists"

### 49.1 Introduction to Lists

This section provides an overview of list features and requirements. It includes the following subsections:

- Section 49.1.1, "Understanding Lists"
- Section 49.1.2, "What Happens to Lists at Runtime"

#### 49.1.1 Understanding Lists

WebCenter Portal users can create lists and expose them for placement on portal pages at runtime. At design time, you can make the Lists task flow available in your runtime Resource Catalog. Portal administrators 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 portal pages.

By default, only users assigned the seeded role Administrator have manage permissions 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 Fusion Middleware Control or the Application Policy Manager. Authorized users can change this permission at runtime at the component level (for more information, see Chapter 22, "Modifying Default Security Behavior of Composer Components"). Only role-based permission is supported.

#### 49.1.2 What Happens to Lists 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 Using Oracle WebCenter Portal.
Tip: To enable users to work with lists at runtime, they must be explicitly permitted to do so. For more information, see Section 49.2.3, "Enabling Users to Work with Lists at Runtime."

49.2 Basic Configuration for Lists

This section describes the steps required for adding lists functionality to your portal. It includes the following subsections:

- Section 49.2.1, "Setting Up Connections for Lists"
- Section 49.2.2, "Adding Lists Functionality at Design Time"
- Section 49.2.3, "Enabling Users to Work with Lists at Runtime"

49.2.1 Setting Up Connections for Lists

Lists requires a connection to the database where the WEBCENTER schema is installed. All list data is stored in the database. For details about setting up a database connection to the database where the WEBCENTER schema is installed, see Section 4.2.2, "Setting Up a Database Connection."

Note: For details about installing the database and the WEBCENTER schema, see Installation Guide for Oracle WebCenter Portal.

49.2.2 Adding Lists Functionality at Design Time

This section explains a basic integration of lists functionality. It includes the following subsections:

- Section 49.2.2.1, "Lists Task Flow"
- Section 49.2.2.2, "How to Add Lists Functionality to a Framework Application"

49.2.2.1 Lists Task Flow

Figure 49–1 shows the Lists task flow.

For information about the Lists task flow, see Section 49.1.2, "What Happens to Lists at Runtime."
49.2.2 How to Add Lists Functionality to a Framework Application

To add the Lists task flow to your Framework application:

1. Follow the steps described in Section 4.2, "Preparing Your Framework Application for Tools and Services" to implement security and create a new customizable page in your portal.

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 22, "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.

49.2.3 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 lists functionality to your application, then you must also add the Role Manager task flow. If your application is based on a WebCenter Portal - Framework Application template, 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/appcontextroot/faces/admin

Note: Roles and permissions created and granted through the Role Manager at design time are not packaged in the portal 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 74.4, "Using the Role Manager Task Flow."

49.3 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 14, "Developing Resource Catalogs." This section provides an example of a code snippet to use to add lists to a resource catalog (Example 49–1).
49.4 Using the Lists REST API

Oracle WebCenter Portal provides a REST API to allow access to Lists functionality through interfaces other than the provided task flows. You can use the Lists REST API to perform the following actions:

- Work with lists in a portal, 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 REST API methods associated with lists. It includes the following subsections:

- Section 49.4.1, "Entry Point for Lists"
- Section 49.4.2, "Lists Resource Type Taxonomy"
- Section 49.4.3, "Lists Security Considerations"
- Section 49.4.4, "Lists Resource Types"

**See Also:** For an introduction to the REST APIs, see Chapter 53, "Using Oracle WebCenter Portal REST APIs."

### 49.4.1 Entry Point for Lists

To get to the REST entry point for lists in a portal in WebCenter Portal, you search for the specific portal and retrieve the resource index. The corresponding href or template element provides the URI entry point, which retrieves the lists in the portal.

1. Navigate to the portal or subportal for which to retrieve the lists.
2. Retrieve the resource index.
3. Navigate to the returned href.
4. Search for the resource type. For example, for all the lists in a portal, search for:

\[ \text{urn:oracle:webcenter:space:lists} \]

**See Also:** For more information about the Resource Index, see Section 53.5.1, "Using the Resource Index."

For more information about resource types, see Section 53.5.2.1, "Resource Type."

### 49.4.2 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:

\[ \text{urn:oracle:webcenter:space:lists} \]

\[ \text{urn:oracle:webcenter:space:list} \]

\[ \text{urn:oracle:webcenter:space:list:rows} \]

\[ \text{urn:oracle:webcenter:space:list:row} \]

\[ \text{urn:oracle:webcenter:space:list:columns} \]

\[ \text{urn:oracle:webcenter:space:list:column} \]

### 49.4.3 Lists Security Considerations

You must be logged in to the REST service to access any of the Lists REST API methods. After that, the underlying service handles permission checking.

**See Also:** For general security considerations, see Section 53.8, "Security Considerations for WebCenter Portal REST APIs."

### 49.4.4 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 49.4.4.1, "urn:oracle:webcenter:space:lists"
- Section 49.4.4.2, "urn:oracle:webcenter:space:list"
- Section 49.4.4.3, "urn:oracle:webcenter:space:list:rows"
- Section 49.4.4.4, "urn:oracle:webcenter:space:list:row"
- Section 49.4.4.5, "urn:oracle:webcenter:space:list:columns"
- Section 49.4.4.6, "urn:oracle:webcenter:space:list:column"

#### 49.4.4.1 urn:oracle:webcenter:space:lists

The lists response provides a means of retrieving the lists in a given portal (GET) and adding lists to a portal (POST). This section includes the following subsections:

- Section 49.4.4.1.1, "Navigation Paths to lists"
- Section 49.4.4.1.2, "Supported Methods for lists"
- Section 49.4.4.1.3, "Resource Types Linked to from lists"
49.4.4.1 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

49.4.4.2 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 portal.

- **Resource URI**, for example:

  ```
  Kl8lalI3QRdi8TgQkOCGEzxl5x_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 49–1)

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Name of list</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Description of list</td>
</tr>
<tr>
<td>creator</td>
<td>string</td>
<td>User who created list</td>
</tr>
<tr>
<td>created</td>
<td>date</td>
<td>Date the list was created</td>
</tr>
<tr>
<td>modifier</td>
<td>string</td>
<td>User who modified list</td>
</tr>
<tr>
<td>modified</td>
<td>date</td>
<td>Date the list was last modified</td>
</tr>
</tbody>
</table>

- **Response status**: 200 [OK]
- **Response body**: `<lists>`
- **Example**:

  ```
  <lists resourceType="urn:oracle:webcenter:space:lists">
  <links>
  ```
Method (lists): POST
This method creates a list in a portal.
- Resource URI, for example:
  
  http://host:port/rest/api/spaces/vs1/lists?utoken=FKld8lalI3QRdi8TgQkOCGEzzL5x_ 
  w**

- Request body: <list>
- Writable elements (Table 49–2)

Table 49–2  Create List Writable Elements

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Constraints</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name(^1)</td>
<td>string</td>
<td>—</td>
<td>Name of list</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>—</td>
<td>Description of list</td>
</tr>
<tr>
<td>columns(^1)</td>
<td>urn:oracle:webcenter:space:list:columns</td>
<td>1 to 30</td>
<td>List columns</td>
</tr>
</tbody>
</table>

\(^1\) Denotes required element

- Example:
Using the Lists REST API

```xml
<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>
  
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>string</td>
<td>List ID</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>List name</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>List description</td>
</tr>
<tr>
<td>scope.guid</td>
<td>string</td>
<td>GUID of portal to which the list belongs</td>
</tr>
<tr>
<td>scope.name</td>
<td>string</td>
<td>Name of the portal to which the list belongs</td>
</tr>
<tr>
<td>creator</td>
<td>string</td>
<td>Name of the user who created the list</td>
</tr>
<tr>
<td>created</td>
<td>date</td>
<td>Date the list was created</td>
</tr>
<tr>
<td>modifier</td>
<td>string</td>
<td>Name of the user who last modified the list</td>
</tr>
<tr>
<td>modified</td>
<td>date</td>
<td>Date the list was last modified</td>
</tr>
<tr>
<td>columns</td>
<td>urn:oracle:webcenter:space:list:columns</td>
<td>List columns</td>
</tr>
</tbody>
</table>

- Example:

```xml
<list resourceType="urn:oracle:webcenter:space:list">
  
  <link resourceType="urn:oracle:webcenter:space:list" rel="self" href="http://host:port/rest/api/spaces/vsl/lists/\(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/1a_c9cecb7_756b_417b_a35f_fb03874979c0.xml\)?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w*** capabilities="urn:oracle:webcenter:read urn:oracle:webcenter:write urn:oracle:webcenter:delete"/>
  
  <link template="http://host:port/rest/api/spaces/vsl/lists/\(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/1a_c9cecb7_756b_417b_a35f_fb03874979c0.xml\)/rows?"/>
</list>
```
Integrating Lists

```
<link resourceType="urn:oracle:webcenter:space:list:rows" href="http://host:port/rest/api/spaces/vs1/lists/(oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9ceecbc7_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://host:port/rest/api/spaces/vs1/lists/(oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9ceecbc7_756b_417b_a35f_fb03874979c0.xml)/columns?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w*** capabilities="urn:oracle:webcenter:read urn:oracle:webcenter:create"/>
```

</link>

</id>/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9ceecbc7_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>
  <link resourceType="urn:oracle:webcenter:people:person" rel="via" href="http://host:port/rest/api/people/E9A35E80F0BC11DFBFBE0FE339E55B21/lists/@self?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w*** capabilities="urn:oracle:webcenter:read"/>

</author>

<displayName>weblogic</displayName>

<guid>E9A35E80F0BC11DFBFBE0FE339E55B21</guid>

</id>/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9ceecbc7_756b_417b_a35f_fb03874979c0.xml</id>

<author>
  <displayName>weblogic</displayName>
</author>

<created>2010-11-18T06:10:32.250-08:00</created>

<modifier>weblogic</modifier>

<modifiedByUser>
  <displayName>weblogic</displayName>
</modifiedByUser>
Using the Lists REST API

```
<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://host:port/rest/api/spaces
            /vs1/lists/\{oracle/webcenter/list/scopedMD/
            s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cecbcc7_756b_417b_a35f_f
            fb03874979c0.xml\}/columns?utoken=
            FKld8alI3Q8d87gQkOCGEzxL5x_w***
            capabilities="urn:oracle:webcenter:read
            urn:oracle:webcenter:create"/>
        <link resourceType="urn:oracle:webcenter:space:list" rel="urn:oracle:webcenter:parent"
            href="http://host:port/
            rest/api/spaces/vs1/lists/\{oracle/webcenter/list/scopedMD/
            s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/
            lists/ls_c9cecbcc7_756b_417b_a35f_f
            fb03874979c0.xml\}?utoken=FKld8alI3Q8d87gQkOCGEzxL5x_w***"/>
    </links>
    <items>
        <metaColumn resourceType="urn:oracle:webcenter:space:list:column">
            <links>
                <link resourceType="urn:oracle:webcenter:space:list:column" rel="self"
                    href="http://host:port/
                    rest/api/spaces/vs1/lists/\{oracle/webcenter/list/scopedMD/
                    s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/ls_c9cecbcc7_756b_417b_a35f_f
                    fb03874979c0.xml\}/columns/(lco_9bdd1418_6004_40ba_a052_04e8335b7ee8)?utoken=
                    FKld8alI3Q8d87gQkOCGEzxL5x_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>
    </items>
</columns>
```
Using the Lists REST API

49.4.4.1.3 Resource Types Linked to from lists

Table 49–4 lists the resource types that the client can link to from the lists resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:space:lists</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:space:list</td>
</tr>
</tbody>
</table>

49.4.4.2 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 49.4.4.2.1, "Navigation Paths to list"
- Section 49.4.4.2.2, "Supported Methods for list"
- Section 49.4.4.2.3, "Resource Types Linked to from list"

49.4.4.2.1 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

49.4.4.2.2 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.
Using the Lists REST API

- Resource URI, for example:
  http://host:port/rest/api/spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_bb806794_0652_4d49_9354_5fb62dc3514d.xml)?utoken=FKld81a113QRdi8TgQkOCGEzL5x_w**

- Request body: none
- Request headers: [Accept = application/xml | application/json]
- Response status: 200 [OK]
- Response body: <list>
- Retrieved elements (Table 49–5)

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>string</td>
<td>ID of the list</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Name of the list</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Description of the list</td>
</tr>
<tr>
<td>scope.guid</td>
<td>string</td>
<td>GUID of the portal to which the list belongs</td>
</tr>
<tr>
<td>scope.name</td>
<td>string</td>
<td>Name of the portal to which the list belongs</td>
</tr>
<tr>
<td>creator</td>
<td>string</td>
<td>User who created the list</td>
</tr>
<tr>
<td>created</td>
<td>Date</td>
<td>Date on which the list was created</td>
</tr>
<tr>
<td>modifier</td>
<td>string</td>
<td>User who last modified the list</td>
</tr>
<tr>
<td>modified</td>
<td>Date</td>
<td>Date on which the list was last modified</td>
</tr>
<tr>
<td>columns</td>
<td>urn:oracle:webcenter:space:list:columns</td>
<td>The columns that make up the list</td>
</tr>
</tbody>
</table>

- For example:
  <list resourceType="urn:oracle:webcenter:space:list">
    <links>
      <link resourceType="urn:oracle:webcenter:space:list" rel="self"
        href="http://host:port/rest/api/spaces/vs1/
        lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_
        f7dac2a7ceed/lists/ls_c9cecb7_756b_417b_a35f_
        fb03874979c0.xml)?utoken=FKld81a113QRdi8TgQkOCGEzL5x_w**
        capabilities="urn:oracle:webcenter:read
        urn:oracle:webcenter:update
        urn:oracle:webcenter:delete"/>
      <link template="http://host:port
        /rest/api/spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/
        s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cecb7_756b_
        417b_a35f_fb03874979c0.xml)/rows?
        utoken=FKld81a113QRdi8TgQkOCGEzL5x_w**
        &startIndex={startIndex}&itemsPerPage={itemsPerPage}&q=
        {searchTerms}&utoken=FKld81a113QRdi8TgQkOCGEzL5x_w**
        resourceType="urn:oracle:webcenter:space:list:rows"
        href="http://host:port/rest/api
        /spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/s355923f0_
        2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cecb7_756b_417b_a35f_
        fb03874979c0.xml)/rows?utoken=FKld81a113QRdi8TgQkOCGEzL5x_w**
        capabilities="urn:oracle:webcenter:read
        urn:oracle:webcenter:create"/>
    </links>
  </list>
Using the Lists REST API

Integrating Lists

```xml
<lists>
  <list>
    <url>
      http://host:port/rest/api/spaces/vs1/lists/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7cecd/lists/ls_c9cebc7_756b_417b_a35f_fb0387e979c0.xml?utoken=FKld81al13Qd18Tyq0CGExzL5x_w*** capabilities="urn:oracle:webcenter:read urn:oracle:webcenter:create"/>
  </url>
  <name>RestExample</name>
  <description>Created using rest api</description>
  <scope>
    <guid>s355923f0_2f04_4fd0_83ad_f7dac2a7cecd</guid>
    <name>vs1</name>
  </scope>
  <creator>weblogic</creator>
  <author>
  </author>
  <created>2010-11-18T06:10:32.250-08:00</created>
  <modifier>weblogic</modifier>
  <modifiedByUser>
    <displayName>weblogic</displayName>
    <guid>E9A35E80F0BC11DFBFBFBD6E339E55B21</guid>
  </modifiedByUser>
  <modified>2010-11-18T06:10:32.250-08:00</modified>
  <columns resourceType="urn:oracle:webcenter:space:list:columns">
    <link resourceType="urn:oracle:webcenter:space:list:columns" rel="self" href="http://host:port/rest/api/spaces/vs1/lists/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7cecd/lists/ls_c9cebc7_756b_417b_a35f_fb0387e979c0.xml?utoken=FKld81al13Qd18Tyq0CGExzL5x_w*** capabilities="urn:oracle:webcenter:read urn:oracle:webcenter:create"/>
  </columns>
</list>
</lists>
```
Using the Lists REST API

```xml
<items>
  <metaColumn resourceType="urn:oracle:webcenter:space:list:column">
    <links>
      <link resourceType="urn:oracle:webcenter:space:list:column" rel="self" href="http://host:port/rest/api/spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cecbce7_756b_417b_a35f_fb03874979c0.xml)/columns/(lco_9bdd1418_6004_40ba_a052_04e8335b7e8e)?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w***" capabilities="urn:oracle:webcenter:read urn:oracle:webcenter:update urn:oracle:webcenter:delete"/>
      <link resourceType="urn:oracle:webcenter:space:list:column" rel="urn:oracle:webcenter:parent" href="http://host:port/rest/api/spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cecbce7_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_9bdd1418_6004_40ba_a052_04e8335b7e8e</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://host:port/rest/api/spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cecbce7_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"/>
      <link resourceType="urn:oracle:webcenter:space:list:column" rel="urn:oracle:webcenter:parent" href="http://host:port/rest/api/spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cecbce7_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>
  </metaColumn>
</items>
```

49-14 Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper
Method (list): PUT
Use this method to update a list name or description.

- Resource URI, for example:
  
  ```
  http://host:port/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: `<list>`

<table>
<thead>
<tr>
<th>Writable elements for list (Table 49–6)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Element</strong></td>
</tr>
<tr>
<td>name</td>
</tr>
<tr>
<td>description</td>
</tr>
</tbody>
</table>

  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://host:port/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

49.4.4.2.3 Resource Types Linked to from list

Table 49–7 lists the resource types that the client can link to from the list resource.
49.4.4.3 Using the Lists REST API

The rows response provides a means of retrieving and adding rows to a list. This section includes the following subsections:

- Section 49.4.4.3.1, "Navigation Paths to rows"
- Section 49.4.4.3.2, "Supported Methods for rows"
- Section 49.4.4.3.3, "Resource Types Linked to from rows"

49.4.4.3.1 Navigation Paths to rows

This section shows how the client can navigate through the hypermedia to access the rows resource:

```plaintext
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
```

49.4.4.3.2 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:

  ```plaintext
  http://host:port/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:
    ```plaintext
    '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

---

**Table 49–7 Resource Types Linked to from list**

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:space:list</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:space:list:rows</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:space:list:columns</td>
</tr>
</tbody>
</table>
Using the Lists REST API

Integrating Lists

- 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:

```xml
<rows resourceType="urn:oracle:webcenter:space:list:rows">
  <link template="http://host:port/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***&capabilities="urn:oracle:webcenter:read urn:oracle:webcenter:update urn:oracle:webcenter:delete"/>
</rows>
```

<item resourceType="urn:oracle:webcenter:space:list:row">
  <link resourceType="urn:oracle:webcenter:space:list:row" rel="self" href="http://host:port/rest/api/spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cecbc7_756b_417b_a35f_fb03874979c0.xml)/(lr_a14d427f_8515_4c82_956f_a60e6e08668c)?utoken=FKld8lalI3QRdi8TgQkOCGEzxl5x_w***&capabilities="urn:oracle:webcenter:read urn:oracle:webcenter:update urn:oracle:webcenter:delete"/>
</item>
```

```xml
<listId>urn:oracle:webcenter:space:list</listId>
<scope>s355923f0_2f04_4fd0_83ad_f7dac2a7ceed</scope>
<creator>weblogic</creator>
```
```xml
<id>lr_a14d427f_8515_4c82_956f_a60e6e08668c</id>
```

```xml
<creator>weblogic</creator>
```
Using the Lists REST API
Method (rows): POST
Use this method to create a row in a list.

- Resource URI, for example:
  http://host:port/rest/api/spaces/vs1/lists/\{oracle/webcenter/list/scopedMD/s35923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_bb806754_0652_4d49_9354_5fb62dc3514d.xml\}/rows?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5w**

- Request body: <row>

- Writable elements (Table 49–8)

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Constraints</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>columns.column.id¹</td>
<td>string</td>
<td></td>
<td>Column ID</td>
</tr>
<tr>
<td>columns.column.value¹</td>
<td>string</td>
<td>Valid value for column data type</td>
<td>Column value</td>
</tr>
</tbody>
</table>

¹ Denotes required element

- For example:
  <row>
  <columns>
    <column>
      <id>lco_9bdd1418_6004_40ba_a052_04e8335b7ee8</id>
      <value>No.</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 49–9)

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>string</td>
<td>Row ID</td>
</tr>
<tr>
<td>list id</td>
<td>string</td>
<td>List ID</td>
</tr>
<tr>
<td>scope</td>
<td>string</td>
<td>GUID of portal to which list belongs</td>
</tr>
<tr>
<td>creator</td>
<td>string</td>
<td>Person who created the list</td>
</tr>
<tr>
<td>created</td>
<td>date</td>
<td>Date the list was created</td>
</tr>
<tr>
<td>modifier</td>
<td>string</td>
<td>Last user to modify the list</td>
</tr>
<tr>
<td>modified</td>
<td>date</td>
<td>Date the list was last modified</td>
</tr>
<tr>
<td>columns</td>
<td>—</td>
<td>Column values</td>
</tr>
</tbody>
</table>

Table 49–9  Retrieved Elements for rows

For example:

```xml
    <row resourceType="urn:oracle:webcenter:space:list:row">
        <id>lr_4cc07327_49cb_4cd5_a270_182ddcc8db4a</id>
        <listId>/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fdd0_83ad_f7dac2a7ceed/lists/ls_bb806754_0652_4d49_9354_5fb62dc351d.xml</listId>
        <scope>s355923f0_2f04_4fdd0_83ad_f7dac2a7ceed</scope>
        <creator>weblogic</creator>
        <author>
            <link resourceType="urn:oracle:webcenter:people:person" rel="via" href="http://host:port/people/E9A35E80F0BC11DFFBE0FE339E56B21/lists/@self?utoken=FKld81al13QrdI87gKqOCXexL5x_w***" capabilities="urn:oracle:webcenter:read"/>
        </author>
    </row>
```
49.4.4.3.3 Resource Types Linked to from rows  

Table 49–10 lists the resource types that the client can link to from the rows resource.
Using the Lists REST API

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:space:list:rows</td>
</tr>
<tr>
<td>parent</td>
<td>urn:oracle:webcenter:space:list</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:space:list:row</td>
</tr>
</tbody>
</table>

### 49.4.4.4 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 49.4.4.4.1, "Navigation Paths to row"
- Section 49.4.4.4.2, "Supported Methods for row"
- Section 49.4.4.4.3, "Resource Types Linked to from row"

#### 49.4.4.4.1 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`

#### 49.4.4.4.2 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://host:port/rest/api/spaces/vsl/lists/\(\{\{oracle/webcenter/list/scopedMD/s355923f0_2f04_4f8d0_83ad_f7dac2a7ceed/lists/ls_c9cecbc7_756b_417b_a35f_f8b0387497\c0.xml\}/rows/\(lr_a14d427f_8515_4c82_956f_a60e6e08668c\)?utoken=FKld8lalI3QRdi8TgQkOGCEzxL5x_w**
  ```

- Request body: none
- Request headers: [Accept = application/xml | application/json]
- Response status: 200 [OK]
- Response body: `<row>`
- Retrieved elements (Table 49–11)
Using the Lists REST API

Integrating Lists

For example:

```xml
<row resourceType="urn:oracle:webcenter:space:list:row">
   <links>
      <link resourceType="urn:oracle:webcenter:space:list:row" rel="self" href="http://host:port/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***
   </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:space:list:row" rel="self" href="http://host:port/rest/api/people/E9A35E80F0BC11DFBFBE0FE339E55B21/lists/@self?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w***
            capabilities="urn:oracle:webcenter:read"/>
            capabilities="urn:oracle:webcenter:read"/>
      </links>
      <displayName>weblogic</displayName>
   </author>
   <columns urn:oracle:webcenter:space:list:columns>
      Column values
   </columns>
</row>
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>string</td>
<td>ID of the row</td>
</tr>
<tr>
<td>listId</td>
<td>string</td>
<td>ID of the parent list</td>
</tr>
<tr>
<td>scope</td>
<td>string</td>
<td>GUID of the portal to which the list belongs</td>
</tr>
<tr>
<td>creator</td>
<td>string</td>
<td>User who created the row</td>
</tr>
<tr>
<td>created</td>
<td>Date</td>
<td>Date on which the row was created</td>
</tr>
<tr>
<td>modifier</td>
<td>string</td>
<td>User who last modified the row</td>
</tr>
<tr>
<td>modified</td>
<td>Date</td>
<td>Date on which the row was last modified</td>
</tr>
<tr>
<td>columns</td>
<td>urn:oracle:webcenter:space:list:columns</td>
<td>Column values</td>
</tr>
</tbody>
</table>
Method (row): **PUT**

Use this method to update row data.

- **Resource URI**, for example:

  http://host:port/rest/api/spaces/vs1/lists/[/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cebc7_756b_417b_a35f_f0b3874979c0.xml/rows/(lr_a14d427f_8515_4c82_956f_a60e6e08668c)?utoken=FKld8la1Qdri8TgQkOCGExzL5x_w**

- **Request body**: `<row>`

- **Writable elements** (Table 49–12)
Using the Lists REST API

Integrating Lists

49.4.4.3 Resource Types Linked to from row

Table 49–13 lists the resource types that the client can link to from the row resource.

49.4.4.5 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 49.4.4.5.1, "Navigation Paths to columns"
49.4.4.1 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

49.4.4.2 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://host:port/rest/api/spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cebc7_756b_417b_a35f_fb03874979c0.xml)/columns?utoken=FKld8lal1I3QRdi8TgQkOCGEzxL5x_w**`

- **Request body:** none
- **Request headers:** [Accept = application/xml | application/json]
- **Request parameters:** none
- **Response status:** 200 [OK]
- **Response body:** `<metaColumns>`

For example:

```xml
<metaColumns resourceType="urn:oracle:webcenter:space:list:columns">
  <links>
    <link resourceType="urn:oracle:webcenter:space:list:columns" rel="self" href="http://host:port/rest/api/spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cebc7_756b_417b_a35f_fb03874979c0.xml)/columns?utoken=FKld8lal1I3QRdi8TgQkOCGEzxL5x_w**"
    capabilities="urn:oracle:webcenter:read urn:oracle:webcenter:create"/>

    <link resourceType="urn:oracle:webcenter:space:list" rel="urn:oracle:webcenter:parent" href="http://host:port/rest/api/spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cebc7_756b_417b_a35f_fb03874979c0.xml)?utoken=FKld8lal1I3QRdi8TgQkOCGEzxL5x_w**"/>
  </links>
  <items>
    <metaColumn resourceType="urn:oracle:webcenter:space:list:column">
      <links>
        <link resourceType="urn:oracle:webcenter:space:list:column" rel="self" href="http://host:port/"/>
      </links>
    </metaColumn>
  </items>
</metaColumns>
```
Using the Lists REST API

Integrating Lists

Method (columns): POST
Use this method to create a column in a list.

- Resource URI, for example:
  
  http://host:port/rest/api/spaces/vs1/lists/\{oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cecbc7_756b_417b_a35f_fb03874979c0.xml\}/columns?utoken=FKld81alI3QRdi8TqQkOCGEzxL5x_w**

  capabilities="urn:oracle:webcenter:read
turn:oracle:webcenter:delete"

- Request body: <metaColumn>
- Writable elements (Table 49–14)
Table 49–14  Writable Elements for columns

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Constraints</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>—</td>
<td>The name of the column</td>
</tr>
<tr>
<td>dataType</td>
<td>string</td>
<td>■ string</td>
<td>The data type of the column</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ number</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ datetime</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ boolean</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ person</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ image</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ richtext</td>
<td></td>
</tr>
<tr>
<td>required</td>
<td>boolean</td>
<td>■ true</td>
<td>Whether a value is required for the column</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ false</td>
<td></td>
</tr>
<tr>
<td>defaultValue</td>
<td>data type of</td>
<td>Object must match data type</td>
<td>The default value of the column</td>
</tr>
<tr>
<td>column</td>
<td></td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>maxLength</td>
<td>int</td>
<td>Valid only for string data type</td>
<td>The maximum length for a string value</td>
</tr>
<tr>
<td>rangeLow</td>
<td>int</td>
<td>Valid only for number data type</td>
<td>The low range value for a number data type</td>
</tr>
<tr>
<td>rangeHigh</td>
<td>int</td>
<td>Valid only for number data type</td>
<td>The high range value for a number data type</td>
</tr>
<tr>
<td>format</td>
<td>string</td>
<td>For number data type:</td>
<td>The format of the column</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ number</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ currency</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ percent</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>For datetime data type:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ date</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ time</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ both</td>
<td></td>
</tr>
<tr>
<td>allowLinks</td>
<td>boolean</td>
<td>■ true</td>
<td>Whether a hyperlink can be specified for a</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ false</td>
<td>column value</td>
</tr>
<tr>
<td></td>
<td></td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>linkTarget</td>
<td>string</td>
<td>■ _blank</td>
<td>_blank opens link in a new window, _self opens</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ _self</td>
<td>link in the same window</td>
</tr>
<tr>
<td></td>
<td></td>
<td>—</td>
<td>Valid only if allowLinks=true</td>
</tr>
<tr>
<td>editLines</td>
<td>int</td>
<td>Valid only for string data type</td>
<td>The number of lines when editing a column value</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(default=1)</td>
<td></td>
</tr>
<tr>
<td>peopleScope</td>
<td>string</td>
<td>■ GLOBAL</td>
<td>Whether valid users are all users in directory</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ SUB_SCOPE</td>
<td>or members of the portal that contains the list</td>
</tr>
<tr>
<td></td>
<td></td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>displayWidth</td>
<td>int</td>
<td>—</td>
<td>Display width of column in pixels</td>
</tr>
<tr>
<td>hint</td>
<td>string</td>
<td>—</td>
<td>Hint displayed to help user when entering</td>
</tr>
<tr>
<td></td>
<td></td>
<td>—</td>
<td>column value</td>
</tr>
</tbody>
</table>

1 Denotes required element

For example:

```xml
<metaColumn>
```
Using the Lists REST API

**Request headers:** Content-Type = application/xml | application/json, [Accept = application/xml | application/json]

**Response status:** 201 [Created]

**Response body:**<metaColumn>

**Retrieved elements** (Table 49–15)

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>string</td>
<td>Column ID</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the column</td>
</tr>
<tr>
<td>dataType</td>
<td>string</td>
<td>The data type of the column</td>
</tr>
<tr>
<td>required</td>
<td>boolean</td>
<td>Whether a value is required for the column</td>
</tr>
<tr>
<td>defaultValue</td>
<td>data type of column</td>
<td>The default value of the column</td>
</tr>
<tr>
<td>maxLength</td>
<td>int</td>
<td>The maximum length for a string value</td>
</tr>
<tr>
<td>rangeLow</td>
<td>int</td>
<td>The low range value for a number data type</td>
</tr>
<tr>
<td>rangeHigh</td>
<td>int</td>
<td>The high range value for a number data type</td>
</tr>
<tr>
<td>format</td>
<td>string</td>
<td>The format of the column</td>
</tr>
<tr>
<td>allowLinks</td>
<td>boolean</td>
<td>Whether a hyperlink can be specified for a column value</td>
</tr>
<tr>
<td>linkTarget</td>
<td>string</td>
<td>_blank opens link in a new window, _self opens link in the same window</td>
</tr>
<tr>
<td>editLines</td>
<td>int</td>
<td>The number of lines when editing a column value (default=1)</td>
</tr>
<tr>
<td>peopleScope</td>
<td>string</td>
<td>Whether valid users are all users in directory or members of the portal that contains the list</td>
</tr>
<tr>
<td>displayWidth</td>
<td>int</td>
<td>Display width of column in pixels</td>
</tr>
<tr>
<td>hint</td>
<td>string</td>
<td>Hint displayed to help user when entering column value</td>
</tr>
</tbody>
</table>

**For example:**

```xml
<metaColumn resourceType="urn:oracle:webcenter:space:list:column">
  <links>
    <link resourceType="urn:oracle:webcenter:space:list:column" rel="self" href="http://host:port/rest/api/spaces/vs1/lists//oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cebc7_756b_417b_a35f_fb03874979c0.xml)/columns/(lco_fe2b9856_32f3_449a_a277_18dc7f6a779e)?utoken=FKld8lalI3QRdi8TgQkOCGExzL5x_w***" capabilities="urn:oracle:webcenter:read urn:oracle:webcenter:read urn:oracle:webcenter:read urn:oracle:webcenter:read urn:oracle:webcenter:read urn:oracle:webcenter:read urn:oracle:webcenter:read urn:oracle:webcenter:read"/>
  </links>
  <id>lco_fe2b9856_32f3_449a_a277_18dc7f6a779e</id>
  <name>Notes</name>
  <dataType>richtext</dataType>
  <required>false</required>
  <displayLength>20</displayLength>
</metaColumn>
```
49.4.4.5.3 Resource Types Linked to from columns

Table 49–16 lists the resource types that the client can link to from the columns resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:space:list.columns</td>
</tr>
<tr>
<td>parent</td>
<td>urn:oracle:webcenter:space:list</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:space:list:column</td>
</tr>
</tbody>
</table>

49.4.4.6 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 49.4.4.6.1, "Navigation Paths to column"
- Section 49.4.4.6.2, "Supported Methods for column"
- Section 49.4.4.6.3, "Resource Types Linked to from column"

49.4.4.6.1 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

49.4.4.6.2 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://host:port/rest/api/spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/s155923f0_2f04_83ad_f7dac2a7ceed/lists/ls_c9cecb0_756b_417f_a35f_fb03874979cc0.xml)/columns/(lco_fe2b8863_32f3_449a_a277_18dc7f6a779e)?utoken=FKld81al13QRdi8TgQkOCGExL5x_w**

- Request body: none
- Request headers: [Accept = application/xml | application/json]
- Response status: 200 [OK]
- Response body: <metaColumn>

<allowLinks>false</allowLinks>
</metaColumn>
Using the Lists REST API

- Retrieved elements (Table 49–17)

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>string</td>
<td>Column ID</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the column</td>
</tr>
<tr>
<td>dataType</td>
<td>string</td>
<td>The data type of the column</td>
</tr>
<tr>
<td>required</td>
<td>boolean</td>
<td>Whether a value is required for the column</td>
</tr>
<tr>
<td>defaultValue</td>
<td>data type of column</td>
<td>The default value of the column</td>
</tr>
<tr>
<td>maxLength</td>
<td>int</td>
<td>The maximum length for a string value</td>
</tr>
<tr>
<td>rangeLow</td>
<td>int</td>
<td>The low range value for a number data type</td>
</tr>
<tr>
<td>rangeHigh</td>
<td>int</td>
<td>The high range value for a number data type</td>
</tr>
<tr>
<td>format</td>
<td>string</td>
<td>The format of the column</td>
</tr>
<tr>
<td>allowLinks</td>
<td>boolean</td>
<td>Whether a hyperlink can be specified for a column value</td>
</tr>
<tr>
<td>linkTarget</td>
<td>string</td>
<td>_blank opens link in a new window, _self opens link in the same window</td>
</tr>
<tr>
<td>editLines</td>
<td>int</td>
<td>The number of lines when editing a column value (default=1)</td>
</tr>
<tr>
<td>peopleScope</td>
<td>string</td>
<td>Whether valid users are all users in directory or members of the portal that contains the list</td>
</tr>
<tr>
<td>displayWidth</td>
<td>int</td>
<td>Display width of column in pixels</td>
</tr>
<tr>
<td>hint</td>
<td>string</td>
<td>Hint displayed to help user when entering column value</td>
</tr>
</tbody>
</table>

- For example:

```xml
<metaColumn resourceType="urn:oracle:webcenter:space:list:column">
  <links>
    <link resourceType="urn:oracle:webcenter:space:list:column"
            rel="self" href="http://host:port/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://host:port/rest/api/spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cecbc7_756b_417b_a35f_
Using the Lists REST API

- Request body: `<metacolumn>`
- Writable elements (Table 49–18)

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Constraints</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name¹</td>
<td>string</td>
<td>—</td>
<td>The name of the column</td>
</tr>
<tr>
<td>dataType²</td>
<td>string</td>
<td>■ string</td>
<td>The data type of the column</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ number</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ datetime</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ boolean</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ person</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ image</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ richtext</td>
<td></td>
</tr>
<tr>
<td>required</td>
<td>boolean</td>
<td>■ true</td>
<td>Whether a value is required for the column</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ false</td>
<td></td>
</tr>
<tr>
<td>defaultValue</td>
<td>data type of column</td>
<td>Object must match data type</td>
<td>The default value of the column</td>
</tr>
<tr>
<td>maxLength</td>
<td>int</td>
<td>Valid only for string data type</td>
<td>The maximum length for a string value</td>
</tr>
<tr>
<td>rangeLow</td>
<td>int</td>
<td>Valid only for number data type</td>
<td>The low range value for a number data type</td>
</tr>
<tr>
<td>rangeHigh</td>
<td>int</td>
<td>Valid only for number data type</td>
<td>The high range value for a number data type</td>
</tr>
<tr>
<td>format</td>
<td>string</td>
<td>For number data type:</td>
<td>The format of the column</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ number</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ currency</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ percent</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>For datetime data type:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ date</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ time</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ both</td>
<td></td>
</tr>
<tr>
<td>allowLinks</td>
<td>boolean</td>
<td>■ true</td>
<td>Whether a hyperlink can be specified for a column value</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ false</td>
<td></td>
</tr>
<tr>
<td>linkTarget</td>
<td>string</td>
<td>■ _blank</td>
<td>_blank opens link in a new window, _self opens link in the same window</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ _self</td>
<td></td>
</tr>
<tr>
<td>editLines</td>
<td>int</td>
<td>Valid only for string data type</td>
<td>The number of lines when editing a column value (default=1)</td>
</tr>
</tbody>
</table>

---

49-32  Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper
49.4.4.6.3 Resource Types Linked to from column

Table 49–18 (Cont.) column Writable Elements

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Constraints</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>peopleScope</td>
<td>string</td>
<td>■ GLOBAL</td>
<td>Whether valid users are all users in directory or members of the portal that contains the list</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ SUB_SCOPE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid only for person data type</td>
<td></td>
</tr>
<tr>
<td>displayWidth</td>
<td>int</td>
<td>—</td>
<td>Display width of column in pixels</td>
</tr>
<tr>
<td>hint</td>
<td>string</td>
<td>—</td>
<td>Hint displayed to help user when entering column value</td>
</tr>
</tbody>
</table>

1 Denotes required element

- For example:
  ```xml
  <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://host:port/rest/api/spaces/vs1/lists/\{oracle/webcenter/list/scopedMD/
  s355923f0_2f04_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

49.4.4.6.3 Resource Types Linked to from column

Table 49–19 lists the resource types that the client can link to from the column resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td><code>urn:oracle:webcenter:space:list:column</code></td>
</tr>
</tbody>
</table>

49.5 Troubleshooting Lists

**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 4.2.2, "Setting Up a Database Connection."

**Problem**
A 401 Unauthorized error appears.

**Solution**
It is likely that security is not set up for the portal. Or it may be that there was insufficient permission to perform the attempted task.
This chapter describes how to integrate notifications with a Portal Framework application. Notifications provide a means of subscribing to services and application objects and, when those objects change, receiving notification across selected messaging channels.

This chapter includes the following topics:

- Section 50.1, "Introduction to Notifications"
- Section 50.2, "Basic Configuration for Notifications"
- Section 50.3, "Advanced Information for Notifications"

50.1 Introduction to Notifications

This section provides an overview of Notifications. It includes the following subsections:

- Section 50.1.1, "Understanding Notifications"
- Section 50.1.2, "Requirements for Notifications"
- Section 50.1.3, "What Happens with Notifications at Runtime"

50.1.1 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 50–1 describes the types of activities that can trigger a notification and indicates the level at which the subscription is made.
Introduction to Notifications

Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper

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.

### 50.1.2 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 50.2.1, “Adding the Notifications Namespace.”

Subscription capability is built into the services that support it and does 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

<table>
<thead>
<tr>
<th>Activity</th>
<th>Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>A user sends you an invitation to connect</td>
<td>Application</td>
</tr>
<tr>
<td>A user posts a message to your Message Board</td>
<td>Application</td>
</tr>
<tr>
<td>A user likes your Message Board post (messages explicitly set on a Message Board and not those added from Publisher to the Activity Stream)</td>
<td>Application</td>
</tr>
<tr>
<td>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)</td>
<td>Application</td>
</tr>
<tr>
<td>A user posts feedback for you</td>
<td>Application</td>
</tr>
<tr>
<td>A user replies to a discussion topic</td>
<td>Object</td>
</tr>
<tr>
<td>A user comments on a discussion topic</td>
<td>Object</td>
</tr>
<tr>
<td>A user deletes a discussion topic</td>
<td>Object</td>
</tr>
<tr>
<td>A user comments on a document</td>
<td>Object</td>
</tr>
<tr>
<td>A user likes a document</td>
<td>Object</td>
</tr>
<tr>
<td>A user updates a document</td>
<td>Object</td>
</tr>
<tr>
<td>A user deletes a document</td>
<td>Object</td>
</tr>
<tr>
<td>A user comments on a wiki document</td>
<td>Object</td>
</tr>
<tr>
<td>A user likes a wiki document</td>
<td>Object</td>
</tr>
<tr>
<td>A user updates a wiki document</td>
<td>Object</td>
</tr>
<tr>
<td>A user deletes a wiki document</td>
<td>Object</td>
</tr>
<tr>
<td>A user comments on a blog entry</td>
<td>Object</td>
</tr>
<tr>
<td>A user likes a blog entry</td>
<td>Object</td>
</tr>
<tr>
<td>A user updates a blog entry</td>
<td>Object</td>
</tr>
<tr>
<td>A user deletes a blog entry</td>
<td>Object</td>
</tr>
</tbody>
</table>

Table 50–1  Activities that Can Trigger Notifications

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.

### 50.1.2 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 50.2.1, “Adding the Notifications Namespace.”

Subscription capability is built into the services that support it and does 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 50.2.2, "Creating a Notifications Connection to a Mail Server," and Section 50.2.3, "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 50.2.4, "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 50.2.6, "Setting Security for Notifications."

50.1.3 What Happens with Notifications at Runtime

At runtime, users set up their preferred messaging channels through UMS Preferences. For more information, see the "Establishing and Managing Your Messaging Channels and Filters" section in Using Oracle WebCenter Portal.

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. The task flow presents slightly different options depending on whether it is located on the Home portal (Figure 50–1) or a portal (Figure 50–2).

Figure 50–1  Subscription Preferences Task Flow - Home Portal
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 50–3).

See Also: For more information about Notifications task flows, see Section 50.2.5.1, "Notifications Task flows."

50.2 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 50.2.1, "Adding the Notifications Namespace"
Basic Configuration for Notifications

50.2.1 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 50–1):

```
Tip: You can find the adf-config.xml file in the Application Resources panel, under Descriptors, in the ADF META-INF folder.
```

Example 50–1 Configuring the Application to Recognize Notifications Configuration

```xml
xmlns:notificationC="http://xmlns.oracle.com/webcenter/notification/config"
```

For example (bold) (Figure 50–2):

Example 50–2 Notifications Configuration in the adf-config Element

```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"
    ...
```

50.2.2 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 35.2.2, "Setting up Connections for Mail"). Once this is established, you add the following entry in the adf-config.xml file to associate the Mail server with Notifications (Example 50–3).

Example 50–3 Associating a Mail Server with Notifications in adf-config.xml

```xml
<notificationC:adf-notification-config
    xmlns="http://xmlns.oracle.com/webcenter/notification/config"
    xmlns:connectionType="mailto">
    <ConnectionType>MAIL</ConnectionType>
    <ConnectionName>someMailConnection</ConnectionName>
</notificationC:adf-notification-config>
```

Where ConnectionName is the name of a configured Mail server.

Example 50–4 presents a code sample, with the previous code sample in place (bold).

Example 50–4 A Mail Connection for Notifications in adf-config.xml

```xml
<?xml version="1.0" encoding="US-ASCII" ?>
```
Basic Configuration for Notifications

50.2.3 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 41.3.1, “Setting up Connections for Worklists”). 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 50–5).

Example 50–5 Configuring Notifications to Use the BPEL Server for Messaging

```xml
<notificationC:adf-notification-config
    xmlns:notificationC="http://xmlns.oracle.com/webcenter/notification/config">
    <ConnectionType>BPEL</ConnectionType>
    <ConnectionName>worklistConnection</ConnectionName>
</notificationC:adf-notification-config>
```
Basic Configuration for Notifications

Integrating Notifications

```xml
<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 50–6 presents a code sample, with the previous code sample in place (bold).

**Example 50–6  Notifications Entries in the adf-config.xml File**

```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/webcenter/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>

    <CredentialStoreContext credentialStoreClass="oracle.adf.share.security.providers.jps.CSCredentialStore"
        credentialStoreLocation=".../.../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>
</notificationC:adf-notification-config>
```

Integrating Notifications  50-7
Basic Configuration for Notifications

50.2.4 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 50.2.4.1, “About Subscription Defaults”
- Section 50.2.4.2, “Setting Notifications Subscription Defaults”

50.2.4.1 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 selected by default in the Subscription Preferences task flow. If subscription-enabled="false", then the associated subscription option is not selected 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 50–2 illustrates the effect of custom administrator-level subscriptions settings on the appearance of the Subscription Preferences task flow.
Integrating Notifications

### Table 50–2  Effect of Administrator Defaults on Subscriptions Preferences

<table>
<thead>
<tr>
<th>subscription-enabled</th>
<th>end-user-configurable</th>
<th>Option in Preferences</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>True</td>
<td>Rendered normally, check box is selected</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>Grayed out, check box is selected</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>Rendered normally, check box is deselected</td>
</tr>
<tr>
<td>False</td>
<td>False</td>
<td>Hidden, check box is hidden</td>
</tr>
</tbody>
</table>

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, check box is deselected.

### Table 50–3  Application-Level Activities that Can Trigger Notifications

<table>
<thead>
<tr>
<th>Activity</th>
<th>Related Service ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>A user sends you an invitation to connect</td>
<td><code>oracle.webcenter.peopleconnections.connections</code></td>
</tr>
<tr>
<td>A user posts a message to your Message Board</td>
<td><code>oracle.webcenter.peopleconnections.wall</code></td>
</tr>
<tr>
<td>A user likes your post on another user’s Message Board</td>
<td><code>oracle.webcenter.peopleconnections.wall</code></td>
</tr>
<tr>
<td>A user comments on your post on another user’s Message Board</td>
<td><code>oracle.webcenter.peopleconnections.wall</code></td>
</tr>
<tr>
<td>A user posts feedback for you</td>
<td><code>oracle.webcenter.peopleconnections.kudos</code></td>
</tr>
</tbody>
</table>

### 50.2.4.2 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 50–7** provides sample content for an application-wide subscription preferences setting file and an example of each required option.

### Example 50–7  Sample Subscriptions Settings XML File

```xml
<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'/>
</subscription-settings>
</notification-service_settings>
```
Basic Configuration for Notifications

```xml
<subscription-settings>
  <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 "importMetadata" section in WebLogic Scripting Tool Command Reference.

For example:

```bash
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 50–4 describes the effect of various combinations of settings for the service ID oracle.webcenter.peopleconnections.connections.

<table>
<thead>
<tr>
<th>subscription-enabled</th>
<th>end-user-configurable</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>■ The subscribing user receives a notification message when another user sends him an invitation to connect.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ The user can change this default.</td>
</tr>
</tbody>
</table>

Table 50–4 Effects of Subscription Configurations for Connections
### Table 50–4  (Cont.) Effects of Subscription Configurations for Connections

<table>
<thead>
<tr>
<th>subscription-enabled</th>
<th>end-user-configurable</th>
<th>Effect</th>
</tr>
</thead>
</table>
| 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 50–5 describes the effect of various combinations of settings for the service ID oracle.webcenter.peopleconnections.wall.

### Table 50–5 Effects of Subscription Configurations for Message Board

<table>
<thead>
<tr>
<th>subscription-enabled</th>
<th>end-user-configurable</th>
<th>Effect</th>
</tr>
</thead>
</table>
| 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. |
| 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 50–6 describes the effect of various combinations of settings for the service ID oracle.webcenter.peopleconnections.kudos.
Table 50–6  Effect of Subscription Configurations for Feedback

<table>
<thead>
<tr>
<th>subscription-enabled</th>
<th>end-user-configurable</th>
<th>Effect</th>
</tr>
</thead>
</table>
| 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. |

50.2.5 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 50.2.5.1, "Notifications Task flows"
- Section 50.2.5.2, "How to Add a Notifications Task Flow to a Page"

50.2.5.1 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. The task flow presents slightly different options depending on whether it is located on the Home portal (Figure 50–1) or a portal (Figure 50–2)

Figure 50–4  Subscription Preferences Task Flow - Home Portal

Notify Me  About

- Message Board  
  New posts for me; Commented on my posts; Liked my posts
- Portal Management  
  My new membership; My role changes; My membership removal
- Connections  
  Connection invitations to me
- Feedback  
  New posts for me
Subscription Viewer provides controls for viewing and unsubscribing from object-level subscriptions (Figure 50–6).

Figure 50–6  Subscription Viewer Task Flow

<table>
<thead>
<tr>
<th>Portal Name</th>
<th>Service</th>
<th>Resource Type</th>
<th>Resource Name</th>
<th>Created On</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sales Portal</td>
<td>Discussions</td>
<td>All Resources</td>
<td></td>
<td>5/2/13 10:21 PM</td>
</tr>
<tr>
<td>Sales Portal</td>
<td>Discussions</td>
<td>Topic</td>
<td>Sales Giveaway Ideas</td>
<td>5/2/13 10:20 PM</td>
</tr>
<tr>
<td>Sales Portal</td>
<td>Documents</td>
<td>All Resources</td>
<td></td>
<td>5/2/13 10:21 PM</td>
</tr>
</tbody>
</table>

50.2.5.2 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 4.2.1, "How to Prepare Your Application to Consume Tools and 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.
The task flow appears on the page. For example, in the Source view, you will see the content depicted in Example 50–8.

**Example 50–8  Source View Content of the Subscription Preferences Task Flow**

```html
<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. The options shown in the task flow are slightly different depending on whether you added the task flow to the Home portal (Figure 50–7) or a portal (Figure 50–7).

**Figure 50–7  Subscription Preferences Task Flow - Home Portal**

```
Subscription Preferences
```

- **Notify Me About**
  - **Message Board**
    - New posts for me; Commented on my posts; Liked my posts
  - **Portal Management**
    - My new membership; My role changes; My membership removal
  - **Connections**
    - Connection invitations to me
  - **Feedback**
    - New posts for me

**Figure 50–8  Subscription Preferences Task Flow - Portal**

```
Subscription Preferences
```

- **Notify Me About**
  - **Events**
    - New events; Deleted events; Updated events
  - **Documents**
    - New documents; New wikis
  - **Blogs**
    - New blog posts
  - **Discussions**
    - New topics; New forums
  - **Announcements**
    - New announcements

See Also: For information about Subscription Preferences task flow input parameters, see Section 50.2.5.3, "Subscription Preferences Task Flow Input Parameters."
50.2.5.3 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 50–7 lists and describes the input parameters associated with this task flow.

<table>
<thead>
<tr>
<th>Input Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| hideActions     | A control for showing or hiding the **Save** button and **Refresh** icon on the task flow  
|                 |  
|                 |  
|                 |  
|                 |  
| scopeName       | The name of the scope for which to set subscription preferences. |

50.2.6 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 Portal Framework application, see Section 74.3, “Configuring ADF Security.” For details on how to implement a complete security solution for your Portal Framework application, see Chapter 74, "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.

50.3 Advanced Information for Notifications

This section provides information about the APIs that are available for use with Notifications. It includes the following subsections:

- Section 50.3.1, "Using Notifications Java APIs"
- Section 50.3.2, "Using Notifications Data Controls"

50.3.1 Using Notifications Java APIs

This section provides an overview of Notifications Java APIs. It includes the following subsections:

- Section 50.3.1.1, "Configuration Settings Required to Use Notifications Java APIs"
- Section 50.3.1.2, "Introduction to Notifications Java APIs"
- Section 50.3.1.3, "How to Set up Your Application to Use Notifications Java APIs"

**See Also:** For detailed information about the Notifications Java APIs, see the *Java API Reference for Oracle WebCenter Portal.*
50.3.1.1 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:

```xml
<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.

50.3.1.2 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 subscriptions. 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 removing subscriptions for the corresponding service.

**See Also:** For detailed information about the Notifications Java APIs, see the Java API Reference for Oracle WebCenter Portal.

50.3.1.3 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 WebCenter Portal - Notification Service, as shown in Figure 50–9. For information about adding the WebCenter Portal extension, see Chapter 2, "Setting Up Your Development Environment."
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 50–10).

3. Click **OK**.
4. Click OK.

50.3.2 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 Portal Framework application, see Section 4.1.3, "Using WebCenter Portal Data Controls."

This section includes the following subsections:

- Section 50.3.2.1, "Adding a Data Control to Your Project"
- Section 50.3.2.2, "Understanding the Notification Subscription Data Control"
- Section 50.3.2.3, "Notification Subscription Data Control Methods and Attributes"
- Section 50.3.2.4, "Notification Subscription Data Control Classes"

50.3.2.1 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
50.3.2.2 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. You can also build a custom subscription viewer UI, where all of the current user's object-level subscriptions are listed.

50.3.2.3 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 50.3.2.3.1, "Method: getUserSubscriptionPreferences"
- **Section 50.3.2.3.2, "Method: getScopeSubscriptionPreferences"
- **Section 50.3.2.3.3, "Method: saveUserSubscriptionPreferences"
- **Section 50.3.2.3.4, "Method: saveScopeSubscriptionPreferences"
- **Section 50.3.2.3.5, "Method: getSubscriptions"
- **Section 50.3.2.3.6, "Method: unsubscribe"

50.3.2.3.1 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.

50.3.2.3.2 Method: getScopeSubscriptionPreferences This method returns the scope level subscription preferences for the current user.
This method is primarily useful in a WebCenter Portal. In a Portal 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 WebCenter Portal, the scope referenced here is equivalent to a portal.

**Return Values**

Returns a collection of `SubscriptionPreference` objects. Each object indicates the current user’s subscription preferences for each service within the specified scope.

### 50.3.2.3.3 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

### 50.3.2.3.4 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

### 50.3.2.3.5 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 50–8 lists and describes the input parameters supported by this method.

<table>
<thead>
<tr>
<th>Input Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FilterOption</td>
<td>An object that contains filter conditions for querying a user’s subscriptions. <code>FilterOption</code> supports filtering based on portal name (WebCenter Portal only) and object name.</td>
</tr>
<tr>
<td>SortOption</td>
<td>An object that contains the sort or ordering condition for querying subscriptions. It supports sorting by object name, scope name (WebCenter Portal only), or the creation date of subscription.</td>
</tr>
<tr>
<td>offset</td>
<td>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.</td>
</tr>
</tbody>
</table>
**Table 50–8 (Cont.) Input Parameters Supported by the method getSubscriptions**

<table>
<thead>
<tr>
<th>Input Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fetchSize</td>
<td>This input parameter takes an integer as a value. The maximum number of rows to be returned in one call.</td>
</tr>
</tbody>
</table>

**Return Values**

Returns a collection of Subscription objects matching the filter criteria and other input parameters in the order specified by the sort option.

**50.3.2.3.6 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.

Unsubscribing from a particular service at the application and portal level, or at either level, can be achieved 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 50–9 lists and describes the input parameters supported by this method.

**Table 50–9 Input Parameters Supported by the method Unsubscribe**

<table>
<thead>
<tr>
<th>Input Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scopeGuid</td>
<td>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 WebCenter Portal, the scope referenced here is equivalent to a portal.</td>
</tr>
<tr>
<td>serviceId</td>
<td>The ID of the unsubscribed service</td>
</tr>
<tr>
<td></td>
<td>Valid service IDs include:</td>
</tr>
<tr>
<td></td>
<td>■ oracle.webcenter.collab.forum</td>
</tr>
<tr>
<td></td>
<td>■ oracle.webcenter.doclib</td>
</tr>
<tr>
<td>objectId</td>
<td>The GUID of the unsubscribed object.</td>
</tr>
</tbody>
</table>

**Return Values**

None

**50.3.2.4 Notification Subscription Data Control Classes**

The Notification Subscriptions Data Control exposes two classes as return values:

■ SubscriptionPreference

■ Subscription

**50.3.2.4.1 SubscriptionPreference**

The class SubscriptionPreference represents application- or scope-level subscriptions as a preference and enables setting and removing subscriptions for the corresponding service (this is followed by a save call on the data control).

Table 50–10 lists and describes the properties associated with the SubscriptionPreference class.
Subscription

The class `Subscription` is purely read-only and represents portal- (WebCenter Portal only) and object-level subscriptions. Table 50–11 lists and describes the properties associated with the `Subscription` class.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hintsText</td>
<td>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.</td>
</tr>
<tr>
<td>serviceIcon</td>
<td>An icon for the service</td>
</tr>
<tr>
<td>serviceId</td>
<td>ID of the corresponding service</td>
</tr>
<tr>
<td>serviceName</td>
<td>Display name of the service</td>
</tr>
<tr>
<td>subscriptionEnabled</td>
<td>A flag that indicates whether the subscription is enabled. This attribute also has a setter to support the select/deselect operation on the UI.</td>
</tr>
<tr>
<td>userOverrideAllowed</td>
<td>A flag that indicates whether to allow the user to override the default setting</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>creationDate</td>
<td>The datetime this subscription was created</td>
</tr>
<tr>
<td>objectId</td>
<td>The ID of the object, if this represents an object-level subscription, otherwise null.</td>
</tr>
<tr>
<td>objectName</td>
<td>The name of the object, if this represents an object-level subscription, otherwise null.</td>
</tr>
<tr>
<td>objectType</td>
<td>The object type (as defined by the corresponding service) when this subscription represents object-level subscriptions, otherwise null.</td>
</tr>
<tr>
<td>scopeGuid</td>
<td>The GUID of the scope or portal</td>
</tr>
<tr>
<td>scopeName</td>
<td>The display name of the scope or portal</td>
</tr>
<tr>
<td>serviceId</td>
<td>The ID of the service for this subscription object</td>
</tr>
<tr>
<td>serviceName</td>
<td>The name of the service</td>
</tr>
<tr>
<td>userId</td>
<td>The ID of the user who created this subscription</td>
</tr>
</tbody>
</table>
This chapter describes how to integrate recent activities with a Portal Framework application. Recent Activities stream the most recent application actions.

This chapter includes the following topics:

- **Section 51.1, "Introduction to Recent Activities"
- **Section 51.2, "Basic Configuration for Recent Activities"
- **Section 51.3, "Advanced Information for Recent Activities"

**Note:** For information about using recent activities, see the "Working with Recent Activities" chapter in *Using Oracle WebCenter Portal*.

### 51.1 Introduction to Recent Activities

Recent activities enable users to view the most recent changes in your application. For example, the documents tool automatically produces the information that recent activities uses to display the most recent changes to the document library. This 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.

Recent activities can track creation, modification, and deletion of the following portal objects:

- Documents
- Announcements
- Discussion forums
- Pages
- List definitions

For more information, see Chapter 32, "Integrating Announcements," Chapter 33, "Integrating Discussions," Chapter 28, "Integrating Documents," and Chapter 17, "Enabling Runtime Creation and Management of Pages."

Through APIs, you can control the display of business activities.

**See Also:** For information about tracking business activities using APIs, see *Java API Reference for Oracle WebCenter Portal*.

This section contains the following subsections:

- **Section 51.1.1, "Understanding Recent Activities"**
51.1.1 Understanding Recent Activities

Recent activities provides a summary view of recent changes to a variety of tools and 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, recent activities works with documents, announcements, discussions, and pages. You can limit what recent activities are tracked. For more information about this setting, see Section 51.3, "Advanced Information for Recent Activities." You can also learn more about using Oracle WebCenter Portal tools and services in Chapter 4, "Preparing Your Application for WebCenter Portal Tools and Services."

Recent activities functionality is similar to the activity stream functionality (for more information, see Section 37.1.1.1, "Activity Stream"). Both track the application activities of integrated tools and services, though the activity stream tracks a broader range of tools and services. For example, recent activities tracks documents, announcements, discussions, and pages. The activity stream tracks those, as well as people, wikis, and blogs. Recent activities tracks activities no matter who performs the action. The 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 recent activities and the activity stream can be summarized as follows: recent activities provides an overview of what is happening in an application; the activity stream provides an overview of what is happening with a user’s connections.

51.1.2 What Happens at Runtime

At runtime, the Recent Activities task flow fetches the latest information about the tools and services that are configured and running in your application. Figure 51–1 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 tool
- Five discussion threads have been modified or added using the Discussions tool
- One announcement has been modified or added using the Announcements tool

The time range choices (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 51–1).

Recent activities uses the search capabilities of WebCenter to query activities for specified periods. Search capabilities are implemented across the following WebCenter Portal tools and services: documents, announcements, discussions, and pages. These services, if configured in an application, respond to queries from recent activities and return items pertinent to the search criteria. Responses to queries (search results) are always based on the search predicate, $DCMI_CHANGED > [start time]$. Here $start time$ is one of the time periods that you configured in the Edit Task Flow Binding dialog at design time. Users can select a select a time period in the Show list at runtime.
51.2 Basic Configuration for Recent Activities

This section describes how to add recent activities functionality to your Portal Framework application.

This section contains the following subsections:

- Section 51.2.1, "Setting up Connections for Recent Activities"
- Section 51.2.2, "Adding Recent Activities at Design Time"
- Section 51.2.3, "Modifying Recent Activities Task Flow Parameters"
- Section 51.2.4, "Setting Security for Recent Activities"

51.2.1 Setting up Connections for Recent Activities

You do not need to set up specific connections for recent activities. You only need to ensure that the WebCenter Portal components you want to track are configured.

51.2.2 Adding Recent Activities at Design Time

To use recent activities, you must add its task flow to your application. Ensure that you have at least one of the components 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.

Recent activities automatically picks up other components in your application and looks for recent activities within these components. Before running your application containing recent activities, however, you may want to ensure your application contains at least one of the components you want to monitor.

To add the Recent Activities task flow to your application:

1. Follow the steps in Section 4.2.1, "How to Prepare Your Application to Consume Tools and Services" to implement security and create a customizable page in your Portal Framework application.

2. Open the source of the customizable page.

3. Ensure that your application includes at least one of the components that recent activities tracks.

   **Note:** If you run an application that does not contain one of these components, then recent activities displays an error saying

   Warning: Unable to obtain recent activity data.

4. In the Resource Palette, open the WebCenter Portal - Services Catalog, then open the Task Flows folder.
5. Click **Recent Activities** and drag it onto your page after the `<af:form>` tag, and select **Region**.

**Figure 51–2   Edit Task Flow Binding Dialog**

In the Edit Task Flow Binding dialog shown in **Figure 51–3**, 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, then recent activities uses the default values.

**Table 51–1** 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 51.3.1, "Refining the Behavior of Recent Activities."**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupSpace</td>
<td>Leave the value of this parameter as null.</td>
</tr>
<tr>
<td>timePeriodShort</td>
<td>Default value: Today, Possible values: Today, Yesterday, or a time in minutes</td>
</tr>
<tr>
<td>timePeriodMedium</td>
<td>Default value: Since Yesterday, Possible values: Today, Yesterday, or a time in minutes</td>
</tr>
<tr>
<td>timePeriodLong</td>
<td>Default value: 10080 (10080 minutes is equivalent to 7 days.), Possible values: Today, Yesterday, or a time in minutes</td>
</tr>
<tr>
<td>timePeriodLongest</td>
<td>Default value: 43200 (43200 minutes is equivalent to 30 days.), Possible values: Today, Yesterday, or a time in minutes</td>
</tr>
</tbody>
</table>
6. Click OK. The binding appears on your page, as shown in Figure 51–4.

7. Save your project and run your page to the browser.
51.2.3 Modifying Recent Activities Task Flow Parameters

The Recent Activities task flow has required and optional task flow binding parameters. You can adjust the parameter values when you drop a task flow onto a page or after you have placed a task flow on a page. For information about modifying parameters, see Section 32.2.2.3, "How to Modify Announcement Task Flow Parameters." The procedure for modifying parameters is identical across all task flows.

51.2.4 Setting Security for Recent Activities

Recent activities uses the security applied to the underlying components being analyzed. So, the information returned by the component 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 only sees activities returned on components exposed to the public that do not require authentication to view.

51.3 Advanced Information for Recent Activities

In addition to the task flow parameters, you can further refine the behavior of recent activities. This section describes how to perform that refinements. It also describes how you can obtain RSS news feed for recent activities.

This section contains the following subsections:

- Section 51.3.1, "Refining the Behavior of Recent Activities"
- Section 51.3.2, "Obtaining RSS News Feed URL for Recent Activities"

51.3.1 Refining the Behavior of Recent Activities

You can change a variety of the recent activities parameters after you have added the task flow 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:

```xml
<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>
```

In this entry, you can change the settings described in Table 51–2.
### Table 51–2  Recent Activities Settings in the adf-config.xml File

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
</table>
| numServiceRows | Default value: 25  
The maximum number of rows displayed for a single component.  
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. |
| shortPeriod | Default value: Today  
See Table 51–1. 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 51–1. 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 51–1. 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 51–1. 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 components you do not want to track. The following example omits any changes to the Page service from displaying in recent activities:  
```xml
<services-filter>  
<exclude>oracle.webcenter.page</exclude>  
</services-filter>
```

By default, no components are omitted.  
To filter out other components use the service ID values listed in Table G–7. |

#### 51.3.2 Obtaining RSS News Feed URL for Recent Activities

You can expose WebCenter Portal functionality in a Portal Framework application. Your Portal Framework application users can find out what is happening in a specific portal through RSS news feeds.  

Configure RSS news feeds for recent activities to enable users to view recent activities from within a portal. To obtain the portal RSS news feed URL for recent activities, use either of the following WebCenter Portal APIs:

- `getServiceRssFeedURL`
- `GetServiceRssFeedURLbyGUID`

To obtain an RSS feed URL, you must identify the portal (by name or GUID) and specify the component required (by service ID). The service ID for recent activities is `GroupSpaceWSClient.RECENT_ACTIVITY_SERVICE_ID`. For information about how to use these APIs, see Section 56.1.5.3.9, "Retrieving RSS Feed URLs for WebCenter Portal"
Tools and Services."
This chapter describes how to integrate the RSS Viewer at design time, allowing users to add and view RSS 2.0 formatted feeds within a Portal Framework application.

This chapter includes the following sections:

- Section 52.1, "Introduction to RSS"
- Section 52.2, "Basic Configuration for the RSS"

For more information about managing and including RSS feeds, see:

- the "Managing RSS" chapter in Administering Oracle WebCenter Portal
- the "Adding RSS Feeds to a Portal" chapter in Building Portals with Oracle WebCenter Portal
- the "Monitoring RSS Feeds" chapter in Using Oracle WebCenter Portal

### 52.1 Introduction to RSS

Really Simple Syndication (RSS) provides a means of accessing the content of many different web sites from a single location—a news reader. WebCenter Portal provides the functionality that encompasses the RSS Viewer and the ability to show RSS feeds from various WebCenter Portal components. The RSS Viewer enables you to view external news feeds from different web sites from within your Portal Framework applications.

**Note:** Although RSS is supported within discussion forums for portals, Portal Framework applications do not support RSS within discussion forums.

This section provides an overview of the features and requirements with RSS. It includes the following subsections:

- Section 52.1.1, "Understanding RSS"
- Section 52.1.2, "Requirements for RSS"
- Section 52.1.3, "What Happens at Runtime"

### 52.1.1 Understanding RSS

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 52–1. 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 74.13, "Working with External Applications."

**Figure 52–1  RSS Details Specified at Design Time**

52.1.2 Requirements for RSS

The RSS functionality does not require any back-end server. You do not need to set up a connection to use this. However, you can set up a proxy server for RSS, if required.

52.1.3 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 RSS at runtime, see the "Monitoring RSS Feeds" chapter in *Using Oracle WebCenter Portal*. 
52.2 Basic Configuration for the RSS

This section describes how to set up the proxy server for RSS and add the RSS Viewer task flow to your application.

This section includes the following subsections:

- Section 52.2.1, "Setting Up a Proxy Server for RSS"
- Section 52.2.2, "Adding RSS Functionality at Design Time"
- Section 52.2.3, "Setting Security for the RSS Viewer"

52.2.1 Setting Up a Proxy Server for RSS

The RSS functionality 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 RSS:

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 52–3).
4. Click OK.
52.2.2 Adding RSS Functionality 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 52.2.2.1, "About RSS Viewer Task Flow"
- Section 52.2.2.2, "How to Add the RSS Viewer Task Flow to Your Application"
- Section 52.2.2.3, "How to Modify RSS Viewer Task Flow Parameters"

52.2.2.1 About RSS Viewer Task Flow

WebCenter Portal includes the RSS Viewer task flow, 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.

52.2.2.2 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 4.2.1, "How to Prepare Your Application to Consume Tools and 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 74.13, "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 52–4.

![Figure 52–4 Example of the Edit Task Flow Binding Dialog for RSS](image)

Table 52–1 describes the possible values for the RSS Viewer task flow binding parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>rssFeedLocation</td>
<td>Enter the location of the RSS feed. For example, to use the Oracle Press Releases RSS feed, enter: ${'<a href="http://www.oracle.com/rss/rss_ocom_pr.xml'%7D">http://www.oracle.com/rss/rss_ocom_pr.xml'}</a></td>
</tr>
<tr>
<td>extAppId</td>
<td>Enter the name of the external application you want to use to authenticate the Portal Framework application with the RSS feed. If the RSS feed does not require authentication, then you do not need to set up and identify an external application for this.</td>
</tr>
</tbody>
</table>

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 52–1.

Example 52–1 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>
```

52.2.2.3 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:
Basic Configuration for the RSS

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 52–5).

**Figure 52–5  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.

### 52.2.3 Setting Security for the RSS Viewer

To use RSS with a public RSS feed, you do not need to set security. To use RSS with an RSS feed that requires authentication, you can set up an external application for your 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 74.13, "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 Portal Framework application, the View grant is automatically added to the authenticated-role.
Part IX contains the following chapters:

- Chapter 53, "Using Oracle WebCenter Portal REST APIs"
- Chapter 54, "Integrating Other Oracle Applications"
- Chapter 55, "Developing Components for WebCenter Portal Using JDeveloper"
- Chapter 56, "Integrating with WebCenter Portal"
This chapter describes the WebCenter Portal REST APIs, with which you can retrieve, adding, modify, and delete server data.

This chapter includes the following topics:

- Section 53.1, "Introduction to REST"
- Section 53.2, "Understanding the Username-Based Security Token Encryption"
- Section 53.3, "Benefits of Using REST"
- Section 53.4, "Introduction to WebCenter Portal's REST APIs"
- Section 53.5, "Understanding the Link Model"
- Section 53.6, "Understanding Items Hypermedia"
- Section 53.7, "Navigating Hypermedia Using HTTP"
- Section 53.8, "Security Considerations for WebCenter Portal REST APIs"
- Section 53.9, "Security Considerations for CMIS REST APIs"
- Section 53.10, "Understanding Common Types"
- Section 53.11, "Managing Caches"
- Section 53.12, "Configuring a Proxy Server"
- Section 53.13, "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/technetwork/middleware/webcenter/ps3-samples-176806.html

53.1 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](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 application. The native iPhone client is written in Objective-C, and the REST APIs enable the client to send and retrieve application data.

### 53.2 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.

### 53.3 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 53.5, "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 53.11, "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.
53.4 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 data. In general, the WebCenter REST commands provide a more natural and easy-to-use alternative to a SOAP-style Web services approach.

Table 53–1 describes the Oracle WebCenter Portal REST APIs provided for WebCenter Portal tools and services features.

---

Note: Portal Framework applications only support the REST API for Navigation. All other REST APIs are supported only by the WebCenter Portal application.

---

<table>
<thead>
<tr>
<th>REST API</th>
<th>Description</th>
<th>Chapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discussions</td>
<td>Enable a client to post, read, update, and delete discussion forums, topics, and messages.</td>
<td>Section 33.3.8, &quot;Using the Discussions REST API.&quot;</td>
</tr>
<tr>
<td>Lists</td>
<td>Enables a client to browse all the lists associated with a named portal; search list columns given a search term; create new lists; add, update, and remove list rows; and similar sorts of list-related tasks</td>
<td>Section 49.4, &quot;Using the Lists REST API.&quot;</td>
</tr>
<tr>
<td>People Connections</td>
<td>Enable a client to view profile data; manage connection lists, feedback, and messages; create new activities and view activities for users, lists, and portals.</td>
<td>Chapter 42, &quot;Using the People Connections REST APIs.&quot;</td>
</tr>
<tr>
<td>WebCenter Portal</td>
<td>Enable a client to retrieve portal metadata and view, create, update, and delete portal lists and list items. You can also retrieve portal membership information.</td>
<td>Section 56.2, &quot;Using the WebCenter Portal REST API.&quot;</td>
</tr>
<tr>
<td>Content Management</td>
<td>Uses the CMIS (Content Management Interoperability Services) RESTful server binding to provide access to the CM VCR (Content Management Virtual Content Repository).</td>
<td>Chapter 31, &quot;Content Management REST API.&quot;</td>
</tr>
<tr>
<td>Activity Graph</td>
<td>Enables you to retrieve recommendations for connections, portals, and items using the underlying Activity Graph engine.</td>
<td>Section 46.3.2, &quot;Using the Activity Graph REST APIs.&quot;</td>
</tr>
<tr>
<td>Events</td>
<td>Lets you access calendar events associated with a named portal.</td>
<td>Section 48.3, &quot;Using the Events REST API.&quot;</td>
</tr>
<tr>
<td>Feedback</td>
<td>Enables a client to create, read, and delete feedback in a social networking application.</td>
<td>Chapter 42.3, &quot;Feedback REST API.&quot;</td>
</tr>
<tr>
<td>Search</td>
<td>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 &quot;smith&quot; in all portals, documents and wiki pages.</td>
<td>Section 45.3.4, &quot;Using the Search REST APIs.&quot;</td>
</tr>
<tr>
<td>Tags</td>
<td>Enables a client to read, post, update, and delete tags and tagged items.</td>
<td>Section 44.3.2, &quot;Using the Tags REST APIs.&quot;</td>
</tr>
</tbody>
</table>
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 the 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.

### Table 53–1 (Cont.) Summary of WebCenter Tools and Services Features Supported by REST APIs

<table>
<thead>
<tr>
<th>REST API</th>
<th>Description</th>
<th>Chapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Navigation</td>
<td>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.</td>
<td>Section 10.11.2, &quot;Using the Navigation REST APIs.&quot;</td>
</tr>
<tr>
<td>Personalization</td>
<td>Enables you to access the Personalization Conductor.</td>
<td>Chapter 72, &quot;Conductor API Reference.&quot;</td>
</tr>
<tr>
<td>Pagelets</td>
<td>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.</td>
<td>Section 62.7.2.2, &quot;Accessing Pagelets Using REST.&quot;</td>
</tr>
</tbody>
</table>

Note: XSD files for the following URNs:

- urn:oracle:webcenter:activities:stream -> activitystream.xsd
- urn:oracle:webcenter:messageBoard -> wall.xsd
- urn:oracle:webcenter:people -> people.xsd
- urn:oracle:webcenter:people:invitations -> people.xsd
- urn:oracle:webcenter:people:person -> people.xsd
- urn:oracle:webcenter:search:results -> search.xsd
- urn:oracle:webcenter:spaces -> spaces.xsd

can be found in your <WCP_ORACLE_HOME>/webcenter/schemas/directory.
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 53.5.1, "Using the Resource Index"
- Section 53.5.2, "Anatomy of a Link"

### 53.5.1 Using 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.

**Tip:** A REST client is helpful for generating custom REST requests. For example, a Firefox RESTClient add-on is available at:

http://restclient.net/

Other similar REST clients can also be easily obtained.

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 53.9, "Security Considerations for CMIS REST APIs,” and Section 53.8, "Security Considerations for WebCenter Portal 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 53–1** 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 53–1  Response to a GET on the Resource Index**

```json
{
   'resourceType': 'urn:oracle:webcenter:resourceindex',
   'links': [
      {  
```
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 53–2 shows a method that locates a URI given the Resource Index JSON data.
Example 53–2  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;
    }
  }
}

53.5.2 Anatomy of a Link
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.

Example 53–3 and Example 53–4 show the anatomy of a hypermedia link as XML and
in JSON document fragments, respectively.

Example 53–3  Link in an XML Document Fragment
<link>
  <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...
</link>

Example 53–4  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...
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) as shown in Example 53–5 and Example 53–6.

Example 53–5  XML Naming Convention

```xml
<myElement>text</myElement>
```

Example 53–6  JSON Naming Convention

```json
{"myElement": "text"}
```

This section includes the following subsections that describe the different attributes of the hypermedia link:

- Section 53.5.2.1, "Resource Type"
- Section 53.5.2.2, "Relationship"
- Section 53.5.2.3, "Capabilities"
- Section 53.5.2.4, "Media Type"
- Section 53.5.2.5, "Templates"

### 53.5.2.1 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 53.7, "Navigating Hypermedia Using HTTP."

### 53.5.2.2 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.

### 53.5.2.3 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.

---

### 53.5.2.4 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 the CMIS REST API, see Chapter 31, "Content Management REST API."

---

### 53.5.2.5 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 53–7** shows a URI template including several request query parameters.
Example 53–7  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 53–7, to see 10 list items (default) 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 53–8.

Example 53–8  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 supply 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 URL 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 53–9.

Example 53–9  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):
  
  ```
  [\[\[\[\text{field1}[:\text{operand}]:\text{value1}\]\]::\[\text{field2}[:\text{operand}]:\text{value2}\]]
  ```

  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 \textit{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.

- \textit{projection} - Reserved for implementation-specific projection of model representations, such as variable recursion depth, field or attribute filtering. Valid values are \textit{summary} or \textit{details}.

  For example, requesting a projection of \textit{summary} for a collection of lists, returns only the title, description, and hypermedia links. Requesting a projection of \textit{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.

  \url{http://host:port/...lists&projection=details}

- \textit{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 \textit{projection} and \textit{data} query string parameters are present, the \textit{data} parameter will be used to determine which data to return. If you specify the constant \textit{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.

### 53.6 Understanding Items Hypermedia

A collection of \textit{items} makes up the actual content of responses. This is at the same level as the \textit{links} section described previously. Each item (including the top-level tag in each response) has one common attribute (\textit{resourceType}) in addition to resource-specific content and format. For details beyond the \textit{resourceType}, see the chapter for the specific service.

See Example 53–13 for an example that describes a collection of entities/items. Example 53–15 describes a single entity/item.

### 53.7 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 53–2 describes the general pattern followed when constructing opaque resource URIs. The \textit{resourceType} differentiates whether the HTTP method operates on a collection of resources or an individual resource.
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 53–3 describes the potential response status codes.

**Table 53–3  HTTP Response Status Codes**

<table>
<thead>
<tr>
<th>HTTP Response Status Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>OK. Upon successful completion of a GET or PUT. This is accompanied by a resource representation as a response body.</td>
</tr>
<tr>
<td>201</td>
<td>Created. Upon successful completion of a POST. This has a location header to the newly-created resource.</td>
</tr>
<tr>
<td>204</td>
<td>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.</td>
</tr>
<tr>
<td>400</td>
<td>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.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. Client may retry by submitting credentials. This may be accompanied with a fault response body to help diagnose the issue.</td>
</tr>
<tr>
<td>403</td>
<td>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.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. Referencing a specific resource with an ID, but that resource does not exist.</td>
</tr>
<tr>
<td>405</td>
<td>Method Not Allowed. This includes a list of valid methods for the requested resource.</td>
</tr>
<tr>
<td>406</td>
<td>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.</td>
</tr>
</tbody>
</table>
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.

<table>
<thead>
<tr>
<th>HTTP Response Status Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>409</td>
<td>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.</td>
</tr>
<tr>
<td>422</td>
<td>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.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. The server encountered an unexpected condition that prevented it from fulfilling the request.</td>
</tr>
<tr>
<td>501</td>
<td>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.</td>
</tr>
</tbody>
</table>

53.8 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 53.9, "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 "Configuring SSL" chapter in Administering 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 "Configuring SSL" chapter in Administering 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:

```xml
<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" />
```
WebCenter Portal REST APIs operate under the identity of the authenticated user. For example, the portal REST APIs only return information for, and allow changes to, portals to which the user has access.

53.9 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 Chapter 31, “Content Management REST API.”

53.10 Understanding Common Types

This section describes the common types that are shared by multiple WebCenter Portal REST APIs.

- Section 53.10.1, "Common Types"
- Section 53.10.2, "Portable Contact Types"

53.10.1 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 53.10.1.1, "personReference"
- Section 53.10.1.2, "groupSpaceReference"

53.10.1.1 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.

### 53.10.1.2 `groupSpaceReference`

This is a generic data type that represents a portal. It is used by several APIs, for example in the activity stream, to identify a portal in which a particular activity occurred. It is made up on the following elements:

- `guid` - The GUID of the portal
- `name` - The name of the portal
- `displayName` - The display name of the portal

The `groupSpaceReference` also includes an html link, rest link, and icon link.

### 53.10.2 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 53.10.2.1, "name Portable Contact Type"
- **Section 53.10.2.2, "address Portable Contact Type"
- **Section 53.10.2.3, "organization Portable Contact Type"
- **Section 53.10.2.4, "value Portable Contact Type"

#### 53.10.2.1 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.

#### 53.10.2.2 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:
Managing Caches

- 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.

53.10.2.3 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.

53.10.2.4 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

53.11 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, refer to the Hypertext Transfer Protocol specification:

http://www.w3.org/Protocols/

53.12 Configuring a Proxy Server

This section explains how to set up a simple, response-rewriting reverse HTTP proxy on an Apache 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.

Note: This section illustrates a simple example of setting up a proxy server on Apache. For more detailed information, refer to the Apache Server documentation available at http://httpd.apache.org/docs. You can also use Oracle HTTP Server (OHS) for your proxy server. For more information, see Administrator’s Guide for Oracle HTTP Server.

The basic steps for setting up a proxy server on Apache are:

1. Obtain access to an Apache server. Oracle recommends Apache 2.2.7 or a later version.

2. Make sure the server has the mod_substitute and mod_proxy modules installed. Note that Apache versions 2.2.7 and later include mod_substitute by default. It is also possible to use mod_sed or mod_line_edit, however these configurations are not supported by Oracle.

3. Open the httpd.conf or the virtual host configuration file, and add the following lines, substituting your server name/information where appropriate:

```
ProxyRequests          Off
LoadModule             substitute_module       modules/mod_substitute.so
SetOutputFilter        SUBSTITUTE
ProxyPass              /rest/api/              http://myhost:8888/rest/api/
ProxyPassReverse       /rest/api/              http://myhost:8888/rest/api/
Substitute             s|myhost|yourhost
ProxyPass              /pathname/rest/api/     http://myhost:8888/rest/api/
Substitute             s|myhost:8888/rest/api|yourhost/pathname/rest/api|n
```

Note: Two servers are being proxied in this example scenario. Note that the following two calls are actually talking to these two different servers, but they appear to clients to be the same server host:

http://myhost/rest/api/resourceIndex

http://myhost/pathname/rest/api/resourceIndex

4. Restart the Apache server. For example, on Linux, you could do this:
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 53.13.1, "Navigating the Message Board Hypermedia"
- Section 53.13.2, "Displaying Activity Stream Data"
- Section 53.13.3, "Updating User Status"

53.13.1 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, delete unwanted messages, and filter message board messages.

This section includes the following subsections:

- Section 53.13.1.1, "Accessing the Resource Index"
- Section 53.13.1.2, "Reading Messages"
- Section 53.13.1.3, "Creating a New Message"
- Section 53.13.1.4, "Updating a Message"
- Section 53.13.1.5, "Deleting a Message"
- Section 53.13.1.6, "Filtering Messages"

53.13.1.1 Accessing the Resource Index

The first step is always to access the Resource Index (Example 53–10).

Example 53–10  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 53–11).

Example 53–11  Response to Accessing the Resource Index

200 OK
Accept: application/json;charset=UTF-8

{
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.

### 53.13.1.2 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 53–12).

To read messages on a message board, you must be logged in.

**Example 53–12 Retrieving Messages from Your Message Board (GET)**

GET `/opaque-messageBoard-URI`

The response provides information about all the messages on your message board (Example 53–13).

**Example 53–13 Response to Retrieving Messages from Your Message Board**

200 OK
Accept: application/json; charset=UTF-8

```json
{
  "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"
    }
  ]
}
```

```json
{  
    "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"
      }
    ]
}
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

### 53.13.1.3 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 53–14).

#### Example 53–14 Creating a Message on Another User's Message Board (POST)

```plaintext
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 53–15).

#### Example 53–15 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",
            "rel": "self",
            "href": "opaque-message-URI"
        },
        "author": {
            "id": "mike",
            "displayName": "mike",
            "guid": "649657609D5C11DEBFAA799CBD41D9B8",
            "links": [
```
53.13.1.4 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 53–16).

Example 53–16  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 53–17).

Example 53–17  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."
}
```
53.13.1.5 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 53–18).

**Example 53–18  Deleting a Message (DELETE)**

```
DELETE opaque:message:URI
```

The response is simply a status code of 204 (Example 53–19).

**Example 53–19  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.

---

53.13.1.6 Filtering Messages

Messages can be filtered (using the HTTP verbs GET, POST, and PUT) based on visibility criteria. Messages posted on message boards falls into the following visibility categories:

- Public
- Private
- Hidden
- Public and hidden
- Private and hidden

**Public vs Private Messages**

The owner of a message board can mark any message as private. When a message is marked as private, that message is not visible to anyone other than the owner of the message. By default, all messages are public.

Messages can also be sent as private messages. If Mike, for example, is viewing Carl's message board, only public messages (including those sent or received as private) will be visible to user Mike.

**Hidden vs Non-hidden Messages**

The owner of a message board can mark any message as hidden. When a message is marked as hidden, that message will not be visible to the message board's owner, but will remain visible to anyone else viewing that person's message board.

The following examples show how to retrieve,

**Example 53–20  Retrieving Filtered Messages (GET)**

```
me
```

```
all
  rest/api/messageBoards/person/@me
```
WebCenter Portal's REST API Examples

private
rest/api/messageBoards/person/@me/private
public
rest/api/messageBoards/person/@me/public
hidden and public
rest/api/messageBoards/person/@me/hidden
private and hidden
rest/api/messageBoards/person/@me/private_hidden

- **person**
  rest/api/messageBoards/person/<GUID>

  Note that the GUID of the logged in user for whom to retrieve messages is required.

- **space-guid**
  rest/api/messageBoards/space/<GUID>

  Note that visibility-based filtering is not available for space-guid.

**Example 53–21  Using Filtering for New Messages (POST)**

- **me**

  all
  rest/api/messageBoards/person/@me
  private
  
  ```
  {"body" : "<BODY_CONTENT>" , "visibilityType" : 'private'}
  ```
  public
  
  ```
  {"body" : "<BODY_CONTENT>" , "visibilityType" : 'public'}
  ```
  hidden and public
  
  ```
  {"body" : "<BODY_CONTENT>" , "visibilityType" : 'hidden'}
  ```
  private and hidden
  
  ```
  {"body" : "<BODY_CONTENT>" , "visibilityType" : 'private_hidden'}
  ```

- **person**

  rest/api/messageBoards/person/<GUID>

  For a GUID other than that of "me":

  public
  
  ```
  {"body" : "<BODY_CONTENT>" , "visibilityType" : 'public'}
  ```
  private
  
  ```
  {"body" : "<BODY_CONTENT>" , "visibilityType" : 'private'}
  ```

- **space-guid**

  rest/api/messageBoards/space/<GUID>

  Note that visibility-based filtering is not available for space-guid.

**Example 53–22  Using Filtering for Modified Messages (PUT)**

- **me**

  all
  rest/api/messageBoards/person/@me/messages/<msg guid>
  private
  
  ```
  {"body" : "<BODY_CONTENT>" , "visibilityType" : 'private'}
  ```
  public
Using Oracle WebCenter Portal REST APIs

53.13.2 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/technetwork/middleware/webcenter/ps3-samples-176806.html

In Example 53–23:

- 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 53–23  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;

53.13.3 Updating User Status

The following example shows how to use REST APIs to update a user's WebCenter Portal 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/technetwork/middleware/webcenter/ps3-samples-176806.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 53–24) 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 53–24  HTML Body**

```html
<body>
    <div>New status message:&nbsp;<input id="statusMessage" type="text" onkeyup="if (event.keyCode==13) updateStatus();" maxlength="250" size='60' />
    </div>
    <button id="button1" onclick="updateStatus();">Update Status</button>
</div>
<div id='statusResults'></div>
</body>
```

Figure 53–1 shows the HTML page.
The code in Example 53–25 retrieves the Resource Index (resourceIndexURL variable set to /rest/api/resourceIndex). The Resource Index is returned as an AJAX request object (resourceIndexRequest).

**Example 53–25 Retrieving the Resource Index**

```javascript
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 53–26 traverses the links in the data returned by a REST call to find a specified string (URN) that identifies the required resource type.

**Example 53–26 Traversing the Links**

```javascript
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 53–27 uses getResourceURL to parse the Resource Index to find the user profile URL. The data is returned as an AJAX request object (profileRequest).

**Example 53–27 Retrieving the User Profile**

```javascript
function resourceIndexCallback(data) {
    // get my user profile
    profileRequest.get(
        getResourceURL(data, 'urn:oracle:webcenter:people'),
        profileCallback,
        clearUIBusy);
}
```

Example 53–28 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 53–28 Retrieving the Status Object**

```javascript
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, Esc(""));

// 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 53–29, renders the new status object.

Example 53–29  Rendering the New Status Object

function renderStatusPutResults(data) {
    var name = profile.displayName;
    // get the URL for my profile HTML page in WebCenter Portal
    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 53–30 shows the code for disabling and reenabling the UI.

Example 53–30  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 53–31 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 53–31  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 XMLHttpRequest();

        // open a request
        xhr.open(method, url, async);

        // send the request
        xhr.send(data);

        // handle success or failure
        xhr.onload = function () {
            if (xhr.status >= 200 && xhr.status < 300) {
                success(
                    JSON.parse(xhr.responseText),
                    function (error) {
                        failure(error);
                    });
            }
       );

        // handle errors
        xhr.onerror = function () {
            failure(xhr.statusText);
        };
    }
}

Using Oracle WebCenter Portal REST APIs 53-29
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 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.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 {
            errorCallback(xhr.responseText);
        }
    }
}
// we got an error. Format an error string
data = 'Error: ' + xhr.status + ' ' + xhr.statusText + '<br />'
   + xhr.responseText;
errorCallback(data);
}
This chapter describes how you can integrate other Oracle applications with your WebCenter Portal or Portal Framework application.

This chapter contains the following topics:

- Section 54.1, "Integrating Other Oracle Applications in WebCenter Portal"
- Section 54.2, "Integrating Siebel Applications"
- Section 54.3, "Integrating E-Business Suite Applications"
- Section 54.4, "Integrating JD Edwards Applications"
- Section 54.5, "Integrating PeopleSoft Applications"
- Section 54.6, "Integrating Oracle Business Intelligence Presentation Services"

54.1 Integrating Other Oracle 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 54–1 shows the technologies involved in WebCenter Portal integration with other Oracle applications.
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 54–1 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.

54.2 Integrating Siebel Applications

This section describes how to integrate a Siebel Web service in a Portal Framework application. It also describes how to integrate Siebel objects using the Siebel Web Engine (SWE). Siebel and WebCenter can work together to include Siebel’s CRM capabilities as portlets within your Portal Framework application. You can integrate Siebel applications as Web services, or using the Siebel Web Engine (SWE) as described in the following subsections:

- Section 54.2.1, "Integrating Siebel Applications as Web Services"
- Section 54.2.2, "Integrating Siebel Applications Using the Siebel Web Engine"

54.2.1 Integrating Siebel Applications as Web Services

This section describes how to integrate Siebel applications as Web services in a portal or Portal Framework application.
This section contains the following subsections:

- Section 54.2.1.1, "Preparing the Siebel Application"
- Section 54.2.1.2, "Consuming a Siebel Web Service in a Framework Application"
- Section 54.2.1.3, "Consuming a Siebel Web Service Data Control in a Portal"

54.2.1.1 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 portal or Framework application.

This section contains the following subsections:

- Section 54.2.1.1.1, "Creating an Inbound Web Service"
- Section 54.2.1.1.2, "Creating Operations for the Inbound Web Service"

54.2.1.1.1 Creating an Inbound Web Service

To create an inbound Web service:

1. Log into the Siebel application as an administrator.
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:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Namespace</td>
<td><a href="http://xmlns.oracle.com">http://xmlns.oracle.com</a></td>
</tr>
<tr>
<td>Name</td>
<td>Siebel Customer Account</td>
</tr>
<tr>
<td>Status</td>
<td>Active</td>
</tr>
<tr>
<td>Comment</td>
<td>For Fusion Middleware</td>
</tr>
</tbody>
</table>

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:

14. From the **Menu** drop-down list, select **Save Record**.

### 54.2.1.1.2 Creating Operations for the Inbound Web Service

After creating the inbound Web service (see **Section 54.2.1.1.1**, "Creating an Inbound Web Service"), continue by adding operations to the inbound Web service and then create a WSDL file, follow these 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:

<table>
<thead>
<tr>
<th>Authentication Type</th>
<th>Session Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>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.</td>
</tr>
<tr>
<td>Username and password</td>
<td>None</td>
<td>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.</td>
</tr>
<tr>
<td>Username and password</td>
<td>Stateless</td>
<td>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.</td>
</tr>
<tr>
<td>Session token (stateless)</td>
<td>Stateless</td>
<td>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.</td>
</tr>
<tr>
<td>Session token (stateless)</td>
<td>None</td>
<td>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.</td>
</tr>
</tbody>
</table>

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**.

54.2.1.2 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 Portal 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 54.2.1.2.1, "Creating a Data Control Based on a Siebel Web Service"**
- **Section 54.2.1.2.2, "Creating a Page Flow Diagram"**
- **Section 54.2.1.2.3, "Adding Pages and Navigation to the Page Flow Diagram"**
- **Section 54.2.1.2.4, "Creating the Query Page"**
- **Section 54.2.1.2.5, "Creating the Results Page"**
- **Section 54.2.1.2.6, "Testing the Application"**

54.2.1.2.1 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 Portal Framework application or create a new Portal 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.

54.2.1.2.2 Creating a Page Flow Diagram

This section describes how to create the page flow diagram to which you can 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.

54.2.1.2.3 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.

54.2.1.2.4 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
   - Customizable 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:* 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.
54.2.1.2.5 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
   - JSP 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.

54.2.1.2.6 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 QueryByld 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 Row # field and click Submit.
   The Details page displays the detail information for the corresponding account.

54.2.1.3 Consuming a Siebel Web Service Data Control in a Portal

This section describes how you can create a Web Service data control and add it to a portal page. The steps in this section assume that you have prepared the application and generated a WSDL as described in Section 54.2.1.1, "Preparing the Siebel Application."

---

Note: Before you can add a data control or task flow containing a data control to a portal page you must first have configured WS-Security for WebCenter Portal. For more information about configuring WS-Security, see the "Configuring WS-Security" chapter in Administering Oracle WebCenter Portal.

---

For more information about creating a Web service data control, see the "Creating a Web Service Data Control" section in Building Portals with Oracle WebCenter Portal. For information about Web service data controls, see also the "Web Service Data Controls" section in Building Portals with Oracle WebCenter Portal.

To create a Web service data control:

1. In WebCenter Portal or the portal in which you want to create the data control, go to either the Shared Assets or Assets page.
2. Select Data Controls and click Create.
   The Create New Data Control dialog displays (see Figure 54–33).

Figure 54–2 Create New Data Control Dialog
3. In the Create New Data Control dialog, enter a **Name** and **Description** for the data control, select **Web Service** as the **Data Control Type**, and then click **Continue**.

4. Enter the WSDL URL that you generated in **Section 54.3.4.1, “Generating the WSDL”** and other details for the data control and click **Continue**.

5. Click **Show Methods**.

6. Select the method(s) to make available and click **Next**.

7. Enter the parameter default values, if any, and click **Create**.

8. To make the data control available, from the Shared Assets or Assets page, select **Task Flows**. The Create New Task Flow dialog displays (see **Figure 54–35**).

**Figure 54–3  Create New Task Flow Dialog**

9. Enter the task flow **Name** and **Description**, select the **Mashup Style** to use click **Create** to create the task flow.

10. Select the task flow and click the **Edit** icon.

11. Add the data control (with parameter form) as a table onto the task flow and verify the data.

12. To make the task flow available, navigate to **Administration > Business Role Pages**.

13. Select **Business Role Page** and click the **Create** icon.

14. Edit the page and save the changes.

15. Drop the task flow onto the page and verify the data.

**54.2.2 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 Portal 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 Portal Framework application.

For instructions on how to use SWE within a Portal 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.

54.3 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 54.3.1, "Introduction to Integrating EBS Applications"
- Section 54.3.2, "Required Configurations for Integrating EBS"
- Section 54.3.3, "Integrating EBS Applications as WSRP Portlets"
- Section 54.3.4, "Integrating EBS Applications as Data Controls in WebCenter Portal"

54.3.1 Introduction to Integrating EBS Applications

This section describes the integration points and requirements integrating Oracle E-Business Suite portlets in portals and Framework applications.

This section includes the following subsections:

- Section 54.3.1.1, "Understanding EBS Integration"
- Section 54.3.1.2, "Requirements for Integrating EBS Applications"

54.3.1.1 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 portals or Framework applications, by simply adding the portlet onto an application page. Follow the instructions in Section 54.3.3.2.3, "Creating a JSF Page to Consume the Remote Producer" to add them to a Framework application page, or Section 54.3.3.3.3, "Adding the EBS Portlet to a Portal Page" to add them to a portal page.

You can also create new E-Business Suite portlets that are WSRP and JSR 168-compliant that can similarly be added to a Portal Framework application. Creating and consuming WSRP and JSR 168 compliant portals in WebCenter Portal is described in Section 54.3.3.2, "Integrating EBS Applications in a Framework Application" and Section 54.3.3.3, "Integrating EBS Applications in a Portal."

54.3.1.2 Requirements for Integrating EBS Applications

The following requirements apply for integrating Oracle E-Business Suite portals into Oracle 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 can 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.

E-Business Suite can also be configured to OID without OAM or OSSO. For more information, see Section 54.3.2.1, "Preparing OID for Use Without Single Sign-On."

---

**Caution:** Both WebCenter and Oracle E-Business Suite must share the same OID instance and user IDs.

---

**Note:** Although Oracle E-Business Suite can be configured to use Oracle Internet Directory (OID) without single sign-on, this is not a recommended approach as users will be prompted for credentials each time they move to or from the integrated portal or data control.

---

You must have granted the portal or Framework 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.

### 54.3.2 Required Configurations for Integrating EBS

This section contains configurations that should be undertaken prior to attempting to integrate portal or data controls into your portal or Framework application.

This section contains the following subsections:

- Section 54.3.2.1, "Preparing OID for Use Without Single Sign-On"
- Section 54.3.2.2, "Creating a User in EBS and Assigning a Responsibility"
- Section 54.3.2.3, "Configuring the EBS Applications Profile Options"
- Section 54.3.2.4, "Adding the WebCenter Host as a Trusted Portal Using AutoConfig"

#### 54.3.2.1 Preparing OID for Use Without Single Sign-On

This section describes the steps to configure OID as an optional standalone environment without using either OAM or OSSO. Note that this is not a recommended approach as users will be prompted for credentials each time they move to or from an integrated portal or data control. If you have installed an SSO solution, continue with Section 54.3.2.2, "Creating a User in EBS and Assigning a Responsibility."
Integrating E-Business Suite Applications

Caution: Both WebCenter and Oracle E-Business Suite must share the same OID instance and the same user IDs.

1. Register the OID instance on the host server by following the steps below:
   a. Run the following command:
   
   ```
   $FND_TOP/bin/txkrun.pl -script=SetSSOReg -registerinstance=yes
   ```
   b. Supply the required information at the following prompts:
   
   Enter the host name where the Oracle iAS Infrastructure database is installed? <Enter the OID Host>
   Enter the LDAP Port of the Oracle Internet Directory server? <Enter the LDAP Port>
   Enter SSL LDAP Port of the Oracle Internet Directory server? <Enter the LDAP SSL Port>
   Enter the Oracle Internet Directory Administrator (orcladmin) Bind password? <Password>
   Enter Oracle E-Business apps database user password? <Password>
   
   c. Restart all the services. To do this, navigate to $ADMIN_SCRIPTS_HOME and run the following script to stop all services:
   
   ```
   ./adstpall.sh apps/apps
   ```
   and then run the following script to start all services:
   
   ```
   ./adstrtal.sh apps/apps
   ```

2. Register OID by following the steps below:
   a. Run the following command:
   
   ```
   $FND_TOP/bin/txkrun.pl -script=SetSSOReg -registeroid=yes
   ```
   b. Supply the required information at the following prompts:
   
   Enter LDAP Host name? <Enter the OID Host>
   Enter the LDAP Port on Oracle Internet Directory server? <Enter the LDAP Port>
   Enter the Oracle Internet Directory Administrator (orcladmin) Bind password? <Password>
   Enter the instance password that you would like to register this application instance with? <Password>
   Enter Oracle E-Business apps database user password? <Password>
   
   c. Restart all the services. To do this, navigate to $ADMIN_SCRIPTS_HOME and run the following script to stop all services:
   
   ```
   ./adstpall.sh apps/apps
   ```
   and then run the following script to start all services:
   
   ```
   ./adstrtal.sh apps/apps
   ```

3. Continue by configuring EBS profile options as shown in Section 54.3.2.3, "Configuring the EBS Applications Profile Options."

54.3.2.2 Creating a User in EBS and Assigning a Responsibility

For integration with EBS to work, WebCenter and EBS must have a common OID identity store. With a common OID, you can either create a new user in EBS, or use an existing user in OID, and then assign a responsibility to that user. This will ensure that
the user has access to the portlets in WebCenter. Else the register producer will not have any portlets if we do not assign a responsibility.

To create a new user and assign a responsibility:

1. Log into EBS as a system administrator if not already logged in.
2. In the Navigation pane, expand the System Administrator node, expand Security, expand User, and then click Define.

The Users window displays (Figure 54–4).

**Figure 54–4 Users Window**

3. Enter the User Name, and Password. The Password Expiration options should be set to None.
4. Open the Direct Responsibilities tab, and search for the Responsibility to add and assign the Application to associate with it (for example, search for Preferences SSWA and assign Oracle iProcurement to it), and then click Save (see Figure 54–5).
5. To confirm, log in with the newly created user and check that the application associated with the Responsibility is listed.

6. Continue by configuring the EBS Application Profile Options as described in Section 54.3.2.3, "Configuring the EBS Applications Profile Options."

**54.3.2.3 Configuring the EBS Applications Profile Options**

This section describes how to configure EBS Applications Profile Options and is a requirement for both SSO and non-SSO configurations.

To configure the EBS profile options:

1. Log into EBS as a system administrator.

2. In the Navigation pane, expand the System Administrator node, and then click **Define Profile Options**.

3. Close the Profiles window.

4. In the Navigator, select **Profile System Values** and click **Open**.

   The Find System Profile Values window displays (see Figure 54–6)
5. Enter the **Profile** name to update and click **Find**.

The System Profile Values window displays (see Figure 54–7).

**Figure 54–6  Find System Profile Values Window**

![Find System Profile Values Window](image)

Update the values for the following profiles, saving your entries after each update:

- **Applications SSO Enable OID Identity Add Event** = **Enabled**
- **Applications SSO Login Types** = **Both**
- **Application SSO LDAP Synchronization** = **Enabled**
- **Applications SSO Type** = **SSWA w/ SSO**
- **Link Applications user with OID user with same username** = **Enabled**

6. Restart all the services. To do this, navigate to `$ADMIN_SCRIPTS_HOME` and run the following script to stop all services:

```
./adstpall.sh apps/apps
```

and then run the following script to start all services:
Integrating Other Oracle Applications

54.3.2.4 Adding the WebCenter Host as a Trusted Portal Using AutoConfig

The EBS WSDL is protected and before you can access it you must first add an entry for the consuming WebCenter instance’s host using the EBS AutoConfig tool. Note that without this configuration step you will get a "403 Forbidden" error if you try to access the WSDL.

To add the WebCenter host as a trusted portal:

1. Log into EBS as a system administrator if you are not already logged in.
2. In the Navigation pane, expand the System Administrator node, expand Oracle Applications Manager, and then click Workflow.
3. Open the Sitemap tab and click AutoConfig.
4. In the Edit Parameter column, click the Edit icon in the Applications Tier row.
5. Open the System tab and expand the oa_web_server node.
6. In the list of nodes, look for any that have access to Portlet Producer URLs, add the WebCenter Host and click Save. If you need to add multiple host name, add them separated by space.
7. Run the autoconfig script entering apps as the password when prompted:
   cd $ADMIN_SCRIPTS_HOME
   ./adautocfg.sh
8. Restart all the services. To do this, navigate to $ADMIN_SCRIPTS_HOME and run the following script to stop all services:
   ./adstpall.sh apps/apps
   and then run the following script to start all services:
   ./adstrtal.sh apps/apps

54.3.3 Integrating EBS Applications as WSRP Portlets

This section describes how to integrate EBS regions as WSRP portlets in a portal or Framework application. To start, you’ll need to generate the portlet for the region using the Portlet Generator, and then continue by registering the producer and integrating it in your application.

This section contains the following subsections:

- Section 54.3.3.1, "Preparing the EBS Portlet for Remote Access"
- Section 54.3.3.2, "Integrating EBS Applications in a Framework Application"
- Section 54.3.3.3, "Integrating EBS Applications in a Portal"

54.3.3.1 Preparing the EBS Portlet for Remote Access

Oracle E-Business Suite provides a tool called Portlet Generator to convert existing standalone Oracle Application Framework regions into portlets. To be available for portletization, a region must have the following properties.
- Regions must have an Application Module (AM) defined and must have its standalone property set to true.
- Inline regions must have an AM defined and have its standalone property set to true.
- Content regions must have an AM defined (content regions do not have a standalone property)

To expose EBS functionality as a portlet using Portlet Generator:

1. Log into EBS as a system administrator.
2. In the Navigation pane, expand the Functional Administrator node, and then click Home.
   The Application Administration page displays (see Figure 54–8).

**Figure 54–8  Application Administration Page**

![Application Administration Page](image)

3. Open the Portletization tab and click the Search icon for the Application Short Name field (or enter the Application Short Name if you know it).
4. Select the Search By criteria (for example, select Application Name and enter Oracle iProcurement) and search for the functionality to portletize.

5. Select the row returned in the search results and click Select.

6. Click Go to list the EBS functionality that can be portletized.

7. Click the Portletize icon for the functionality you want to expose (for example, AdvisoryWarningRN).

The Create Portlet dialog displays (see Figure 54–10).

8. Enter the Responsibility to associate the region with (for example, Preferences SSA) or use the Search function.

9. Click Apply.
10. Continue by registering the EBS producer and integrating it in a portal (see Section 54.3.3.2, "Integrating EBS Applications in a Framework Application") or Framework application (see Section 54.3.3.3, "Integrating EBS Applications in a Portal").

54.3.3.2 Integrating EBS Applications in a Framework Application
This section describes how to integrate an EBS portlet in a Framework application. This section includes the following subsections:

- Section 54.3.3.2.1, "Preparing the EBS Portlet for Remote Access"
- Section 54.3.3.2.2, "Registering the EBS WSRP Producer in the Framework Application"
- Section 54.3.3.2.3, "Creating a JSF Page to Consume the Remote Producer"
- Section 54.3.3.2.4, "Testing the Framework Application"

54.3.3.2.1 Preparing the EBS Portlet for Remote Access
Prepare the standalone regions to be portletized as described in the section on Section 54.3.3.1, "Preparing the EBS Portlet for Remote Access."

Before adding the portlets in your Framework application, be sure to bounce the Apache listener as the menu and function definitions are cached.

54.3.3.2.2 Registering the EBS WSRP Producer in the Framework Application
Follow the instructions below to create or open an existing Portal Framework application, and register the EBS WSRP producer.

1. Create a new Portal 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 54.3.3.2.3, "Creating a JSF Page to Consume the Remote Producer."

54.3.3.2.3 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 Portal 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 Portal Framework application appear in the EBS application as described in Section 54.3.3.2.4, "Testing the Framework Application."

54.3.3.2.4 Testing the Framework Application
Use this procedure to test the Portal Framework application by modifying content on the JSF page in the Portal 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.

### 54.3.3.3 Integrating EBS Applications in a Portal

This section contains the following subsections:

- **Section 54.3.3.3.1**, "Preparing the EBS Portlet for Remote Access"
- **Section 54.3.3.3.2**, "Registering the EBS WSRP Producer in WebCenter Portal"
- **Section 54.3.3.3.3**, "Adding the EBS Portlet to a Portal Page"
- **Section 54.3.3.3.4**, "Testing the Portal Portlet Connection"

54.3.3.3.1 Preparing the EBS Portlet for Remote Access
Prepare the standalone regions to be portletized as described in the section on **Section 54.3.3.1**, "Preparing the EBS Portlet for Remote Access."

Before adding the portlets in WebCenter Portal, be sure to bounce the Apache listener as the menu and function definitions are cached.

54.3.3.3.2 Registering the EBS WSRP Producer in WebCenter Portal
You can register the EBS WSRP producer directly in WebCenter Portal, as described in the "Registering Portlet Producers with the Administration Console" section in *Administering Oracle WebCenter*
To register the EBS WSRP producer using Fusion Middleware Control:

1. Prepare the EBS page that you want to consume in WebCenter Portal for remote access as described in Section 54.3.3.3.1, "Preparing the EBS Portlet for Remote Access."
2. Log into Fusion Middleware Control for the WebCenter Portal domain (we_domain by default).
   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 portal page as described in Section 54.3.3.3.3, "Adding the EBS Portlet to a Portal Page."

54.3.3.3.3 Adding the EBS Portlet to a Portal Page Use the following procedure to consume the EBS remote producer in a portal page.

1. Go to the page, or create a new page, where you want to add the EBS portlet.
2. Open the page in edit mode, as explained in the "Opening a Page in the Page Editor (Composer)" section in Building Portals with Oracle WebCenter Portal.
3. In the resource catalog, select UI Components and then select Portlets.
   Note: 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 "Working with Resource Catalogs" chapter in Building Portals with Oracle WebCenter Portal.
4. Click Open to open the Portlets catalog.
5. Click the portlet you added in Fusion Middleware Control.
6. Click Add for the EBS portlet you want to add to your portal page. You can also drag and drop the portlet from the inline resource catalog.
7. Click Save to save your changes.
8. Select the Edit icon for the portlet.
9. In the Component Properties dialog, select the Display Options tab.
10. For the Render Portlet In IFrame option, select True.
11. Continue by checking the portlet connection as described in Section 54.3.3.3.4, “Testing the Portal Portlet Connection.”

54.3.3.3.4 Testing the Portal Portlet Connection Use this procedure to test the portal portlet connection by modifying content and checking that the modification shows up in the EBS application.

1. On the portal page to which you added the EBS portlet, 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.

54.3.4 Integrating EBS Applications as Data Controls in WebCenter Portal

This section describes how to add EBS applications as Web service data controls on portal pages.

This section contains the following subsections:

- Section 54.3.4.1, "Generating the WSDL"
- Section 54.3.4.2, "Adding a Web Service Data Control to a Portal Page"

54.3.4.1 Generating the WSDL

This section describes how to create the WSDL.

1. Log on to E-Business Suite as the SYSADMIN user.
2. In the Navigation pane, expand the Integrated SOA Gateway node and under the Integrated SOA Gateway sub-node click Integration Repository.
3. From the Integration Repository tab, navigate to the part of the EBS application to expose. For example, for the price request interface, you would go to Order Management Suite > Advanced Pricing > Price List, and then selecting Price Request from the list of integration points.
4. Click Generate WSDL to expose the integration point (for our example, a PL/SQL API integration point) as a Web service.
5. Right-click the View WSDL link and open the link in a new tab or new window (be sure to keep the tab or window open as you’ll need it later).
6. On the Integration Repository page under Procedures and Functions (see Figure 54-11), check the box for the object to grant access to, and then click Grant Access.
7. Select the **Grantee Type** and **Grantee Name** (the user you want to grant access to the exposed object), or use the Search tool. For our example, we will grant access to ASADMIN.

**Figure 54–12  Integration Repository - Create Grants Page**

### 54.3.4.2 Adding a Web Service Data Control to a Portal Page

Once you have the WSDL, you can continue by using it to create a Web service data control.

**Note:** Before you can add a data control or task flow containing a data control to a portal page you must first have configured WS-Security for WebCenter Portal. For more information about configuring WS-Security, see the "Configuring WS-Security" chapter in Administering Oracle WebCenter Portal.

For more information about creating a Web service data control, see the "Creating a Web Service Data Control" section in Building Portals with Oracle WebCenter Portal.
For information about Web service data controls, see also the "Web Service Data Controls" section in Building Portals with Oracle WebCenter Portal.

To create a Web service data control:

1. In WebCenter Portal or the portal in which you want to create the data control, go to either the Shared Assets or Assets page.
2. Select Data Controls and click Create.
   The Create New Data Control dialog displays (see Figure 54–33).

![Figure 54–13  Create New Data Control Dialog](image)

3. In the Create New Data Control dialog, enter a Name and Description for the data control, select Web Service as the Data Control Type, and then click Continue.
4. Enter the WSDL URL that you generated in Section 54.3.4.1, "Generating the WSDL" and other details for the data control and click Continue.
5. Click Show Methods.
6. Select the method(s) to make available and click Next.
7. Enter the parameter default values, if any, and click Create.
8. To make the data control available, from the Shared Assets or Assets page, select Task Flows. The Create New Task Flow dialog displays (see Figure 54–35).

![Figure 54–14  Create New Task Flow Dialog](image)

9. Enter the task flow Name and Description, select the Mashup Style to use click Create to create the task flow.
10. Select the task flow and click the Edit icon.
11. Add the data control (with parameter form) as a table onto the task flow and verify the data.
12. To make the task flow available, navigate to Administration > Business Role Pages.
13. Select Business Role Page and click the Create icon.
14. Edit the page and save the changes.
15. Drop the task flow onto the page and verify the data.

54.4 Integrating JD Edwards Applications

This section describes how to integrate JD Edwards applications into WebCenter Portal or a Framework application.

This section contains the following subsections:

- Section 54.4.1, "Integrating JD Edwards Applications in a Framework Application"
- Section 54.4.2, "Integrating JD Edwards Applications in a Portal"

54.4.1 Integrating JD Edwards Applications in a Framework Application

This section includes the following subsections:

- Section 54.4.1.1, "Preparing the JD Edwards Application for Remote Access"
- Section 54.4.1.2, "Registering the JD Edwards WSRP Producer in the Framework Application"
- Section 54.4.1.3, "Creating the JSF Page to Consume the Remote Producer"
- Section 54.4.1.4, "Testing the Framework Application"

54.4.1.1 Preparing the JD Edwards Application for Remote Access

Before you can add JD Edwards standalone regions to Portal 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 Portal Framework application page. To view the XML content of the JDE WSDL in the browser, open the Page source of the page in the browser.

54.4.1.2 Registering the JD Edwards WSRP Producer in the Framework Application

Follow the instructions below to create or open an existing Portal Framework application, and register the JD Edwards WSRP producer.

1. Create a new Portal 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 JDE as the 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 54.4.1.3, "Creating the JSF Page to Consume the Remote Producer."
54.4.1.3 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 Portal 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 Portal Framework application appear in the JD Edwards application as described in Section 54.4.1.4, “Testing the Framework Application.”

54.4.1.4 Testing the Framework Application

Use this procedure to test the Portal Framework application by modifying content on the JSF page in the Portal 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.

54.4.2 Integrating JD Edwards Applications in a Portal

This section contains the following subsections:

- Section 54.4.2.1, "Registering the Producer"
- Section 54.4.2.2, "Adding the JD Edwards Portlet to a Portal Page"
- Section 54.4.2.3, "Testing the Portal Portlet Connection"

54.4.2.1 Registering the Producer

You can register the JD Edwards WSRP producer directly in WebCenter Portal, as described in the "Registering Portlet Producers with the Administration Console"
section in Administering Oracle WebCenter Portal. 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 WebCenter Portal for
remote access as described in Section 54.4.1.1, "Preparing the JD Edwards
Application for Remote Access."
2. Log into Fusion Middleware Control for the WebCenter Portal domain (we_domain
by default).
3. Expand WebCenter Portal in the Navigation bar and from the WebCenter Portal
menu, and select Register Producer.
   The Add Portlet Producer page displays.
4. Enter JDE as the 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 portal page as described in Section 54.4.2.2,
"Adding the JD Edwards Portlet to a Portal Page."

54.4.2.2 Adding the JD Edwards Portlet to a Portal Page
Use the following procedure to consume the JD Edwards remote producer in
WebCenter Portal.

1. Go to the page, or create a new page, where you want to add the JD Edwards
portlet.
2. Open the page in edit mode, as explained in the "Opening a Page in the Page
Editor (Composer)" section in Building Portals with Oracle WebCenter Portal.

---

**Note:** By default, the resource catalog appears inline on the page.
The following instructions assume that you have not changed this
default view (that is, you have not selected Hide Catalog).

---

3. In the resource catalog, select UI Components and then select Portlets.

---

**Note:** 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 "Working
with Resource Catalogs" chapter in Building Portals with Oracle
WebCenter Portal.

---

4. Click Open to open the Portlets catalog.
5. Click the portlet you added in Fusion Middleware Control.
6. Click Add for the EBS portlet you want to add to your portal page. You can also
drag and drop the portlet from the inline resource catalog.
7. Click Save to save your changes.
8. Select the Edit icon for the portlet.
9. In the Component Properties dialog, select the Display Options tab.
10. For the Render Portlet In IFrame option, select True.

11. Continue by checking the portlet connection as described in Section 54.3.3.4, "Testing the Portal Portlet Connection."

54.4.2.3 Testing the Portal Portlet Connection
Use this procedure to test the portal portlet connection by modifying content and checking that the modification shows up in the JD Edwards application.

1. On the portal 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.

54.5 Integrating PeopleSoft Applications
This section describes how to integrate PeopleSoft applications in WebCenter Portal or Framework application.

This section contains the following subsections:
- Section 54.5.1, "Introduction to Integrating PeopleSoft Applications"
- Section 54.5.2, "Integrating PeopleSoft Applications as WSRP Portlets"
- Section 54.5.3, "Integrating PeopleSoft Applications as Data Controls in WebCenter Portal"

54.5.1 Introduction to Integrating PeopleSoft Applications
This section describes the benefits and methods involved in integrating PeopleSoft applications into WebCenter Portal and Framework applications.

This section includes the following subsections:
- Section 54.5.1.1, "Understanding PeopleSoft Integration"
- Section 54.5.1.2, "Requirements for Integrating PeopleSoft Applications"

54.5.1.1 Understanding PeopleSoft Integration
PeopleTools 8.51 and later lets you expose PeopleSoft applications as WSRP portlets in remote applications such as WebCenter Portal or a Framework application. This allows people who only need access to a small portion of PeopleSoft's functionality to access it through WebCenter Portal or a Framework application without needing to open or learn the entire PeopleSoft application.

54.5.1.2 Requirements for Integrating PeopleSoft Applications
This section the prerequisites for integrating PeopleSoft objects in WebCenter Portal or a Portal Framework application.
- PeopleSoft 9.0 or later.
- PeopleTools 8.51 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 Oracle 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 the Oracle WebCenter Portal server.
For PeopleTools 8.51, you may need to create and configure a custom OWSM policy in order to fully support WS-Security. For more information, see Section 54.5.2.6, "Configuring WS-Security for PeopleTools 8.51."

For PeopleTools 8.51, only upper case subject names are supported, requiring that only fully upper case user IDs can be used in WebCenter for the integration to work.

54.5.2 Integrating PeopleSoft Applications as WSRP Portlets

This section describes how to expose PeopleSoft applications as WSRP portlets in a portal or Framework application.

This section includes the following subsections:

- Section 54.5.2.1, "Preparing the PeopleSoft Application for Remote Access"
- Section 54.5.2.2, "Configuring WS-Security for PeopleTools 8.52 and Later"
- Section 54.5.2.3, "Attaching a WS-Security Policy to WebCenter Portal"
- Section 54.5.2.4, "Integrating PeopleSoft Applications in a Portal"
- Section 54.5.2.5, "Integrating PeopleSoft Applications in a Framework Application"
- Section 54.5.2.6, "Configuring WS-Security for PeopleTools 8.51"

54.5.2.1 Preparing the PeopleSoft Application for Remote Access

This section describes how to prepare the PeopleSoft application so that it can be consumed by WebCenter Portal 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 Portal.
4. Select Structure and Content.

The Structure and Content page displays a list of folders containing PeopleSoft objects that could be exposed as a WSRP Portlet (see Figure 54–15).
5. Navigate to the folder and subfolder (if required) containing the service that you want to expose as portlet in WebCenter Portal or Framework application and click Edit to open it. For example, you could select **Self Service**, **Personal Information**, and then **Personal Information Summary**.

The Content Ref Administration page displays (see Figure 54–16).
6. On the General tab, select the **WSRP Producible** checkbox.

7. Save the page.

8. In the PeopleSoft Application Designer, open the component object to the Request Details page that gets displayed in PeopleSoft, and in the Component Properties section, check the **WSRP Compliant** check box.

9. From the main PeopleSoft menu, expand PeopleTools and then Portal and select **WSRP Production**.
   
   The Producer Offered Portlets page displays (see Figure 54–17).
Verify that the service is exposed, and then expand Web Service Endpoint URL and copy the URL (the WSDL).

Open a new tab in your browser, and paste the copied URL into the Navigation Bar to access the WSDL page.

Copy the URL to the clipboard.

Continue by integrating the PeopleSoft WSRP producer in WebCenter Portal or a Framework application as described in Section 54.5.2.5, "Integrating PeopleSoft Applications in a Framework Application" and Section 54.5.2.4, "Integrating PeopleSoft Applications in a Portal."

54.5.2.2 Configuring WS-Security for PeopleTools 8.52 and Later
This section describes how to create a keystore for both Oracle WebCenter Portal and PeopleSoft, and exchange the private key between them. This step is required prior to adding WS-Security policies for both portals and FrameWork applications.

First, we will create WebCenter keystore as webcenter.jks with orakey as the private key, and PeopleSoft's public key rootCA and the certificate that PeopleSoft will use as the WS-Security recipient using the following keytool commands:

```bash
./keytool -genkeypair -keyalg RSA -dname "cn=orakey,dc=us,dc=oracle,dc=com"
```
Integrating PeopleSoft Applications

2. Next, we will create PeopleSoft keystore as peoplesoft.jks with rootCA as the private key and WebCenter's public key orakey and the certificate that WebCenter will use as the WS-Security recipient.

   ./keytool -genkeypair -keyalg RSA -dname "cn=rootCA,dc=us,dc=oracle,dc=com" -alias rootCA -keypass password -keystore peoplesoft.jks -storepass password -validity 720

   ./keytool -exportcert -v -alias rootCA -keystore peoplesoft.jks -storepass password -rfc -file rootca.cer

   ./keytool -importcert -trustcacerts -alias rootCA -file rootca.cer -keystore webcenter.jks -storepass password

3. After creating the key stores for Oracle WebCenter Portal and PeopleSoft, copy the peoplesoft.jks to the PeopleSoft host and webcenter.jks to WebCenter host:
   
   ■ Copy peoplesoft.jks to /home/psadm2/psft/pt/8.52/webserv/<Domain_Name>/applications/peoplesoft/pspc.war/WEB-INF/classes

   ■ Copy webcenter.jks to <Domain_Home>/config/fmwconfig/

4. Install the certificate in PeopleSoft as shown below:
   

   The Digital Certificates page displays (see Figure 54–18).
b. Click + to add a new entry.

We need to add digital certificates for Remote and RootCA as shown in Figure 54–19.
Figure 54–19  Digital Certificates Page

- Enter the **Type** as RootCA, **Alias** as orakey, **Issuer Alias** as orakey, and then click the **Search** icon (magnifying glass).
- Click **Import** and in the popup, enter the entire text of orakey.cer created earlier and click **OK**.
- Click + to add another new entry, and enter the **Type** as Remote, **Alias** as orakey, **Issuer Alias** as orakey and then click the **Search** icon.
- Click **Import** and in the popup, enter the entire text of orakey.cer created earlier and click **OK**.


6. Use PSCipher.sh to create an Encrypted Password and update the KeyStore password as shown in Figure 54–20.
7. Check the local node definition in PeopleSoft:
      The Nodes page displays.
   b. Click Search and click PSFT-HR.
c. Select Password from the Authentication Option drop-down list and click Save.

8. Continue by adding a WS-Security policy to WebCenter Portal or the Framework application as described in Section 54.5.2.3, "Attaching a WS-Security Policy to WebCenter Portal."

54.5.2.3 Attaching a WS-Security Policy to WebCenter Portal
This section describes how to attach a WS-Security policy to WebCenter Portal.

---

**Note:** Before continuing with the steps below you must have configured the WebCenter and PeopleSoft key stores as described in Section 54.5.2.2, "Configuring WS-Security for PeopleTools 8.52 and Later."

---

- Section 54.5.2.3.1, "Configuring WSS 1.0 SAML Token with Message Integrity"
- Section 54.5.2.3.2, "Configuring WSS 1.0 Username Token Without Password"
- Section 54.5.2.3.3, "Configuring WSS 1.0 SAML Token with Message Protection"
- Section 54.5.2.3.4, "Configuring WSS 1.0 Username Token with Password"

---

54.5.2.3.1 Configuring WSS 1.0 SAML Token with Message Integrity
Follow the steps below to configure the WSS1.0 SAML Token with Message Integrity policy for WebCenter Portal:
1. Navigate to the following directory on the PeopleSoft server:
   
   `/home/psadm2/psft/pt/8.53/webserv/peoplesoft/plabin`
   
   and run the following command:
   
   `.redeployWSRP.sh 6`
   
   This will update the PeopleSoft WSRP security options to use WSRPBaseService with SAMLToken Full Security.

2. In PeopleSoft, navigate to PeopleTools > Security > SAML Administration Setup > SAML Inbound Setup.

   The SAML Inbound Setup page displays (see Figure 54–22).

   **Figure 54–22  SAML Inbound Setup Page**

   ![SAML Inbound Setup Page](image)

3. Open the Add a New Value tab and map the WebCenter user with the PeopleSoft user if they use a different OID (example settings are shown below), and then click Save.

   Example:
   
   - **Certificate Alias** - orakey
   - **Issuer** - WWW.ORACLE.COM
   - **SubjectName** - pat
   - **QualifierName** - WWW.ORACLE.COM
   - **Mapping PeopleSoft UserID** - PS

4. Continue by registering the WSRP producer and adding the portlet to a portal page as shown in Section 54.5.2.4, "Integrating PeopleSoft Applications in a Portal."

54.5.2.3.2  Configuring WSS 1.0 Username Token Without Password  Follow the steps below to attach a WSS 1.0 Username Token without Password policy to WebCenter Portal.

1. Create a WebCenter user in PeopleSoft:

The Copy User Profiles page displays (see Figure 54–23).

**Figure 54–23  Copy User Profiles Page - Search Criteria**

b. Search for the user to add (PS, for example).

The search results display (see Figure 54–24).

**Figure 54–24  Copy User Profiles Page - Search Results**

c. Enter the New User ID (for example, Pat), a Description, the New Password, check the Copy ID Type Information option and click Save.
2. Log into Fusion Middleware Control, select the domain and navigate to **Security > Security Provider Configuration.**

The Security Provider Configuration page displays.

3. Open the Keystore section and click **Configure.**

The Keystore Configuration page displays.

4. Enter `.webcenter.jks` for the **KeyStore Path**, `orakey` for the **Key Alias**, `orakey` for the **Crypt Alias**. Enter the associated passwords and click **OK**.

Note that you must restart the entire domain for the configuration changes to take effect.

5. Navigate to `/home/psadm2/psft/pt/8.53/webserv/peoplesoft/piabin` and run the following command:

   ```
   ./redeployWSRP.sh 8
   ```

   This will update the PeopleSoft WSRP security options to use WSRPBaseService with UsernameToken, No Password Full Security Option With WSS Response.

6. Continue by registering the WSRP producer and adding the portlet to a portal page as shown in **Section 54.5.2.4, "Integrating PeopleSoft Applications in a Portal."**

**54.5.2.3.3 Configuring WSS 1.0 SAML Token with Message Protection** Follow the steps below to attach the WSS1.0 SAML Token with Message Protection policy to WebCenter Portal.

1. Log into Fusion Middleware Control, select the domain and navigate to **Security > Security Provider Configuration.**

The Security Provider Configuration page displays.

2. Open the Keystore section and click **Configure.**

The Keystore Configuration page displays.

3. Enter `.webcenter.jks` for the **KeyStore Path**, `orakey` for the **Key Alias**, `orakey` for the **Crypt Alias**. Enter the associated passwords and click **OK**.

Note that you must restart the entire domain for the configuration changes to take effect.

4. Navigate to `/home/psadm2/psft/pt/8.53/webserv/peoplesoft/piabin` and run the following command:

   ```
   ./redeployWSRP.sh 10
   ```

   This will update the PeopleSoft WSRP security options to use WSRPBaseService with SAMLToken Full Security Option With WSS Response.

5. Continue by registering the WSRP producer and adding the portlet to a portal page as shown in **Section 54.5.2.4, "Integrating PeopleSoft Applications in a Portal."**

**54.5.2.3.4 Configuring WSS 1.0 Username Token with Password** Follow the steps below to attach the WSS1.0 SAML Token with Message Protection policy to WebCenter Portal.

1. Log into Fusion Middleware Control, select the domain and navigate to **Security > Security Provider Configuration.**

The Security Provider Configuration page displays.
2. Open the Keystore section and click **Configure**.
The Keystore Configuration page displays.

3. Enter `/webcenter.jks` for the **KeyStore Path**, `orakey` for the **Key Alias**, `orakey` for the **Crypt Alias**. Enter the associated passwords and click **OK**.
Note that you must restart the entire domain for the configuration changes to take effect.

4. Navigate to `/home/psadm2/psft/pt/8.53/webserv/peoplesoft/piabin` and run the following command:
   
   ```sh
   ./redeployWSRP.sh 7
   ```

   This will update the PeopleSoft WSRP security options to use WSRPBaseService with UsernameToken Full Security Option With WSS Response.

5. Continue by registering the WSRP producer and adding the portlet to a portal page as shown in Section 54.5.2.4, "Integrating PeopleSoft Applications in a Portal."

54.5.2.4 Integrating PeopleSoft Applications in a Portal
This section describes how to integrate a PeopleSoft application in a portal.

This section contains the following subsections:

- Section 54.5.2.4.1, "Registering the PeopleSoft WSRP Producer for a Portal"
- Section 54.5.2.4.2, "Adding the PeopleSoft Portlet to a Portal Page"
- Section 54.5.2.4.3, "Testing the Portal Portlet Connection"

54.5.2.4.1 Registering the PeopleSoft WSRP Producer for a Portal
You can register the PeopleSoft WSRP producer directly in WebCenter Portal, as described in the "Registering Portlet Producers with the Administration Console" section in Administering Oracle WebCenter Portal. 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 WebCenter Portal for remote access as described in Section 54.5.2.1, "Preparing the PeopleSoft Application for Remote Access."

2. Log into Fusion Middleware Control for the WebCenter Portal domain (`wc_domain` by default).

3. Expand **WebCenter Portal** in the Navigation bar and from the WebCenter Portal menu, select **Register Producer**.
The Add Portlet Producer page displays.

4. Set the **Producer Type** to **WSRP Producer**, enter a **Connection Name**, and paste the WSDL endpoint URL that you copied in step 1 into the **URL End Point** field.

5. If required, configure WS-Security in WebCenter Portal as described in Section 54.5.2.6, "Configuring WS-Security for PeopleTools 8.51."

6. Click **OK** and verify that the producer connection was created successfully.

7. Continue by adding the portlet to a portal page as described in Section 54.5.2.4.2, "Adding the PeopleSoft Portlet to a Portal Page."
54.5.2.4.2 Adding the PeopleSoft Portlet to a Portal Page  Follow the steps below to add the PeopleSoft portlet to a portal page.

1. Go to the page, or create a new page, where you want to add the PeopleSoft portlet.

   **Note:** If you configured WS-Security, be sure to use the user account that was used in the SAML Inbound Setup page in PeopleSoft (see Section 54.5.2.3, "Attaching a WS-Security Policy to WebCenter Portal").

2. Open the page in edit mode, as explained in the "Opening a Page in the Page Editor (Composer)" section in Building Portals with Oracle WebCenter Portal.

   **Note:** By default, the resource catalog appears inline on the page. The following instructions assume that you have not changed this default view (that is, you have not selected Hide Catalog).

3. In the resource catalog, select **UI Components** and then select **Portlets**.

   **Note:** 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 "Working with Resource Catalogs" chapter in Building Portals with Oracle WebCenter Portal.

4. Click **Open** to open the Portlets catalog.
5. Click the portlet you added in Fusion Middleware Control.
6. Click **Add** for the EBS portlet you want to add to your portal page. You can also drag and drop the portlet from the inline resource catalog.
7. Click **Save** to save your changes.
8. Select the **Edit** icon for the portlet.
9. In the Component Properties dialog, select the **Display Options** tab.
10. For the Render Portlet In IFrame option, select **True**.
11. Continue by checking the portlet connection as described in Section 54.3.3.3.4, "Testing the Portal Portlet Connection.”

54.5.2.4.3 Testing the Portal Portlet Connection  Use this procedure to test the portal portlet connection by modifying content and checking that the modification shows up in the PeopleSoft application.

1. On the portal 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.
54.5.2.5 Integrating PeopleSoft Applications in a Framework Application
This section describes how to create or modify a Portal Framework application that integrates with PeopleSoft Applications.

This section includes the following subsections:

- **Section 54.5.2.5.1, "Registering the PeopleSoft WSRP Producer in the Framework Application"
- **Section 54.5.2.5.2, "Creating the JSF Page to Consume the Remote (WSRP) Producer"
- **Section 54.5.2.5.3, "Testing the Framework Application"

54.5.2.5.1 Registering the PeopleSoft WSRP Producer in the Framework Application
Follow the instructions below to create or open an existing Portal Framework application, and register the PeopleSoft WSRP producer.

1. Use JDeveloper to create a new Framework application (using the WebCenter Portal Framework Application template) 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 54–25).

![Figure 54–25 Specify Producer Name Page](image)

3. Enter a name for the WSRP producer and click **Next**.

   The Specify Connection Details page displays (see Figure 54–25).
Integrating PeopleSoft Applications

Figure 54–26  Specify Connection Details

4. Paste the WSDL (the URL you copied earlier) into the WSDL URL field and click Next.

   The Configure Security Attributes page displays.

5. If you are configuring WS-Security, continue with the steps below. Otherwise, continue with step 6.

   Note that to configure WS-Security, you must have set up the key stores and attached the policies as described in Section 54.5.2.2, "Configuring WS-Security for PeopleTools 8.52 and Later."

   a. From the Token Profile dropdown list, select the WS-Security policy to use.
   
   b. Set the Configuration option to Custom, the Default User to weblogic, and the Issuer Name to www.oracle.com and click Next.

      The Specify Keystore page displays.

   c. Enter the settings for the keystore based on the configuration settings you defined in Section 54.5.2.2, "Configuring WS-Security for PeopleTools 8.52 and Later."

6. Click Finish.

   The WSRP producer should now appear under Connections.

7. Continue by creating a JSF page to consume the WSRP producer as described in Section 54.5.2.5.2, "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.
54.5.2.5.2 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. In the Navigation panel, under Connections, expand the newly created WSRP producer and drag the portlet you created 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 Portal Framework application appear in the PeopleSoft application as described in Section 54.5.2.5.3, "Testing the Framework Application."

54.5.2.5.3 Testing the Framework Application Use this procedure to test the Portal Framework application by modifying content on the JSF page in the Portal 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.

54.5.2.6 Configuring WS-Security for PeopleTools 8.51

This section describes the supported OWSM policies for PeopleTools 8.51. It is important to note that PeopleTools release 8.51 does not support outgoing WS-Security headers in its messages. However, some out-of-the-box Oracle 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. The different integration scenarios that would require you to create custom WS-Security policies, and the steps required on the Oracle WebCenter Portal side to configure them are also described in this section.

For integration scenarios with PeopleTools 8.51, you can use WSS10 SAML Token with Message Integrity, WSS10 SAML Token with Message Protection, or WSS10 Username Token with Password as OWSM the policy.
This section includes the following subsections:

- **Section 54.5.2.6.1, "Configuring WS-Security for WSS10 SAML Token with Message Integrity"**
- **Section 54.5.2.6.2, "Configuring WS-Security for WSS10 SAML Token with Message Protection"**
- **Section 54.5.2.6.3, "Configuring WS-Security for WSS10 Username Token with Password"**

### 54.5.2.6.1 Configuring WS-Security for WSS10 SAML Token with Message Integrity (PeopleSoft policy: WSRPBaseService with SAMLToken Full Security Option (timestamp))

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:

2. Generate a certificate containing the public key of the Oracle 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 WebCenter Portal or a Framework application page.

### 54.5.2.6.2 Configuring WS-Security for WSS10 SAML Token with Message Protection (PeopleSoft policy: WSRPBaseService with SAMLToken Full Security Option (timestamp) With WSS Response)

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 8.51 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 Oracle 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 check boxes under Message Signing Setting and Message Encrypt Setting, and save the policy.
6. Check that the public certificate of the PeopleSoft keystore is imported into the keystore used in the Oracle WebCenter Portal domain.
7. 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://xmlns.oracle.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')
```

Use the alias for the imported peoplesft public key as the value for the recptAlias parameter.

---

**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'.

---

### 54.5.2.6.3 Configuring WS-Security for WSS10 Username Token with Password (PeopleSoft policy: WSRPBaseService with UsernameToken Full Security Option With WSS Response)

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 8.51 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 Oracle 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 check boxes under Message Signing Setting and Message Encrypt Setting, and save the policy.
6. Check that the public certificate of the PeopleSoft keystore is imported into the keystore used in the Oracle WebCenter Portal domain.
7. Use WLST to register the producer using the newly created policy as shown in the following example:

```
registerWSRPProducer('webcenter', '<Producer_Name>', '<URL>', timeout=100,
    tokenType='oracle/wss10_username_token_with_message_protection_plaintext_response_client_policy', extApp='<Ext_App_Name>',
    enforcePolicyURI='false', issuer='www.oracle.com',
    sigKeyAlias='webcenter', sigKeyPswd='welcome1', encKeyAlias='webcenter',
    encKeyPswd='welcome1', recptAlias='peopleTools_public')
```
Integrating PeopleSoft Applications

Use the alias for the imported peoplesft public key as the value for the recepAlias parameter.

---

**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'.

---

54.5.3 Integrating PeopleSoft Applications as Data Controls in WebCenter Portal

This section describes how to add PeopleSoft applications as Web service data controls on portal pages.

This section includes the following subsections:

- Section 54.5.3.1, "Preparing the WSDL"
- Section 54.5.3.2, "Creating a Web Service Data Control"

54.5.3.1 Preparing the WSDL

Follow the steps below to prepare a the WSDL.

1. Log into the PeopleSoft Console as an administrator.
2. Navigate to PeopleTools > Web Profile > Web Profile Configuration.
3. Click Search and select DEV from the results list.
4. Open the General tab and enter the **Authentication Domain** for your host.
   For example, if your host name is `ps.example.com`, enter `.example.com` in the **Authentication Domain** field.

5. Save your changes and close the application.

6. Open the `C:\Windows\System32\drivers\etc\hosts` file for editing.

7. On a new line enter the IP address and the full host name with the authentication domain.
   For example:
   ```
   193.128.1.113 ps.example.com
   ```

8. Save the file and reboot the server.

9. Log into the PeopleSoft application using the following URL:
   ```
   http://<host_name>:8000/ps/signon.html
   ```
   For example:
   ```
   http://ps.example.com:8000/ps/signon.html
   ```

10. From the Main Menu, navigate to **PeopleTools > Integration Broker > Configuration > Gateways**.

11. Search for the **GatewayID** LOCAL. The Local Gateway URL is set to
12. Using the Local Gateway URL, ping the gateway to make sure it's active.
13. Open the Gateway Setup Properties and log in as an administrator.
14. On the PeopleSoft Node Configuration page, check that the node being used is PSFT_HR.
15. Ping the node
16. From the Main Menu, navigate to PeopleTools > Integration Broker > Configuration > Service Configuration.
17. Open Setup Target Locations and check that the Target Location is set to <Local Gateway URL>/PSFT_HR.
18. From the Main Menu, navigate to PeopleTools > Integration Broker > Integration Setup > Nodes.
19. Click Search.
20. Click the Default Local Node PSFT_HR.
21. On the Nodes tab, check that the Default UserID is set correctly as in the example in Figure 54–28.

Figure 54–28  Nodes Page - Node Definitions

22. Click Return to Search.
23. Click the ANONYMOUS node.
24. Change the default UserID to the PeopleSoft Login ID (for example, PS) as in the example in Figure 54–29.
25. Save the changes and navigate to Main Menu > PeopleTools > Integration Broker > Web Services > CI-Based Services.

26. Search for and select the Component Interface Name (for example, CURRENCY) as in the example in Figure 54–30.
27. Click Review CI Status.

The CI-Based Services - Review Status page displays (see Figure 54–31).

28. Select the available methods (Get and Find, in this case) and click Display Selected Actions.

29. On the Confirm Actions dialog, click Perform Selected actions.

30. Click View Service Definition.

31. Click Provide Web Service.

The Select Service Operations page displays (see Figure 54–32).
32. Select the **Select All** check box and click **Next** until you reach the last page.

33. Click **Finish** to generate the WSDL.

   You should now be able to access the WSDL URL. For this example, the URL would be:

   http://ps.example.com:8000/PSIGN/PeopleSoftServiceListeningConnector/PSFT_HR/CI_CURRENCY.1.wsdl

34. Continue by creating a Web service data control as shown in Section 54.5.3.2, "Creating a Web Service Data Control."

### 54.5.3.2 Creating a Web Service Data Control

Once you have the WSDL, you can continue by using it to create a Web service data control. In this section we’ll continue with the example we started in Section 54.5.3.1, "Preparing the WSDL."

---

**Note:** Before you can add a data control or task flow containing a data control to a portal page you must first have configured WS-Security for WebCenter Portal. For more information about configuring WS-Security, see the "Configuring WS-Security" chapter in Administering Oracle WebCenter Portal.

---

For more information about creating a Web service data control, see the "Creating a Web Service Data Control" section in Building Portals with Oracle WebCenter Portal. For information about Web service data controls, see also the "Web Service Data Controls" section in Building Portals with Oracle WebCenter Portal.

To create a Web service data control:

1. In WebCenter Portal or the portal in which you want to create the data control, go to either the **Shared Assets** or **Assets** page.

2. Select **Data Controls** and click **Create**.

   The Create New Data Control dialog displays (see Figure 54–33).
3. In the Create New Data Control dialog, enter a Name and Description for the data control, select Web Service as the Data Control Type, and then click Continue.

4. Enter the WSDL URL and other details for the data control and click Continue. For our example, the URL would be:

   http://ps.example.com:8000/PSIGW/PeopleSoftServiceListeningConnector/PSFT_HR/CI_CURRENCY.1.wsdl

5. For our example, enter the Default Value for CURRENCY_CD as USD and click Create (see Figure 54–34).

6. To make the data control available, from the Shared Assets or Assets page, select Task Flows. The Create New Task Flow dialog displays (see Figure 54–35).

7. Click Create to create the task flow.

8. Select the task flow and click the Edit icon.

9. Add the data control (with parameter form) as a table onto the task flow and verify the data.

10. To make the task flow available, navigate to Administration > Business Role Pages.

11. Select Business Role Page and click the Create icon.
54.6 Integrating Oracle Business Intelligence Presentation Services

This section explains how to use JDeveloper to configure WebCenter Portal to integrate with the Oracle Business Intelligence Presentation Services catalog. At runtime, users can add business intelligence objects to their WebCenter Portal application pages. For information about adding Oracle Business Intelligence objects to Portal 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 54.6.1, "Introduction to Integrating Oracle Business Intelligence Presentation Services"
- Section 54.6.2, "Configuring Credentials for Connecting to the Oracle BI Presentation Catalog"
- Section 54.6.3, "Integrating Oracle Business Intelligence Objects in WebCenter Portal"

54.6.1 Introduction to Integrating Oracle Business Intelligence Presentation Services

This section explains how you use JDeveloper to configure WebCenter Portal to integrate with the Oracle Business Intelligence Presentation Services catalog. For information about adding Oracle Business Intelligence objects to Portal 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 54.6.1.1, "Understanding Oracle Business Intelligence Presentation Services Integration"
- Section 54.6.1.2, "Requirements for Integrating Oracle Business Intelligence Presentation Services"

54.6.1.1 Understanding Oracle Business Intelligence Presentation Services Integration

You can use JDeveloper to create Portal 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 Portal Framework application pages. Any filters, prompts, and actions links included in the Business Intelligence objects will work within the Portal Framework application or portal pages.

At runtime, Oracle 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 Portal Framework application pages.

54.6.1.2 Requirements for Integrating Oracle Business Intelligence Presentation Services

To create a Portal 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 Portal Framework application, you must configure the library settings, update the `weblogic.xml` and `weblogic-application.xml` files, and have properly configured security.

For both Portal Framework applications and portals, you must also set up a connection to the BI application as well as configuring security as described in the "Creating an Oracle BI EE Presentation Services Connection" section in 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 54.6.2, "Configuring Credentials for Connecting to the Oracle BI Presentation Catalog."

The following prerequisites apply:

**Oracle WebCenter Portal**

- The WC_Spaces server has been installed and configured, including the database connection, Content Server connection, and Fusion Middleware Control

**OBIEE**

- Oracle Business Intelligence Applications
- OBI Enterprise Edition version 11.1.1.7.0
- OBI Applications version 7.9.6.3 (Optional)
- OBIEE is already installed, configured, and up and running (Database – OBI Enterprise Edition)
- OBI Applications is installed and set up and all content is available from the OBIEE environment (Optional)

**Security**

The OBIEE integration requires that the identity store user name population be the same across WebCenter and OBIEE. This can be done by either:

- Having WebCenter and OBIEE share the same identity store (recommended)
- Maintaining identical user names across separate WC and OBIEE identity stores

54.6.1.3 Advanced Integration Options

The sections below focus on adding resources such as business intelligence analyses, dashboards, and scorecard components that can be easily dropped onto a page. As well as these approaches, there are also options for using Web services and BI EE Logical SQL view object to embed business intelligence data into a Framework application. For more information about using Web services, see the “Introduction to Oracle Business Intelligence Web Services” section in Oracle Fusion Middleware.
Integrating Oracle Business Intelligence Presentation Services

Integrator’s Guide for Oracle Business Intelligence Enterprise Edition. For more information about using the BI EE Logical SQL view object, see the “Using the Oracle BI EE Logical SQL View Object” section in Oracle Fusion Middleware Developer’s Guide for Oracle Business Intelligence Enterprise Edition.

54.6.2 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 54.6.2.1, “Checking for the BIImpersonatorUser”
- Section 54.6.2.2, “Creating the BIImpersonatorUser”
- Section 54.6.2.3, “Granting Permissions to BIImpersonatorUser”

54.6.2.1 Checking for the BIImpersonatorUser

Use the following steps to check if a BIImpersonatorUser user already exists, and that the roles assigned to it are correct:

1. Open WLS Administration Console for your Oracle BI EE instance using an Administrator account.
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. Check that BIImpersonatorUser appears in the list of users.
   If the BIImpersonatorUser does not appear in the list, continue by creating the BIImpersonatorUser as shown in Section 54.6.2.2, “Creating the BIImpersonatorUser.”
6. Log into Fusion Middleware Control with an administrator account.
7. In the Navigation panel, expand the Weblogic Domain node and select bifoundation_domain.
9. On the Application Policies page under Search, choose obi from the Application Stripe dropdown list.
10. From the Principal Type drop down list, select User.
11. In the Name field, enter BIImpersonatorUser and start the search (Figure 54–36).
12. If found, check that:

- **Resource Name** = oracle.bi.server.impersonatorUser
- **Resource Type** = oracle.bi.server.permission
- **Permission Actions** = _all_
- **Permission Class** = oracle.security.jps.ResourcePermission

13. If the BIImpersonatorUser is not found, continue by adding permissions for the BIImpersonatorUser as shown in Section 54.6.2.3, "Granting Permissions to BIImpersonatorUser."

### 54.6.2.2 Creating the BIImpersonatorUser

Use the following procedures to create a BIImpersonatorUser 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 impersonator 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 group to each user in this realm. For more information, see "How to Create and Use Impersonate User" in the Oracle Fusion Middleware Developer's Guide for Oracle Business Intelligence Enterprise Edition.

Follow the steps below to create the BIImpersonatorUser user:
1. Open WLS Administration Console for your Oracle BI EE instance using an Administrator account.

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 BIImpersonatorUser for the user name and enter a password.

7. Click OK.

54.6.2.3 Granting Permissions to BIImpersonatorUser

Follow the steps below to use Fusion Middleware Control to grant permissions to BIImpersonatorUser:

1. Open Fusion Middleware Control for your Oracle BI EE instance.

2. In the Navigation panel, expand the Weblogic Domain node and select `bifoundation_domain`.
   The `bifoundation_domain` pane displays.

3. In the `bifoundation_domain` pane, open the WebLogic Domain menu and select Security > Application Policies.
   The Search pane displays.

4. On the Application Policies page under Search, choose `obi` from the Application Stripe dropdown list.

5. Click Create.
   The Create Application Grant pane displays (Figure 54–37).

![Figure 54–37 Create Application Grant Pane](image-url)
   The Add Principal dialog displays.
7. From the Type drop down list, select User.
8. In the Principal Name field, enter BIImpersonatorUser and click OK.
   The Add Permission dialog displays (Figure 54–38).

**Figure 54–38  Add Permission Dialog**

10. Select Permissions and in the Permission Class field, enter oracle.security.jps.ResourcePermission.
11. In the Resource Name field, enter oracle.bi.server.impersonatorUser, start the search and then click Continue.
   Note that since we selected only Permissions, you may see an error message complaining that the resource type is missing (i.e., oracle.bi.server.permission). In that case, select Resource Types in the Add Permissions dialog and choose the appropriate one.
12. Click Select.
13. On the Application Grant page, click OK.
14. 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

### 54.6.3 Integrating Oracle Business Intelligence Objects in WebCenter Portal

This section describes how to configure WebCenter Portal integration with BI objects. For information about adding Oracle Business Intelligence objects to Portal Framework applications, see "Embedding Business Intelligence Objects in ADF Applications" in the Oracle Fusion Middleware Developer’s Guide for Oracle Business Intelligence Enterprise Edition.

- Section 54.6.3.1, "Adding or Modifying a Presentation Services Connection After Deployment"
54.6.3.1 Adding or Modifying a Presentation Services Connection After Deployment

Before you can begin integrating BI objects in WebCenter Portal, you must first configure a connection from WebCenter Portal to the BI server. Oracle BI EE provides an ADF MBean that lets you add a new connection to a deployed portal or BI ADF 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 54.6.2, "Configuring Credentials for Connecting to the Oracle BI Presentation Catalog" to specify credentials to connect to the Oracle BI Presentation Catalog.

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 WebCenter Portal domain, select the WebCenter tree node, and then the WebCenter Portal tree node, and then the webcenter(11.1.1.4.0) (WC_Spaces) tree node. Note that the WC_Spaces server must be running in order to perform this step.

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:
   - Select the Application Defined MBeans tree node.
   - Select the oracle.adf.share.connections tree node.
   - Select the Server: <my server name> tree node.
     For example, Server:DefaultServer or Server:WC_Spaces.
   - Select the Application:<your application's name> tree node.
     For example, Application:Application2 or Application:webcenter.
   - Open the ADF Connections tree node.
   - Open the child 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` 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.

   **Note:** The connection name is case sensitive. For SSO to work, the connection name and SSO account name must match exactly.

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 as shown below, and then click `Apply` to apply your changes.
   - From the OBIEE URL to the login page you need:
     
     ```
     context = analytics
     host = <host name>
     port = 7001 or the local value
     protocol = HTTP
     ```
   - From BIImpersonatorUser you need:
     
     ```
     username = BIImpersonatorUser
     password = <password>
     ```

10. Click on the ADFConnections folder in the Navigation pane, open the Operations tab, and then click `Save` to save the connection.

11. When you click `Invoke`, you should get the following message:
    
    "Confirmation Operation executed successfully."

### 54.6.3.2 Adding Oracle BI Objects to the WebCenter Portal Resource Catalog

Before you can add Oracle BI content to a portal page, you must add objects stored in the Oracle BI Presentation Catalog to the WebCenter Portal Resource Catalog.

1. Log into WebCenter Portal as an administrator.

2. In WebCenter Portal, click `Administration`.
   
   The Administration page displays.

3. Open the `Shared Assets` tab and click `Resource Catalogs`.

4. Click `Create`.

5. In the `Name` field, enter the name of the catalog you are creating. Complete the other fields on this dialog, as necessary.
6. Make the Resource Catalog available by checking its **Available** check box.

7. Select the new resource catalog and click **Edit**.

8. Select the **Add** menu and then choose **Add From Library**.

9. Double-click **Connections**.

   The BI Presentation Services folder displays.

10. Open this folder to display the Oracle BI objects and browse to and select the objects that you want to add.

11. Click **Add** to add the selected objects to the catalog.

   Continue by adding the object to a Framework application. For more information about managing resource catalogs at runtime, see the "Working with resource catalogs" section in *Building Portals with Oracle WebCenter Portal*. For information about managing Resource Catalogs in JDeveloper, see Chapter 14, "Developing Resource Catalogs."

### 54.6.3.3 Adding Oracle BI Content at Runtime

Use this procedure to create a portal page and add Oracle BI objects to it. Before you perform this procedure, you must have added Oracle BI objects to the WebCenter Portal Resource Catalog.

1. Log into WebCenter Portal and create a new portal or access an existing portal.

2. Make the Resource Catalog the default catalog for the portal.

   a. From the Pages and Portals dropdown list, select **Manage > Settings** and under Assets, select the catalog created in Section 54.6.3.2, "Adding Oracle BI Objects to the WebCenter Portal Resource Catalog" as the default resource catalog for Pages.

   b. Click **Save**.

3. From the Pages and Portals dropdown list, select **Create > Page**.

4. Browse to the folder containing the Oracle BI objects.

5. Select an analysis or dashboard and click **Add**.

   The object that you selected is added to the page.

### 54.6.3.4 Modifying a Business Intelligence Object’s Prompt Values

Use this procedure to test the portal 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 54.6.3.3, "Adding Oracle BI Content at Runtime."

4. Without exiting the edit mode of the page, save the portal page.

5. In the portal 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.
54.6.3.5 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 54.6.3.3, "Adding Oracle BI Content at Runtime."

4. Without exiting the edit mode for the page, save the portal page.

5. Locate the business intelligence object and click the Edit (wrench) icon.

   The Component Properties dialog displays.

6. On the portal 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.
Developing Components for WebCenter Portal Using JDeveloper

This chapter describes how a developer can leverage JDeveloper to develop components that can be deployed and used in WebCenter Portal. Components can be ADF task flows, data controls, managed beans, and other assets such as skins, page styles, task flow styles, page templates, content present display templates, navigations, and resource catalogs.

Out-of-the-box, WebCenter Portal provides tools targeted at knowledge workers and developers to quickly and easily create portals, intranets, extranets, and self-service applications that offer users a more effective and efficient way to access and interact with information, business applications, and process. WebCenter Portal can be customized using browser-based tools and also using JDeveloper.

This chapter includes the following sections:

- Section 55.1, "Developing Assets for WebCenter Portal"
- Section 55.2, "Developing Task Flows, Data Controls, and Managed Beans for WebCenter Portal"

55.1 Developing Assets for WebCenter Portal

This section includes the following subsections:

- Section 55.1.1, "About the WebCenterSpacesResources Project"
- Section 55.1.2, "Setting Up Your JDeveloper Environment for Asset Development"
- Section 55.1.3, "Enabling Direct Download and Uploads to WebCenter Portal"
- Section 55.1.4, "Opening and Exploring the WebCenterSpacesResources Project"
- Section 55.1.5, "Editing WebCenter Portal Assets in JDeveloper"
- Section 55.1.6, "Uploading Assets Directly to WebCenter Portal"
- Section 55.1.7, "Exporting WebCenter Portal Assets to an Archive"
- Section 55.1.8, "Troubleshooting Asset Round-Trip Development"

55.1.1 About the WebCenterSpacesResources Project

WebCenter Portal’s round-trip development features provide a simple, convenient way to create and modify WebCenter Portal assets without redeploying the entire application. Round-trip development refers to features and techniques that allow you to download assets from WebCenter Portal and upload them to JDeveloper for
Developing Assets for WebCenter Portal

Out-of-the-box, WebCenter Portal provides some default assets, such as skins and page templates, for people to use or modify. If your asset requirements extend beyond the browser-based editing capabilities of Portal Builder, you can further develop assets using the JDeveloper project WebCenterSpacesResources (available as a download from OTN). This project provides everything you need to create, modify, and upload the following WebCenter Portal assets:

- Page templates
- Navigations
- Page styles
- Skins
- Content Presenter display templates
- Resource catalogs
- Task flow styles
When you are ready to deploy asset customizations, you can upload them directly to a live WebCenter Portal application simply by providing JDeveloper with the name of the host and port where WebCenter Portal is running.

Alternatively, you can log in to WebCenter Portal and upload the asset through the relevant asset management page. If you decide to use this method you must save the asset to an export archive (.ear file) as shown in Figure 55–2.

**Figure 55–2  Developing Assets Using the WebCenterSpacesResources Project**

---

**Note:** This chapter describes how to manage asset customizations through the `WebCenterSpacesResources` project. For information on how to build/develop assets, see Section 55.1.5.4, "How to Build Assets for WebCenter Portal".

---

55.1.2 Setting Up Your JDeveloper Environment for Asset Development

Before you start, complete the following steps:

1. Download and install Oracle JDeveloper 11g (11.1.1.9.0).
   
   For details, see Section 2.2.1, "Installing Oracle JDeveloper".

2. Install the WebCenter Portal Extension for JDeveloper (11.1.1.9.0).
   
   For details, see Section 2.2.2, "Installing the WebCenter Portal Extension for JDeveloper".

3. Download the WebCenter Portal asset development ZIP file (`DesignWebCenterSpaces_11.1.1.9.0.zip`) from Oracle Technology Network:
4. Unzip the content locally (Table 55–1).

**Table 55–1 DesignWebCenterSpaces_11.1.1.9.0.zip - Unzip Folders**

<table>
<thead>
<tr>
<th>Unzip Folders</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\DesignWebCenterSpaces</td>
<td>Includes DesignWebCenterSpaces.jws. This workspace provides the WebCenterSpacesResources project—a design environment for editing and redeploying assets for WebCenter Portal. <strong>Note:</strong> In previous releases, this workspace was also used to configure WebCenter Portal’s <em>shared library list</em>. In this release, you must use the WebCenter Portal Server Extension workspace for shared library development. For details, see Section 55.2, “Developing Task Flows, Data Controls, and Managed Beans for WebCenter Portal.”</td>
</tr>
<tr>
<td>\copy_to_common</td>
<td>Includes a common directory that contains WebCenter Portal WLST commands. WebCenter Portal’s WLST commands are required to upload customized assets directly to the portal server from DesignWebCenterSpaces.jws.</td>
</tr>
<tr>
<td>\copy_to_jdev_ext</td>
<td>Includes oracle.webcenter.portal.jar. Required update to the Oracle JDeveloper WebCenter Portal Extension.</td>
</tr>
</tbody>
</table>

5. 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 55–2).

   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/`.

6. 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 55–2).

d. Copy the `oracle.webcenter.portal.jar` file here.

7. Restart JDeveloper if it is open.

### 55.1.3 Enabling Direct Download and Uploads to WebCenter Portal

If you build or modify skins, page templates, and so on, for a WebCenter Portal application that is live or running in a test environment you can upload your updates directly to WebCenter Portal from JDeveloper. To enable direct updates from the JDeveloper `WebCenterSpacesResources` project you must:

- Specify the name of the host and port where the WebCenter Portal application is running in `config.properties`.
- Check that you have the appropriate roles and permissions to upload the asset to WebCenter Portal. You must have the following:
  - **WebLogic Server role** - `monitor` AND
  - **WebCenter Portal asset permissions** - For example, to upload a new skin you must have the `Create, Edit, Delete Skins` permission.

**Note:** Out-of-the-box, only the default Fusion Middleware administrator (`weblogic`) has all the required permissions.

Configuring the WebCenter Portal 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 WebCenter Portal 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.

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 WebCenter Portal installation, as shown in Table 55–2:

   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>
<thead>
<tr>
<th>Table 55–2 config.properties Settings to Enable Direct Uploads</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Configuration Property</strong></td>
</tr>
<tr>
<td><code>jdeveloper.install.home.directory</code></td>
</tr>
</tbody>
</table>
3. Save your updates to config.properties.

4. Verify and, if necessary, request permissions shown in Table 55–3.

<table>
<thead>
<tr>
<th>Asset Type</th>
<th>Required Role or Permission</th>
<th>Granted By</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shared Assets</td>
<td>■ monitor</td>
<td>WebLogic Server</td>
<td>■ This role enables you to run the WLST scripts which upload assets from JDeveloper to WebCenter Portal. See also, “Administrative Users and Roles” section in Securing Applications with Oracle Platform Security Services.</td>
</tr>
<tr>
<td></td>
<td>■ Create, Edit, Delete AssetType</td>
<td>WebCenter Portal</td>
<td>■ This permission enables you to create and manage various shared assets for WebCenter Portal.</td>
</tr>
<tr>
<td></td>
<td>■ Manage Configuration</td>
<td>WebCenter Portal</td>
<td>■ This application-level permission gives you access to Portal Builder Administration. Note: The Administrator role in WebCenter Portal includes both these permissions, that is, Create, Edit, Delete Assets and Manage Configuration. See also, the section “Managing Application Roles and Permissions” in Administering Oracle WebCenter Portal.</td>
</tr>
</tbody>
</table>
5. To test the connection, upload a sample asset and verify that the asset is available in WebCenter Portal.

For details, see Section 55.1.4, "Opening and Exploring the WebCenterSpacesResources Project" and Section 55.1.6, "Uploading Assets Directly to WebCenter Portal".

See also, Section 55.1.8, "Troubleshooting Asset Round-Trip Development"

### 55.1.4 Opening and Exploring the WebCenterSpacesResources Project

1. Download and unzip the DesignWebCenterSpaces.jws workspace.

   See Section 55.1.2, "Setting Up Your JDeveloper Environment for Asset Development".

2. Open DesignWebCenterSpaces\DesignWebCenterSpaces.jws in JDeveloper (Figure 55–3).

   This workspace contains a project dedicated to asset development called WebCenterSpacesResources.

   **Note:** You can ignore the WebCenterSpacesExtensionLibrary project; this project is not used for asset development.

   **Figure 55–3 Projects in the DesignWebCenterSpaces Workspace**

   ![Application Navigator](image)

   ![DesignWebCenterSpaces](image)

   ![Projects](image)

   ![WebCenterSpacesExtensionLibrary](image)

   ![WebCenterSpacesResources](image)

3. Expand the WebCenterSpacesResources project.
The WebCenterSpacesResources project contains several files and folders (Figure 55–4):

Figure 55–4 Exploring the WebCenterSpacesResources Project in JDeveloper

![Exploring the WebCenterSpacesResources Project in JDeveloper](image)

Table 55–4 describes the files and folders in the WebCenterSpacesResources project.

Table 55–4 Files and Folders in the WebCenterSpacesResources Project

<table>
<thead>
<tr>
<th>Configuration Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\Application Sources</td>
<td>Supporting files and folders.</td>
</tr>
<tr>
<td>\Web Content</td>
<td>Contains assets for WebCenter Portal.</td>
</tr>
<tr>
<td>\oracle\webcenter\siteresources</td>
<td>Directory containing assets. You can develop and manage all your assets for WebCenter Portal under this folder.</td>
</tr>
<tr>
<td>\scopedMD</td>
<td>Contains shared assets and portal assets:</td>
</tr>
<tr>
<td></td>
<td>■ Shared assets are stored under the default scope directory named s8bba98ff_4cb_4bb8_bbee_296c916a23ed. Several sample shared assets are available under this directory.</td>
</tr>
<tr>
<td></td>
<td>■ Portal assets are stored under portal-specific directories with the same name as the portal’s internal ID. No sample portal assets are provided. For more information, see &quot;Shared Assets vs Portal Assets&quot;.</td>
</tr>
<tr>
<td>&lt;Default Scope ID&gt;</td>
<td>Contains sample shared assets.</td>
</tr>
<tr>
<td>\contenttemplates</td>
<td>Includes a sample Content Presenter display template (content-default-list-item-view.jsff) - Sample Content List Item View.</td>
</tr>
<tr>
<td>\navigation</td>
<td>includes a sample navigation (Navigation.xml) - SampleNavigation.</td>
</tr>
<tr>
<td>\pageStyle</td>
<td>Includes a blank page style (TemplateHome.jspx) - Sample Home Page Style.</td>
</tr>
</tbody>
</table>
Shared Assets vs Portal Assets

In WebCenter Portal:

- **Shared assets are available to all portals**
  
  You must develop all shared assets under the WebCenterSpacesResources project directory `oracle/webcenter/siteresources/scopedMD/s8bb98ff_4cbb_40b8_be8e_296c916a23ed`.
  
  To upload shared assets to the WebCenter Portal application you must have WebCenter Portal permissions: Application-Manage Configuration plus Create, Edit, Delete <AssetType>.

- **Portal assets are only available within a particular portal**
  
  You must develop a resource for a specific portal under the appropriate namespace, so that it can be uploaded to that portal:
  
  `oracle/webcenter/siteresources/scopedMD/<space_internal_id>`
  
  Similarly, assets downloaded from an existing portal and imported into JDeveloper, can only be uploaded back to the same portal. You cannot upload such assets to a different portal.
  
  To upload assets to a particular portal you must have the Portal-Manage Configuration permission, plus one of:

  - Create, Edit, Delete Assets (standard permission model)
  - Create, Edit, Delete <AssetType> (advanced permission model)

  If you want to upload assets *directly* from JDeveloper, you must also have the WebLogic Server monitor role. See Section 55.1.3, "Enabling Direct Download and Uploads to WebCenter Portal".

### Table 55–4 (Cont.) Files and Folders in the WebCenterSpacesResources Project

<table>
<thead>
<tr>
<th>Configuration Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\resourceCatalog</td>
<td>Includes a sample catalog (ResourceCatalog.xml) - Sample Home Portal Catalog</td>
</tr>
<tr>
<td>\siteTemplate</td>
<td>Includes a sample page templates (Template.jspx) - Sample Side Navigation</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong>: This directory is the correct directory for developing page templates (even though the directory name is siteTemplate).</td>
</tr>
<tr>
<td>\skin</td>
<td>Includes a sample skin (Skin.css) - Sample Portal FX Skin.</td>
</tr>
<tr>
<td>\taskFlowStyle</td>
<td>Includes files for a sample task flow style (pform.jsff and pform-definition.xml) - Sample Parameter Form.</td>
</tr>
<tr>
<td>\shared</td>
<td>Legacy directory in MDS for images and content used by assets, such as, icons, images, and so on.</td>
</tr>
<tr>
<td></td>
<td>Oracle recommends that you relocate all such artifacts to your content server and that you use a folder structure on your content server specifically for asset artifacts so that content is easy to identify and move, if required.</td>
</tr>
<tr>
<td></td>
<td>If you choose not to move artifacts stored in MDS, your administrator can use the MDS WLST commands exportMetadata/importMetadata to migrate content to and from MDS.</td>
</tr>
<tr>
<td>\Page Flows</td>
<td>Supporting files and folders.</td>
</tr>
</tbody>
</table>
55.1.5 Editing WebCenter Portal Assets in JDeveloper

You do not have to develop WebCenter Portal assets from scratch. Most developers will download an existing asset from the WebCenter Portal application and edit it in JDeveloper.

There are two techniques for bringing assets into JDeveloper. You can either download the asset to JDeveloper directly from WebCenter Portal or you can export the asset to an archive (.ear file) and then import the asset archive to JDeveloper.

For more information, see:

- Section 55.1.5.1, "Downloading an Asset Directly from WebCenter Portal to JDeveloper"
- Section 55.1.5.2, "Downloading an Asset to an Archive from WebCenter Portal"
- Section 55.1.5.3, "Importing a WebCenter Portal Asset to JDeveloper"
- Section 55.1.5.4, "How to Build Assets for WebCenter Portal"

55.1.5.1 Downloading an Asset Directly from WebCenter Portal to JDeveloper

You can download an asset directly from WebCenter Portal into JDeveloper from within the WebCenterSpacesResources project.

1. If you have not done so already, configure WebCenter Portal connection details in config.properties and request the appropriate asset management permissions.

   See also, Section 55.1.3, "Enabling Direct Download and Uploads to WebCenter Portal".

2. In the Navigator, right-click the WebCenterSpacesResources project and then select Download Portal Resource from Server, as shown in Figure 55–5.
3. In the Download Portal Resources dialog, choose to download either a shared asset (Application-Level Resource) or a portal asset (Portal-Level Resource), as shown in Figure 55–6.

See "Shared Assets vs Portal Assets" for information on these selections.
4. To filter the list of possible asset names, select the type of asset you wish to download from the Resource Type menu, as shown in Figure 55–7.

**Figure 55–7  Selecting the Type of Asset to Download**

5. Finally, select the name of the asset you wish to download from the Resource Name menu.

6. Click OK.

55.1.5.2 Downloading an Asset to an Archive from WebCenter Portal

For detailed steps, see the "Downloading an Asset" section in *Building Portals with Oracle WebCenter Portal*.

55.1.5.3 Importing a WebCenter Portal Asset to JDeveloper

When you import an asset from a WebCenter Portal asset archive (.ear file) into the WebCenterSpacesResources project, the asset is uploaded to the appropriate project directory.

- **Shared assets are imported to:**

  WebCenterSpacesResources\oracle\webcenter\siteresources\scopedMD\application_ID\resource_type\resource_ID

  For example, a shared skin with the asset ID gsr5a8c2fccc_bc7f_4cba_9254_36df58d66e60 is created under the application directory s8bb9a8ff_4cbb_40b8_beee_296c916a23ed:

  WebCenterSpacesResources\oracle\webcenter\siteresources\scopedMD\s8bb9a8ff_4cbb_40b8_beee_296c916a23ed\skin\gsr5a8c2fccc_bc7f_4cba_9254_36df58d66e60

- **Portal assets are imported to:**

  WebCenterSpacesResources\oracle\webcenter\siteresources\scopedMD\Space_ID\resource_type\resource_ID

  For example, a portal skin with the asset ID gsr9cecf7f_28b2_4f46_a69a_25beac543382 is created under the portal directory sc48d7f54a06_4fa9_8d51_0e23bed7ae5c:
It's important to know the internal IDs of assets and the parent portal (portal assets only) so that you can locate the appropriate asset folders in JDeveloper. In WebCenter Portal, internal IDs are published in the asset's About dialog (Figure 55–8). For details, see the "Viewing Information About an Asset" section in Building Portals with Oracle WebCenter Portal.

**Figure 55–8 Checking Asset IDs in WebCenter Portal**

Internal IDs are not easy to remember like display names. If you want to identify an imported asset 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.jspx, mycatalog.xml), and then click Update Resource from the right mouse menu (Figure 55–9).
To import a WebCenter Portal asset 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 file) that contains the asset that you want to import.

   If the archive does not exist yet, login to WebCenter Portal and download the asset to a .ear file. For details, see the "Downloading an Asset" section in Building Portals with Oracle WebCenter Portal.

3. Click OK.

4. Navigate to the asset as follows:
   - Locate shared assets under:
     WebCenterSpacesResources\oracle\webcenter\siteresources\scopedMD\s8bbad98ff_4cb840b8_bee0296c9166a23ed
   - Locate portal assets under:
     WebCenterSpacesResources\oracle\webcenter\siteresources\scopedMD\<portal_GUID>

**Note:** If you import an asset that was not originally created in WebCenter Portal or not developed using the WebCenterSpacesResources project the asset will be imported to:

WebCenterSpacesResources\oracle\webcenter\portalapp

In this instance, to consolidate future development, Oracle recommends that you manually move the asset to the appropriate shared or portal asset directory under WebCenterSpacesResources\oracle\webcenter\siteresources\scopedMD.
55.1.5.4 How to Build Assets for WebCenter Portal
To learn more about building assets, read the appropriate chapter:

- Page Templates - Developing Page Templates
- Page Styles - Developing Page Styles
- Skins - Developing Skins
- Navigations - Developing a Navigation Model
- Content Presenter Display Templates - Creating Content Presenter Display Templates
- Resource Catalogs - Developing Resource Catalogs
- Task Flow Styles - Developing Task Flow Styles

55.1.6 Uploading Assets Directly to WebCenter Portal

If WebCenter Portal is up and running you can deploy new or updated assets directly to WebCenter Portal from your JDeveloper environment.

You must have appropriate asset management permissions in WebCenter Portal to upload assets in this way. If you do not have the correct privileges, you are not allowed to upload assets. See also Section 55.1.3, "Enabling Direct Download and Uploads to WebCenter Portal".

1. If you have not done so already, configure WebCenter Portal connection details in config.properties and request the appropriate asset management permissions.

   See also, Section 55.1.3, "Enabling Direct Download and Uploads to WebCenter Portal".

2. In JDeveloper, open the WebCenterSpacesResources project, and navigate to the asset you want to upload.

   For example, to upload the sample shared navigation SampleNavigation, navigate to (Figure 55–10):

   WebCenterSpacesResources\Web Content\oracle\webcenter\siteresources\scopedMD\s8bba98ff_4cbb_40b8_beee_296c916a23ed\navigation\gsrcc8db1c_6161_4bb8_8764_127b4ecce01b
Developing Assets for WebCenter Portal

Figure 55–10

Navigating to WebCenter Portal Resource Folders

3.

Verify that the asset is complete.

4.

Ensure that any artifact referenced by the asset, such as images, icons, and so on, is
available and accessible on the target content server.

5.

Right-click the file associated with the asset you want to upload to WebCenter
Portal, and choose Upload Portal Resource to Server.
There is a single file associated with most asset types. The exception is task flow
style which has two files: definition.xml and .jsff. To upload a task flow style
you must right-click the definition.xml file.

6.

If prompted, enter your WebCenter Portal 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_<asset_
type_id>.log.

Figure 55–11

7.

Asset Upload Complete Message

Review the upload log file.
The following message indicates that the upload was successful.
Imported <temp_log_directory>\<asset_type_id>.ear
If you do not see this message, refer to Section 55.1.8, "Troubleshooting Asset
Round-Trip Development"

8.

Log in to WebCenter Portal and verify that the asset is available as expected.
a.

Log in to WebCenter Portal.

55-16 Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper


Developing Assets for WebCenter Portal

b. Navigate to the appropriate Asset page.
   To verify shared assets, navigate to the following URL:
   http://host:port/webcenter/portal/builder/assets

   To verify portal assets, navigate to the following URL:
   http://host:port/webcenter/portal/portalName/admin/assets

c. Select the appropriate asset type from the panel on the left, and verify that the asset you just uploaded is available and working as expected.
   Dependent images and so on, must be available and accessible on your content server. If an image or file is missing, check that you copied it to the appropriate directory.

55.1.7 Exporting WebCenter Portal Assets to an Archive

If for some reason you do not want or cannot upload your asset directly to WebCenter Portal from JDeveloper, you can export the asset to an archive (.ear file) and import the asset at some other time.

To export a WebCenter Portal asset to an archive:

1. In the Application Navigator, navigate to the asset directory and drill down to the asset file (for example, myskin.css, mypagetemplate.jspx, mycatalog.xml).

2. Click Export Portal Resource 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. Click OK.

The steps to upload the asset to WebCenter Portal from the archive are available in the "Uploading an Asset" section in Building Portals with Oracle WebCenter Portal.

Note: Artifacts referenced by the asset, such as images, icons, and so on, are not included in the asset archive. If you have not done so already, ensure that all such artifacts are available and accessible on the target content server.

55.1.8 Troubleshooting Asset Round-Trip Development

This troubleshooting section includes the following subsections:

- Examining the Asset Upload Log File
- Unable to Upload Assets to WebCenter Portal - Access Denied
- Unable to Upload Assets to WebCenter Portal - java.lang.Exception

Examining the Asset Upload Log File

When you upload an asset to WebCenter Portal, progress details are recorded in a log file. The name of the log file includes the type of asset you are uploading and a unique ID, in the format—upload_asset_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:
The following log entry indicates that the upload process was successful.

Imported `<temp_log_directory>`\`<asset_type_id>`.ear

Insufficient permissions result in "Access denied" or "java.lang.Exception" errors, as described below.

**Unable to Upload Assets to WebCenter Portal - Access Denied**
The following error displays in the asset upload log file if you do not have the monitor role:


The monitor role is a Weblogic Server role. This role enables you to run the WLST scripts which upload assets from JDeveloper to WebCenter Portal. Ask your system administrator to grant you this role through the WebLogic Server Administration Console. See also, the "Administrative Users and Roles" section in *Securing Applications with Oracle Platform Security Services*.

**Unable to Upload Assets to WebCenter Portal - java.lang.Exception**
The following error displays in the asset upload log file if you do not have permission to manage any assets in WebCenter Portal or the particular type of asset 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.SiteResourcesHelper.updateResource(SiteResourcesHelper.java)

Ask your WebCenter Portal administrator to grant you appropriate permissions through Portal Builder Administration. Or, if the asset you are trying to upload is for a particular portal, contact the portal moderator to request the required permission. See also, Table 55–3, "Permissions to Upload and Manage Assets Through JDeveloper".

### 55.2 Developing Task Flows, Data Controls, and Managed Beans for WebCenter Portal

This section describes how you can package and deploy your ADF task flows, data controls, and managed beans in your own custom shared libraries (WAR files) for use in WebCenter Portal.

WebCenter Portal provides a JDeveloper template called *WebCenter Portal Server Extension* that enables you to specify one or more custom shared libraries that want to use with WebCenter Portal.
You can also use the same WebCenter Portal Server Extension template as a starter template to build and deploy ADF library components, such as task flows, data controls, and managed beans for WebCenter Portal.

This section includes the following subsections:

- **Section 55.2.1, "Understanding the WebCenter Portal Server Extension Template"**
- **Section 55.2.2, "Understanding Shared Library Development for WebCenter Portal"**
- **Section 55.2.3, "Creating a WebCenter Portal Server Extension Workspace"**
- **Section 55.2.4, "Registering Additional Shared Libraries With WebCenter Portal"**
- **Section 55.2.5, "Deploying Extensions to the WebCenter Portal Shared Library (extend.spaces.webapp.war)"**
- **Section 55.2.6, "Developing ADF Library Components for WebCenter Portal Using the PortalExtension Project"**
- **Section 55.2.7, "Troubleshooting WebCenter Portal Shared Library Deployment"**

### 55.2.1 Understanding the WebCenter Portal Server Extension Template

The WebCenter Portal Server Extension template creates a workspace with two projects:

- **PortalSharedLibrary** – A project that when deployed creates `extend.spaces.webapp.war`, a shared library that:
  
  - References one or more custom shared libraries for WebCenter Portal.
  
  - Wraps the ADF Library JAR created from the PortalExtension project (only if used).

  **Note:** Do not add any code to this project; this project only contains descriptor files that reference other shared libraries and/or includes ADF Library JARs.

- **PortalExtension** – A starter project in which you create custom ADF components, such as task flows, data controls, and managed beans. You can deploy this project to an ADF Library, which is added to the `extend.spaces.webapp.war` shared library.

  The components you create in a PortalExtension project typically require Java, ADF, and other related software development skills.

---

**Note:** Oracle recommends that you use the WebCenter Portal Server Extension template to configure shared libraries for WebCenter Portal (11.1.1.8.0).

For backward compatibility, if you developed and deployed custom shared libraries using SampleWebCenterSpacesExtension.jws and DesignWebCenterSpaces.jws then this approach continues to work as previously documented. You must only use one approach, you cannot use the WebCenter Portal Server Extension template and DesignWebCenterSpaces.jws.
This section describes:

- Section 55.2.1.1, "How the WebCenter Portal Server Extension Workspace Is Organized"
- Section 55.2.1.3, "The PortalExtension Project"
- Section 55.2.1.2, "The PortalSharedLibrary Project"

### 55.2.1.1 How the WebCenter Portal Server Extension Workspace Is Organized
The WebCenter Portal Server Extension workspace contains two default projects PortalExtension and PortalSharedLibrary each containing specific libraries and default files (Figure 55–13).

**Figure 55–13 WebCenter Portal Server Extension Workspace in Application Navigator**

![Application Navigator](image)

### 55.2.1.2 The PortalSharedLibrary Project
The PortalSharedLibrary project, shown in Figure 55–14, enables you to register one or more custom shared libraries that you want to use in WebCenter Portal and also provides a convenient mechanism for deploying task flows, data controls, and managed beans (developed in the PortalExtension project) to a managed server running WebCenter Portal.

The PortalSharedLibrary project includes a WAR deployment profile called extend.spaces.webapp. When you deploy to extend.spaces.webapp.war, all the custom shared libraries that you register (in weblogic.xml) and any ADF library that you create and using a PortalExtension project are automatically included as dependencies of extend.spaces.webapp.war.

See Section 55.2.4, "Registering Additional Shared Libraries With WebCenter Portal" and Section 55.2.5, "Deploying Extensions to the WebCenter Portal Shared Library (extend.spaces.webapp.war)".

**Note:** Do not add any code to the PortalSharedLibrary project; this project only contains descriptor files that reference other shared libraries or include ADF Library JARs.

For general information on shared libraries, see Section 55.2.2.2, "General Documentation for Shared Library Deployment".
55.2.1.2.1 Versioning extend.spaces.webapp The PortalSharedLibrary project contains a MANIFEST.MF file. Each time you update and deploy extend.spaces.webapp.war, edit this MANIFEST.MF file to increment the shared library implementation version. With each iteration, you must increment the Implementation-Version number in the MANIFEST.MF file, as shown in Example 55–1. The default implementation version is 11.1.2.

**Example 55–1 Manifest File for extend.spaces.webapp**

Manifest-Version: 1.0  
Archiver-Version: Plexus Archiver  
Created-By: Apache Maven  
Built-By: builder  
Build-Jdk: 1.6.0_20  
Extension-Name: extend.spaces.webapp  
Implementation-Label: 11.1.1.9.0  
Implementation-Title: extend.spaces.webapp  
Implementation-Vendor: Oracle  
Implementation-Version: 11.1.2  
Specification-Title: extend.spaces.webapp  
Specification-Vendor: Oracle  
Specification-Version: 11.1.1

For detailed steps on how to change the implementation version, see Section 55.2.5, "Deploying Extensions to the WebCenter Portal Shared Library (extend.spaces.webapp.war)."

55.2.1.3 The PortalExtension Project

The PortalExtension project is a standard ADF project where you can create custom components like task flows, data controls, and managed beans.

A PortalExtension project is deployed to an ADF Library. This library is automatically added as a dependency to the extend.spaces.webapp.war shared library file generated from the PortalSharedLibrary project. See Section 55.2.6, "Developing ADF Library Components for WebCenter Portal Using the PortalExtension Project."

For more information on ADF Libraries, see Section 55.2.2.1, "General Documentation for ADF Library Development".
55.2.2 Understanding Shared Library Development for WebCenter Portal

WebCenter Portal includes the standard shared library `extend.spaces.webapp.war`. This WAR file can include a deployment descriptor (`weblogic.xml`) which references other shared libraries containing custom ADF library components.

If you have custom code or task flows deployed in several shared libraries from multiple sources that you want to use in WebCenter Portal you can list them in `extend.spaces.webapp.war`, as illustrated in Figure 55–16.

This development model provides an easy way to utilize additional shared libraries in WebCenter Portal from multiple contributors, including developers, customers, partners.

Whenever you deploy a new shared library that includes extensions for WebCenter Portal you must register the name of your shared library with WebCenter Portal and redeploy `extend.spaces.webapp.war`. For details, see Section 55.2.4, "Registering Additional Shared Libraries With WebCenter Portal".

This Guide does not teach you how to develop ADF task flows, data controls, and managed beans and package them into custom shared libraries as such techniques and procedures are not specific to WebCenter Portal deployments. If you are new to shared library development, Oracle recommends that you familiarize yourself with the documentation listed at:
This Guide explains how to reference shared libraries that you want to use in WebCenter Portal using the WebCenter Portal Server Extension template. Table 55–5 provides an overview of the ADF library and shared library development processes, highlighting information that is specific to WebCenter Portal and pointing to other documentation where applicable.

<table>
<thead>
<tr>
<th>Subject</th>
<th>WebCenter Portal Documentation</th>
<th>General Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developing ADF library components</td>
<td>WebCenter Portal supports several component types deployed to ADF Library JARs: task flows, data controls, and managed beans. Create task flows, data controls, or managed beans for WebCenter Portal the same way as any standard ADF Library component. Alternatively, use the starter project supplied with WebCenter Portal. See Section 55.2.6, “Developing ADF Library Components for WebCenter Portal Using the PortalExtension Project”.</td>
<td>The section “Introduction to Reusable Components” in Fusion Developer’s Guide for Oracle Application Development Framework describes various types of reusable ADF components, such as a task flows, that you can deploy to an ADF Library JAR. Note that the subsection &quot;Extension Libraries” does not apply to WebCenter Portal. See also, the section &quot;Creating Reusable Components” in Fusion Developer’s Guide for Oracle Application Development Framework.</td>
</tr>
<tr>
<td>- Naming conventions</td>
<td>Every ADF Library JAR file that you want to use with WebCenter Portal must have a unique file name. You cannot add two ADF Library JAR files with the same name even if their content is completely different.</td>
<td>See also, the section “Naming Conventions” in Fusion Developer’s Guide for Oracle Application Development Framework.</td>
</tr>
<tr>
<td>- Managing connections</td>
<td>If your ADF Library JAR utilizes a connection you must manually register that connection with WebCenter Portal using the same connection name and details (just including the connection within the ADF Library JAR does not expose that connection in WebCenter Portal). To create connections for WebCenter Portal, you must use Fusion Middleware Control or WLST commands. To add a WebCenter Portal specific connection, refer to the &quot;Managing Tools and Services&quot; section in Administering Oracle WebCenter Portal. To add a database connection, refer to the &quot;Creating and Managing JDBC Data Sources&quot; section in the Administrator’s Guide. To add a Web Service connection, configure the ADFConnections MBean using the System MBean Browser. Refer to the &quot;System MBean Browser&quot; section in Administering Oracle WebCenter Portal and the &quot;Web Service Connection&quot; section in the Administrator’s Guide for Oracle Application Development Framework.</td>
<td>See also, the section “Naming Considerations for Connections” in Fusion Developer’s Guide for Oracle Application Development Framework.</td>
</tr>
<tr>
<td>- Including descriptors</td>
<td>Descriptors in custom shared libraries, such as web.xml and application.xml, merge with corresponding descriptors deployed with the WebCenter Portal application. To avoid corrupting your WebCenter Portal installation, Oracle recommends that you do not include descriptor files in your custom shared libraries.</td>
<td></td>
</tr>
</tbody>
</table>
Each time you redeploy the WebCenter Portal shared library WAR file (extend.spaces.webapp.war) to add an ADF library JAR or to reference a custom shared library you must increment the shared library implementation version number.

For details, see Section 55.2.1.2.1, "Versioning extend.spaces.webapp.war" and Section 55.2.5, "Deploying Extensions to the WebCenter Portal Shared Library (extend.spaces.webapp.war)".

**Packaging components into ADF Library JAR**

Package your task flow, data control, or managed bean into a standard ADF Library JAR.

Alternatively, use the starter PortalExtension project. For details, see Section 55.2.6, "Developing ADF Library Components for WebCenter Portal Using the PortalExtension Project".

The section "Packaging a Reusable ADF Component into an ADF Library" in Fusion Developer's Guide for Oracle Application Development Framework describes how to package a reusable component, such as a task flow, to a ADF Library file.

Note that the subsection 'How to Place and Access JDeveloper JAR Files' does not apply to WebCenter Portal.

**Adding ADF library components to a shared library**

To use ADF Library components in WebCenter Portal, you must:

1. Deploy your ADF Library components to a shared library WAR file.
   
   See, the "Creating Shared Java EE Libraries and Optional Packages" section in Developing Applications for Oracle WebLogic Server.

2. Deploy your shared library to the managed server on which WebCenter Portal is deployed.

3. Register your shared library in the WebCenter Portal shared library (extend.spaces.webapp.war).


5. Use Fusion Middleware Control or WLST to create any connections that your custom ADF Library components require.

6. Restart the managed server on which WebCenter Portal is deployed.

**Note:** You can add ADF Library components directly to the WebCenter Portal shared library WAR (extend.spaces.webapp.war) if you want.

For details, see Section 55.2.5, "Deploying Extensions to the WebCenter Portal Shared Library (extend.spaces.webapp.war)".

The section "Adding ADF Library Components into Projects" in Fusion Developer's Guide for Oracle Application Development Framework describes how to deploy ADF library components, such as a task flow, to a project.

The information in this section does not apply to WebCenter Portal.
Developing Task Flows, Data Controls, and Managed Beans for WebCenter Portal

Developing Components for WebCenter Portal Using JDeveloper

55.2.2.1 General Documentation for ADF Library Development

- For information about creating ADF task flows, data controls, and managed beans, refer to the Fusion Developer’s Guide for Oracle Application Development Framework.

- See also, the "Reusing Application Components" section in Fusion Developer’s Guide for Oracle Application Development Framework.

55.2.2.2 General Documentation for Shared Library Deployment

- For additional information on versioning of shared libraries, see the "Deploying Shared Java EE Libraries and Dependent Applications" section in Developing Applications for Oracle WebLogic Server.

- See also, the "Creating Shared Java EE Libraries and Optional Packages" section in Developing Applications for Oracle WebLogic Server.

---

Table 55–5 (Cont.) Developing and Deploying ADF Libraries and Shared Libraries for WebCenter Portal

<table>
<thead>
<tr>
<th>Subject</th>
<th>WebCenter Portal Documentation</th>
<th>General Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Using ADF library components</td>
<td>An ADF Library adapter in WebCenter Portal’s resource catalog automatically picks up any task</td>
<td>The section &quot;What You May Need to Know About Using ADF Library Components&quot; in Fusion</td>
</tr>
<tr>
<td></td>
<td>flows, data controls and managed beans that you deploy and reference through extend.spaces.webapp</td>
<td>Developer’s Guide for Oracle Application Development Framework describes how various</td>
</tr>
<tr>
<td></td>
<td>.war and adds them to WebCenter Portal’s resource registry. Custom components in the</td>
<td>different ADF Library component types appear in JDeveloper. For WebCenter Portal, only</td>
</tr>
<tr>
<td></td>
<td>resource registry are exposed to WebCenter Portal users through resource catalogs as</td>
<td>the information pertaining to task flows, data controls, and managed beans applies.</td>
</tr>
<tr>
<td></td>
<td>components that you can drop onto portal pages.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1. Log in to WebCenter Portal.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2. Add your custom component to a resource catalog.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>See, the “Adding a Resource to a Resource Catalog” section in Building Portals with Oracle</td>
<td></td>
</tr>
<tr>
<td></td>
<td>WebCenter Portal.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3. Open a page with access to the resource catalog.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>See, the “Adding a Component to a Page” section in Building Portals with Oracle WebCenter</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Portal.</td>
<td></td>
</tr>
<tr>
<td>Creating a shared library</td>
<td>The WebCenter Portal Server Extension template provides a deployment profile for the WebCenter</td>
<td>The section &quot;Creating Shared Java EE Libraries&quot; in Developing Applications for Oracle</td>
</tr>
<tr>
<td></td>
<td>For details, see Section 55.2.3, 'Creating a WebCenter Portal Server Extension Workspace'.</td>
<td></td>
</tr>
<tr>
<td>Deploying a shared library</td>
<td>The WebCenter Portal Server Extension template enables you to redeploy the WebCenter Portal</td>
<td>The section &quot;Deploying Shared Java EE Libraries and Dependent Applications&quot; in</td>
</tr>
<tr>
<td>WAR</td>
<td>shared library WAR extend.spaces.webapp.war.</td>
<td>Developing Applications for Oracle WebLogic Server describes how to deploy any shared</td>
</tr>
<tr>
<td></td>
<td>For details, see Section 55.2.5, 'Deploying Extensions to the WebCenter Portal Shared Library</td>
<td>library WAR file.</td>
</tr>
<tr>
<td></td>
<td>(extend.spaces.webapp.war)’.</td>
<td></td>
</tr>
<tr>
<td>Reverting to a previous library</td>
<td>Use the WebLogic Server Administration Console to revert to a previous shared library</td>
<td>The section &quot;Removing an ADF Library JAR from a Project&quot; in Fusion Developer’s Guide</td>
</tr>
<tr>
<td>version</td>
<td>WAR version or remove unwanted versions.</td>
<td>for Oracle Application Development Framework describes how remove ADF Library</td>
</tr>
<tr>
<td></td>
<td>For details, see Section 55.2.5.3, 'Reverting to a Previous WebCenter Portal Shared Library</td>
<td>components in JDeveloper.</td>
</tr>
<tr>
<td></td>
<td>Version’.</td>
<td></td>
</tr>
</tbody>
</table>

---

Table 55–5 (Cont.) Developing and Deploying ADF Libraries and Shared Libraries for WebCenter Portal

55-25
55.2.3 Creating a WebCenter Portal Server Extension Workspace

This section explains how to create a WebCenter Portal Server Extension workspace in JDeveloper in which you can:

- Register one or more custom shared libraries to use with WebCenter Portal.
- Create and deploy a custom ADF component for WebCenter Portal such as a task flow, data control, or managed bean.

To create a WebCenter Portal Server Extension workspace:

1. Download and install Oracle JDeveloper 11g (11.1.1.9.0).
   For details, see Section 2.2.1, "Installing Oracle JDeveloper".
2. Install the WebCenter Portal Extension for JDeveloper (11.1.1.9.0).
   For details, see Section 2.2.2, "Installing the WebCenter Portal Extension for JDeveloper".
3. Open JDeveloper and choose File > New.
4. In the New Gallery dialog, open the General node and select Applications.
5. In the Items list, select WebCenter Portal Server Extension, and click OK.
   See Figure 55–17.

Figure 55–17 New Gallery Dialog

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

6. In the Create WebCenter Portal Server Extension dialog, enter a name for the workspace in the Application Name field, as shown in Figure 55–18.
7. In the Directory field, enter a path to the base directory in which to store the workspace, or accept the default path.

For example:

'/scratch/jdeveloper/mywork/MyPortalExtension'

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

8. Optionally, enter an Application Package Prefix.

This prefix is used for Java classes, interfaces, and packages created within the workspace. For example, a common prefix pattern is `com.companyDomainName`, like `com.oracle`.

**Tip:** At this point, you could click Finish to create a project with default portal project names and directories. The following steps continue and review the rest of the wizard pages.

9. Click Next.

10. In the Project 1 Name part of the wizard, enter a Project Name, or keep the default name, PortalExtension, as shown in Figure 55–19.

This project provides all of the project technology components that are required for building and testing custom ADF components for WebCenter Portal (task flows, data controls, and managed beans).
11. If you want to change the default directory for the project, enter it or use the Browse button to select a directory.

   Also called technology scopes, project technologies are a collection of libraries and files designed to provide specific functionality within a project.

   **Note:** Technology scopes are attributes on the project that can be used to identify the different technologies used for that particular project. These attributes are used only within JDeveloper to assist you as you work. With technology scopes, the choices presented to you in the New Gallery and in the menus and palettes are filtered so that you see only those items that are most relevant to you as you work. Technology scopes have no effect on the data in the project itself. For more information, see refer to JDeveloper Online Help.

13. Optionally, select the Generated Components and Associated Libraries tabs to view additional configuration files and libraries that will be added to the project.

14. Click Next.

15. Optionally, edit the default package name and directories for Java source files and output class files for the project, as shown in Figure 55–21.
16. In the **Project 2 Name** part of the wizard, enter a project name, or keep the default name, **PortalSharedLibrary**, as shown in **Figure 55–21**. This project allows you to register one or more shared libraries that you want to use in WebCenter Portal, as well as deploy custom ADF components built using the PortalExtension project. See Section 55.2.5, "Deploying Extensions to the WebCenter Portal Shared Library (extend.spaces.webapp.war)"

17. If you want to change the default directory for the project, enter it or use the **Browse** button to select a directory.
18. Optionally add Project Technologies, also called technology scopes.

**Note:** Technology scopes are attributes on the project that can be used to identify the different technologies used for that particular project. These attributes are used only within JDeveloper to assist you as you work. With technology scopes, the choices presented to you in the New Gallery and in the menus and palettes are filtered so that you see only those items that are most relevant to you as you work. Technology scopes have no effect on the data in the project itself. For more information, see refer to JDeveloper Online Help.

19. Optionally, select the Generated Components and Associated Libraries tabs to view additional configuration files and libraries that will be added to the project.

20. Click Finish.

### 55.2.4 Registering Additional Shared Libraries With WebCenter Portal

If you have one or more shared libraries containing custom task flows, data controls or managed beans that you want to use them in WebCenter Portal, you must register them in the *weblogic.xml* file associated with WebCenter Portal's shared library `extend.spaces.webapp.war`.

1. Select the PortalSharedLibrary project (Figure 55–22).

   If you have not created a WebCenter Portal Extension workspace yet, refer to Section 55.2.3, “Creating a WebCenter Portal Server Extension Workspace”.

**Figure 55–22 PortalSharedLibrary Project**

2. If the deployment descriptor *weblogic.xml* does not exist yet under `PortalSharedLibrary\Web Content\WEB-INF`, create the deployment descriptor as follows:

   a. Select New > Deployment Descriptors > WebLogic Deployment Descriptor
   b. Select OK.
   c. Select *weblogic.xml* from the list (Figure 55–23).
d. Select Finish.

3. Open weblogic.xml (under PortalSharedLibrary\Web Content\WEB-INF). Initially, no additional shared libraries are listed in the file.

4. In the Overview page, select Libraries.

5. Specify the Library Name for each shared library that you want to use in WebCenter Portal.

   You can register a single shared library or multiple shared libraries, as shown in Figure 55–24.

   Click the Help icon for more information, if required.

Figure 55–24  weblogic.xml - Multiple Shared Library References

6. Ensure each shared library that you reference is deployed on the WebCenter Portal managed server (named WC_Spaces by default).
7. Redeploy a new version of the WebCenter Portal shared library WAR (extend.spaces.webapp.war) that includes your shared library references. Remember to increment the version number in MANIFEST.MF. For details, see Section 55.2.5, "Deploying Extensions to the WebCenter Portal Shared Library (extend.spaces.webapp.war)."

55.2.5 Deploying Extensions to the WebCenter Portal Shared Library (extend.spaces.webapp.war)

This section explains how to deploy extensions to the WebCenter Portal shared library (extend.spaces.webapp.war) where your extensions are either:

- Existing shared libraries containing task flows, data controls, or managed beans that you have registered in weblogic.xml. See Section 55.2.4, "Registering Additional Shared Libraries With WebCenter Portal"

- An ADF Library JAR created from the PortalExtension project and included in the WebCenter Portal shared library. See Section 55.2.6, "Developing ADF Library Components for WebCenter Portal Using the PortalExtension Project"

This section includes the following topics:

- Section 55.2.5.1, "Deploying Extensions Directly to the Portal Server"
- Section 55.2.5.2, "Deploying Extensions to an Archive"
- Section 55.2.5.3, "Reverting to a Previous WebCenter Portal Shared Library Version"

55.2.5.1 Deploying Extensions Directly to the Portal Server

Before deploying extensions to WebCenter Portal from JDeveloper:

- Verify that you have at least the WebLogic Monitor role.
  For information on roles, see the "Users, Groups, And Security Roles" section in Securing Resources Using Roles and Policies for Oracle WebLogic Server.

- Ensure a connection exists to the managed server that is running WebCenter Portal.
  For information about creating a connection see Chapter 7.3.3.4, "Creating a WebLogic Managed Server Connection."

- Review background information on shared libraries in the "Creating Shared Java EE Libraries and Optional Packages" section in Developing Applications for Oracle WebLogic Server.

To deploy extensions for WebCenter Portal from JDeveloper:

1. In JDeveloper, open the workspace containing extensions to be deployed. See also Section 55.2.4, "Registering Additional Shared Libraries With WebCenter Portal" and Section 55.2.6, "Developing ADF Library Components for WebCenter Portal Using the PortalExtension Project".

2. In the Application Navigator, navigate to the PortalSharedLibrary > Application Sources > META-INF folder.

3. Open the MANIFEST.MF file in a text editor.

**Note:** You must increment the shared library Implementation-Version number each time you update and redeploy the WebCenter Portal shared library extend.spaces.webapp.war. Otherwise, an error is reported from WebLogic Server in the Deployment - Log tab. For more information, see Section 55.2.1.2, "The PortalSharedLibrary Project."

5. In the Application Navigator, right-click the PortalSharedLibrary project, select Deploy, and then select extend.spaces.webapp (Figure 55–25).

**Note:** Note that extend.spaces.webapp is the name of the deployment profile for the shared library that you are deploying to the server. The shared library created by a subsequent deployment action is called extend.spaces.webapp.war.

**Figure 55–25  Selecting the Shared Library extend.spaces.webapp**

6. In the Deploy dialog, select Deploy to Application Server and click Next (Figure 55–26).
7. Select the connection that points to the WebLogic Server where you want to deploy the shared library, for example, WC_Spaces, and click Next.

8. In the Deploy extend.spaces.webapp dialog, select Deploy to selected instances in the domain, as shown in Figure 55–27.

9. In the list of servers, select the managed server on which WebCenter Portal is deployed, as shown in Figure 55–28.

Note: The name of the managed server depends on how the system administrator set up the server. By default, the name is WC_Spaces; however, this name might be different on your system.
10. Select **Deploy as a shared library**, as shown in Figure 55–29

**Note:** This step is critical: you **must** select **Deploy as a shared library**, otherwise, the deployment will not be successful.

11. Click **Next**.

12. On the Deployment Summary screen, verify the deployment options, and click **Finish** to deploy the portal extension to the server.
13. Open the Deployment -Log tab to examine for any deployment issues. If deployment is successful, you should see this log entry: "Application Deployed Successfully," as shown in Chapter 55–30, "Deployment - Log Tab."

Figure 55–30 Deployment - Log Tab

<table>
<thead>
<tr>
<th>Time</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>10:29:57 AM</td>
<td>Running dependency analysis...</td>
</tr>
<tr>
<td>10:29:57 AM</td>
<td>Moving MDF-INF/afdc-config.war to MTA-INF/afdc-config.war</td>
</tr>
<tr>
<td>10:29:57 AM</td>
<td>Writing Archive Module to /scratch/afdc/nxs/scratch/shared/afdc-config.war</td>
</tr>
<tr>
<td>10:30:00 AM</td>
<td>Deploying generate...</td>
</tr>
<tr>
<td>10:30:00 AM</td>
<td>Writing Web Application Module to /scratch/afdc/nxs/scratch/shared/afdc-config.war</td>
</tr>
<tr>
<td>10:30:00 AM</td>
<td>Moving Application...</td>
</tr>
<tr>
<td>10:30:05 AM</td>
<td>Application Deployed Successfully.</td>
</tr>
<tr>
<td>10:30:03 AM</td>
<td>---- Deployment finished. ----</td>
</tr>
</tbody>
</table>

14. To verify the new deployment, log in to the WebLogic Server Administration Console, navigate to the Deployments Overview page, and check the implementation version displayed.

a. Log in to the WebLogic Server Administration Console.

b. Click Deployments, and locate extend.spaces.webapp.

c. Note the entries for the extend.spaces.webapp shared library and verify the one with the implementation version that you updated previously in the MANIFEST.MF file. See also Section 55.2.1.2.1, “Versioning extend.spaces.webapp.war.”

Note: WebLogic Server only uses the latest shared library version when an application starts up. If you go through several "change-build-deploy-test" iterations, incremental versions are retained by default. You can use the Administration Console to remove unwanted shared library versions if you want, but it is recommended that you retain the first version (extend.spaces.webapp(11.1.1,11.1.1)) as a backup so you can revert to the default behavior if necessary.

15. (Optional) Create any connections that your custom ADF Library components may require:

- To add a database connection, refer to the "Creating and Managing JDBC Data Sources" section in Administrator’s Guide.

- To add a Web Service connection, configure the ADFConnections MBean using the System MBean Browser in Fusion Middleware Control. Refer to the "System MBean Browser" section in Administering Oracle WebCenter Portal and the "Web Service Connection" section in the Administrator’s Guide for Oracle Application Development Framework.

- To add a connection type specific to WebCenter Portal, for example to a portlet producer or content repository connection, refer to the "Managing Tools and Services" section in Administering Oracle WebCenter Portal.

16. Restart the managed server on which WebCenter Portal is running.

This step is required for the task flows, data controls, and managed beans in the shared library to show up in the WebCenter Portal Resource Registry. For more
Developing Task Flows, Data Controls, and Managed Beans for WebCenter Portal

information, see the "Starting and Stopping Managed Servers for WebCenter Portal Application Deployments" section in Administering Oracle WebCenter Portal.

---

**Note:** The name of the managed server depends on how the system administrator set up the server. By default, the name is WC_Spaces; however, this name might be different on your system.

---

17. Log in to WebCenter Portal and verify that all newly deployed task flows, data controls, or managed beans are available in the Resource Registry.

Once successfully deployed components are added to the Resource Registry, you can add them to resource catalogs where they are made available for drag and drop onto portal pages. See also, the "About the Resource Registry" section in Building Portals with Oracle WebCenter Portal.

55.2.5.2 Deploying Extensions to an Archive

If you do not have a connection to the portal server or permissions to deploy the extend.spaces.webapp shared library on the portal server, you can deploy extensions to the extend.spaces.webapp shared library to an archive (.war file) and give the .war file to an administrator for deployment to a managed server running WebCenter Portal.

To deploy extensions to the extend.spaces.webapp shared library to an archive:

1. In JDeveloper, open the workspace containing extensions to be deployed.

   See also Section 55.2.4, "Registering Additional Shared Libraries With WebCenter Portal" and Section 55.2.6, "Developing ADF Library Components for WebCenter Portal Using the PortalExtension Project".

2. In the Application Navigator, navigate to the PortalSharedLibrary > Application Sources > META-INF folder.

3. Open the MANIFEST.MF file in a text editor.

4. If required, increment the Implementation-Version number. The default version is Implementation-Version: 11.1.2.

   **Note:** You do not need to increment the version number every time you generate a new archive (.war file). You must however, increment the shared library Implementation-Version number each time you deploy a new extension version to the server or an error is reported from WebLogic Server in the Deployment - Log tab. For more information, see Section 55.2.1.2, "The PortalSharedLibrary Project."

5. In the Application Navigator, right-click the PortalSharedLibrary project, select **Deploy** and then select the deployment profile **extend.spaces.webapp**.

   **Note:** Note that extend.spaces.webapp is the name of the deployment profile for the shared library that you are deploying to a file. The shared library created by a subsequent deployment action is called extend.spaces.webapp.war.

6. In the Deploy dialog, select **Deploy to WAR**, and then click **Next** (Figure 55–31).
7. Review the Summary page and click Finish.

The location of the .war file displays in the "Deployment - Log Tab." This new shared library .war file version can now be deployed to any server instance on which WebCenter Portal is deployed.

For More Information
See "Deploying Applications" in JDeveloper Online Help. See also the "Creating Shared Java EE Libraries and Optional Packages" section in Developing Applications for Oracle WebLogic Server.

55.2.5.3 Reverting to a Previous WebCenter Portal Shared Library Version
If there is a problem with the latest WebCenter Portal shared library 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 WebLogic Server 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 WebCenter Portal 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 WebCenter Portal 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-of-the-box version if necessary.

1. Stop the managed server on which WebCenter Portal is deployed.

   For details, see the "Starting and Stopping Managed Servers for WebCenter Portal Application Deployments" section in Administering Oracle WebCenter Portal.

2. Log in to the Weblogic Server Administration Console.

3. Click Deployments, then locate and select the latest extend.spaces.webapp version you want to remove.

4. Click Delete to undeploy the latest shared library version.
To revert to the original extend.spaces.webapp.war shared library, delete all other shared library versions, except for extend.spaces.webapp.war version 11.1.1.

5. Start the managed server on which WebCenter Portal is deployed:

For details, see the "Starting and Stopping Managed Servers for WebCenter Portal Application Deployments" section in Administering Oracle WebCenter Portal.

55.2.6 Developing ADF Library Components for WebCenter Portal Using the PortalExtension Project

Developing an ADF Library for WebCenter Portal is exactly the same as any other ADF Library. If you are not familiar with ADF library development, refer to the documentation links in Section 55.2.2.1, "General Documentation for ADF Library Development".

Development teams typically build and deploy ADF libraries to a set of well defined shared libraries that they can reuse across multiple applications, such as WebCenter Portal.

To help you get started, Oracle provides the PortalExtension project in which you can build ADF libraries containing task flows, data controls, and managed beans and deploy them to WebCenter Portal's own shared library extend.spaces.webapp.war.

If you want to use the PortalExtension project, follow these steps:

   For details, see Section 55.2.3, "Creating a WebCenter Portal Server Extension Workspace".

2. Open the PortalExtension project and add the custom code for your ADF library.
   See also Section 55.2.1.3, "The PortalExtension Project".

3. Deploy the ADF Library to the WebCenter Portal shared library extend.spaces.webapp.war.
   See also Section 55.2.5, "Deploying Extensions to the WebCenter Portal Shared Library (extend.spaces.webapp.war)".

55.2.7 Troubleshooting WebCenter Portal Shared Library Deployment

- Changes are not available after deployment even though deployment successful.

WebCenter Portal always uses the latest shared library version. Check that the implementation version in MANIFEST.MF matches the implementation version displayed in the WebLogic Server Administration Console.

For example, check the value in PortalSharedLibrary/Application Sources/META-INF/MANIFEST.MF is the same as that displayed in the Administration Console under Deployments> extend.spaces.webapp>Overview

- "DeployerException: Task 9 failed" displays:

weblogic.Deployer$DeployerException:
weblogic.deploy.api.tools.deployer.DeployerException: Task 9 failed:
[Deployer:id49117]deploy library com.mycompanyname.shared.lib

This error occurs if the implementation version of the new deployment and the existing deployment are the same. Use the Administration Console to verify the
current implementation version and if necessary change the version and redeploy the shared library.

- "java.lang.IllegalArgumentException" displays:
  
  Restart the managed server on which WebCenter Portal is deployed.
This chapter describes how to expose portal functionality in a Portal Framework application, using the WebCenter Portal API. Through this API, you can create new portals, manage portal membership, retrieve portal information, and more, from your Portal Framework applications. This chapter also describes how to expose information from other applications, such as Portal Framework applications in portals.

This chapter includes the following sections:

- Section 56.1, "Exposing Portals in Your Application Using the WebCenter Portal API"
- Section 56.2, "Using the WebCenter Portal REST API"
- Section 56.3, "Exposing Enterprise Applications in Portals"

---

**Note:** Custom-built task flows that implement the WebCenter Portal API (described in this chapter) can be deployed on Portal Framework applications, but you cannot deploy them to a WebCenter Portal application.

### 56.1 Exposing Portals in Your Application Using the WebCenter Portal API

This section includes the following subsections:

- Section 56.1.1, "Introduction to the WebCenter Portal API"
- Section 56.1.2, "Case Study 1: Purchasing Application Uses a Portal to Evaluate Suppliers"
- Section 56.1.3, "Case Study 2: Customer Support Center Application Uses a Portal to Discuss Customer Escalations"
- Section 56.1.4, "How to Set Up Your Framework Application to Use the WebCenter Portal API"
- Section 56.1.5, "Providing WebCenter Portal Functionality in Your Portal Framework Application"
- Section 56.1.6, "How to Handle Exceptions Raised by the WebCenter Portal API"
- Section 56.1.7, "Finding More Information on the WebCenter Portal API"
- Section 56.1.8, "Troubleshooting Issues with the WebCenter Portal API"
56.1.1 Introduction to the WebCenter Portal API

While using your Portal Framework application, users may encounter situations where portal functionality 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 WebCenter Portal. To this end, WebCenter Portal provides access to a subset of its functionality through an API. Using the API’s classes and methods, you can integrate powerful portal functionality into your Framework application.

You can use the WebCenter Portal API to:

- **Create and manage portals and portal templates.** You can create and delete portals, and add custom attributes. For more information, see Section 56.1.5.1, "Managing Portals and Portal Templates."

- **Manage portal membership.** You can add and remove portal members. For more information, see Section 56.1.5.2, "Managing Portal Membership."

- **Retrieve information about portals and portal templates.** For example, you can retrieve a portal’s URL or the URL of a specific portal. You can also retrieve portal and portal template metadata. For more information, see Section 56.1.5.3, "Retrieving Information for Portals and Portal Templates."

Portal methods are contained within several classes. Table 56–1 lists the different classes and describes the purpose of the methods within each class.

### Table 56–1 WebCenter Portal API Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Contains Methods for</th>
<th>More Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>GroupSpaceWSClient</td>
<td>Creating and managing portals and portal templates</td>
<td>Section 56.1.5, &quot;Providing WebCenter Portal Functionality in Your Portal Framework Application&quot;</td>
</tr>
<tr>
<td></td>
<td>Managing portal membership</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Retrieving portal information</td>
<td></td>
</tr>
<tr>
<td>GroupSpaceWSContext</td>
<td>Establishing context before calling other API methods</td>
<td>Section 56.1.4.4, &quot;Setting Up the WebCenter Portal Client Context&quot;</td>
</tr>
<tr>
<td>GroupSpaceWSException</td>
<td>Managing exceptions raised by the API</td>
<td>Section 56.1.6, &quot;How to Handle Exceptions Raised by the WebCenter Portal API&quot;</td>
</tr>
<tr>
<td>GroupSpaceWSMembers</td>
<td>Retrieving information about portal members</td>
<td>Section 56.1.5.3, &quot;Retrieving Information for Portals and Portal Templates&quot;</td>
</tr>
<tr>
<td>GroupSpaceWSMetadata</td>
<td>Retrieving portal information</td>
<td>Section 56.1.5.3, &quot;Retrieving Information for Portals and Portal Templates&quot;</td>
</tr>
</tbody>
</table>

From your perspective, as a Portal Framework developer, here are a couple of case studies describing scenarios where building and working with WebCenter Portal through its API may be useful:

- In a purchasing application, a purchasing manager may create a portal to discuss the suitability of a new supplier. For more information, see Section 56.1.2, "Case Study 1: Purchasing Application Uses a Portal to Evaluate Suppliers."

- In a Customer Support Center application, a support analyst may create a portal to collaborate with other users about a customer escalation. For more information,
56.1.2 Case Study 1: Purchasing Application Uses a Portal to Evaluate Suppliers

Consider a purchasing application built using the 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 portal can provide this collaborative environment. So the purchasing application includes a call to the `createGroupSpace` method, and this enables the purchasing manager to click a link (Create a New Portal) that displays a "Create Portal" dialog, directly from the purchasing application, as shown in Figure 56–1.

![Figure 56–1 Purchasing Application Includes a Portal Creation Link](image)

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 portal,
and also selects a template (Purchasing Projects) as shown in Figure 56–2. The Purchasing Projects template sets up the portal 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 API enables you to create your own “Create Portal” dialog, you can apply your own look and feel and terminology.

Figure 56–2 Custom Create Portal 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 portal from the Supplier ID so that the purchasing manager would not have to enter any details at all. The portal could then be created with just the click of a link.

When the purchasing manager clicks **OK**, the new Supplier Evaluation portal displays, similar to that shown in Figure 56–3.
Figure 56–3  Supplier Evaluation Portal

Figure 56–3 shows a Home page for the Supplier Evaluation portal, 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 portal 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 portal, the purchasing manager becomes the moderator of the portal automatically. The moderator can add content to the portal, initiate discussions, invite members, and collaborate with interested parties. Once consensus is reached regarding the supplier, the purchasing manager can approve or reject that supplier.

56.1.3 Case Study 2: Customer Support Center Application Uses a Portal 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 portal. The support analyst creates a portal from within the Customer Support Center application and adds the required members. These members are then notified and use the portal to start discussing the customer situation. The support analyst views their updates to the portal inside the support application, navigating to the portal whenever necessary to obtain more specific details. Based on the information she gets from the other members of the portal, she can diagnose the problem and offer the customer a solution very quickly.

56.1.4 How to Set Up Your Framework Application to Use the WebCenter Portal API

Before you can use the WebCenter Portal API you must ensure that WebCenter Portal is up and running and that your project is set up correctly in JDeveloper.

This section includes the following subsections:

- Section 56.1.4.1, "Verifying That WebCenter Portal Is Up and Running"
- Section 56.1.4.2, "Setting Up WebCenter Portal to Use the WebCenter Portal API"
- Section 56.1.4.3, "Securing the Connection Between the Portal Framework Application and WebCenter Portal"
- Section 56.1.4.4, "Setting Up the WebCenter Portal Client Context"

56.1.4.1 Verifying That WebCenter Portal Is Up and Running

If you want to use the WebCenter Portal API, WebCenter Portal must be up and running.

To verify that WebCenter Portal is up and running:

1. Start WebCenter Portal. The URL is:

   http://host:port/webcenter

   You do not need to log in.

2. Verify that the WebCenter Portal 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 56–4. If you do not see this page, contact your Fusion Middleware administrator.
56.1.4.2 Setting Up WebCenter Portal to Use the WebCenter Portal API

Before you can call the WebCenter Portal API in your Portal Framework application, you must ensure that your application contains the correct libraries and that there is a connection to the WebCenter Portal Web service.

To set up your application to use the WebCenter Portal API:

1. Start JDeveloper.
2. In the Application Navigator, expand the application for which you want to provide WebCenter Portal functionality.
   
   The application should be based on the WebCenter Portal Application template.
3. Right-click the view-controller project and choose Project Properties.
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

For more information about setting up WebCenter Portal, see the “Getting WebCenter Portal Up and Running” chapter in the *Administering Oracle WebCenter Portal*.
Exposing Portals in Your Application Using the WebCenter Portal API

- JAX-RPC Client
- JAX-WS Client
- Oracle JEWT
- WebCenter Portlet Client View (portlet-client-adf-view.jar, portlet-client-adf-diagnostic.jar)
- WebCenter Portlet Tag Library (portlet-client-adf-taglib.jar)
- WebCenter Portlet Skin (portlet-client-adf-skin.jar)
- WebCenter Portlet Client Dependencies (tidy.jar, wsrp-types.jar, wsrp-jaxb.jar)
- WebCenter Portlet Client WSRP Stubs (wsrp-stubs.jar)
- WebCenter Portlet Designtime (adfp-portletdt-share.jar)
- JDEV_MW_HOME/oracle_common/modules/oracle.adf.model_11.1.1/adfm.jar
- JDEV_MW_HOME/oracle_common/modules/oracle.xdk_11.1.0/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 WebCenter Portal 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 56–1 appears:

   **Example 56–1  Portal Web Service Endpoint URL Connection**

   ```xml
   <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>
   ```
Exposing Portals in Your Application Using the WebCenter Portal API

If you need to set the WebCenter Portal Web Service endpoint at runtime, you can use the `setGroupSpaceWebServiceEndpoint` method (Example 56–2). You can write a wrapper method 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 56–2  Setting the WebCenter Portal Web Service Endpoint at Runtime**

```java
client.setGroupSpacesWebServiceEndpoint("http://xmlns.oracle.com/webcenter/SpacesWebService");
```

56.1.4.3 Securing the Connection Between the Portal Framework Application and WebCenter Portal

Before using the WebCenter Portal API in your Portal Framework application, you must ensure that the communication between the application (the consumer) and WebCenter Portal (the producer) is secure. This is so that the identity of the user invoking the API is propagated to WebCenter Portal in a secure manner where the integrity and confidentiality of the communication is maintained.

To do this, the WebCenter Portal API uses a policy of SAML-based token passing with message protection. Your administrator must create a Java keystore and update the credential store so that WebCenter Portal 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 the "Securing WebCenter Portal for Applications Consuming WebCenter Portal Client API with WS-Security" section in the *Administering Oracle WebCenter Portal*.

56.1.4.4 Setting Up the WebCenter Portal Client Context

Before calling any of the WebCenter Portal API methods in your application code, a few setup steps are required.

To set up the WebCenter Portal client context:

1. Domain-wide configuration settings are used when you call the WebCenter Portal API from an application deployed on WebLogic Server (WLS), unless you specifically choose to override them. To override or set security configuration parameters, use the methods provided by the `GroupSpaceWSContext` class.

   In particular, you must set the SAML issuer name and the public key alias for WebCenter Portal, which are required for data encryption. If the WebCenter Portal Web service endpoint is not set in the `connections.xml` file, you can set this too.

   For example:

   ```java
   GroupSpaceWSContext context = new GroupSpaceWSContext();
   context.setEndPoint("endPointUrl");
   context.setSamlIssuerName("samlIssuer");
   context.setRecipientKeyAlias("producer");
   groupSpaceWSClient = new GroupSpaceWSClient(context);
   
   Where:
   ```
   - `endPointUrl` is the WebCenter Portal Web service endpoint, for example, `http://xmlns.oracle.com:8912/webcenter/SpacesWebService`.
■ `samlIssuer` is the issuer URI of the SAML Authority issuing assertions for this SAML Asserting Party, for example, `www.oracle.com`. Out-of-the-box, `www.oracle.com` is the only trusted SAML issuer.

■ `producer` is the public key alias for WebCenter Portal, for example, `orakey` if you are using a simple topology.

For information about the steps that the administrator must perform, see the "Securing WebCenter Portal for Applications Consuming WebCenter Portal Client API with WS-Security" section in the *Administering Oracle WebCenter Portal*.

For a full list of the methods provided by the `GroupSpaceWSContext` class, see *Java API Reference for Oracle WebCenter Portal*.

2. Initialize the WebCenter Portal client by passing the context. For example:

   ```java
   GroupSpaceWSClient client = new GroupSpaceWSClient(context);
   ```

3. Once you have initialized the client, check everything is set up correctly by calling:

   ```java
   getWebCenterSpacesURL();
   ```

   If the correct URL is returned, everything is set up correctly and you can start using the WebCenter Portal API.

56.1.5 Providing WebCenter Portal Functionality in Your Portal Framework Application

The WebCenter Portal API enables you to perform common portal-related operations within a Framework application. Most of the API's methods are provided by the `GroupSpaceWSClient` class. The WebCenter Portal API methods can be grouped into three main categories:

■ Section 56.1.5.1, "Managing Portals and Portal Templates"

■ Section 56.1.5.2, "Managing Portal Membership"

■ Section 56.1.5.3, "Retrieving Information for Portals and Portal Templates"

Table 56–2 lists the methods in the `GroupSpaceWSClient` class.

<table>
<thead>
<tr>
<th>Portal Methods</th>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>createGroupSpace</td>
<td>Managing Portals and Portal Templates</td>
<td>Creates a new portal based on a template. See Section 56.1.5.1.1, &quot;Creating a Portal.&quot;</td>
</tr>
<tr>
<td>setCustomAttribute</td>
<td>Managing Portals and Portal Templates</td>
<td>Creates one or more custom attributes for a portal. See Section 56.1.5.1.2, &quot;Creating a Custom Attribute.&quot;</td>
</tr>
<tr>
<td>deleteGroupSpace</td>
<td>Managing Portals and Portal Templates</td>
<td>Permanently removes a portal from Portal. See Section 56.1.5.1.4, &quot;Deleting a Portal.&quot;</td>
</tr>
<tr>
<td>createGroupSpaceTemplate</td>
<td>Managing Portals and Portal Templates</td>
<td>Creates a new portal template based on an existing portal. See Section 56.1.5.1.3, &quot;Creating a Portal Template.&quot;</td>
</tr>
<tr>
<td>addMember</td>
<td>Managing Portal Membership</td>
<td>Makes a user (or a group) a portal member with a specific role. See Section 56.1.5.2.1, &quot;Adding Members to a Portal.&quot;</td>
</tr>
</tbody>
</table>
### Table 56–2 (Cont.) Methods for Performing Common Portal Operations

<table>
<thead>
<tr>
<th>Portal Methods</th>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inviteMember</td>
<td>Managing Portal Membership</td>
<td>Invites a list of users to become members of a portal. See Section 56.1.5.2.2, “Inviting Users to Join a Portal.”</td>
</tr>
<tr>
<td>removeMember</td>
<td>Managing Portal Membership</td>
<td>Revokes portal membership. See Section 56.1.5.2.3, “Removing Members from a Portal.”</td>
</tr>
<tr>
<td>getRoles</td>
<td>Managing Portal Membership</td>
<td>Retrieves all the roles for a given portal. See Section 56.1.5.2.4, “Retrieving Role Information.”</td>
</tr>
<tr>
<td>getGroupSpaces</td>
<td>Retrieving Information for Portals and Portal Templates</td>
<td>Retrieves a list of portals for a given query string. See Section 56.1.5.3.1, “Retrieving a List of Portals.”</td>
</tr>
<tr>
<td>getPublicGroupSpaces</td>
<td>Retrieving Information for Portals and Portal Templates</td>
<td>Retrieves a list of public portals for a given query string. See Section 56.1.5.3.2, “Retrieving a List of Public Portals.”</td>
</tr>
<tr>
<td>getDiscoverableGroupSpaces</td>
<td>Retrieving Information for Portals and Portal Templates</td>
<td>Retrieves a list of discoverable portals for a given query string. See Section 56.1.5.3.3, “Retrieving a List of Discoverable Portals.”</td>
</tr>
<tr>
<td>getGroupSpaceMetadata</td>
<td>Retrieving Information for Portals and Portal Templates</td>
<td>Retrieves information (metadata) about the portal. See Section 56.1.5.3.4, “Retrieving Portal Metadata.”</td>
</tr>
<tr>
<td>getGroupSpaceMetadataByGuid</td>
<td>Retrieving Information for Portals and Portal Templates</td>
<td>Retrieves information (metadata) about a portal given the portal’s GUID. See Section 56.1.5.3.4, “Retrieving Portal Metadata.”</td>
</tr>
<tr>
<td>getGroupSpaceTemplate</td>
<td>Retrieving Information for Portals and Portal Templates</td>
<td>Retrieves a list of portal templates for a given query string. See Section 56.1.5.3.5, “Retrieving a List of Portal Templates.”</td>
</tr>
<tr>
<td>getGroupSpaceTemplateMetadata</td>
<td>Retrieving Information for Portals and Portal Templates</td>
<td>Retrieves information (metadata) about the portal template. See Section 56.1.5.3.6, “Retrieving Portal Template Metadata.”</td>
</tr>
<tr>
<td>getGroupSpaceTemplateMetadataByGuid</td>
<td>Retrieving Information for Portals and Portal Templates</td>
<td>Retrieves information (metadata) about the portal template given the template’s GUID. See Section 56.1.5.3.6, “Retrieving Portal Template Metadata.”</td>
</tr>
<tr>
<td>getWebCenterSpacesURL</td>
<td>Retrieving Information for Portals and Portal Templates</td>
<td>Retrieves the Portal application URL. See Section 56.1.5.3.7, “Retrieving the Portal URL.”</td>
</tr>
<tr>
<td>getGroupSpaceURL</td>
<td>Retrieving Information for Portals and Portal Templates</td>
<td>Retrieves a portal URL. See Section 56.1.5.3.8, “Retrieving a Portal URL.”</td>
</tr>
<tr>
<td>getServiceRSSFeedURL</td>
<td>Retrieving Information for Portals and Portal Templates</td>
<td>Retrieves RSS feed URLs for the specified portal service.</td>
</tr>
<tr>
<td>getServiceRSSFeedURLByGuid</td>
<td>Retrieving Information for Portals and Portal Templates</td>
<td>Retrieves RSS feed URL for a given portal by GUID and a given service.</td>
</tr>
</tbody>
</table>
56.1.5.1 Managing Portals and Portal Templates

Use the following WebCenter Portal API methods to manage your portals and portal templates:

- createGroupSpace
- setCustomAttribute
- createGroupSpaceTemplate
- deleteGroupSpace

Before you begin

Before using any of these methods, you must complete all the steps listed in Section 56.1.4, "How to Set Up Your Framework Application to Use the WebCenter Portal API."

This section includes the following subsections:

- Section 56.1.5.1.1, "Creating a Portal"
- Section 56.1.5.1.2, "Creating a Custom Attribute"
- Section 56.1.5.1.3, "Creating a Portal Template"
- Section 56.1.5.1.4, "Deleting a Portal"

56.1.5.1.1 Creating a Portal

Use the createGroupSpace method to create a portal that is based on an existing portal template.

To use this method, specify:

- The internal name of the new portal. This name can contain blank spaces.
- A display name for the new portal.
- A description of the portal. Although this is not mandatory, we recommend that you provide a description to help users identify the portal’s purpose.
- The name of the portal 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 portal in searches.

Example 56–3 creates a portal with the name Databases that is based on the CommunityofInterest template. The example also specifies two search keywords (databases and Oracle).

Example 56–3 Creating a Portal

```java
//create the portal
GroupSpaceWSMetadata gsMetadata =
client.createGroupSpace("Databases", "Databases" "A community for people
interested in databases", "databases, oracle", "CommunityofInterest");

//print the portal GUID to provide confirmation of creation
System.out.println("GUID: 	 + gsMetadata.getGuid());
```

For an example of how to provide custom attributes for a new portal, see Section 56.1.5.1.2, “Creating a Custom Attribute.”

56.1.5.1.2 Creating a Custom Attribute

Every portal 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 portal and its characteristics.

Use the `setCustomAttribute` method to specify a custom attribute for a portal. To use this method, specify the name of the portal and a name, description, type, and value for the custom attribute.

**Example 56–4** creates the Databases portal and then creates a custom attribute for that portal. The attribute is named `Vendors`, with the description *List of vendors*. It takes string values of Oracle and IBM.

**Example 56–4  Creating a Custom Attribute**

```java
//create the portal
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");
```

56.1.5.1.3  Creating a Portal Template  Use the `createGroupSpaceTemplate` method to create a portal template based on an existing portal. To use this method specify a name, display name, and description for the template, and the name of the portal to use to create the template.

**Example 56–5** creates a portal template based on the `Databases` portal.

**Example 56–5  Creating a Portal Template**

```java
GroupSpaceWSMetadata templMetadata =
client.createGroupSpaceTemplate("DatabasesTemplate", "Databases Template", 
"A template based on the Databases portal", "Databases");
//print the template GUID to provide confirmation of creation
System.out.println("GUID: " + templMetadata.getGuid());
```

56.1.5.1.4  Deleting a Portal  Use the `deleteGroupSpace` method to permanently delete a portal from WebCenter Portal. To use this method, specify the name of the portal to delete. The method returns a boolean value to indicate whether the deletion was successful.

**Example 56–6** deletes the portal with the name `Databases`. The example uses the boolean value returned by the method to print a message about the success or failure of the operation.

**Example 56–6  Deleting a Portal**

```java
//delete the portal
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");
}
```
56.1.5.2 Managing Portal Membership

Use the following WebCenter Portal API methods to manage portal membership:

- addMember
- inviteMember
- removeMember
- getRoles

Before you begin

Before using any of these methods, you must complete all the steps listed in Section 56.1.4, "How to Set Up Your Framework Application to Use the WebCenter Portal API."

This section includes the following subsections:

- Section 56.1.5.2.1, "Adding Members to a Portal"
- Section 56.1.5.2.2, "Inviting Users to Join a Portal"
- Section 56.1.5.2.3, "Removing Members from a Portal"
- Section 56.1.5.2.4, "Retrieving Role Information"

56.1.5.2.1 Adding Members to a Portal

Use the addMember method to give users (and groups) portal membership and assign the new members to a specific role. To use this method, specify the name of the portal, 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 portal.

You must specify the name of a valid user or user group that exists in the WebCenter Portal 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 portal, use the getRoles method. For more information, see Section 56.1.5.2.4, "Retrieving Role Information." For more information about roles, see the "Managing Roles and Permissions for a Portal" section in Using Oracle WebCenter Portal.

Example 56–7 makes users (of type GroupSpaceWSMembers) named Pat and Vicki and everyone in the Sales and Marketing user group, members of the Databases portal. 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 56–7 Adding Members to a Portal**

```java
//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 Portal
client.addMember("Databases", addMem);
//print a list of members to confirm the new members were added
GroupSpaceWSMetadata memMetaData = client.getGroupSpaceMetadata("Databases");
for (GroupSpaceWSMembers mems : memMetaData.getMembers())
{
    System.out.println(mems.getMember());
    System.out.println(mems.getRole());
}

56.1.5.2.2 Inviting Users to Join a Portal Use the inviteMember method to invite users to become members of a portal. To use this method, specify the name of the portal 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 portal. An invitation to join the portal 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 WebCenter Portal 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 portal, use the getRoles method. For more information, see Section 56.1.5.2.4, "Retrieving Role Information." For more information about roles, see the "Managing Roles and Permissions for a Portal" section in Using Oracle WebCenter Portal.

Example 56–8 invites users (of type GroupSpaceWSMembers) named Pat and Vicki to become members of the Databases portal with Viewer and Participant roles respectively.

Example 56–8 Inviting Users to Join a Portal

   //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 portal
   client.inviteMember("Databases", inviteMem);

56.1.5.2.3 Removing Members from a Portal Use the removeMember method to revoke portal membership. To use this method, specify the name of the portal, and a list of users. The list must contain objects of type GroupSpaceWSMembers that specify the user name. To obtain a list of current portal members, use the getGroupSpaceMetadata method. For more information, see Section 56.1.5.3.4, "Retrieving Portal Metadata."

Example 56–9 removes users (of type GroupSpaceWSMembers) Pat and Vicki from the membership of the Databases portal.

Example 56–9 Removing Members from a Portal

   //create the list of users
   List remMem = new ArrayList();
   //create the GroupSpaceWSMembers objects
   GroupSpaceWSMembers mem1 = new GroupSpaceWSMembers("pat");
   GroupSpaceWSMembers mem2 = new GroupSpaceWSMembers("vicki");
   //add the GroupSpaceWSMembers objects to the list of users
   remMem.add(mem1);
remMem.add(mem2);

//remove the users from the portal
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());
}

56.1.5.2.4 Retrieving Role Information  Use the getRoles method to retrieve all the roles for a given portal. This is useful for determining which roles are available when adding or inviting members to a portal. Roles include out-of-the box roles (Moderator, Participant, Viewer) and also custom roles (if any). To use this method, specify the name of the portal.

Example 56–10 retrieves and displays role information for the Databases portal.

Example 56–10  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);
}

56.1.5.3 Retrieving Information for Portals and Portal Templates
Use the following Portal methods to retrieve portal information:

- getGroupSpaces
- getPublicGroupSpaces
- getDiscoverableGroupSpaces
- getGroupSpaceMetadata
- getGroupSpaceMetadataByGuid
- getGroupSpaceTemplates
- getGroupSpaceTemplateMetadata
- getGroupSpaceTemplateMetadataByGuid
- getWebCenterSpacesURL
- getGroupSpaceURL

Before you begin
Before using the WebCenter Portal API methods, you must complete all the steps listed in Section 56.1.4, "How to Set Up Your Framework Application to Use the WebCenter Portal API."

This section includes the following subsections

- Retrieving a List of Portals
- Retrieving a List of Public Portals
Exposing Portals in Your Application Using the WebCenter Portal API

- Retrieving a List of Discoverable Portals
- Retrieving Portal Metadata
- Retrieving a List of Portal Templates
- Retrieving Portal Template Metadata
- Retrieving the Portal URL
- Retrieving a Portal URL
- Retrieving RSS Feed URLs for WebCenter Portal Tools and Services

56.1.5.3.1 Retrieving a List of Portals  Use the `getGroupSpaces` method to obtain a list of portals that match a given query string. To use this method, specify a query string. A null value query string returns a list of all portals that are accessible to the current user. The method returns portals that are accessible to the current user, up to a maximum of 500.

Example 56–11 returns a list of portals containing the string `Database`.

Example 56–11  Retrieving a List of Specific Portals

```java
List<String> allGroupSpaces = client.getGroupSpaces("Database");
```

Example 56–12 returns a list of all portals to which the current user has access. This is achieved by specifying a null query string.

Example 56–12  Retrieving a List of All Portals

```java
List<String> allGroupSpaces = client.getGroupSpaces(null)
```

56.1.5.3.2 Retrieving a List of Public Portals  Use the `getPublicGroupSpaces` method to obtain a list of public portals that match a given query string. To use this method, specify a query string. A null value query string returns a list of all public portals. The method returns portals that are accessible to all users, even those who are not logged in to the WebCenter Portal application, up to a maximum of 500.

Example 56–13 returns a list of public portals containing the string `Database`.

Example 56–13  Retrieving a List of Specific Public Portals

```java
List<String> allPublicGroupSpaces = client.getPublicGroupSpaces("Database");
```

Example 56–14 returns a list of all public portals. This is achieved by specifying a null query string.

Example 56–14  Retrieving a List of All Public Portals

```java
List<String> allPublicGroupSpaces = client.getPublicGroupSpaces(null)
```

56.1.5.3.3 Retrieving a List of Discoverable Portals  Use the `getDiscoverableGroupSpaces` method to obtain a list of discoverable portals that match a given query string. To use this method, specify a query string. A null value query string returns a list of all discoverable portals.

The method returns portals that are accessible to all users who are logged into WebCenter Portal, up to a maximum of 500.

Example 56–13 returns a list of discoverable portals containing the string `Database`.
Exposing Portals in Your Application Using the WebCenter Portal API

Example 56–15  Retriving a List of Discoverable portals

List<String> allDiscoverableGroupspaces = client.getDiscoverableGroupspaces("Database");

Example 56–14 returns a list of all discoverable portals. This is achieved by specifying a null query string.

Example 56–16  Retriving a List of All Discoverable Portals

List<String> allDiscoverableGroupSpaces = client.getDiscoverableGroupSpaces(null)

56.1.5.3.4  Retrieving Portal Metadata

Use the getGroupSpaceMetadata or getGroupSpaceMetadataByGuid methods to obtain information (metadata) about a particular portal. This includes information such as the description of the portal, the name of the user who created it, the date on which it was last updated, and so on.

To use the getGroupSpaceMetadata method, specify the name of the portal. To use the getGroupSpaceMetadataByGuid method, specify the GUID of the portal. Note that while the portal name may be changed during the existence of the portal, the GUID always remains the same. You can obtain the GUID of a portal as follows:

getGroupSpaceMetadata("spaceName").getGuid();

Both methods return a bean object that contains more methods that you can then use for retrieving the portal metadata. These methods are provided by the GroupSpaceWSMetadata class. Table 56–3 lists the methods returned by the bean object:

<table>
<thead>
<tr>
<th>Portal Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>getCreatedBy</td>
<td>Returns the name of the person who created the portal.</td>
</tr>
<tr>
<td>getSpaceAttributes</td>
<td>Returns the custom attributes for a portal.</td>
</tr>
<tr>
<td>getDescription</td>
<td>Returns the description of the portal.</td>
</tr>
<tr>
<td>getDisplayName</td>
<td>Returns the display name of the portal.</td>
</tr>
<tr>
<td>getGroupSpaceState</td>
<td>Returns the state of the portal: active, offline, deleted.</td>
</tr>
<tr>
<td>getGuid</td>
<td>Returns the GUID of the portal.</td>
</tr>
<tr>
<td>getIconURL</td>
<td>Returns the location of the portal icon.</td>
</tr>
<tr>
<td>getKeywords</td>
<td>Returns a comma separated list of keywords used to describe the portal.</td>
</tr>
<tr>
<td>getLastUpdated</td>
<td>Returns the date the portal was last updated.</td>
</tr>
<tr>
<td>getLogoURL</td>
<td>Returns the location of the portal logo.</td>
</tr>
<tr>
<td>getMailingList</td>
<td>Returns the mailing list for the portal.</td>
</tr>
<tr>
<td>getMembers</td>
<td>Returns the current list of portal members.</td>
</tr>
<tr>
<td>getName</td>
<td>Returns the internal name of the portal.</td>
</tr>
<tr>
<td>isDiscoverable</td>
<td>Returns if the portal is returned in searches made by users who are not members of the portal.</td>
</tr>
<tr>
<td>isPublic</td>
<td>Returns if the portal is available to users who are not logged in.</td>
</tr>
</tbody>
</table>

Example 56–17 retrieves the Description, Keywords, and Last Updated Date information for the Databases portal given the portal’s name.
Example 56–17  Retrieving Portal Metadata Using the Portal Name

//get the exact name of the portal
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 portal description
    System.out.println("Description: " + metadata.getDescription());
    //get portal keywords
    System.out.println("Keywords: " + metadata.getKeywords());
    //get the date the portal was last updated
    System.out.println("Last Updated Date: " + metadata.getLastUpdated().toString());
}

Example 56–18 retrieves the Display Name, Creator, and Last Updated Date information for the Databases portal given the portal GUID.

Example 56–18  Retrieving Portal Metadata Using Portal GUID

GroupSpaceWSMetadata metadata = client.getGroupSpaceMetadataByGuid("Guid");
//get portal display name
System.out.println("Display Name: " + metadata.getDisplayName());
//get the name of the user who created the portal
System.out.println("Created By: " + metadata.getCreatedBy());
//get the date the portal was last updated
System.out.println("Last Updated Date: " + metadata.getLastUpdated().toString());

Example 56–19 retrieves the Name, Description, Type, and Value for every custom attribute associated with the Databases portal.

Example 56–19  Retrieving Custom Attribute Metadata

//get the exact name of the portal
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 56–20 retrieves a list of members of the Databases portal.

Example 56–20  Retrieving Membership Information

//get the exact name of the portal
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);
  }
}

56.1.5.3.5 Retrieving a List of Portal Templates  Use the getGroupSpaceTemplates method to obtain a list of portal templates that match a given query string. To use this method, specify a query string. A null value query string returns a list of all templates that are accessible to the current user.

The method returns templates that are accessible to the current user, up to a maximum of 500.

Example 56–21 returns a list of portal templates containing the string Interest.

Example 56–21  Retrieving a List of Specific Portal Templates
List<String> allGroupSpaceTemplates = client.getGroupSpaceTemplates("Interest");

Example 56–22 returns a list of all portal templates to which the current user has access. This is achieved by specifying a null query string.

Example 56–22  Retrieving a List of All Portal Templates
List<String> allGroupSpaceTemplates = client.getGroupSpaceTemplates(null);

56.1.5.3.6 Retrieving Portal Template Metadata  Use the getGroupSpaceTemplateMetadata and getGroupSpaceTemplateMetadataByGuid methods to obtain information (metadata) about a particular portal 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 method, specify the name of the template. To use the getGroupSpaceTemplateMetadataByGuid method, specify the GUID of the template. Note that while the portal template name may be changed during the existence of the template, the GUID always remains the same.

Both methods return a bean object that contains more methods that you can then use for retrieving the portal template metadata. These methods are provided by the GroupSpaceWSMetadata class. Table 56–4 lists the methods returned by the bean object:

<table>
<thead>
<tr>
<th>Portal Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>getCreatedBy</td>
<td>Returns the name of the person who created the template.</td>
</tr>
<tr>
<td>getDescription</td>
<td>Returns the description of the template.</td>
</tr>
<tr>
<td>getDisplayName</td>
<td>Returns the display name of the template.</td>
</tr>
<tr>
<td>getGuid</td>
<td>Returns the GUID of the template.</td>
</tr>
<tr>
<td>getIconURL</td>
<td>Returns the location of the template icon.</td>
</tr>
<tr>
<td>getKeywords</td>
<td>Returns a comma separated list of keywords used to describe the template.</td>
</tr>
</tbody>
</table>
Exposing Portals in Your Application Using the WebCenter Portal API

Example 56–23 retrieves the GUID, Description, and Created By information for the CommunityofInterest portal template given the template name.

Example 56–23 Retrieving Template Metadata Using the Template Name

```java
Example 56–23
Retrieving Template Metadata Using the Template Name
GroupSpaceWSMetadata metadata =
client.getGroupSpaceTemplateMetadata(myGroupSpaceTemplate);
//get the exact name of the portal 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 56–24 retrieves the name of a portal template given the template GUID.

Example 56–24 Retrieving Template Metadata Given the Template GUID

```java
Example 56–24
Retrieving Template Metadata Given the Template GUID
GroupSpaceWSMetadata templGuidMetadata =
client.getGroupSpaceTemplateMetadataByGuid("Guid");
System.out.println("Template Name: " + templGuidMetadata.getName());
```

56.1.5.7 Retrieving the Portal URL Use the getWebCenterSpacesURL method to obtain the URL of the WebCenter Portal instance.

Example 56–25 retrieves the URL of the currently running instance of WebCenter Portal.

Example 56–25 Retrieving the WebCenter Portal URL

```java
Example 56–25
Retrieving the WebCenter Portal URL
String myWebCenterSpacesURL = client.getWebCenterSpacesURL();
```

56.1.5.8 Retrieving a Portal URL Use the getGroupSpaceURL method to obtain the URL of a specific portal. This is useful for when you want to construct a hyperlink for a portal or you have a relative URL that you need to make into an absolute URL. To use this method, specify the name of the portal.

Example 56–26 retrieves the URL of the Databases portal.

Example 56–26 Retrieving a Portal URL

```java
Example 56–26
Retrieving a Portal URL
String myGroupSpaceURL = client.getGroupSpaceURL("Databases");
```

Example 56–27 prints a list of portals as hyperlinks.
**Example 56–27  Printing a List of Portals as Hyperlinks**

```java
//get the list of portals
List<String> spaces = client.getGroupSpaces("*");
//print the list of portals 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 portal, retrieve the portal URL and then add the page information. For information about how to create pretty URLs for portal pages, see the "WebCenter Portal Pretty URLs" section in *Building Portals with Oracle WebCenter Portal*.

56.1.5.3.9 Retrieving RSS Feed URLs for WebCenter Portal Tools and Services  WebCenter Portal members can find out what is happening in a portal through various RSS news feeds. The following portal RSS feeds are available:

- **Announcements RSS** - View portal 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 from WebCenter Portal, using the following WebCenter Portal API methods:

- `getServiceRSSFeedURL`
- `getServiceRSSFeedURLbyGuid`

To obtain an RSS feed URL, you must identify the portal (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 Portal RSS News Feed URLs Using `getServiceRSSFeedURL`**

Use the `getServiceRSSFeedURL` method to obtain service-related RSS news feed URLs for a particular portal by specifying the portal name.

To retrieve the RSS news feed URL for a service using `getServiceRSSFeedURL`, use the following:

```java
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 portal
- `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`. 

---

56-22  Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper
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 56–28** retrieves the recent activity RSS feed URL for a portal named Finance_Project:

**Example 56–28 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` method to obtain service-related RSS news feed URLs for a particular portal by specifying the portal'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 portal GUID. For information about obtaining a portal's GUID, see Section 56.1.5.3.4, "Retrieving Portal 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 56–29** retrieves the recent activity RSS feed URL for a portal with the GUID: s2201fa44_b441_4bdd_950e_47307f6f9800:

**Example 56–29 Retrieving the RSS Feed URL for Recent Activity**

```
String recentActivityURL = client.getServiceRSSFeedURLbyGuid("s2201fa44_b441_4bdd_950e_47307f6f9800", GroupSpaceWSClient.RECENT_ACTIVITY_SERVICE_ID);
```

### 56.1.6 How to Handle Exceptions Raised by the WebCenter Portal API

The `GroupSpaceWSException` class provides methods for handling exceptions raised by WebCenter Portal API methods.

<table>
<thead>
<tr>
<th>Method</th>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>getLocalizedMessage</td>
<td>GroupSpaceWSException</td>
<td>Composes the localized error message.</td>
</tr>
</tbody>
</table>
This section includes the following subsections:

- Section 56.1.6.1, "Providing Localized Error Messages"
- Section 56.1.6.2, "Listing the Error Stack"

### 56.1.6.1 Providing Localized Error Messages

Sometimes you may find that the default error messages provided by the WebCenter Portal API is not specific enough for your particular application. In these cases you can provide your own error messages.

Use the `getLocalizedMessage` method to compose application-specific error messages. Example 56–30 shows a servlet that includes code to create a portal. If any exceptions are raised during the creation process, a localized message is printed.

**Example 56–30 Printing a Localized Error Message**

```java
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 portal");
  }
  catch(GroupSpaceWSException ex)
  //For example:
  {
    if(ex instanceof GroupSpaceNameNullException)
      print(ex.getLocalizedMessage());
    else if (ex instanceof GroupSpaceDescNullException)
      print(ex.getLocalizedMessage());
  }
  ...
}
```

### 56.1.6.2 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` method to list all the errors that caused a particular exception. Example 56–31 prints the exception and all the errors leading up to it.

**Example 56–31 Listing the Error Stack**

```java
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 portal');
}
catch(GroupSpaceWSException ex)
{
    ex.printStackTrace();
}
...

56.1.7 Finding More Information on the WebCenter Portal API

API Reference Documentation
For detailed syntax information for the WebCenter Portal API, see Java API Reference for Oracle WebCenter Portal.

56.1.8 Troubleshooting Issues with the WebCenter Portal API
If you experience issues with the WebCenter Portal API, check the following:

1. Verify that the credential stores for both WebCenter Portal and the client Framework application are configured correctly.
   See the "Configuring the Policy and Credential Store" section in Administering Oracle WebCenter Portal.

2. Ensure the WebCenter Portal client context is set up correctly in the Portal Framework application by checking the alias passed in the context.setRecipientKeyAlias. The alias should be the public key alias of the producer (Portal), 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, Section 56.1.4.4, "Setting Up the WebCenter Portal Client Context."

3. Check that keystores exist at both ends of the connection. For example:
   - webcenter.jks (copied to Portal end)
   - clientapi.jks (copied to Portal Framework 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
keytool -importcert -alias clientapi -file clientapi.cer -keystore webcenter.jks -storepass mypassword

See also, Section 56.1.4.3, "Securing the Connection Between the Portal Framework Application and WebCenter Portal."

56.2 Using the WebCenter Portal REST API

WebCenter Portal provides a REST API to support various portal-related operations. You can use REST API methods to perform the following actions in the portal:

- List all portals in the application, list public portals only, list joined portals, list discoverable portals.
- View details for a single portal.
- List members of a portal.
- List role assignments for portal members.
- List custom attributes defined for a portal.
- View the details of a particular custom attribute.
- View and add lists for a portal.
- View and update list items (rows) for a list.
- View, update, and delete list item details (columns).

This section describes the Portal REST API. It contains the following subsections:

- Section 56.2.1, "Portal Entry Point"
- Section 56.2.2, "Portal Resource Type Taxonomy"
- Section 56.2.3, "Portal Security Considerations"
- Section 56.2.4, "Portal Resource Types"

For an introduction to the REST API, see Section 53, "Using Oracle WebCenter Portal REST APIs."

56.2.1 Portal 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 WebCenter Portal REST API, 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 portals accessible to the current user. The client sends HTTP requests to this entry point to work with WebCenter Portal.

For more information about the Resource Index, see Section 53.5.1, "Using the Resource Index."
For more information about resource types, see Section 53.5.2.1, "Resource Type."

56.2.2 Portal 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 56.2.4, "Portal Resource Types."

The taxonomy for Portal 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

56.2.3 Portal Security Considerations

There are no specific security considerations for this service. For general security considerations, see Section 53.8, "Security Considerations for WebCenter Portal REST APIs."

56.2.4 Portal Resource Types

The following sections provide all the information you need to know about each resource type:

- Section 56.2.4.1, "urn:oracle:webcenter:spaces"
- Section 56.2.4.2, "urn:oracle:webcenter:siteresources"
- Section 56.2.4.3, "urn:oracle:webcenter:spaces:resource:templates"
- Section 56.2.4.4, "urn:oracle:webcenter:space"
- Section 56.2.4.5, "urn:oracle:webcenter:space:resourceindex"
- Section 56.2.4.6, "urn:oracle:webcenter:space:roles"
- Section 56.2.4.9, "urn:oracle:webcenter:space:attributes"
- Section 56.2.4.10, "urn:oracle:webcenter:space:attribute"
- Section 56.2.4.11, "urn:oracle:webcenter:space:members"
- Section 56.2.4.12, "urn:oracle:webcenter:space:member"
56.2.4.1  urn:oracle:webcenter:spaces
Use this resource type to identify the URI to use to view a list of portals (GET) and to
create a new portal (POST). The response from a GET operation includes each portal,
and each portal includes links to operate on that portal.

Navigation Paths to portals
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 portal.
  - request - body: None
  - query parameters:
    * startIndex
    * itemsPerPage
    * projection - Allowed values are summary and details. The default is
    summary.
    * visibility - Determines which portals are included in the list: all portals,
    public portals only, joined portals, or discoverable portals.
      Allowed values are all, public, joined and discoverable. The default is
    joined.
  - response - body: collection of portals

- POST – Creates a new portal.
  - request - body: space

```xml
<space>
  <name>aName</name>
  <description>aDescription</description>
  <templateName>aTemplate</templateName>
</space>
```
Using the WebCenter Portal REST API

Integrating with WebCenter Portal

56.2.4.2 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 portal.

The returned list includes the resources that are found under the Resources tab of the WebCenter Portal 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**:
    
    * 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:**

---

**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 56–6** lists the resource types that the client can link to from this resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:spaces</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:space</td>
</tr>
</tbody>
</table>

---

**Table 56–6**  **Related Resource Types for urn:oracle:webcenter:spaces**

---

56.2.4.2 urn:oracle:webcenter:siteresources
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:siteTemplate;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 56–7 lists the read-only elements for siteresources.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>guid</td>
<td>String</td>
<td>Global ID for the site resource.</td>
</tr>
<tr>
<td>displayName</td>
<td>String</td>
<td>Name of the site resource as displayed to end users.</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the site resource. A complete list of all of the site resources, by name, is provided in Table 56–8.</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>Description of the site resource.</td>
</tr>
<tr>
<td>siteResourceType</td>
<td>String</td>
<td>Type of the site resource.</td>
</tr>
<tr>
<td>scopeName</td>
<td>String</td>
<td>Either the name of the space or the default scope at the application level.</td>
</tr>
<tr>
<td>seeded</td>
<td>Boolean</td>
<td>Set to true if the site resource is pre-populated.</td>
</tr>
<tr>
<td>visible</td>
<td>Boolean</td>
<td>Set to true if this site resource is visible in the user interface.</td>
</tr>
<tr>
<td>createdBy</td>
<td>String</td>
<td>Username of the user who created the site resource.</td>
</tr>
<tr>
<td>createdDate</td>
<td>Date</td>
<td>Date the site resource was created.</td>
</tr>
<tr>
<td>modifiedBy</td>
<td>String</td>
<td>Username of the user who updated the site resource.</td>
</tr>
<tr>
<td>modifiedDate</td>
<td>Date</td>
<td>Date on which the site resource was modified.</td>
</tr>
<tr>
<td>version</td>
<td>String</td>
<td>Version of the site resource.</td>
</tr>
</tbody>
</table>

**List of Site Resources by Name**

Table 56–8 lists the site resources by name, and gives the type and description of each resource.
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accordion View</td>
<td>contentPresenter</td>
<td>Displays multiple content items in an accordion (stacked) format. When an item is clicked, the other items collapse to reveal the selected items details.</td>
</tr>
<tr>
<td>Blank</td>
<td>pageStyle</td>
<td>Blank Page Style</td>
</tr>
<tr>
<td>Blank</td>
<td>taskFlowStyle</td>
<td>Blank</td>
</tr>
<tr>
<td>Blog</td>
<td>pageStyle</td>
<td>Blog Page Style</td>
</tr>
<tr>
<td>Bulleted View</td>
<td>contentPresenter</td>
<td>Displays multiple content items as a bulleted list. Only content items are displayed; sub-folders are omitted.</td>
</tr>
<tr>
<td>Bulleted with Folder Label</td>
<td>contentPresenter</td>
<td>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.</td>
</tr>
<tr>
<td>Carousel View</td>
<td>contentPresenter</td>
<td>Displays multiple content items in a carousel format. Users browse through items using a slider bar from left to right.</td>
</tr>
<tr>
<td>Default Document Details</td>
<td>contentPresenter</td>
<td>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.</td>
</tr>
<tr>
<td>Default Home Portal Catalog</td>
<td>resourceCatalog</td>
<td>Used when editing application-level pages and task flows in your Home portal.</td>
</tr>
<tr>
<td>Default List Item View</td>
<td>contentPresenter</td>
<td>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.</td>
</tr>
<tr>
<td>Default List Item View for</td>
<td>contentPresenter</td>
<td>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.</td>
</tr>
<tr>
<td>Folders</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Default Page Template Catalog</td>
<td>resourceCatalog</td>
<td>Used when editing page templates</td>
</tr>
<tr>
<td>Default Portal Catalog</td>
<td>resourceCatalog</td>
<td>Used when editing portal-level pages and task flows</td>
</tr>
<tr>
<td>Default View</td>
<td>contentPresenter</td>
<td>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.</td>
</tr>
<tr>
<td>Fusion FX</td>
<td>skin</td>
<td></td>
</tr>
<tr>
<td>Home Page</td>
<td>pageStyle</td>
<td>Home Page style</td>
</tr>
<tr>
<td>Icon View</td>
<td>contentPresenter</td>
<td>Displays multiple content items in a tiled format, using icons to represent sub-folders and content items.</td>
</tr>
</tbody>
</table>
### Table 56–8  List of Site Resources by Name Provided by WebCenter Portal (Cont.)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Left Narrow</td>
<td>pageStyle</td>
<td>Left Narrow Page Style</td>
</tr>
<tr>
<td>List View</td>
<td>contentPresenter</td>
<td>Displays multiple content items in a simple list.</td>
</tr>
<tr>
<td>List with Details Panel View</td>
<td>contentPresenter</td>
<td>Displays multiple content items in a list on the left. A panel on the right displays the details of a selected item.</td>
</tr>
<tr>
<td>Parameter Form</td>
<td>taskFlowStyle</td>
<td></td>
</tr>
<tr>
<td>Right Narrow</td>
<td>pageStyle</td>
<td></td>
</tr>
<tr>
<td>Sortable Table View</td>
<td>contentPresenter</td>
<td>Displays multiple content items in a sortable table that includes the document name, date created, and date modified.</td>
</tr>
<tr>
<td>Html Page</td>
<td>pageStyle</td>
<td>HTML page style</td>
</tr>
<tr>
<td>Spaces Default Navigation Model</td>
<td>navigation</td>
<td>Navigation model that displays current pages in a portal</td>
</tr>
<tr>
<td>Portals Navigation with Blogs/Wikis/Lists Submenus</td>
<td>navigation</td>
<td>Navigation model that displays current pages and submenus for blogs, wikis and lists</td>
</tr>
<tr>
<td>Stretch</td>
<td>taskFlowStyle</td>
<td></td>
</tr>
<tr>
<td>Tabbed View</td>
<td>contentPresenter</td>
<td>Displays multiple content items as tabs. Clicking a tab displays the associated item’s details.</td>
</tr>
<tr>
<td>Three Column</td>
<td>pageStyle</td>
<td>Three Column Page Style</td>
</tr>
<tr>
<td>Web Page</td>
<td>pageStyle</td>
<td>Web Page Page Style</td>
</tr>
<tr>
<td>Public-Pages Template</td>
<td>siteTemplate</td>
<td>Default Page Template for Public pages.</td>
</tr>
<tr>
<td>Portal FX</td>
<td>skin</td>
<td>Portal FX Skin</td>
</tr>
<tr>
<td>Wiki</td>
<td>pageStyle</td>
<td>Wiki Page Style</td>
</tr>
</tbody>
</table>

**Writable Elements for siteresources**

There are no writeable elements for this resource.
56.2.4.3 urn:oracle:webcenter:spaces:resource:templates

Use this resource type to identify the URI to use to retrieve a list of portal 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 portal 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 53.5.2.5, "Templates."

**Resource Types Linked to from templates**

Table 56–9 lists the resource types that the client can link to from this resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:spaces:resource:templates</td>
</tr>
</tbody>
</table>

56.2.4.4 urn:oracle:webcenter:space

Use this resource type to identify the URI to use to view details for a single portal (GET), create a nested portal (POST), or delete a portal (DELETE). The response from a GET operation includes the specific portal identified by the URI, including its creator, description, members, and custom attributes.

**Note:** The response from this resource also includes a link to the portal’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 portal that is a child of the named portal (a subportal). This feature supports creating hierarchical or nested portals.
  - **request - body:** Space
    
    ```xml
    <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 portal.
  - **request - body:** None
  
  On delete, a 204 status code is returned. If the portal does not exist or the user does not have permissions to delete the portal, a 403 Forbidden status is returned.

For more information about query parameters, see Section 53.5.2.5, "Templates."

Writable Elements for space
Table 56–10 lists the writable elements for this resource.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>String</td>
<td>Description of the portal.</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>Internal name for the portal.</td>
</tr>
</tbody>
</table>
Table 56–10 (Cont.) Writable Elements for urn:oracle:webcenter:space

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>templateName</td>
<td>String</td>
<td>Must be a valid template name returned from urn:oracle:webcenter:spaces:resource:templates. For example, the default template names are: Basic CommunityOfInterest ProjectSpace</td>
</tr>
</tbody>
</table>

Read-only Elements for space

Table 56–11 lists the read-only elements for this resource.

Table 56–11 Read-only Elements for urn:oracle:webcenter:space

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>guid</td>
<td>String</td>
<td>Global ID for the portal.</td>
</tr>
<tr>
<td>creator</td>
<td>String</td>
<td>ID of the user who created the portal.</td>
</tr>
<tr>
<td>author</td>
<td>personReference</td>
<td>User information about the user who created the portal, including GUID, ID, display name, and a link to the profile icon (same user as creator).</td>
</tr>
<tr>
<td>displayName</td>
<td>String</td>
<td>Name of the portal as displayed to members.</td>
</tr>
<tr>
<td>iconUrl</td>
<td>String</td>
<td>Fully qualified URL for the portal’s icon image.</td>
</tr>
<tr>
<td>logoUrl</td>
<td>String</td>
<td>Relative URL for the portal’s logo image.</td>
</tr>
<tr>
<td>isOffline</td>
<td>Boolean</td>
<td>Indicates whether the portal is offline or online.</td>
</tr>
<tr>
<td>isDiscoverable</td>
<td>Boolean</td>
<td>Specifies whether the portal is discoverable.</td>
</tr>
<tr>
<td>isPublic</td>
<td>Boolean</td>
<td>Specifies whether the portal is public.</td>
</tr>
<tr>
<td>keywords</td>
<td>String</td>
<td>Lists keywords for the portal.</td>
</tr>
<tr>
<td>mailingList</td>
<td>String</td>
<td>Lists the mailing list email for the portal if it has one.</td>
</tr>
<tr>
<td>parentDisplayName</td>
<td>String</td>
<td>Shows the parent display name if the portal is a child portal and has a parent. In this case, there will also be an isInheritFromParent boolean element that signifies whether the permissions of the parent portal are inherited by the child portal or not. Specifically, members of the parent portal will be inherited.</td>
</tr>
<tr>
<td>serviceDataCopied</td>
<td>String</td>
<td>List of service ids whose data is to be copied to new portals/templates when created based off this portal.</td>
</tr>
<tr>
<td>defaultLanguage</td>
<td>String</td>
<td>The default language.</td>
</tr>
<tr>
<td>landingPage</td>
<td>String</td>
<td>URI to the portal landing page if specified.</td>
</tr>
<tr>
<td>isTemplate</td>
<td>Boolean</td>
<td>Specifies whether or not this is a Template.</td>
</tr>
</tbody>
</table>
Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper

Table 56–11 (Cont.) Read-only Elements for urn:oracle:webcenter:space

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>isUnsubscriptionApproval Required</td>
<td>Boolean</td>
<td>Specifies whether or not you need approval to unsubscribe.</td>
</tr>
<tr>
<td>isPublishRss</td>
<td>Boolean</td>
<td>Specifies whether or not the portal has an associated RSS feed.</td>
</tr>
<tr>
<td>isClosed</td>
<td>Boolean</td>
<td>Specifies whether or not the portal is closed.</td>
</tr>
<tr>
<td>isBlockAllAccess</td>
<td>Boolean</td>
<td>Specifies whether or not all access to portal is blocked (not sure how this happens).</td>
</tr>
<tr>
<td>members</td>
<td>String</td>
<td>Returns the members element for the portal.</td>
</tr>
<tr>
<td>attributes</td>
<td>String</td>
<td>Returns the attributes element for the portal.</td>
</tr>
</tbody>
</table>

Resource Types Linked to from space

Table 56–12 lists the resource types that the client can link to from this resource.

Table 56–12 Related Resource Types for urn:oracle:webcenter:space

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:space</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:space:attributes</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:space:members</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:space:roles</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:space:icon</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:activities:stream</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:messageBoard</td>
</tr>
</tbody>
</table>

56.2.4.5 urn:oracle:webcenter:space:resourceindex

Use this resource type to identify the URI to use to view the Portal 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 56–13 lists the resource types that the client can link to from this resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>urn:oracle:webcenter:space:lists</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:discussions:forum</td>
</tr>
</tbody>
</table>

56.2.4.6 urn:oracle:webcenter:space:roles

Use this resource type to get a list of roles for the named portal (GET). The results returned from this call can be used to add a new member to a portal 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 portal.

Resource Types Linked to from roles

Table 56–14 lists the resource types that the client can link to from this resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:space:roles</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:space:role</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:space:role</td>
</tr>
<tr>
<td>urn:oracle:webcenter:parent</td>
<td>urn:oracle:webcenter:space</td>
</tr>
</tbody>
</table>

56.2.4.7 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 portal (GET). The results returned from this call can be used to add a new member to a portal 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 56–14 lists the resource types that the client can link to from this resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:space:role</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:space:role</td>
</tr>
<tr>
<td>urn:oracle:webcenter:parent</td>
<td>urn:oracle:webcenter:space</td>
</tr>
</tbody>
</table>

56.2.4.8 urn:oracle:webcenter:space:icon
Use this resource type to get the icon used for the named portal (GET).

Note: This resource type is used for portal references, not for portal responses. This resource can be used anywhere a portal is referenced, like in an activity stream or a portal 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 portal.

Resource Types Linked to from icon
Table 56–14 lists the resource types that the client can link to from this resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:space:icon</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:space</td>
</tr>
<tr>
<td>urn:oracle:webcenter:parent</td>
<td>urn:oracle:webcenter:space</td>
</tr>
</tbody>
</table>

56.2.4.9 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 portal (GET). The response from a GET operation includes each custom attribute for the portal, and each attribute includes links to operate on that attribute. Also use this resource type to add custom attributes to a portal (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 portal.
  - request - body:
    ```
    <attribute>
      <name>anAttribute</name>
      <value>aValue</value>
    </attribute>
    ```

Writeable Elements for attributes
Table 56–17 lists the writeable elements for this resource.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Attribute name.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>Attribute value.</td>
</tr>
</tbody>
</table>

Resource Types Linked to From attributes
Table 56–18 lists the resource types that the client can link to from this resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:space:attributes</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:space:attribute</td>
</tr>
<tr>
<td>urn:oracle:webcenter:space:parent</td>
<td>urn:oracle:webcenter:space</td>
</tr>
</tbody>
</table>

56.2.4.10 urn:oracle:webcenter:space:attribute
Use this resource type to identify the URI to use to view the name and value of single portal 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:
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 56–19 lists the read-only elements for this resource.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the custom attribute.</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>Description of the custom attribute.</td>
</tr>
<tr>
<td>type</td>
<td>String</td>
<td>Data type of the custom attribute.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>Value of the custom attribute.</td>
</tr>
</tbody>
</table>

Resource Types Linked to From attribute
Table 56–20 lists the resource types that the client can link to from this resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:space:attribute</td>
</tr>
<tr>
<td>urn:oracle:webcenter:parent</td>
<td>urn:oracle:webcenter:space</td>
</tr>
</tbody>
</table>

56.2.4.11 urn:oracle:webcenter:space:members
Use this resource type to identify the URI to use to view a list of members (enterprise users and groups) for the portal (GET). The response from a GET operation includes each member of the portal, 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:

Supported Methods for members
The following methods are supported by this resource:

- **GET**
  - **request - body**: None
Using the WebCenter Portal REST API

- **query parameters:** None
- **response - body:** collection of members

**POST** – Adds a member (user or group) to the portal.

- **request - body:** member
  
  To grant the participant role to the Sales group, use:
  
  ```xml
  <member>
    <name>sales</name>
    <role>Participant</role>
    <group>true</group>
  </member>
  ```

  To grant the participant role to the user Vicki, use:
  
  ```xml
  <member>
    <name>vicki</name>
    <role>Participant</role>
    <group>false</group>
  </member>
  ```

**DELETE** – Removes the named member (user or group) from the portal. 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 56–21 Writeable Elements of spaces:space:members

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The member's name.</td>
</tr>
<tr>
<td>role</td>
<td>String</td>
<td>The member's role. Obtain this value from <code>urn:oracle:webcenter:space:roles</code>.</td>
</tr>
</tbody>
</table>

#### Resource Types Linked to From members

Table 56–22 lists the resource types that the client can link to from this resource.

#### Table 56–22 Related Resource Types for urn:oracle:webcenter:space:members

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td><code>urn:oracle:webcenter:space:members</code></td>
</tr>
<tr>
<td></td>
<td><code>urn:oracle:webcenter:space:member</code></td>
</tr>
<tr>
<td>urn:oracle:webcenter:space:enter:parent</td>
<td><code>urn:oracle:webcenter:space</code></td>
</tr>
</tbody>
</table>

#### 56.2.4.12 urn:oracle:webcenter:space:member

Use this resource type to identify the URI to use to view details for a portal 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 56–23 lists the read-only elements for this resource.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>guid</td>
<td>String</td>
<td>Global ID for the portal member.</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>User ID of the portal member.</td>
</tr>
<tr>
<td>role</td>
<td>String</td>
<td>Role assigned to the member.</td>
</tr>
<tr>
<td>displayName</td>
<td>String</td>
<td>Display name of the member.</td>
</tr>
</tbody>
</table>

Resource Types Linked to From member
Table 56–24 lists the resource types that the client can link to from this resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:space:member</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:people:person</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:people:icon</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:people:person</td>
</tr>
</tbody>
</table>

56.2.4.13 urn:oracle:webcenter:space:lists
Use this resource type to identify the URI to use to view (GET) and add (POST) lists for the portal. The response from a GET operation includes each list in the portal, 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
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 53.5.2.5, "Templates.

Resource Types Linked to From lists
Table 56–25 lists the resource types that the client can link to from this resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:space:lists</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:space:list</td>
</tr>
</tbody>
</table>

56.2.4.14 **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 portal. 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 56–26 lists the writable elements for this resource.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Required</th>
<th>Constraints</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Yes</td>
<td>1 or more characters</td>
<td>Name of the portal list.</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>No</td>
<td>0 or more characters</td>
<td>Description of the list.</td>
</tr>
<tr>
<td>columns</td>
<td>urn:oracle:webcenter:space:list:columns</td>
<td>Yes</td>
<td>1 or more characters</td>
<td>Columns that make up the list.</td>
</tr>
</tbody>
</table>

**Read-only Elements for list**
Table 56–27 lists the read-only elements for this resource.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>ID of the list.</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the list.</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>Description of the list.</td>
</tr>
</tbody>
</table>
Integrating with WebCenter Portal

56.2.4.15 urn:oracle:webcenter:space:list

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

Table 56–27 (Cont.) Read-only Elements for urn:oracle:webcenter:list

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scopeguid</td>
<td>String</td>
<td>Global ID of the parent portal.</td>
</tr>
<tr>
<td>scopename</td>
<td>String</td>
<td>Name of the parent portal.</td>
</tr>
<tr>
<td>creator</td>
<td>String</td>
<td>ID of the user that created the list.</td>
</tr>
<tr>
<td>author</td>
<td>personReference</td>
<td>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).</td>
</tr>
<tr>
<td>created</td>
<td>Date</td>
<td>Date the list was created.</td>
</tr>
<tr>
<td>modifier</td>
<td>String</td>
<td>ID of the user that last modified the list.</td>
</tr>
<tr>
<td>modifiedByUser</td>
<td>personReference</td>
<td>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).</td>
</tr>
<tr>
<td>modified</td>
<td>Date</td>
<td>Date the list was last modified.</td>
</tr>
<tr>
<td>columns</td>
<td>String</td>
<td>Columns that make up the list.</td>
</tr>
</tbody>
</table>

Table 56–28 Related Resource Types for urn:oracle:webcenter:space:list

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:space:list</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:space:list:rows</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:space:list:columns</td>
</tr>
</tbody>
</table>
Using the WebCenter Portal REST API

- 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 53.5.2.5, “Templates.”

Resource Types Linked to From rows

Table 56–29 lists the resource types that the client can link to from this resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:space:list:rows</td>
</tr>
<tr>
<td>parent</td>
<td>urn:oracle:webcenter:space:list</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:space:list:row</td>
</tr>
</tbody>
</table>

56.2.4.16 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
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 53.5.2.5, "Templates."

Writable Elements for row
Table 56–30 lists the writable elements for this resource.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Required</th>
<th>Constraints</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>columns.column.id</td>
<td>String</td>
<td>Yes</td>
<td>1 or more characters</td>
<td>ID of the column (containing list item detail).</td>
</tr>
<tr>
<td>columns.column.name</td>
<td>String</td>
<td>No</td>
<td>1 or more characters</td>
<td>Name of the column.</td>
</tr>
<tr>
<td>columns.column.value</td>
<td>String</td>
<td>Yes</td>
<td>1 or more characters</td>
<td>Value of the column.</td>
</tr>
</tbody>
</table>

Read-only Elements for row
Table 56–31 lists the read-only elements for this resource.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>ID of the row (list item).</td>
</tr>
<tr>
<td>listId</td>
<td>String</td>
<td>ID of the list.</td>
</tr>
<tr>
<td>scope</td>
<td>String</td>
<td>Global ID of the parent portal.</td>
</tr>
<tr>
<td>creator</td>
<td>String</td>
<td>ID of the user that created the list item.</td>
</tr>
<tr>
<td>author</td>
<td>personReference</td>
<td>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).</td>
</tr>
<tr>
<td>created</td>
<td>Date</td>
<td>Date the list item was created.</td>
</tr>
<tr>
<td>modifier</td>
<td>String</td>
<td>ID of the user that last modified the list item.</td>
</tr>
</tbody>
</table>
Using the WebCenter Portal REST API

Table 56–31 (Cont.) Read-only Elements for urn:oracle:webcenter:space:list:row

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>modifiedByUser</td>
<td>personReference</td>
<td>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).</td>
</tr>
<tr>
<td>modified</td>
<td>Date</td>
<td>Date the list item was last modified.</td>
</tr>
</tbody>
</table>

Resource Types Linked to From row

Table 56–32 lists the resource types that the client can link to from this resource.

Table 56–32 Related Resource Types for urn:oracle:webcenter:space:list:row

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:space:list:row</td>
</tr>
</tbody>
</table>

56.2.4.17 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 53.5.2.5, “Templates.”
Resource Types Linked to From columns

Table 56–33 lists the resource types that the client can link to from this resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:space:list:columns</td>
</tr>
<tr>
<td>parent</td>
<td>urn:oracle:webcenter:space:list</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:space:list:column</td>
</tr>
</tbody>
</table>

56.2.4.18 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 56–34 lists the writable elements for this resource.
56.3 Exposing Enterprise Applications in Portals

You can expose various enterprise applications inside a portal, including:

- **Portal Framework 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 WebCenter Portal to consume and present application data in a unified way. The following sections tell you how.

This section includes the following subsections:

- **Section 56.3.1, "Exposing Portal Framework Applications in Portals"**
- **Section 56.3.3, "Exposing Non-Oracle Applications in Portals"**

56.3.1 Exposing Portal Framework Applications in Portals

You can expose Portal Framework applications, built using the WebCenter Portal Framework, as portlets and task flows in a portal:

- **Portlets**: Web components that are deployed inside a container and generate dynamic content. For more information, see Chapter 57, "Introduction to Portlets."

---

### Table 56–34  Writable Elements for urn:oracle:webcenter:space:list:column

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Required</th>
<th>Constraints</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>columns.column.id</td>
<td>String</td>
<td>Yes</td>
<td>1 or more characters</td>
<td>ID of the column (containing list item detail).</td>
</tr>
<tr>
<td>columns.column.name</td>
<td>String</td>
<td>No</td>
<td>1 or more characters</td>
<td>Name of the column.</td>
</tr>
<tr>
<td>columns.column.value</td>
<td>String</td>
<td>Yes</td>
<td>1 or more characters</td>
<td>Value of the column.</td>
</tr>
</tbody>
</table>

Table 56–35  Read-only Elements for urn:oracle:webcenter:space:list:column

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>ID of the column (list item detail).</td>
</tr>
</tbody>
</table>

Resource Types Linked to From column

Table 56–28 lists the resource types that the client can link to from this resource.

### Table 56–36  Related Resource Types for urn:oracle:webcenter:space:list:column

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>urn:oracle:webcenter:space:list:column</td>
</tr>
</tbody>
</table>
Exposing Enterprise Applications in Portals

- **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 58, "Creating Portlets from JSF Applications Using 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 the "Getting Started with ADF Task Flows" section in Fusion Developer’s Guide for Oracle Application Development Framework.

56.3.2 Exposing Other Oracle Applications in WebCenter Portal

You can expose Oracle Applications Unlimited products, for example, E-Business Suite, Siebel, PeopleSoft, and JDEdwards, in WebCenter Portal 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 WebCenter Portal.

- **Prebuilt portlets**: Enterprise applications such as Oracle E-Business Suite ship portlets out of the box that you can register with WebCenter Portal and consume in the same way as any other portlet.

For more information about integrating Oracle Applications Unlimited in WebCenter Portal and Portal Framework applications, see Chapter 54, "Integrating Other Oracle Applications."

56.3.3 Exposing Non-Oracle Applications in Portals

You can expose Web applications developed using non-Oracle platforms in the WebCenter Portal, 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 the "Registering External Applications" section in Administering Oracle WebCenter Portal, and the "Working with External Applications" chapter in Using Oracle WebCenter Portal.

- **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 the "Working with the Web Page Component" section in Building Portals with Oracle WebCenter Portal.

- **Raw HTML markup injection in Composer**: The Composer HTML Markup layout component injects HTML or JavaScript snippets into pages in Portal. You can use this component to display content from an external source. For more information see the "Working with the HTML Markup Component" section in Building Portals with Oracle WebCenter Portal.

- **OmniPortlet**: A portlet that publishes data from various data sources using a variety of layouts. For more information, see Chapter 64, "Creating Portlets with OmniPortlet."
Part X contains the following chapters:

- Chapter 57, "Introduction to Portlets"
- Chapter 58, "Creating Portlets from JSF Applications Using the Oracle JSF Portlet Bridge"
- Chapter 59, "Building Standards-Based Java Portlets Using JSR 286"
- Chapter 60, "Building Java Portlets Using the Oracle PDK-Java"
- Chapter 61, "Deploying Portlet Producers"
- Chapter 62, "Creating Pagelets with Pagelet Producer"
- Chapter 63, "Consuming Portlets"
- Chapter 64, "Creating Portlets with OmniPortlet"
- Chapter 65, "Creating Content-Based Portlets with Web Clipping"
This chapter provides an overview of portlets and describes their uses. It explains portlet anatomy and the resources you can use 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 topics:

- Section 57.1, "Introduction to Portlets"
- Section 57.2, "Portlet Anatomy"
- Section 57.3, "Portlet Resources"
- Section 57.4, "Portlet Development"

57.1 Introduction to Portlets

A portlet is a reusable web component that can draw content from many different sources. Portlets can contain anything from static HTML content to Java controls to complex web services and process-heavy applications.

Figure 57–1  The Lottery Sample 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 several different portlets can be placed on a
single page, users benefit from a single-source experience even though, in reality, the content may be derived from multiple sources.

Portlets can communicate with each other using events and other techniques. 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. Page designers can customize portlets to meet the needs of their specific audience. With the correct privileges, individual end users can further personalize portlets for their own particular requirements.

Oracle WebCenter Portal supports the development of portlets using JSR 286, an industry standard. You can also create portlets from existing JSF applications using the Oracle JSF Portlet Bridge. For more information, see Section 57.3, "Portlet Resources."

## 57.2 Portlet Anatomy

Portlet anatomy is the visual representation of the portlet on a page. Figure 57–2 shows some aspects of portlet anatomy that you might expect to see in a typical portlet in a WebCenter Portal Framework application. Note that the same portlet displayed in a different application could look different.

![A Sample Portlet Showing Typical Portlet Anatomy](image)

Aspects of portlet anatomy (some, but not all, of which are illustrated in Figure 57–2) include:

- **Portlet chrome**—A collective term for the different types of decoration that surround the portlet, including the header, shadow, resize handle, Actions menu, and icons.

- **Portlet header**—The area of the portlet that displays the portlet icon, title, Actions menu, and Customize and Personalize icons.

- **Portlet icon**—A small image in the portlet header used to visually identify the portlet.

- **Portlet title**—Text in the portlet header that indicates the purpose of the portlet. End users may be able to personalize this title at runtime to make it more meaningful to their particular usage.
- **Personalize icon**—An icon in the portlet header that enables end users to make changes to the portlet that only they can see.

  The **Personalize** icon is displayed only to authenticated users (that is, users who are logged in). It does not display to public users or unauthenticated users. You must implement some form of application security for users to be able to personalize their portlet views.

- **Customize icon**—An icon in the portlet header that enables application administrators to make changes to the default settings of the portlet at runtime. These changes are visible to all users.

  A typical customization setting is the portlet title. At runtime, the application administrator can determine what title should appear in the portlet header.

- **Actions icon**—An icon in the portlet header that displays the **Actions** menu when clicked.

- **Actions menu**—A menu that lists various actions that users can perform on the portlet. The actions listed depend on whether the user is logged in, what privileges that user has, the functionality provided by the portlet, and the options specified when the portlet was added to the page at design time. Actions include **Move**, **Remove**, **Refresh**, **Move Up**, and **Move Down**. The **Actions** menu also provides access to other portlet modes available for the portlet, such as About, Help, Configure, and Print.

  If the user who added the portlet to the page chose to not display the portlet header, the **Actions** menu is displayed on a fade in/fade out toolbar that displays on mouse rollover.

- **Resize handle**—An area at the bottom right of the portlet that enables users to change the size of the portlet.

- **Scroll bars**—Display when the portlet content does not fit the width and height specified for the portlet, providing access the content that is not initially displayed.

- **Portlet content**—The actual content of the portlet as determined by the portlet logic.

The portlet anatomy rendered on the page is controlled by two factors:

- The portlet’s own logic, as determined by the portlet developer. This includes which portlet modes are supported by the portlet.

- The attributes of the portlet tag that binds the portlet to the page, as determined by the application developer who added the portlet to the page at design time.

For example, when designing the portlet, the portlet developer may implement Help mode for the portlet. When an application developer adds the portlet to the page, he or she can determine, using the portlet tag attribute `isHelpModeAvailable`, whether or not to include the Help command in the portlet’s **Actions** menu. If the application developer sets `isHelpModeAvailable` to false, the Help command is not included in the **Actions** menu even though it is provided in the portlet logic. Conversely, if the portlet developer has not implemented Help mode for a portlet, the Help command is not displayed in the **Actions** menu, even if `isHelpModeAvailable` is set to true.

For information about the different attributes available for the portlet tag, see Section 63.6, "Setting Attribute Values for the Portlet Tag."
57.3 Portlet Resources

The portlets in your applications can come from a variety of sources, including other Oracle products and third-party vendors. Portlet resources also include custom built Java portlets built using WebCenter Portal's Java portlet wizard for JSR 286. Each of these resources offers different product features and are targeted toward users with different levels of experience.

This section includes the following topics:

- Section 57.3.1, "Prebuilt Portlets"
- Section 57.3.2, "Java Server Faces (JSF) Portlets"
- Section 57.3.3, "Custom Java Portlets"
- Section 57.3.4, "OmniPortlet"
- Section 57.3.5, "Web Clipping Portlet"
- Section 57.3.6, "Portlet Technologies"
- Section 57.3.7, "When To Use Portlets Versus Task Flows?"

57.3.1 Prebuilt Portlets

What Are They?
Oracle provides integration with other Oracle applications by enabling you to expose functionality as portlets:

- **Peoplesoft**—You can expose Peoplesoft applications as WSRP portlets. For more information, see Section 54.5, "Integrating PeopleSoft Applications."
- **JD Edwards**—You can expose JD Edwards standalone regions as portlets. For more information, see Section 54.4, "Integrating JD Edwards Applications."
- **Oracle E-Business Suite**—Oracle E-Business Suite provides several prebuilt portlets, such as Applications Navigator, Favorites, and Worklist, that you can add to your Portal Framework applications. For more information, see Section 54.3, "Integrating E-Business Suite Applications."

Other prebuilt portlets are available through Oracle's partnerships with leading system integrators, software vendors, and content providers.

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. They are available for use by all levels of experience.

When Should They Be Used?
Use prebuilt portlets when your needs are met 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.
57.3.2 Java Server Faces (JSF) Portlets

**What Are They?**
JSF portlets are created from JSF applications 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 Java portlets. The Oracle JSF Portlet Bridge simplifies the integration of JSF applications with WSRP portlet consumers, such as WebCenter Portal.

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 portletized JSF application, the JSF portlets are also deployed with it. Therefore, using the Oracle JSF Portlet Bridge eliminates the requirement to store, maintain, and deploy your portlets separately from your application.

For more information about creating JSF portlets, see Chapter 58, "Creating Portlets from JSF Applications Using the Oracle JSF Portlet Bridge."

**Who Is the Intended User?**
Application developers with knowledge of Faces.

**When Should They Be Used?**
JSF portlets are best suited when application developers intend to display content from a JSF application, or from individual task flows within the 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 the same as registering any WSRP producer.

57.3.3 Custom Java Portlets

Custom Java portlets are portlets that you write yourself, in Java, using the Java Portlet Specification 2.0 (JSR 286). WebCenter Portal provides a declarative wizard in JDeveloper for simplifying the creation of standards-based JSR 286 Java portlets. This wizard assists in the construction of the framework within which you create the portlet.

---

**Note:** You can consume legacy JSR 168 and WSRP 1.0 portlets in your applications, for example from Oracle WebLogic Portal. There is no need to convert these portlets to JSR 286 first. However, when creating new Java portlets we recommend using JSR 286 to take advantage of features such as public render parameters and portlet events.

---

**Who Is the Intended User?**
Use of the wizard is easy, but creation of the portlet logic is best performed by experienced and knowledgeable Java developers who are comfortable with JSR 286 and who understand the configuration of producers.

**When Should They Be Used?**
Consider using custom Java portlets when you cannot find a prebuilt portlet or existing JSF application to address your requirements.
Use custom Java portlets when you have very specialized business rules or logic, or when you require personalized authentication, granular processing of dynamic results, or complete user interface control.

For more information about creating custom Java portlets, see Chapter 59, "Building Standards-Based Java Portlets Using JSR 286."

57.3.4 OmniPortlet

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.

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.

For more information about OmniPortlet, see Chapter 64, "Creating Portlets with OmniPortlet."

As an alternative to OmniPortlet, if you want to expose data from a SQL database or web service, you should consider using Data Presenter. For more information, see the "Working with Data Presenter" chapter in Building Portals with Oracle WebCenter Portal.

57.3.5 Web Clipping Portlet

Note: Instead of using the Web Clipping portlet, consider using a clipper pagelet using Oracle WebCenter Portal’s Pagelet Producer. For more information, see the "Managing the Pagelet Producer" chapter in Administering Oracle WebCenter Portal.

What Is It?

The Web Clipping portlet is a browser-based declarative tool that enables the integration of any web application with a Framework application. The Web Clipping portlet is designed to provide quick integration by leveraging the web application’s existing user interface. The Web Clipping portlet has been implemented as an Oracle PDK-Java portlet.

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.

Who Is the Intended User?

The Web Clipping portlet 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 the Web Clipping portlet. (For more information, see Section 57.3.4, "OmniPortlet."

For more information about using the Web Clipping portlet, see Chapter 65, "Creating Content-Based Portlets with Web Clipping."

### 57.3.6 Portlet Technologies

When creating new portlets, there are several different technologies you can use. The technology you use depends on several factors.

*Figure 57–3* provides a decision tree to enable you to determine which portlet implementation technology best fits your environment and requirements. The list below *Figure 57–3* presents the same information in a non-graphical format and provides additional notes.
1. Do you need your components to be distributed?
   - Yes—Go to Question 2.
- **No**—You don’t need portlets. Use Oracle ADF task flows (or some other technology) that are local to your consuming application. For more information, see the “Creating ADF Task Flows” part in the Fusion Developer’s Guide for Oracle Application Development Framework.

Having a distributed environment results in a more complicated deployment and potentially makes it more difficult to track down issues. However, it also means that you can spread the processing load across multiple servers and enables your components to be used by several different consumers.

2. Do you need to expose .NET components to the consuming application?
   - **Yes**—Use the Oracle WebCenter WSRP Producer for .NET. For more information see the Developer’s Guide for Oracle WebCenter WSRP Producer for .NET.
   - **No**—Go to Question 3.

   The Oracle WebCenter WSRP Producer for .NET allows any .NET application to act as a WSRP producer.

3. Do you have existing Oracle ADF or JSF pages or task flows that you want to expose remotely?
   - **Yes**—Use the Oracle JSF Portlet Bridge to expose your existing components. For more information see Chapter 58, "Creating Portlets from JSF Applications Using the Oracle JSF Portlet Bridge."
   - **No**—Go to Question 4.

   Using existing Oracle ADF components in this way provides a very convenient way of developing portlets, but be aware of the implications of using Oracle ADF to implement portlets, as outlined in Table 57-1.

4. Do you have existing JSR 168 or JSR 286 portlets?
   - **Yes**—Use your existing JSR 168 or JSR 286 portlets. For more information, see Chapter 63, "Consuming Portlets."
   - **No**—Go to Question 5.

   WebCenter Portal exposes JSR 168 and JSR 286 portlets over WSRP, which are fully compatible with the WebCenter consumer, assuming such portlets do not make assumptions that are specific to a particular consumer.

5. Do you have existing Oracle PDK-Java portlets that will work correctly with the WebCenter Portal consumer?
   - **Yes**—Use your existing PDK-Java portlets. For more information, see Chapter 63, "Consuming Portlets."
   - **No**—Go to Question 6.

6. Do you have a requirement for your components to be rendered directly in the consumer page (that is, not in an inline frame)?
   - **Yes**—Write JSR 286 portlets that do not use markup that requires them to be rendered in an inline frame. For more information, see Chapter 59, "Building Standards-Based Java Portlets Using JSR 286."
   - **No**—Go to Question 7.

   See Section 63.5.4, "What You May Need to Know About Inline Frames" for considerations relating to the use of inline frames to render portlets.

7. Do you want to use Oracle ADF to implement your component?
Yes—Use the Oracle JSF Portlet Bridge to expose Oracle ADF pages or task flows as portlets. For more information, see Chapter 58, "Creating Portlets from JSF Applications Using the Oracle JSF Portlet Bridge."

No—Write JSR 286 portlets. For more information, see Chapter 59, "Building Standards-Based Java Portlets Using JSR 286."

The decision of whether to use Oracle ADF for implementing portlets is broadly the same as whether you would choose to use it for implementing a normal web application. However, the additional network hop (for markup, resource, and PPR) and consumer processing that is inherent when using remote portlets may make Oracle ADF a less attractive option than it is in the non-portlet case.

Also, see the Oracle ADF Overview white paper on OTN for specific benefits of using Oracle ADF to develop Java EE applications. Another benefit of using an Oracle ADF task flow is that you can later run the same task flow locally if you encounter any issues running in an inline frame.

Table 57–1 outlines the implications of choosing a particular portlet technology for implementing components for use with WebCenter Portal against various criteria.

<table>
<thead>
<tr>
<th>Portlet Building Technologies Comparison Matrix</th>
<th>Local Oracle ADF Pages and Task Flows</th>
<th>WebCenter WSRP Producer for .NET</th>
<th>JSR 286 Portlets</th>
<th>JSF Portlets</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distributed Architecture</td>
<td>No.</td>
<td>Yes, using WSRP SOAP/HTTP.</td>
<td>Yes, using WSRP SOAP/HTTP.</td>
<td>Yes, using WSRP SOAP/HTTP.</td>
</tr>
<tr>
<td>Data Access Support</td>
<td>No restrictions. This leverages the underlying .NET framework, which provides data access to a wide variety of data repositories.</td>
<td>No restrictions, but must be implemented by the portal developer, possibly using a third party framework.</td>
<td>Full support for working with databases, web services, and legacy systems.</td>
<td></td>
</tr>
<tr>
<td>Navigation/Control Flow Support</td>
<td>Provides comprehensive support for declarative navigation.</td>
<td>Provides comprehensive support for declarative navigation.</td>
<td>No restrictions, but must be implemented by the portal developer, possibly using a third party framework.</td>
<td>Provides comprehensive support for declarative navigation.</td>
</tr>
<tr>
<td>Richness of UI</td>
<td>Provides a rich, web-based UI with many AJAX-enabled components.</td>
<td>No restrictions. Supports both standard AJAX and ASP.NET AJAX toolkit.</td>
<td>No restrictions, but must be implemented by the portal developer, possibly using a third party framework.</td>
<td>Provides a rich, web-based UI with many AJAX-enabled components.</td>
</tr>
<tr>
<td>Security</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Introduction to Portlets

- **Fully integrated with the application.**
  - User identity propagated using WS-Security or asserted by consumer using WSRP.
  - Authorization checks must be implemented by the developer.
  - User profile information can be passed using WSRP user profile items.

### Inter-Component Communication (Events)

- **ADF contextual events enable task flows to communicate.**
  - We provide samples to show how it can be accomplished client-side. Current version does not support WSRP server-side eventing model.
  - Full support for JSR 286 events, including automatic event delivery between portlets and event mapping between portlet events and ADF contextual events in the consumer.
  - As per JSR 286 portlets in addition to ADF contextual events allowing task flows to communicate as in the local case.

### Consumer to Component Interaction (Parameters)

- **Task flow input parameters can take values from various sources.**
  - Full support for WSRP portlet parameters. All parameters can be passed from the consumer application.
  - Full support for JSR 286 parameters, including passing public parameter values to and from the consumer and automatic sharing of parameter values between portlets.
  - As per JSR 286 portlets in addition to task flow input parameters taking values from various sources in the consumer application as in the local case.

### Performance and Caching

- **No additional network hop since the component runs on the local server.**
  - Remote component requires additional network hop.
  - Configuration parameters allow for resources (CSS, JavaScript, and .axd resource files) to be proxied via the consumer or direct access to reduce load and take advantage of browser caching.
  - Remote component requires additional network hop.
  - Resources should be proxied by the consumer. Component performance depends on implementation.
  - Expires and validation based caching are supported and controlled by the portlet developer.
  - Remote component requires additional network hop.
  - Many resources are proxied via the consumer. Heavyweight nature of framework means simple applications are likely to suffer performance-wise versus handcrafted JSR 286 equivalents. However, for applications offering more complex functionality, the performance benefits of using the framework (for example, partial page rendering) become more apparent.
  - JavaScript, CSS, and other resources are cached in the browser. Caching is controlled by the framework.

### Concurrency

- **Multiple task flows are invoked sequentially.**
  - Portlets are invoked in parallel.
  - Portlets are invoked in parallel.
  - Portlets are invoked in parallel.

---

**Table 57–1 (Cont.) Portlet Building Technologies Comparison Matrix**

<table>
<thead>
<tr>
<th>Local Oracle ADF Pages and Task Flows</th>
<th>WebCenter WSRP Producer for .NET</th>
<th>JSR 286 Portlets</th>
<th>JSF Portlets</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fully integrated with the application</td>
<td>User identity propagated using WS-Security or asserted by consumer using WSRP. Authorization checks must be implemented by the developer. User profile information can be passed using WSRP user profile items.</td>
<td>User identity propagated using WS-Security or asserted by consumer using WSRP. Authorization checks must be implemented by the developer. User profile information can be passed using WSRP user profile items.</td>
<td>User identity propagated using WS-Security. Authorization is implemented by Oracle ADF.</td>
</tr>
<tr>
<td>Inter-Component Communication (Events)</td>
<td>We provide samples to show how it can be accomplished client-side. Current version does not support WSRP server-side eventing model.</td>
<td>Full support for JSR 286 events, including automatic event delivery between portlets and event mapping between portlet events and ADF contextual events in the consumer.</td>
<td>As per JSR 286 portlets in addition to ADF contextual events allowing task flows to communicate as in the local case.</td>
</tr>
<tr>
<td>Consumer to Component Interaction (Parameters)</td>
<td>Task flow input parameters can take values from various sources.</td>
<td>Full support for WSRP portlet parameters. All parameters can be passed from the consumer application.</td>
<td>Full support for JSR 286 parameters, including passing public parameter values to and from the consumer and automatic sharing of parameter values between portlets.</td>
</tr>
<tr>
<td>Performance and Caching</td>
<td>Configuration parameters allow for resources (CSS, JavaScript, and .axd resource files) to be proxied via the consumer or direct access to reduce load and take advantage of browser caching.</td>
<td>Remote component requires additional network hop. Resources should be proxied by the consumer. Component performance depends on implementation. Expires and validation based caching are supported and controlled by the portlet developer.</td>
<td>Remote component requires additional network hop. Many resources are proxied via the consumer. Heavyweight nature of framework means simple applications are likely to suffer performance-wise versus handcrafted JSR 286 equivalents. However, for applications offering more complex functionality, the performance benefits of using the framework (for example, partial page rendering) become more apparent. JavaScript, CSS, and other resources are cached in the browser. Caching is controlled by the framework.</td>
</tr>
<tr>
<td>Concurrency</td>
<td>Multiple task flows are invoked sequentially.</td>
<td>Portlets are invoked in parallel.</td>
<td>Portlets are invoked in parallel.</td>
</tr>
</tbody>
</table>
**Table 57–1 (Cont.) Portlet Building Technologies Comparison Matrix**

<table>
<thead>
<tr>
<th>Local Oracle ADF Pages and Task Flows</th>
<th>WebCenter WSRP Producer for .NET</th>
<th>JSR 286 Portlets</th>
<th>JSF Portlets</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Geometry Management</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Components are rendered directly in the containing page.</td>
<td>Portlets are automatically rendered in a proxied inline frame and resized accordingly. The slider bars are hidden from view.</td>
<td>Portlet geometry and inline versus framed rendering is controlled by the portlet developer. Complex UIs are more likely to require rendering in an inline frame.</td>
<td>Portlets must be rendered in an inline frame.</td>
</tr>
<tr>
<td><strong>Use of JavaScript</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JavaScript used heavily as a fundamental part of the framework to provide a rich user experience. Developer does not generally need to write JavaScript.</td>
<td>JavaScript used heavily as a fundamental part of the framework to provide a rich user experience. Developer does not generally need to write JavaScript, but samples are provided to show how JavaScript can be used. All scripts can be proxied via the consumer.</td>
<td>Level of JavaScript use and packaging are controlled by the portlet developer. Developers must be aware of special considerations regarding the use of JavaScript if portlets are to be rendered inline.</td>
<td>JavaScript used heavily as a fundamental part of the framework to provide a rich user experience. Developer does not generally need to write JavaScript. All scripts must be proxied via the consumer.</td>
</tr>
<tr>
<td><strong>Skills Requirements/Ease of Development</strong></td>
<td>Requires knowledge of ADF and possibly Java. ADF provides an integrated framework across all layers. JDeveloper provides a fully integrated environment for developing ADF applications. This includes declarative options for all layers of the framework.</td>
<td>Requires knowledge of .NET web parts development or ASP.NET web application development.</td>
<td>Requires knowledge of Java, Java portlets, and any other frameworks used to implement components. Development time comparable with that required to develop a servlet-based component.</td>
</tr>
<tr>
<td><strong>Debugging</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JDeveloper provides a comprehensive solution for debugging application code.</td>
<td>.NET framework provides a comprehensive solution for debugging application code and network trace for SOAP and HTTP traffic. Log4net plug-in for WSRP consumer-producer debug messages.</td>
<td>Made more difficult by remote nature of deployment. Producer provides tool for monitoring consumer-producer communication. JDeveloper provides a comprehensive solution for debugging application code.</td>
<td>Made more difficult by remote nature of deployment. Producer provides tool for monitoring consumer-producer communication. JDeveloper provides a comprehensive solution for debugging application code.</td>
</tr>
<tr>
<td><strong>Deployment</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
57.3.7 When To Use Portlets Versus Task Flows?

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 portals built with WebCenter Portal. You can drag and drop task flows onto any customizable WebCenter Portal portal 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. 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 58, "Creating Portlets from JSF Applications Using 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 the “Getting Started with ADF Task Flows” chapter in the *Fusion Developer’s Guide for Oracle Application Development Framework*.

For more information about when to use portlets versus task flows, see the “Oracle WebCenter & ADF Architecture Team” blog.

### 57.4 Portlet Development

This section includes the following topics:

- Section 57.4.1, "Portlet Producer Applications"
- Section 57.4.2, "Portlet Modes"
- Section 57.4.3, "Interportlet Communication"
- Section 57.4.4, "Portlet Personalization and Customization"
- Section 57.4.5, "Portlet Performance"
- Section 57.4.6, "Multilanguage Portlets"
- Section 57.4.7, "Portlet Deployment"

#### 57.4.1 Portlet Producer Applications

A portlet producer application is an application specifically configured for building and deploying Java portlets.

To readily find and use the appropriate components in your portlet producer application, you must ensure that the appropriate 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.

JDeveloper provides an application template to ensure that scopes are set properly and the right tag and Java libraries are added to create portlet producer applications. The WebCenter Portlet Producer Application template provides a way of easily creating an application with a single project scoped for the creation of Java portlets.

To create an application using the WebCenter Portlet Producer Application template:

1. Access the Create WebCenter Portlet Producer Application wizard in any of the following ways:

   - From the File menu, choose New. In the New Gallery dialog, expand General, select Applications then WebCenter Portlet Producer Application, and then click OK.
   - From the Application menu, choose New. In the Create Generic Application wizard, in the Application Template list select WebCenter Portlet Producer Application.
   - If you have an existing application open, in the Application Navigator, click the application name and choose New Application. In the Create Generic Application wizard, in the Application Template list select WebCenter Portlet Producer Application.
   - If you have an existing application open, then in the Application Navigator, right-click the application name and choose New. In the New Gallery dialog, expand General, select Applications then WebCenter Portlet Producer Application, and then click OK.
2. In the Name your application page of the creation wizard, in the Application Name field, enter a name for the application.

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\myApplication

   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 the application.

5. Click Finish to create the portlet producer application with default project configurations.

   Figure 57–4 shows a newly created portlet producer application, with its single Portlets project, in the Application Navigator.

   **Figure 57–4  New Portlet Producer Application in the Application Navigator**

   ![Application Navigator with Portlets project]

   Options exposed in the JDeveloper user interface are limited to those appropriate for developing portlets.

   When you right-click the Portlets project and choose New, options to create Java portlets are provided in the New Gallery (Figure 57–5).
Portlet Development

### 57.4.2 Portlet Modes

Portlet modes exhibit the runtime functionality seen by users. WebCenter Portal supports the following standard portlet modes:

- View mode
- Edit mode
- Edit Defaults mode
- Help mode
- About mode

WebCenter Portal defines an extended set of modes that it supports. Different modes are available to JSR 286 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 accommodate any other functionality you may want the portlet to provide.

Defining custom portlet 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.
This section includes the following topics:

- Section 57.4.2.1, "View Mode"
- Section 57.4.2.2, "Edit Mode"
- Section 57.4.2.3, "Edit Defaults Mode"
- Section 57.4.2.4, "Help Mode"
- Section 57.4.2.5, "About Mode"

57.4.2.1 View Mode

A portlet uses View mode 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, other modes are optional. 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 end user settings, the browser width, and the amount and style of content in the portlet.

57.4.2.1.1 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, and 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 to not appear at all. Ensure that you adhere to the following rules:

- **Use standard HTML**—The official HTML specification is available from the W3C. For more information, see [http://www.w3c.org/](http://www.w3c.org/).

- **Avoid lengthy, complex HTML**—The HTML you use 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. If portlets must render complex HTML or wait for external resources, such as third party applications, then it can greatly slow down the rendering of the page.

- **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.

- **Consider restrictions imposed by container objects**—If your portlet is contained inside 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 cannot 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.

- **Do not create fixed-width HTML tables in portlet**—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 some browsers.

- **Avoid long, unbroken lines of text**—The result is similar to what happens with fixed-width tables. Your portlet might overlap other portlets in some 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**—End users may choose whatever font size they want and they can change it at any time. Your portlet should handle these situations gracefully.

### 57.4.2.1.2 CSS 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 Cascading Style Sheets (CSS). 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 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. Consider the following guidelines for using CSS:

- **Use CSS instead of hard coding**—Hard coding fonts and colors is not advised. If you hard code fonts and colors, then your portlet may look out of place if the end user changes the page style settings. Since you have no way of knowing the end user's font and color preferences, 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's content will not be visible to that user.

- **Avoid using CSS for absolute positioning**—Since users may be able to personalize pages, you cannot guarantee that your portlet can appear in a particular spot on the page.

- **Follow accessibility standards**—You should ensure that you code your style sheets according to existing accessibility standards. For more information, see [http://www.w3c.org/TR/WCAG10-CSS-TECHS](http://www.w3c.org/TR/WCAG10-CSS-TECHS).

### 57.4.2.2 Edit Mode

A portlet uses *Edit mode* to enable users to personalize the behavior of the portlet. Personalizations are visible only to the user who performs 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 else that affects the appearance or content of the portlet.

Users typically access a portlet's Edit mode by clicking the **Personalize** icon in the portlet header (Figure 57–6).
When users click **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.

### 57.4.2.2.1 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 end 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 from the cache. If you do not do this, the user may see incorrect content.

- **Do not use Edit mode as an administrative tool**—Edit mode is meant to give end users a way of changing the behavior of portlets. If you want to change producer settings or perform other administrative tasks, then you should create secured portlets specifically for these tasks.

### 57.4.2.2 Guidelines for Buttons in Edit Mode

For consistency and user convenience, Edit mode should implement the following buttons in the order listed:

- **OK**—Saves the end user’s personalizations and returns the portlet to View mode.

- **Apply**—Saves the end user’s personalizations and reloads the current page.

- **Cancel**—Returns the portlet to View mode without saving changes.

### 57.4.2.3 Edit Defaults Mode

A portlet uses **Edit Defaults mode** to enable application administrators to customize the default behavior of a particular portlet instance at runtime. Edit Defaults mode provides a list of settings that the application administrator can change. These settings may include the title, type of content, formatting, amount of information, defaults for form elements, or anything that affects the appearance or content of the portlet.

Whereas personalizations affect a portlet instance for an individual user, **customizations** made in Edit Defaults mode affect the portlet instance 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.

Application administrators access Edit Defaults mode, when editing a page, by clicking the **Customize** icon in the portlet header (**Figure 57–7**).

*Figure 57–7 The Customize Icon for Accessing a Portlet’s Edit Defaults Mode*

When users click **Customize**, a new page appears in the same browser window. The portlet typically creates a web page representing a dialog to customize the portlet.
instance settings. After applying the settings, users are automatically returned to the original page.

57.4.2.3.1 Guidelines for Edit Defaults Mode Operation The following guideline should govern what you expose to application administrators in Edit Defaults mode:

■ **Do not use Edit Defaults mode as an administrative tool**—Edit Defaults mode gives application administrators 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.

57.4.2.3.2 Guidelines for Buttons in Edit Defaults Mode For consistency and user convenience, Edit Defaults mode should implement the following buttons in the order listed:

■ **OK**—Saves the application administrator's customizations and returns the portlet to View mode.

■ **Apply**—Saves the application administrator's customizations and reloads the current page.

■ **Cancel**—Returns the portlet to View mode without saving changes.

57.4.2.3.3 Guidelines for Rendering Customization Values When you show the forms used to change customization settings, you should default the values so that the application administrator does not have to constantly reenter settings. When rendering customization values, use the following sequence to provide consistent behavior:

■ **Instance preferences**—Query and display system defaults for the portlet instance.

■ **Portlet defaults**—If no system default customizations are found, then display general portlet defaults, which may be blank. General portlet defaults are sometimes hard coded into the portlet but should be overwritten by system defaults.

57.4.2.4 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 option in the Actions menu (Figure 57–8).

*Figure 57–8 The Help Option in the Portlet Actions Menu*
57.4.2.4.1 Guidelines for Help Mode The following guideline should govern what you expose in Help mode:

- **Describe how to use the portlet**—Users may not be able to ascertain all the features your portlet offers just from its interface. Describe the features and how to get the most out of them.

57.4.2.5 About Mode

A portlet uses **About mode** to provide users with information such as the version of the portlet that is currently running, its publication and copyright information, and how to contact the portlet developer. Portlets that require registration could also use About mode to link to a web-based registration application or contact information.

Users access a portlet’s About mode by choosing the **About** option in the **Actions** menu (Figure 57–9).

**Figure 57–9 The About Option in the Portlet Actions Menu**

A new page appears in the same browser window. The portlet can either generate the content for this page or take the user to an existing page or application.

57.4.2.5.1 Guidelines for About Mode The following guideline should govern what you expose to users in About mode:

- **Display relevant copyright, version, and developer 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.

57.4.3 Interportlet Communication

Interportlet communication enables portlets to share data and react to events. WebCenter Portal supports interportlet communication through parameters and events:

- **Parameters**—Public render parameters enable interportlet communication by sharing parameter values between portlets. For example, if a Map portlet and a Weather portlet are both configured to use a Zipcode parameter, entering a zip code in the Map portlet updates the same parameter value in the Weather portlet.

  For information about public render parameters, see Section 59.3.5, "How to Use Public Render Parameters in JSR 286 Portlets."

- **Events**—Events enable interportlet 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.

For information about portlet events, see Section 59.3.6, "How to Use Portlet Events in JSR 286 Portlets."

In WebCenter Portal, interportlet communication happens automatically by default, as long as parameters and events share the same name or define an appropriate alias. You do not need to supply any extra code to enable this communication. For more information, see Section 63.5.3, "What You May Need to Know About Interportlet Communication."

For interportlet communication guidelines specific to portlets created using the Oracle JSF Portlet Bridge, see Section 58.7, "Using Events to Link JSF Portlets with Other Portlets."

57.4.4 Portlet Personalization and Customization

Portlet preferences enable end users to personalize or customize portlets at runtime. Personalizations are visible only to the user that performs them; not to other users. Customizations are visible to all users who have not personalized the portlet.

Portlet preferences greatly enhance the reusability of portlets. The same portlet can be instantiated on multiple pages, or multiple times on the same page. Using portlet preferences, each instance of the portlet can behave in a different way, but still use the same code and user interface.

For example, by default when you create a JSR 286 portlet using the JDeveloper wizard, a portlet preference is created to enable users to change 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 users to perform other modifications on the portlet at runtime.

Portlet preferences are stored in the persistence store, along with information about the registration and portlet instances, such as registration and portlet handles. Portlet producers can use one of three types of persistence store:

- **Consumer**—A consumer persistence store ties the producer metadata to the consumer application. It is recommended that you should use a consumer persistence store in your production environment.
- **Database**—A database persistence store persists data using a relational database.
- **File**—A file-based persistence store persists data using the file system.

A file-based or consumer persistence store may be used for testing, since it removes the dependency on a database. But, for production configurations, it is recommended that you use a consumer persistence store.

By default, JSR 286 and JSF portlets use a consumer persistence store. For information about how to change this default setting, see Section 59.3.14.1, "Setting Up a Persistence Store for a WSRP Producer."

For information about migrating the persistence store, for example, when moving from a test to production environment, see Section 59.3.14.2, "Migrating a WSRP Producer Persistence Store."

57.4.5 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:

- 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**.

Portlet producers generate dynamic content (that is, portlets) and they reside remotely from the Portal 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. Portlet caching is key to rapid response to user requests.

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.

  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.

  For information about how to implement expiry-based caching for your portlets, see Section 59.3.10.1, "Implementing Expiry-Based Caching in JSR 286 Portlets."

- **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 content, it must determine whether the cached content 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 that 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 cached content, rather than relying on a fixed period.

  Consider using validation-based caching for portlets with dynamic content that changes frequently or unpredictably.

  For information about how to implement validation-based caching for your portlets, see Section 59.3.10.2, "Implementing Validation-Based Caching in JSR 286 Portlets."
Caching rules can be specified at a portlet's container level, encoded in the portlet's own logic, or 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 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.

JSF portlets and custom Java portlets support expiry- and validation-based caching.

### 57.4.6 Multilanguage Portlets

There are three steps involved in making your portlets available in multiple languages:

- **Declare the locales supported by the portlet** using the `<supported-locale>` element in the `portlet.xml` file. For example:

  ```xml
  <portlet id="1328624289503">
    ...
    <supported-locale>en</supported-locale>
    <supported-locale>fr</supported-locale>
    <supported-locale>es</supported-locale>
    ...
  </portlet>
  ```

- **Make the portlet metadata that is displayed in the portlet** (for example, the portlet title, description, keywords, and so on) translatable. You can include the translations directly in the `portlet.xml` file, for example:

  ```xml
  <portlet id="1328624289503">
    <portlet-name>LotteryPortlet</portlet-name>
    <display-name xml:lang="en">Lottery Numbers</display-name>
    <display-name xml:lang="fr">Numéros de Loterie</display-name>
    <display-name xml:lang="es">Números de la Lotería</display-name>
    ...
    <supported-locale>en</supported-locale>
    <supported-locale>fr</supported-locale>
    <supported-locale>es</supported-locale>
    ...
  </portlet>
  ```

  Alternatively, you can provide translations in a separate resource bundle. You must declare the resource bundle in the `portlet.xml` file, for example:

  ```xml
  <portlet id="1328624289503">
    <portlet-name>LotteryPortlet</portlet-name>
    <display-name>Lottery Numbers</display-name>
    ...
    <supported-locale>en</supported-locale>
    <supported-locale>fr</supported-locale>
    <supported-locale>es</supported-locale>
    <resource-bundle>portlet.resource.LotteryPortletBundle</resource-bundle>
    ...
  </portlet>
  ```

  Then, provide the translations in the resource bundle, for example:

  ```
  # English Resource Bundle
  ```
Make the portlet content translatable the same way as you would for any other web application.

## 57.4.7 Portlet Deployment

Before a portlet can be registered with and consumed by an application, you must first deploy it. JSR 286 portlets are deployed to a **WSRP producer**, which is remote and communicates with the consumer application through Web Services for Remote Portlets (WSRP). JSF portlets are deployed as JSR 286 portlets, that is, to a WSRP producer, either from JDeveloper or manually.

WebCenter Portal supports WSRP versions 1.0 and 2.0. This standard provides support for interportlet communication and export or import of portlet customizations.

WSRP is a communication protocol between portlet consumer applications 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, or 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.

*Figure 57–10* illustrates the basic architecture of WSRP producers.
When end users display a page in their web browsers, the flow of the request works as follows:

1. The end user requests a page from the web browser by entering a URL in the browser’s Location field.

2. The browser transmits the request to the consumer 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 consumer application using the WSRP 1.0 or 2.0 protocol.

6. The consumer application transmits the portlet content to the browser over HTTP.

To make standards-based portlets available to a consumer application in WebCenter Portal, you must package them in a Portlet Producer application and deploy them to a WSRP container. For more information, see Chapter 61, "Deploying Portlet Producers."
This chapter describes how to use the Oracle JSF Portlet Bridge to expose a JSF application as a portlet.

This chapter includes the following topics:

- Section 58.1, "Introduction to the Oracle JSF Portlet Bridge"
- Section 58.2, "Creating a Portlet from a JSF Application"
- Section 58.3, "Updating the Portlet Entry for a Portletized Page or Task Flow"
- Section 58.4, "Testing a Portletized Page or Task Flow in JDeveloper"
- Section 58.5, "Testing a JSF Portlet Using the Integrated WebLogic Server"
- Section 58.6, "Deploying JSF Portlets to a WebLogic Managed Server"
- Section 58.7, "Using Events to Link JSF Portlets with Other Portlets"
- Section 58.8, "What You May Need to Know When Creating a JSF Portlet"
- Section 58.9, "Copying a Runtime-Created Skin to a JSF Portlet Producer Application"

Note: The Oracle JSF Portlet Bridge extends the Apache Reference implementation of 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:


58.1 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.
Creating a Portlet from a JSF Application

- 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 WebCenter Portal.
- 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 JSR 286 portlets, this also enables you to use your task flows in a distributed environment.

Note: The Oracle JSF Portlet Bridge uses WSRP extensions that may prevent JSF portlets working as intended with third party WSRP consumers.

58.2 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) 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 topics:
- Section 58.2.1, "How to Create a JSF Portlet Based on a Page"
- Section 58.2.2, "What Happens When You Create a JSF Portlet Based on a Page"
- Section 58.2.3, "How to Create a JSF Portlet Based on a Task Flow"
- Section 58.2.4, "What Happens When You Create a Portlet Based on a Task Flow"

58.2.1 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 JDeveloper, open the application that contains the JSPX page that you want to portletize.
2. In the Application Navigator, right-click the JSPX page and choose Create Portlet Entry.

Note: You cannot generated a portlet based on a page that contains a portlet.
3. In the Create Portlet Entry dialog, in the Portlet Name field, enter a name for the portlet.

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.

58.2.2 What Happens When You Create a JSF Portlet Based on a Page

When you create a JSF portlet based on a page, a portlet.xml file is created (or updated if it already exists) to contain the portlet entry (Example 58–1) for the page. The file 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 58–1  Generated Portlet Entry for a Page

```xml
<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">
</portlet>
```
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 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 /. In Example 58–1, 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>.

For information about portlet modes, see Section 57.4.2, "Portlet Modes."

### 58.2.3 How to Create a JSF Portlet Based on a Task Flow

An advantage of using Oracle ADF is the existence of task flows, which 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. Bounded task flows can use pages or page fragments, however only those that use page fragments can be portletized.

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 *Fusion Developer’s Guide for Oracle Application Development Framework*.

This section includes the following topics:

- Section 58.2.3.1, "Creating a Portlet From a Task Flow Using the Create Portlet Entry Dialog"
- Section 58.2.3.2, "Creating a Portlet From a Task Flow Using the Manage Portlet Entries of Task Flows Dialog"

### 58.2.3.1 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 JDeveloper, open the JSF application that contains the task flow from which you want to make a portlet.
2. In the Application Navigator, right-click the task flow that you want to portletize, and choose **Create Portlet Entry**.
3. In the Create Portlet Entry dialog, in the **Portlet Name** field, enter a name for the portlet.
4. If the task flow is unbounded, the **Entry Point View** drop-down list displays all the view activities of the task flow. From this drop-down 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** drop-down 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.

**58.2.3.2 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.
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.

**58.2.4 What Happens When You Create a Portlet Based on a Task Flow**

When your portlet, or portlets, have 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 58–2) and is opened ready for viewing or editing.

**Example 58–2 Generated Portlet Entry for a Task Flow**

```xml
<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>
  </init-param>
  <supports>
    <mime-type>text/html</mime-type>
    <portlet-mode>VIEW</portlet-mode>
  </supports>
</portlet>
```
58.3 Updating the Portlet Entry for a Portletized Page or Task Flow

If you edit a page or task flow that has been previously portletized, if those edits include updates to the task flow or page's parameters or events, then you must update the portlet entry for the page or task flow to keep the portlet in sync with the underlying page or task flow.

**Note:** If you delete a page that has been portletized, you must manually remove the portlet entry for that page from the `portlet.xml` file.

To update the portlet entry for a portletized page or task flow:

1. In the Application Navigator, open the JSF application that contains the portletized page or task flow that you want to update.
2. Right-click the page or task flow, and choose Update Portlet Entry.
3. If the portlet entry is for an unbounded task flow, then a dialog prompts you to select the view activity that was used to create the portlet.
4. The portlet entry is updated as follows:
   - The Create portlet events for contextual events option is set to true.
   - No updates are made to the portlet name, display name, title, short title, or description.
   - Any event-definition elements (at the portlet application level) are created as necessary.
   - None of the existing event-definition elements are deleted or updated.

---

<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. The file can also include standard JSR 286 (non bridge) portlets.

---
Testing a Portletized Page or Task Flow in JDeveloper

Any public-render-parameter elements (at the portlet application level) are created as necessary.

None of the existing public-render-parameter elements are deleted or updated.

The list of supported-processing-event, supported-publishing-event, and supported-public-render-parameter elements in the portlet entry are re-created to reflect the current events and input parameters of the page or task flow. This means that some existing portlet events and public parameters may be deleted and new ones added.

58.4 Testing a Portletized Page or Task Flow in JDeveloper

After you have portletized a page or task flow, you must test it to ensure that it is working correctly. Testing a portletized page or task flow requires the following steps:

1. Portletize the page or task flow to create a portlet entry in the portlet.xml file.
2. Deploy the portletized application to a portlet server.
3. Create an application to consume the portletized page or task flow.
4. Register the portletized application with the consumer application.
5. Create a page in the consumer application and add the portletized page or task flow to it.
6. Deploy the consumer application and display the page that contains the portletized page or task flow.

WebCenter Portal provides a menu option, Run as Portlet, within JDeveloper to simplify this process by automatically performing these steps.

Tip: You can also use the Run as Portlet menu option to test how a page or task flow behaves as a portlet before explicitly creating the portlet entry.

Before You Begin

For the Run as Portlet option to work correctly, the following conditions must be met:

- You must deploy the Portlet Bridge Tester Consumer application to the Integrated WebLogic Server.
  
  If the application is not yet deployed, from the Run menu, choose WebCenter Portal Deployments, then select Portlet Bridge Tester Consumer and click OK.

- Integrated WLS must be using the default port, 7101.

- When you start Integrated WLS, it must listen to the local IP address, 127.0.0.1 (you specify this in the Configure WebLogic Domain dialog by explicitly entering the IP address in the Listen Address field, or leaving the field blank).

- There must be no other application deployed to Integrated WLS using the context root portletBridgeTester-Portlets-context-root.

- The application containing the page or task flow must be able to be successfully deployed to Integrated WLS.

If any of these conditions cannot be met, you must manually test the portletized page or task flow using the steps provided in Section 58.5, "Testing a JSF Portlet Using the Integrated WebLogic Server."
To test a portletized page or task flow:

1. In the Application Navigator, open the JSF application that contains the portletized page or task flow that you want to test.

2. Right-click the page or task flow, and choose Run as Portlet.

3. If you selected to run an unbounded task flow as a portlet, then a dialog prompts you to select the view activity to use as the entry point for the portlet.

4. The following happens:
   a. If a portlet entry already exists for the page or task flow, then the portlet entry is updated to ensure that it reflects the current functionality of the page or task flow. For more information, see Section 58.3, "Updating the Portlet Entry for a Portletized Page or Task Flow."

      If a portlet entry does not already exist for the page or task flow, then a portlet entry is created using the default values.

   b. If the selected task flow is an unbounded task flow, a dialog prompts you to select the view activity to use as the entry point for the portlet.

   c. If the selected task flow is a bounded task flow with input parameters, a dialog enables you to enter default values for those parameters.

   d. The application is deployed to the Integrated WebLogic Server, using the context root portletBridgeTester-Portlets-context-root.

   e. When deployment is complete, the portletized page or task flow is displayed in a browser window.

5. Test the functionality of the portletized page or task flow within the test consumer application.

58.5 Testing 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.

Before You Begin

Before you test your portletized page or task flow, it is a good idea to first test that the page or task flow works correctly in the running ADF application. You can do this, using the Integrated WebLogic Server, by running the page or, for task flows, adding the task flow to a page and running that page.

For more information, see the "Running an ADF Application in Integrated WebLogic Server" section in the Fusion Developer’s Guide for Oracle Application Development Framework.

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 2.2.4, "Managing the Integrated WebLogic Server."

2. You are now ready to run your Portlet Producer application on 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 your web application as you normally would (see the "Running an ADF Application in Integrated WebLogic Server" section in the Fusion Developer’s Guide for Oracle Application Development Framework) or you can run your portlet following the instructions in Section 59.4.1, "How to Run a WSRP Portlet Producer on Integrated WebLogic Server."

3. When the application is deployed, you can view the WSRP Producer Test Page by going to:

http://host:port/context-root/info

where:
- **host** is the server to which the application 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.

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 63.2.1.1, "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 63.5, "Adding Portlets to a Page."

### 58.6 Deploying JSF Portlets to a WebLogic Managed Server

When you are satisfied that the application works correctly both as a web application and a portlet application, you can deploy it to your production Oracle WebLogic Managed Server.

Because your web application and portlet application are one and the same (the portlet application is your existing web application with additional portlet artifacts), deploying the web application also deploys the portlet producer application.

For information about how to deploy an application to a WebLogic Managed Server, see the “Deploying Fusion Web Applications” chapter in the Fusion Developer’s Guide for Oracle Application Development Framework.
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

58.7 Using Events to Link JSF Portlets with Other Portlets

If you portletize an Oracle ADF task flow that triggers a contextual event, the Oracle JSF Portlet Bridge creates a JSR 286 portlet event to wrap the contextual event. This means that when the portletized task flow is consumed on a page, it can be linked to other JSF portlets or to JSR 286 portlets.

The examples in this section use the following portlets:

- **Departments portlet**, a portletized ADF task flow that displays a list of departments. The list of departments can be restricted by location ID. This portlet raises a contextual event, `departmentSelected`, when a department row is selected in the portlet. The payload of the departmentSelected event is an object of type `hr.Department` (a Java object representing one row of the Departments table).

- **Employees portlet**, a portletized ADF task flow that displays a list of employees. The list of employees can be restricted by department ID.

- **Department Address portlet**, a JSR 286 portlet that displays the address of a specified department.

- **Department Locations portlet**, a JSR 286 portlet that displays a list of locations. This portlet raises a portlet event, named `locationId` with the namespace `http://xmlns.oracle.com/adfm/contextualevent`, when one of the listed locations is selected.

This section includes the following topics:

- **Section 58.7.1, "How to Link a JSF Portlet to Another JSF Portlet"
- **Section 58.7.2, "How to Link a JSF Portlet to a JSR 286 Portlet"
- **Section 58.7.3, "How to Link a JSR 286 Portlet to a JSF Portlet"
- **Section 58.7.4, "What You May Need to Know About Portlet Events in the Oracle JSF Portlet Bridge"

58.7.1 How to Link a JSF Portlet to Another JSF Portlet

The Departments task flow raises a contextual event, `departmentSelected`, when a department row is selected within the task flow. The event is ultimately raised by an `eventBinding` which is triggered from a row selection listener (Example 58–3).

The payload of the event (an object of type `hr.Department`, a Java object representing one row of the departments table) is generated by using the `customPayload` attribute which points to the current row from the `departments` table iterator.
For more information about standard ADF contextual event techniques, like the ones used in this example, see the "Creating Contextual Events" section in the *Fusion Developer’s Guide for Oracle Application Development Framework*.

**Example 58–3  Contextual Event for the Departments Task Flow**

```xml
<eventBinding Listener="org.apache.myfaces.trinidad.event.SelectionListener"
 id="DepartmentSelectedEvent">
 <events xmlns="http://xmlns.oracle.com/adfm/contextualEvent">
   <event name="departmentSelected"
     customPayLoad="#{bindings.departments.currentRow.dataProvider}"/>
 </events>
</eventBinding>
```

When the Departments task flow is portletized, using the steps described in Section 58.2.3, "How to Create a JSF Portlet Based on a Task Flow," a `portlet.xml` file is created containing a definition for a portlet event that is used to transfer the corresponding contextual event during portlet communication with the portlet over WSRP (Example 58–4).

**Example 58–4  Portlet Entry for the Portletized Departments Task Flow**

```xml
<portlet id="adf_taskflow_WEB_INF_departments_xml">
  ...
  <supported-publishing-event id="departmentSelected">
    <qname xmlns:x="http://xmlns.oracle.com/adfm/contextualEvent">
      x:departmentSelected
    </qname>
  </supported-publishing-event>
  ...
</portlet>
```

There is also a corresponding event definition in `portlet.xml` for the Departments portlet's `departmentSelected` event (Example 58–5).

**Example 58–5  Event Definition for the departmentSelected Event**

```xml
<event-definition id="departmentSelected">
  <qname xmlns:x="http://xmlns.oracle.com/adfm/contextualEvent">
    x:departmentSelected
  </qname>
  <value-type>
    oracle.portlet.bridge.adf.lifecycle.ADFmPayloadWrapper
  </value-type>
</event-definition>
```
The Employees task flow uses standard ADF techniques to handle a departmentSelected event. This is in the form of a methodAction binding that invokes a method on a data control, delivering the event payload from the departmentSelected event as a parameter to that method. Example 58–6 shows the definition of the methodAction binding.

**Example 58–6  Definition of methodAction for the Employees Task Flow**

```xml
<methodAction id="updateTableForDepartmentId" RequiresUpdateModel="true"
  Action="invokeMethod" MethodName="updateTableForDepartmentId"
  IsViewObjectMethod="false" DataControl="EmployeesDataControl"
  InstanceName="EmployeesDataControl.dataProvider">
  <NamedData NDName="departmentIdPayLoad" NDType="hr.Department"/>
</methodAction>
```

The updateTableForDepartmentId method takes the hr.Department object, passed as the payload of the departmentSelected event, and filters the data queried from the EmployeesDataControl so that it shows only employees who are in the selected department. It also takes steps to ensure that the iterator is re-executed and that the table view component is marked as needing to be re-rendered in this request.

The Employees task flow includes an event map to specifically map the incoming event onto the appropriate methodAction when the task flow is portletized (Example 58–7).

**Example 58–7  Event Map for Portletized Employees Task Flow**

```xml
<eventMap xmlns="http://xmlns.oracle.com/adfm/contextualEvent" >
  <event name="departmentSelected">
    <producer region="*">*
      <consumer region="*" handler="updateTableForDepartmentId"
        handleCondition="${ADFPortletBridge.bridgeRequest}">
        <parameters>
          <parameter name="p1" value="#{payLoad}"/>
        </parameters>
      </consumer>
    </producer>
  </event>
</eventMap>
```
Using Events to Link JSF Portlets with Other Portlets

**Note:** The `<producer region="*">` attribute accepts events from any region. This is because when the task flow is running as a portlet, with the event coming into the portlet from a remote application, the remote application does not have any knowledge of the producer region.

The EL expression within `handleCondition="${ADFPortletBridge.bridgeRequest}"` evaluates to true if the current request is running under the Oracle JSF Portlet Bridge. This ensures that this event map is used only when the task flow is running as a portlet. It is not essential to have this `handleCondition`. However it can be useful to be able to define a separate `eventMap` which is used only when the task flow is running as a portlet.

When the two portletized task flows are added to a page, as long as the name of the event triggered by the Departments portlet is the same as that expected by the Employees portlet, no further coding is required; the portlets are automatically linked and when a department is selected in the Departments portlet, the Employees portlet is updated to display employees belong to the selected department.

![Figure 58–1 Linking JSF Portlets](image)

You may choose to not use automatic event listening, for example, if you have multiple portlets on the same page that use the same events and you wish to control the flow of events between them. Or perhaps the names of the contextual events used by the JSF portlets that you want to link do not match. In this case you must explicitly create an event map in the page definition of the page containing the portletized task flows. This event map links the portlet events of the portlets that you want to wire together. Example 58–8 shows how this would work in our example if the Employees task flow was expecting an event named `departmentId` instead of `departmentSelected`. 
58.7.2 How to Link a JSF Portlet to a JSR 286 Portlet

As well as linking JSF portlets to other JSF portlets, you can also link a JSF portlet to a regular JSR 286 portlet. To do this, you must ensure that the JSR 286 portlet explicitly declares that it can receive contextual events.

For example, we can link the Departments task flow from our previous example to a Department Address portlet, a JSR 286 portlet, to display the address of a selected department.

For automatic event wiring to work, the Department Address portlet must declare that it can receive the contextual event from the Department portlet. It can do this in two ways:

- The Department Address portlet directly supports the `departmentSelected` contextual event:

```
<supported-processing-event id="departmentSelected">
  <qname xmlns:x="http://xmlns.oracle.com/adfm/contextualEvent">
    x:departmentSelected
  </qname>
</supported-processing-event>
```

- The event definition for the Department Address portlet’s event, `department`, provides an alias to associate it with the `departmentSelected` contextual event:

```
<event-definition id="department">
  <qname xmlns:x="http://xmlns.oracle.com/portlet/EventSample">
    x:department
  </qname>
  <alias xmlns="http://xmlns.oracle.com/adfm/contextualEvent">
    x:departmentSelected
  </alias>
  <value-type>hr.Department</value-type>
</event-definition>
```
The Department Address portlet includes code in its `processEvent` implementation to handle the event (Example 58–9). This method looks for either the `departmentSelected` or the `department` event with the alias as defined above. The payload type for both of these events is `hr.Department`. However, the method shown in Example 58–9 illustrates how to use the `ADFmPayloadWrapper` to unwrap the payload. `ADFmPayloadWrapper` is the event type that the Oracle JSF Portlet Bridge uses by default when raising events.

Example 58–9  Code to Handle the `departmentSelected` Event

```java
public void processEvent(EventRequest request, EventResponse response) throws PortletException, IOException {
    response.setRenderParameters(request);
    Event event = request.getEvent();
    String eventName = event.getName();
    if ("departmentSelected".equals(eventName) || "department".equals(eventName)) {
        Object payload = event.getValue();
        if (payload instanceof ADFmPayloadWrapper) {
            payload = ((ADFmPayloadWrapper)payload).getPayload();
        }
        if (payload instanceof Department) {
            Department department = (Department)payload;
            int locationId = department.getLocation();
            response.setRenderParameter("locationId", Integer.toString(locationId));
        }
    }
}
```

When these two portlets are displayed together on a page, selecting a department in the Departments portlet automatically updates the Department Address portlet to display the address of the selected department.

**Note:** The namespace for portlet events used to carry contextual event is always:

http://xmlns.oracle.com/adfm/contextualEvent

The namespace is used in the event declarations or alias given above. This namespace indicates to the consumer that the portlet wishes to receive a contextual event with the given name as a portlet event. This is essential for automatic event wiring to work.
Using Events to Link JSF Portlets with Other Portlets

Creating Portlets from JSF Applications Using the Oracle JSF Portlet Bridge

58.7.3 How to Link a JSR 286 Portlet to a JSF Portlet

You can also send portlet events from a JSR 286 portlet to a JSF portlet. The portlet event is converted into a contextual event by the Oracle JSF Portlet Bridge and is then delivered into the producer ADF application to be consumed using standard contextual event techniques.
The Departments task flow contains a methodAction binding that binds to a method on the DepartmentsDataControl (Example 58–11).

**Example 58–11  Definition of methodAction for the Departments Task Flow**

```xml
<methodAction id="updateTableForLocationId" RequiresUpdateModel="true"
    Action="invokeMethod" MethodName="updateTableForLocationId"
    IsViewObjectMethod="false"
    DataControl="DepartmentsDataControl"
    InstanceName="DepartmentsDataControl.dataProvider">
    <NamedData NDName="locationId" NDType="int"/></methodAction>
```

The updateTableForLocationId method receives, as a parameter, a location ID that causes the list of departments returned by the data control to be filtered by that location ID.

The task flow contains an event map to map the contextual event onto the appropriate methodAction (Example 58–12).

**Example 58–12  Event Map for Portletized Departments Task Flow**

```xml
<eventMap xmlns="http://xmlns.oracle.com/adfm/contextualEvent">
    <event name="locationId">
        <producer region="*">
            <consumer region="" handler="updateTableForLocationId" handleCondition="${ADFPortletBridge.bridgeRequest}">
                <parameters>
                    <parameter name="p1" value="#{payLoad}"/>
                </parameters>
            </consumer>
        </producer>
    </event>
</eventMap>
```

*Note:* The event map cannot reference the region where the event originates because it is not present when the task flow is running as a portlet, so we use a wildcard as the event source in the event map.

In the Department Locations portlet, when a location is selected, the processAction method raises a portlet event, locationId, giving the location ID of the selected location (Example 58–13).

**Example 58–13**

```java
public void processAction(ActionRequest request, ActionResponse response)
    throws PortletException
{
    String locationId = request.getParameter('locationId');
    Location location = getLocation(Integer.parseInt(locationId));

    if (location != null)
    {
        // QName matches the event declared as a supported-publishing-event in
        // portlet.xml for this portlet.

        // LocationId event. Raised in the ContextualEvent namespace makes it readily
        // consumable by ADF Bridge portlets. Likewise we wrap with the
        // ADFmPayloadWrapper payload object that the ADF Bridge expects.
```
response.setEvent(new QName("http://xmlns.oracle.com/adfm/contextualEvent", "locationId"),
        new ADFmPayloadWrapper(Integer.valueOf(location.getLocationId())));
}

---

**Note:** The `locationId` event raised in the `processAction` method has a payload of type of `Integer`. However, because JSF portlets always expect portlet events with the contextual event payloads wrapped with `ADFmPayloadWrapper`, we must wrap the payload to suit the receiving portlet's requirements if we want automatic delivery of events to work.

Likewise, the QName for the event we raise must be:

```
(http://xmlns.oracle.com/adfm/contextualEvent)locationId
```

When you drop the portlets on a page, selecting a location in the Department Location portlet automatically updates the Departments portlet to display the departments at the selected location.

**Figure 58–3  Linking a JSR 286 Portlet to a JSF Portlet**

If the two portlets do not declare events with matching names (for example the contextual event from the Departments portlet is called `locId`), or if automatic event listening is disabled, you can still link the two portlets by creating an event map in the page definition to map the contextual event from the Departments portlet to the portlet event from the Department Location portlet (**Example 58–14**).

**Example 58–14**

```xml
<eventMap xmlns="http://xmlns.oracle.com/adfm/contextualEvent">
  <event name="locationId">
    <producer region="DepartmentLocations1_1">
      <consumer handler="DepartmentsADFBridgePortlet1_1.
(http://xmlns.oracle.com/adfm/contextualEvent)locId"/>
    </producer>
  </event>
</eventMap>
```
58.7.4 What You May Need to Know About Portlet Events in the Oracle JSF Portlet Bridge

Using a Serialized Type for the Contextual Event Payload

Part of the JSR 286 event's payload is the wrapped contextual event payload, which is serialized. Therefore, while contextual event payloads can be any object type, when you are sending contextual events via WSRP, only serialized types are supported.

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.

Raising Undeclared Events

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:

```xml
<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. For information about setting container runtime options, see Section 59.3.4.3, "Setting Container Runtime Options for Individual Portlets."

You can also enable undeclared portlet events to automatically be forwarded on from the portlet binding in the consumer application as ADFm events.

In the portlet binding, set the raiseUndeclaredContextualEvent attribute to true, for example:

```xml
<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.

**Using Partial Page Rendering**

The Oracle JSF Portlet Bridge uses portlet events to transport contextual events to and from the consumer application. However, partial page rendering (PPR) requests are implemented using Portlet 2.0 resource requests. It is not possible to send or receive portlet events in resource requests. Therefore proprietary mechanisms are used to transport the contextual events from the JSF Portlet to the consumer application in the case of PPR requests.

### 58.8 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, which naturally render markup in a style that is either compatible with JSF portlets or can be automatically interpreted by the Oracle JSF Portlet Bridge into a compatible style (for example, by rendering the portlet in an inline frame).

The sections that follow provide some guidance on how to avoid common issues that you may encounter as you code Oracle ADF pages that you later choose to publish as portlets.

This section includes the following topics:

- Section 58.8.1, "General Guidelines"
- Section 58.8.2, "Portlet Guidelines"
- Section 58.8.3, "Security Guidelines"
- Section 58.8.4, "JSF Guidelines"
- Section 58.8.5, "Oracle ADF Guidelines"

#### 58.8.1 General Guidelines

This section includes the following topics:

- Section 58.8.1.1, "Application Must Run as a Web Application"
- Section 58.8.1.2, "The Portlet Is Contained by a Different Page"
- Section 58.8.1.3, "Servlet Dependencies"
- Section 58.8.1.4, "Secured Applications"
- Section 58.8.1.5, "Deep Links"
- Section 58.8.1.6, "Java EE Authentication"
- Section 58.8.1.7, "Timeout Period"

#### 58.8.1.1 Application Must Run as a Web Application

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.
58.8.1.2 The Portlet Is Contained by a Different Page

One area that you should consider when creating task flows that will be exposed as portlets is that the page on which you render the task flow as a portlet is different to the one in the original producer application.

Some common errors that occur include:

- Expecting to access JavaScript in the consuming page.
- Expecting to find managed beans or objects added to scopes as part of the consuming page.
- Expecting components in the JSF component tree from the consumer to be present.

58.8.1.3 Servlet Dependencies

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:

```java
HttpServletRequest request = getRequestFromFacesContext();
String localContextPath = request.getContextPath();
```

Use the following:

```java
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:

```java
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;
    }
}
```

58.8.1.4 Secured Applications

If your application is secured, ensure that you have secured identity propagation as described in Section 74.17, "Securing Identity Propagation Through WSRP Producers with WS-Security."

58.8.1.5 Deep Links

For normal interactions with a portlet, there is a session maintained between the consumer and the producer. This is done by storing a session cookie in the consumer and sending it to the producer on each request. On the original request, the session is established and on subsequent requests, the portlet sends the session cookie to the producer and the session is reused.
For deep links, that is, links from the portlet to the producer application, the request is from the browser direct to the producer application. In this case, the session cookie cannot be sent so a new session is established. If a shared session is required for deep links, then this must be implemented by the developer. A common implementation is to write this state to a common location, accessible from the producer and the application, and pass a reference to this state in the deep link.

58.8.1.6 Java EE Authentication
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.

58.8.1.7 Timeout Period
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.

58.8.2 Portlet Guidelines
This section includes the following topics:

- Section 58.8.2.1, "Portlet Sizing"
- Section 58.8.2.2, "Resources and Links"
- Section 58.8.2.3, "Redirecting Requests"
- Section 58.8.2.4, "Memory Consumption"
- Section 58.8.2.5, "WSRP Version"

58.8.2.1 Portlet Sizing
When consuming a JSF portlet, the consumer application automatically renders the portlet content within an inline frame. This means that any inline popup is then confined within the inline frame. You should take this into consideration when specifying the size of the portlet.

If the JSF portlet is a portletized page, then you have full control of the content and can decide how that content fills the inline frame window, utilizing exactly the same techniques as you would to determine how the content would fill the browser window.

If the portlet is being stretched by its ancestor in the consumer application, then the maximized attribute is set to true on the af:document component in the document produced by the Oracle JSF Portlet Bridge. This means that components that support it, such as panelStretchLayout, will fill the entire inline frame.

For more information about portlet sizing, see Section 63.5.5, "What You May Need to Know About Portlet Sizing."
58.8.2.2 Resources and Links
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.

58.8.2.3 Redirecting Requests
Do not redirect or forward a request within your JSP. JSR 286 only supports requestDispatcher.include(). The use of servletResponse.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.

58.8.2.4 Memory Consumption
The Oracle JSF Portlet Bridge stores the content of the request scope between requests so that the request scope is preserved between a portlet Action and a portlet Render (which are two separate requests). Because of this, to minimize overall memory consumption, you must especially avoid storing too much data in the request scope when the application contains JSF portlets.

58.8.2.5 WSRP Version
The Oracle JSF Portlet Bridge makes use of Portlet 2.0 features. Consequently, it should be used only with WSRP V2. The portlet's minimumWsrpVersion container runtime option should be set to 2. For more information, see Section 59.3.4, "How to Customize the Runtime Environment for JSR 286 Portlets."

58.8.3 Security Guidelines
This section includes the following topics:
- Section 58.8.3.1, "Oracle ADF Secured Applications"
- Section 58.8.3.2, "Role Based Authorization"

58.8.3.1 Oracle ADF Secured Applications
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:

  ```xml
  <permission>
  <class>oracle.adf.controller.security.TaskFlowPermission</class>
  ```
58.8.3.2 Role Based Authorization

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 if the propagated user does not have permission to view the task flow.

For information about setting up WS-Security on the producer, see Section 74.17, "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 63.2.1.1, "How to Register a WSRP Portlet Producer."

58.8.4 JSF Guidelines

This section includes the following topics:

- Section 58.8.4.1, "Using h:commandLink"

58.8.4.1 Using h:commandLink

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.

```xml
<context-param>
    <param-name>com.sun.faces.externalizeJavaScript</param-name>
    <param-value>false</param-value>
</context-param>
```

58.8.5 Oracle ADF Guidelines

This section includes the following topics:

- Section 58.8.5.1, "Task Flow Returns"
- Section 58.8.5.2, "Mismatched Style Sheets"
- Section 58.8.5.3, "ADF Faces Dialog Framework"
- Section 58.8.5.4, "Composer Components"
- Section 58.8.5.5, "The fileDownloadActionListener Component"
What You May Need to Know When Creating a JSF Portlet

- Section 58.8.5.6, "The prepareModel Phase"
- Section 58.8.5.7, "Accessing Request Parameters"
- Section 58.8.5.8, "Referencing Page Parameters"
- Section 58.8.5.9, "Trinidad Components"
- Section 58.8.5.10, "Using ADF Faces Client-Side APIs to Find Components"
- Section 58.8.5.11, "Encoded URLs"
- Section 58.8.5.12, "Issues with goLink"
- Section 58.8.5.13, "In-Protocol Resource Requests"

58.8.5.1 Task Flow Returns

When an ADF bounded task flow completes it may use a task flow return to send control back to the calling task flow. In a portletized task flow, there is no calling task flow, so the task flow return does not make any sense in this context. This may cause what appears to be an empty portlet to display as the task flow navigates to a state where there is no view selected. The task flow will stay in this state until it is refreshed either when the input parameters change or the session on the producer is reset.

If you plan to portletize your task flow, you should avoid using task flow returns. If you must use a task flow return, you can provide a dummy task flow with a default activity of the task flow that you want to portletize. When the task flow return is executed, it returns to the dummy task flow. The dummy task flow then executes its default activity, which is to call back to the original task flow and thus restart the task flow when the portlet is next called.

58.8.5.2 Mismatched Style Sheets

In general, when you consume a task flow using the Oracle JSF Portlet Bridge, the portlet producer tries to use the skin of the consumer page. When the producer cannot match the consumer skin, the portlet generates and references its own style sheet. This is known as a mismatched skin. A mismatched skin can occur when:

- The consumer and producer use different versions of ADF Faces. In this case, the producer still uses the consumer skin.
- The consumer has additional skin-additions that the producer does not have, or vice versa. In this case, the producer still uses the consumer skin.
- The producer does not have the consumer skin. In this case, the producer uses the portlet skin.

Mismatched skins can cause performance problems because the browser has to fetch an extra style sheet and the generated portlet markup uses uncompressed styles resulting in larger markup.

If a mismatched skin occurs, the producer log includes the following:

The skin $skinName$ specified on the requestMap will not be used because the styleSheetDocument id on the requestMap does not match the local skin’s styleSheetDocument’s id. It could mean the jars are not identical. For example, one might have trinidad-skins.xml skin-additions in a jar file on the class path that the other does not have.

The portlet markup inside the inline frame also includes a link tag to the portlet style sheet resource, for example:

```html
<link rel="stylesheet" charset="UTF-8" type="text/css"/>
```
You can use an HTTP monitoring tool to see that a request is made to the portlet style sheet resource.

There are a number of reasons for mismatched skins. For a skin to match, the producer and consumer must be the same for all the following conditions:

- The ADF Faces version; a different version of ADF Faces may have different style selectors.
- The style compression, defined in `web.xml`, for example:
  ```xml
  <context-param>
      <param-name>org.apache.myfaces.trinidad.DISABLE_CONTENT_COMPRESSION</param-name>
      <param-value>false</param-value>
  </context-param>
  ```
- Tonal style or themes, defined in `web.xml`, for example:
  ```xml
  <context-param>
      <param-name>xyz</param-name>
      <param-value>xyz</param-value>
  </context-param>
  ```
- Availability of skin additions. Skin additions are defined in `META-INF/trinidad-skins.xml` in a JAR file. These are then aggregated from all the JAR files in the class path. If there are any JAR files that exist on the producer but not on the consumer, or vice versa, you get a mismatch.

To determine if the JAR files match, turn on Trinidad logging to show which skin-addition it is processing. To do this, update the `logging.xml log level of both the producer and consumer WebLogic Servers to FINEST` then restart the servers:

```xml
<logger name="org.apache.myfaces.trinidad.skin.SkinUtils" level=FINEST"/>
```

When you run the consumer page, any skin entries that do not match in the consumer and producer log files are the cause of the skin mismatch.

### 58.8.5.3 ADF Faces Dialog Framework

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>`

58.8.5.4 Composer Components
Portletization of pages that contain Composer components is not supported.

58.8.5.5 The fileDownloadActionListener Component
The `<af.fileDownloadActionListener>` component is not supported.

58.8.5.6 The prepareModel Phase
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.

58.8.5.7 Accessing Request Parameters
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 lifecycle 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:

```xml
<invokeAction id="doExecuteWithParams"
  Binds="ExecuteWithParams"
  Refresh="prepareModel"
  RefreshCondition="${param.id != null}"
/>
```

```xml
<invokeAction id="doExecuteWithParams"
  Binds="ExecuteWithParams"
  Refresh="renderModel"
  RefreshCondition="${param.id != null}"
/>
```

58.8.5.8 Referencing Page Parameters
Do not reference page parameters in the `prepareModel` phase. This issue relates to the same problem described for request parameters in Section 58.8.5.7, "Accessing 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 fails. Instead, move any dependency on a page parameter value into the model before JSF transitions from its execute phases to its render phase.

58.8.5.9 Trinidad Components
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.

58.8.5.10 Using ADF Faces Client-Side APIs to Find Components
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 the "What You May Need to Know About Using Naming Containers" section in the Web User Interface Developer’s Guide for Oracle Application Development Framework.

58.8.5.11 Encoded URLs
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.

Another example is if your task flow includes an inline frame (for example, using an af:inlineFrame tag). In this case you need to ensure that the inline frame URL does not get encoded to go via the portlet resource proxy in the consumer. If you do not do this, any relative URLs in the content inside the inline frame will not work because they will be relative to the URL to the resource proxy rather than their correct location relative to the original URL for the inline frame.

To bypass the URL encoding, set the _xEncodeUrl parameter to false on URLs that are passed to the Oracle JSF Portlet Bridge for encoding. This parameter is picked up by the Oracle JSF Portlet Bridge, which ensures that the URL does not get encoded via the resource proxy. The parameter is ignored if the task flow is not running as a portlet.

For example, replace:

```xml
<af:inlineFrame source="http://myiframe.example.com">
```

with:

```xml
<af:inlineFrame source="http://myiframe.example.com?_xEncodeUrl=false">
```

---

**58.8.5.12 Issues with goLink**

The adf:goLink component encodes the link URL as an action for the current JSF application. In other words, it encodes the link using `ExternalContext.encodeActionURL(...)`. This is different to the JSF h:outputLink component, which encodes the URL as a resource URL using `ExternalContext.encodeResourceURL(...)`. If you are using goLink to link to a resource that is not a Faces View, then the URL must be encoded as a Resource URL. To do this, you must tell the Oracle JSF Portlet Bridge the link is not a Faces View by adding the following parameter to the URL:

```
javax.portlet.faces.ViewLink=false
```

For example:

```xml
<af:goLink text="goLink 1" id="gl1" destination="/mynonfaceslink?javax.portlet.faces.ViewLink=false"/>
```

---

**58.8.5.13 In-Protocol Resource Requests**

Within a JSF application running under the Oracle JSF Portlet Bridge, when a resource URL is created using `ExternalContext.encodeResourceURL(...)`, that resource URL is encoded in such a way that it is executed as an out-of-protocol resource request. This means that when the portlet consumer accesses that resource it does so by making a direct call to the target URL. It is also possible for resource URLs to be in-protocol. This means that the portlet consumer accesses the resource by making a WSRP web service call to the producer application, which in turn ultimately leads to the serverResource method being called on the portlet. It is then the portlet’s responsibility to provide the resource which is sent back in the WSRP response. When that portlet is the Oracle JSF Portlet Bridge, it serves the resource by using a request dispatcher to forward to the target URL.
Generally out-of-protocol resource requests are to be preferred as they do not incur the additional overhead of the web service call. In-protocol requests can also cause large amounts of memory to be allocated as the WSRP SOAP response is constructed, if the resource being served is large. In the case of the Oracle JSF Portlet Bridge, the only time that you would want to use in-protocol resources is when you need to share the portlet scoped HTTP session between the resource requests and the other portlet requests, such as portlet action or render requests.

To cause the Oracle JSF Portlet Bridge to encode resource URLs as in-protocol resource requests, add the following parameter to the URL, before `ExternalContext.encodeResourceURL(...)` is called:

```java
javax.portlet.faces.InProtocolResourceLink=true
```

For example:

```xml
<af:goLink text="goLink 1" id="gl1" destination="/mynonfaceslink?javax.portlet.faces.ViewLink=false&javax.portlet.faces.InProtocolResourceLink=true"/>
```

## 58.9 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 Portal Framework application to an EAR file.
   
   For information about how to do this, see Section 9.6.1, "How to Download a Portal Resource Using the Assets Page."

2. Repackage the exported skin as a shared library (a JAR file):
   
   a. Extract the transport.mar file:

   ```bash
   $ jar xvf myskin.ear
   inflated: transport.mar
   ```

   b. Extract the metadata files:

   ```bash
   $ 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:

   ```xml
   <resourceType name="skin" ...>
   ```
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
   <?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
   
   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.
This chapter describes how to build standards-based Java portlets using the Java Portlet Specification, JSR 286.

This chapter includes the following topics:

- Section 59.1, "Introduction to Standards-Based Java Portlets"
- Section 59.2, "Creating a JSR 286 Java Portlet"
- Section 59.3, "Developing JSR 286 Java Portlets"
- Section 59.4, "Testing JSR 286 Portlets"
- Section 59.5, "Deploying JSR 286 Portlets to a WebLogic Managed Server"
- Section 59.6, "Migrating WebLogic Portal Portlets to WebCenter Portal"
- Section 59.7, "Troubleshooting JSR 286 Java Portlets"

59.1 Introduction to Standards-Based Java Portlets

WebCenter Portal supports standards-based Java portlets built using the JSR 286 standard.

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 that provide:

- 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
- Interportlet communication using portlet events and public render parameters

For more information, see the JSR 286 specification at:


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:

- Web Services Definition Language (WSDL) interface for the invocation of WSRP services
59.2 Creating a JSR 286 Java Portlet

WebCenter Portal provides a wizard, available in JDeveloper, for quickly and easily creating the initial framework of your JSR 286 standards-based Java portlets.

This section includes the following topics:

- Section 59.2.1, "How to Create a JSR 286 Java Portlet"
- Section 59.2.2, "What Happens When You Create a JSR 286 Portlet Using the JDeveloper Wizard"
59.2.1 How to Create a JSR 286 Java Portlet

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 JDeveloper, open the portlet producer application under which you want to create your portlet, or create a new portlet producer application.

   For information about how to create a portlet producer application, see Section 57.4.1, "Portlet Producer Applications."

   **Note:** If you do not use the WebCenter Portal - Portlet Producer Application template to create the portlet producer application, you must manually add the appropriate portlet building technology scopes to the application.

   Applications built using the WebCenter Portal - 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 (Figure 59–2), expand Web Tier, select Portlets and then Standards-based Java Portlet (JSR 286), and click OK.

*Figure 59–2 The New Gallery with Standards-based Java Portlet (JSR 286) Selected*
4. On the General Portlet Information page of the Create JSR 286 Java Portlet wizard, replace the default name provided in the Name field with one that better describes the purpose of the portlet.

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 drop-down list, select the package in which to create the class, or enter a package name.

   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 drop-down 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 57.4.2.2, "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, 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.

    **Tip:** The Display Name, Short Title, Description, and Keyword attributes are not implemented in WebCenter Portal. 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.

11. 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.

12. On the Content Types and Portlet Modes page, in the Content Types and Portlet Modes list, select view.

13. In the Implementation Method section, select how you want to implement View mode for your portlet (for more information about View mode, see...
Section 57.4.2.1, "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.

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 58, "Creating Portlets from JSF Applications Using 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 doMODE_NAMECONTENT_TYPE) in the generated portlet java class. You must update this code to render useful content.

**Tip:** If you want this portlet mode to support a different content type, for example text/xml, see Step 16.

14. 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 57.4.2.2, "Edit Mode"), as described in step 13.

15. 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 mode or 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 57.4.2, "Portlet Modes."

   d. Select each of the portlet modes and specify the implementation method to use for rendering content, as described in step 13.
The content type identifies what type of content the portlet supports. Portlets can support multiple content types. By default, portlets created using the Create JSR 286 Java Portlet wizard support the `text/html` content type.

To add a different 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.

- `text/html`—The portlet supports text encoded with HTML. This is the default selection.
- `text/xml`—The portlet supports text encoded with XML.
- `text/plain`—The portlet supports plain, unencoded text.
- `text/vnd.oracle.mobilexml`—The portlet supports text encoded with Oracle Mobile XML. Note, however, that WebCenter Portal does not support Oracle Mobile XML.
- `application/xhtml+xml`—The portlet supports text encoded with XHTML.
- `application/xml`—The portlet supports any XML content, including XHTML.
17. 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 portlet preferences to enable users of the portlet to specify values for the portlet at runtime. Go to step 18.

If you did not select this option, go to step 24.

18. On the Customization Preferences page, click **Add** to add a new portlet preference to the portlet.

By default, the wizard includes a portlet preference to enable users to customize the portlet title.

19. 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 and cannot contain spaces.

20. In the **Default Values** field, enter one or more default values for the new preference. Separate multiple values with commas.

21. Select the **Translate Preference** check box if you want to be able to make the preference available in different languages.

   For example, if the portlet is likely to be consumed by a multilingual portal you want to ensure that the preference displays in the appropriate language.

   If you enable this option, then entries for the preference are added to the resource bundle class that JDeveloper generates for the portlet.

   **Tip:** Edit the resource bundle to provide translations for the preference name and default values. The name will almost always require translating, but the default values may not, for example, if the value is an integer.
22. Click OK.

23. Repeat the preceding steps to add more preferences. When you are done click Next.

24. On the Security Roles page, 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.

25. Click Next.

26. On the Caching Options page, select Cache Portlet to enable expiry-based caching for your portlet.

For more information about expiry-based caching, see Section 57.4.5, "Portlet Performance" and Section 59.3.10.1, "Implementing Expiry-Based Caching in JSR 286 Portlets."

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 information about how to implement validation-based caching, see Section 59.3.10.2, "Implementing Validation-Based Caching in JSR 286 Portlets."

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, 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.

For example, you might want to turn on some kind of debugging mode for the portlet or display different menu options in different environments.

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 to create the portlet.

59.2.2 What Happens When You Create a JSR 286 Portlet Using the JDeveloper Wizard

When you use the Create JSR 286 Java Portlet wizard, JDeveloper generates a default implementation of the portlet. Specifically, the following files are created:

- **Java classes:**
  - `portletName.java` is invoked by the portlet container and contains all the methods required by the portlet standards.
  - `portletNameBundle.java` contains all the translation strings for the portlet.
  - `portletNameBacking.java` contains the JSF backing bean for the JSPX pages that implement portlet modes. This class is created if you implement portlet modes as ADF-Faces JSPX pages.

- **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.

- **faces-config.xml** is a configuration file that registers an application's resources, such as custom validators and managed beans and describes the page flow of the application. This file is created if you implement portlet modes as ADF-Faces JSPX pages.

- **trinidad-config.xml** is a configuration file that contains information about the application skin family. This file is created if you implement portlet modes as ADF-Faces JSPX pages.

You can see all these files in the Application Navigator, as shown in Figure 59–5.
59.3 Developing JSR 286 Java Portlets

When you have built your initial portlet implementation using the Create JSR 286 Java Portlet wizard, the next step is to create the code that drives the portlet content and behavior. 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, including:


This section includes the following topics:

- Section 59.3.1, "How to Edit the Portlet Deployment Descriptor File"
- Section 59.3.2, "How to Add Custom Portlet Modes to JSR 286 Portlets"
- Section 59.3.3, "How to Access User Information in JSR 286 Portlets"
- Section 59.3.4, "How to Customize the Runtime Environment for JSR 286 Portlets"
- Section 59.3.5, "How to Use Public Render Parameters in JSR 286 Portlets"
- Section 59.3.6, "How to Use Portlet Events in JSR 286 Portlets"
- Section 59.3.7, "How to Add Portlet Preferences to JSR 286 Portlets"
- Section 59.3.8, "How to Use Portlet Filters in JSR 286 Portlets"
59.3.1 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. Example 59–1 shows an example portlet deployment descriptor file.

Example 59–1 Example Portlet Deployment Descriptor File

```xml
  <portlet id="1328624289503">
    <portlet-name>myPortlet</portlet-name>
    <display-name xml:lang="en-US">Portlet1</display-name>
    <display-name xml:lang="en">English Display Name</display-name>
    <portlet-class>portlet.Portlet1</portlet-class>
    <expiration-cache>300</expiration-cache>
    <supports>
      <mime-type>text/html</mime-type>
      <portlet-mode>edit</portlet-mode>
    </supports>
    <supported-locale>en-US</supported-locale>
    <resource-bundle>portlet.resource.Portlet1Bundle</resource-bundle>
    <portlet-info>
      <title>This Is My Portlet</title>
      <short-title>My Portlet</short-title>
      <keywords/>
    </portlet-info>
    <portlet-preferences>
      <preference>
        <name>portletTitle</name>
      </preference>
    </portlet-preferences>
  </portlet>
  <custom-portlet-mode>
    <portlet-mode>about</portlet-mode>
  </custom-portlet-mode>
  <custom-portlet-mode>
    <portlet-mode>config</portlet-mode>
  </custom-portlet-mode>
  <custom-portlet-mode>
    <portlet-mode>edit_defaults</portlet-mode>
  </custom-portlet-mode>
</portlet-app>
```
Developing JSR 286 Java Portlets

After you have created your JSR 286 portlet using the wizard, you can edit the portlet.xml file to edit the portlet 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 the Source view.

To edit the portlet deployment descriptor file:

1. In the Application Navigator, open the portlet producer application.
2. Expand the portlet project (for example, Portlets).
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.

   **Tip:** If you prefer to edit the source code directly, click the Source tab.

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 59.3.6, "How to Use Portlet Events in JSR 286 Portlets."
   - **Parameters**—Use to create and manage public render parameters for use by all portlets in the application. For more information, see Section 59.3.5, "How to Use Public Render Parameters in JSR 286 Portlets."
   - **Filters**—Use to specify filter information for the portlets in the application. For more information, see Section 59.3.8, "How to Use Portlet Filters in JSR 286 Portlets."
   - **Filter Mappings**—Use to assign portlet filters to individual portlets. For more information, see Section 59.3.8, "How to Use Portlet Filters in JSR 286 Portlets."

### 59.3.2 How to Add Custom Portlet Modes to JSR 286 Portlets

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 59.2, "Creating 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:
WebCenter Portal supports the following additional custom modes for JSR 286 portlets:

- About
- Config
- Edit Defaults
- Preview
- Print

After the initial creation of the portlet, you can also define your own custom portlet modes.

The principles of implementing portlet modes are the same for all modes.

To add a custom portlet mode:

1. Open the Overview Editor for the `portlet.xml` file.
   For more information, see Section 59.3.1, "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 check box if applications that consume the portlet do not need to be aware of the portlet mode, that is, the mode is used internally by the portlet.
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.

59.3.3 How to Access User Information in JSR 286 Portlets

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 59.3.1, "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. For a list of attribute names, see the "User Information Attribute Names” appendix in the Java Portlet Specification.

   **Tip:** For a list of the user information attributes supported by WebCenter Portal, see the "Mapping User Attributes to LDAP Directories” section in the Securing Applications with Oracle Platform Security Services.

6. In the Description field, enter a description for the user attribute.

7. Save the portlet.xml file.

   At runtime, the portlet container maps the defined user attributes to the appropriate values. 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.

59.3.4 How to Customize the Runtime Environment for JSR 286 Portlets

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 topics:

- Section 59.3.4.1, "Supported Container Runtime Options"
- Section 59.3.4.2, "Setting Container Runtime Options for All Portlets in an Application"
- Section 59.3.4.3, "Setting Container Runtime Options for Individual Portlets"

59.3.4.1 Supported Container Runtime Options

Table 59–1 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:


<table>
<thead>
<tr>
<th>Container Runtime Option</th>
<th>Supported by</th>
<th>JSR 286 Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>javax.portlet.actionScopedRequestAttributes</td>
<td>Application</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Portlet</td>
<td></td>
</tr>
<tr>
<td>Container Runtime Option</td>
<td>Supported by</td>
<td>JSR 286 Standard</td>
</tr>
<tr>
<td>--------------------------</td>
<td>--------------</td>
<td>------------------</td>
</tr>
<tr>
<td>javax.portlet.escapeXml</td>
<td>Application</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Portlet</td>
<td></td>
</tr>
<tr>
<td>java.portlet.renderHeaders</td>
<td>Not supported</td>
<td>Yes</td>
</tr>
<tr>
<td>javax.portlet.servletDefaultSessionScope</td>
<td>Application</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Portlet</td>
<td></td>
</tr>
<tr>
<td>com.oracle.portlet.allowEventPayloadsWithoutJAXBBinding</td>
<td>Application</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Portlet</td>
<td></td>
</tr>
<tr>
<td>com.oracle.portlet.allowWsrpExport</td>
<td>Application</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Portlet</td>
<td></td>
</tr>
<tr>
<td>com.oracle.portlet.compatibilityMode</td>
<td>Application</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Portlet</td>
<td></td>
</tr>
<tr>
<td>com.oracle.portlet.defaultProxiedResourceRequiresWsrpRewrite</td>
<td>Application</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Portlet</td>
<td></td>
</tr>
<tr>
<td>com.oracle.portlet.defaultServedResourceRequiresWsrpRewrite</td>
<td>Application</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Portlet</td>
<td></td>
</tr>
<tr>
<td>com.oracle.portlet.disallowResourceServing</td>
<td>Application</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Portlet</td>
<td></td>
</tr>
<tr>
<td>com.oracle.portlet.escapeXmlEncodeUrls</td>
<td>Application</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Portlet</td>
<td></td>
</tr>
<tr>
<td>com.oracle.portlet.eventPayloadsXmlType</td>
<td>Application</td>
<td>No</td>
</tr>
<tr>
<td>com.oracle.portlet.excludedActionScopeRequestAttributes</td>
<td>Application</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Portlet</td>
<td></td>
</tr>
<tr>
<td>com.oracle.portlet.externalScopeRequestAttributes</td>
<td>Application</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Portlet</td>
<td></td>
</tr>
<tr>
<td>com.oracle.portlet.importCssToIFrame</td>
<td>Application</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Portlet</td>
<td></td>
</tr>
<tr>
<td>com.oracle.portlet.minimumWsrpVersion</td>
<td>Application</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Portlet</td>
<td></td>
</tr>
<tr>
<td>com.oracle.portlet.offerPortletOverWsrp</td>
<td>Application</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Portlet</td>
<td></td>
</tr>
<tr>
<td>com.oracle.portlet.portletInfoProvider</td>
<td>Application</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Portlet</td>
<td></td>
</tr>
<tr>
<td>com.oracle.portlet.redirectAfterAction</td>
<td>Application</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Portlet</td>
<td></td>
</tr>
<tr>
<td>com.oracle.portlet.renderHeaders</td>
<td>Application</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Portlet</td>
<td></td>
</tr>
<tr>
<td>com.oracle.portlet.requireIFrame</td>
<td>Application</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Portlet</td>
<td></td>
</tr>
<tr>
<td>com.oracle.portlet.streamingOptimized</td>
<td>Application</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Portlet</td>
<td></td>
</tr>
</tbody>
</table>
Developing JSR 286 Java Portlets

javax.portlet.actionScopedRequestAttributes
Specifies whether to store action-scoped request attributes so that they are available to ports 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 these cases, for JSR 286, the output uses only ampersand characters. To use ampersand entities or to be able to specify the XML encoding to use when generating URLs not using the tag libraries, see the BaseURL.write() methods.

### Table 59-1 (Cont.) Supported Container Runtime Options

<table>
<thead>
<tr>
<th>Container Runtime Option</th>
<th>Supported by</th>
<th>JSR 286 Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.oracle.portlet.suppressWsrpOptimisticRender</td>
<td>Application</td>
<td>No</td>
</tr>
<tr>
<td>Portlet</td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.oracle.portlet.trapWsrpRenderExceptions</td>
<td>Application</td>
<td>No</td>
</tr>
<tr>
<td>Portlet</td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.oracle.portlet.trimEncodeUrls</td>
<td>Application</td>
<td>No</td>
</tr>
<tr>
<td>Portlet</td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.oracle.portlet.useWsrpUserContextForUserAuthentication</td>
<td>Application</td>
<td>No</td>
</tr>
<tr>
<td>com.oracle.portlet.wsrpHeaderMode</td>
<td>Application</td>
<td>No</td>
</tr>
<tr>
<td>Portlet</td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.oracle.portlet.wsrpLegacyPortletHandle</td>
<td>Portlet</td>
<td>No</td>
</tr>
<tr>
<td>com.oracle.portlet.wsrpPortletHandle</td>
<td>Portlet</td>
<td>No</td>
</tr>
<tr>
<td>oracle.portlet.bridge.adf.raiseUndeclaredContextualEvents</td>
<td>Portlet</td>
<td>No</td>
</tr>
</tbody>
</table>

Table 59-1 (Cont.) Supported Container Runtime Options
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.
Developing JSR 286 Java Portlets

com.oracle.portlet.defaultServedResourceRequiresWsrpRewrite

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 (\([/\text{namespace}]\text{localpart}\)). 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.excludedActionScopeRequestAttributes` 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.trapWsrfRenderExceptions**
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.

59.3.4.2 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 59.3.1, "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 59–1.

   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, for example:

   `<container-runtime-option>
    <name>com.oracle.portlet.requireIFrame</name>
    <value>true</value>
   </container-runtime-option>`


59.3.4.3 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 59.3.1, "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 59–1.

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, for example:

```xml
<container-runtime-option>
  <name>com.oracle.portlet.requireIFrame</name>
  <value>false</value>
</container-runtime-option>
```


59.3.5 How to Use Public Render Parameters in JSR 286 Portlets

Public render parameters enable portlets to share parameter values, allowing a form of interportlet 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 file (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.

If you declare and define public render parameters as described below, interportlet communication happens automatically, without any further coding required on your part. For more information, see Section 63.5.3, "What You May Need to Know About Interportlet Communication."

This section includes the following topics:

- Section 59.3.5.1, "Declaring a Public Render Parameter at the Application Level"
- Section 59.3.5.2, "Defining a Public Render Parameter for a Portlet"
- Section 59.3.5.3, "Example: Public Render Parameters"

For more information about public render parameters and how to implement them, see the JSR 286 specification at:


59.3.5.1 Declaring 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 file (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 59.3.1, "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 drop-down list, select whether you are specifying a fully qualified name (QName) for the parameter, or just the local part of the name.
   - **Qualified 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 unintentionally interfere with the other portlets on a page regardless of where those portlets come from.
   - **Unqualified Name**—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. Enter a name for the parameter.
   - If you selected **Qualified Name**—in the Namespace field, enter the namespace for the parameter and in the Name field, enter the local part of the parameter name.
   - 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).

6. 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.

7. 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.

8. (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.

**Tip:** 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.

9. The public render parameter is added to the application definition section of the `portlet.xml` file:

```xml
<portlet-app ...
  <portlet>...</portlet>
  <portlet>...</portlet>
  ...
  <public-render-parameter>
    <description>Parameter for zip code</description>
    <identifier>Zip</identifier>
    <qname xmlns:x="http://example.com/params">x:Zip</qname>
    <alias xmlns:x="http://example.com/params">x:ZipCode</alias>
  </public-render-parameter>
  ...
</portlet-app>
```

10. Save the `portlet.xml` file.

59.3.5.2 Defining a Public Render Parameter for 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 59.3.1, "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 59.3.5.1, "Declaring 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`:

   ```xml
   <portlet-app ...
   ```
7. Save the portlet.xml file.
8. You can now use this public render parameter in the code for your portlet.

59.3.5.3 Example: Public Render Parameters

The following example shows how to use public parameters to link the behavior of two portlets using parameters with different names. In the example, we take a generic Parameter Form portlet that allows us to update public render parameters, and show how that can be linked to a Stock Price portlet that renders stock prices, taking the stock symbol from a public render parameter. The generic name of the parameter from the Parameter Form portlet (parameter1) does not match the name of the parameter in the Stock Price portlet (stocksymbol), so the developer of the Stock Price portlet uses an alias to provide a link between the two parameters.

Example 59–2 shows an excerpt from the portlet.xml file for the portlet producer application that contains the Parameter Form portlet. The portlet producer application defines a public render parameter (parameter1) that is supported by the Parameter Form portlet.

Example 59–2 Parameter Definition for the Parameter Form Portlet

```
<portlet-app ...>
  <portlet id="1234567890">
    ...
    <supported-public-render-parameter>parameter1</supported-public-render-parameter>
    ...
  </portlet>
  ...
  <public-render-parameter>
    <identifier>parameter1</identifier>
    <qname xmlns:op="http://xmlns.oracle.com/portlet/oracle-portlet-app">
      op:parameter1
    </qname>
    <description xml:lang="en">First parameter set by portlet</description>
  </public-render-parameter>
  ...
</portlet-app>
```

The code for the Parameter Form portlet sets the value of the parameter1 portlet when a value is entered in the portlet (Example 59–3).

Example 59–3 Parameter Processing for the Parameter Form Portlet

```
public void processAction(ActionRequest request,
```
Developing JSR 286 Java Portlets

Building Standards-Based Java Portlets Using JSR 286

```
ActionResponse actionResponse)
  throws PortletException, IOException
{
  //
  //  Read the new parameter value posted in the portlet form
  //
  String param1 = actionResponse.getParameter("form_param1");

  //
  //  Set the new parameter values. These will be interpreted by the
  //  container as public render parameters as the names match the names
  //  of the declared parameters.
  //
  actionResponse.setRenderParameter("parameter1", param1);
}
```

Example 59–4 shows an excerpt from the portlet.xml file for the portlet producer application that contains the Stock Price portlet. This portlet producer application also defines a public render parameter (stocksymbol) that is supported by the Stock Price portlet. For the stocksymbol parameter to link to the Parameter Form portlet’s parameter, an alias for parameter1 is added.

**Example 59–4  Parameter Definition for the Stock Price Portlet**

```
<portlet-app ...>
<portlet>
  ...
  <supported-public-render-parameter>
    stocksymbol
  </supported-public-render-parameter>
  ...
</portlet>

<public-render-parameter>
  <description xml:lang="en">The stock symbol</description>
  <identifier>stocksymbol</identifier>
  <qname xmlns:s="http://www.oracle.com/stocks">s:stocksymbol</qname>
  <!-- Alias matches the Parameter Form portlet's first parameter name -->
  <alias xmlns:op="http://xmlns.oracle.com/portlet/oracle-portlet-app">
    op:parameter1
  </alias>
</public-render-parameter>
...
</portlet-app>
```

The code for the Stock Price portlet reads the value of the public render parameter posted by the Parameter Form portlet (Example 59–5).

**Example 59–5  Parameter Processing for the Stock Price Portlet**

```
public void doView(RenderRequest request, RenderResponse response)
  throws PortletException, IOException
{
  response.setContentType("text/html; charset=UTF-8");
  PrintWriter out = response.getWriter();

  // Retrieve any values for the public render parameter.
  // The parameter is looked up in the request using its identifier in portlet.xml
  // (not it's name or alias).
  String symbol = request.getParameter("stocksymbol");
```
When these two portlets are dropped onto the same page, they are automatically linked together. Entering a value in the first parameter field of the Parameter Form portlet causes the Stock Price portlet to be updated with the new value.

59.3.6 How to Use Portlet Events in JSR 286 Portlets

Portlet events are a JSR 286 feature that enables interportlet 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.

If you declare and define portlet events as described below, interportlet communication happens automatically, without any further coding required on your part. For more information, see Section 63.5.3, "What You May Need to Know About Interportlet Communication."

This section includes the following topics:

- Section 59.3.6.1, "Declaring a Portlet Event at the Application Level"
- Section 59.3.6.2, "Defining a Processing Event for a Portlet"
- Section 59.3.6.3, "Defining a Publishing Event for a Portlet"
- Section 59.3.6.4, "Example: Portlet Events"

For more information about portlet events and how to implement them, see the JSR 286 specification at:


59.3.6.1 Declaring 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 file (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 59.3.1, "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 drop-down list, select whether you are specifying a fully qualified name (QName) for the portlet event, or just the local part of the name.
- **Qualified Name**—A QName uniquely identifies the portlet event across applications by specifying a namespace for the event 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.

- **Unqualified Name**—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.

  **Tip:** To check whether an application default namespace has been defined, click the Application tab and see if a value has been entered in the Default Namespace field.

5. Enter a name for the portlet event.
   - If you selected **Qualified Name**—In the **Namespace** field, enter the namespace for the portlet event and in the **Name** field, enter the local part of the event name.
   - 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 default namespace is defined for the application).

6. In the **Payload Type** field, enter or browse for the data type of the payload provided by the portlet event.

7. In the **Description** field, enter a description for the portlet event.

8. (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.

9. The portlet event is added to the application definition section of the `portlet.xml` file:

   ```xml
   <portlet-app ...
   <portlet>...</portlet>
   <portlet>...</portlet>
   ...
   <event-definition id="latLong">
   <qname xmlns:x="http://xmlns.oracle.com/portlet/EventSample">
     x:latLong
   </qname>
   <value-type>portlet.LatLong</value-type>
   </event-definition>
   ...
   </portlet-app>
   ```

10. Save the `portlet.xml` file.
59.3.6.2 Defining a Processing Event for a Portlet

If you 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 59.3.1, "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 59.3.6.1, "Declaring 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:
   
   ```xml
   <portlet-app ...>
     <portlet id="map">
       ...
       <supported-processing-event id="latLong">
         <qname xmlns:x="http://xmlns.oracle.com/portlet/EventSample">
           x:latLong
         </qname>
       </supported-processing-event>
       ...
     </portlet>
     ...
   </portlet-app>
   ...
   <event-definition id="latLong">
     <qname xmlns:x="http://xmlns.oracle.com/portlet/EventSample">
       x:latLong
     </qname>
     <value-type>portlet.LatLong</value-type>
   </event-definition>
   ...
   
   You can now use this portlet event in the code for your portlet.

59.3.6.3 Defining a Publishing Event for 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.
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 59.3.6.3, "Defining a Publishing Event for a Portlet."

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:

```xml
<portlet-app ...
    <portlet id="locations">
        ...
        <supported-publishing-event>
            <qname xmlns:x="http://xmlns.oracle.com/portlet/EventSample">
                x:latLong
                <qname>
            </supported-publishing-event>
        ...
    </portlet>
    ...
</portlet-app>
```


You can now use this portlet event in the code for your portlet.

59.3.6.4 Example: Portlet Events

The following example shows how to use portlet events to use the actions in one portlet to affect another portlet on the same page. In the example, we have a portlet, the Department Locations portlet, that lists the geographical locations of a company’s various offices. The example shows how that portlet can be linked to another portlet, the Map portlet, which displays a Google map of the location selected in the Department Locations portlet.

Example 59–6 shows an excerpt from the portlet.xml file for the portlet producer application that contains the two portlets. The portlet producer application defines a portlet event (latLong) that is supported by the two portlets. The payload of the event is an object that encapsulates the latitude and longitude of a location.
**Example 59–6  Portlet Event Definition**

```
<event-definition id="latLong">
  <qname xmlns:x="http://xmlns.oracle.com/portlet/EventSample">x:latLong</qname>
  <value-type>portlet.LatLong</value-type>
</event-definition>
```

Example 59–7 shows that the Department Locations portlet uses the `latLong` event as a publishing event. That is, it raises the event under particular circumstances.

**Example 59–7  Publishing Event Definition for the Location Portlet**

```
<portlet id="locations">
  ...
  <supported-publishing-event>
    <qname xmlns:x="http://xmlns.oracle.com/portlet/EventSample">x:latLong</qname>
  </supported-publishing-event>
  ...
</portlet>
```

The Department Locations portlet raises the `latLong` event in its `processAction` method if a location is selected in the portlet (Example 59–8). It also sets the payload of the event to the latitude and longitude of the selected location.

**Example 59–8  Event Processing for the Department Locations Portlet**

```java
public void processAction(ActionRequest request, ActionResponse response)
  throws PortletException {
  String locationId = request.getParameter("locationId");
  Location location = getLocation(Integer.parseInt(locationId));
  if (location != null) {
    // QName matches the event declared as a supported-publishing-event in
    // portlet.xml for this portlet.
    // LatLong event used by the Map portlet.
    response.setEvent(new QName("http://xmlns.oracle.com/portlet/EventSample", "latLong"),
      new LatLong(location.getLatitude(), location.getLongitude()));
  }
}
```

Example 59–9 shows that another portlet, the Map portlet, also uses the `latLong` event, but this time as a processing event. That is, it listens out for the event and takes a particular action if the event is raised elsewhere on the page.

**Example 59–9  Processing Event Definition for the Map Portlet**

```
<portlet id="map">
  ...
  <supported-processing-event id="latLong">
    <qname xmlns:x="http://xmlns.oracle.com/portlet/EventSample">x:latLong</qname>
  </supported-processing-event>
  ...
</portlet>
```

The `processEvent` method for the Map portlet receives the `latLong` event and uses the payload from that event to set the `LAT_LONG_PARAMETER` render parameter, which the
portlet uses during rendering to determine the location to display in the Google map (Example 59–10).

**Example 59–10  Event Processing for the Map Portlet**

```java
@override
class MapPortlet
{
    public void processEvent(EventRequest request, EventResponse response)
    throws PortletException, IOException
    {
        response.setRenderParameters(request);

        Event event = request.getEvent();
        if (event.getName().equals("latLong"))
        {
            response.setRenderParameter(LAT_LONG_PARAMETER, event.getValue().toString());
        }
    }
}
```

**Tip:** In this case, the events used by the Department Locations and Map portlets have the same name (latLong). If the Map portlet listens for an event using a different name (for example, geolocation), the two portlets could still be automatically linked by defining an alias for the geolocation event in the portlet.xml file as follows:

```xml
<event-definition id="geolocation">
    <qname xmlns:x="http://xmlns.oracle.com/portlet/Map">
        x:geolocation
    </qname>
    <alias xmlns:x="http://xmlns.oracle.com/portlet/EventSample">
        x:latLong
    </alias>
    <value-type>portlet.LatLong</value-type>
</event-definition>
```

Note, however, that the payload type of the two events must be the same for automatic linking to work.
**Note:** For an event payload to be passed between portlets it must be possible to form an XML payload from event payload. For this to be possible one of the following must be true:

- The payload must have a JAXB binding, for example by adding the `XmlRootElement` annotation to the portlet class:

```java
package portlet;

import java.io.Serializable;
import javax.xml.bind.annotation.XmlRootElement;

@XmlRootElement(namespace="http://xmlns.oracle.com/portlet/EventSample")
public class LatLong implements Serializable {
    private final double _latitude;
    private final double _longitude;
    ...
```

- The `allowEventPayloadsWithoutJAXBBindings` container runtime option must be set to `true` in `portlet.xml`.

For more information, see Section 59.3.4, "How to Customize the Runtime Environment for JSR 286 Portlets."

### 59.3.7 How to Add Portlet Preferences to JSR 286 Portlets

*Portlet preferences* enable end users to personalize or customize portlets at runtime. Personalizations are visible only to the user that performs them; not to other users. Customizations are visible to all users who have not personalized the portlet.

By default, when you create a JSR 286 portlet using the JDeveloper wizard, a portlet preference is created to enable users to change 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 make other modifications to the portlet at runtime.

The procedure below describes how to add a portlet preference to the `portlet.xml` file after portlet creation. For information about how to add a portlet preference during portlet creation, see Section 59.2.1, "How to Create a JSR 286 Java Portlet."

This section includes the following topics:

- Section 59.3.7.1, "Adding a Portlet Preference to a JSR 286 Portlet"
- Section 59.3.7.2, "Example: Simple Portlet Personalization"

#### 59.3.7.1 Adding a Portlet Preference to a JSR 286 Portlet

To enable users to modify portlet behavior or content at runtime, you must create portlet preferences for those attributes of the portlet that you want users to be able to modify.

To add a portlet preference to a portlet:

1. Open the Overview Editor for the `portlet.xml` file.

   For more information, see Section 59.3.1, "How to Edit the Portlet Deployment Descriptor File."
2. Click the Portlets tab and select the portlet for which you want to add the portlet preference.

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.

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.

   The portlet preference is added to the application definition section of the portlet.xml file:

   ```xml
   <portlet-app ...>
   <portlet>...</portlet>
   <portlet>...</portlet>
   ...
   <portlet-preference>
   <name>portletTitle</name>
   </portlet-preference>
   <portlet-preference>
   <name>myPreference</name>
   <value>7</value>
   </portlet-preference>
   ...
   </portlet-app>
   ```

8. Save the portlet.xml file.

   You can now use this preference in your portlet code using the `getPreference` method.

   **Tip:** If you want the preference to be translatable, you must add the appropriate key-value pairs to the resource bundle class.

59.3.7.2 Example: Simple Portlet Personalization

The following example provides a simple illustration of how you can enable personalization in portlets.

To add simple personalization to a portlet:

1. In JDeveloper, create a Portlet Producer application, accepting the default values.

2. Create a JSR 286 portlet, accepting the default values.

3. In the Application Navigator, expand the Portlets project.

4. Right-click portlet.xml and choose Open.

5. Click the Portlets tab and select portlet1.

6. Click the Advanced tab.

7. Click the Add icon to create a new portlet preference.
8. In the Name field, enter portletContent.

9. In the Application Navigator, right-click view.jsp and choose Open.

10. In the visual editor, click the Source tab and add the code indicated in bold in Example 59–11 to display the portletContent portlet preference:

**Example 59–11  view.jsp Sample Code**

```html
<%@ 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"]; PremtletPreferences prefs = renderRequest.getPreferences(); str = prefs.getValues("portletContent",str); for (int i=0; i<str.length; i++) {
   %><%=i<str.length-1)?str[i]+", ":str[i]%><%=}%>

11. 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.

12. In the visual editor, click the Source tab and add the code indicated in bold in Example 59–12 to implement a form field for users to enter a value for the portletContent customization preference:

**Example 59–12  edit.jsp Sample Code**

```html
<%@ 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>
</form>
```
13. In the Application Navigator, right-click Portlet1.java and choose Open.

14. In the visual editor, click the Source tab and add the following two lines of code (indicated in bold) to the processAction method:

```java
// 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));
}
```
When this portlet is available on a page, users can personalize it. The Edit mode of the portlet provides a Portlet Content field, where users can enter text. The text entered is then displayed as the content of the portlet.

59.3.8 How to Use Portlet Filters in JSR 286 Portlets

Portlet filters are a JSR 286 feature that enables 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 topics:

- Section 59.3.8.1, "Adding a Portlet Filter to an Application"
- Section 59.3.8.2, "Applying a Portlet Filter to a Portlet"

For more information about portlet filters and how to implement them, see the JSR 286 specification at:


59.3.8.1 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 59.3.1, "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 you can either create a new portlet filter, or select a previously created portlet filter:

   - To create a new portlet filter:
     a. Select New Portlet Filter.
     b. 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.

     c. In the Package field, enter or browse for the package to contain the new filter class.

     d. Select one or more of the Lifecycle Phase check boxes to specify which of the java.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.
Tip: 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.

- To select an existing portlet filter:
  a. Select Choose from Existing Class.
  b. Enter or browse for the filter class.

5. Click OK.

6. In the Display Name field, enter a more user-friendly name for the portlet filter.

7. In the Description field, enter a description for the portlet filter.

8. 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.

The portlet filter is added to the application definition section of the portlet.xml file:

```xml
<portlet-app ...
  <portlet>...</portlet>
  <portlet>...</portlet>
  ...
  <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>
  ...
</portlet-app>
```


59.3.8.2 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 59.3.1, "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** drop-down list, select the portlet filter that you want to apply to the portlet.

   The drop-down list is populated with all the portlet filters that are defined in the portlet.xml file.

5. From the **Portlet Name** drop-down list, select the portlet to which you want to apply the portlet filter.

   The drop-down 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.

   The filter mapping is added to the application definition section of the portlet.xml file:

   ```xml
   <filter-mapping>
     <filter-name>myFilter</filter-name>
     <portlet-name>myPortlet</portlet-name>
   </filter-mapping>
   ```

6. Use the **Top**, **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:** Portlet 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. That is, 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.

7. Save the portlet.xml file.

59.3.9 How to Implement Interportlet Communication Across Different Pages

Generally, interportlet communication occurs between portlets that are displayed on the same page. However, it is possible to communicate between portlets on different pages, as shown in the following example.

To implement interportlet communication across different pages:
1. Create a custom data control and expose a method to capture the event that initiates the interportlet communication.

```java
public void handleEvent(Object payload) {
    if(!(payload instanceof HashMap)){
        throw new IllegalArgumentException("Payload not a HashMap<String,String>.");
    }

    String p1 = "", p2 = "", p3 = "";
    HashMap map = (HashMap) payload;
    String[] pa1 = (String[])map.get("parameter1");
    String[] pa2 = (String[])map.get("parameter2");
    String[] pa3 = (String[])map.get("parameter3");
    if(pa1 != null)
        p1 = pa1[0];
    if(pa2 != null)
        p2 = pa2[0];
    if(pa3 != null)
        p3 = pa3[0];
    //Only forward when all 3 parameters have values
    if(!p1.equals("") && !p2.equals("") && !p3.equals("") &&
    HashMap<String,String> paramMap = new HashMap<String,String>();
    paramMap.put("parameter1", p1);
    paramMap.put("parameter2", p2);
    paramMap.put("parameter3", p3);
    Map<String, Object> pageFlowScope =
    ADFContext.getCurrent().getPageFlowScope();
    pageFlowScope.put("paramMap", paramMap);
    //now the navigation
    FacesContext fctx = FacesContext.getCurrentInstance();
    Application application = fctx.getApplication();
    NavigationHandler navHandler = application.getNavigationHandler();
    navHandler.handleNavigation(fctx, null, "target");
}
```

2. Add this method to the page bindings and create an event from it.

```xml
<methodAction id="handleEvent" InstanceName="EventDataControl.dataProvider"
             DataControl="EventDataControl" RequiresUpdateModel="true"
             Action="invokeMethod" MethodName="handleEvent"
             IsViewObjectMethod="false">
    <NamedData NDName="payload" NDType="java.lang.Object"/>
    <events xmlns="http://xmlns.oracle.com/adfm/contextualEvent">
        <event name="handleEvent"/>
    </events>
</methodAction>
```

3. Wire the event to the portlet event. This ensures that the payload of the event triggered by the portlet is passed to the custom method.

```xml
<eventMap xmlns="http://xmlns.oracle.com/adfm/contextualEvent">
    <event name="ParameterFormPortlet1_1_Event">
        <producer region="">
            <consumer region="" handler="handleEvent" handleCondition="">
                <parameters>
                    <parameter name="payload" value="#{payLoad}"/>
                </parameters>
            </consumer>
        </producer>
    </event>
</eventMap>
```
4. Create a HashMap to pass the payload to the pageFlowScope of the page that contains the second portlet.

```java
HashMap<String, String> paramMap = new HashMap<String, String>();
paramMap.put("parameter1", p1);
paramMap.put("parameter2", p2);
paramMap.put("parameter3", p3);
Map<String, Object> pageFlowScope =
  ADFContext.getCurrent().getPageFlowScope();
pageFlowScope.put("paramMap", paramMap);
```

5. Define a navigation rule in the faces-config.xml file to navigate to the target page.

```java
FacesContext fctx = FacesContext.getCurrentInstance();
Application application = fctx.getApplication();
NavigationHandler navHandler = application.getNavigationHandler();
navHandler.handleNavigation(fctx, null, "target");
```

6. Pass the parameters in the pageFlowScope to the target portlet using the parameterMap attribute.

```xml
<portlet id="ParameterDisplayPortlet1_1"
  portletInstance="/oracle/adf/portlet/PortletExample/ap/Ei2default_354ce3cl_013d_1000_8002_c0a8301a490"
  class="oracle.adf.model.portlet.binding.PortletBinding"
  retainPortletHeader="false"
  listenForAutoDeliveredPortletEvents="true"
  listenForAutoDeliveredParameterChanges="true"
  xmlns="http://xmlns.oracle.com/portlet/bindings"
  parameterMap="#{pageFlowScope.paramMap}">
  <events>
    <event eventType="ParametersChange"
      name="ParameterDisplayPortlet1_1_Event"/>
  </events>
</portlet>
```

**Tip:** You can also create a single generic eventHandler and add it to the page template. By adding a method binding in the page template, this is available on all pages. By using a wildcard for the publisher, you can also make sure that it listens to all events with a specific name.

For more information and to download a sample application, see the blog entry at:

http://www.ateam-oracle.com/inter-portlet-communication-between-pages/

### 59.3.10 How to Enhance JSR 286 Portlet Performance with 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. JSR 286 portlets support expiry-based and validation-based caching.

For more information about caching, see Section 57.4.5, "Portlet Performance."

This section includes the following topics:

- Section 59.3.10.1, "Implementing Expiry-Based Caching in JSR 286 Portlets"
59.3.10.1 Implementing Expiry-Based Caching in JSR 286 Portlets

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 59.3.1, "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:
     ```xml
     <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 (n), in seconds, in the adjacent field. Setting this option results in the following code being added to the portlet.xml file:
     ```xml
     <portlet>
     ...
     <expiration-cache>n</expiration-cache>
     ...
     </portlet>
     ```
7. Save the portlet.xml file.
Implementing Validation-Based Caching in JSR 286 Portlets

Implementation of validation-based caching takes place after the initial portlet creation and requires hand coding.

Example 59–13 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 59–13  A JSR 286 Portlet Implementing Validation-Based Caching

```java
protected void doView (RenderRequest request, RenderResponse response)
throws PortletException, IOException
{
    CacheControl cacheControl = response.getCacheControl();
    String eTag = request.getETag();
    if (isMarkupStillValid(eTag))
    {
        // Wait 30 seconds before checking ETag again
        cacheControl.setExpirationTime(30);
        // Tell consumer to use its cached content
        cacheControl.setUseCachedContent(true);
        return;
    }
    // ETag not valid so set a new one ...
    // Define a new ETag
    cacheControl.setETag(generateETag(...));
    // Wait 60 seconds before checking ETag again
    cacheControl.setExpirationTime(60);
    // ... 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:


59.3.11 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 you 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 59–14 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 59–14  Resource Proxy for WSRP

```java
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`.

You can use the `defaultProxiedResourceRequiresWsrpRewrite` container runtime option to specify the default WSRP `requiresRewrite` flag to use. The option specified by this container runtime option (which is set to `true` by default) is used unless overridden by the request attribute. For more information, see "com.oracle.portlet.defaultProxiedResourceRequiresWsrpRewrite."

59.3.12 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 59–15 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 59–15  Stateless Resource Proxying

```java
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.

59.3.13 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 74.17, "Securing Identity Propagation Through WSRP Producers with WS-Security."

59.3.14 How to Manage the Persistence Store for JSR 286 Portlets

The portlet persistence store is used for persisting consumer registration handles and portlet preference data. Portlet producers can use one of three types of persistence store:

- **Consumer**—Ties the producer metadata to the consumer application. This is the recommended method.
- **Database**—Persists data using a relational database. This is mainly provided for backward compatibility but may be useful if there is likely to be a large number of customizations.
- **File**—Persists data using the file system. This is provided for backward compatibility. You should not use a file based persistence store in your production environment as it does not support multi-tier or high availability environments. You may, however, want to use a file based persistence store while testing your application using Integrated WLS.

For more information, see Section 57.4.4, "Portlet Personalization and Customization."

This section includes the following topics:

- Section 59.3.14.1, "Setting Up a Persistence Store for a WSRP Producer"
- Section 59.3.14.2, "Migrating a WSRP Producer Persistence Store"
- Section 59.3.14.3, "Enabling Java Object Cache for Database Persistence Store Access"

59.3.14.1 Setting Up a Persistence Store for a WSRP Producer

WSRP portlet producers use a JNDI variable (persistentStore) to determine which type of persistence store to use. When you create a portlet for the first time in an application, by default this variable is set to Consumer. You can change the value of this variable in the web.xml file of the portlet producer application.

**Note:** If you create a portlet in an application that already contains other portlets, the existing persistence store setting is maintained.

If you use a File persistence store, you must also use the fileStoreRoot JNDI variable to specify the path to the root directory to be used by the persistence store.

Table 59–2 lists and describes the JNDI variables used to specify the persistence store for WSRP producers.
To set up the persistence store for a WSRP portlet producer:

1. In JDeveloper, open the portlet producer application for which you want to set up the persistence store.
2. Expand the portlet project (for example, Portlets).
3. Expand the Web Content node and then the WEB-INF node.
4. Right-click web.xml and choose Open.
5. Click the Source tab.
6. To specify a Database persistence store, add the following code:

   ```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>Database</env-entry-value>
   </env-entry>
   ```

   To specify a File-based persistence store, add the following code:

   ```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>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>
   ```

   To specify a Consumer persistence store, add the following code:

   ```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>
   ```

---

**Table 59–2  WSRP Producer Database Preference Store-Related JNDI Variable**

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Variable Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>oracle/portal/wsrp/server/persistentStore</td>
<td>Database</td>
<td>Determines which data store (File, Database, or Consumer) is used for persisting a portlet producer application’s consumer registration handles and portlet preferences.</td>
</tr>
<tr>
<td></td>
<td>File</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Consumer</td>
<td></td>
</tr>
<tr>
<td>oracle/portal/wsrp/server/fileStoreRoot</td>
<td>portletdata</td>
<td>Defines the path to the root directory to be used by the file preference store.</td>
</tr>
<tr>
<td></td>
<td>portletdata</td>
<td>Absolute paths are interpreted relative to the file system root. Relative paths are interpreted relative to the ${WC_ORACLE_HOME}/portal directory.</td>
</tr>
<tr>
<td></td>
<td>portletdata</td>
<td>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.</td>
</tr>
</tbody>
</table>

---

To specify a Consumer persistence store, add the following code:
7. Save the web.xml file.
8. Restart your WebLogic Server.

59.3.14.2 Migrating a WSRP Producer Persistence Store
If you want to change the type of persistence store used by your portlet producer, for example when moving from a testing to production environment, you can migrate the existing data from the old to the new persistence store.

This section includes the following topics:

- Section 59.3.14.2.1, "Migrating a Database or File-Based Persistence Store"
- Section 59.3.14.2.2, "Migrating to or from a Consumer Persistence Store"
- Section 59.3.14.2.3, "Moving a WSRP Portlet Producer"

59.3.14.2.1 Migrating a Database or File-Based Persistence Store
The WSRP portlet producer persistence store migration utility, PersistenceMigrationTool, enables you to migrate existing data between Database and File-based persistence stores (for example, from a File-based persistence store used for testing to a production Database persistence 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:** 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.

To migrate a database or file-based persistence store:

1. Ensure that the appropriate libraries are referenced in the classpath:

   ```
   ./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_11.1.1.1/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

2. Run the PersistenceMigrationTool using the following syntax:
   
   java oracle.portlet.server.containerimpl.PersistenceMigrationTool
   -sourceType [file | db]
   -destType [file | db]
   {-sourcePath [dir | 
     -sourceUsername username -sourcePassword password -sourceDatabase db
     -sourceDriver srcDriver]}
   {-destPath [dir | destUsername username -destPassword password -destDatabase db
     -destDriver dstDriver]}
   [-debug]

   where:

   ■ sourceType indicates whether the source store is in a file (file) or database (db). 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 (file) or database (db). 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 persistence store. This argument is required when sourceType is file.

   ■ sourceUsername is the database user name for a persistence store database. This argument is required when sourceType is db.

   ■ sourcePassword is the database password for a persistence store database. This argument is required when sourceType is db.

   ■ sourceDatabase is the name of a persistence store database. This argument is required when sourceType is db.

   ■ sourceDriver is the name of a database driver to use. For example, for a SQL Server database, the database driver is com.microsoft.sqlserver.jdbc.SQLServerDriver. If you do not specify a value for this argument, the migration tool attempts to determine the correct driver to use from the sourceDatabase argument.

   ■ destPath is the location of a file-based persistence store. This argument is required when destType is file.

   ■ destUsername is the database user name for a persistence store database. This argument is required when destType is db.

   ■ destPassword is the database password for a persistence store database. This argument is required when destType is db.

   ■ destDatabase is the name of a persistence store database. This argument is required when destType is db.
- **destDriver** is the name of a database driver to use. For example, for a SQL Server database, the database driver is `com.microsoft.sqlserver.jdbc.SQLServerDriver`. If you do not specify a value for this argument, the migration tool attempts to determine the correct driver to use from the `destDatabase` argument.

- **debug** turns on full logging through standard output to enable users to diagnose issues that arise when the tool runs.

Example 59–16 demonstrates running the `PersistenceMigrationTool` utility. In this example, preferences from a File store are copied to a Database store.

**Example 59–16  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/oracle-portlet-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 file
-sourcePath /data/prefs
-destType db
-destUsername scott
-destPassword tiger
-destDatabase abc.mycompany.com:1521:yourdatabase
```

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/ojdbc6.jar` directory into a temporary directory on the WebCenter Portal Oracle home and reference the `ojdbc6.jar` library from this temporary directory in the classpath.

59.3.14.2.2 Migrating to or from a Consumer Persistence Store  You cannot use the persistence store migration utility to migrate to or from a consumer persistence store. To migrate to or from a consumer persistence store, you must export the data from one producer from the consumer and import into another.

To migrate a consumer persistence store:

1. Export the producer metadata from your consumer application to an EAR file. This contacts the remote producer to get customization data, and so on. You can use the following tools to do this:

   - JDeveloper—see Section 59.3.15.1, "Exporting Portlet Producers at Design Time."
   - WLST—see the "Exporting and Importing Portal Framework Applications for Data Migration" section in *Administering Oracle WebCenter Portal*. 


2. Either remap the producer connection to another producer, which can be using any persistence store type, or change the preference store configuration of the current producer, and restart it.

For information about how to remap the producer connection, see Section 63.3.1, "How to Edit Portlet Producer Registration Settings."

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 persistence store.

59.3.14.2.3 Moving a WSRP Portlet Producer  If you move a portlet producer to a different server, you may also have to move the persistence store for the producer.

After installing the new producer, move the persistence store according to the following:

- Consumer persistence store—Because the data is stored with the consumer, there is no requirement to move the persistence store.
- Database persistence store—Perform one of the following:
  - Configure the new producer to point the WebLogic Server data source to the same database as the original producer.
  - Migrate the persistence store using the Persistence Store Migration Utility (see Section 59.3.14.2, "Migrating a WSRP Producer Persistence Store" for more information).
- File persistence store—Perform one of the following:
  - Configure the new producer to point the same file system location as the original producer.
  - Migrate the persistence store using the Persistence Store Migration Utility (see Section 59.3.14.2, "Migrating a WSRP Producer Persistence Store" for more information).

Finally, update the URL of the producer registration by using Enterprise Manager Fusion Middleware Control or the WLST commands: setWSRPProducerRegistration or setPDKJavaProducerRegistration.

59.3.14.3 Enabling Java Object Cache for Database Persistence Store Access

To improve persistence store performance, you can use Java Object Cache for persistence store access. This avoids the need to access the persistence store on each request. Enable Java Object Cache for the persistence store using the enableJavaObjectCache JNDI variable.

To enable Java Object Cache for persistence store access:

1. In JDeveloper, open the portlet producer application for which you want to set up the persistence store.
2. Expand the portlet project (for example, Portlets).
3. Expand the Web Content node and then the WEB-INF node.
4. Right-click web.xml and choose Open.
5. Click the Source tab.
6. Add the following code:

   `<env-entry>`
<env-entry-name>
  oracle/portal/wsrf/server/enableJavaObjectCache
</env-entry-name>
<env-entry-type>java.lang.Boolean</env-entry-type>
<env-entry-value>true</env-entry-value>
</env-entry>

7. Save the web.xml file.

59.3.15 How to Export and Import Portlet Producers at Design Time

You can export and import portlet producers at design time.

This section includes the following topics:

- Section 59.3.15.1, "Exporting Portlet Producers at Design Time"
- Section 59.3.15.2, "Importing Portlet Producers at Design Time"

59.3.15.1 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.
59.3.15.2 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.

For information about how to create an export archive of producer metadata from a deployed application, see the "Exporting Portlet Client Metadata for Portal Framework Applications" section in Administering 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.

59.4 Testing JSR 286 Portlets

Before making your portlets available in a production environment, it is highly advisable to test them first to make sure that they behave as expected.

This section includes the following topics:

- Section 59.4.1, "How to Run a WSRP Portlet Producer on Integrated WebLogic Server"
- Section 59.4.2, "What Happens When You Run a WSRP Portlet Producer on Integrated WebLogic Server"
- Section 59.4.3, "How to Deploy a WSRP Portlet Producer to the Integrated WebLogic Server"
- Section 59.4.4, "Testing Portlet Personalization"
59.4.1 How to Run a WSRP Portlet Producer on Integrated WebLogic Server

The Integrated WebLogic Server (Integrated WLS) provides a quick and easy way of testing your portlets because it is preconfigured so that you can run applications within JDeveloper without needing to create deployment profiles.

To test a JSR 286 portlet on Integrated WLS:

1. In JDeveloper, open the portlet producer application that owns the portlet that you want to test.
2. Expand the portlet project (for example, Portlets).
3. Expand the Web Content node and then the WEB-INF node.
4. Right-click portlet.xml and choose Run.
   
   Running portlet.xml triggers the packaging and deployment of your portlet producer application on an Integrated WLS instance named after the application.
5. Check 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 59-6 displays in a browser window.
6. 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 the test page in your browser.

7. When the producer is successfully running, you should register it 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 of JSR 286 portlets, follow the instructions provided in Section 63.2.1.1, "How to Register a WSRP Portlet Producer."

To add your portlets to a page, follow the instructions provided in Section 63.5, "Adding Portlets to a Page."

8. To stop an Integrated WLS instance, in the Run Manager tab, select DefaultServer and click the red Stop icon. When the instance stops, the application is undeployed, and therefore, becomes unavailable.

   Tip: You can also stop an Integrated WLS instance by clicking the red Stop icon in the IntegratedWebLogicServer - Log window and selecting DefaultServer.

59.4.2 What Happens When You Run a WSRP Portlet Producer on Integrated WebLogic Server

When you run a WSRP portlet producer 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.

59.4.3 How to Deploy a WSRP Portlet Producer to the Integrated WebLogic Server

When you run a WSRP portlet producer application on the Integrated WLS instance, when the instance stops, the application is undeployed and therefore becomes unavailable. For a more persistent testing scenario, you can deploy your portlet producer application to the Integrated WLS so that it is always available while the Default Server is running.

If you choose this method, then you must first create deployment profiles, as described in Section 61.4, “Creating a WAR Deployment Profile.” If you deploy your application to the Integrated WLS, then the Deployment Configuration dialog displays to enable you to configure and customize deployment settings. The file system MDS repository precreated by JDeveloper displays in the Repository Name field.

For information about deploying and running an application on the Integrated WLS, see Section 7.2, “Deploying a Portal Framework Application to the Integrated WebLogic Server.”

59.4.4 Testing Portlet Personalization

If you have implemented personalization for your portlet, then the Personalize icon appears on the portlet only for authenticated users. Hence, to test the personalization of a portlet, 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 74.12, "Configuring Basic Authentication for Testing Portlet Personalization."

59.4.5 Hiding or Removing the WSRP Test Page

For security purposes, you may want to hide the WSRP test page so that it is visible only to administrators, or you may want to remove it entirely.

This section includes the following subsections:

- Section 59.4.5.1, "Hiding the WSRP Test Page"
- Section 59.4.5.2, "Removing the WSRP Test Page"
Hiding 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 hide the WSRP test page:

1. In JDeveloper, open the portlet producer application for which you want to hide the test page.
2. Expand the portlet project (for example, Portlets).
3. Expand the Web Content node and then the WEB-INF node.
4. Right-click web.xml and choose Open.
5. Click the Source tab.
6. Add the following code:
   ```xml
   <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>
   ```
7. Save the web.xml file.

Removing the WSRP Test Page

You can remove the WSRP test page completely.

To remove the WSRP test page, you must edit an element that is injected into the web.xml file at packaging time, so you must edit the web.xml file in the resulting EAR file.

To remove the WSRP test page:

---

**Note:** There is also a Webservice test page, which enables you to build up SOAP requests to the producer in a web browser. For WSRP portlet producers this test page is disabled by default. Although this test page does not provide any useful information for testing your portlet producers, you can enable it, if desired, by extracting the oracle-webservices.xml file from the deployed producer's WAR file, setting the expose-testpage flag to true for both WSRP v1 and WSRP v2 producers, and then repacking the WAR and EAR files and redeploying the producer.
1. Extract the `web.xml` file from the EAR file created at packaging time and open it in an editor of your choice.

2. Comment out the following code:

   ```xml
   <servlet-mapping>
     <servlet-name>WSRPTestPage</servlet-name>
     <url-pattern>/info</url-pattern>
   </servlet-mapping>
   ```

3. Save the `web.xml` file and add the edited file to the EAR file.

59.5 Deploying JSR 286 Portlets to a WebLogic Managed Server

For information about how to deploy JSR 286 portlets to a WebLogic Managed Server, see Chapter 61, "Deploying Portlet Producers."

When you deploy a portlet producer containing JSR 286 portlets to a WebLogic Managed Server, the configuration settings described in Section 59.4.2, "What Happens When You Run a WSRP Portlet Producer on Integrated WebLogic Server" are added to the EAR file.

59.6 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.

This section includes the following topics:

- Section 59.6.1, "Migrating Java Portlets from WebLogic Portal to WebCenter Portal"
- Section 59.6.2, "Problems With Migrating WLP Portlets to a WebCenter Portal Application"
- Section 59.6.3, "General Tips for Migrating WLP Portlets to a WebCenter Portal Portlet Producer Application"

**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.

59.6.1 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.
59.6.2 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.

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.

59.6.3 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 59.6, "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 59.6.1, "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 the "JSR-286/JSR-168 Portlet Compatibility" section 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 portal 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 the "Working With JSF-Java Portlets" chapter 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 59.6.1, "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 the "Publishing Content Using Content Presenter" chapter in Building Portals with Oracle WebCenter Portal.

- **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 the "WSRP Interoperability with Oracle WebCenter Portal and Oracle Portal" and "Configuring WSRP Security Between WLP and a
59.7 Troubleshooting JSR 286 Java Portlets

This section provides information to assist you in troubleshooting problems you may encounter while creating and testing JSR 286 portlets.

This section includes the following topics:

- Section 59.7.1, "Issues with Creating JSR 286 Portlets"
- Section 59.7.2, "Issues with Testing JSR 286 Portlets"

59.7.1 Issues with Creating JSR 286 Portlets

This section includes the following topics:

- Section 59.7.1.1, "Cannot Access the Create Portlet Wizard"
- Section 59.7.1.2, "Cannot Add the Portlet Functionality that I Want to the Portlet"

59.7.1.1 Cannot Access the Create Portlet Wizard

**Problem**

In the New Gallery, I cannot find the Standards-based Java Portlet (JSR 286) option.

**Cause**

The application in which you are trying to create the JSR 286 portlet was created using WebCenter Portal's Portal 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 Portal 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.

59.7.1.2 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 JSR 286 Java Portlet wizard.

**Cause**

The Create JSR 286 Java Portlet wizard does not provide the option to add certain features to portlets.

**Solution**

After you create the portlet using the Create JSR 286 Java Portlet wizard, edit the portlet.xml file using the Overview Editor to add advanced functionality. For more information, see Section 59.3, "Developing JSR 286 Java Portlets."
59.7.2 Issues with Testing JSR 286 Portlets

For a list of the issues you may encounter when consuming portlets during the testing phase of portlet development, see Section 63.11, "Troubleshooting Portlets."
This chapter describes how to build Java portlets using the Oracle PDK-Java. Oracle PDK-Java is an Oracle proprietary technology for building Java portlets.

This chapter includes the following topics:

- Section 60.1, "Introduction to PDK-Java Portlets"
- Section 60.2, "Creating a PDK-Java Portlet"
- Section 60.3, "Developing PDK-Java Portlets"
- Section 60.4, "Testing PDK-Java Portlets"
- Section 60.5, "Deploying PDK-Java Portlets"
- Section 60.6, "Creating a Struts Portlet"
- Section 60.7, "Troubleshooting PDK-Java Portlets"

Tip: For building new Java portlets, you should consider using the JSR 286 standard. For more information, see Chapter 59, "Building Standards-Based Java Portlets Using JSR 286."

60.1 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 Portal 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 Portal 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)

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.

Oracle JPDK producers use open standards, such as XML, SOAP, HTTP, or Java EE for deployment, definition, and communication with applications. WebCenter Portal incorporates portlets from a Oracle JPDK producer, which communicates with the consumer application using SOAP over HTTP.

There are several benefits to developing portlets and exposing them through Oracle JPDK producers:
Creating a PDK-Java Portlet

- 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 an Oracle JPDK producer, you must first create a producer that manages your portlets and communicates with WebCenter Portal using SOAP. For more information, see Section 60.5, "Deploying PDK-Java Portlets."

For more information about the PDK-Java, see the Oracle Fusion Middleware Java API Reference for Oracle PDK-Java at:


60.2 Creating a PDK-Java Portlet

WebCenter Portal provides a wizard, available in JDeveloper, for quickly and easily creating the initial framework of your PDK-Java portlets.

This section includes the following topics:

- Section 60.2.1, "How to Create a PDK-Java Portlet"
- Section 60.2.2, "What Happens When You Create a PDK-Java Portlet"

60.2.1 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 JDeveloper, open the portlet producer application under which you want to create your portlet, or create a new portlet producer application.

For information about how to create a portlet producer application, see Section 57.4.1, "Portlet Producer Applications."

Note: If you do not use the WebCenter Portal - Portlet Producer Application template to create the portlet producer application, you must manually add the appropriate portlet building technology scopes to the application.

Applications built using the WebCenter Portal - 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 **Oracle PDK-Java Portlet**, and click **OK**.

4. On the Provider Details page of the Create Oracle PDK-Java Portlet wizard, enter a name for the new producer to contain your portlet. This name must be unique within the project.

   **Tip:** In the PDK-Java, the term provider is used instead of producer. A provider is the same thing as a producer.

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, 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 Portal 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.

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, under **Show Page**, from the **Implementation Style** drop-down 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.
Creating a PDK-Java Portlet

1. 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.

2. 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.

3. 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 57.4.2.1, "View Mode."**

12. To implement Full Screen mode for your portlet, select **Show Details Page** and then select an implementation style as described for Shared 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, **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 57.4.2.2, "Edit Mode."**

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 57.4.2.3, "Edit Defaults Mode."**

16. Click **Next.**

17. On the Additional Modes page, 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 57.4.2.4, "Help Mode."**

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 57.4.2.5, "About Mode."**
19. Click Next.

20. On the Public Portlet Parameters page, 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 60.3.3, "How to Implement Public Parameters."

21. Oracle PDK-Java events are not supported in Portal Framework applications, so on the Public Portlets Events page, click **Finish**.

   For more information about events, see the *Portal Developer’s Guide for Oracle Portal*.

### 60.2.2 What Happens When You Create a PDK-Java Portlet

When you use the Create Oracle PDK-Java Portlet wizard, JDeveloper generates a default implementation of the portlet. Specifically, the following files are created:

- Files for each portlet mode you selected, for example `portletnameEditPage.jsp`.
- `provider.xml` is the producer definition file that contains details of the portlets belonging to the producer.
- `web.xml` is the web deployment descriptor 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 60–1.
60.3 Developing PDK-Java Portlets

When you have built your initial portlet implementation using the Create Oracle PDK-Java Portlet wizard, the next step is to create the code that drives the portlet content and behavior.

You can find the JavaDoc reference for the PDK-Java in the Oracle Fusion Middleware Java API Reference for Oracle PDK-Java at:


This section includes the following topics:

- Section 60.3.1, "General Guidelines for PDK-Java Portlets"
- Section 60.3.2, "How to Add Portlet Modes"
- Section 60.3.3, "How to Implement Public Parameters"
- Section 60.3.4, "How to Implement Private Parameters"
- Section 60.3.5, "How to Use JNDI Variables"
- Section 60.3.6, "How to Access Session Information"
- Section 60.3.7, "How to Enhance PDK-Java Portlet Performance with Caching"
- Section 60.3.8, "How to Manage the Persistence Store for PDK-Java Portlets"
- Section 60.3.9, "How to Move a PDK-Java Portlet Producer"
- Section 60.3.10, "How to Export and Import PDK-Java Portlet Producers at Design Time"

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 the PDK-Java, you can find the examples in:
60.3.1 General Guidelines for PDK-Java Portlets

When you build PDK-Java portlets for use in WebCenter Portal, you should consider the following:

- 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.

60.3.2 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 60.2.1, "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 by updating provider.xml and creating 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 at:


To add a portlet mode:

1. In JDeveloper, 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.
   
   **Tip:** 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:
8. In the Application Navigator, right-click the `provider.xml` file for the provider that owns the portlet and choose **Open**.

   **Tip:** 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:

   ```xml
   <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:

    ```xml
    <helpPage>/htdocs/myprovider/myportlet/myHelpPage.html</helpPage>
    ```

11. Save your changes.

### 60.3.3 How to Implement Public Parameters

PDK-Java and WebCenter Portal Framework provide public and private portlet parameters to enable you to easily write reusable, complex portlets. The Create Oracle PDK-Java Portlet wizard 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`.

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.

**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.

To create a portlet with public parameters:

1. In JDeveloper, 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 60.2.1, "How to Create a PDK-Java Portlet" for basic information about going through the wizard.

5. On the Public Portlet Parameters page, click **Add**.

   This adds a new row to the table of parameters.

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 60–1**.

---

**Example 60–1  provider.xml Sample, Public Parameters**

```xml
<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>
    </portlet>
  </provider>
</providerDefinition>
```
Developing PDK-Java Portlets

Building Java Portlets Using the Oracle PDK-Java

9. In the Application Navigator, right-click the portlet name 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 60–2.

Example 60–2  ShowPage.jsp Sample

```jsp
<%@page contentType="text/html; charset=windows-1252"
    import="oracle.portal.provider.v2.render.PortletRenderRequest"
    import="oracle.portal.provider.v2.ParameterDefinition"%>

<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>
    <tr><td>name</td><td><%= pReq.getParameterValues(name) %></td></tr>
</table>
```

```jsp
9. In the Application Navigator, right-click the portlet name 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 60–2.

Example 60–2  ShowPage.jsp Sample

```jsp
<%@page contentType="text/html; charset=windows-1252"
    import="oracle.portal.provider.v2.render.PortletRenderRequest"
    import="oracle.portal.provider.v2.ParameterDefinition"%>

<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>
    <tr><td>name</td><td><%= pReq.getParameterValues(name) %></td></tr>
</table>
```
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.";
}
}

</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 Portal Framework application. For more information, see Section 63.2.2, "Registering an Oracle PDK-Java Portlet Producer with a WebCenter Portal Framework Application."

13. Add the two portlets to a page in the application. For more information, see Section 63.5, "Adding Portlets to a Page."


The page definition should look similar to Example 60–3. Notice the variables at the page level and the parameters at the portlet level (indicated in bold).

Example 60–3  Page Definition File Sample

```xml
<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">
```
15. Run the application.

16. Enter values in the first portlet and the same values should be displayed in the second portlet.

60.3.4 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 private parameters to jump to another resource of the portlet.

This section includes the following topics:

- Section 60.3.4.1, "About Private Parameters"
- Section 60.3.4.2, "About Portlet URL Types"
- Section 60.3.4.3, "Building Links with the Portlet URL Types"
- Section 60.3.4.4, "Building Forms with the Portlet URL Types"
- Section 60.3.4.5, "Implementing Navigation within a Portlet"
- Section 60.3.4.6, "Restricting Navigation to Resources"

60.3.4.1 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 Portal 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:

```java
HttpPortletRendererUtil.portletParameter(HttpServletRequest request, String param);
```

HttpPortletRendererUtil is in the package oracle.portal.provider.v2.render.http.

For example:

```java
qualParamQ = HttpPortletRendererUtil.portletParameter(r, "q");
```

To fetch the value of a portlet parameter from the incoming request, you can use the following API:

```java
PortletRenderRequest.getQualifiedParameter(String name)
```

PortletRenderRequest is in the package oracle.portal.provider.v2.render.

For example:

```java
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 60.3.4.3, "Building Links with the Portlet URL Types" and Section 60.3.4.4, "Building Forms with the Portlet URL Types."

### 60.3.4.2 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).

### 60.3.4.3 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:

```java
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.constructLink` 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 60.3.4.4, "Building Forms with the Portlet URL Types."

```java
<% 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>
    <!-- Output the HTML content -->
</center>
```

```java
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>

60.3.4.4 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 60.3.4.3, "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 60.3.4.3, "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>
```

60.3.4.5 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 60.3.4.3, "Building Links with the Portlet URL Types" and Section 60.3.4.4, "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 60.3.6, "How to Access Session Information."

The following portlet code comes from the multi-page example in the sample producer of PDK-Java:

```xml
<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.
Developing PDK-Java Portlets

ThesaurusForm.jsp:

```jsp
<%  PortletRenderRequest pRequest = (PortletRenderRequest)
     request.getAttribute(HttpCommonConstants.PORTLET_RENDER_REQUEST);
  String paramNameSubmit = "submit";
  String paramNameQ = "q";
  String qualParamNameQ =
      HttpPortletRendererUtil.portletParameter(pRequest, paramNameQ);
  String qualParamNameNextPage =
      HttpPortletRendererUtil.portletParameter(pRequest, paramNameNextPage);
  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:
    <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:

```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);
NameValuePair[] 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>
    <%= relatedWords[i] %> <%= UrlUtils.constructHTMLLink(
        pRequest,
        UrlUtils.PAGE_LINK,
        "(words related to " + relatedWords[i] + ")",
        '',
        linkParams,
        true,
        true)%>
    </li>
<% }
</ul>
</center>

60.3.4.6 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 60–4 allows resources under /htdocs/multiportlet to be shown. All other resources are restricted.
**Example 60–4  Whitelist Excerpt from the provider.xml File**

```xml
<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 `pageParameterName` 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`.

### 60.3.5 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 topics:

- **Section 60.3.5.1, "Declaring JNDI Variables"**
- **Section 60.3.5.2, "Setting JNDI Variable Values"**
- **Section 60.3.5.3, "Retrieving JNDI Variables"**

#### 60.3.5.1 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:

```xml
<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>
```
Developing PDK-Java Portlets

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 topics:

- Section 60.3.5.1.1, "Variable Types"
- Section 60.3.5.1.2, "Variable Naming Conventions"
- Section 60.3.5.1.3, "Examples"

60.3.5.1.1 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.

60.3.5.1.2 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 60.3.5.3, "Retrieving JNDI Variables."

60.3.5.1.3 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
```

60.3.5.2 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:

   ```xml
   <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):

   ```xml
   <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.
60.3.5.3 Retrieving JNDI Variables

JNDI is a standard Java EE technology. As such, you can access JNDI variables through Java EE APIs. For example:

```java
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 60.3.5.1.2, "Variable Naming Conventions" and are of the form:

```
oracle/portal/provider_service_name/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:

```java
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.

```java
EnvLookup el = new EnvLookup();
Object o = el.lookup("myVariableName");
```

Table 60–1 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>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>oracle/portal/provider/provider_name/autoReload</td>
<td>Boolean auto reload flag, Defaults to true.</td>
</tr>
</tbody>
</table>
60.3.6 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 at:


In the example in this section, session storage is used to count the number of times your portlet has rendered in Shared Screen mode.

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:

   ```jsp
   <%@page contentType="text/html; charset=windows-1252"
   import="oracle.portal.provider.v2.render.PortletRenderRequest"
   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.

   ```jsp
   <%=pSession.getAttribute(key)%> = pSession.getAttribute(key);
   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);
   %>
   Render count in this session: <%=i%> </p>
   ```

   ```jsp
   <%%
   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);
   %>
   Render count in this session: <%=i%> </p>
   ```

   ```jsp
   <%%
   else
   {
   %>
   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>
    </provider>

4. Register your producer with session support. For a reminder on how to register your portlet, see Section 63.2.2, "Registering an Oracle PDK-Java Portlet Producer with a WebCenter Portal Framework Application."

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.

### 60.3.7 How to Enhance PDK-Java Portlet Performance with 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. JSR 286 portlets support expiry-based and validation-based caching.

For more information about caching, see Section 57.4.5, "Portlet Performance."

This section includes the following topics:

- Section 60.3.7.1, "Activating Caching for PDK-Java Portlets"
- Section 60.3.7.2, "Implementing Expiry-Based Caching in PDK-Java Portlets"
- Section 60.3.7.3, "Implementing Validation-Based Caching in PDK-Java Portlets"

#### 60.3.7.1 Activating Caching for PDK-Java Portlets

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.

#### 60.3.7.2 Implementing Expiry-Based Caching in PDK-Java Portlets

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.

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 60.2.1, "How to Create a PDK-Java Portlet," edit the provider.xml file and set the pageExpires property tag of showPage to the
required value. For example, to cache portlet output for 1 minute, set `pageExpires` to 1.

By default the wizard generates a standard and compressed tag for `showPage`. Expand the tag to include a subtag of `pageExpires`:

```xml
<showPage class="oracle.portal.provider.v2.render.http.ResourceRenderer">
  <resourcePath>/htdocs/mycacheportlet/MyCachePortletShowPage.jsp
  <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.

```jsp
<%@page contentType="text/html; charset=windows-1252"
import="oracle.portal.provider.v2.render.PortletRenderRequest"
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.

### 60.3.7.3 Implementing Validation-Based Caching in PDK-Java Portlets

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:

```java
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:

  ```java
  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.
Developing PDK-Java Portlets

method. It also instructs the RenderManager that it is not necessary to call renderBody() to render the portlet body.

```java
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.

```java
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.

60.3.8 How to Manage the Persistence Store for PDK-Java Portlets

The portlet persistence store is used for persisting consumer registration handles and portlet preference data. PDK-Java portlet producers can use one of two types of persistence store: File or Database. The database used for the persistence store must be JDBC-compatible.

For more information, see Section 57.4.4, "Portlet Personalization and Customization."

This section includes the following topics:

- Section 60.3.8.1, "Setting up a Persistence Store for a PDK-Java Producer"
- Section 60.3.8.2, "Migrating a PDK-Java Producer Persistence Store"

60.3.8.1 Setting up a Persistence Store for a PDK-Java Producer

The type of persistence store used for PDK-Java producers is determined by the `preferenceStore` tag in the `provider.xml` file.

`Table 60–2` lists and describes the attributes and parameters used with the `preferenceStore` tag.

If you have installed Oracle PDK-Java, then you can manage the information stored in the persistence 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.

### 60.3.8.2 Migrating a PDK-Java Producer Persistence Store

**Table 60–2 Attributes and Parameters of the preferenceStore Tag**

<table>
<thead>
<tr>
<th>Attribute/Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>class</strong></td>
<td>This required attribute specifies the Java class that defines the location and other details of portlet preferences. For example, a file-based persistence store might use:</td>
</tr>
<tr>
<td></td>
<td><code>&lt;preferenceStore class=&quot;oracle.portal.provider.v2.preference.FilePreferenceStore&quot;&gt;</code></td>
</tr>
<tr>
<td></td>
<td>A database persistence store might use:</td>
</tr>
<tr>
<td></td>
<td><code>&lt;preferenceStore class=&quot;oracle.portal.provider.v2.preference.DBPreferenceStore&quot;&gt;</code></td>
</tr>
<tr>
<td></td>
<td>OmniPortlet uses its own class, for example:</td>
</tr>
<tr>
<td></td>
<td><code>&lt;preferenceStore class=&quot;oracle.webdb.reformlet.ReformletFilePreferenceStore&quot;&gt;</code></td>
</tr>
<tr>
<td><strong>name</strong></td>
<td>This required parameter provides a name for the persistence store. Use any value you choose. For example:</td>
</tr>
<tr>
<td></td>
<td><code>&lt;preferenceStore class=&quot;oracle.portal.provider.v2.preference.DBPreferenceStore&quot;&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;name&gt;MyPDKProducerPreferenceStore&lt;/name&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;preferenceStore&gt;</code></td>
</tr>
<tr>
<td><strong>connection</strong></td>
<td>This required parameter for a database persistence store points to a JNDI connection that connects with a schema containing the portlet persistence store. For example:</td>
</tr>
<tr>
<td></td>
<td><code>&lt;preferenceStore class=&quot;oracle.portal.provider.v2.preference.PCFDBPreferenceStore&quot;&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;name&gt;MyPDKProducerPreferenceStore&lt;/name&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;connection&gt;java:comp/env/jdbc/portletPrefs&lt;/connection&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;preferenceStore&gt;</code></td>
</tr>
<tr>
<td><strong>rootDirectory</strong></td>
<td>This optional parameter specifies the location where the file-based persistence store preferences are stored. When this parameter is not included with the preferenceStore tag, the default file-based persistence store location is used. This is the same folder where the producer’s provider definition file is stored.</td>
</tr>
<tr>
<td><strong>useHashing</strong></td>
<td>This optional parameter is used when specifying a file based persistence store, and accepts the value <code>true</code> or <code>false</code>. When it is set to <code>true</code>, 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:</td>
</tr>
<tr>
<td></td>
<td><code>&lt;preferenceStore class=&quot;oracle.portal.provider.v2.preference.FilePreferenceStore&quot;&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;name&gt;PDKProducerPreferenceStore&lt;/name&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;useHashing&gt;true&lt;/useHashing&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;preferenceStore&gt;</code></td>
</tr>
</tbody>
</table>
The syntax of the migration utility is:

```
java oracle.portal.provider.v2.preference.MigrationTool
   -mode [file | db | filetodb | 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 60.3.8.2.1, "Migration Mode" for more information about this mode.
  - `file` or `db` indicates to run in upgrade mode. See Section 60.3.8.2.2, "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`.

---

**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\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
```
■ `-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`.

■ `-upfixwp` indicates a log file for the operation.

This section includes the following subsections:

■ **Section 60.3.8.2.1, "Migration Mode"**

■ **Section 60.3.8.2.2, "Upgrade Mode"**

**60.3.8.2.1 Migration Mode** Use a migration mode to copy data from a source persistence store to a target persistence store. When the utility is run in this mode, the persistence stores for all the portlet definitions are updated.

Table 60–3 describes the migration modes in which you can run the utility.

<table>
<thead>
<tr>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filetodb</td>
<td>Use when data must be copied from a <code>FilePreferenceStore</code> to a <code>DBPreferenceStore</code>.</td>
</tr>
<tr>
<td>filetofile</td>
<td>Use when data must be copied from one <code>FilePreferenceStore</code> to another <code>FilePreferenceStore</code> that is in a different location.</td>
</tr>
<tr>
<td>dbtotofile</td>
<td>Use when data must be copied from a <code>DBPreferenceStore</code> to a <code>FilePreferenceStore</code>.</td>
</tr>
<tr>
<td>dbtodb</td>
<td>Use when data must be copied from one <code>DBPreferenceStore</code> to another <code>DBPreferenceStore</code> that is based on a different database table.</td>
</tr>
</tbody>
</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 the *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 persistence stores involved in the upgrade or migration process. These options must correspond to the tags you specify in `provider.xml` to describe the persistence stores.

Properties beginning with the prefix `-pref1` correspond to properties of the source persistence store (in an upgrade mode, this is the only persistence store). For example, specifying `-pref1UseHashing true`
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 persistence 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 60–5  PDK-Java Migration Utility Command Line, Migration**

```java
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
```

### 60.3.8.2.2 Upgrade Mode

Use an upgrade mode to upgrade data in place, and to modify existing locale-specific preferences in the persistence store so that the naming format used is compatible with the current version of WebCenter Portal and a given localePersonalizationLevel setting.

Table 60–4 describes the upgrade modes in which you can run the utility.

<table>
<thead>
<tr>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>file</td>
<td>Use when data in a FilePreferenceStore must be upgraded.</td>
</tr>
<tr>
<td>db</td>
<td>Use when data in a DBPreferenceStore must be upgraded.</td>
</tr>
</tbody>
</table>

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 60–6  PDK-Java Migration Utility Command Line, Upgrade

```java
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
```

60.3.9 How to Move a PDK-Java Portlet Producer

In some situations, you may want to move a portlet producer. For example, you may need to move it to a new server.

To move a PDK-Java portlet producer:

1. Install the new producer.
2. Make the persistence store of the original producer available to the new producer by performing one of the following tasks:
   - Migrate the persistence store using the Preference Store Migration and Upgrade Utilities (see Section 60.3.8.2, "Migrating a PDK-Java Producer Persistence Store" for more information), or
   - Configure the new producer to use the same persistence store as the original producer. If the database persistence 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.

60.3.10 How to Export and Import PDK-Java Portlet Producers at Design Time

You can export and import portlet producers at design time. For more information, see Section 59.3.15, "How to Export and Import Portlet Producers at Design Time."

60.4 Testing PDK-Java Portlets

Before making your portlets available in a production environment, it is highly advisable to test them first to make sure that they behave as expected.

This section includes the following topics:

- Section 60.4.1, "How to Test PDK-Java Portlet Producer Applications on Integrated WebLogic Server"
- Section 60.4.2, "What Happens When You Test PDK-Java Portlet Producer Applications on Integrated WebLogic Server"
- Section 60.4.3, "How to Deploy a PDK-Java Portlet Producer to the Integrated WebLogic Server"
- Section 60.4.4, "How to Test Portlet Personalization"
60.4.1 How to Test PDK-Java Portlet Producer Applications on Integrated WebLogic Server

The Integrated WebLogic Server (Integrated WLS) provides a quick and easy way of testing your portlets because it is preconfigured so that you can run applications within JDeveloper without needing to create deployment profiles.

To test a PDK-Java portlet on Integrated WLS:

1. In JDeveloper, open the portlet producer application that owns the portlet that you want to test.
2. Expand the portlet project (for example, Portlets).
3. 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.
4. Check the IntegratedWebLogicServer - Log window to monitor the deployment progress. The log shows the URL of the application test page, as shown in Figure 60–2. The format of the URL is http://host:port/application_name-Portlets-context-root/index.jsp.

![Figure 60–2 DefaultServer - Log](image)

5. The PDK-Java Application Test page displays in a browser window, as shown in Figure 60–3.

![Figure 60–3 Portlet Producer Application Test Page](image)

6. Click the link underneath Service Name. Your browser should open with a page similar to the one shown in Figure 60–4. The URL of this page is the one required to register the producer with another application.
7. 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 60–4.

8. 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 the test page in your browser.

9. When the producer is successfully running, you should register it 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 of PDK-Java portlets, follow the instructions provided in Section 63.2.2, “Registering an Oracle PDK-Java Portlet Producer with a WebCenter Portal Framework Application.”

To add your portlets to a page, follow the instructions provided in Section 63.5, "Adding Portlets to a Page.”
10. To stop an Integrated WLS instance, in the Run Manager tab, select **DefaultServer** and click the red **Stop** icon. When the instance stops, the application is undeployed, and therefore, comes unavailable.

**Tip:** You can also stop an Integrated WLS instance by clicking the red **Stop** icon in the IntegratedWebLogicServer - Log window and selecting **DefaultServer**.

### 60.4.2 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`.

### 60.4.3 How to Deploy a PDK-Java Portlet Producer to the Integrated WebLogic Server

When you run a PDK-Java portlet producer application on the Integrated WLS instance, when the instance stops, the application is undeployed and therefore becomes unavailable. For a more persistent testing scenario, you can deploy your portlet producer application to the Integrated WLS so that it is always available while the Default Server is running.

If you choose this method, then you must first create deployment profiles, as described in Section 61.4, "Creating a WAR Deployment Profile." If you deploy your application to the Integrated WLS, then the Deployment Configuration dialog displays to enable you to configure and customize deployment settings. The file system MDS repository precreated by JDeveloper displays in the **Repository Name** field.

For information about deploying and running an application on the Integrated WLS, see Section 7.2, "Deploying a Portal Framework Application to the Integrated WebLogic Server."

### 60.4.4 How to Test 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 74.12, "Configuring Basic Authentication for Testing Portlet Personalization."

### 60.5 Deploying PDK-Java Portlets

You can deploy your portlet producer to any WebLogic Managed Server that is configured to support WebCenter Portal portlet producers. For a list of prerequisites that you must complete before deployment, see Section 61.1, "Introduction to
Deploying Portlet Producers.

The first step of deploying a PDK-Java portlet producer is to create a WAR deployment profile to indicate how the producer and its associated files should be packaged. For more information, see Section 61.4, "Creating a WAR Deployment Profile."

You can deploy PDK-Java portlets only as EAR files, so after creating the WAR deployment profile, you must also create an EAR deployment profile to package the entire application. For more information, see Section 7.3.2.2, "Creating Deployment Profiles."

---

**Note:** While creating the EAR deployment profile, ensure that the WAR file is included in the application’s EAR file.

---

After you have created the EAR deployment profile, you can deploy your portlet producer to the managed server. For more information, see Section 61.5, "Deploying a Portlet Producer to a WebLogic Managed Server."

---

**Note:** When deploying a PDK-Java portlet producer, you will not see the Select deployment type dialog, as the producer does not contain JSR 286 portlets.

---

When you deploy a PDK-Java portlet producer to a WebLogic Managed Server, the configuration settings described in Section 60.4.2, "What Happens When You Test PDK-Java Portlet Producer Applications on Integrated WebLogic Server" are added to the application EAR file at design time.

### 60.6 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):

```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 60–5 and Figure 60–6 show how you submit a blog and save a blog entry.

**Figure 60–5  Submitting a Blog**

```
/prepareNewBlog
success
```

`View/enterNewBlog.jsp`

**Figure 60–6  Saving a Blog Entry**

```
/saveNewBlog
success
```

`View/newBlogConfirmation.jsp`
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:

```xml
<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 topics:

- Section 60.6.1, "How to Create a New Flow and View to Host the Portlet Actions"
- Section 60.6.2, "How to Create the New JSPs"
- Section 60.6.3, "How to Create a Portlet"
- Section 60.6.4, "How to Extend the Portlet to Add Business Logic"
- Section 60.6.5, "How to Register the Producer"

### 60.6.1 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.

```xml
<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/portal/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.

### 60.6.2 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.
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 60.2.1, "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 synchronizer token mechanism is used or <actionInSession> is set to true), then enable session handling:

   ```xml
   <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:

   ```xml
   <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:

   ```xml
   <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:

   ```xml
   <showPage class="oracle.portal.provider.v2.render.http.StrutsRenderer">
   <defaultAction>/portal/prepareNewBlog.do</defaultAction>
   </showPage>
   ```

   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.
60.6.4 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.

60.6.5 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 63.2.2, "Registering an Oracle PDK-Java Portlet Producer with a WebCenter Portal Framework Application."

60.7 Troubleshooting PDK-Java Portlets
This section provides information to assist you in troubleshooting problems you may encounter while creating and testing PDK-Java portlets.

This section includes the following topics:
- Section 60.7.1, "Cannot Access the Create Portlet Wizard"
- Section 60.7.2, "Cannot Add the Portlet Functionality that I Want to the Portlet"

60.7.1 Cannot Access the Create Portlet Wizard

**Problem**
In the New Gallery, I cannot find the Oracle PDK-Java Portlet option.

**Cause**
The application in which you are trying to create the PDK-Java portlet was created using WebCenter Portal's Portal 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 Portal 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.

60.7.2 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 Oracle PDK-Java Portlet wizard.

**Cause**
The Create Oracle PDK-Java Portlet wizard does not provide the option to add certain features to portlets.
Solution

After you create the portlet using the Create Oracle PDK-Java Portlet wizard, edit the portlet to add advanced functionality. For more information, see Section 60.3, "Developing PDK-Java Portlets."
This chapter describes how to use Oracle JDeveloper to deploy portlet producers to an Oracle WebLogic Managed Server configured to test application deployment or host applications for production.

This chapter includes the following topics:

- **Section 61.1, "Introduction to Deploying Portlet Producers"**
- **Section 61.2, "Creating and Provisioning a WebLogic Managed Server for Deploying Portlet Producers"**
- **Section 61.3, "Creating a Connection to a Portlet Producer WebLogic Managed Server"**
- **Section 61.4, "Creating a WAR Deployment Profile"**
- **Section 61.5, "Deploying a Portlet Producer to a WebLogic Managed Server"**

For information about how to deploy producers using Fusion Middleware Control, Oracle WebLogic Server Administration Console, or WLST, see the "Deploying Portlet Producer Applications" section in *Administering Oracle WebCenter Portal*.

For more general information about deployment of applications, see Chapter 7, "Deploying and Testing Your Portal Framework Application."

### 61.1 Introduction to Deploying Portlet Producers

You can deploy your portlet producers directly from JDeveloper to any Oracle WebLogic Managed Server that is configured to support WebCenter Portal portlet producers.

A WebLogic Managed Server resides outside of 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.

The steps for deploying a portlet producer to a WebLogic Managed Server are:

1. Create and provision a WebLogic Managed Server instance that contains all the required shared libraries. For more information, see Section 61.2, "Creating and Provisioning a WebLogic Managed Server for Deploying Portlet Producers."

2. As the WebLogic Managed Server resides outside JDeveloper, you must create a connection to it. For more information, see Section 61.3, "Creating a Connection to
a Portlet Producer WebLogic Managed Server.”

3. Create an application WAR deployment profile. For more information, see Section 61.4, "Creating a WAR Deployment Profile."

4. Deploy the portlet producer to the WebLogic Managed Server. For more information, see Section 61.5, "Deploying a Portlet Producer to a WebLogic Managed Server."

**Note:** Before you deploy the portlet producer, you must remove the test-all application role if you enabled the automatic grants feature in the Configure ADF Security wizards (see Section 74.3.1, "Configuring ADF Security Settings"). 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 "How to Remove the test-all Role from the Application Policy Store" section in the Fusion Developer's Guide for Oracle Application Development Framework.

### 61.2 Creating and Provisioning a WebLogic Managed Server for Deploying Portlet Producers

Before you can deploy your portlet producer to a WebLogic Managed Server, you must create a WebLogic Managed Server based on the Custom Services Producer template, oracle.wc_custom_services_producer_template_11.1.1.jar, which contains all the required shared libraries.

Using this template creates a Custom Services Producer managed server named WC_CustomServicesProducer and targets all the necessary resources to it. It also adds the following JDBC data sources:

- mds-CustomServicesProducerDS (for accessing the MDS schema)
- WebCenter-CustomServicesProducerDS (for accessing the WebCenter schema)
- Activities-CustomServicesProducerDS (for accessing the Activities schema)
- Portlet-CustomServicesProducerDS (for accessing the Portlet schema)

To create a Custom Services Producer managed server:

1. Start the Configuration Wizard in graphical mode.

   For more information see the "Starting the Configuration Wizard in Graphical Mode" section in Creating Domains Using the Configuration Wizard.

2. On the Welcome page, select **Extend an existing WebLogic domain**, and then click Next.

**Note:** If you prefer, you can create a new domain for your managed server. For more information, see the "Creating a New Domain" section in Installation Guide for Oracle WebCenter Portal.

3. On the Select a WebLogic Domain Directory page, select a valid domain directory, and then click Next.
Creating a Connection to a Portlet Producer WebLogic Managed Server

A domain directory contains a config directory, which contains a config.xml file. In the navigation tree, these directories are indicated by a blue folder icon.

4. On the Select Extension Source page, select **Extend my domain using an existing extension template**, then specify the path of the template as follows:
   - On UNIX operating systems:
     
     
     WebCenter_ORACLE_HOME/common/templates/applications/oracle.wc_custom_services_producer_template_11.1.1.jar
   - On Windows operating systems:
     
     WebCenter_ORACLE_HOME\common\templates\applications\oracle.wc_custom_services_producer_template_11.1.1.jar

5. Click **Next**.

6. For information about the remaining steps in this wizard, see the "Extending a WebLogic Domain in Graphical Mode" section in Creating Domains Using the Configuration Wizard.

---

**Note:** Although portlet producers only require access to the MDS schema, it is advisable to configure all of the JDBC data sources created for the managed server to avoid warnings later.

### 61.3 Creating a Connection to a Portlet Producer WebLogic Managed Server

Before you can use JDeveloper to deploy your portlet producer to a WebLogic Managed Server instance that resides outside JDeveloper, you must create a connection to the domain that contains the managed server instance where you will deploy the portlet producer. Before you create a connection to a WebLogic Managed Server instance, ensure that the target managed server is up and running and has the required libraries.

To create a WebLogic Managed Server connection for portlet producers:

1. In JDeveloper, from the File menu, choose **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_MyPortletProd**), 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 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.
9. Click Finish.

### 61.4 Creating a WAR Deployment Profile

To deploy portlet producers to a managed server that resides outside JDeveloper, you must create a deployment profile to indicate how the producer and its associated files should be packaged. To do this, you must create a Web Application Archive (WAR) file.

To create a WAR deployment profile:

1. In the Application Navigator, right-click the project folder that contains your portlet producer, for example, `Portlets`, and choose New.
2. In the New Gallery, expand General, select Deployment Profiles, then WAR File, and click OK.
3. In the Create Deployment Profile -- WAR File dialog, enter a name for your deployment profile and click OK.
4. In the Edit WAR Deployment Profile Properties dialog, select Specify Java EE Web Context Root, enter a context root, then click OK.
5. In the Project Properties dialog, under Deployment Profiles, select the WAR file you just created and click OK.

### 61.5 Deploying a Portlet Producer to a WebLogic Managed Server

After you have created the deployment profile, you can deploy your portlet producer to the managed server.

To deploy a portlet producer to a WebLogic Managed Server:

1. In the Application Navigator, right-click the project folder, for example, `Portlets`, and choose Deploy, then the deployment profile (WAR file) that you created in Section 61.4, "Creating a WAR Deployment Profile."
2. In the Deploy dialog, on the Deployment Action page, select Deploy to Application Server, then click Next.
3. On the Select Server page, select the connection to the Managed Server, then click Next.
4. On the WebLogic Options page, select Deploy to selected instances in the domain, and select the managed server to which you want to deploy the portlet producer, and then click Finish.
5. If your application contains JSR 286 portlets, the Select deployment type dialog displays (Figure 61–1). Select Yes and then click OK to add the configuration required to expose this application as a WSRP service.
Tip: This dialog does not display if, during a previous deployment, you selected Do not show this dialog again.

6. 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.

Note: If you are deploying your portlet producer in a cluster environment, you will receive the following warning:

WARNING: oracle.webcenter.lifecycle.exception.LifecycleLockedException: A lock exists that prevents 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.

7. After the deployment has finished, check the log for any errors that may have occurred during deployment.

8. To confirm that your application has been successfully deployed, display the producer test page in your browser. The URL for the test page is:
http://host:port/context-root/info

where:
- host is the server to which you deployed the application.
- port is the HTTP Listener port. Typically this is 7101.
context-root is the application’s context root as defined in the Edit WAR Deployment Profile Properties (see Section 61.4, "Creating a WAR Deployment Profile").

Figure 61–2  WSRP Producer Test Page

WSRP Producer Test Page

Your WSRP Producer Contains the Following Portlets:

Portlet Name (Minimum WSRP Version)

• Portlet 1.0

Container Configuration

Persistent Store Type: Consumer

Value obtained from environment entry java.com.pren/oracle/portal/wps/pserver/persistentStore

Container Version

Implementation version: 11.1.1.8.0, Label: WORKPLACE_11.1.1.8.0_GENERIC_130305.1533

WSDL URLs

WSRF v1 WSDL
WSRP v2 WSDL

SOAP Monitor

SOAP Monitor
Creating Pagelets with Pagelet Producer

This chapter describes Oracle WebCenter Portal's Pagelet Producer (previously known as Oracle WebCenter Ensemble), which provides a collection of useful tools and features that facilitate dynamic pagelet development.

This chapter includes the following sections:

- Section 62.1, "About Pagelet Producer"
- Section 62.2, "Configuring Pagelet Producer Settings"
- Section 62.3, "Creating Pagelet Producer Objects"
- Section 62.4, "Working with Pagelet Chrome for WSRP and Oracle JPDK Portlets"
- Section 62.5, "Working with OpenSocial Gadgets"
- Section 62.6, "Building Custom Pagelets Using Pagelet Producer"
- Section 62.7, "Using Pagelets in Web Applications"
- Section 62.8, "Examples and Advanced Topics"
- Section 62.9, "Troubleshooting Pagelets"

## 62.1 About Pagelet Producer

This section provides an introduction to Pagelet Producer concepts and features. It describes how you can use Pagelet Producer, its underlying architecture and components, and introduces key concepts.

This section includes the following topics:

- Section 62.1.1, "Overview"
- Section 62.1.2, "Pagelet Producer Architecture"
- Section 62.1.3, "Requirements"

### 62.1.1 Overview

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 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.

Pagelet Producer provides a collection of useful tools and features that facilitate dynamic pagelet development, and lets you leverage investments in existing Web
infrastructure by making it usable in new ways. Pagelet Producer can be used to consume:

- Web UI from applications that do not expose integration APIs or portlets out of the box
- Standard portlets such as WSRP (JSR-168 or JSR-286), JPDK, CSP, and WebParts

It can also deliver Web UI to WebCenter Portal, Sites, or pretty much any HTML page with the added benefits of role-based access control to Pagelet Producer resources, and reverse proxying of Web resources with HTML markup parsing, clipping, injection and event handling (inter-pagelet communication).

Pagelet Producer provides:

- Tools for consuming existing Web UI and re-using it as portable components called pagelets
- Alternatives for integrating non-standard/non-compliant containers and components that do not support portlet standards. For example, you can:
  - Capture functionality from Web applications that do not expose portlets
  - Make existing WSRP, JPDK, or Content Syndication Protocol (CSP) portlets available for use in a non-portal setting
-Injectors and parsers to modify HTML markup
- A JavaScript framework for inter-pagelet communication and event handling

62.1.2 Pagelet Producer Architecture

This section describes the Pagelet Producer architecture and key components...

This section includes the following subsections:

- Section 62.1.2.1, "Pagelet Producer Architecture and Components"
- Section 62.1.2.2, "Pagelet Producer Key Concepts"
- Section 62.1.2.3, "Pagelet Producer Console"
- Section 62.1.2.4, "Using HTTP and Content Syndication Protocol"
- Section 62.1.2.5, "Pagelet Producer Proxy"

62.1.2.1 Pagelet Producer Architecture and Components

Figure 62–1 shows a high-level view of the Pagelet Producer components and their interactions. Figure 62–1 also reflects the way in which the pagelet request is being handled in Pagelet Producer. The request is received by the Proxy Servlet that maps incoming requests to the resource. Next, the request is authenticated. If the user can access the requested resource, pagelet processing starts by requesting HTML markup from the backend system. After the markup is received it is modified using any defined clippers, parsers, and so forth.
Figure 62–1 Pagelet Producer - High-Level Architecture

Figure 62–2 provides a different view of component interaction:

Figure 62–2 Pagelet Producer - Interaction Flow

A: A Pagelet Producer application runs on a J2EE application server. Note that Pagelet Producer does not require the WebCenter schema, but it does require an MDS Instance (typically shared with WebCenter).

B: User calls a WebCenter Portal page that contains a pagelet.

C: WebCenter Portal returns the page, plus the JavaScript that will call Pagelet Producer to render the pagelet.

D: The JavaScript makes a render call to Pagelet Producer.

E: The Pagelet Producer can call external (E1) or internal (E2) Web applications as sources of HTML content. Pagelet Producer transforms the content (i.e., rewrites the URL) and returns it to the browser for insertion into the page. Pagelet Producer itself can be externally accessible (i.e., in the DMZ).
62.1.2.2 Pagelet Producer Key Concepts
The following key concepts are useful when working with Pagelet Producer:

■ **Resources** are core objects used to register applications within Pagelet Producer, including stand-alone Web applications, Pagelet Producers and OpenSocial containers. Creating a resource allows the proxy to map internal applications to external URLs, manage authentication, and transform applications. Registering a Web application as a Pagelet Producer resource allows you to do the following:
  ■ Proxy internal Web applications to external addresses
  ■ Manage authentication, both at the proxy level and at the resource level
  ■ Transform proxied Web applications, including URL rewriting
For more information about creating resources, see Section 62.3.1, "Creating Resources."

■ **Pagelets** are sub-components of a Web page accessed through Pagelet Producer that can be injected into any proxied application. Any application on a Pagelet Producer resource that returns markup can be registered as a pagelet, which can then be displayed in a portal or Framework application, or any other Web application.

Any HTML fragment can be a pagelet. Pagelet developers can create pagelets that are parameterized and configurable, dynamically interact with other pagelets, and respond to user input using Asynchronous Javascript and XML (AJAX) patterns.

For more information about creating pagelets, see Section 62.3.2, "Creating Pagelets."

Pagelet Producer registration is dynamic. Additions and updates to existing producers are immediately available. In most cases, it is not necessary to restart the portal or Framework application or the managed server.

62.1.2.3 Pagelet Producer Console
The **Pagelet Producer Console** is a browser-based administration tool used to create and manage the various objects in your Pagelet Producer deployment. From the Console you can register Web applications as resources, create pagelets, manage proxy and transformation settings, and more.

The Pagelet Producer Console is accessible from any Web browser at the following URL:

http://<host>:<port>/pagelets/admin

The Pagelet Producer Console can also be launched in accessibility mode at:

http://<host>:<port>/pagelets/admin/accessible

During runtime, the Pagelet Producer Console is also accessible from the WebCenter Portal Shared Assets tab.

---

**Note:** Pagelet Producer Console supports the standard administration languages and Dutch only. If you configure the browser language to something other than one of these languages, it will revert to the language defined for the current server.

---

For more information about using the Pagelet Producer Console to configure Pagelet Producer and create objects, see Section 62.2, "Configuring Pagelet Producer Settings"
and Section 62.3, "Creating Pagelet Producer Objects." For information about using the Pagelet Producer Console to register producers and migrate pagelet data, see the "Managing the Pagelet Producer" chapter in Administering Oracle WebCenter Portal.

62.1.2.4 Using HTTP and Content Syndication Protocol

HTTP is a protocol used mostly for transferring Web page content and XML between a server and a client. Content Syndication Protocol (CSP) is a platform-independent protocol based on the open standard of HTTP 1.1 that extends HTTP defining the syntax of communication between Pagelet Producer and external CSP resources.

Services on external resources communicate with Pagelet Producer using CSP. For example, when a browser requests a page, 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 CSP headers sent by Pagelet Producer and sends back the appropriate HTML. Pagelet Producer inserts the HTML into the page markup. Any images stored in the Image Service are retrieved and displayed by the browser.

This section contains the following subsections:

- Section 62.1.2.4.1, "HTTP"
- Section 62.1.2.4.2, "Content Syndication Protocol"

62.1.2.4.1 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 (for example, what language the content is in, or how long the browser should cache it). The request and response each contain specific information as described below. 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.xmlns.oracle.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 more information.

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 62.6.3, "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.

### 62.1.2.4.2 Content Syndication Protocol

Content Syndication Protocol (CSP) extends HTTP and defines proprietary headers to pass settings between Pagelet Producer and external CSP resources (specifically, Oracle WebCenter Interaction portlets). CSP outlines how these services use HTTP to communicate and modify settings. The latest version of the CSP protocol is 1.4, and is available from:

http://docs.oracle.com/cd/E13158_01/alui/wci/docs103/devguide/references/CSP1.4.pdf

The custom CSP headers used to communicate system and user configuration variables (see Table 62–1) can also be used by pagelets.

<table>
<thead>
<tr>
<th>Table 62–1</th>
<th>Pagelet Producer Headers</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Header Name</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>User ID</td>
<td>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.</td>
</tr>
<tr>
<td>User Name</td>
<td>The name of the logged in user. The user’s name can be used to personalize display or pre-fill form fields.</td>
</tr>
<tr>
<td>Image Service URL</td>
<td>The URL to the root virtual directory of the Image Service in the user’s implementation of Pagelet Producer. This location should be used for all static images used in services.</td>
</tr>
<tr>
<td>Stylesheet URL</td>
<td>The URL to the current user’s style sheet. In each implementation of 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 Pagelet Producer.</td>
</tr>
<tr>
<td>Pagelet ID</td>
<td>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.</td>
</tr>
<tr>
<td>Host Page URL</td>
<td>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.</td>
</tr>
</tbody>
</table>
62.1.2.5 Pagelet Producer Proxy

Pagelet Producer acts as a proxy server, brokering transactions between client computers and external resources. This configuration is typically used to serve content to clients that would otherwise be unable to access the external resource, but it can also be used to impose additional security restrictions on the client, including the use of policies. The proxy hides the external resource so that the content appears to come directly from the proxy server.

This architecture makes 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 Pagelet Producer can connect to the external resource, users can view the content, even if they cannot access it directly. To the browser, 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 Pagelet Producer. To the user, the content appears to come from 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**: Pagelet Producer intercepts request from pagelets, which allows the response to be modified dynamically, using information from MDS, the request, or the response.

- **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.

---

**WARNING:** The proxy is a powerful feature and can compromise security if incorrectly configured. Allowing direct access to an external resource that hosts unprotected private content could create a dangerous security gap.

---

- **Performance**: 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.

Note that all URLs to other proxied CSP or Web resources will be proxied by default unless specified in the **Do not proxy hosts** field. For more information about configuring proxying, see Section 62.2.3, "Configuring Proxy Settings."

Keep the following warnings and best practices in mind when implementing services that use the proxy:

- **URL transformation**: Pagelet Producer must transform code so that proxied URLs open correctly. Before Pagelet Producer sends a response, it parses the HTML and looks for URLs containing the HTTP Proxy Server URL field prefix (see Section 62.2.3, "Configuring Proxy Settings"). 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 parsers and Web injectors, see Section 62.6.2, "Modifying Pagelet Functionality at Runtime."

- **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 that for .NET, there is no need to encode the redirect URL; this is handled automatically by the back end.

### 62.1.3 Requirements

Install WebCenter Portal following the instructions in the *Oracle Fusion Middleware Installation Guide for Oracle WebCenter*. Make sure that all dependent components are also installed and configured, including WebCenter Content Server, the discussions server, and Pagelet Producer.

By default, the Services Producer is automatically deployed to the WC_Portlet managed server in the WebCenter Portal domain. If the WC_Portlet managed server was not created during installation or there is a requirement to run the Services Producer separately, there is a custom Services Producer template that can be used to create the WC_CustomServicesProducer managed server. This managed server contains the Oracle WebCenter Services Producer component as well as pre-configured JDBC data sources for accessing the WebCenter Portal, Activities, and MDS schemas. For more information, see the "Oracle WebCenter Portal: Custom Services Producer Template" section in the *Domain Template Reference*.

### 62.2 Configuring Pagelet Producer Settings

This section explains important configuration settings that affect all Pagelet Producer resources and how to set them using the Settings pages of the Pagelet Producer Console. For an introduction to the Pagelet Producer Console, see Section 62.1.2.3, "Pagelet Producer Console."

The following pages are included in the Settings section:

- Section 62.2.1, "Configuring a WCI Data Source"
- Section 62.2.2, "Configuring Logging Settings"
- Section 62.2.3, "Configuring Proxy Settings"
- Section 62.2.4, "Configuring Transform Settings"
- Section 62.2.5, "Configuring CSP Settings"
- Section 62.2.6, "Configuring Kerberos Settings"
- Section 62.2.7, "Configuring OpenSocial Settings"

### 62.2.1 Configuring a WCI Data Source

A CSP login token is required to consume WCI resources, and a data source is required for that login token to be propagated from Pagelet Producer to CSP portlets. Pagelet Producer generates a login token that is sent to all remote portlets to propagate user identity and validate incoming requests. Follow the steps in the "Configuring JDBC Data Sources" section in *Configuring and Managing JDBC Data Sources for Oracle*. 

---

**Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper**
WebLogic Server to configure the data source. Note that the data source must be named wcidb.

62.2.2 Configuring Logging Settings

The Logging page lets you set logging levels for Pagelet Producer components.

Figure 62–3  Pagelet Producer Console: Settings - Logging

To enable logging for any component, choose one of the following logging levels:

- Severe
- Warning
- Info
- Config
- Fine
- Finer
- Finest

Logging messages can be viewed in the managed server log files. In Oracle WebLogic Server, this would be domain_home/servers/managed_server_name/managed_server_name.log and managed_server_name-diagnostic.log.

62.2.3 Configuring Proxy Settings

The Proxy page lets you define a list of URLs that will not be proxied, and set the URL, user name, and password for the HTTP proxy if required.

You must restart Pagelet Producer after changing the proxy settings.

Note: If an HTTP proxy is required for external connections in the deployment environment, initial server startup will report connection errors. These errors are not an indication of problems with the environment; they indicate that some external OpenSocial libraries cannot be loaded remotely. The errors can be resolved by configuring the proxy settings as described above and restarting the server.
Configuring Pagelet Producer Settings

Note for Oracle WebLogic Server deployments: The HTTP Proxy Server URL entered on the Proxy page is applied to all applications running on the server that hosts Pagelet Producer (on Oracle WebLogic Server, this setting is applied to all applications running on the WC_Portlet managed server). Oracle WebLogic Server users should pay particular attention to this setting and make sure that the WLS Administrative Server host and all clustered managed servers are included in the "Do not proxy hosts" list.

Figure 62–4  Pagelet Producer Console: Settings - Proxy

The Proxy page contains the following settings:

- The Do not proxy hosts field lets you define a list of semicolon-separated list of URLs that will not be proxied (wildcards are allowed). For Oracle WebLogic Server deployments, be sure to include the WLS Administrative Server host and all clustered managed servers in this list.

- The HTTP Proxy Server Password field lets you set the password for the proxy server.

- The HTTP Proxy Server URL field lets you define the proxy server URL.

- The HTTP Proxy Server Username field lets you define the username with which to access the proxy server.

62.2.4 Configuring Transform Settings

The Transform page lets you enter the path to the credential vault provider and configure secure and insecure ports for Pagelet Producer. This page also lets you choose whether or not to log pre- and post- transformation content.
Configuring Pagelet Producer Settings

Creating Pagelets with Pagelet Producer

Figure 62–5  Pagelet Producer Console: Settings - Transform

The Transform page contains the following fields:

- The **Credential Vault Plugin Path** field lets you define a path to the credential vault, where plug-ins are stored.
- The **Nonsecure Port** field lets you define a non-secure port for Pagelet Producer.
- The **Secure Port** field lets you define a non-secure port for Pagelet Producer.
- The **Trace Transformation** option, when checked, turns on logging for pre- and post-transformation content. Note that this option, although useful for debugging, should not be turned on in production environments.

62.2.5  Configuring CSP Settings

CSP is a platform-independent protocol based on the open standard of HTTP 1.1. CSP defines the syntax of communication between Pagelet Producer and external resources. It also defines custom headers as well as outlines how services use HTTP to communicate and modify settings.

The CSP page lets you configure the Oracle WebCenter Interaction Credential Mapper, SOAP API service, and image service.

**Note:** This page may be ignored if Oracle WebCenter Interaction is not present in your deployment. These settings are used for backwards compatibility with CSP portlets written for Oracle WebCenter Interaction.

62.2.6  Configuring Kerberos Settings

The Kerberos page is used to define the paths to the Kerberos configuration files (java.security.krb5.conf and java.security.auth.login.conf). These two configuration files are needed to configure the Kerberos realm and KDC to instruct HTTPClient where to retrieve the Kerberos Service ticket.

62.2.7  Configuring OpenSocial Settings

The OpenSocial page lets you configure Pagelet Producer for use with OpenSocial gadgets.
The OpenSocial page includes the following settings:

- The **Cache** option enables Pagelet Producer internal caching for OpenSocial gadgets.

- The **Context** field defines the context to which Pagelet Producer is deployed. This value should be left at the default setting ("pagelets") unless a change is required by the OpenSocial container. For details on changing the context, see the "Redeploying Pagelet Producer to a Different Context" section in *Administering Oracle WebCenter Portal*.

- The **Debug** option enables debugging for OpenSocial gadgets (disables JavaScript obfuscation).

- The **Host** field should contain the fully-qualified domain name of Pagelet Producer host. This value should be retrieved automatically from the environment; if it is not, restart the server to pick up the correct configuration settings. As with the Context setting, this value should be left at the default unless a change is required by the OpenSocial container.

- The **SSL** option enables SSL for OpenSocial gadgets.

---

**Note:** You must also configure the secure (HTTPS) and nonsecure (HTTP) ports before importing OpenSocial gadgets. For details on these settings, see Section 62.2.4, "Configuring Transform Settings."

### 62.3 Creating Pagelet Producer Objects

This section describes how to create and configure the various objects in a Pagelet Producer deployment, including resources, pagelets, Web injectors, parsers and hosted files.

As explained in Section 62.1.2.3, "Pagelet Producer Console," the Pagelet Producer Console is a browser-based administration tool used to create and manage the various objects in a Pagelet Producer deployment. This section describes the settings required for each type of object:

- Section 62.3.1, "Creating Resources"
- Section 62.3.2, "Creating Pagelets"
- Section 62.3.3, "Creating Web Injectors"
- Section 62.3.4, "Creating Custom Parsers"
- Section 62.3.5, "Creating Hosted Files"
62.3.1 Creating Resources

**Resources** are applications registered with Pagelet Producer. Registering an application as a resource allows the proxy to map internal applications to external URLs, manage authentication, and transform application functionality.

To create a new resource, use the steps below:

1. Select the **Resources** option from the dropdown list in the Pagelet Producer Console navigator.
2. Click on any existing resource and click the **Create** icon in the navigator toolbar. (This button is only enabled when you have selected an object type that can be created.)

![Figure 62–7  Pagelet Producer Console - Create Resource](image)

3. In the **Select Producer Type** dialog box, choose the type of producer that the resource will be configured to proxy from the dropdown list:
   - **Web**: Any standard Web application
   - **CSP**: CSP portlet provider application (for use with Oracle WebCenter Interaction)
   - **WSRP/JPDK**: WSRP or Oracle JPDK portlet producer (before you create a resource of this type, register the associated producer as described in the "Managing the Pagelet Producer" chapter in *Administering Oracle WebCenter Portal*).
   - **OpenSocial**: OpenSocial Container

![Figure 62–8  Select Producer Type Dialog](image)

4. Click **OK**. An entry called "<new>" will be added to the list of resources. This new resource will include the necessary configuration pages for the producer type chosen.

5. Configure the resource in the Pagelet Producer Console. The required configuration pages differ based on the type of producer. To save your changes at any time, click the **Save** icon in the navigator toolbar.

   - Section 62.3.1.1, "Configuration Pages: Web and CSP Resources"
Once the resource is defined, create the pagelets and other objects within the resource. For more information, refer to the following sections:

- Section 62.3.2, "Creating Pagelets"
- Section 62.3.3, "Creating Web Injectors"
- Section 62.3.4, "Creating Custom Parsers"
- Section 62.3.5, "Creating Hosted Files"

62.3.1.1 Configuration Pages: Web and CSP Resources

This section describes how to configure Web and CSP resources.

This section includes the following subsections:

- Section 62.3.1.1.1, "Configuring General Settings"
- Section 62.3.1.1.2, "Configuring CSP settings"
- Section 62.3.1.1.3, "Configuring Policy Settings"
- Section 62.3.1.1.4, "Configuring Autologin"
- Section 62.3.1.1.5, "Configuring Autologin: Form Login"
- Section 62.3.1.1.6, "Configuring Autologin: Basic Login and NTLM Login"
- Section 62.3.1.1.7, "Configuring Autologin: Kerberos Login"
- Section 62.3.1.1.8, "Configuring Autologin: Authentication Sources"
- Section 62.3.1.1.9, "Configuring Headers"

This section uses the welcome_resource created when Pagelet Producer is installed as an example.

62.3.1.1.1 Configuring General Settings

Use the General page (see Figure 62–9), to enter basic information about the resource.
Enter a **Name** for the resource.

Enter a **Description** for the resource (optional).

In the **Source URL** field, type the URL to the location of the Web application resource to be proxied. For example, `http://internalServer/foo/`.

---

**Note:** If you are configuring an ADF Web Application as a resource, the Source URL cannot be any more specific than `http://hostname:portnumber/context-root/`.

---

By default, Pagelet Producer attempts to connect to the resource for 30 seconds before returning an error message. To change this value, enter a new **Source Timeout** period in seconds.

In the **Destination URL** field, type the relative path to be used to access the resource. This path will be used to create an URL to the resource on the server that hosts Pagelet Producer.

To enable Dynamic HTML, choose **DHTML Rewriting**. This option supports URLs that are not in the original HTML returned from the server, but are added by DHTML. In most cases, this option should be enabled.

To enable rewriting of DHTML through asynchronous Ajax calls select **Asynchronous Rewriting**.

### 62.3.1.1.2 Configuring CSP settings

By default, the CSP login token is not passed to the proxied resource. To enable this feature, choose **Send CSP Login Token**. Note that the CSP page is only available for CSP resources.

### 62.3.1.1.3 Configuring Policy Settings

The Policy page (see Figure 62–10) lets you limit access to a resource to specific roles within WebCenter Portal.
Creating Pagelet Producer Objects

Figure 62–10  Pagelet Producer Console: Resource - Policy Page

The J2EE container hosting Pagelet Producer (such as Oracle WebLogic Server) is responsible for establishing the role memberships associated with the current user. A resource can specify multiple roles on the Policy page, and users will be allowed access if they are a member of any of the specified roles; otherwise they will be directed to a suitable J2EE container delegated authentication page to establish the required credentials. If no roles are entered in the list, anonymous access is allowed, and the resource is termed as an "anonymous resource".

Note: The role name(s) entered on this page must match those created in the J2EE container (such as Oracle WebLogic Server).

62.3.1.1.4 Configuring Autologin

The Autologin feature allows Pagelet Producer to supply credentials to applications automatically. The Autologin page (see Figure 62–11) lets you configure authentication information for all users who access the resource.

Figure 62–11  Pagelet Producer Console: Resource - Autologin Page

The following sections describe how to configure credential mapping for authentication:

- Section 62.3.1.1.5, "Configuring Autologin: Form Login" describes how to configure autologin for a resource that prompts for authentication with an HTML form.
Section 62.3.1.1.6, "Configuring Autologin: Basic Login and NTLM Login" describes how to configure autologin for a resource that prompts for authentication with basic authentication or NTLM.

Section 62.3.1.1.7, "Configuring Autologin: Kerberos Login" describes how to configure autologin for a resource that prompts for authentication with Kerberos.

Section 62.3.1.1.8, "Configuring Autologin: Authentication Sources" describes the static, user profile, and credential vault authentication field sources.

62.3.1.1.5 Configuring Autologin: Form Login

This section describes how to configure Autologin for a resource that prompts for authentication with an HTML form.

1. On the Autologin page for the resource, expand the Form Login section (Figure 62–12).

2. The login page can be identified by an URL or a regular expression. In the Login Form Identification section, choose one of the following options:
   - If the login form is located at a static URL, select URL and enter the URL in the Value field. You can choose to Automatically Detect Form Fields on the page or enter them manually as described in step 4 below.
   - If the login form is dynamic, select RegEx and type the regular expression pattern into the box.

3. Set the login form action. In the Form Submit Location section, choose one of the following options:
   - If the login form action is a static URL, select URL and type the URL into the box. Choose the action for the form submission: POST or GET.
   - If the login form is dynamic, select RegEx and type the regular expression pattern into the box.

4. To map fields from the form to authentication field sources, either click Automatically Detect Form Fields in the Login Form Identification section as described above or enter them manually using the process below:
   a. Click Create to add a new row to the Form Fields list.
   b. Enter the name of the HTML form input in the Field Name box.
c. For details on how to configure the **Source** and **Value** properties, see Section 62.3.1.1.8, "Configuring Autologin: Authentication Sources."

---

**Note:** Sensitive fields should be stored securely using the credential vault (User Vault or Secured).

---

d. To delete field mappings, click **Delete**.

5. The logout page and login error pages can also be identified by an URL or a regular expression. In the **Logout Page Identification** and **Login Error Page Identification** sections, choose one of the following options:

- If the page is located at a static URL, select **URL** and type the URL into the field provided.
- If the page is dynamic, select **RegEx** and type the regular expression pattern into the field provided.

62.3.1.1.6 **Configuring Autologin: Basic Login and NTLM Login** This section describes how to configure autologin for a resource that prompts for authentication with basic authentication or NTLM.

1. On the **Autologin** page for the resource, expand the **Basic Login** or **NTLM Login** section.

---

**Note:** Basic authentication transmits passwords as plain text, and therefore, it must not be used in production systems. Further, it is strongly recommended that the underlying transport is HTTPS.

---

2. In the **Username** and **Password** sections, choose the appropriate authentication source and enter a value as necessary. For details on how to configure these properties, see Section 62.3.1.1.8, "Configuring Autologin: Authentication Sources."

62.3.1.1.7 **Configuring Autologin: Kerberos Login** This section describes how to configure autologin for a resource that prompts for authentication using Kerberos. For information on defining basic Kerberos settings, see Section 62.2, "Configuring Pagelet Producer Settings."

1. On the **Autologin** page for the resource, expand the **Kerberos Login** section.

2. In the **Username** and **Password** sections, choose the appropriate authentication source and enter a value as necessary. For details on how to configure these properties, see the next section, Section 62.3.1.1.8, "Configuring Autologin: Authentication Sources."

3. In the **SPN** field, enter the Service Principal Name (SPN) for the Kerberos account, in the format `http://hostname_with_kerberos`. (Before the Kerberos authentication service can use an SPN to authenticate a service, the SPN must be registered on the account object that the service instance uses to log on.)

62.3.1.1.8 **Configuring Autologin: Authentication Sources** Authentication sources define the source for login fields. **Table 62–2** describes each of the authentication field source values:
Creating Pagelet Producer Objects

### Configuring Headers

The Headers page (see Figure 62–13) lets you choose request and response headers that should be dropped from the HTTP that is provided by Pagelet Producer.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unsecured</td>
<td>Uses the provided authentication information for all users accessing the resource. Type the static value in the field provided.</td>
</tr>
<tr>
<td>Secured</td>
<td>Uses the field value is entered by an administrator in the Pagelet Producer Administration Console. The value is stored in the Credential Vault and is shared among all users, including non-authenticated (anonymous) users. The field key is auto-generated and displayed as a read-only field in the Administration Console.</td>
</tr>
<tr>
<td>User Vault</td>
<td>Prompts the user for credentials the first time the resource is accessed. The supplied credentials are encrypted and stored in the credential vault, and each subsequent access to the resource is authenticated with the stored credentials. The field key is auto-generated and displayed as a read-only field in the Administration Console. In the second field, enter the name of the credential vault to use, or leave the entry as “default” to use the server vault. <strong>Note:</strong> When you choose User Vault, the user will be presented with an error page that includes a link: “Click here to access pagelet directly.” This link opens the authentication dialog. This is a known issue and occurs only the first time the resource is accessed by a user.</td>
</tr>
<tr>
<td>Generated</td>
<td>(Form Login only) The field value is taken from the backend server response markup.</td>
</tr>
</tbody>
</table>

#### 62.3.1.1.9 Configuring Headers

The Headers page (see Figure 62–13) lets you choose request and response headers that should be dropped from the HTTP that is provided by Pagelet Producer.

![Pagelet Producer Console: Resource - Headers Page](image)

**ORACLE® WebCenter Portal: Pagelet Producer**

Some header elements should be blocked from being passed to back-end applications. For example, when using delegated (third-party SSO) authentication, the SSO system might insert some headers that need not be passed to the back-end applications. When passed, these headers might interfere with the back-end application functionality.

The following headers are dropped by default:
Creating Pagelet Producer Objects

The Content-Length header is always implicitly dropped, because manipulating content during the proxying operation renders the content length invalid in almost all cases.

To add a header to the list, click Create and enter the header name in the field provided.

Once the Web or CSP resource is defined, you can create pagelets and other objects. For more information, see the following sections:

- Section 62.3.2, "Creating Pagelets"
- Section 62.3.3, "Creating Web Injectors"
- Section 62.3.4, "Creating Custom Parsers"
- Section 62.3.5, "Creating Hosted Files"

### 62.3.1.2 Configuring WSRP and JPDK Resources

This section describes how to configure WSRP/JPDK resources based on WSRP or Oracle JPDK portlet producers.

#### Note:
Before you can create a resource based on a WSRP or Oracle JPDK portlet producer, you must register the producer as described in the "Managing the Pagelet Producer" chapter in Administering Oracle WebCenter Portal.

This section includes the following subsections:

- Section 62.3.1.2.1, "Configuring General Settings"
- Section 62.3.1.2.2, "Configuring Policy Settings"

#### 62.3.1.2.1 Configuring General Settings

Use the General page to enter basic information about a WSRP or JPDK resource.

- Choose the portlet producer type from the Portlet Producer dropdown list. This list is populated with the producers that have been registered as described in the "Managing the Pagelet Producer" chapter in Administering Oracle WebCenter Portal.
Enter a Name for the resource.

Enter a Description for the resource (optional).

62.3.1.2.2 Configuring Policy Settings The Policy page lets you limit access to a resource to specific roles within Oracle WebCenter Portal.

The J2EE container hosting Pagelet Producer (such as Oracle WebLogic Server) is responsible for establishing the role memberships associated with the current user. A resource can specify multiple roles on the Policy page, and users will be allowed access if they are a member of any of the specified roles; otherwise they will be directed to a suitable J2EE container delegated authentication page to establish the required credentials. If no roles are entered in the list, anonymous access is allowed, and the resource is termed as an "anonymous resource".

---

Notes:
The role name(s) entered on this page must match those created in the J2EE container (such as Oracle WebLogic Server).

Once the WSRP or JPDK resource is defined, you can create pagelets and Web injectors. See the following sections for more information:

- Section 62.3.2, "Creating Pagelets"
- Section 62.3.3, "Creating Web Injectors"

62.3.1.3 Configuring OpenSocial Resources (OpenSocial Gadget Producers)

This section describes how to configure OpenSocial resources based on OpenSocial gadget producers.

- Section 62.3.1.3.1, "Configuring General Settings"
- Section 62.3.1.3.2, "Configuring Policy Settings"

---

Note:
Configure Pagelet Producer for use with OpenSocial before creating an OpenSocial resource. For more information, see Section 62.2.7, "Configuring OpenSocial Settings."

62.3.1.3.1 Configuring General Settings Use the General page to enter basic information about the resource.

- Enter a Name for the resource.
- Enter a Description for the resource (optional).

62.3.1.3.2 Configuring Policy Settings The Policy page lets you limit access to a resource to specific roles within Oracle WebCenter Portal.

The J2EE container hosting Pagelet Producer (such as Oracle WebLogic Server) is responsible for establishing the role memberships associated with the current user. A resource can specify multiple roles on the Policy page, and users will be allowed access if they are a member of any of the specified roles; otherwise they will be directed to a suitable J2EE container delegated authentication page to establish the required credentials. If no roles are entered in the list, anonymous access is allowed, and the resource is termed as an "anonymous resource".
62.3.2 Creating Pagelets

The pagelets collection lists the pagelets associated with the resource. To create a new pagelet, select the Pagelets section under the resource you want to use in the Pagelet Producer Console and click the Create icon in the toolbar. A pagelet called “<new>” will be added to the list. To modify an existing pagelet, click the pagelet name.

This section contains the following subsections:

- Section 62.3.2.1, “Configuring General Settings”
- Section 62.3.2.2, “Configuring Preferences”
- Section 62.3.2.3, “Configuring Parameters”
- Section 62.3.2.4, “Configuring the Clipper”
- Section 62.3.2.5, “Accessing the Pagelet and Preference Editor”

62.3.2.1 Configuring General Settings

Enter a Name for the pagelet and the Library name with which to associate the pagelet. (A pagelet library is a user-defined way to group related pagelets.) Enter a Description for the pagelet (optional).

- For Web and CSP pagelets, type the relative path to the pagelet in the URL Suffix field (do not include the Source URL prefix you entered for the resource). If you leave the URL Suffix blank, then the entire resource will be considered the pagelet.

- For pagelets based on WSRP or Oracle JPDK portlets, choose the portlet on which to base the pagelet from the Portlet dropdown list. This list is populated with the portlets on the WSRP or Oracle JPDK portlet producer associated with the parent resource. Any public parameters associated with the portlet will automatically be imported as pagelet parameters. For more information on support for WSRP and Oracle JPDK portlets, see Section 62.4, "Working with Pagelet Chrome for WSRP and Oracle JPDK Portlets."

- For pagelets based on OpenSocial gadgets, enter the location of the gadget XML schema in the Gadget URL field. Click the Import Gadget Metadata button to import the following information from the XML schema:
  - Gadget name: This value will be imported into the Description field on the General page.
  - Gadget user preferences: The pagelet parameters on the Configuring Parameters page will be populated with the gadget’s user preferences.

   For more information on support for OpenSocial gadgets, see Section 62.5, "Working with OpenSocial Gadgets."

To block access to the pagelet, choose Disabled. If the pagelet is included on any pages, it will display a simple error message.
62.3.2.2 Configuring Preferences

On the Preferences page (see Figure 62–14), enter the relative URLs to any preference pages required by the pagelet: Global, Customize, or Personalize. Do not include the Source URL prefix you entered for the resource. (As noted above, for OpenSocial gadgets with user preferences, a default entry will be created; this entry should not be modified.)

The Preferences page is not used by WSRP or Oracle JPDK-based pagelets.

Figure 62–14  Pagelet Producer Console: Pagelets - Preferences Page

62.3.2.3 Configuring Parameters

Data can be passed to pagelets using pagelet parameters. Parameters pass name-value pairs to the pagelet application as described below.

On the Parameters page (see Figure 62–15), enter the Parameters that should be passed to the pagelet.

To add a parameter, click Create.

- Enter the Name of the parameter.
- If the parameter is required by the pagelet, select the Required checkbox.
- Choose the appropriate data Type: string or numeric.
- Enter a Description (optional).

Figure 62–15  Pagelet Producer Console: Pagelets - Parameters Page
62.3.2.4 Configuring the Clipper

Clipping lets you create a pagelet by clipping a portion of a larger Web page in a proxied application. A news Web page, for example, may contain a box listing the latest headlines. By identifying the containing HTML for that box, you can clip only the headlines and serve that subset of the news Web page as a pagelet.

To create a clipper, select the Clipper section under the pagelet and click the Create icon in the toolbar. A new clipper will be created with two configuration pages:

- On the General page, enter a name for the clipper.
- On the Content page (see Figure 62–16), define the clipper content.

**Figure 62–16  Pagelet Producer Console: Clipper - Content Page**

- To use a graphical tool to select page content, click Launch Clipper.

- To specify HTML tag attributes that describe the section to be clipped, expand the Advanced Clipper section (Figure 62–17) and enter the tag name and associated attribute(s).

**Figure 62–17  Advanced Clipper**

Keep the following in mind when using the clipper:

- If the back-end resource is accessed over HTTPS, make sure the Pagelet Producer Console is also accessed over a secure port.

- If the clip source is protected by a login form or other form of authentication, make sure to configure Autologin for the parent resource as described in Section 62.3.1.1.4, "Configuring Autologin." If you are using the vault to store credential values (see Section 62.3.1.1.8, "Configuring Autologin: Authentication Sources"), make sure to capture the credentials prior to using the clipper.
Creating Pagelet Producer Objects

- If you are having problems with the clipper, make sure the configured pagelet URL can be loaded by the browser without redirects. If necessary, change the pagelet suffix to reflect the final URL loaded by the browser after following all the redirects.

- Images and code overwritten using hosted files cannot be clipped. (For more information about hosted files, see Section 62.3.5, "Creating Hosted Files.")

- If the graphical clipper cannot be used (for example, if the page ID is not available), use the Advanced Clipper to define the page region to clip using a regular expression.

62.3.2.5 Accessing the Pagelet and Preference Editor

Use the Documentation page (see Figure 62–18) to display sample code with which to access the pagelet and preference editor using either JavaScript or the REST API.

![Figure 62–18  Pagelet Producer Console: Pagelets - Documentation Page](image)

62.3.3 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.

To create a Web injector, select the Injectors section under the resource you want to use and click the Create icon in the toolbar. A new injector called "<new>" will be added to the list. Injectors can then be configured using the General and Content configuration pages as described in the following sections:

- Section 62.3.3.1, "Configuring General Settings"
- Section 62.3.3.2, "Injecting Content"

62.3.3.1 Configuring General Settings

Use the General page (see Figure 62–19) to configure basic settings for the injector. Enter a Name for the Web injector.
The injector can be applied to a subset of the resource by typing a URL pattern into the **URL Filter** box. 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.

In the **Inject Location** section, define where in the resource’s output the injection will be made in relation to a unique string. Enter the unique string in the field provided and choose **Before**, **After** or **Replace** to define where to put the content relative to the string. If you choose to replace the string, you can use the **Enclose tag** to replace both the string and the enclosing tag. You can choose to ignore the case of the string by selecting **Ignore case**.

**Figure 62–19  Pagelet Producer Console: Injectors - General Page**

### 62.3.3.2 Injecting Content

Use the Content page to define content to be injected using the injector. Content to be injected may be any text, including HTML, CSS, JavaScript, and pagelet declarations.

For example, the following code could be injected at the beginning of a page. The 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).

```html
<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.className == 'page_header') {
```
```javascript
    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>

62.3.4 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. Note that parsers cannot be created for WSRP or Oracle JPDK portlet producers or 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 to display the Parser's General page (see Figure 62–20):

- 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.
Creating Pagelet Producer Objects

For example, the regular expression `XMLFile=([^\?]*?[^\w])` 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”.

62.3.5 Creating Hosted Files

Pagelet Producer can host all types of content (HTML, JavaScript, CSS, etc.) and present the file at a virtual URL location. Hosted files can be used for a range of purposes:

- Overwrite content and functionality in a proxied application, by uploading files and configuring them to use the same URL as the original file.
- Use hosted files in injectors to insert images or content into a proxied application.
- Host pagelet files on the Pagelet Producer server.

To upload a file, select the **Files** section under the resource you want to use and click the **Create** icon in the toolbar.

- On the **General** page (see Figure 62–21), enter the relative path to the file in the **Name** field. Do not use a leading forward-slash ("/"). The directory structure of the Files collection in the navigator is updated to match the path to the file.

Enter the **MIME type** of the file.

- On the **Content** page, enter the path to the file or click **Browse** to navigate to the file.
Click **Upload** to upload the file to the Pagelet Producer server. If you entered text or html as the MIME type, you can also use the editor on the Content page to enter or edit file content. (The editor is only available for text and html files.) If you entered an image type as the MIME type, the uploaded image will be displayed on the Content page.

**Note:** Text files (text/plain) uploaded to the Pagelet Producer must be saved as UTF-8.

- Click the **Save** icon in the navigator toolbar.

After the file is uploaded, it will be available for use in injectors and pagelets at the following URL: `http://<host_name>:<port_number>/pagelets/<resourcename>/<filepath>`.

For example, if the file was uploaded under the "welcome_resource" resource and the name for the file was entered as "images/myimage.jpg" the path to the hosted file on the Pagelet Producer server would be: `http://<host_name>:<port_number>/pagelets/welcome_resource/images/myimage.jpg`

Keep the following in mind when working with hosted files:

- The hosted file feature should not be used for bulky files.
- 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.
- Hosted files cannot be created for WSRP or Oracle JPDK portlet producers.

### 62.4 Working with Pagelet Chrome for WSRP and Oracle JPDK Portlets

Pagelet Producer can be used to present WSRP and Oracle JPDK portlets for use in any Web application. This section explains how to use pagelet chrome to modify markup at runtime in WSRP or Oracle JPDK portlets. For details on configuring the Pagelet Producer to connect to a WSRP or Oracle JPDK portlet producer, see the "Managing the Pagelet Producer" chapter in *Administering Oracle WebCenter Portal*.

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 use the following reserved tokens (see Table 62–3) to identify key portlet elements that Pagelet Producer then substitutes at runtime.

<table>
<thead>
<tr>
<th>Token</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$$PORTLET TITLE$$</td>
<td>Portlet title</td>
</tr>
<tr>
<td>$$REPEAT MENU ITEM$$</td>
<td>Used to indicate the beginning of a repeating section for navigational items</td>
</tr>
</tbody>
</table>

---

90x370 Working with Pagelet Chrome for WSRP and Oracle JPDK Portlets

Pagelet Producer can be used to present WSRP and Oracle JPDK portlets for use in any Web application. This section explains how to use pagelet chrome to modify markup at runtime in WSRP or Oracle JPDK portlets. For details on configuring the Pagelet Producer to connect to a WSRP or Oracle JPDK portlet producer, see the "Managing the Pagelet Producer" chapter in *Administering Oracle WebCenter Portal*.

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 use the following reserved tokens (see Table 62–3) to identify key portlet elements that Pagelet Producer then substitutes at runtime.

**Table 62–3** Pagelet Chrome Template Tokens

<table>
<thead>
<tr>
<th>Token</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$$PORTLET TITLE$$</td>
<td>Portlet title</td>
</tr>
<tr>
<td>$$REPEAT MENU ITEM$$</td>
<td>Used to indicate the beginning of a repeating section for navigational items</td>
</tr>
</tbody>
</table>
The example below is a very simple pagelet chrome template:

```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>
<!-- $$PORTLET CONTENT$$ -->

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 62.7.2, "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

<table>
<thead>
<tr>
<th>Token</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{END REPEAT MENU ITEMS}$</td>
<td>Used to indicate the end of a repeating section for navigational items</td>
</tr>
<tr>
<td>$\text{MENU ITEM URL}$</td>
<td>Navigation URL (to switch portlet modes or window states)</td>
</tr>
<tr>
<td>$\text{MENU ITEMS}$</td>
<td>Display name of navigational item (for example, Customize)</td>
</tr>
<tr>
<td>$\text{TOKEN}$</td>
<td>Unique identifier for pagelet instance on the page</td>
</tr>
<tr>
<td>$\text{PORTLET CONTENT}$</td>
<td>Portlet content</td>
</tr>
<tr>
<td><code>pt://images</code></td>
<td>Tag used to indicate the imageserver URL</td>
</tr>
</tbody>
</table>

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.
template can be overridden with a custom template or with the standard default template by setting `chrome=chrometemplate.html`.

62.5 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. To support OpenSocial gadgets, you must first define an OpenSocial container as described in Section 62.2.7, "Configuring OpenSocial Settings."

Pagelet Producer supports most of the standard OpenSocial APIs excluding OAuth. The complete OpenSocial API reference documentation can be found here: http://docs.opensocial.org/display/OSD/Specs

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.

This section contains the following subsections:

- Section 62.5.1, "Configuring Authentication"
- Section 62.5.2, "Storing User Preferences"
- Section 62.5.3, "Accessing WebCenter Portal Profile Information"
- Section 62.5.4, "Accessing a User’s Activity Stream"
- Section 62.5.5, "Using Gadget Eventing"
- Section 62.5.6, "Example: Consuming an External OpenSocial Gadget"
- Section 62.5.7, "Example: Consuming a Local OpenSocial Gadget"

62.5.1 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 (see Section 62.3.1.3.2, "Configuring Policy Settings"). 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.

62.5.2 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 customizations).
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)

### 62.5.3 Accessing WebCenter Portal Profile Information

OpenSocial gadgets can query the current user's profile data and people connections via the standard OpenSocial API. To use this feature, you must manually target the WebCenterDS data source to the WC_Portlet server as described in the "Managing the Pagelet Producer" chapter in *Administering Oracle WebCenter Portal*.

The supported user profile fields are listed in Table 62–4.

**Table 62–4  User Profile Fields**

<table>
<thead>
<tr>
<th>OpenSocial Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aboutme</td>
<td>string</td>
<td>A general statement about the person.</td>
</tr>
<tr>
<td>addresses</td>
<td>Plural-Field &lt;Address&gt;</td>
<td>A physical mailing address for the person.</td>
</tr>
<tr>
<td>appData</td>
<td>Plural-Field &lt;AppData&gt;</td>
<td>A collection of AppData keys and values, used for preferences.</td>
</tr>
<tr>
<td>birthday</td>
<td>Date</td>
<td>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.</td>
</tr>
<tr>
<td>emails</td>
<td>Plural-Field &lt;string&gt;</td>
<td>Email address for the person.</td>
</tr>
<tr>
<td>location</td>
<td>string</td>
<td>Physical address for the person.</td>
</tr>
<tr>
<td>name</td>
<td>Name</td>
<td>The broken-out components and formatted version of the person's real name.</td>
</tr>
<tr>
<td>organizations</td>
<td>Plural-Field &lt;Organization&gt;</td>
<td>Current or past organizational affiliation of the person.</td>
</tr>
<tr>
<td>phoneNumbers</td>
<td>Plural-Field &lt;string&gt;</td>
<td>Phone number for the person. In addition to the standard canonical values for type, this field defines the additional values mobile, fax, and pager.</td>
</tr>
<tr>
<td>photos</td>
<td>Plural-Field &lt;string&gt;</td>
<td>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.</td>
</tr>
<tr>
<td>preferredUsername</td>
<td>string</td>
<td>The preferred username of this person on sites that ask for a username (e.g. jsmarr or daveman692).</td>
</tr>
<tr>
<td>status</td>
<td>string</td>
<td>The person's status, headline or shoutout.</td>
</tr>
<tr>
<td>thumbnailUrl</td>
<td>string</td>
<td>The person's photo thumbnail URL, specified as a string. This URL MUST be fully qualified.</td>
</tr>
</tbody>
</table>

### 62.5.4 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 Table 62-5.

<table>
<thead>
<tr>
<th>OpenSocial Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>appId</td>
<td>Object-Id</td>
<td>The application with which the activity is associated.</td>
</tr>
<tr>
<td>body</td>
<td>string</td>
<td>An optional expanded version of the activity. Bodies may only have the following HTML tags: &lt;b&gt; &lt;i&gt;, &lt;a&gt;, &lt;span&gt;, but this formatting may be ignored.</td>
</tr>
<tr>
<td>externalId</td>
<td>Object-Id</td>
<td>An optional ID generated by the posting application.</td>
</tr>
<tr>
<td>id</td>
<td>Object-Id</td>
<td>An ID that is permanently associated with the activity.</td>
</tr>
<tr>
<td>postedTime</td>
<td>string</td>
<td>The time at which the activity took place in milliseconds since the epoch.</td>
</tr>
<tr>
<td>priority</td>
<td>number</td>
<td>A number between 0 and 1 representing the relative priority of the activity in relation to other activities from the same source.</td>
</tr>
<tr>
<td>title</td>
<td>string</td>
<td>The primary text of the activity. Titles may only have the following HTML tags: &lt;b&gt; &lt;i&gt;, &lt;a&gt;, &lt;span&gt;, but this formatting may be ignored.</td>
</tr>
<tr>
<td>userId</td>
<td>Object-Id</td>
<td>ID of the user for whom the activity is defined.</td>
</tr>
</tbody>
</table>

62.5.5 Using Gadget Eventing

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.

```javascript
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://docs.opensocial.org/display/OSD/Specs

62.5.6 Example: Consuming an External OpenSocial Gadget

The example that follows is simplified for illustration. To support OpenSocial gadgets, you must first define an OpenSocial container as described in Section 62.2.7, "Configuring OpenSocial Settings."

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
Working with OpenSocial Gadgets

editable, 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.

62.5.7 Example: Consuming a Local OpenSocial Gadget

The example that follows is simplified for illustration. To support OpenSocial gadgets, you must first define an OpenSocial container as described in Section 62.2.7, "Configuring OpenSocial Settings."

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.

   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 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

   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.
editable, Pagelet Producer will create a preference editor using the imported preferences.

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.

62.6 Building Custom Pagelets Using Pagelet Producer

This section provides detailed information about building pagelets using Pagelet Producer, including how to use the Adaptive Pagelet Scripting Framework, configure security settings for pagelets and resources to manage authentication for users, use custom Web injectors and parsers to modify pagelet functionality at runtime, and access advanced logging traces for debugging.

This section contains the following subsections:

- Section 62.6.1, "Using the Adaptive Pagelet Scripting Framework"
- Section 62.6.2, "Modifying Pagelet Functionality at Runtime"
- Section 62.6.3, "Debugging Pagelets"

62.6.1 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.

This section includes the following subsections:

- Section 62.6.1.1, "Handling Structured HTTP Responses"
- Section 62.6.1.2, "Using Event Notification"
- Section 62.6.1.3, "Using In-Place Refresh"
- Section 62.6.1.4, "Using Session Preferences"

For a full list of classes and methods, see the JSPortlet API documentation. For additional information about adaptive pagelets, see Section 62.6.1.5, "Adaptive Pagelet Development Tips."

62.6.1.1 Handling Structured HTTP Responses

This section describes how the adaptive pagelet scripting framework can be used as a client-side response handler for structured HTTP, typically encoded as XML.

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 (structuredresponseportlet.html) accesses an RSS feed from a selection of news sites.

```html
<!-- 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>

<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();

// Select RSS Feed:
<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';
</script>
62.6.1.2 Using Event Notification

This section describes how 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 62.6.1.2.1, "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.

![Broadcast Portlet](image)

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.

```javascript
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>
```
The second listener pagelet displays the cube root of the number entered in the broadcast pagelet.

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**

```html
<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>
<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');

  PTPortlet.raiseEvent('onBroadcastUpdate', {name:prefName, value:prefValue});
}
</script>
</div>
```
Listener Pagelet #1

```html
<p style="padding:10px; text-align:center">
  <b>Square root:</b>
  <div style="height:21px; border:2px solid black; padding:2px; overflow:visible; font-size:14px; id="listener1-swatch">
  </div>
</p>

<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;
}
</script>
```
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></p>
<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 = '{D9DF3F4-EAE7-5478-0F4C-2BDCA1C7000}';
    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>

### 62.6.1.2.1 Page-Level Events for Use with the Scripting Framework

The scripting framework automatically has access to the following page-level events.

<table>
<thead>
<tr>
<th>Event</th>
<th>Triggered</th>
</tr>
</thead>
<tbody>
<tr>
<td>onload</td>
<td>immediately after the browser loads the page</td>
</tr>
<tr>
<td>onbeforeunload</td>
<td>prior to a page being unloaded (browser window closes or navigates to different location)</td>
</tr>
<tr>
<td>onunload</td>
<td>immediately before the page is unloaded (browser window closes or navigates to different location)</td>
</tr>
<tr>
<td>onactivate</td>
<td>the page is set as the active element (receives focus)</td>
</tr>
<tr>
<td>onbeforeactivate</td>
<td>immediately before the page is set as the active element (receives focus)</td>
</tr>
<tr>
<td>ondeactivate</td>
<td>when the active element is changed from the current page to another page in the parent document</td>
</tr>
<tr>
<td>onfocus</td>
<td>when the page receives focus</td>
</tr>
<tr>
<td>onblur</td>
<td>when the page loses focus</td>
</tr>
<tr>
<td>oncontrolselect</td>
<td>when the user is about to make a control selection of the page</td>
</tr>
<tr>
<td>onresize</td>
<td>when the size of the page is about to change</td>
</tr>
<tr>
<td>onresizestart</td>
<td>when the user begins to change the dimensions of the page in a control selection</td>
</tr>
<tr>
<td>onresizeend</td>
<td>when the user finishes changing the dimensions of the page in a control selection</td>
</tr>
<tr>
<td>onhelp</td>
<td>when the user presses the F1 key while the browser is the active window</td>
</tr>
<tr>
<td>onerror</td>
<td>when an error occurs during page loading</td>
</tr>
<tr>
<td>onafterprint</td>
<td>immediately after an associated document prints or previews for printing</td>
</tr>
</tbody>
</table>

### 62.6.1.3 Using In-Place Refresh

This section describes how pagelets can reload their internal content without refreshing the page using 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.
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.)

```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;
  }
  refreshPortlet.refresh();
}

function refresh_debug(str)
{
  if (window.PTDebugUtil)
  {
    PTDebugUtil.debug(str);
  }
}

var t = new Date();
document.getElementById('refreshTimeSpan').innerHTML = t;
```

### 62.6.1.4 Using Session Preferences

This section describes how 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 using 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 62–26  Color Code**

When the user enters a color code in the text box, the color in the detail pagelet changes.

**Figure 62–27  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**

```html
<div style="padding:10px;" align="center">
<p>Enter color:</p>
</div>
```
Building Custom Pagelets Using Pagelet Producer

<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> &nbsp;
<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)
Creating Pagelets with Pagelet Producer 62-45


62.6.1.5 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.

- **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 proceed your code with an If statement that alerts the user if the scripting framework is not supported.

  ```html
  <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.

62.6.2 Modifying Pagelet Functionality at Runtime
This section describes how to modify pagelet functionality at runtime using custom injectors and parsers.
This section includes the following subsections:

- Section 62.6.2.1, "Using Web Injectors"
- Section 62.6.2.2, "Using Custom Parsers"

62.6.2.1 Using 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. For details on creating a Web injector, see Section 62.3.3, "Creating Web Injectors."

62.6.2.2 Using 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. For details on creating a custom parser, see Section 62.3.4, "Creating Custom Parsers."

62.6.3 Debugging Pagelets

This section describes how to use 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 more information, see Section 62.2, "Configuring Pagelet Producer Settings."

This section includes the following subsections:

- Section 62.6.3.1, "Viewing HTTP Requests and Responses"
- Section 62.6.3.2, "Viewing Transformation Content"

62.6.3.1 Viewing HTTP Requests and Responses

To view the HTTP requests and responses received and sent by Pagelet Producer, set the HTTP component on the Logging Settings page to Finest. The traces in Example 62–1, Example 62–2, Example 62–3, and Example 62–4 were captured from a test environment.

Example 62–1  HTTP Request Received By Pagelet Producer

URL: http://example.com:7001/pagelets/bidwiki/includes/js/ajax.js
METHOD: GET
SESSION ID: GdYGNJzeMuxy1CJBMVTX8xTNq32GmLXNY9VgFBcdprFnhcyQtzdp!1377086614!1305769149498
HEADERS:
  Host: example.com
  Connection: keep-alive
  Referer: http://example.com: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
  (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
Example 62–2  HTTP Response Sent By Pagelet Producer

URL: http://example.com: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://example.com:7001/pagelets/bidwiki/download/resources/leftnav_bg.jpg);*/
    /*background-repeat: repeat-y;*/
    background-color: #F0F0F0;
    /*border-bottom:1px solid #F0F0F0;*/
  } ...

Example 62–3  HTTP Request Sent By Pagelet Producer

URL: http://xmlns.oracle.com/includes/js/ajax.js
METHOD: GET
HEADERS:
  CSP-Ensemble-REST-API: http://example.com:7001/pagelets
  X-Client-IP: example.com
  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 2012 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://example.com: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 62–4  HTTP Response Received By Pagelet Producer

Original URI: http://xmlns.oracle.com/styles/main-action.css
Effective URI: http://xmlns.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:
62.6.3.2 Viewing Transformation Content

To view the content proxied by 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 in Example 62–5 and Example 62–6 were captured from a test environment.

**Example 62–5 Untransformed Markup**

Original request:
URL: http://example.com:7001/pagelets/bidwiki/styles/main-action.css
METHOD: GET
SESSION ID: GdYGNJzMxxy1CJBMVTX8xTNq32GmLXN9YVqFBCdprFnhcyQtzdp!1377086614!1305769149498
HEADERS:
  
  Host: example.com:7001
  Connection: keep-alive
  Referer: http://example.com: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: GdYGNJzMxxy1CJBMVTX8xTNq32GmLXN9YVqFBCdprFnhcyQtzdp!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 62–6 Transformed Markup (Transformed by Transformer Class)**

Transformed by: class com.plumtree.server.impl.portlet.transformers.CSSTurboParser

Original request:
URL: http://example.com:7001/pagelets/bidwiki/styles/main-action.css
METHOD: GET
SESSION ID: GdYGNJzMxxy1CJBMVTX8xTNq32GmLXN9YVqFBCdprFnhcyQtzdp!1377086614!1305769149498
HEADERS:
  
  Host: 10.148.118.211:7001
  Connection: keep-alive
  Referer: http://example.com: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
Using Pagelets in Web Applications

This section describes how to add pagelets to a JSF page in JDeveloper, to a Web page, and to a portal page. Before you can add a pagelet to any Web application, you must deploy and configure the resource and pagelet in Pagelet Producer as described in the "Deploying Portals, Templates, Assets, and Extensions" chapter in Administering Oracle WebCenter Portal.

This section includes the following subsections:

- Section 62.7.1, "Adding a Pagelet to a JSF Page in Oracle JDeveloper"
- Section 62.7.2, "Adding a Pagelet to a Web Page"
- Section 62.7.3, "Adding a Pagelet to a Portal Page"

62.7.1 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 Pagelet Producer with your 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 62.7.1.1, "Registering Pagelet Producer with a Framework Application"
- Section 62.7.1.2, "Adding a Pagelet to a JSF Page"
- Section 62.7.1.3, "Securing a Pagelet"

62.7.1.1 Registering Pagelet Producer with a Framework Application

Before you can add a pagelet to a JSF page, you must register 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 Portal 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.
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, Pagelet Producer is created under the Pagelet Producer directory in the Connections directory, as shown in Figure 62–28.

Figure 62–28 Pagelet Producer in the Application Resources Panel

62.7.1.2 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 62.7.2.3, “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 62–29.
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 62–30 shows a sample pagelet in a browser window.

Figure 62–30  Sample Pagelet

62.7.1.3 Securing a Pagelet
For information about securing pagelets, see the "Configuring Oracle Single Sign-On (OSSO)" section in Administering Oracle WebCenter Portal.

62.7.2 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 62.7.2.1, "Inserting Pagelets Using Javascript": Pagelet Producer allows you to insert pagelets into non-proxied pages using a simple JavaScript function.
- Section 62.7.2.2, "Accessing Pagelets Using REST": Pagelet Producer REST APIs allow remote Web services to retrieve information about resources and pagelets from Pagelet Producer, and inject pagelets into proxied and non-proxied pages.
- Section 62.7.2.3, "Using Automatic Resizing with IFrames": The pagelet inject function can automatically resize the IFrame that encapsulates pagelet content.

62.7.2.1 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:

```javascript
function injectPagelet(library, name, iframe_options, 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.
- **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:

```html
<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>
```
62.7.2.1 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 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`.

62.7.2.2 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 Pagelet Producer to act as a portlet provider for Oracle WebCenter Interaction, Oracle WebLogic Portal, or other third-party portals. For details, see Section 62.7.2.2.1, "Pagelet Inject API."
- Allow remote Web services to retrieve information about resources and pagelets from Pagelet Producer. For details, see Section 62.7.2.2.2, "Data Retrieval APIs."

62.7.2.2.1 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 source attribute of an IFrame tag in any HTML page.

The URL must use the following format:

http://host:port/pagelets/inject/v2/libraryname/pageletname?content-type=html

where libraryname and pageletname refer to the library and pagelet configured in 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:


---

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.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>ifwidth</td>
<td>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 62.7.2.3, &quot;Using Automatic Resizing with IFrames&quot;.</td>
<td>100%</td>
</tr>
</tbody>
</table>
Using Pagelets in Web Applications

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>ifheight</td>
<td>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 62.7.2.3, &quot;Using Automatic Resizing with IFrames&quot;.</td>
<td>No default</td>
</tr>
<tr>
<td>ifborder</td>
<td>Sets the border of the IFrame.</td>
<td>'none'</td>
</tr>
<tr>
<td>ifalign</td>
<td>Sets the align rule within the IFrame, for example: ifalign=center.</td>
<td>No default</td>
</tr>
<tr>
<td>ifdesc</td>
<td>Sets the description of the IFrame.</td>
<td>No default</td>
</tr>
<tr>
<td>ifmarginheight</td>
<td>Sets the margin height; can be specified in percent '%' or pixels 'px', for example: ifmarginheight=500px.</td>
<td>No default</td>
</tr>
<tr>
<td>ifmarginwidth</td>
<td>Sets the margin width; can be specified in percent '%' or pixels 'px', for example: ifmarginwidth=500px.</td>
<td>No default</td>
</tr>
<tr>
<td>ifscrolling</td>
<td>Sets the scrollbars of the IFrame. Accepted values: yes/no/auto.</td>
<td>auto</td>
</tr>
<tr>
<td>ifstyle</td>
<td>Sets the CSS style of the IFrame</td>
<td>No default</td>
</tr>
<tr>
<td>ifclass</td>
<td>Sets the CSS class of the IFrame.</td>
<td>No default</td>
</tr>
</tbody>
</table>

- **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).

- **onhttperror**: Optional. When a pagelet request results in a 403, 404 or any other error code, 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): 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 Pagelet Producer controls the parent page.

- **inline-refresh**: Optional. Valid values are true or false (default). If the inline-refresh option is set to true, Pagelet Producer does not use an iFrame to embed pagelet markup onto the page. When the user interacts with the pagelet, markup updates are done inline, hence the term 'inline-refresh'. To achieve this, Pagelet Producer re-writes the URLs in the pagelet markup to use a JavaScript method call instead. That is, when the user clicks on a link or submits a form request to the backend, it's done by JavaScript on the browser and then the response is dynamically injected.

The inline-refresh option works well for relatively simple HTML markup. If the pagelet exposes a very rich UI, such as an ADF task flow, inline-refresh may not work well or may simply break. Consequently, the inline-refresh option is set to false by default (that is, inject pagelet markup in an iFrame).
The following example 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.

```
<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=true&content-type=html&jswrap=false&csapi=true">
<html>
<head>
<script src="http://proxy:loginserverport/loginserver/ensemblestatic/imageserver/plumtree/common/private/js/jsutil/LATEST/PTDateFormats.js" type="text/javascript"></script>
</head>
<body>
<div id="pt-pagelet-content-1" class="pagelet-container" style="display:inline;">
<span xmlns:pt="http://www.xmlns.oracle.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/csapi-pagelet.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>
```

**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).
62.7.2.2 Data Retrieval APIs

Two REST APIs are available to retrieve data from Pagelet Producer:

- **Pagelet API**: Allows remote applications to retrieve pagelet data from Pagelet Producer. (See Example 62–7, "All Pagelets" and Example 62–8, "Pagelets By Library and Name".)

- **Resource API**: Allows remote applications to retrieve resource data from Pagelet Producer. (See Example 62–9, "All Resources" and Example 62–10, "Resource By Name".)

The base URL for all requests is

http://host:port/pagelets/restservice/pageletproducer/.

**Example 62–7  All Pagelets**

http://host:port/pagelets/restservice/pageletproducer/pagelets/

http://host:port/pagelets/restservice/pageletproducer/pagelets/?format=xml

**Example 62–8  Pagelets By Library and Name**

http://host:port/pagelets/restservice/pageletproducer/pagelet/libraryname/pageletname/

http://host:port/pagelets/restservice/pageletproducer/pagelet/libraryname/pageletname/?format=xml

**Example 62–9  All Resources**

http://host:port/pagelets/restservice/pageletproducer/resources/

http://host:port/pagelets/restservice/pageletproducer/resources/

**Example 62–10  Resource By Name**

http://host:port/pagelets/restservice/pageletproducer/resource/name

http://host:port/pagelets/restservice/pageletproducer/resource/name/?format=xml

62.7.2.3 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&ifwidth=auto&ifheight=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
<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>

62.7.3 Adding a Pagelet to a Portal Page

Use Composer to add a pagelet to a portal page. 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 a portal, see the "Working with Resource Catalog Components on a Page" section in Using Oracle WebCenter Portal.

To configure a pagelet within a page, view the page in Edit mode and click the Edit icon (wrench) for the pagelet. The Pagelet Properties tab in the Component Properties dialog allows you to define pagelet parameters and IFrame options.
62.8 Examples and Advanced Topics

This section contains several examples and advanced topics:

- Section 62.8.1, "Creating a Simple Pagelet (an Example)"
- Section 62.8.2, "Consuming a Pagelet in WebCenter Portal (an Example)"
- Section 62.8.3, "Consuming a Pagelet in WebCenter Interaction (an Example)"
- Section 62.8.4, "Consuming a Pagelet in Oracle WebCenter Sites (an Example)"
- Section 62.8.5, "Consuming WSRP Portlets as Pagelets"
- Section 62.8.6, "Consuming WebCenter Portal Services as Pagelets in Sites"
- Section 62.8.7, "Consuming Applications as Pagelets Using EBS11i"
- Section 62.8.8, "Consuming OpenSocial Gadgets Using Pagelet Producer"
- Section 62.8.9, "Manipulating HTML Markup"
- Section 62.8.10, "Advanced URL Rewriting"

62.8.1 Creating a Simple Pagelet (an Example)

This section will give you a basic feel of how you can use Pagelet Producer to proxy a Web page. We will proxy a simple static "Hello World" Web page, cut one section out of that page, and present it as a pagelet that you can later insert in WebCenter Portal, WebCenter Interaction, on your own application page.

This section includes the following subsections:

- Section 62.8.1.1, "Configuring the Initial Pagelet Producer Setup"
- Section 62.8.1.2, "Creating a Resource"
- Section 62.8.1.3, "Creating a Pagelet"
62.8.1.1 Configuring the Initial Pagelet Producer Setup

For this example, let's assume that the Pagelet Producer server is running on http://pageletserver.company.com:8889/pagelets/.

1. First, let's check that Pagelet Producer is up and running.
   To do that we just need to access its URL
   (http://pageletserver.company.com:8889/pagelets/). Figure 62–32 shows what
   should be returned:

   ![Pagelet Producer Welcome Page](image)

   Figure 62–32 Pagelet Producer Welcome Page

2. Access the Pagelet Producer administration screens using this URL:
   http://pageletserver.company.com:8889/pagelets/admin

   ![Pagelet Producer Administration Screen](image)

   Figure 62–33 Pagelet Producer Administration Screen

3. If you connect to the internet via a proxy server, you need to configure proxy in
   the Pagelet Producer settings:
   a. In the Navigator pane's Jump To drop down list, select Settings.
   b. Click Proxy.
   c. Enter your proxy server configuration as shown in Figure 62–34.
62.8.1.2 Creating a Resource

The first thing that you need to do is to create a resource for your Web page. This will tell Pagelet Producer that all sub-paths of the Web page should be proxied. It also will allow you to set up common rules for how your Web page should be proxied and will serve as a container for your pagelets.

1. From the Navigator pane's Jump To drop down list select Resources.
2. Click any existing resource (for example, welcome_resource).
3. Click Create selected type.
4. From the Select Producer Type dialog, select Web and click OK.
5. After the resource is created, click the General node in the Navigation pane and specify the following values as shown in Figure 62–35:
   - Name: AppServer
   - Source URL: http://appserver.company.com:1234/
   - Destination URL: /appserver/

6. Click Save.
After the resource is created our Web page becomes accessible by the URL:
http://pageletserver.company.com:8889/pagelets/appserver/helloworld/

Figure 62–36  Hello World Pagelet

The original Web page address Source URL has now been replaced with the Pagelet Producer URL (http://pageletserver.company.com:8889/pagelets) + Destination URL.

62.8.1.3 Creating a Pagelet
Let's continue by creating our "Hello World" pagelet.

1. Under the Resource node, open the Pagelets node.
2. Click Create selected type.
3. Click the General node of the newly created pagelet and specify the following values as shown in Figure 62–37:
   
   **Name:** Hello_World
   **Library:** MyLib

   Library is used for logical grouping. The portals use the Library value to group pagelets in their respective UIs. For example, when adding pagelets to a WebCenter Portal portal you would see the individual pagelets listed under "Library".

   **URL Suffix:** helloworld/index.html

   The URL Suffix is where the Hello World page HTML is served from.
4. Click Save.

The Library name can be anything you want, it doesn't have to match the resource name at all. It is used as a logical grouping of pagelets, and you can include pagelets from multiple resources into the same library or create a new library for each pagelet.

After you save the pagelet you can access it here:

http://pageletserver.company.com:8889/pagelets/inject/v2/pagelet/MyLib/Hello_World

which is:

http://pageletserver.company.com:8889/pagelets/inject/v2/pagelet/ + [Library] + [Name]

Or to test the injection of a pagelet into iFrame you can click on the pagelet's Documentation node and use the Access Pagelet using REST URL.
If you click the URL on the Documentation page you should see the following:

**Figure 62–38 Pagelet’s Documentation Page**

![Pagelet's Documentation Page](image)

![Pagelet's Documentation Page](image)

62.8.1.4 Clipping the Pagelet

The pagelet that we just created would cover the whole Web page. Since we only want the "Hello World" segment of it we’ll need to clip it as described in the following steps:

1. Under the **Hello_World** pagelet node, click **Clipper**.
2. Click **Create selected type**.
3. Specify a Name for the newly created clipper (for example: c1).
4. Click the clipper’s **Content** node and click **Launch Clipper**.
5. In the browser window, use the mouse to select the area you want to clip.
   When you click the mouse button the browser window disappears and a Clipping Path is automatically generated.

6. Save the clipped pagelet and access the link from the Documentation page again.

7. Our pagelet is now nicely clipped and ready to be used in a WebCenter Portal portal as shown in Figure 62–41:

62.8.2 Consuming a Pagelet in WebCenter Portal (an Example)

This section provides an example of how you can consume a simple "Hello World" pagelet in WebCenter Portal. Before continuing, first follow the instructions in Section 62.8.1, "Creating a Simple Pagelet (an Example)" to create the "Hello World" pagelet that will be used in this example.
This section includes the following subsections:

- Section 62.8.2.1, "Registering the Pagelet Producer with WebCenter Portal"
- Section 62.8.2.2, "Inserting the Pagelet into a Portal"

62.8.2.1 Registering the Pagelet Producer with WebCenter Portal

In order to use our newly created pagelet from WebCenter Portal, we first need to register Pagelet Producer:

1. Log into WebCenter Portal as an administrator.
2. Click Administration.
3. Open the Configuration tab.
4. Click Services.
5. Click Portlet Producers.
6. Click Register.
7. Select Pagelet Producer and enter the values shown for the following fields:
   - Producer Name: MyPageletProducer
   - Server URL: http://pageletserver.company.com:8889/pagelets/
8. Click Test to make sure that the connection was correctly configured.
   If everything is successful you should see the Test Status dialog display a message indicating that the test connection was successful.
9. Click OK.
   The Pagelet Producer is now registered.

62.8.2.2 Inserting the Pagelet into a Portal

For our exercise we will need to create a portal. So let's start by logging into WebCenter Portal and creating a portal called "myportal" using "Portal Site" template.

1. Log into WebCenter Portal as an administrator.
2. Create a portal called myportal using the Portal Site template.
3. On the myportal portal click on the Actions icon and select Edit Page.
4. Click Add Content in one of the portal frames and select Mash-Ups.
5. Select Pagelet Producers.
6. Click the Pagelet Producer you previously registered.
   You will see the library MyLib containing the Hello_World pagelet.
7. Click MyLib, select the Hello_World pagelet and click Add.
8. Click Close.
9. Click Save, and then click Close.
   The Hello World pagelet is now inserted into the myportal page.

62.8.3 Consuming a Pagelet in WebCenter Interaction (an Example)

This section provides an example of how you can consume a simple "Hello World" pagelet in WebCenter Interaction (WCI) 10.3.0 or later. Before continuing, first follow
the instructions in Section 62.8.1, "Creating a Simple Pagelet (an Example)" to create the "Hello World" pagelet that will be used in this example.

This section includes the following subsections:

- Section 62.8.3.1, "Registering the Pagelet Producer Remote Server"
- Section 62.8.3.2, "Creating the "Hello World" Web Service"
- Section 62.8.3.3, "Creating the "Hello World" Portlet"
- Section 62.8.3.4, "Using the Hello World Portlet on the WCI Home Page"

62.8.3.1 Registering the Pagelet Producer Remote Server

In order to use our newly created pagelet from WCI, we first need to register Pagelet Producer:

1. Log into WCI as an administrator.
2. Click Administration.
3. Create the folder where we are going to keep all Pagelet Producer related objects by following the steps below:
   a. Open the Create Object... dropdown and select Administrative Folder.
   b. Enter PageletProducer as the Name and click OK.

Figure 62–42  WCI - Create Administration Folder

4. Click the newly created PageletProducer folder.
5. Open the Create Object... dropdown and select Remote Server.

6. In the Base URL field, enter http://pageletserver.company.com:8889/pagelets (this is the address of our Pagelet Producer server in this example).

7. Click Finish.

8. Select the PageletProducer folder and in the Save As field enter Pagelet Producer Remote Server.
9. Click **Save**.

You’ve now created a Pagelet Producer Remote Server connection in the PageletProducer folder.

---

**Figure 62–45  WCI - Saving the Pagelet Producer Remote Server Object**

![Pagelet Producer Remote Server Object]

**Save In**
- Security

**Save into the selected folder below:**
- Administrative Resources
- Default Experience Definition
- Experience Definition Objects
- Introsel Operations
- Page Layouts
- PageletProducer
- Portal Resources

---

**Figure 62–46  WCI - Pagelet Producer Remote Server Object**

![Pagelet Producer Remote Server Object]

This folder contains no subfolders.

- **PageletProducer Remote Server**
  - Last Modified: 12/2/11 4:09 PM
62.8.3.2 Creating the "Hello World" Web Service
In this section we'll continue by creating the "Hello World" Web service.

1. Open Create Object... dropdown and select Web Service - Remote Portlet.
2. Click Browse and select Pagelet Producer Remote Server.

Figure 62–47  WCI - Choose Remote Server Dialog

3. Click OK.
4. In the Portlet URL field enter:
   
   inject/v2/portlet/MyLib/Hello_World?content-type=iframe&csapi=true&ifheight=300px

   Where MyLib is a library that contains our "Hello World" pagelet and Hello_World
   is the name of our pagelet.
5. Click Finish.

6. Select the PageletProducer folder and in the Save As field enter Hello World Web Service.

7. Click Save.

   You've now created the Hello World Web service.

### 62.8.3.3 Creating the "Hello World" Portlet

In this section we'll continue by creating a Hello World portlet.

1. Open the Create Object... dropdown list and select Portlet.

   The Choose Template or Web Service dialog displays (Figure 62–49).
2. From the list of templates or Web services select Hello World Web Service and click OK.

3. Under Portlet Type/Size, select the portlet type and size you want to use and click Finish.

4. Select the PageletProducer folder.

5. In the Save As field, enter Hello World Portlet and click Save.

You've now created the Hello World portlet.

62.8.3.4 Using the Hello World Portlet on the WCI Home Page

Now that we've registered the Pagelet Producer, set up the Pagelet Producer Web service, and created the Hello World portlet, let's continue by using the portlet on a page in WCI. For the sake of simplicity, we'll use the WCI Home Page, but you could use any other WCI page.

1. In WCI, navigate to My Pages > Home Page.
2. Click Edit Page and open the PageletProducer folder.
3. Click Hello World Portlet.
4. Click Close Editor.

Here's our Hello World pagelet inserted into the Home Page in WCI:
62.8.4 Consuming a Pagelet in Oracle WebCenter Sites (an Example)

This section is aimed at developers who need to integrate content into Oracle WebCenter Sites 11g pages, including existing WSRP portlets or elements of Web UI exposed by backend applications such as Oracle EBS. Developers should be familiar with Oracle WebCenter Sites and Oracle WebCenter Pagelet Producer and have a solid understanding of Web technologies.

This section includes the following subsections:

- Section 62.8.4.1, "Adding Pagelets to Oracle WebCenter Sites"
- Section 62.8.4.2, "Using Identity Propagation"
- Section 62.8.4.3, "Propagating Identity from Pagelet Producer to the Backend Application"
- Section 62.8.5, "Consuming WSRP Portlets as Pagelets"
- Section 62.8.6, "Consuming WebCenter Portal Services as Pagelets in Sites"
- Section 62.8.7, "Consuming Applications as Pagelets Using EBS11i"

### 62.8.4.1 Adding Pagelets to Oracle WebCenter Sites

Once a pagelet and its related resources are configured, you can insert it into a Web page using JavaScript or REST. For more information, see Section 62.7.2, "Adding a Pagelet to a Web Page."

Here, we'll use an approach in which pagelets are added directly to a page template in Oracle WebCenter Sites using an iFrame with a REST URL for accessing pagelets as a content source. This requires an addition to the page template script tag to load the CSAPI JavaScript library (which adds necessary functions into the parent page), as well as the actual iFrame that loads pagelet content:
<script type="text/javascript" src="http://%PAGELET_PRD_HOST%/pagelets/inject/v2/csapi"/>
<iframe id='p-pagelet-iframe-1' width='100%' frameborder='0' src='http://%PAGELET_PRD_HOST%/pagelets/inject/v2/pagelet/lib_name/pagelet_name?content-type=html&consumepage=true&ifheight=auto'></iframe>

Where lib_name and pagelet_name refer to the library and pagelet configured in Pagelet Producer. For more information about parameters, see Section 62.7.2.2, "Accessing Pagelets Using REST."

62.8.4.1.1 Enabling IFrame Auto-Resizing To provide a more seamless integration of the pagelet into the consuming page, you can make the pagelet iFrame behave more like inline markup by dynamically resizing to accommodate its content. To enable this feature, some extra configuration is required in both Sites and Pagelet Producer.

This section uses the AviSports sample site shipped with WebCenter Sites. To add any new asset to this site, you must log in to the Sites Administration page and open the Dev section in the left hand navigation as shown in Figure 62–51:

Figure 62–51 Sites Administration Page

First, create a new template with the following settings:

- Name
  - Name: iFrameResizeRelay (or name of your choice)
  - For Asset Type: Applicable to various asset types (typeless)
- Element
  - Usage: Element defines a whole HTML page and can be called externally
  - Element Storage Path/Filename: iframe-resize.html (or name of your choice)
  - Element Logic:
<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>

- Site Entry
  - Cache Rules: Cached (default)

- Save

- Record SiteCatalog Pagename as %PAGENAME% (if you used the default name, this would be <site_name>/iFrameResizeRelay)

The new template is addressable as follows:
http://%SITES_HOST%/cs/Satellite?pagename=%PAGENAME%

Make a note of this URL; it will be used to configure the Injector for the pagelet.

Next, apply the iFrame resizing feature by adding a new Injector to the pagelet using the Pagelet Producer Administration Console. Open the pagelet, select Injectors and create a new Injector with the following settings:

- General
  - Name: auto_resizer (or name of your choice)
  - URL Filter: <none>
Examples and Advanced Topics

Creating Pagelets with Pagelet Producer

- MIME Filter: text/html
- Inject Location: Before </head>

Content: Select Text (default) and copy the JavaScript below into the text area

Replace SITES_RESIZE_RELAY_PAGE in the JavaScript below with the URL to the template you created in the previous step.

```javascript
<script type="text/javascript">
var SITES_RESIZE_RELAY_PAGE = "http://%SITES_HOST%/cs/Satellite?pagename=%PAGENAME%"; // CHANGE ME!

if (window.addEventListener) {
    window.addEventListener('load', calculateSizeFixed, false);
} else if (window.attachEvent) {
    window.attachEvent('onload', calculateSizeFixed);
} else {
    window.onload=calculateSizeFixed;
}

function calculateSizeFixed() {
    var PTResizeIFrame = PTResizeIFrame || {};
    if (PTPortalPage && PTPortalPage.portlets) {
        for (var i in PTPortalPage.portlets) {
            if ( PTPortalPage.portlets[i].id != 'page' ) {
                PTResizeIFrame.pageletInstanceID = PTPortalPage.portlets[i].id;
                break;
            }
        }
    } else if (!PTResizeIFrame.pageletInstanceID) {
        PTResizeIFrame.pageletInstanceID = 1;
    }

    if (!PTResizeIFrame.pageletResizePage) {
        var match = window.location.search.match(/resizepage=.*$/);
        if (match != null && match.length > 0) PTResizeIFrame.pageletResizePage = unescape(match[0].substr('resizepage='.length));
        else PTResizeIFrame.pageletResizePage = SITES_RESIZE_RELAY_PAGE;
    }

    var agent = navigator.userAgent;
    var ffversion = agent.indexOf('Firefox') >= 0 ? agent.substring(agent.indexOf('Firefox')).split('/').length : -1;
    var FFextraHeight = parseFloat(ffversion) >= 0.1 ? 25 : 0;
    var scrollHeightExtra = 15;
    if (agent.indexOf('MSIE') > 0) {
        scrollHeightExtra = 35;
    }

    if (FFextraHeight > 0) scrollHeightExtra = FFextraHeight;
    var wrapper = document.getElementById('pt-inner-iframe-wrapper-' + PTResizeIFrame.pageletInstanceID);
    if (wrapper == null) wrapper = createRelayIFrame(PTResizeIFrame.pageletInstanceID);
    var iframe = document.getElementById('pt-inner-iframe-' + PTResizeIFrame.pageletInstanceID);
    var height = 0;
    var width = 0;
    var iframename = '';

    if( (document.contentDocument) && (document.contentDocument.documentElement.offsetHeight) ) {
        height = document.contentDocument.documentElement.offsetHeight +
```
Add the pagelet to the page template in Sites using a REST URL within an IFrame. In the IFrame, the id parameter should use the unique number identifying the pagelet IFrame (for example, "pt-pagelet-iframe-1"). You must add the following query string parameters to the pagelet URL to support IFrame auto-resizing:

- **resizepage**=http://%SITES_HOST%/cs/Satellite?pagename=%PAGENAME%
- **ifheight**=auto

For example:

```
<script type="text/javascript" src="http://%PAGELET_PRD_HOST%/pagelets/inject/v2/csapi" />
<iframe id="pt-pagelet-iframe-1" width="100%" frameborder="0"
src="http://%PAGELET_PRD_HOST%/pagelets/inject/v2/pagelet/lib_name/pagelet_name?content-type=html&consumepage=true&resizepage=http://%SITES_HOST%/cs/Satellite?pagename=%PAGENAME%&ifheight=auto">
</iframe>
```

**Note:** In certain situations automatic resizing may not work properly when multiple pagelets are present on a page. This is known to occur with pagelets that require form auto-login to external authentication servers. In this case, only one pagelet can be configured for auto-resizing, while the others should use static IFrame sizing.
62.8.4.1.2 Changing Pagelet Styling To make a pagelet fit better visually in a consuming WebCenter Sites page, an Injector may be used to add styles that override the original CSS. In order for the override to work correctly, it is important that the style definitions supplied by the Injector come after the styles defined by the backend application. The following example does this by injecting replacement CSS into the end of the <HEAD> section.

To restyle a pagelet, add a new Injector to the pagelet using the Pagelet Producer Administration Console. Open the pagelet, select Injectors and create a new Injector with the following settings:

- **General:**
  - Name: new_styles (or name of your choice)
  - URL Filter: <none>
  - MIME Filter: text/html
  - Injector Location: Before </head>

- **Content**

Section 62.8.7, "Consuming Applications as Pagelets Using EBS11i" includes a working example of an Injector used to re-style the EBS UI to match the sample AviSports site in WebCenter Sites.

62.8.4.2 Using Identity Propagation

The core purpose of Pagelet Producer is to proxy backend applications. Since Pagelet Producer acts as a "middle-man" between the browser and the backend application, identity propagation must be broken into two parts:

- **Section 62.8.4.2.1, "Establishing Identity in Pagelet Producer"
- **Section 62.8.4.2.2, "Using a Login Page Supplied by Pagelet Producer"**

62.8.4.2.1 Establishing Identity in Pagelet Producer Pagelet Producer typically injects Web content into WebCenter Sites using an IFrame as shown in Figure 62–52.

*Figure 62–52  Pagelet Producer Content as IFrame in Sites*

Since the content is injected as an IFrame, user identity must be established in both the Sites container and the Pagelet Producer container as shown in Figure 62–53. Note that Figure 62–53 leaves out the user directory that is assumed to be shared between the two authentication schemes.
The ideal way to manage identity between the browser and the two containers would be to use Oracle Access Manager (OAM) as shown in Figure 62–54. Note that Figure 62–54 leaves out the user directory that is assumed to be shared between the two authentication schemes, and excludes the OAM access server. Note also that this configuration is not supported for pre-11g versions.

To set up OAM for WebCenter Sites 11g, refer to the chapter on "Oracle Access Manager Integration Setup" in Oracle WebCenter Sites Configuring Supporting Software.

### 62.8.4.2.2 Using a Login Page Supplied by Pagelet Producer

Before setting up OAM, it is sometimes useful to test the integration of Pagelet Producer in Sites. Without OAM, users must authenticate separately with Pagelet Producer and with Sites (see Figure 62–53). This section describes how to configure separate authentication with Pagelet Producer.

If OAM SSO is not present on either Pagelet Producer or Sites, the authentication scheme shown in Figure 62–55 must be supported by ensuring that Pagelet Producer provides a login form.

To force Pagelet Producer to display a login form:

1. Identify a J2EE role to restrict access to the pagelet(s) or create a new role in the J2EE application server hosting Pagelet Producer.

   This example uses the global WebLogic Server (WLS) "Anonymous" J2EE role that is present in a default WLS installation.
2. Log in to the Pagelet Producer Administration Console, navigate to the resource(s) containing the pagelet(s) that must be surfaced in Sites and add the role you chose in step 1.

Roles are defined on the Policy page for the resource.

**Figure 62–55 Pagelet Producer Administration Console - Policy Page**

62.8.4.3 Propagating Identity from Pagelet Producer to the Backend Application

The way in which identity is established depends on the type of application, as described in the following sections:

- Section 62.8.4.3.1, "WSRP/JPDK Portlets"
- Section 62.8.4.3.2, "Stand-alone Backend Application Protected with Native Authentication (Non-SSO)"
- Section 62.8.4.3.3, "Identity Propagation with Autologin"
- Section 62.8.4.3.4, "Stand-alone Backend Application Protected with SSO"
- Section 62.8.4.3.7, "Reference: Common Autologin Configurations"

62.8.4.3.1 WSRP/JPDK Portlets

WSRP portlets must use WSS tokens for identity propagation. For details on configuring security tokens, see the "Registering WSRP and Oracle JPDK Portlet Producers in the Pagelet Producer" section in the Administering Oracle WebCenter Portal, and Section 62.8.5, "Consuming WSRP Portlets as Pagelets."

Each Oracle JPDK portlet that requires an identity will be passed credentials based upon information supplied by the external application's login form. For details, see the "Managing External Applications" and "Registering Oracle JPDK Portlet Producers" sections in Administering Oracle WebCenter Portal.

62.8.4.3.2 Stand-alone Backend Application Protected with Native Authentication (Non-SSO)

Pagelet Producer has a feature called Autologin that allows Pagelet Producer to interact with the "native" authentication mechanism established by a backend application.

62.8.4.3.3 Identity Propagation with Autologin

Since each backend application has its own means of authentication, a separate login prompt must be initiated by Pagelet Producer to collect any required credentials. Pagelet Producer resources can be configured to send credentials to backend applications as basic authentication headers, NTLM tokens, or Kerberos tokens, as well as manage forms-based authentication.
(typically via a session cookie). For information about configuring Autologin, see Section 62.3.1.1.4, "Configuring Autologin."

---

**Note:** Set the **Username** and **Password** fields to use the User Vault so that each user will be prompted for his or her own unique credentials to access the backend application.

---

### 62.8.4.3.4 Stand-alone Backend Application Protected with SSO

Determining the proper Autologin settings for a backend application can be very challenging. Doing so requires looking at request headers, identifying redirects, and examining markup (sometimes dynamically generated). This inspection and configuration process can be daunting if there are multiple applications for which Autologin is required. Therefore, if Oracle Access Manager (OAM) or Oracle SSO (OSSO) is available, it is highly recommended that one of these single sign-on solutions be used to protect backend applications.

Once the backend application is protected with OAM, there are two ways that Pagelet Producer can supply credentials as described in the following subsections:

- **Section 62.8.4.3.5, "Using Direct Identity Propagation to OAM Protected Backend Application"**
- **Section 62.8.4.3.6, "Using Identity Propagation with Autologin and SSO"**

### 62.8.4.3.5 Using Direct Identity Propagation to OAM Protected Backend Application

An OAM 11g WebGate is an Apache module running on Oracle HTTP Server (OHS) that manages all validation of user credentials with the OAM server. Once it has established the validity of a user it sends the OAM_REMOTE_USER header to the Web application on the application server that OHS is proxying. The application server will have an OAM identity asserter running on it that will set the JAAS user principal name to the OAM_REMOTE_USER header value.

If Pagelet Producer is protected with an OAM 11g WebGate it will receive an OAM_REMOTE_USER header. All Web applications that it proxies will also receive this header. If the Web application it proxies is on an application server with the OAM Identity Asserter, then the OAM Identity Asserter will use the OAM_REMOTE_USER header value passed from Pagelet Producer to set the user principal. This will all work as long as Pagelet Producer is not trying to proxy the URL to the WebGate for the protected Web application. The rest of this section describes in details how this can be achieved in the context of Sites integration.

#### Example Setup

This example assumes that both WebCenter Sites and Pagelet Producer are protected by OAM WebGate as illustrated in Figure 62–54 above, and the following hosts and ports are used by the applications:

- **OAM SSO WebGate:** `www.example.com:80`
- **WebCenter Sites:** `sites_host:8080`
- **WebCenter Pagelet Producer:** `pp_host:8889`
- **Backend application:** `backendhost:8001`
- **Full URL to backend application as viewed from browser (via WebGate):**

  `http://www.example.com/backend/console`
- Full URL to backend application if accessed directly through application server (available only in internal network) http://backendhost:8001/console

Set up the Pagelet Producer resource to use the direct URL to the backend application, thus bypassing the WebGate (i.e., http://backendhost:8001/console).

Figure 62–56  Pagelet Producer Administration Console - General Properties Page

By setting up Pagelet Producer to call directly into the backend application (bypassing the WebGate) all the OAM secure headers that Pagelet Producer receives are passed to the OAM Identity Asserter on the application server hosting the backend application. Once the OAM Identity Asserter on the application server hosting the backend application receives the OAM headers it will set the proper user principal (see Figure 62–57). Note that Figure 62–57 leaves out the user directory that is assumed to be shared between the two authentication schemes. It also excludes the OAM access server.

Figure 62–57  Pagelet Producer Calling the Backend Application Directly
62.8.4.3.6 Using Identity Propagation with Autologin and SSO

If bypassing the OAM WebGate (as described in Section 62.8.4.3.5, "Using Direct Identity Propagation to OAM Protected Backend Application") is not a feasible option then credentials can still be supplied using Pagelet Producer's Autologin feature. The advantage of using Autologin with an Oracle SSO solution (OAM or OSSO) login page as opposed to individual login forms for each backend application (see Section 62.8.4.3.3, "Identity Propagation with Autologin") is that Autologin only has to be set up once and the configuration steps are the same for every implementation.

The steps for setting up Autologin with OSSO are described in Section 62.8.7.2, "Configuring Autologin." In a production environment, these steps can be followed as is with one exception: the ssousername and password form fields should be linked to the "User Vault".

Backend applications protected with OAM can also be set up following the steps described for OSSO in the Configure Autologin section in this document with the following exceptions:

- The login form URL will be a set location (e.g., http://oam11g.mycompany.com:14100/oam/server/obrareq.cgi).
  Be sure to remove the query string after the fixed location argument when setting the Autologin form identification URL.
- The form action URL will be a different value, but it should be a static value. Inspect the OAM form to determine the form action URL.
- The username and password fields may be different values. Inspect the OAM form fields to determine these values.
- Include the following two generated values: "request_id" and "OAM_REQ".

62.8.4.3.7 Reference: Common Autologin Configurations

This section provides a quick reference of Autologin settings in Pagelet Producer for common backend SSO scenarios.

This section contains the following subsections:

- Section 62.8.4.3.8, "Autologin Settings for Oracle Access Manager (OAM)"
- Section 62.8.4.3.9, "Autologin Settings for Oracle SSO (OSSO)"

62.8.4.3.8 Autologin Settings for Oracle Access Manager (OAM)

Login Form Identification:

- URL: http://%OAM_ROOT%/server/obrareq.cgi

Form Submit Location:

- URL: /oam/server/auth_cred_submit
Form Fields:
- **actionURL**: static: /oam/server/auth_cred_submit
- **request_id**: Generated
- **username**: Unsecured or Credential Vault
- **OAM_REQ**: Generated
- **password**: Unsecured or Credential Vault

### 62.8.4.3.9 Autologin Settings for Oracle SSO (OSSO)

**Login Form Identification:**
- **URL**: %OSSO_HOST%/sso/jsp/login.jsp

**Form Submit Location:**
- **URL**: %OSSO_HOST/sso/auth - POST

**Form Fields:**
- **appctx**: Generated
- **locale**: Generated
- **password**: Unsecured or Credential Vault
- **site2pstoretoken**: Generated
- **ssousername**: Unsecured or Credential Vault
- **v**: Generated

### 62.8.5 Consuming WSRP Portlets as Pagelets

Pagelet Producer incorporates the WebCenter Common Consumer and therefore can expose WSRP and Oracle JPDK portlets as pagelets for use in WebCenter Sites or any other Web container that does not include a WSRP consumer.

The WSRP Portlet Producer can be registered with Pagelet Producer using Fusion Middleware Control, WLST, or the Pagelet Producer Administrative Console. For more information, see the "Registering WSRP and Oracle JPDK Portlet Producers in Pagelet Producer" section in *Administering Oracle WebCenter Portal*. Once registration is complete, the WSRP producer will appear as a resource in the Pagelet Producer Administrative Console, and the portlets associated with the WSRP endpoint will be listed in the pagelets collection for the resource.

---

**Note:** Auto-generated WSRP resources and pagelets cannot be modified. To create a version that can be modified, choose the resource in the Pagelet Producer Administrative Console and click **Copy**. The cloned version of the resource can be edited and various elements such as Injectors can be added to customize pagelet functionality. For example, a custom Injector could be used to inject CSS classes to modify portlet markup to make it look and feel like the Sites that will host the pagelet.

---

If the WSRP Producer that you are registering with Pagelet Producer requires authentication, the following steps are required:
Configure WS-Security between Pagelet Producer and the WSRP Producer as described in Configuring WS-Security in the Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter Portal. You must ensure that the Java Key Store (JKS) is properly configured between the WSRP producer and Pagelet Producer; for details, see the “Setting Up the WebCenter Portal Domain Keystore” section in Administering Oracle WebCenter Portal.

During the WSRP Producer registration, select the appropriate Token Profile in the Security section and enter the necessary configuration information. The path to the keystore must be absolute.

62.8.5.1 Exposing Custom ADF Task Flows as WSRP Portlets

Although it is possible to consume an ADF task flow in Pagelet Producer by ‘clipping’ it out of the page on which it is hosted, this is not the recommended approach. Considering the complexity of the markup generated by an ADF page and potential dependencies on other task flows or ADF page parameters, the recommended approach is to create a WSRP portlet based on the task flow and then consume the portlet in Pagelet Producer. One advantage of this approach is that ADF task flow parameters become portlet parameters that can be mapped to pagelet parameters, allowing information to be passed from the Sites page to the task flow. For step-by-step instructions on how to make a portlet from an ADF task flow, see Section 58.2.3, "How to Create a JSF Portlet Based on a Task Flow."

After the ADF task flow is exposed as a portlet, deploy a new WSRP Producer to a managed server in your WLS domain (for example, the WC_Portlet server on your WebCenter domain). Obtain the WSDL URL from the info page (http://%PRODUCER_HOST%/APP_CONTEXT%/info) and make note of it. To register the new WSRP Producer with Pagelet Producer, see the "Registering WSRP and Oracle JPDK Portlet Producers in Pagelet Producer" section in Administering Oracle WebCenter Portal.

62.8.5.1.1 Exposing WSRP Portlets Developed for Oracle WebLogic Portal

Oracle WebLogic Portal (WLP) supports a variety of portlet types, including Java Portlets, Java Server Faces Portlets, Java Server Page and HTML Portlets (for a complete list see the "Portlet Types" section in Oracle Fusion Middleware Portlet Development Guide for Oracle WebLogic Portal). Remote (Proxy) Portlets are WSRP portlets and can be directly exposed as pagelets by registering the WSRP Producer with Pagelet Producer. This section describes how to expose local WLP portlets such as JSF, JSP or Java Portlets as pagelets for use in Sites.

62.8.5.1.2 Exposing WLP Portlets Using WLP as WSRP Producer

WLP can expose Java, Page Flow, JSP, JSF, and Struts Portlets as a "complex producer" that includes the required WSRP interfaces, optional interfaces and some extended interfaces (for details see "WebLogic Portal Producers" in the Oracle Fusion Middleware Federated Portals Guide for Oracle WebLogic Portal). Consequently, native WLP portlets can be consumed by Pagelet Producer as any other WSRP portlets. Configuration and best practices for WSRP interoperability between WLP and the WebCenter Common Consumer are described in the "WSRP Interoperability with Oracle WebCenter Portal and Oracle Portal" section in Oracle Fusion Middleware Federated Portals Guide for Oracle WebLogic Portal.

Note: WLP portlets are WSRP-enabled by default since version 9.2. In earlier versions, you must edit the portlet properties and set the "Offer as Remote" property to true.
62.8.5.1.3 Configuring WS Security between WLP WSRP Producer and WebCenter Consumer

To consume WSRP portlets exposed by WLP in the WebCenter Common Consumer (Pagelet Producer), configuration is required on both the producer and consumer sides.

Typically it is impossible to consume WSRP portlets exposed by WebLogic Portal as an anonymous user, so it is necessary to configure SAML security on WLP for the producer and the WebCenter Common Consumer. Step-by-step instructions can be found in the "SAML Security Between a WebCenter Portal: Framework Application Consumer and a WebLogic Portal Producer" section in Oracle Fusion Middleware Federated Portals Guide for Oracle WebLogic Portal. These instructions are comprehensive, with the following adjustments:

- In section 5 of "Configuring the Producer", the steps for configuring keystore information on the WSRP consumer are executed in JDeveloper. You must complete this step in the WebCenter Common Consumer (Pagelet Producer). You can specify the signing alias and keystore password using WLST or Fusion Middleware Control. For details, see "Configuring the Keystore Using WLST" or "Configuring the Keystore Using Fusion Middleware Control" in the Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal.

  **Note:** Once you have configured the keystore values you must restart both the managed server hosting Pagelet Producer (by default WC_Portlet) and the WLS Admin Server for the JPS configuration changes to be picked up.

- In section 4 of "Configuring the Producer", the instructions do not emphasize the importance of the issuer name. The name placed here must match that sent by the consumer. For Oracle JPS, the default issuer ID sent is www.oracle.com.

62.8.5.1.4 Registering the WLP WSRP Producer in Pagelet Producer

To register the WLP WSRP Producer in Pagelet Producer, follow the standard steps for creating and configuring a new WSRP Producer described in the "Registering WSRP and Oracle JPDK Portlet Producers in Pagelet Producer" section in Administering Oracle WebCenter Portal. Make sure to include the following settings:

- **WSDL URL:**
  
  http://%WLP_HOST%/%PORTAL_APP_NAME%/producer/wsrp-1.0/markup?WSDL

- **Security:**
  
  Token Profile: Select "WSS 1.0 Token with Message Integrity"

After registration, a new Pagelet Producer resource is automatically created and populated with pagelets to represent the WLP portlets associated with this WSRP endpoint.

  **Note:** To allow user identity propagation to the portlets, both WLP and Pagelet Producer should be configured as described in Section 62.8.4.3, "Propagating Identity from Pagelet Producer to the Backend Application." (For testing purposes or for portlet content that can be accessed without authentication, you can specify a valid username in the Default User field under Security.)
62.8.5.1.5 Adding WLP WSRP Portlets to Sites

To add WLP WSRP portlets to a page in Sites, follow the steps in Section 62.8.4.1, "Adding Pagelets to Oracle WebCenter Sites." The URL to use for the src attribute in the IFrame tag that loads pagelet content can be found under "Documentation" in Section 62.7.2.2, "Accessing Pagelets Using REST."

Auto-generated WSRP resources and pagelets cannot be modified. To use Pagelet Producer features to alter the markup of the WLP portlet such as modify look and feel of the portlet UI by injecting custom CSS styles, you must create a new version of the resource. Choose the resource that was created for your WLP WSRP Producer in the Pagelet Producer Administrative Console and click Copy. The cloned version can be modified, including adding elements such as Injectors to customize pagelet functionality.

62.8.6 Consuming WebCenter Portal Services as Pagelets in Sites

WebCenter Portal includes services and tools, such as announcements and documents, that can be used to facilitate user collaboration in WebCenter Sites. For an overview of these tools, see the "Part IV" and "Part V" parts in Using Oracle WebCenter Portal.

The recommended approach for exposing these services and tools in Sites is to consume the associated WSRP portlets in Pagelet Producer by leveraging the Services Producer component included with Oracle WebCenter Portal (11.1.1.6 and later).

The following portal tools are exposed as WSRP portlets by default:

- Activity Stream
- Discussion Forums
- Mail
- Polls Manager
- Document Manager
- Lists
- Tag Cloud
- Blogs
- Work list
- Polls
- Announcements

**Note:** Additional tools can be exposed by extending the Services Producer to a new JSPX page and then adding the taskflow to the page.

62.8.6.1 Requirements

Install WebCenter Portal following the instructions in the Oracle Fusion Middleware Installation Guide for Oracle WebCenter. Make sure that all dependent components are also installed and configured, including WebCenter Content Server, the discussions server, and Pagelet Producer. For more information, see Section 62.1.3, "Requirements."
62.8.6.2 Configuring Security and Single Sign-On

There are two prerequisites to allow access to WebCenter Portal services from a Sites-driven Web site:

- Create a common user base between the Web site and WebCenter Portal
- Establish user identity propagation from the Web site to WebCenter Portal

These are described in the following subsections:

- Section 62.8.6.2.1, "Creating a Common User Base: LDAP Integration"
- Section 62.8.6.2.2, "Establishing User Identity Propagation: OAM Configuration"
- Section 62.8.6.2.3, "Configuring the GUID Attribute in the Identity Store"
- Section 62.8.6.2.4, "Configuring SSO for Discussions Server"

62.8.6.2.1 Creating a Common User Base: LDAP Integration

Creating a common user base requires configuring a common LDAP repository to be used by WebCenter Sites and all WebCenter Portal components. When selecting an LDAP server, make sure it is compliant with the WebLogic Server that runs WebCenter Portal components.

Things to remember when configuring security on WLS:

- Set the priority of the Authenticators to SUFFICIENT
- Set the Control Flag (in parenthesis) and order of authentication providers as follows:
  - OAM Identity Asserter (REQUIRED)
  - LDAP Authenticator (SUFFICIENT)
  - Default Authenticator (SUFFICIENT)
  - Default Identity Asserter (SUFFICIENT)

62.8.6.2.2 Establishing User Identity Propagation: OAM Configuration

To establish user identity propagation, we recommend configuring the SSO solution for WebCenter Sites and WebCenter Portal using Oracle Access Manager (OAM).

If you are using OAM, make sure it is configured to pass the user principal name in the ORACLE_REMOTE_USER header. Both ObSSOCookie and OAM_REMOTE_USER must be set to active on the OAM Identity Asserter on the WebLogic Server.

62.8.6.2.3 Configuring the GUID Attribute in the Identity Store

Specify the GUID attribute value to ensure that the value that is used in the identity store matches the value configured in the LDAP authentication provider. This value is configured in the jps-config.xml file.

Set the GUID for your LDAP authenticator in the jps-config.xml file.

```xml
<serviceInstance provider="idstore.ldap.provider" name="idstore.ldap">
  <property value="oracle.security.jps.wls.internal.idstore.WlsLdapIdStoreConfigProvider" name="idstore.config.provider"/>
  <property value="oracle.security.idm.providers.stdldap.JNDIPool" name="CONNECTION_POOL_CLASS"/>
  <property value="GUID=uuid" name="PROPERTY_ATTRIBUTE_MAPPINGS"/>
</serviceInstance>
```
62.8.6.2.4 Configuring SSO for Discussions Server

To use discussions, configure the discussions server for SSO before ordering the Authenticators in WebLogic Server (using the Default Authenticator and Default Identity Asserter). It is also possible to use WLST to configure SSO for the discussions server. For details, see the "Configuring the Discussions Server for SSO" chapter in *Administering Oracle WebCenter Portal*.

62.8.6.3 Registering WebCenter Services Exposed as WSRP Portlets

To import the WSRP portlets that expose WebCenter Services into Pagelet Producer, simply register the Services Producer as a WSRP Producer in Pagelet Producer using Enterprise Manager, WLST or the Pagelet Producer Administrative Console.

Figure 62–58 shows an example of WSRP endpoint registration in the Pagelet Producer Administrative Console. For details, see the "Registering WSRP and Oracle JSDK Portlet Producers in Pagelet Producer" sections in *Administering Oracle WebCenter Portal*.

**Figure 62–58  Pagelet Producer Administrative Console - Registering WSRP Endpoint**

The Token Profile in the Security section of this page must be set to WSS 1.0 SAML Token to support the Services Producer. You should also enter a suitable execution timeout for requests made by Pagelet Producer to the Services Producer (the default is 30 seconds).

Once registration is complete, the Services Producer will appear as a resource in the Pagelet Producer Administrative Console and all WSRP portlets exposed by the Producer will be listed in the pagelets collection for the resource:

62.8.6.3.1 Adding Pagelets to Sites Pages

Any pagelet exposing WebCenter Services can be added to a WebCenter Sites page following the steps described in Adding Pagelets to Oracle WebCenter Sites. Figure 62–59 shows the Document Manager Service embedded into a page of the AviSports sample site in WebCenter Sites.
This section provides detailed instructions for consuming the Oracle E-Business Suite 11i Order Information module UI as a pagelet. It covers autologin, navigation suppression, restyling, and URL rewriting.

This example was created using an EBS 11i instance that is protected with Oracle SSO. The following common terms and variables are used in the example:

- `%EBS_HOST%`: root URL of the EBS host (for example, http://my-ebs.company.com:port/)
- `%OSSO_HOST%`: root URL of the OSSO host (for example, http://sso.company.com:port/)

This section contains the following subsections:

- **Note:** Auto-generated WSRP resources and pagelets such as the ones associated with the Services Producer cannot be modified. To add an Injector or make other modifications, you must create a version that can be edited. Choose the resource in the Pagelet Producer Administrative Console and click **Copy**. The cloned version of the resource can be edited and various elements such as Injectors can be added to customize pagelet functionality.

**Figure 62–59  AviSports Sample Site with Document Embedded**

---

62.8.7 Consuming Applications as Pagelets Using EBS11i

This section provides detailed instructions for consuming the Oracle E-Business Suite 11i Order Information module UI as a pagelet. It covers autologin, navigation suppression, restyling, and URL rewriting.

This example was created using an EBS 11i instance that is protected with Oracle SSO. The following common terms and variables are used in the example:

- `%EBS_HOST%`: root URL of the EBS host (for example, http://my-ebs.company.com:port/)
- `%OSSO_HOST%`: root URL of the OSSO host (for example, http://sso.company.com:port/)
62.8.7.1 Creating a Resource for Basic URL Mapping (Proxy)

The following steps set up the basic URL mapping for a new EBS resource in Pagelet Producer:

1. Create new "Web" resource with the following settings:
   - Name: EBS 11 (or your choice)
   - Source URL: %EBS_HOST%
     Make sure the URL is not more specific to avoid missing URLs used in the UI
   - Destination URL: /ebs11/ (or your choice)
   - URL Rewriting: on
   - DHTML Rewriting: on
2. Click Save.

62.8.7.2 Configuring Autologin

This step assumes that the EBS System is protected by OSSO and that Pagelet Producer will be configured to provide form Autologin rather than participate in SSO (a common scenario if the Web site is not protected by SSO). EBS credentials may be shared for all users (useful for testing) or stored in the Pagelet Producer credential vault for each user.

1. Create a new "Web" resource with the following settings:
   - Name: OSSO (or your choice)
   - Source URL: %OSSO_HOST%
   - Destination URL: /osso/ (or your choice)
   - URL Rewriting: on
   - DHTML Rewriting: off
2. Click Save.
3. Under the newly created OSSO resource, configure Autologin using the Form Login option as follows:
   - Login Form Identification: URL - %OSSO_HOST%/sso/jsp/login.jsp
   - Form Submit Location: URL - %OSSO_HOST/sso/auth - POST
   - Form Fields:
     appctx: Generated
     locale: Generated
     password: Unsecured or User Vault
     site2pstoretoken: Generated
     ssousername: Unsecured or User Vault
4. Click Save.

**Note:** The names of login form fields are usually obtained by investigating the HTML source for the login page that protects the backend resource.

### 62.8.7.3 Creating a Pagelet

This step associates a pagelet with the Order Status page in the EBS Order Information module. To create a new pagelet, select the Pagelets section under the EBS resource you created and click the Create icon. Create the new pagelet with the following settings:

- **Name:** order_status (or your choice)
- **Library:** ebs11 (or your choice)
- **URL Suffix:** OA_HTML/RF.jsp?function_id=1005664&resp_id=22480&resp_appl_id=660&security_group_id=0&lang_code=US
- **Refresh Inline:** off

You can test the configured pagelet by accessing it via the REST link on the Documentation page. You should be taken to the Sales Order page without logging in (unless the pagelet uses the User Vault to store credentials and you are accessing it for the first time). From this page, you can use Simple Search to locate orders, using '%' as a wildcard character.

### 62.8.7.4 Making Corrective Configurations

After a pagelet has been created, it often requires additional features (Parser, Injector and/or Clipper) to either address issues with URL re-writing in the proxied markup, or simply to modify the markup for style or to hide unnecessary elements. This section describes corrective configurations that were applied to the sample pagelet that exposes the Order Information UI in EBS.

This section includes the following subsections:

- **Section 62.8.7.4.1, "Creating a Pluggable Parser for Popup URL Rewriting"**
- **Section 62.8.7.4.2, "Creating an Injector to Disable Frame-busting"**
- **Section 62.8.7.4.3, "Creating an Injector to Auto-Resize the iFrame"**
- **Section 62.8.7.4.4, "Setting Up Clipping by JavaScript Injection"**
- **Section 62.8.7.4.5, "Creating an Injector for Pagelet Restyling"**
- **Section 62.8.7.4.6, "Testing the Pagelets"**
- **Section 62.8.7.4.7, "Final Result"**
- **Section 62.8.7.4.8, "Troubleshooting"**

#### 62.8.7.4.1 Creating a Pluggable Parser for Popup URL Rewriting

Parsers allow you to supplement or change the built-in logic for parsing content and finding URLs that need to be rewritten. When built-in parsers fail to identify URLs, custom parsers can be used to correct this failure. In the case of the EBS 11i UI, this step is required to fix popup handling in Advanced Search.

Under the EBS Resource, create a new Parser with the following settings:
• Name: popupRewriter (or your choice)
• URL Filter: *.jsp*
• Use Base Parser: on
• Fragment Locations
  – var _jspDir='(.*?)'; (type Static URL)
  – <frame .? src="(.?)" (type Static URL)

Figure 62–60 shows the settings for the popupRewriter.

62.8.7.4.2 Creating an Injector to Disable Frame-busting  Injectors insert content into a specified location in the proxied resource page. The content can be any text, including HTML, CSS, JavaScript, and pagelet declarations. An empty injector may also be used to remove unwanted content from the page. The following injector was created for the EBS pagelet to prevent it from taking over the page, which was happening on the first displayed page where the user is asked to choose responsibility.

Under the EBS resource create a new Injector with the following settings:

• General
  – Name: frameBustDisabler
  – URL Filter: <none>
  – MIME Filter: text/html
  – Inject Location: Replace target=_top
  – Content: (Text) alt="

Figure 62–61 shows the parameter settings for the new Injector.
62.8.7.4.3 Creating an Injector to Auto-Resize the iFrame

In order to provide a more seamless integration into the consuming page, the pagelet iFrame can be configured to behave more like inline content by dynamically resizing to accommodate its contents. For details, see Enabling IFrame Auto-Resizing in this document.

62.8.7.4.4 Setting Up Clipping by JavaScript Injection

One option for suppressing page elements exposed as a pagelet is to create a clipper using graphical or Regular Expression-based clipping, available in Pagelet Producer. However, the recommended approach is to use custom JavaScript that is injected into the proxied page. This approach is often more flexible as it can easily accommodate the need to dynamically adjust rules that identify the element(s) to suppress or accommodate rules for suppressing multiple elements on the page.

Custom JavaScript can be added to the proxied resource page using an injector. To suppress EBS 11i standard header and navigation elements, use the following injector definition to insert a snippet of JavaScript that hides the unnecessary elements at page load time. (The standard Clipper feature in Pagelet Producer is not used due to the complexity of the EBS Web UI.)

Under the EBS Resource, create a new Injector with the following settings:

- **General:**
  - Name: nav_suppressor (or your choice)
  - URL Filter: <none>
  - MIME Filter: text/html
  - Inject Location: Before </body>

- **Content:** (Text - copy and paste the content below)

```html
<script type="text/javascript">
    // this script is injected into the page by ensemble
    // it hides header and footer tables at page load

    // register handler function to the page load event
    if (window.addEventListener) {
        window.addEventListener('load', hideHeaderFooter, false);
    } else if (document.attachEvent) {
        window.attachEvent('onload', hideHeaderFooter);
    }

    // this function does the actual hiding
    function hideHeaderFooter() {
```

Figure 62–61 frameBustDisabler Injector
var form = document.forms[0];
if (typeof(form) == 'undefined') return;
// main span is form's second child span
var spansFound = 0;
var mainSpan = null;
for (var i = 0; i < form.childNodes.length; i++) {
    var child = form.childNodes[i];
    if (child.tagName && child.tagName.toLowerCase() == 'span') {
        if (++spansFound == 2) {
            mainSpan = child;
            break;
        }
    }
}
if (typeof(mainSpan) != "undefined" && mainSpan != null) {
    for (var i = 0; i < mainSpan.childNodes.length; i++) {
        var child = mainSpan.childNodes[i];
        if (child.tagName && child.tagName.toLowerCase() == 'table') {
            child.style.display = 'none';
        }
    }
}
// call this function directly
hideHeaderFooter();
</script>

62.8.7.4.5 Creating an Injector for Pagelet Restyling To make the EBS pagelet fit better visually in the consuming page, an injector can be used to add styles to override the original CSS.

The example injector definition below shows how to implement restyling for the AviSports sample Web site in WebCenter Sites.

---

**Note:** In order for the override to work correctly, it is important that the style definitions supplied by the injector come after the styles defined by the backend application. In the example below, the content is injected into the end of the `<head>` section on the page.

---

Under the EBS Resource, create a new Injector with the following settings:

- **General:**
  - Name: avisports_styles
  - URL Filter: <none>
  - MIME Filter: text/html
  - Inject Location: Before `<head>`

- **Content:** (Text - copy and paste the content below)

```html
<style>
* {
    font: 11px Arial, Helvetica, sans-serif;
}
body {
    background: url("http://sites-host:port/cs/avisports/images/BlueSliver.png") repeat-x scroll center bottom transparent;
}
```
62.8.7.4.6 Testing the Pagelets The EBS11 resource now includes the pagelet, the custom injectors, and the custom parser as shown in Figure 62–62.

Figure 62–62 EBS11 Project Resources

Test the pagelet by browsing to the Documentation page and using the URL shown below, substituting the local host ID for "example.com."
62.8.7.4.7 Final Result  The following screenshots show the EBS 11i Order Information screens embedded into a page of the AviSports sample site in WebCenter Sites using a REST URL for pagelet access in an iFrame:
62.8.7.4.8 Troubleshooting  If some images are not rendered properly, make sure DHTML rewriting is enabled. (The DHTML Rewriting setting is on the General page for the resource.)

62.8.8 Consuming OpenSocial Gadgets Using Pagelet Producer

Pagelet Producer is an OpenSocial container that exposes the following features:

- People (exposes People Connections)
- Activities (exposes Activity Stream)
- Appdata (supports gadget personalization)
- Pub-Sub mechanism

Note that the People and Activities features require WebCenter Portal. To use the People or Activities features, target the WebCenterDS at the WC_Portlet managed server (i.e., at Pagelet Producer). For Appdata, Pagelet Producer supports user, instance and pagelet-scoped preferences.

To follow the example:

1. Open Pagelet Producer and create a new OpenSocial resource.
2. Define the URLs and policies as required.
3. Create a new pagelet specifying the URL for an external gadget.

For example:

http://bayareacoder.com/gogo/localweather/localweather.xml
4. If the gadget supports Preferences, configure the required parameters (see Figure 62–63 and Figure 62–64).

![Preferences Editor](image)

Figure 62–63  Preferences Editor

![Pagelet Parameters Dialog](image)

Figure 62–64  Pagelet Parameters Dialog

5. Create or upload the gadget XML file (see Figure 62–65).

![Pagelet Producer with Activities JavaScript and XML Options](image)

Figure 62–65  Pagelet Producer with Activities JavaScript and XML Options

Note that the path supplied in the Name field is a ‘virtual’ URL that Pagelet Producer will use to serve the file and any path or name can be used.

6. Create or upload any other files that the new gadget requires (for example, any JavaScript files, images, and so forth), and you’re done.

### 62.8.9 Manipulating HTML Markup

This section discusses how you can inject JavaScript to modify HTML markup, and use an injector with SSL.

This section includes the following subsections:

- Section 62.8.9.1, "Using an Injector with JavaScript"
- Section 62.8.9.2, "Using an Injector with SSL"

#### 62.8.9.1 Using an Injector with JavaScript

Injecting and/or replacing just HTML markup has limited potential. A more useful approach to consider is injecting JavaScript that directly modifies HTML markup. To do so:
Register a handler function on the page load event as shown in the following code snippet:

```javascript
if (window.addEventListener) {
    window.addEventListener('load', hideHeaderFooter, false);
} else if (document.attachEvent) {
    window.attachEvent('onload', hideHeaderFooter);
}
```

Use the handler to do the markup manipulation as shown in the rest of the code sample:

```javascript
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>
```

62.8.9.2 Using an Injector with SSL
Consuming an SSL resource requires a Pagelet Producer connection over HTTPS, including:

- Secure port mapping between WLS and Pagelet Producer
- Container should also be accessed via HTTPS

The injector can modify HTML markup that Pagelet Producer proxies:

- Based on URL pattern (resource scope)
- Based on MIME type
- Based on location (top or bottom) or exact text match (before, after, replace)
- Content page defines what gets injected

62.8.10 Advanced URL Rewriting
This section describes how to use Pagelet Producer's auto-login feature, and how to design a custom parser.

This section contains the following sub-sections:
62.8.10.1 Using Auto-Login

Auto-login lets you pass username/password credentials to a back-end resource. These can be passed as either:

- Static credentials – one set for all users
- Dynamically acquired from user at first resource access and then stored in secure Credential Vault
- User Profile (capture and pass username)

The supported authentication mechanisms for auto-login are:

- Basic login
- Form login (used in the following example)
- NTLM login
- Kerberos login

The example in Figure 62–66 and the code snippet below shows how you can use auto-login to redirect to a login page:
The following code reflects the settings in Figure 62–66:

```html
<form action="wwv_flow.accept" method="post" name="wwv_flow" id="wwvFlowForm"
autocomplete="off">
  <input type="hidden" name="p_flow_id" value="53557" id="pFlowId" />
  <input type="hidden" name="p_flow_step_id" value="101" id="pFlowStepId" />
  <input type="hidden" name="p_instance" value="724836084846701" id="pInstance" />
  <input type="hidden" name="p_page_submission_id" value="1431956852758801" id="pPageSubmissionId" />
  <input type="hidden" name="p_request" value="" id="pRequest" />
  <div id="login">
    <div id="messages"></div>
    <div id="login-main">
      <table id="myapp_layout_45149677586857064102" border="0" class="formlayout" summary="" role="presentation" datatable=0 >
        <tr>
          <td align="right"><label for="P101_USERNAME" tabindex="999"><a class="optional-w-help" href="javascript:popupFieldHelp('45149677686365064111','724836084846701')" tabindex="999">User Name</a></label></td>
          <td colspan="1" rowspan="1" align="left"><input type="hidden" name="p_arg_names" value="45149677686365064111" /><input type="text" id="P101_USERNAME" name="p_t01" value="" size="20" maxlength="100" class="text_field" /></td>
        </tr>
        <tr>
          <td align="right"><label for="P101_PASSWORD" tabindex="999">Password</label></td>
          <td colspan="1" rowspan="1" align="left"><input type="hidden" name="p_arg_names" value="45149677686365064111" /><input type="text" id="P101_PASSWORD" name="p_t01" value="" size="20" maxlength="100" class="text_field" /></td>
        </tr>
      </table>
    </div>
  </div>
</form>
```
62.8.10.2 Using a Custom Parser

Parsers let you extend or change built-in logic for URL parsing. Using a URL Filter and regular expressions, you can narrow down the part of the page that the parser should modify (Figure 62–67).

**Figure 62–67  Pagelet Producer - Custom Parser Example**

![Pagelet Producer - Custom Parser Example](image)

Using regular expressions lets you specify the HTML fragment to parse. The first grouping expression, denoted by parentheses, identifies the fragment itself, while the rest of the expression provides the context for finding it. A pluggable parser framework lets you fine-tune how the proxy recognizes different types of content based on MIME type, HTTP headers or URL.

In this example, there are three URLs in two places within the page markup that need to be modified to make Flash work:

```html
```

The following three regular expressions (shown under Regular Expressions in Figure 62–67) are used to achieve this:

- `XMLFile=(.*?)"`
- `param name="movie" value="(.*?)\.swf"`
- `embed src="(.*?)\.swf"`

62.9 Troubleshooting Pagelets

Keep the following common configuration errors in mind when troubleshooting resources and pagelets. For details on configuring Pagelet Producer settings, see Section 62.2, "Configuring Pagelet Producer Settings."
Troubleshooting Pagelets

- If you are using SSL, make sure both HTTP (nonsecure) and HTTPS (secure) ports are configured properly on the Transform page of the Pagelet Producer settings.

- If you are proxying external sites on a network that requires an HTTP proxy, you must configure the proxy URL on the Proxy page of the Pagelet Producer settings.

- Confirm that the login_resource and pagelet_api resources, created by default, are present and correctly configured.

- If you are proxying KD Browser or other CSP pagelets, make sure the image service URL is absolute and the CSP SOAP API URL is set correctly in the CSP page of the Pagelet Producer settings.

- Due to restrictions in the Sun Java Virtual Machine (JVM), pagelets cannot use HTTPS content where the underlying certificate uses MD2 signing algorithm.

For additional troubleshooting information, use logging. Pagelet Producer logs messages to the standard Oracle Diagnostic Logging facility. In Oracle WebLogic Server, that location is: user-projects/domains/<domain>/servers/<server>/logs/<server>-diagnostic.log. For details on configuring logging, see Section 62.2.2, "Configuring Logging Settings." For more information on debugging pagelets, see the "Creating Pagelets with Oracle WebCenter Portal's Pagelet Producer" chapter in Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper.
This chapter describes how to add portlets to the pages of your WebCenter Portal Framework applications and the options that accompany this process.

This chapter includes the following topics:

- Section 63.1, "Introduction to Consuming Portlets"
- Section 63.2, "Registering Portlet Producers with a WebCenter Portal Framework Application"
- Section 63.3, "Managing Portlet Producer Connections"
- Section 63.4, "Adding the Portlet Producer Task Flow"
- Section 63.5, "Adding Portlets to a Page"
- Section 63.6, "Setting Attribute Values for the Portlet Tag"
- Section 63.7, "Manually Wiring Parameters and Events in JSR 286 Portlets"
- Section 63.8, "Copying Portlets"
- Section 63.9, "Deleting Portlets from Application Pages"
- Section 63.10, "Consuming WebCenter Services Portlets"
- Section 63.11, "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 Portal Framework application pages. Therefore, you should familiarize yourself with the information covered in the 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 Portal Framework application pages, see the following chapters:

- Chapter 57, "Introduction to Portlets"
- Chapter 58, "Creating Portlets from JSF Applications Using the Oracle JSF Portlet Bridge"
- Chapter 59, "Building Standards-Based Java Portlets Using JSR 286"

### 63.1 Introduction to Consuming Portlets

Oracle WebCenter Portal Framework enables you to consume a portlet by registering its producer with the application. Your application can consume portlets that you build using WebCenter Portal or other Oracle products, and portlets that you receive from a third party, such as a packaged-application vendor. Table 63–1 lists some of the
products supported as portlet producers within WebCenter Portal.

### Table 63–1  Supported Portlet Producers

<table>
<thead>
<tr>
<th>Portlet Producer</th>
<th>Supported?</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle WebLogic Portal</td>
<td>Yes</td>
<td>For more information, see the &quot;Exporting Java Portlets for Use on Other Systems&quot; section in the Oracle Fusion Middleware Portlet Development Guide for Oracle WebLogic Portal.</td>
</tr>
<tr>
<td>Oracle WebCenter Interaction</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>Oracle Portal</td>
<td>Yes</td>
<td>For more information, see Section F.2, &quot;Reusing Portlets.&quot;</td>
</tr>
<tr>
<td>E-Business Suite application</td>
<td>Yes</td>
<td>For more information, see Section 54.3, &quot;Integrating E-Business Suite Applications.&quot;</td>
</tr>
<tr>
<td>PeopleSoft application</td>
<td>Yes</td>
<td>For more information, see Section 54.5, &quot;Integrating PeopleSoft Applications.&quot;</td>
</tr>
<tr>
<td>JD Edwards application</td>
<td>Yes</td>
<td>For more information, see Section 54.4, &quot;Integrating JD Edwards Applications.&quot;</td>
</tr>
</tbody>
</table>

**Note:** WSRP producers built by a third party and consumed by WebCenter Portal should function correctly provided:

- The producer does not rely on any vendor-specific extensions to WSRP.
- The portlets do not make assumptions about the application in which they are consumed, for example by expecting a particular JavaScript method to exist in the page.

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 Portal 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.
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.

63.2 Registering Portlet Producers with a WebCenter Portal Framework Application

This section includes the following topics:

- Section 63.2.1, "Registering a WSRP Portlet Producer with a WebCenter Portal Framework Application"
- Section 63.2.2, "Registering an Oracle PDK-Java Portlet Producer with a WebCenter Portal Framework Application"

63.2.1 Registering a WSRP Portlet Producer with a WebCenter Portal Framework Application

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.

This section includes the following topics:

- Section 63.2.1.1, "How to Register a WSRP Portlet Producer"
- Section 63.2.1.2, "How to Map a Producer's Declared User Categories to an Application's Defined Java EE Security Roles"

63.2.1.1 How to Register a WSRP Portlet Producer

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 58, "Creating Portlets from JSF Applications Using the Oracle JSF Portlet Bridge."

To register a WSRP portlet producer:

1. In the Application Resources panel of the Application Navigator, right-click Connections, choose New Connection and then choose WSRP Producer.

2. In the Specify Producer Name page 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.

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, 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?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:
Registering Portlet Producers with a WebCenter Portal Framework Application

http://host:port/context-root/info

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, 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.

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 63.2.1.2, "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, 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.

11. Select the configuration type:
Registering Portlet Producers with a WebCenter Portal Framework Application

- **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 taken from the policy being used. The WebLogic Server where the application is deployed must be configured for WS-Security. For more information, see the "Securing a WSRP Producer with WS-Security" section in Administering 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 Portal 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 Portal 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 74.13.3, "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 63.2.1.2, "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.
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 63.2.1.2, "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.

63.2.1.2 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 Portal Framework application's `web.xml` file properties.
For more information about security roles in JSR 286 portlets, see Section 74.17, "Securing Identity Propagation Through WSRP Producers with WS-Security."

This procedure continues forward from Section 63.2.1.1, "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, 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 63.3.1, "How to Edit Portlet Producer Registration Settings."

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.

**63.2.2 Registering an Oracle PDK-Java Portlet Producer with a WebCenter Portal Framework Application**

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.

2. In the Specify Producer Name page 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.

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, 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

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 74.13.3, "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, in the Default Timeout Interval (Seconds) field, enter the number of seconds to wait for the producer to respond during design time operations.

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 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.

63.3 Managing Portlet Producer Connections

After registering a portlet producer, you can edit the registration settings, test the connection, refresh the producer to obtain the latest list of portlets, or delete the connection to the portlet producer.

This section includes the following topics:

- Section 63.3.1, "How to Edit Portlet Producer Registration Settings"
- Section 63.3.2, "How to Test a Portlet Producer Connection"
- Section 63.3.3, "How to Refresh a Portlet Producer"
- Section 63.3.4, "How to Delete a Portlet Producer"
- Section 63.3.5, "How to Edit Portlet Client Configuration"

63.3.1 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 63.2.1, "Registering a WSRP Portlet Producer with a WebCenter Portal Framework Application."

- For information about Oracle PDK-Java Producer settings, see Section 63.2.2, "Registering an Oracle PDK-Java Portlet Producer with a WebCenter Portal Framework Application."

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.

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 63.3.2, "How to Test a Portlet Producer Connection."

### 63.3.2 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.

### 63.3.3 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.
Managing Portlet Producer Connections

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 you want to refresh, and choose Refresh Producer Registration.
3. In the Portlet Producer Refresh dialog, click Yes.

63.3.4 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 63.9, "Deleting Portlets from Application Pages."

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 you want to delete, and choose Delete.
3. In the Delete Confirmation dialog, click Yes.

63.3.5 How to Edit Portlet Client Configuration

The adf-config.xml file contains configuration information for WebCenter Portal tools and services. Portlet client configuration details, such as timeout and cache settings, are specified in the adf-portlet-config section of the file.

You can edit the adf-config.xml file for your application and edit the portlet client configuration.

| Table 63–2 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 63–2  Child Elements of adf-portlet-config

<table>
<thead>
<tr>
<th>Element (init-param)</th>
<th>Description</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>parallelPoolSize</code> (parallel.pool.size)</td>
<td>The number of threads to use for parallel execution of tasks.</td>
<td>10</td>
</tr>
<tr>
<td><code>parallelQueueSize</code> (parallel.queue.size)</td>
<td>The size of the queue of tasks waiting for parallel execution. Tasks are rejected when the queue size is exceeded.</td>
<td>20</td>
</tr>
<tr>
<td><code>defaultTimeout</code> (default.timeout)</td>
<td>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.</td>
<td>10</td>
</tr>
<tr>
<td><code>minimumTimeout</code> (minimum.timeout)</td>
<td>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.</td>
<td>0.1</td>
</tr>
<tr>
<td><code>maximumTimeout</code> (maximum.timeout)</td>
<td>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.</td>
<td>300</td>
</tr>
<tr>
<td><code>resourceProxyPath</code> (resource.proxy.path)</td>
<td>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.</td>
<td>/resourceproxy</td>
</tr>
<tr>
<td><code>supportedLocales</code> (supportedlocales)</td>
<td>The set of supported locales defined using strings of the form: language[_country[_variant]].</td>
<td>Commented out by default. You should uncomment this element if you have multiple locales. See Example 63–1.</td>
</tr>
<tr>
<td><code>portletTechnologies</code> (portlet.technologies)</td>
<td>The set of portlet technologies supported by the client defined by the fully qualified names of classes that implement the PortletTechnologyConfig interface.</td>
<td><code>{o.p.c.ci.web.WebPortletTechnologyConfig, o.p.c.ci.wsrp.WSRPPortletTechnologyConfig}</code></td>
</tr>
<tr>
<td><code>cacheSettings</code> (cache.*)</td>
<td>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.</td>
<td>The cache is enabled and no size restrictions are imposed.</td>
</tr>
</tbody>
</table>
**Example 63–1** illustrates the usage of the `adf-portlet-config` section in `adf-config.xml`.

### Example 63–1  The adf-portlet-config Section of adf-config.xml

```xml
<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>
```

### Table 63–2  (Cont.) Child Elements of adf-portlet-config

<table>
<thead>
<tr>
<th>Element (init-param)</th>
<th>Description</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>maximumResourceUrlLength</td>
<td>The maximum length allowed for resource URLs. This prevents the URL from becoming too 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.</td>
<td>2048</td>
</tr>
</tbody>
</table>

The maximum length allowed for resource URLs. This prevents the URL from becoming too 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.
Tip: To edit portlet client configuration at runtime, without having to redeploy the application, you can use the setPortletClientConfig WLST command. For more information, see the “setPortletClientConfig” section in the WebLogic Scripting Tool Command Reference.

### 63.4 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 Portal Framework application template, the portlet producer task flow is available out of the box through the runtime administration console (Services tab). For details, see the "Configuring Services, Portlet Producers, and External Applications for Portal Framework Applications" chapter in Administering 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 Portal Framework application template and the runtime 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 the JSF page in your application on which you want to add the task flow.
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 "Creating Test Users" section in the 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 63–1 appears.
63.5 Adding Portlets to a Page

Placing a portlet on a Portal 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 topics:

- Section 63.5.1, "How to Add a Portlet to a Page"
- Section 63.5.2, "What Happens When You Add a Portlet to a Page"
- Section 63.5.3, "What You May Need to Know About Interportlet Communication"
- Section 63.5.4, "What You May Need to Know About Inline Frames"
- Section 63.5.5, "What You May Need to Know About Portlet Sizing"
- Section 63.5.6, "What You May Need to Know About Minimize, Restore, and Move"
- Section 63.5.7, "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 Portal Framework application. For more information, see Section 6.1, "Creating a New WebCenter Portal Framework Application."
2. Creating an application page. For more information, see Section 15.2, "Creating Pages in a WebCenter Portal Framework Application."
3. Registering the portlet’s producer with the application. For more information, see Section 63.2.1, "Registering a WSRP Portlet Producer with a WebCenter Portal Framework Application" or Section 63.2.2, "Registering an Oracle PDK-Java Portlet Producer 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
Adding Portlets to a Page

external application and identify it to the portlet producer that provides it. For more information, see Chapter 74, "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 74.14, "Registering Custom Certificates with the Keystore."

63.5.1 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.
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.
Adding Portlets to a Page

### 63.5.2 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 63.6, "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, the adfp:portlet tag is used.
- 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 15.2, "Creating Pages in a WebCenter Portal Framework Application," the page is a rich client page. In this case, the portlet is added using the adfp:portlet tag, as shown in Example 63–2.

---

**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 add a portlet as a Trinidad component, that is, inside a tr:form component, and the portlet implements modes other than View mode, you must include the following in the application’s web.xml file:

```xml
<context-param>
  <param-name>oracle.adf.view.rich.newWindowDetect.OPTIONS</param-name>
  <param-value>off</param-value>
</context-param>
```

**Note:** If you are adding a PeopleSoft portlet to a page in a Portal Framework application, you must set the renderPortletInIFrame portlet tag attribute to true.

For more information, see Section 63.6, "Setting Attribute Values for the Portlet Tag."
Adding Portlets to a Page

Example 63–2  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 adfp:portlet tag is used, as shown in Example 63–3.

Example 63–3  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"
<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 63–4:

Example 63–4  Hierarchical Placement of the Portlet Tag

af:document
  af:form
    cust:panelCustomizable
    adfp:portlet
Adding Portlets to a Page

### 63.5.3 What You May Need to Know About Interportlet Communication

One way to make your WebCenter Portal Framework application more interactive is by linking related components such that their contents are synchronized based upon the context. For example, suppose you have two stock portlets on a page, one provides data about a stock's price while the other provides headline news items for a stock. Both portlets are based upon the stock ticker symbol, hence it would make sense that, when the ticker symbol is changed in the stock price portlet, the stock headlines portlet picks up that change and refreshes itself with headlines pertaining to the same ticker symbol.

The JSR 286 standard introduces public render parameters, which you can use to link portlets on a page. For more information, see Section 59.3.5, "How to Use Public Render Parameters in JSR 286 Portlets."

The JSR 286 standard also supports portlet events. You can use portlet events to drive the content of a portlet based on actions that occur elsewhere on the 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. Portlet events can also be consumed by ADF contextual events and contextual events can be consumed by portlets. You can use this to contextually link portlets and task flows. For more information, see Section 59.3.6, "How to Use Portlet Events in JSR 286 Portlets."

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 63–5).

**Example 63–5  Portlet Binding**

```xml
<portlet id='LotteryPortlet2_1'
    portletInstance='oracle/adf/portlet/SamplePortlets/ap/817default_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'/>
```

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 18, "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 property, 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.
Setting these attributes to true (the default) means that any parameters that are changed or events that are raised elsewhere on the page that match those supported by the portlet (that is, they have the same QName) automatically cause the portlet to be updated accordingly. In addition, the portlet can define aliases for its parameters and events to match parameters and events with different QNames.

You can set these attributes to false to disable this automatic parameter and event listening. 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, or if your portlet defines parameters or events with names that do not match those you want to use for linking, you must manually configure the portlet wiring. For more information, see Section 63.7, “Manually Wiring Parameters and Events in JSR 286 Portlets.”

### 63.5.4 What You May Need to Know About Inline Frames

Rendering a portlet directly on a page provides a better user experience as compared to placing it in an inline frame (iframe). However, at times, it may be required to include the portlet markup inside an inline frame. You can use the renderPortletInIFrame attribute to determine whether or not a portlet should be rendered in an inline frame.

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 inline frame.

However, in the following circumstances, the portlet is rendered within an inline frame:

- 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 inline frame 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 inline frame.

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.
Adding Portlets to a Page

Consuming Portlets

If you want to ensure that a portlet is never rendered in an inline frame, 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 inline frame may interfere with other components on the Oracle ADF page.

For more information about the accessibility implications of using inline frames, see Section I.1.3, "Accessibility Considerations for Portlets."

### 63.5.5 What You May Need to Know About Portlet Sizing

ADF Faces components can stretch their children, or not. It is usually a requirement that a parent only contains a single child for it to be able to stretch that child. Components also support being stretched or not. When a component that supports being stretched is placed inside a parent that is stretching its child, the dimension of the child is determined by those of the parent. The child is said to be stretched by the parent and it is sized to fill the parent.

When a component is not stretched by its parent, it is said to be in a flow layout. Here it is not taking its size from its parent. It must determine its own size, either from its children or from dimension style settings specified by the page designer.

The `portlet` Faces component supports being stretched by its parent. It does not explicitly stretch its children.

There are three ways the size of a `portlet` component is determined, depending on the portlet’s parent component and attributes set on the `portlet` component itself.

- **If the portlet is being stretched by its parent, the size is determined only by the parent.** The portlet is in a stretch layout.
  - It does not matter how big the content is, this is irrelevant to the size of the portlet. If the content is bigger than the portlet, scrollbars are provided to enable users to display all the content.
  - If you specify a height for the portlet, it is ignored; stretching takes precedence.
  - This is the preferred layout method; it is always reliable.
  - The ADF Faces Tag documentation tells you whether each component stretches its children or not.

- **If the portlet is not being stretched by its parent and explicit sizes are set, they are used.**
  - This is also always reliable; you'll get the size you specified.
  - If the content is bigger than the specified size, scrollbars are provided to enable users to display all the content.
  - The size is fixed; the size of the content is not taken into account.

- **If the portlet is not being stretched by its parent and no explicit sizes are set, the portlet attempts to auto-size to the content.**

---

**Note:** If you render a portlet within an inline frame, 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 inline frame.

---
Auto-Sizing

When a portlet is in a flow layout and has no explicit size set, it attempts to set the size of the portlet so that it is exactly big enough to display all of the portlet content without scrollbars.

The portlet does this by asking the browser how much space is required to show the content without scrollbars and using that to set the height of the portlet. More specifically, it looks at the scrollHeight property of the inline frame window.

How well this works depends on what is inside the portlet. Some layouts have a nice intrinsic size which auto-sizes well. However, some layouts and components inside the portlet, typically those that allow their content to overflow invisibly or with scrollbars, effectively hide the size of their contents. In specific terms of the HTML markup are HTML elements with overflow setting of hidden, auto, or scroll. This can come from inline styles or CSS styles applied to the element.

The problem arises because the scrollHeight is the minimum height required to correctly display the content. If you have, for example, a div element with an overflow="auto", then if that div is smaller than its content, then it uses scroll bars. So, when you ask what minimum size is needed to display the div, unless it has an explicit height or minimum height set, the answer is 0. That means that the content of this div, even if it has a definite height does not contribute to the overall scroll height of the content.

In ADF terms, this is often associated with components that expect to take their dimensions from their parents, for example panelStretchLayout with dimensionFrom="parent" or that scroll their contents, for example, panelGroupLayout with layout="scroll". These are the components that tend to have hidden or scrollable overflows.

Thinking about the ADF Faces layout model also gives us an alternative way of understanding the problem. The ADF Faces layout guidelines dictate starting with an unbroken chain of the components that are stretched by their parents and within that islands of flow layout, usually initiated by an af:panelGroupLayout where the components have fixed heights or take their dimensions from their children. If in the root of the portlet, we have that unbroken chain of components looking to take their dimensions from their parent but the portlet itself is in a flow layout and therefore is looking to take its dimensions from the content there is an impasse. The portlet is looking to take its dimensions from its parent for the size. It should be clear then that the solution to this is to stretch the portlet (that is, take it out of the flow layout island), set a height on the portlet, or make the content of the portlet entirely and island of flow layout content.

If the content cannot be auto-sized, the height defaults to 200px in Firefox and 0px in Internet Explorer. The fact that Internet Explorer is 0 is caused by incorrect behavior where the scrollHeight does not correctly respect the min-height CSS style setting. In Firefox, the 200px height comes from the minheight setting on the region that renders the portletized task flow. To work around this, you can set the height or min-height CSS in the consumer.

Auto-Sizing Best Practice

To get the best results when using the portlet in a flow layout where you want it auto-size the content:

- Make the portlet content an unbroken chain of components that take their dimensions from their children.
Adding Portlets to a Page

- Use the `dimensionsFrom="children"` attribute on components that support it, for example, `af:panelStretchLayout`, to make them take their dimensions from their children.

- Use `layout="vertical"` rather than `layout="scroll"` on `panelGroupLayout` to make the size of its children contribute to the overall auto-sized dimensions.

- When switching from the flow layout chain of components to a stretch layout, set an explicit height on the first component that stretches its children. Typically, this will be where you have used `panelStretchLayout` with `dimensionsFrom="parent"`.

### 63.5.6 What You May Need to Know About Minimize, Restore, and Move

To accommodate the needs of the development environment, the behavior of the actions Minimize, 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 63–6):

#### Example 63–6 Persistence Setting in the Application’s web.xml File

```xml
<context-param>
  <param-name>oracle.adf.view.faces.CHANGE_PERSISTENCE</param-name>
  <param-value>oracle.adfinternal.view.faces.change.HybridChangeManager</param-value>
</context-param>
```

Replace it with the following (Example 63–7):

#### Example 63–7 Turning Runtime Persistence Off in the Application’s web.xml File

```xml
<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, 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. Minimize 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.
63.5.7 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 64, "Creating Portlets with OmniPortlet." For more information about the Web Clipping portlet, see Chapter 65, "Creating Content-Based Portlets with Web Clipping." For more information about portlets generally, see Chapter 57, "Introduction to Portlets."

When running a portlet that has an Edit mode (in a Portal 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 74.12, "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:

```xml
<factory nameSpace="http://xmlns.oracle.com/portlet/bindings"
  className="oracle.adf.model.portlet.binding.PortletBindingDefFactoryImpl"/>
<dtfactory
  className="oracle.adfdtinternal.view.faces.portlet.PortletDefinitionDTFactory"/>
```

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 74.13, "Working with External Applications."

63.6 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 20.10, "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 topics:

- Section 63.6.1, "How to Set Attribute Values for the Portlet Tag Using the Property Inspector"
- Section 63.6.2, "How to Set Attribute Values for the Portlet Tag in Source Code"
- Section 63.6.3, "Common Attributes of the Portlet Tag"
- Section 63.6.4, "Appearance Attributes of the Portlet Tag"
- Section 63.6.5, "Behavior Attributes of the Portlet Tag"
- Section 63.6.6, "Portlet Modes Attributes of the Portlet Tag"
- Section 63.6.7, "Style Attributes of the Portlet Tag"
- Section 63.6.8, "Binding Attributes of the Portlet Tag"
- Section 63.6.9, "Customization Attributes of the Portlet Tag"
- Section 63.6.10, "Other Attributes of the Portlet Tag"

### 63.6.1 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.

### 63.6.2 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 63–8** shows an edited portlet tag.

   **Example 63–8**  **An Edited Portlet Tag**

   ```xml
   <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**.

63.6.3 **Common Attributes of the Portlet Tag**

   **Table 63–3** describes the common attributes of the portlet tag.
### Table 63–3  Common Attributes of the Portlet Tag

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Text string. For example:</td>
<td>The unique identifier of the portlet. This attribute is populated with a</td>
</tr>
<tr>
<td></td>
<td>id=&quot;newsBrief&quot;</td>
<td>unique value by default when you add the portlet to a page.</td>
</tr>
<tr>
<td></td>
<td>The value must follow a</td>
<td></td>
</tr>
<tr>
<td></td>
<td>subset of the syntax</td>
<td></td>
</tr>
<tr>
<td></td>
<td>allowed in HTML:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ Must not be a zero-length String.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ First character must be a</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ASCII letter (A-Z a-z) or</td>
<td></td>
</tr>
<tr>
<td></td>
<td>an underscore (_).</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ Subsequent characters</td>
<td></td>
</tr>
<tr>
<td></td>
<td>must be ASCII letter or</td>
<td></td>
</tr>
<tr>
<td></td>
<td>digits (a-Z a-z 0-9),</td>
<td></td>
</tr>
<tr>
<td></td>
<td>underscores (_), or dashes</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(-).</td>
<td></td>
</tr>
<tr>
<td>title</td>
<td>Text string. For example:</td>
<td>The portlet title, which is displayed in the portlet header.</td>
</tr>
<tr>
<td></td>
<td>title=&quot;Announcements&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The value specified here</td>
<td>The value specified here takes precedence over any title specified elsewhere</td>
</tr>
<tr>
<td></td>
<td>takes precedence over any</td>
<td>(for example, in the portlet markup).</td>
</tr>
<tr>
<td></td>
<td>title specified elsewhere</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(for example, in the</td>
<td></td>
</tr>
<tr>
<td></td>
<td>portlet markup)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>If no value is specified</td>
<td>If no value is specified here, the portlet extracts its title from the</td>
</tr>
<tr>
<td></td>
<td>here, the portlet extracts</td>
<td>portlet markup (response).</td>
</tr>
<tr>
<td></td>
<td>its title from the portlet</td>
<td></td>
</tr>
<tr>
<td></td>
<td>markup (response).</td>
<td></td>
</tr>
<tr>
<td></td>
<td>If no value is specified</td>
<td>If no value is specified either here or in the portlet markup, the portlet</td>
</tr>
<tr>
<td></td>
<td>here or in the portlet</td>
<td>extracts its title from the portlet definition.</td>
</tr>
<tr>
<td></td>
<td>markup, the portlet extracts</td>
<td></td>
</tr>
<tr>
<td></td>
<td>its title from the portlet</td>
<td></td>
</tr>
<tr>
<td></td>
<td>definition.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Note: Supplying a value</td>
<td>Note: Supplying a value to the title attribute at design time means that</td>
</tr>
<tr>
<td></td>
<td>to the title attribute</td>
<td>any change made to the title at runtime in Edit or Edit Defaults mode is</td>
</tr>
<tr>
<td></td>
<td>at design time means that</td>
<td>ignored.</td>
</tr>
<tr>
<td></td>
<td>any change made to the title</td>
<td></td>
</tr>
<tr>
<td></td>
<td>at runtime in Edit or Edit</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Defaults mode is ignored.</td>
<td></td>
</tr>
<tr>
<td>width</td>
<td>Number expressed in pixels</td>
<td>The width of the area to allow for portlet display.</td>
</tr>
<tr>
<td></td>
<td>or as a percentage of</td>
<td></td>
</tr>
<tr>
<td></td>
<td>available area:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ For pixels, enter npx,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>for example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>width = 300px</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ For percentage, enter n%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>for example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>width = 50%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The actual portlet width</td>
<td>If the actual portlet width is larger than the width value entered here, a</td>
</tr>
<tr>
<td></td>
<td>is larger than the width</td>
<td>scrollbar appears, provided displayScrollBar is set to auto or true. If</td>
</tr>
<tr>
<td></td>
<td>value entered here, a</td>
<td>displayScrollBar is set to false, and the actual portlet width exceeds the</td>
</tr>
<tr>
<td></td>
<td>scrollbar appears, provided</td>
<td>value expressed for the width attribute, the width attribute value is</td>
</tr>
<tr>
<td></td>
<td>displayScrollBar is set to</td>
<td>considered and the portlet content is truncated.</td>
</tr>
<tr>
<td></td>
<td>false, and the actual portlet</td>
<td></td>
</tr>
<tr>
<td></td>
<td>width exceeds the value</td>
<td></td>
</tr>
<tr>
<td></td>
<td>expressed for the width</td>
<td></td>
</tr>
<tr>
<td></td>
<td>attribute, the width</td>
<td></td>
</tr>
<tr>
<td></td>
<td>attribute value is</td>
<td></td>
</tr>
<tr>
<td></td>
<td>considered and the portlet</td>
<td></td>
</tr>
<tr>
<td></td>
<td>content is truncated.</td>
<td></td>
</tr>
</tbody>
</table>
Table 63–3 (Cont.) Common Attributes of the Portlet Tag

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>height</td>
<td>Number expressed in pixels, for example: height = 300px</td>
<td>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.</td>
</tr>
<tr>
<td>icon</td>
<td>URI to an image. For example: icon=&quot;coffee.png&quot;</td>
<td>A URI specifying the location of an image to use as an icon, displayed to the left of the portlet tile 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.</td>
</tr>
<tr>
<td>partialTriggers</td>
<td>One or more component IDs. For example: partialTriggers=&quot;_id1 _id2 componentID5&quot;</td>
<td>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.</td>
</tr>
</tbody>
</table>

63.6.4 Appearance Attributes of the Portlet Tag

Table 63–4 describes the appearance attributes of the portlet tag.
### Table 63–4  Appearance Attributes of the Portlet Tag

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>expansionMode</td>
<td>minimized</td>
<td>The default state of the portlet:</td>
</tr>
<tr>
<td></td>
<td>normal</td>
<td>■ minimized—The portlet's default display mode is collapsed (minimized).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ normal—The portlet's default display mode is neither collapsed nor expanded</td>
</tr>
<tr>
<td></td>
<td></td>
<td>to the width of the page.</td>
</tr>
<tr>
<td></td>
<td>Default:</td>
<td>normal</td>
</tr>
<tr>
<td></td>
<td>normal</td>
<td></td>
</tr>
<tr>
<td>allModesSharedScreen</td>
<td>auto</td>
<td>Whether a change in portlet mode renders the new mode on a new page, rather</td>
</tr>
<tr>
<td></td>
<td>false</td>
<td>than the page on which the portlet resides.</td>
</tr>
<tr>
<td></td>
<td>true</td>
<td>■ auto—All portlet modes are displayed inline if the remote portlet is</td>
</tr>
<tr>
<td></td>
<td></td>
<td>configured, through the com.oracle.portlet.requireIFrame</td>
</tr>
<tr>
<td></td>
<td></td>
<td>container runtime option in its portlet.xml file, to require an inline frame</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(IFRAME). This ensures that Oracle Bridge portlets are displayed inline.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ false—All portlet modes, except View (JSR 286) or Show (PDK-Java), are</td>
</tr>
<tr>
<td></td>
<td></td>
<td>rendered each on their own page. This is useful for portlets such as OmniPort</td>
</tr>
<tr>
<td></td>
<td></td>
<td>let and the Web Clipping portlet, which require that modes other than Show</td>
</tr>
<tr>
<td></td>
<td></td>
<td>mode display on pages other than the page on which the portlet resides.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ true—All portlet modes are displayed inline. One mode is swapped out for</td>
</tr>
<tr>
<td></td>
<td></td>
<td>another on the same page. In other words, this attribute enables all portlet</td>
</tr>
<tr>
<td></td>
<td></td>
<td>modes to display without leaving the context of a given page.</td>
</tr>
<tr>
<td></td>
<td>Default:</td>
<td>auto</td>
</tr>
<tr>
<td></td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>renderPortletInIFrame</td>
<td>auto</td>
<td>Whether the portlet is rendered in an IFRAME:</td>
</tr>
<tr>
<td></td>
<td>false</td>
<td>■ auto—Whenever possible, the portlet consumer attempts to rewrite the</td>
</tr>
<tr>
<td></td>
<td>true</td>
<td>portlet markup so that it renders correctly in an ADF page. Otherwise, the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>portlet is rendered in an IFRAME.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ false—The portlet is never rendered in an IFRAME.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ true—The portlet is always rendered in an IFRAME.</td>
</tr>
<tr>
<td></td>
<td>Default:</td>
<td>auto</td>
</tr>
<tr>
<td></td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>displayScrollBar</td>
<td>auto</td>
<td>Whether a scroll bar is displayed:</td>
</tr>
<tr>
<td></td>
<td>false</td>
<td>■ auto—Render a scroll bar if the portlet content does not fit the width and</td>
</tr>
<tr>
<td></td>
<td>true</td>
<td>height specified.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ false—Never render a scroll bar. If the portlet content does not fit the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>height and width specified, the portlet renders in its actual size.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ true—Always render a scroll bar.</td>
</tr>
<tr>
<td></td>
<td>Default:</td>
<td>auto</td>
</tr>
</tbody>
</table>
### Table 63–4 (Cont.) Appearance Attributes of the Portlet Tag

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>displayHeader</td>
<td>false</td>
<td>Whether the portlet header is displayed:</td>
</tr>
<tr>
<td></td>
<td>true</td>
<td>■ false—The portlet header is not displayed. Icons and links normally displayed in the header are hidden. If</td>
</tr>
<tr>
<td></td>
<td></td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td>Default: true</td>
<td>■ true—The portlet header is displayed. Consequently, header-based icons and links are displayed.</td>
</tr>
<tr>
<td>displayShadow</td>
<td>false</td>
<td>Whether to display a shadow decoration around the portlet:</td>
</tr>
<tr>
<td></td>
<td>true</td>
<td>■ false—Do not display a shadow decoration.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ true—Display a shadow decoration.</td>
</tr>
<tr>
<td>rendered</td>
<td>false</td>
<td>Whether the portlet is rendered.</td>
</tr>
<tr>
<td></td>
<td>true</td>
<td>■ false—Do not render the portlet. No output is rendered.</td>
</tr>
<tr>
<td>background</td>
<td>dark</td>
<td>■ true—Render the portlet. This is the recommended setting. Setting this attribute to false causes problems when you run the page.</td>
</tr>
<tr>
<td></td>
<td>light</td>
<td>The style selector to apply to the skin used by the portlet:</td>
</tr>
<tr>
<td></td>
<td>medium</td>
<td>■ dark—Apply the dark style selector to the skin.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ light—Apply the light style selector to the skin.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ medium—Apply the medium style selector to the skin.</td>
</tr>
<tr>
<td></td>
<td>Default: medium</td>
<td>This provides a way for you to apply a different look and feel to each portlet on a page.</td>
</tr>
<tr>
<td>shortDesc</td>
<td></td>
<td>A short description of the portlet.</td>
</tr>
<tr>
<td>displayActions</td>
<td>always</td>
<td>Whether seeded interactions for the portlet are shown:</td>
</tr>
<tr>
<td></td>
<td>onHover</td>
<td>■ always—Always show seeded interactions.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ onHover—Show seeded interactions when users move the mouse over the portlet.</td>
</tr>
</tbody>
</table>
Table 63–5 (Cont.) Appearance Attributes of the Portlet Tag

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>showMoveAction</td>
<td>menu</td>
<td>Whether to display the Move command in the portlet’s Action menu:</td>
</tr>
<tr>
<td></td>
<td>none</td>
<td>- menu—Display the Move command on the portlet’s Action menu.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- none—Do not display the Move command.</td>
</tr>
<tr>
<td></td>
<td>Default: menu</td>
<td>There is a difference in the way that the Move command behaves at design time and at runtime. For more information, see Section 63.5.6, “What You May Need to Know About Minimize, Restore, and Move.”</td>
</tr>
<tr>
<td>showRemoveAction</td>
<td>menu</td>
<td>Whether to display the Remove icon on the portlet chrome:</td>
</tr>
<tr>
<td></td>
<td>none</td>
<td>- menu—Display the Remove command on the portlet’s Action menu.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- none—Do not display the Remove icon.</td>
</tr>
<tr>
<td></td>
<td>Default: menu</td>
<td>There is a difference in the way that the Remove icon behaves at design time and at runtime. For more information, see Section 63.5.6, “What You May Need to Know About Minimize, Restore, and Move.”</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Note: This attribute is available only for the adfp:portlet tag and not for the adfph:portlet tag.</td>
</tr>
<tr>
<td>showResizer</td>
<td>always</td>
<td>Whether to display the resize handle at the bottom right corner of the portlet.</td>
</tr>
<tr>
<td></td>
<td>never</td>
<td>- always—Always display the resize handle.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- never—Never display the resize handle.</td>
</tr>
<tr>
<td></td>
<td>Default: always</td>
<td>Note: This attribute is available only for the adfp:portlet tag and not for the adfph:portlet tag.</td>
</tr>
<tr>
<td>showMinimizeAction</td>
<td>chrome</td>
<td>Whether to display the Minimize icon on the portlet chrome:</td>
</tr>
<tr>
<td></td>
<td>none</td>
<td>- chrome—Display the Minimize icon on the portlet chrome.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- none—Do not display the Minimize icon.</td>
</tr>
<tr>
<td></td>
<td>Default: chrome</td>
<td>There is a difference in the way that the Minimize icon behaves at design time and at runtime. For more information, see Section 63.5.6, “What You May Need to Know About Minimize, Restore, and Move.”</td>
</tr>
</tbody>
</table>

63.6.5 Behavior Attributes of the Portlet Tag

Table 63–5 describes the behavior attributes of the portlet tag.
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 63–6, 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 63–6  Portlet Modes Attributes of the Portlet Tag

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>isAboutModeAvailable</code></td>
<td>false</td>
<td>Whether to render an <strong>About</strong> command on the portlet’s <strong>Actions</strong> menu.</td>
</tr>
<tr>
<td></td>
<td>true</td>
<td>Users choose <strong>About</strong> to invoke the portlet’s About mode.</td>
</tr>
<tr>
<td></td>
<td>Default: true</td>
<td></td>
</tr>
<tr>
<td><code>isConfigModeAvailable</code></td>
<td>false</td>
<td>Whether to render a <strong>Configure</strong> command on a JSR 286 portlet’s <strong>Actions</strong> menu.</td>
</tr>
<tr>
<td></td>
<td>true</td>
<td>Users choose <strong>Configure</strong> to open the portlet’s Configuration settings.</td>
</tr>
<tr>
<td></td>
<td>Default: true</td>
<td></td>
</tr>
<tr>
<td><code>isCustomizeModeAvailable</code></td>
<td>false</td>
<td>Whether to render a <strong>Customize</strong> icon in the portlet header.</td>
</tr>
<tr>
<td></td>
<td>true</td>
<td>Site administrators choose <strong>Customize</strong> to edit a portlet’s default personalization data.</td>
</tr>
<tr>
<td></td>
<td>Default: true</td>
<td></td>
</tr>
<tr>
<td><code>isDetailModeAvailable</code></td>
<td>false</td>
<td>Whether to render a <strong>Details</strong> command on a PDK-Java portlet’s <strong>Actions</strong> menu.</td>
</tr>
<tr>
<td></td>
<td>true</td>
<td>Users choose <strong>Details</strong> to open the portlet in Full Screen mode.</td>
</tr>
<tr>
<td></td>
<td>Default: true</td>
<td></td>
</tr>
</tbody>
</table>

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.

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.
### Table 63–6 (Cont.) Portlet Modes Attributes of the Portlet Tag

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>isHelpModeAvailable</td>
<td>false</td>
<td>Whether to render a <strong>Help</strong> command on the portlet's <strong>Actions</strong> menu.</td>
</tr>
<tr>
<td></td>
<td>true</td>
<td>Users choose <strong>Help</strong> to open the portlet's Help page.</td>
</tr>
<tr>
<td></td>
<td>Default: true</td>
<td></td>
</tr>
<tr>
<td>isPrintModeAvailable</td>
<td>false</td>
<td>Whether to render a <strong>Print</strong> command on a JSR 286 portlet's <strong>Actions</strong> menu.</td>
</tr>
<tr>
<td></td>
<td>true</td>
<td>Users choose <strong>Print</strong> to display a printer-friendly version of the portlet.</td>
</tr>
<tr>
<td></td>
<td>Default: true</td>
<td></td>
</tr>
<tr>
<td>isNormalModeAvailable</td>
<td>false</td>
<td>Whether to render a <strong>Refresh</strong> command on the portlet's <strong>Actions</strong> menu.</td>
</tr>
<tr>
<td></td>
<td>true</td>
<td>Users choose <strong>Refresh</strong> to redraw the portlet independent of any other content on the page (also known as a partial-page refresh).</td>
</tr>
<tr>
<td></td>
<td>Default: true</td>
<td></td>
</tr>
<tr>
<td>isPersonalizeModeAvailable</td>
<td>false</td>
<td>Whether to render a <strong>Personalize</strong> icon in the portlet header.</td>
</tr>
<tr>
<td></td>
<td>true</td>
<td>Users choose <strong>Personalize</strong> to alter their personal view of the portlet.</td>
</tr>
<tr>
<td></td>
<td>Default: true</td>
<td>This mode is equivalent to the Edit mode selection in the Standards-based Java Portlet (JSR286) Wizard.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The <strong>Personalize</strong> 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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>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 <a href="#">Section 74.12, “Configuring Basic Authentication for Testing Portlet Personalization.”</a></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> A typical personalization setting is Portlet Title. You can set Portlet Title at design time, by providing a value for the <code>title</code> attribute. Consider however that supplying a value to the <code>title</code> attribute at design time prevents personalization and customization of the portlet title at runtime.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>isPreviewModeAvailable</td>
<td>false</td>
<td>Whether to enable previewing of portlet content.</td>
</tr>
<tr>
<td></td>
<td>true</td>
<td>This mode has no particular application in Portal 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.</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
<td></td>
</tr>
</tbody>
</table>

#### 63.6.7 Style Attributes of the Portlet Tag

Table 63–7 describes the style attributes of the portlet tag.
Setting Attribute Values for the Portlet Tag

Table 63–7  Style Attributes of the Portlet Tag

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contentStyle</td>
<td>One or more CSS styles.</td>
<td>The CSS style to apply to the portlet content. Values entered here take precedence over styles</td>
</tr>
<tr>
<td></td>
<td>These should be in compliance with, at least, CSS 2.0 and take the</td>
<td>specified in the inlineStyle attribute and those included in a CSS or skin on the specific</td>
</tr>
<tr>
<td></td>
<td>following format:</td>
<td>portlet instance. For more information, see Understanding contentStyle and inlineStyle</td>
</tr>
<tr>
<td></td>
<td>contentStyle=”color:rgb(255,0,255); font-family:Arial Helvetica</td>
<td>Properties.</td>
</tr>
<tr>
<td></td>
<td>Geneva</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sans-serif;font-size:large;“</td>
<td></td>
</tr>
<tr>
<td>inlineStyle</td>
<td>One or more CSS styles.</td>
<td>The CSS style to apply to the whole portlet. Values entered here take precedence over styles</td>
</tr>
<tr>
<td></td>
<td>These should be in compliance with, at least, CSS 2.0 and take the</td>
<td>included in a CSS or skin on the specific portlet instance. For more information, see</td>
</tr>
<tr>
<td></td>
<td>following format:</td>
<td>Understanding contentStyle and inlineStyle Properties</td>
</tr>
<tr>
<td></td>
<td>inlineStyle=”color:rgb(255,0,255); font-family:Arial Helvetica</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Geneva</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sans-serif;font-size:large;“</td>
<td></td>
</tr>
</tbody>
</table>

63.6.8 Binding Attributes of the Portlet Tag

Table 63–8 describes the binding attributes of the portlet tag.

Table 63–8  Binding Attributes of the Portlet Tag

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>binding</td>
<td>Name of a managed bean. For</td>
<td>A binding reference to store the component instance. The binding reference binds an instance of</td>
</tr>
<tr>
<td></td>
<td>example: binding=”#{frame</td>
<td>the portlet to a managed bean property. Managed beans are any JavaBeans used by the application</td>
</tr>
<tr>
<td></td>
<td>ActionsBean.Binding}”</td>
<td>that are registered in the JSF faces-config.xml file.</td>
</tr>
<tr>
<td></td>
<td>In the Property Inspector,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>click the Property Menu icon</td>
<td></td>
</tr>
<tr>
<td></td>
<td>next to the field and then</td>
<td></td>
</tr>
<tr>
<td></td>
<td>choose Edit to select a managed</td>
<td></td>
</tr>
<tr>
<td></td>
<td>bean and specify the relevant</td>
<td></td>
</tr>
<tr>
<td></td>
<td>managed bean property.</td>
<td></td>
</tr>
</tbody>
</table>

63.6.9 Customization Attributes of the Portlet Tag

Table 63–9 describes the customization attributes of the portlet tag.

Table 63–9  Customization Attributes of the Portlet Tag

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customizationAllowed</td>
<td>false</td>
<td>Whether design time customizations of the portlet tag are allowed on this portlet.</td>
</tr>
<tr>
<td></td>
<td>true</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Default: true</td>
<td></td>
</tr>
<tr>
<td>customizationAllowedBy</td>
<td>Text string</td>
<td>The roles for which design time customizations are allowed. This attribute enables you to allow</td>
</tr>
<tr>
<td></td>
<td></td>
<td>customizations, but restrict who can actually perform them.</td>
</tr>
</tbody>
</table>

63.6.10 Other Attributes of the Portlet Tag

Table 63–10 describes the other attributes of the portlet tag.
Manually Wiring Parameters and Events in JSR 286 Portlets

If the portlet that you add to your page defines parameters or events that it uses to dynamically alter its content, for the most part this happens automatically. For more information, see Section 63.5.3, "What You May Need to Know About Interportlet Communication."

However, in some circumstances you may find it necessary to turn off the default automatic parameter and event listening, or you may be using portlets with parameters or events with names that do not match or define appropriate aliases. In such situations, you must manually wire the parameters and events in your portlets.

This section includes the following topics:

- Section 63.7.1, "How to Manually Link Portlets with Public Render Parameters"
- Section 63.7.2, "How to Manually Link Portlets with Portlet Events"

### 63.7.1 How to Manually Link Portlets with Public Render Parameters

When you place a portlet on a page, any public render parameters with the same QName are automatically linked. For example, if one portlet publishes a value to a parameter with the name myParam, then if a second portlet on the page accepts values from a parameter with the same name, the second portlet is automatically updated with the appropriate value.

Portlets can also link together using parameters with different QNames if the portlet developer has set up aliases for the parameter.

The following example shows how to use public render parameters to link the behavior of two portlets using parameters with different names and no defined aliases. In the example, we take a generic Parameter Form portlet that allows us to update public render parameters, and show how that can be linked to a Stock Price portlet that renders stock prices, taking the stock symbol from a public render parameter.
When these two portlets are dropped onto the same page, they are not automatically linked together. This is because none of the generic parameter names in the Parameter Form portlet match the `stocksymbol` parameter in the Stock Price portlet, and the Stock Price portlet does not define any aliases for the `stocksymbol` parameter that match the generic parameter names in the Parameter Form portlet.

There are two methods for manually linking parameters: using page variables and using the `ParametersChange` ADF contextual event.

This section includes the following topics:

- Section 63.7.1.1, "Manually Linking Parameters Using Page Variables"
- Section 63.7.1.2, "Manually Linking Parameters Using the ParametersChange ADF Contextual Event"

### 63.7.1.1 Manually Linking Parameters Using Page Variables

When the portlets are dropped onto a page, page variables are created for each of the parameters supported by the portlets. In our example, there are three page variables for the Parameter Form portlet and one for the Stock Price portlet (Example 63–9).

#### Example 63–9 Page Variables Created for Portlet Parameters

```xml
<pageDefinition ...>
<parameters/>
<executables>
  <variableIterator id="variables">
    <variable Name="ParameterFormPortlet1_1_Parameter1" Type="java.lang.Object"/>
    <variable Name="ParameterFormPortlet1_1_Parameter2" Type="java.lang.Object"/>
    <variable Name="ParameterFormPortlet1_1_Parameter3" Type="java.lang.Object"/>
    <variable Name="StockPricePortlet1_1_stocksymbol" Type="java.lang.Object"/>
  </variableIterator>
  <portlet id="ParameterFormPortlet1_1">
    <parameters>
      <parameter name="Parameter1" pageVariable="ParameterFormPortlet1_1_Parameter1"/>
      <parameter name="Parameter2" pageVariable="ParameterFormPortlet1_1_Parameter2"/>
      <parameter name="Parameter3" pageVariable="ParameterFormPortlet1_1_Parameter3"/>
    </parameters>
    <events>
      <event eventType="ParametersChange" name="ParameterFormPortlet1_1_Event"/>
    </events>
  </portlet>
  <portlet id="StockPricePortlet1_1">
    <parameters>
      <parameter name="stocksymbol" pageVariable="StockPricePortlet1_1_stocksymbol"/>
    </parameters>
    <events>
      <event eventType="ParametersChange" name="StockPricePortlet1_1_Event"/>
    </events>
  </portlet>
<pageDefinition ...
```
To link the portlets, you can make the Stock Price portlet use the value from one of the page variables created for the Parameter Form portlet (Example 63–10).

Example 63–10  Linking Portlet Parameters Using Page Variables

```xml
<portlet id="StockPricePortlet1_1"

...>

<parameters>
  <parameter name="stocksymbol"
    pageVariable="ParameterFormPortlet1_1_Parameter1"/>
</parameters>

<events>
  <event eventType="ParametersChange"
    name="StockPricePortlet1_1_Event"/>
</events>

</portlet>
```

Now, entering a value in the first parameter field of the Parameter Form portlet sets the corresponding page variable to the same value, which in turn causes the Stock Price portlet to be updated with the new value.

63.7.1.2 Manually Linking Parameters Using the ParametersChange ADF Contextual Event

When you drop a portlet that supports parameters, as well as the page variables discussed in the previous section, an ADF ParametersChange contextual event is created for the portlet. This event is triggered when the values of the portlet's parameters change. The payload of the event is the new value of the parameter. You can use this event to set the value of another portlet's parameter by creating an event map in the page definition that maps the payload of the first portlet's event to the second portlet's parameter. Example 63–11 shows how this event map would look for our example.

Example 63–11  Linking Portlet Parameters Using the ParametersChange ADF Contextual Event

```xml
<eventMap xmlns="http://xmlns.oracle.com/adfm/contextualEvent">
  <event name="ParameterFormPortlet1_1_Event">
    <producer region="ParameterFormPortlet1_1">
      <consumer handler="StockPricePortlet1_1">
        <parameters>
          <parameter name="stocksymbol"
            value="${payLoad.Parameter1}"/>
        </parameters>
      </consumer>
    </producer>
  </event>
</eventMap>
```

Entering a value in the first parameter field of the Parameter Form portlet triggers the ParametersChange event. The payload of this event, the new parameter value, is then sent to the Stock Price Portlet and used to specify the value of the stocksymbol parameter.
63.7.2 How to Manually Link Portlets with Portlet Events

The following example shows how to manually link portlets using portlet events in the case where the names of the events defined for the different portlets do not match.

In the example, we have a portlet, the Department Locations portlet, which lists the geographical locations of a company's various offices. The Department Locations portlet raises a portlet event (latLong) when a particular department is selected in the portlet.

There is another portlet, the Map portlet, which displays a Google map. The Map portlet listens out for a portlet event (geolocation) to use to determine a location to display on a map.

When the two portlets are dropped on a page, there is no way for the Map portlet to know that the event raised by the Department Locations portlet provides information that it can use to display a map. To make this link between the two portlets, you must edit the page definition to explicitly create the mapping between the two events, as shown in Example 63–12.

Example 63–12  Explicit Event Mapping in the Page Definition

```
<pageDefinition ...>
  <parameters/>
  <executables>
    <variableIterator id="variables"/>
    <portlet id="DepartmentLocations1_1" ...
    <events>
      <event name="latLong">
        <portletEvent namespace-uri="http://xmlns.oracle.com/portlet/EventSample"
                       localpart="latLong"/>
      </event>
    </events>
    <portlet id="Map1_1" ...
    <events>
      <event name="geolocation">
        <portletEvent namespace-uri="http://xmlns.oracle.com/portlet/EventSample"
                       localpart="geolocation"/>
      </event>
    </events>
  </portlet>
  <bindings/>
  <eventMap xmlns="http://xmlns.oracle.com/adfm/contextualEvent">
    <event name="http://xmlns.oracle.com/portlet/EventSample}latLong">
      <producer region="DepartmentLocations1_1">
        <consumer handler="Map1_1.(http://xmlns.oracle.com/portlet/EventSample}geolocation"/>
      </producer>
    </event>
  </eventMap>
</pageDefinition>
```

63.8 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 topics:

- Section 63.8.1, "How to Copy a Portlet and Place it on the Same Page"
- Section 63.8.2, "How to Copy a Portlet from One Application Page to Another"

### 63.8.1 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 63–13).

#### Example 63–13 Code Fragment to be Copied When Copying a Portlet

```xml
<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:html>
</f:view>
```

Consuming Portlets 63-41
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 63–14).

**Example 63–14  Changing the Portlet ID**

```xml
<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:

```xml
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 63–15.

**Example 63–15  Creating a New Method for a Managed Bean in the Managed Bean Class**

```java
private PortletBase portlet2;
public void setPortlet2(PortletBase portet2) {
    this.portlet2 = portet2;
}

public PortletBase getPortlet2() {
    return portlet2;
}
```

10. From the main menu, choose File > Save All.
63.8.2 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 63–16).

**Example 63–16  Source Page Code Fragment to Be Copied When Copying a Portlet**

```html
<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_82235fc4e2b" 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 63.2.1, "Registering a WSRP Portlet Producer with a WebCenter Portal Framework Application" or Section 63.2.2, "Registering an Oracle PDK-Java Portlet Producer 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 63–17).

**Example 63–17 Code Fragment to Be Copied From a Page Definition File**

```xml
<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**.

### 63.9 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 63–18:

**Example 63–18  Deleting Portlet-Related Page Variables from a Page Definition File**

```xml
<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>
</pageDefinition>
```

7. From the main menu, choose, select **File > Save All**.

### 63.10 Consuming WebCenter Services Portlets

WebCenter Services Portlets is a preconfigured, out-of-the-box producer that enables you to expose WebCenter Portal tools and services 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
Consuming WebCenter Services Portlets

- Take Polls—Displays the most recently published available poll, or a specific poll identified by the pollId parameter
- 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 *Administering Oracle WebCenter Portal*.

---

This section includes the following topics:

- Section 63.10.1, "How to Consume WebCenter Services Portlets in Oracle Portal"
- Section 63.10.2, "How to Consume WebCenter Services Portlets in Oracle WebLogic Portal"
- Section 63.10.3, "How to Consume WebCenter Services Portlets in Oracle WebCenter Interaction"
- Section 63.10.4, "How to Set WebCenter Services Portlets Parameters"
- Section 63.10.5, "WebCenter Services Portlets Limitations"
- Section 63.10.6, "Troubleshooting Problems with WebCenter Services Portlets"

---

### 63.10.1 How to Consume 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 *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 *User’s Guide for Oracle Portal*. 
63.10.2 How to Consume 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.
   

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.

63.10.3 How to Consume 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 and Oracle JPDK Portlet Producers in the Pagelet Producer" section in Administering Oracle WebCenter Portal.

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.

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.

---

Note: You must select the same Token Profile that was specified when configuring WS-Security for WebCenter Services Portlets.
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.

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" section in the Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Interaction.

63.10.4 How to Set 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 topics:

- Section 63.10.4.1, "Common WebCenter Services Portlets Parameters"
- Section 63.10.4.2, "Document Manager Portlet Parameters"
- Section 63.10.4.3, "Discussion Forums Portlet Parameters"
- Section 63.10.4.4, "Blogs Portlet Parameters"
- Section 63.10.4.5, "Lists Portlet Parameters"
- Section 63.10.4.6, "Polls Manager Portlet Parameters"
- Section 63.10.4.7, "Take Polls Portlet Parameters"
- Section 63.10.4.8, "Announcements Portlet Parameters"
- Section 63.10.4.9, "Mail Portlet Parameters"
- Section 63.10.4.10, "Activity Stream Portlet Parameters"
- Section 63.10.4.11, "Tag Cloud Portlet Parameters"

63.10.4.1 Common WebCenter Services Portlets Parameters

Table 63–11 lists the parameters that are available for all the portlets provided by WebCenter Services Portlets.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>width</td>
<td>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.</td>
</tr>
<tr>
<td>height</td>
<td>Specifies the height of the portlet on the consuming page. If not specified, the portlet height is determined by the portlet consumer.</td>
</tr>
</tbody>
</table>

63.10.4.2 Document Manager Portlet Parameters

Table 63–12 lists the additional parameters available for the Document Manager portlet.
### Table 63–12  Document Manager Portlet Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| **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 for WebCenter Portal or Portal Framework Applications" section in *Administering 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, multfile-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  
  - `${'table'} `—Displays only the contents of the current folder in a single pane, with the capability to click a folder to drill down, refreshing the pane with the folder contents  
  - `${'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'}`
63.10.4.3 Discussion Forums Portlet Parameters

Table 63–13 lists the additional parameters available for the Discussion Forums portlet.

Table 63–13 Discussion Forums Portlet Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>categoryId</td>
<td>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.</td>
</tr>
<tr>
<td>forumId</td>
<td>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.</td>
</tr>
<tr>
<td>showRecursiveForums</td>
<td>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.</td>
</tr>
<tr>
<td>isCategoryView</td>
<td>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.</td>
</tr>
</tbody>
</table>

63.10.4.4 Blogs Portlet Parameters

Table 63–14 lists the additional parameters available for the Blogs portlet.

Table 63–14 Blogs Portlet Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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. |
63.10.4.5 Lists Portlet Parameters
The Lists portlet does not include any additional parameters.

63.10.4.6 Polls Manager Portlet Parameters
Table 63–15 lists the additional parameters available for the Polls Manager portlet.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>showUserDataOnly</td>
<td>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.</td>
</tr>
</tbody>
</table>

63.10.4.7 Take Polls Portlet Parameters
Table 63–16 lists the additional parameters available for the Take Polls portlet.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pollId</td>
<td>The ID of the poll to display.</td>
</tr>
</tbody>
</table>

63.10.4.8 Announcements Portlet Parameters
Table 63–17 lists the additional parameters available for the Announcements portlet.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parentId</td>
<td>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 announcements default to global announcements.</td>
</tr>
</tbody>
</table>

63.10.4.9 Mail Portlet Parameters
Table 63–18 lists the additional parameters available for the Mail portlet.
### Activity Stream Portlet Parameters

Table 63–19 lists the additional parameters available for the Activity Stream portlet.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>advancedQuery</td>
<td>A field for specifying a custom query to filter streamed items. For more information, see Section 39.4, &quot;About the Activity Stream Advanced Query Option.&quot;</td>
</tr>
<tr>
<td>enableContextInfo</td>
<td>A means of showing or omitting detailed information about the object in the current context (that is, in a popup or other contextual instrument).</td>
</tr>
<tr>
<td></td>
<td>- 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.</td>
</tr>
<tr>
<td></td>
<td>- Enter {false} to omit object detail information.</td>
</tr>
<tr>
<td></td>
<td>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.</td>
</tr>
<tr>
<td>hideComments</td>
<td>A means of showing or hiding the Comments feature on streamed activities. Enter true to hide the Comments feature. Enter false to show it.</td>
</tr>
<tr>
<td>hideConfigure</td>
<td>A means of hiding the personalization option on the task flow instance.</td>
</tr>
<tr>
<td>hideInlinePreview</td>
<td>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.</td>
</tr>
<tr>
<td>hideLike</td>
<td>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.</td>
</tr>
<tr>
<td>hideShare</td>
<td>A means of showing or hiding the Share menu on streamed activities. Enter true to hide the Share menu. Enter false to show it.</td>
</tr>
<tr>
<td>pageSize</td>
<td>The number of items to stream in a given task flow instance.</td>
</tr>
<tr>
<td>pagination</td>
<td>The form of pagination to use on a multipage stream. Enter {true} to show Previous and Next links. Enter {false} to hide Previous and Next links.</td>
</tr>
</tbody>
</table>

Instead, a more link is rendered, enabling users to navigate to a fuller view of the task flow where all streamed activities are shown.
### Tag Cloud Portlet Parameters

The Tag Cloud portlet does not include any additional parameters.

### WebCenter Services Portlets Limitations

This section identifies limitations with WebCenter Portal services when they are accessed remotely using WebCenter Services Portlets.

- WebCenter Services Portlets do not include the **Send Mail** option to email links to resources, for example wikis.
- WebCenter Services Portlets do not support the **View Profile** option when clicking a user name.
- WebCenter Services Portlets do not support recommendations.
- WebCenter Services Portlets do not support anonymous access.
- WebCenter Services Portlets no longer supports a Worklist portlet.


- When creating a link to a discussion forum from another WebCenter Services Portlet, such as the Document Manager or Blogs portlet, only discussion forums with a category ID of 0 are listed.
- RSS is not supported within the Discussion Forums, Lists, and Announcements portlets.
- The Mail portlet does not support the **Find Users** option when composing messages. Recipients must be entered manually.

---

**Table 63–19 (Cont.) Activity Stream Portlet Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>profileOnly</td>
<td>A means of streaming activities only from user profiles.</td>
</tr>
<tr>
<td></td>
<td>- Enter <strong>true</strong> to stream only those activities associated with user profiles</td>
</tr>
<tr>
<td></td>
<td>- Enter <strong>false</strong> to stream other types of activities along with those associated with user profiles</td>
</tr>
<tr>
<td>resourceId</td>
<td>The current user ID.</td>
</tr>
<tr>
<td></td>
<td>Enter <code>#{securityContext.userName}</code> to return the current user.</td>
</tr>
<tr>
<td>serviceCategories</td>
<td>A field for entering a comma-separated list of names of services from which to stream activities.</td>
</tr>
<tr>
<td></td>
<td>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:</td>
</tr>
<tr>
<td></td>
<td><code>oracle.webcenter.collab.announcement</code>, <code>oracle.webcenter.collab.forum</code></td>
</tr>
<tr>
<td></td>
<td>For a list of valid service ID, see <a href="#">Table G–7</a>. 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.</td>
</tr>
</tbody>
</table>
The Download URL option is not available from the View menu in the Document Manager. Instead, users must access the URL provided in the Direct URL field to view the document, and then click the Download button.

The Edit with Word option is not available from the File menu in the Document Manager portlet.

When using a link from the Get a Link option of the Document Manager or Blogs portlet, you must open the link in a new browser session. You cannot open the link in a new tab of the same browser session that is currently running the portlet.

The Subscribe option, accessed from the File menu, is not supported in the Document Manager or Blogs portlets.

The Share option is not available from the Edit menu when viewing a blog in the Blogs portlet.

63.10.6 Troubleshooting Problems with WebCenter Services Portlets

This section provides information to assist you in troubleshooting problems you may encounter when using WebCenter Services Portlets.

This section includes the following topics:

- Section 63.10.6.1, "External Link in Announcements or Discussion Forum Portlet Not Working"
- Section 63.10.6.2, "Get a Link Option in Document Manager and Blogs Portlets Not Working"
- Section 63.10.6.3, "Recommendations Tab in Document Manager and Blogs Portlets Is Empty"
- Section 63.10.6.4, "Unable to Embed an Image into a Blog"
- Section 63.10.6.5, "Selected Text Is Lost When Creating a Link in a Blog Post or Wiki"

63.10.6.1 External Link in Announcements or Discussion Forum Portlet Not Working

Problem
When users click a link to an external web page in the Announcements or Discussion Forum portlet nothing displays.

Cause
WebCenter Services Portlets are displayed inside inline frames. Some websites, for example Google, do not display inside an inline frame if the parent frame does not have the same origin. Therefore, if you link to one of these websites within the Announcements or Discussion Forum portlet, the link does not display.

Solution
When linking to an external web page from the Announcements or Discussion Forums portlet, it is recommended that you display the target in a new browser tab or window. To do this, add the target="_blank" attribute to the link. For example:

\(<a \ href=\"http://www.google.com\" \ target=\"_blank\">Google</a>\)
63.10.6.2 Get a Link Option in Document Manager and Blogs Portlets Not Working

**Problem**
When consuming the Document Manager or Blogs portlet in Oracle WebCenter Interaction, the link provided from the Get a Link option of the View menu does not work in Microsoft Internet Explorer.

**Cause**
Oracle WebCenter Interaction accesses WebCenter Services Portlets through a pagelet producer. As such, when you consume a WebCenter Services Portlets portlet on a page in Oracle WebCenter Interaction, the portlet's underlying pagelet is accessed using an injectpagelet JavaScript call. This can result in links within the portlet that exceed Internet Explorer's 2KB limit for URLs.

**Solution**
Replace the injectpagelet JavaScript call with an HTML portlet that references the pagelet in an inline frame. An inline frame results in shorter URLs which are less likely to exceed Internet Explorer's 2KB limit.

For example, by default, Oracle WebCenter Interaction might provide the following call to access the Document Manager portlet:

```
<script type='text/javascript'>
  injectpagelet('ServicesPortlets_PS6_49823a17-f56d-41cf-b50e-3506b4394f25',
  'Document_Manager', 'iframe ifheight=650px ifwidth=100%", "',
  'resourceId=dev-ucm%23dDocName%3ADSS000411", "", "", "none", false);
</script>
```

Replace this call with an HTML portlet that includes the following inline frame:

```
<iframe id='pt-pagelet-iframe-1' frameborder='0' height='650px' width='100%' src='http://server.example.com:7777/inject/v2/pagelet/
ServicesPortlets_PS6_49823a17-f56d-41cf-b50e-3506b4394f25/Document_Manager
?content-type=html&csapi=true&instanceid=1&chrome=none&consumepage=true
&resourceId=dev-ucm%23dDocName%3ADSS000411'>
</iframe>
```

63.10.6.3 Recommendations Tab in Document Manager and Blogs Portlets Is Empty

**Problem**
The Recommendations tab in the Documentation Manager and Blogs portlets does not list any recommendations.

**Cause**
WebCenter Services Portlets do not support recommendations.

**Solution**
To remove the Recommendations tab from the Document Manager or Blogs portlet, edit the portlet and set the featureOff parameter to recommendations.
63.10.6.4 Unable to Embed an Image into a Blog

Problem
When consuming the Blogs portlet in Oracle WebCenter Interaction, images cannot be embedded in a blog post.

Cause
If the file name of the image is long, for example 250 characters or more, the length of the resource URL used for the image in the blog may exceed the supported limit.

Solution
Reduce the length of the image file name.
Alternatively, it is recommended that when consuming WebCenter Services Portlets in Oracle WebCenter Interaction, you should place portlets within an HTML portlet. For more information, see the solution in Section 63.10.6.2, "Get a Link Option in Document Manager and Blogs Portlets Not Working."

63.10.6.5 Selected Text Is Lost When Creating a Link in a Blog Post or Wiki

Problem
When you select text in a blog post or wiki and click the Select Resource or New Resource icon to create a link, the selected text disappears.

Cause
The selected text is temporarily replaced with an insertion cursor while the link is created.

Solution
No action is required. When you finish creating the link, or cancel the operation, the selected text is restored.

63.11 Troubleshooting Portlets

This section provides information to assist you in troubleshooting problems you may encounter while using portlets. It contains the following topics:

- Section 63.11.1, "Diagnostic Tools for Troubleshooting Portlets"
- Section 63.11.2, "Configuring the Portlet Logging File"
- Section 63.11.3, "Portlet Displays a Portlet Consumer Error"
- Section 63.11.4, "Portlet Displays a Portlet Timeout"
- Section 63.11.5, "Portlet Displays a Remote Portlet Communication Error"
- Section 63.11.6, "Portlet Displays a Remote Portlet Error"

63.11.1 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 topics:

- Section 63.11.1.1, "Identify the Portlet Instance"
- Section 63.11.1.2, "Examine the Portlet Consumer Test Page"
- Section 63.11.1.3, "Examine the Producer Test Page"

63.11.1.1 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 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
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.

### 63.11.1.2 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 63–2) to locate the portlet producer and, if necessary, test the portlet in isolation.
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 describes how to use the Portlet Consumer Test Page to diagnose portlet issues. It includes the following topics:

- **Section 63.11.1.2.1**, "Access the Portlet Consumer Test Page"
- **Section 63.11.1.2.2**, "Locate the Portlet Producer"
- **Section 63.11.1.2.3**, "Locate and Run the Portlet Instance"
- **Section 63.11.1.2.4**, "Perform Sanity Checks"
- **Section 63.11.1.2.5**, "Check Consumer Configuration Entries"

### 63.11.1.2.1 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:

2. In the Portlet Consumer Test Page, you can perform further diagnostic steps as described in the following sections.

**63.11.1.2.2 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 63.11.1.1, "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.

**63.11.1.2.3 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 63.11.1.1, "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 63.11.1.1, "Identify the Portlet Instance."
Troubleshooting Portlets

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.

63.11.1.2.4 Perform Sanity Checks The Sanity Checks tab of the Portlet Consumer Test Page (shown in Figure 63–3) provides a quick overview of the state of portlet communication in your application across all producers.

Figure 63–3 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.

**63.11.1.2.5 Check Consumer Configuration Entries** The Configuration tab of the Portlet Consumer Test Page (shown in Figure 63–4) 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 63–4 The Configuration Tab](image)

**63.11.1.3 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 63–5) to identify potential issues with the portlet producer application.
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 topics:

- **Section 63.11.1.3.1, “Access the Producer Test Page”**
Section 63.11.3.2, "Run the JSF Portlet as a Servlet"

Section 63.11.3.3, "Examine the SOAP Monitor"

63.11.3.1 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.

63.11.3.2 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.

63.11.3.3 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.
Troubleshooting Portlets

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.

### 63.11.2 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 63–19 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 63–19 Configuring Log Files for Troubleshooting Portlet Issues**

```xml
<!-- 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"/>
</log_handler>
```
<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.warp" 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
```
63.11.3 Portlet Displays a Portlet Consumer Error

The message **Portlet Consumer Error** (shown in Figure 63–6) typically indicates that an error occurred within the operation of the portlet parts of the portlet consumer application.

**Figure 63–6  Portlet Displaying a Portlet Consumer Error**

![Portlet Displaying a Portlet Consumer Error](image)

**Problem 1**
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 1**
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 63–20 shows a message logged for a portlet error.

**Example 63–20  Example Message Logged for a Portlet Error**

```xml
<PortletRenderer> <setErrorState> <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.

**Problem 2**
The portlet queue is full and the pool is not able to process the new request. This can happen if the portal is under a particularly high load. When the portlet queue is full, no new portlet requests are processed.

If this is the cause of the problem, the diagnostic log contains an error message similar to the one shown in Example 63–21.
**Example 63–21  Example Message Logged for Portlet Queue Full**

Message oracle.portlet.client.container.PortletQueueFullException: Queue Full

```java
cause=na submittedTasks=8,361 queuedTasks=20 queueFreeSpace=0
completedTasks=8,331 activeThreads=10 corePoolSize=10 keepAliveTime=9,223,372,036
largestPoolSize=10 maxPoolSize=10 poolSize=10 isShutdown=false isTerminated=false
isTerminating=false
```

**Solution 2**

The portlet queue size is determined by the `parallelPoolSize` and `parallelQueueSize` parameters in `adf-config.xml`.

- `parallelPoolSize` indicates the number of threads to use for parallel execution of tasks. The default value is 10.
- `parallelQueueSize` determines the size of the queue of tasks waiting for parallel execution. Tasks are rejected when the queue size is reached. The default value is 20.

The default values should be adequate for most portals. However, if you experience recurring errors due to the portlet queue becoming full, you might want to consider increasing the value of the `parallelQueueSize` parameter. To boost performance, you might also consider increasing the `parallelPoolSize` to increase the number of threads opened to handle the portlet requests from the queue.

---

**Note:** Increasing the queue and pool size should be done carefully and in small increments. If the value of either parameter is set too high the increased usage of hardware resources could be counterproductive.

For information about how to change the `parallelQueueSize` and `parallelPoolSize` parameters, see Section 63.3.5, "How to Edit Portlet Client Configuration."

### 63.11.4 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 63–7), 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 63–7  Portlet Displaying a Portlet Timeout Error**

<table>
<thead>
<tr>
<th>Application</th>
<th>task-flow-definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Portlet</td>
<td>Timeout</td>
</tr>
</tbody>
</table>

**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.

63.11.5 Portlet Displays a Remote Portlet Communication Error

When a section of the screen shows the Remote Portlet Communication Error message (as shown in Figure 63–8), 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.

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 Security and Administrator’s Guide for Web Services.

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:


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

63.11.6 Portlet Displays a Remote Portlet Error

If the portlet displays a Remote Portlet Error (as shown in Figure 63–9), 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 63–9 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 Cause 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 bug in the portlet code.

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

Ensure that the persistence store is available and that the configuration of the persistence store in the web.xml file is correct.

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

Ensure that the persistence store is available and that the configuration of the persistence store in the web.xml file is correct.

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.
This chapter describes how to use OmniPortlet in your Oracle JDeveloper environment.

This chapter includes the following topics:

- Section 64.1, "Introduction to OmniPortlet"
- Section 64.2, "Adding OmniPortlet to Your Application"
- Section 64.3, "Customizing OmniPortlet"
- Section 64.4, "OmniPortlet Configuration Tips"
- Section 64.5, "Troubleshooting OmniPortlet Problems"

For information on how to use the OmniPortlet wizard, refer to the "Working with OmniPortlet" chapter in *Building Portals with Oracle WebCenter Portal*.

### 64.1 Introduction to OmniPortlet

OmniPortlet is a data publishing portlet, provided as a subcomponent of Oracle WebCenter Portal. OmniPortlet enables developers to easily publish data from various data sources using a variety of layouts without writing any code. You add OmniPortlet to your application at design time, and customize it at runtime with an easy to use wizard. You can base an OmniPortlet on a variety of data sources, including SQL, XML, web services, spreadsheets (character-separated values), and even application data from an existing web page. You can publish this data to several different layouts, such as customizable charts and tables. Figure 64–1 shows an example of a portlet created using OmniPortlet.

#### Figure 64–1 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.

64.2 Adding OmniPortlet to Your Application

As described in Chapter 63, "Consuming Portlets," you can add an OmniPortlet to a page created through Oracle JDeveloper. OmniPortlet is included in the preconfigured PortalTools portlet producer in the Integrated WebLogic Server (Integrated WLS) that is installed with Oracle JDeveloper.

To add OmniPortlet to your application:

1. Start Integrated WLS. For more information, see Section 2.2.4, "Managing the Integrated WebLogic Server."
2. Deploy the preconfigured PortalTools portlet producer, if it has not already been deployed. For more information, see Section 2.4.1.1, "Deploying the Preconfigured Portlet Producers."
3. Register the OmniPortlet producer with your application. For more information, see Section 63.2.2, "Registering an Oracle PDK-Java Portlet Producer with a WebCenter Portal Framework Application."
4. Add the OmniPortlet to your application by dragging it to the desired JSPX page. For more information, see Section 63.5, "Adding Portlets to a 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 64.4, "OmniPortlet Configuration Tips."

64.3 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 the "Working with OmniPortlet" chapter in Building Portals with Oracle WebCenter Portal.
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 *Using Oracle WebCenter Portal*.

## 64.4 OmniPortlet Configuration Tips

This section contains configuration information for OmniPortlet. To learn more about the OmniPortlet wizard, see Chapter 64, "Creating Portlets with OmniPortlet."

This section includes the following topics:
- **Section 64.4.1, "Configuring the OmniPortlet Producer to Access Data Outside a Firewall"
- **Section 64.4.2, "Configuring the OmniPortlet Producer to Access Other Relational Databases"
- **Section 64.4.3, "Configuring Portal Tools and Web Producers (Optional)"
OmniPortlet Configuration Tips

64.4.1 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: 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 Portal 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 11g 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
```

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 64–1, if you are using an HTTP Proxy Host, or Table 64–2, if you are using an HTTPS Proxy Host.

### Table 64–1 HTTP Proxy Information Command Line Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>http.proxyHost</td>
<td>The host name of a proxy server if one is required to make a URL connection from the OmniPortlet producer to its data sources.</td>
</tr>
<tr>
<td>http.proxyPort</td>
<td>The port number for the HTTP Proxy Host.</td>
</tr>
<tr>
<td>http.nonProxyHosts</td>
<td>The name of any domain or hosts to which you can directly connect, bypassing a proxy server, such as your local machine: localhost</td>
</tr>
<tr>
<td></td>
<td>Hosts can be fully qualified host names or can be IP addresses.</td>
</tr>
</tbody>
</table>
Table 64–1 (Cont.) HTTP Proxy Information Command Line Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>http.proxyUser</td>
<td>The user to log in to the proxy server if the proxy server requires authentication.</td>
</tr>
<tr>
<td>http.proxyPassword</td>
<td>The password to log in to the proxy server if the proxy server requires authentication.</td>
</tr>
<tr>
<td>http.proxyAuthType</td>
<td>The authentication type of the proxy server. Acceptable values: Basic</td>
</tr>
<tr>
<td>http.proxyAuthRealm</td>
<td>The name of the realm of the proxy server. If you do not know the name of the realm, contact the proxy server administrator.</td>
</tr>
</tbody>
</table>

Table 64–2 HTTPS Proxy Information Command Line Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>https.proxyHost</td>
<td>The host name of a proxy server if one is required to make a URL connection from the OmniPortlet producer to its data sources.</td>
</tr>
<tr>
<td>https.proxyPort</td>
<td>The port number for the HTTPS Proxy Host.</td>
</tr>
<tr>
<td>https.nonProxyHosts</td>
<td>The name of any domain or hosts to which you can directly connect, bypassing a proxy server, such as your local machine: localhost</td>
</tr>
<tr>
<td></td>
<td>Hosts can be fully qualified host names or can be IP addresses.</td>
</tr>
<tr>
<td>https.proxyUser</td>
<td>The user to log in to the proxy server if the proxy server requires authentication.</td>
</tr>
<tr>
<td>https.proxyPassword</td>
<td>The password to log in to the proxy server if the proxy server requires authentication.</td>
</tr>
<tr>
<td>https.proxyAuthType</td>
<td>The authentication type of the proxy server. Acceptable values: Basic</td>
</tr>
<tr>
<td>https.proxyAuthRealm</td>
<td>The name of the realm of the proxy server. If you do not know the name of the realm, contact the proxy server administrator.</td>
</tr>
</tbody>
</table>

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 64–3 for a list of parameters and their descriptions.

Table 64–3 Provider.xml Tags

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>httpProxyHost</td>
<td>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.</td>
</tr>
<tr>
<td>httpProxyPort</td>
<td>Enter the port number for the HTTP Proxy Host.</td>
</tr>
</tbody>
</table>
The following is a basic example of using a proxy to access data outside a firewall:

```xml
<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:

```xml
<proxyInfo class="mycompany.portal.provider.v2.ProxyInformation">
  <httpProxyHost>myport.example.com</httpProxyHost>
  <httpProxyPort>8080</httpProxyPort>
  <proxyUseAuth>true</proxyUseAuth>
  <proxyType>Basic</proxyType>
  <proxyRealm>myport</proxyRealm>
</proxyInfo>
```
64.4.2 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 Microsystems'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 includes the following topics:
- Section 64.4.2.1, "Installing DataDirect JDBC Drivers"
- Section 64.4.2.2, "Registering DataDirect Drivers in OmniPortlet"

64.4.2.1 Installing DataDirect JDBC Drivers

The following DataDirect JDBC drivers are included with the WebLogic Server installation:
- YMutil.jar
- YMsybase.jar
- YMsqserver.jar
- YMspy.jar
- YMinformix.jar
- YMdb2.jar
- YMbase.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.

64.4.2.2 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 (default connection):**
  ```xml
  <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>
  ```

- **Microsoft SQL Server (named connection, for example, mycompany.com:port;databaseName=mydb):**
  ```xml
  <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>
    <connHandlerClass>oracle.webdb.reformlet.data.jdbc.JDBCODBCConnectionHandler</connHandlerClass>
    <connPoolSize>5</connPoolSize>
    <loginTimeOut>60</loginTimeOut>
  </driverInfo>
  ```

- **Sybase:**
  ```xml
  <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:**
  ```xml
  <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>
  ```
OmniPortlet Configuration Tips

Creating Portlets with OmniPortlet

Table 64–4 describes the parameters in the driverInfo property.

Table 64–4  Parameters in the driverInfo Property

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the database you want to use. This name is used on the Source tab of the OmniPortlet wizard.</td>
</tr>
<tr>
<td>sourceDataBase</td>
<td>Internal value. Set the value to other.</td>
</tr>
<tr>
<td>subProtocol</td>
<td>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.</td>
</tr>
<tr>
<td>connectString</td>
<td>Description of the connect string format. For DataDirect drivers, the format is: mainProtocol:subProtocol://databaseName</td>
</tr>
<tr>
<td>driverClassName</td>
<td>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.</td>
</tr>
<tr>
<td>dataSourceClassName</td>
<td>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 64–5 for the right data source class name for your driver.</td>
</tr>
</tbody>
</table>
| 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.JDBCDBCConnectionHandler  
  ■ For OmniPortlet versions before 9.0.4.1:  
    oracle.webdb.reformlet.data.jdbc.JDBCODBCConnectionHandler |
Table 64–5 lists the values for the `driverClassName` and `dataSourceClassName` properties for specific DataDirect JDBC drivers.

<table>
<thead>
<tr>
<th>DataDirect Drivers Supported</th>
<th>Properties</th>
</tr>
</thead>
</table>
| Microsoft SQL Server         | Parameter: `driverClassName`  
 Value: `weblogic.jdbc.sqlserver.SQLServerDriver` |
| 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 `FMW_HOME/user_projects/domains/wc_Domain/bin` then issue the following command:

   ```
   ./startManagedWebLogic.sh WC_Portlet
   ```

**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 64, "Creating Portlets with OmniPortlet."

### 64.4.3 Configuring Portal Tools and Web Producers (Optional)

To ensure that the OmniPortlet producer, 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 persistence store migration and upgrade utility that you can use to migrate the existing customizations to the database and upgrade customizations from earlier releases. See Section 60.3.8.2, "Migrating a PDK-Java Producer Persistence Store" for more information about the PDK-Java persistence 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 persistence store in Section 60.3.8.1, "Setting up a Persistence Store for a PDK-Java Producer."

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 persistence store migration utility.

To run the migration utility:

1. Navigate to the WebCenter Portal Oracle home directory using the following command:

   ```bash
cd WC_ORACLE_HOME
   ```

2. Run the following command to migrate OmniPortlet data from a file-based persistence store (FilePreferenceStore) to the database persistence store (DBPreferenceStore):

   ```bash
   ```

   See Section 60.3.8.2, "Migrating a PDK-Java Producer Persistence Store" for more information about the PDK-Java persistence store migration utility.

2. Typically, you perform the HTTP Proxy configuration for OmniPortlet 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 64.4.1, "Configuring the OmniPortlet Producer to Access Data Outside a Firewall."

   c. Propagate the changes made to the `provider.xml` file to middle tier M2 by copying `OmniPortlet_WAR_DIR/WEB-INF/providers/omniPortlet/provider.xml` from M1 to M2.


5. Verify that the OmniPortlet producer works properly through the LBR, by going to the test pages at the following URL:

   `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.

**Note:** To use the Web Page Data Source for OmniPortlet, you must also enable session binding in Oracle Web Cache.

## 64.5 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 Portal 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`. 
Creating Content-Based Portlets with Web Clipping

This chapter describes how to use Web Clipping in your Oracle JDeveloper environment.

This chapter includes the following topics:

- Section 65.1, "Introduction to Web Clipping"
- Section 65.2, "Adding Web Clipping to Your Application"
- Section 65.3, "Integrating Authenticated Web Content Using Single Sign-On"
- Section 65.4, "Advanced Features of Web Clipping"
- Section 65.5, "Web Clipping Portlet Configuration Tips"
- Section 65.6, "Current Limitations of Web Clipping"
- Section 65.7, "Troubleshooting Web Clipping"

For information about how to use Web Clipping at runtime, see the "Working with the Web Clipping Portlet" chapter in Building Portals with Oracle WebCenter Portal.

**Note:** Instead of using Web Clipping, consider using a clipper pagelet using Oracle WebCenter Portal's Pagelet Producer. For more information, see the "Managing the Pagelet Producer" chapter in Administering Oracle WebCenter Portal.

### 65.1 Introduction to 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.

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 Portal Framework application as a portlet.

Web Clipping supports the following features:

- **Navigation through various styles of login mechanisms**
Introduction to Web Clipping

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 Portal Framework application) automatically log in using a user name and password that you provide, while valid users (authenticated into the Portal Framework application) log in using a user name and password that they provide.

  For more information, see Section 65.5.2, "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 65.5.3.1, "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 65.4.1, "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 65.6, “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 Services (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 65.5.1, “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.

### 65.2 Adding Web Clipping to Your Application

As described in Chapter 63, “Consuming Portlets,” you can add a Web Clipping portlet to a page created through Oracle JDeveloper. The Web Clipping portlet producer is included in the preconfigured PortalTools portlet producer in the Integrated WebLogic Server (Integrated WLS) that is installed with Oracle JDeveloper.

To add Web Clipping to your application:

1. Start Integrated WLS. For more information, see Section 2.2.4, "Managing the Integrated WebLogic Server."

2. Deploy the preconfigured PortalTools portlet producer, if it has not already been deployed. For more information, see Section 2.4.1.1, "Deploying the Preconfigured Portlet Producers."

3. Register the Web Clipping producer with your application. For more information, see Section 63.2.2, "Registering an Oracle PDK-Java Portlet Producer with a WebCenter Portal Framework Application."

4. Add the Web Clipping portlet to your application by dragging it to the desired JSPX page. For more information, see Section 63.5, "Adding Portlets to a Page."
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.

To integrate an external application that requires authentication:

1. Open your Portal Framework application in JDeveloper.
2. Register the external application, specifying the authentication information. For more information, see Section 74.13.3.2.1, "Registering an External Application in Oracle JDeveloper."
3. Register the Web Clipping producer. For information, see Section 63.2.2, "Registering an Oracle PDK-Java Portlet Producer with a WebCenter Portal Framework Application."

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 check box, and from the list of values, select the external application that you just registered. The Enable Producer Sessions check box gets selected automatically.

Figure 65–1 shows the external application named MyOracleSupport associated with a Web Clipping producer.

---

**Note:** When you add an instance of the Web Clipping portlet 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 Web Clipping portlet in its own inline frame (iframe) in the View mode.

---

**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 65.5.3.1, "Adding Certificates for Trusted Sites."

---

65.3 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. Portal Framework application users access the authenticated content by using their user name and password for that system, not your credentials.

---

Note: When you add an instance of the Web Clipping portlet 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 Web Clipping portlet in its own inline frame (iframe) in the View mode.

**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 65.5.3.1, "Adding Certificates for Trusted Sites."

Note: You can associate only one external application with a producer. For each external application, you must register a new producer. Portal 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 Portal Framework application in JDeveloper.
2. Register the external application, specifying the authentication information. For more information, see Section 74.13.3.2.1, "Registering an External Application in Oracle JDeveloper."
3. Register the Web Clipping producer. For information, see Section 63.2.2, "Registering an Oracle PDK-Java Portlet Producer with a WebCenter Portal Framework Application."

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 check box, and from the list of values, select the external application that you just registered. The Enable Producer Sessions check box gets selected automatically.

Figure 65–1 shows the external application named MyOracleSupport associated with a Web Clipping producer.
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 65–2 shows the login screen of an external application named MyOracleSupport.

Figure 65–2 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 65–3).
Advanced Features of Web Clipping

65.4 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 topics:

- Section 65.4.1, "Using the Web Clipping Transport API"
- Section 65.4.2, "Rewriting Image Links to Use a Resource Proxy"

65.4.1 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 topics:

- Section 65.4.1.1, "Implementing the Web Clipping Transport API"
- Section 65.4.1.2, "Deploying the Web Clipping Transport API"

65.4.1.1 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:


   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:
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 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

wherewhere mytransport.jar refers to the JAR file users want to create.

65.4.1.2 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.

65.4.2 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>
65.5 Web Clipping Portlet Configuration Tips

Before you use Web Clipping, you must perform some administrative tasks. This section includes the following topics.

- Section 65.5.1, "Web Clipping Repository Configuration"
- Section 65.5.2, "HTTP or HTTPS Proxy Configuration"
- Section 65.5.3, "Web Clipping Producer Security"

65.5.1 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 65–4 shows the Provider Test Page of Web Clipping.
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:

```bash
> 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 Portal 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 11g 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 topics:

- Section 65.5.1.1, "Using Oracle Metadata Services (MDS) as the Web Clipping Repository"
- Section 65.5.1.2, "Using an Oracle Database as the Web Clipping Repository"
- Section 65.5.1.3, "Configuring Web Clipping Repository in provider.xml"
- Section 65.5.1.4, "Attributes and Child Tags of the repositoryInfo Tag"
Section 65.5.1.5, "Migrating the Web Clipping Repository"

65.5.1.1 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.

Example 65–1 shows MDS as the default repository in provider.xml.

Example 65–1  MDS as the Default Web Clipping Repository in provider.xml

```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 Portal 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:

```xml
<property name="metadata-path" value="portletdata/tools/webClipping"/>
```

The location specified for value is relative to JDEV_HOME/portlet. Any relative path specified is interpreted to be relative to JDEV_HOME/portlet. 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.

65.5.1.2 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.
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 9i or later

Note: You cannot use a Microsoft SQL Server or IBM DB2 database as the Web Clipping repository.

This section includes the following topics:

- Section 65.5.1.2.1, "Using the Database Schema Created by RCU"
- Section 65.5.1.2.2, "Creating a Database Schema"

65.5.1.2.1 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 Oracle WebCenter Portal" chapter in the Installation Guide for Oracle WebCenter Portal.

65.5.1.2.2 Creating a Database Schema

You can use any user-defined schema for an Oracle 9i 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 65–2.

Example 65–2 Java Command for Creating a Schema for Web Clipping Portlet Definitions and Clippings

```
```

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:

```java
jdbc:oracle:thin:@//dbhost:dbport/service_name
```

For example:

```java
jdbc:oracle:thin:@//shobeen:1521/sales_us
```

**Note:** The classpath in Example 65–2 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.

### 65.5.1.3 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 65–3.
   - Use the database schema, created for an Oracle database 9i or later, where you can manually specify connection information.

**Example 65–3  Oracle Database as the Web Clipping Repository**

```xml
<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 65.5.1.4, "Attributes and Child Tags of the repositoryInfo Tag."

- Use the database schema, created for an Oracle database 9i or later, where you can manually specify connection information.

To specify Oracle 9i or later database as the Web Clipping repository, specify database connection parameters in the entry shown in Example 65–4.

**Example 65–4  Setting Oracle Database 9i or Later as Web Clipping Repository**

```xml
<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 65.5.1.4, "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 65.5.3, "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.

### 65.5.1.4 Attributes and Child Tags of the repositoryInfo Tag

Table 65–1 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 Portal Framework application implementations.
# Web Clipping Portlet Configuration Tips

## Table 65–1 Attributes and Child Tags of the repositoryInfo Tag

<table>
<thead>
<tr>
<th>Attributes/Parameter</th>
<th>MDS/Database</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>class</strong></td>
<td>Both</td>
<td>The <code>class</code> attribute specifies the type of repository used to store Web Clipping definitions. The possible values for this attribute are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>oracle.portal.wcs.provider.info.MdsInformation</code> This value signifies that MDS is used to store Web Clipping definitions and that MDS configuration is pushed to the <code>mds-config.xml</code> file.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>oracle.portal.wcs.provider.info.DatabaseInformation</code> This value signifies that an Oracle9i Database or later is used to store Web Clipping definitions and that the database connection details are included as children in the <code>repositoryInfo</code> tag.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>oracle.portal.wcs.provider.info.JdbcDbInformation</code> This value signifies that Oracle database installed as part of Oracle WebLogic Server is used to store Web Clipping definitions.</td>
</tr>
<tr>
<td><strong>mdsConfigLocation</strong></td>
<td>MDS</td>
<td>Use the <code>mdsConfigLocation</code> tag when the value for the <code>class</code> attribute indicates an MDS repository. It points to the MDS configuration file <code>mds-config.xml</code>, which specifies the actual MDS location. The <code>mds-config.xml</code> file is located at: <code>MW_HOME/wls/wls_10.3/wc_domain/servers/WLS_Portlet/tmp/_WL_user/portalTools_11.1.1.2.0/yyggl7/war/WEB-INF</code>.</td>
</tr>
<tr>
<td><strong>connectionName</strong></td>
<td>Database</td>
<td>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 <code>jdbc/portletPrefs</code>, which points to the PORTLET schema in Oracle WebLogic Server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>In an interoperability scenario where an Oracle WebLogic Server 11g Middle Tier is paired with an Oracle Application Server 10g repository, the connection points to the PORTAL schema in the Oracle Application Server 10g repository.</td>
</tr>
<tr>
<td><strong>useASO</strong></td>
<td>Database</td>
<td>Set to true or false:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Specify false to omit this option.</td>
</tr>
<tr>
<td><strong>useRAA</strong></td>
<td>Database</td>
<td>Set to true or false:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 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 <code>repositoryInfo</code> child tags.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Specify false to omit this option.</td>
</tr>
<tr>
<td><strong>databaseHost</strong></td>
<td>Database</td>
<td>Specify the host name of the Oracle database. Use only version 9i or later. For example: <code>mycompany.dbhost.com</code>.</td>
</tr>
<tr>
<td><strong>databasePort</strong></td>
<td>Database</td>
<td>Specify the port number of the Oracle database listener. This is usually 1521.</td>
</tr>
</tbody>
</table>
By default, Web Clipping uses MDS, which is file-based, to store Web Clipping definitions and associated metadata. But you can configure Web Clipping to use a database. To migrate this repository for Portal 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 Portal Framework applications that use the Web Clipping producer.

2. Update the producer to use a different repository. For information, see **Section 65.5.1.3, "Configuring Web Clipping Repository in provider.xml."**

3. Run the Predeployment tool in **import** mode on all Portal Framework applications that use the Web Clipping producer.

### 65.5.2 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.

Portal 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 65–5** shows the relevant portion of `provider.xml`.

---

**Example 65–5  Proxy Settings**

```xml
- <!--
  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 65–2, describes the proxy settings you specify in the `provider.xml` file.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>httpProxyHost</td>
<td>Enter the host name of a proxy server if one is required to make a URL connection from the Web Clipping producer to its data sources.</td>
</tr>
<tr>
<td>httpProxyPort</td>
<td>Enter the port number for the HTTP Proxy Host.</td>
</tr>
<tr>
<td>dontProxyFor</td>
<td>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</td>
</tr>
<tr>
<td></td>
<td>the names of a business, or organization, or government agency, for example: <code>*.company.com</code>, <code>*.us.company.com</code></td>
</tr>
<tr>
<td></td>
<td>Hosts can be fully qualified host names or can be IP addresses.</td>
</tr>
<tr>
<td>proxyUseAuth</td>
<td>Acceptable values: true</td>
</tr>
<tr>
<td></td>
<td>Enter true if your proxy server requires authentication. The authentication parameters are specified by the following tags: <code>proxyType</code>, <code>proxyRealm</code>, <code>proxyUseGlobal</code>, <code>proxyUserName</code>, and <code>proxyPassword</code>.</td>
</tr>
<tr>
<td>proxyType</td>
<td>Acceptable values: Basic</td>
</tr>
<tr>
<td></td>
<td>Choose the type of proxy server this provider.</td>
</tr>
<tr>
<td></td>
<td>For more information about basic or digest authentication, see <a href="http://www.faqs.org/rfcs/rfc2617.html">http://www.faqs.org/rfcs/rfc2617.html</a>.</td>
</tr>
<tr>
<td>proxyRealm</td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td>If you do not know the name of the realm, then contact the administrator of the proxy server.</td>
</tr>
<tr>
<td>proxyUseGlobal</td>
<td>Acceptable values: true</td>
</tr>
<tr>
<td></td>
<td>If true, then the <code>&lt;proxyUser&gt;</code> and <code>&lt;proxyPassword&gt;</code> values are used for all users. Users do not see the Proxy Authentication section on the</td>
</tr>
<tr>
<td></td>
<td>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</td>
</tr>
<tr>
<td></td>
<td>the portlet.</td>
</tr>
<tr>
<td></td>
<td>The end user must log in using the Proxy Authentication section on the Personalize screen. If <code>&lt;proxyUsername&gt;</code> and <code>&lt;proxyPassword&gt;</code> are given,</td>
</tr>
<tr>
<td></td>
<td>then they are only used for public users.</td>
</tr>
<tr>
<td>proxyUserName</td>
<td>Enter the user name to log in to the proxy server.</td>
</tr>
<tr>
<td>ProxyPassword</td>
<td>Enter the password for the specified user name. You must prefix <code>!</code> before your plain password text. It is then encrypted in the <code>provider.xml</code></td>
</tr>
<tr>
<td></td>
<td>file for protection when the producer starts.</td>
</tr>
</tbody>
</table>
65.5.3 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.

This section includes the following topics:

- Section 65.5.3.1, "Adding Certificates for Trusted Sites"
- Section 65.5.3.2, "Configuring Oracle Advanced Security for the Web Clipping Producer"

65.5.3.1 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.
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.

### 65.5.3.2 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 65–3), 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.

---

### 65.6 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.

### 65.7 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 contains the following chapters:

- Chapter 66, "Personalizing Oracle WebCenter Portal Applications"
- Chapter 67, "Implementing Custom Data Providers: Introduction"
- Chapter 68, "Implementing Custom Function Providers: Introduction"
- Chapter 69, "Function Provider Reference"
- Chapter 70, "Using the Property Service"
- Chapter 71, "Property Service REST APIs"
- Chapter 72, "Conductor API Reference"
- Chapter 73, "Cache Management for Personalization"
This chapter describes how to integrate Personalization into a Portal Framework or WebCenter Portal application to deliver targeted content based on both user and application context. This chapter also describes the Personalization run-time system and associated tools that allow the declarative definition of application flow.

This chapter includes the following topics:

- Section 66.1, "Introduction to Personalization"
- Section 66.2, "Integrating Personalization in Your Application"
- Section 66.3, "Deploying Personalization Files"
- Section 66.4, "Tutorial: Creating, Testing and Deploying a Simple Application"
- Section 66.5, "Using Personalization in WebCenter Portal"

For more information about Personalization, also see:

- Appendix G.11, "ELs Related to Personalization"
- the "Managing Personalization" chapter in *Administering Oracle WebCenter Portal*

### 66.1 Introduction to Personalization

Personalization provides a dynamically derived user experience for your Portal Framework 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 66.1.1, "Personalization Architecture"
- Section 66.1.2, "The Conductor"
- Section 66.1.3, "Using Data Providers"

#### 66.1.1 Personalization Architecture

Figure 66–1 shows the architecture and out-of-the-box services and components that comprise Personalization. These and other Personalization components are described in:
66.1.2 The Conductor

This section describes the role of the 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 66.1.2.1, "Scenarios"
- Section 66.1.2.2, "Personalized Application Flow"
- Section 66.1.2.3, "Property Set Integration"
- Section 66.1.2.4, "Provider Integration"

66.1.2.1 Scenarios

A scenario is defined by a script you create using the Scenario Editor in JDeveloper (or provide as an XML file). A scenario is based on a simple syntax that allows conditional statement evaluation, resulting in some form of content being returned to the client. Content can be filtered or manipulated prior to it being returned to the client.

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 66.2.2.3, "Creating a New Scenario in the Scenario Editor" and Section 66.2.2.4, "Specifying Scenario Flow Using Node Types." See also Section 66.2.3, "Creating File-Based Scenarios."
66.1.2.2 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 72, "Conductor API Reference" for more information about using Java and EL APIs.

66.1.2.3 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 66.2.2.2, "Creating Property Sets and Property Definitions."

66.1.2.4 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 66.1.3, "Using Data Providers."

66.1.3 Using Data Providers
A data 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, a CMIS provider enabling interaction with an Oracle WebCenter 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 66.1.3.1.1, "Out-of-the-Box Providers"
- Section 66.1.3.2, "Writing Custom Data Providers"
- Section 66.1.3.3, "Writing Custom Function Providers"

66.1.3.1 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 66.1.3.1.1, "Property Service Provider"
- Section 66.1.3.1.2, "CMIS Provider"
- Section 66.1.3.1.3, "Activity Graph Provider"
- Section 66.1.3.1.4, "People Connections Locator"
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 66.2.6.1, "Using the Property Service Provider."

66.1.3.1.2 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 66.2.6.2, "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 Administering Oracle WebCenter Portal.

66.1.3.1.3 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."

The Activity Graph provider is built on two REST services for Activity Graph running on the WebCenter Portal server (WC_Spaces), and consequently requires that WebCenter Portal be installed in your environment. For more information about integrating the Activity Graph provider in your scenario, see Section 66.2.6.3, "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 Administering Oracle WebCenter Portal.

66.1.3.1.4 People Connections Locator

The People Connections locator is an IPropertyLocator used by the Property Provider to interface with People Connections. One of the benefits of People Connections is that it can access user profile information stored on WebCenter Portal. For more information about People Connections, see Section VI, "Working with People Connections." For more information about the People Connections locator, see Section 66.2.6.1, "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 Administering Oracle WebCenter Portal.

66.1.3.2 Writing 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.

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 Chapter 67, "Implementing Custom Data Providers: Introduction."

66.1.3.3 Writing Custom Function Providers

Function providers 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')}. For information on writing your own function providers, see Chapter 68, "Implementing Custom Function Providers: Introduction."

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.11, "ELs Related to Personalization."

66.2 Integrating Personalization in Your Application

This section describes how to integrate Personalization in your Portal Framework application.

This section contains the following subsections:

- Section 66.2.1, "Personalization Requirements"
- Section 66.2.2, "Authoring Personalized Scenarios in JDeveloper"
- Section 66.2.3, "Creating File-Based Scenarios"
- Section 66.2.4, "Displaying Targeted Content at Runtime"

66.2.1 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 and Limitations" in the Administering Oracle WebCenter Portal.

- WebCenter Portal 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 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 Administering 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/scripts-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 Administering 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 Administering Oracle WebCenter Portal.
66.2.2 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 66.2.2.1, "Creating a Properties Namespace File"
- Section 66.2.2.2, "Creating Property Sets and Property Definitions"
- Section 66.2.2.3, "Creating a New Scenario in the Scenario Editor"
- Section 66.2.2.4, "Specifying Scenario Flow Using Node Types"
- Section 66.2.2.5, "Defining Property Sets and Property Locators"
- Section 66.2.2.6, "Using the Out-of-the-Box Providers"

66.2.2.1 Creating a Properties Namespace File

The first step in setting up a scenario is to create a Properties Namespace file. This file will store the settings for the namespace(s) in which your scenario will run, and typically a set of default property definitions. This file will also contain the property definitions and property set definitions you define later on in Section 66.2.2.2, "Creating Property Sets and Property Definitions."

To create a Properties Namespace file in JDeveloper:

1. With the application open in the Application Navigator, select New from the File menu.

   The New Gallery dialog displays (see Figure 66–2).
2. Choose **Properties Namespace** and click **OK**. The Create Namespace dialog displays (see Figure 66–3).

![New Gallery Dialog](image)

**Note:** If the Personalization category doesn’t display, open the All Technologies tab and select it, or click **selected technologies** and add WebCenter Personalization to the project.

3. Enter a name for the namespace, select the **Include default property definitions** check box to add initial default property definitions to the namespace, and click **OK**.

   The **Include default property definitions** check box 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 afterwards if they’re not required. Or, you can leave the check box unselected to start with no initial property definitions and create them as needed in the Properties Editor.
A graphic representation of the namespace file displays in the Design pane (see Figure 66–4).

**Figure 66–4  Design Pane - Namespace Definition**

4. Continue by defining the property sets and properties that you will need in your scenario as described in Section 66.2.2.2, "Creating Property Sets and Property Definitions."

66.2.2.2 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 66–5).

**Figure 66–5  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 66–6).

**Figure 66–6  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 Mapping dialog displays (see Figure 66–7).
4. Enter a **Property Name** and select a **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 66–8).

5. Add any additional properties required to the property set definition (or create new property sets) and then continue by creating the scenario as described in Section 66.2.2.3, "Creating a New Scenario in the Scenario Editor."

### 66.2.2.3 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 and select **New**.

   The New Gallery dialog displays (see Figure 66–9).
2. Select **Scenario** and click **OK**.

   The New Scenario Wizard dialog displays (see Figure 66–10).

   **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.

3. Select **Create a new 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 66–11).
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 choose the node type from the list as shown in Figure 66–12.

**Figure 66–12  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 66.2.2.4, "Specifying Scenario Flow Using Node Types."**

### 66.2.2.4 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.
- **Set Variable**: The Set Variable node lets you define a variable that is scoped within the context of the currently running scenario and initialize it with an expression.
- **Script**: Use the Script node to add a script, including ones that use embedded EL expressions, to your scenario. A script engine can be specified by either the mime type or engine name.
- **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.
- **Error Handler**: The Error Handler node provides a Java-like Catch/Try construct where you can use the Try node to test a condition and execute a response depending on the results.
66.2.2.5 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 66.1.3.1.4, “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 Chapter 70.7, “Implementing a Custom Locator.” Note that although multiple locators can be associated with a property set definition, only a single property in that property set definition can be assigned to one of the locators.

66.2.2.6 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 66.2.2.6.1, "Using the Property Service Provider"
- Section 66.2.2.6.2, "Using the CMIS Provider"
- Section 66.2.2.6.3, "Using the Activity Graph Provider"

66.2.2.6.1 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 66.1.3.1.4, “People Connections Locator.”
information about configuring the Property Service provider, see "Configuring the Oracle People Connections Locator" in the Administering Oracle WebCenter Portal.

66.2.6.2 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 Administering 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:

```xml
<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>
        <return>
          <expression>${exprString}</expression>
        </return>
      </call-provider>
      <name>CMIS Scenario</name>
      <comments>CMIS Scenario comments</comments>
      <tags>
        <tag>CMIS Scenario tag</tag>
      </tags>
    </body>
  </scenario>
</scenarioMetadata>
```

Executing the Scenario

Request URI:

POST http://localhost:8891/wcps/api/conductor/namespaces/myNameSpace/scenarios/CMIS%20Scenario

Request Body:
Integrating Personalization in Your Application

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:

```java
/**
 * 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="getCMISQueryForDocIDs.parameter.description.inputList") String repository,
                                   @PublicParameter(parameterName="ids",descriptionBundleKey="getCMISQueryForDocIDs.parameter.description.inputList") List<String> ids))
```

```java
/**
 * 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-ucml1g#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.)

```xml
<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>testFunctionProviderScenario0 comments</comments>
  <body>
    <assign-variable>
      <variable>ids</variable>
      <expression>${collections:new()}</expression>
    </assign-variable>
    <execute>
      <expression>${collections:append(ids,'StellentRepository#dDocName:MOUNTAINS')}</expression>
    </execute>
    <execute>
      <expression>${collections:append(ids,'StellentRepository#dDocName:PARKS')}</expression>
    </execute>
  </body>
</scenario:scenario>
```
Integrating Personalization in Your Application

Personalizing Oracle WebCenter Portal Applications

```xml
<execute>
<expression>${collections:append(ids,'StellentRepository#dDocName:PATAGONIAPARKS')}
</execute>

<!-- Assign those as IDs to send to CMIS -->
<assign-variable>
<variable>cmisQuery</variable>
</assign-variable>

<expression>${cmisfunction:getCMISQueryForDocIDsFromFullID(ids)}</expression>

<!-- 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>
</scenario:scenario>

66.2.2.6.3 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 Administering 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:

```xml
<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>
```
Integrating Personalization in Your Application

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/QueryCommonItemsScenario

Request Body:
<parameters/>

Using the AGFunctionProvider
A function provider is a utility class that can be invoked from within a scenario. An example using the AGFunctionProvider is shown below:

```java
/**
   * ...
*/
```
* 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.parameter.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.
*/
Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper

```
*/
@PublicFunction
{
    functionName='getContentIDs',
    descriptionBundleKey='getContentIDs.method.description'
}

g 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>
<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>

### 66.2.3 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 **Import an existing scenario file** option when you create the scenario.

Table 66–1 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 66.2.2.4, "Specifying Scenario Flow Using Node Types."

<table>
<thead>
<tr>
<th>Statement</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
</table>
| Set Variable   | 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 66–1  (Cont.) Scenario Statements

<table>
<thead>
<tr>
<th>Statement</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
</table>
| Conditional | 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>                                                                                                                             |
| Error Handler| Allows you to add a try/catch block to a scenario.                         | <body>
|             |                                                                            |   <error-handler id="_1">                                                                                                                |
|             |                                                                            |     <try id="_2">                                                                                                                        |
|             |                                                                            |       <body/>                                                                                                                             |
|             |                                                                            |       </try>                                                                                                                              |
|             |                                                                            |     <catch id="_3">                                                                                                                      |
|             |                                                                            |       <body/>                                                                                                                             |
|             |                                                                            |         <variable>exception</variable>                                                                                                     |
|             |                                                                            |           </catch>                                                                                                                         |
|             |                                                                            |                 </error-handler>                                                                                                           |
|             |                                                                            |                   </body>                                                                                                                 |
| For Each    | 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 66–1 (Cont.) Scenario Statements

<table>
<thead>
<tr>
<th>Statement</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
</table>
| **Script**      | Allows you to run a script in the scenario. The script can be written in JavaScript, Groovy, JRuby, or Python. | `<body>`<script type='text/javascript' resolveAsExpression='true' id='_5'>
  <variable>hello</variable>
  <source>y=5;
z=2;
x=y+z;
hello=x;</source>
</script><return id='_6'>
  <expression>${hello}</expression>
</return></body>` |
| **While**       | Provides looping/iterating functionality as long as the specified EL expression evaluates to true. | `<while>`<body><assign-variable><comments>Sum up customer invoices to arrive at customer total expenditure.</comments>
  <variable>customerTotalSpent</variable>
  <expression>${customerTotalSpent+order.invoic eTotal}</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>
</call-scenario>`<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>` |
Example 66–1  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>
          <assign-variable>
            <variable>ResultVar</variable>
            <expression>conditionSatisfied</expression>
          </assign-variable>
        </condition>
        <otherwise>
          <assign-variable>
            <variable>ResultVar</variable>
            <expression>otherwiseInvoked</expression>
          </assign-variable>
        </otherwise>
      </body>
      <otherwise>
        <assign-variable>
          <variable>ResultVar</variable>
          <expression>otherwiseInvoked</expression>
        </assign-variable>
      </otherwise>
    </conditions>
    <return>
      <expression>${ResultVar}</expression>
    </return>
  </body>
</scenario:scenario>
66.2.4 Displaying Targeted Content at Runtime

Targeted content can be displayed to the user at runtime using 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 27, "Creating Content Presenter Display Templates."

66.3 Deploying Personalization Files

This section explains how to build and deploy Oracle WebCenter Portal Personalization (WCPS) files.

---

**Note:** For information on integrating and configuring Personalization, see "Personalization Prerequisites and Limitations" in the *Administering Oracle WebCenter Portal*.

---

- Section 66.3.1, "Deploying Personalization Files to a Server"
- Section 66.3.2, "Deploying Personalization Files to an Archive File"
- Section 66.3.3, "Importing and Exporting MAR Files"
- Section 66.3.4, "Propagating Personalization Files"

66.3.1 Deploying Personalization Files to a Server

To deploy Personalization files to a server:

---

**Note:** The deployment feature described in this section requires the WebCenter Personalization technology scope. If this scope is not added, the related menu items will be disabled. See Section 66.2, "Integrating Personalization in Your Application."

---

1. In JDeveloper, open your portal project. If you want to use WLST commands to perform this deployment, see Section 66.3.3, "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 66–13.
Deploying Personalization Files

Figure 66–13  The Deploy To Server Dialog

4. Click the ellipsis ("...") 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.

66.3.2 Deploying Personalization Files to an Archive File

You can deploy your Personalization files to a metadata archive (MAR) file on the filesystem. 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 66.3.3, "Importing and Exporting MAR Files."

66.3.3 Importing and Exporting MAR Files

To update a Personalization 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 Personalization files 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')
```
wlst:> exportMetadata(application='wcps-services', server='WC_Utilities', toLocation='<mar file>', remote='true')

For more information on MDS WLST commands, refer to the 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.

### 66.3.4 Propagating Personalization Files

The WebCenter Portal Administration Console includes a propagation tool for moving portal metadata from a staging to a production server without incurring any downtime. The propagation tool supports moving Oracle WebCenter Portal Personalization 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. For more information about the propagation tool, see Section 8.13, "Using the Propagation Tool to Propagate From Staging to Production."

The ability for the propagation tool to move Personalization files depends on the following conditions:

- The Portal Framework application’s MDS repository must point to the same database as the mds-wcpsDS data source. This is the default configuration. 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 Personalization REST services are notified of the MDS changes. This delay does not occur when you use the WLST commands to propagate Personalization files, as explained in Section 66.3.3, "Importing and Exporting MAR Files."

---

### 66.4 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 running 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 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_Application) using the default options and technologies.
   a. From the File menu select New...
      The New Gallery appears.
   b. Select the Applications category.
   d. Enter the Application Name as Hello_Application, as shown in Figure 66–14.

\[Figure 66–14  Creating the Hello_Application - Step 1\]

   e. In the Project 1 Name page of the wizard (Step 2), select WebCenter Personalization in the Available list and move it to the Selected list.
   f. Click Next until you reach Step 5 and click Finish.

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 Applications 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...
c. When prompted, enter welcome1 as the Administrator password and click OK. 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."

d. Start the server by right-clicking IntegratedWeblogicServer to display the context menu, and selecting Start Server Instance. See Figure 66–16.

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.

3. 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.

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 66–17).
c. Enter the (default) data entry fields as shown below to create the connection.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Conductor</td>
</tr>
<tr>
<td>URL Endpoint</td>
<td><a href="http://localhost:7101/wcps/api/conductor/resourceIndex">http://localhost:7101/wcps/api/conductor/resourceIndex</a></td>
</tr>
<tr>
<td>Authentication Type</td>
<td>Basic</td>
</tr>
<tr>
<td>Username</td>
<td>weblogic</td>
</tr>
<tr>
<td>Password</td>
<td>welcome1 (the Administrator password you provided when you created the domain)</td>
</tr>
<tr>
<td>Realm</td>
<td>WCPS</td>
</tr>
</tbody>
</table>

d. Click **Test Connection**. The Status should display as "The connection successfully established."

e. Click **OK** to create the Conductor URL connection.

f. On the Personalization Server page, click **OK**.

4. Create a simple hello_world scenario that returns the string "Hello World" when invoked by the Personalization server.

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 **Scenario** and click **OK**. The New Scenario dialog displays.
d. Leave the option Create a new 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 66–19).

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 66–20.

Figure 66–20  Scenario Editor - hello_world Scenario

![Figure 66–20](image)

h. From the File menu, select Save to save the hello_world.scenarios_diagram.

5. Try running the Scenario directly. Right-click the Start node and select Run.

6. In the Scenario Input dialog, click OK. This dialog allows you to enter any required parameters for the scenario. In this case, there are none.

7. The result "Hello World" appears in a log window.

8. Next, create a JSP page that invokes the hello_world scenario and displays the results.
   a. Right-click the Portal project in the Application Navigator and select New...
      The New Gallery displays.
   b. From the Current Project Technologies tab, select the JSP category.
   c. From the item list select JSP and click OK.
      The Create JSP dialog displays (see Figure 66–21).

Figure 66–21  Create JSP Dialog

![Create JSP Dialog](image)

d. Enter the File Name as hello_world.jsp and click OK to create the JSP page.

e. 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.

```html
<!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("p13nContext")).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/conductor/resourceIndex
```

in place of the Conductor URL to get around the single sign-on constraint. For more information about configuring the WebCenter Trust Services and single sign-on using this script see "Configuring Single Sign-On" in the *Administering Oracle WebCenter Portal*.

9. 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 integrated 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`. 

f. From the File menu, select Save to save the `hello_world.jsp`.
Using Personalization in WebCenter Portal

Oracle WebCenter Portal Personalization provides a dynamically derived user experience that can be applied to any WebCenter Portal object that can be used in conjunction with Expression Language (EL) expressions. This section provides an overview of how the Personalization engine can be used to determine which WebCenter Portal objects are presented to the user, and how these objects are presented based on the outcome of a Personalization scenario.

This chapter includes the following sections:
66.5.1 Introduction to Personalization in WebCenter Portal

WebCenter Portal's Personalization provides a mechanism with which to vary the user experience based on dynamically evaluated criteria defined in a Personalization scenario. The scenario evaluates defined sources of input data, and generates a decision. The results of that decision or outcome can be applied in WebCenter Portal using Expression Language (EL) expressions.

To use Personalization in WebCenter Portal, you first need to create a scenario in a Framework application. The scenario does all the work of supplying the logic and calling one or more data providers to retrieve data. You then use ELs to execute and return the results of the scenario, and modify the user experience accordingly in WebCenter Portal. For information about how to create Personalization scenarios, see Section 66.2.2, "Authoring Personalized Scenarios in JDeveloper." The Personalization ELs used to run a scenario are cataloged in Section G.11, "ELs Related to Personalization."

Once you’ve set up your scenario, you can use ELs in WebCenter Portal to present targeted content or shape the way content is presented based on information about the user. For example, you can use a Personalization scenario to determine a user's role within an enterprise, and based on that selectively show content for that role. You can also tailor the presentation (such as page templates, page styles, skins, resource catalogs, and so on) for the user.

66.5.2 Calling Personalization ELs in WebCenter Portal

Personalization can deliver targeted content or change application flow based on both user and application context. Personalization, for example, can return content based on information about a user in a Human Resources database, or targeting the WebCenter Portal experience for that specific user by selecting the page templates, page styles, skins, and navigation flow within which the content is presented.

You can define the content sources, or data providers, that are used to evaluate or provide data for a scenario. The default data provider for Personalization is the CMIS data provider, which can retrieve data from WebCenter Portal's default content repository (Oracle WebCenter Content Server), and a data provider for Activity Graph and a locator for People Connections are also available out of the box. You can also define your own data providers for other data sources. For more information about data providers, see Section 66.1.3, "Using Data Providers."

In general, you call a scenario from a portal using the following EL expression:

```
#{p13nContext.conductor['CONDUCTORCONNECTION'].namespaces['NAMESPACE'].scenario['SCENARIO'].results}
```

Where:

- **CONDUCTORCONNECTION** - is the name of the ADF Connection configured against the WebCenter application
- **NAMESPACE** - is the name of the namespace within which the scenario will run
- **SCENARIO** - is the name of the scenario to be invoked

For example, you can create a simple example of using Personalization to drive content based on information about the user by creating a scenario that returns "true" if the employee hire date is within the last two months (see Section 66.2.2, "Authoring
Personalized Scenarios in JDeveloper” to learn how to create a Personalization scenario. After creating the scenario:

1. Log in as an administrator in WebCenter Portal.
2. Create a page for new hires, naming it New Hire.
3. Edit the navigation model for the site, and add a new navigation link to the New Hire page. For more information about editing the navigation model in a portal, see Section 10.3, “Editing a Navigation Model.”
4. Select New Hire and click the Edit (Pencil) icon.

Figure 66–24 Navigation Model

5. For the Visible attribute, open the Expression Editor, select Type a value or expression and enter:

   #{p13nContext.conductor.default.namespaces['p13nApp'].scenario['NewHire'].results}

   Where p13nApp is the name of the Framework application containing the scenario, and NewHire is the name of the scenario. This will execute the scenario and return the results (True or False in this case). If the user is a new hire (True) based on the results of the scenario, then the menu item will display when the user logs in.

Figure 66–25 Expression Editor
This chapter introduces data providers, discusses their important role in the Personalization architecture, and presents a basic example that you can build and run in JDeveloper. After reading this chapter, you will have a basic understanding of how to implement a custom data provider.

This chapter includes the following topics:

- Section 67.1, "Introduction to Data Providers"
- Section 67.2, "Do I Need to Write a Custom Data Provider?"
- Section 67.3, "Discovering Data Providers"
- Section 67.4, "Implementing a Simple Custom Data Provider"
- Section 67.5, "Deploying the Custom Data Provider"
- Section 67.6, "Invoking the Data Provider in a Scenario from JDeveloper"
- Section 67.7, "Designing Custom Data Providers: Best Practices"

67.1 Introduction to Data Providers

Data providers are connecting points between services provided by external systems, like business management or inventory systems, and the Conductor. The Conductor is a container for running and managing scripted components called scenarios. Scenarios consume and process the data that one or more data providers retrieve, and thereby make that data available for use in a client application, like a portal or mobile application. See Figure 67–1.

Note: Another type of provider used with Personalization is a function provider. While data providers fetch data from external sources and return the data to a scenario, function providers filter and transform data within a scenario. See Chapter 68, "Implementing Custom Function Providers: Introduction."
Data providers are essential components within the Personalization architecture. Data providers offer the flexibility to connect to and retrieve data from almost any kind of service. WebCenter Portal includes several out-of-the-box providers. You can begin using these out-of-the-box providers immediately, or you can write your own custom data providers to satisfy specific use cases.

For example, you might want to write an inventory provider to retrieve parts and inventory information, a purchasing provider to retrieve purchasing information from a business management system, a sales provider to retrieve sales information from a CRM system, and so on.

### 67.2 Do I Need to Write a Custom Data Provider?

The first thing you need to do is determine if one of the out-of-the-box data providers is sufficient for your Personalization use case. The out-of-the-box data providers include:

- **Property Service Provider** – Allows you to integrate property service data into a scenario. See Section 70.5, "Using the Property Service Data Provider."

- **CMIS Data 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.

- **Activity Graph Data Provider** – Provides integration with the Activity Graph engine (the basis for Activity Graph), which provides recommendations about what a user may be interested in connecting with based on analytics within WebCenter.

These data providers might not be suitable for all use cases. For example, you may need to retrieve data from a legacy application with a non-standard set of APIs. In these cases, you need to implement a custom data provider. See Section 67.4, "Implementing a Simple Custom Data Provider."

### 67.3 Discovering Data Providers

Data providers (both out-of-the-box and custom providers) are listed in the Invoke Provider Properties dialog as shown in Figure 67–2.

To access this dialog, create a scenario and add an Invoke Provider node to it. Right-click the node and select **Invoke Provider Properties**. For more information on
Implementing a Simple Custom Data Provider

This section describes how to implement a simple custom data provider, configure it, and invoke it inside a scenario. Even though this is a very basic example, it illustrates many important aspects of custom provider development and configuration.

- Section 67.4.1, “Overview of the Example and Prerequisites”
- Section 67.4.2, "Data Provider Architecture"
- Section 67.4.3, "Extending the DefaultDataProvider Class"  
- Section 67.4.4, "Creating a Resource Properties File"
- Section 67.4.5, "Extending BaseConnectionConfig and DefaultProviderConnection"
- Section 67.4.6, "Extending DefaultExecutableResource"

67.4.1 Overview of the Example and Prerequisites

This example demonstrates how to implement a data provider that accepts one parameter and returns a message to the server console when it is called in a scenario. In the following sections, we discuss the required provider code and explain how to integrate the provider into the Conductor, use it in a scenario, and run the scenario.

The prerequisites for developing custom Personalization components, including data providers, are described in Section 2.5, "Setting Up JDeveloper for Personalization."
The prerequisites include making certain required components available to your project.

**Note:** This example is presented in the context of a Portal Framework application in JDeveloper. For custom data provider development, JDeveloper is not required. You can use any Java IDE, however, you need to include the required WebCenter Personalization JAR files and Java components to your project. See Section 2.5, "Setting Up JDeveloper for Personalization," for more information.

### 67.4.2 Data Provider Architecture

To create a custom data provider, you must extend a set of Java base classes to provide the appropriate functionality. Table 67–1 gives a quick summary of the data provider classes used in this example section. If you intend to write a custom provider, you’ll become familiar with these classes. If you’re using an out-of-the-box provider, it’s helpful to understand this basic architecture.

<table>
<thead>
<tr>
<th>Interface</th>
<th>Primary Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>DefaultDataProvider</td>
<td>Instantiates a connection configuration object used by the provider. Each data provider has one connection (1-1 relationship).</td>
</tr>
<tr>
<td>BaseConnectionConfig</td>
<td>Specifies a name for the connection. A connection name is required and must be added to a configuration file called wcps-connections.xml, as explained later in this chapter.</td>
</tr>
<tr>
<td>DefaultProviderConnection</td>
<td>Instantiates one or more ExecutableResource objects. A data provider can reference multiple executable resource objects (1-many relationship).</td>
</tr>
<tr>
<td>DefaultExecutableResource</td>
<td>ExecutableResource methods do whatever work is required for your data provider to perform. Typically, this work involves interacting with external services. For example, if your provider is written to interact with Mail, you might retrieve certain Mail connection information and then implement getMail() and sendMail() methods. These methods can then be invoked within the context of a scenario. These concepts will become more clear when you review the simple data provider example presented in this chapter.</td>
</tr>
</tbody>
</table>

### 67.4.3 Extending the DefaultDataProvider Class

The first step in creating our example custom data provider is to extend DefaultDataProvider. This approach is best used for a quick implementation of a custom provider that does not rely on external resources, like databases or applications. For cases where you do need external resources, extend the ExternalDataProvider class instead.

**Example 67–1** lists the code for the extended DefaultDataProvider class. In JDeveloper, refer to online help for detailed information on creating and compiling Java classes.

**Tip:** The general instructions for creating a class are to select New from the File menu and choose to create a new Java class. Copy the example code into the file and save it. If you explicitly enter the package providers.data in the Create Java Class dialog, the package directory will be created for you and your class file will be placed in that package directory. Follow the same approach to create all of the classes presented in this example.

**Example 67–1 Extending DefaultDataProvider**

```java
package providers.data;
```
import oracle.wcps.conductor.annotation.ContextualProvider;
import oracle.wcps.dataprovider.base.DefaultDataProvider;
import oracle.wcps.conductor.provider.IConnection;
import oracle.wcps.dataprovider.base.IConnectionConfig;

@ContextualProvider
{
    contextName="SayHelloProvider",
    resourceBundle="resources.SayHelloProperties",
    descriptionBundleKey="hello.provider.description",
    nameBundleKey="hello.provider.name"
}

public class SayHelloProvider extends DefaultDataProvider {
    
    public Class<?> getConnectionConfigClass()
    {
        return SayHelloConnectionConfig.class;
    }

    @Override
    protected IConnection createConnection(IConnectionConfig config)
    {
        return new SayHelloConnection((SayHelloConnectionConfig)config);
    }
}

The first thing to notice about SayHelloProvider is that it is annotated. Every data provider class requires the following @ContextualProvider annotation stanza immediately before the class definition:

@ContextualProvider
{
    contextName="SayHelloProvider",
    resourceBundle="resources.SayHelloProperties",
    descriptionBundleKey="hello.provider.description",
    nameBundleKey="hello.provider.name"
}

The contextName element specifies the name that will appear in the scenario editor dialog that lets you select the data provider. The resourceBundle element refers to a resource file that exists in the class path of the data provider. You can have multiple resource files in a project. Just remember that the resourceBundle value must correspond to one of them. The other annotations are resource bundle keys used to identify strings in the resource file, which primarily support localization. We’ll discuss creating the resource file in the next section, Section 67.4.4, “Creating a Resource Properties File.”

Note that to use these class-level annotations, you must import the following class:

import oracle.wcps.conductor.annotation.ContextualProvider;

Figure 67–3 shows the new SayHelloProvider.java source file in the Application Navigator in JDeveloper. Note that the source file is in a package called providers.data. You are responsible for creating the package, according to your own conventions.
67.4.4 Creating a Resource Properties File

The class-level Java annotations placed in the data provider source file specify a resource properties file. This file must exist and must include the keys used in the class annotations and provide their values.

**Note:** The resource properties file’s package must be located within the same parent directory as your source code or source code package. You can create multiple resource files, as long as they are referenced properly in the provider class.

For example, the following resource keys were listed in the SayHelloProvider class annotations (see the listing in Example 67–1). The first annotation, `hello.provider.name`, specifies the name of the provider as it will appear in the scenario editor's Invoke Provider dialog. The second annotation provides a brief description and also appears in the Invoke Provider dialog when you hover the mouse over a provider's name.

- `hello.provider.name=SayHelloProvider`
- `hello.provider.description=Simple data provider that greets a user by name`.

**Figure 67–4** shows a properties file in the Application Navigator called `SayHelloProperties.properties` and it is in a package called `resources`. Recall from Example 67–1 that the class annotations specified the resource file to be `resources.DataProviderResources`, which is what is reflected in Figure 67–4.
67.4.5 Extending BaseConnectionConfig and DefaultProviderConnection

If you created the SayHelloProvider class as presented previously, you noticed that two classes are unresolved. We'll create those classes next. The first class, SayHelloConnectionConfig, extends the base class BaseConnectionConfig. The second, SayHelloConnection extends DefaultProviderConnection.

Example 67–2 lists SayHelloConnectionConfig. It's empty because for this example, there are no connections to configure. In this example, you could just use BaseConnectionConfig directly, rather than extending it. If you are writing a provider that needs to connect to external resources, then this class is more important.

**Example 67–2 Extending BaseConnectionConfig**

```java
package providers.data;

import oracle.wcps.dataprovider.base.BaseConnectionConfig;
import oracle.wcps.connection.annotation.ConnectionConfiguration;

@ConnectionConfiguration
{
    connectionType="client.simple.connection"
}

public class SayHelloConnectionConfig extends BaseConnectionConfig {
    public SayHelloConnectionConfig() {
        super();
    }
}
```

The class-level annotation, @ConnectionConfiguration, is important. It specifies the connection type for the provider. The connection type specified in this annotation must match the <connection-type> element in the wcps-connections.xml configuration file, which we'll discuss shortly in Section 67.5.6, "Updating the WCPS Connections Configuration File."

The next class, SayHelloConnection, extends DefaultProviderConnection. This class is primarily responsible for instantiating ExecutableResource objects, which do the work
of the data provider, like retrieving data by calling external APIs. A provider can have multiple ExecutableResource objects; however, for this example, we only need one.

**Example 67–3  Extending DefaultProviderConnection**

```java
package providers.data;

import oracle.wcps.conductor.provider.IExecutableResource;
import oracle.wcps.dataprovider.base.DefaultProviderConnection;

class SayHelloConnection extends DefaultProviderConnection {
    public SayHelloConnection(SayHelloConnectionConfig config) {
        super(config);
    }

    @Override
    protected void createExecutableResources() {
        IExecutableResource resource = new SayHelloExecutableResource();
        executableResources.put("SimpleDataProvider", resource);
    }
}
```

Note that the Map key “SimpleDataProvider” is an arbitrary name. It does not have to match the class name.

**67.4.6 Extending DefaultExecutableResource**

The last class you need to extend is DefaultExecutableResource. ExecutableResource objects are the data provider work horses. These classes let you implement the custom code that runs in a scenario. If you write a provider that fetches data from an external source, the code for performing that task goes in an ExecutableResource class.

---

**Note:** A Connection may have several implementations of an ExecutableResource. For example, the ActivityGraphProvider’s ActivityGraphConnection has three different ExecutableResources related to common items, common users, and recommendations.

---

The ExecutableResource class for this simple example doesn’t do very much: it just returns a message. Review the code in **Example 67–4** and we’ll discuss the details immediately after the example.

**Example 67–4  Extending DefaultExecutableResource**

```java
package providers.data;

import oracle.wcps.conductor.annotation.PublicFunction;
import oracle.wcps.conductor.annotation.PublicParameter;
import oracle.wcps.dataprovider.base.DefaultExecutableResource;

class SayHelloExecutableResource extends DefaultExecutableResource {

    @PublicFunction
    {
        functionName = 'sayHello',
        descriptionBundleKey = 'provider.method.sayHello.description'
    }
}
```
public String sayHello(@PublicParameter(parameterName = "user",
    descriptionBundleKey = "provider.parameter.sayHello.user.description")
        String user)
{
    String msg = 'Hello, ' + user + '
        . How are you today!";
    System.err.println(msg);
    return msg;
}

The most important thing to notice about this code is that the sayHello() method and its parameter are annotated. The annotation is required and allows the scenario editor to pick up and display metadata about the method. As with other annotations, these represent a name/value pair, where the value part is a resource bundle key. This key is then referenced in the same resource properties file that was specified in the SayHelloProvider class.

**Tip:** Use of the resource bundle to isolate the string values is a best practice and assists with localization. You must be sure to include the properties file in your final, deployed JAR file. We'll discuss creating the JAR file in Section 67.5, "Deploying the Custom Data Provider." See also Section 67.4.4, "Creating a Resource Properties File."

That's it for coding our simple SayHelloExecutableResource class. As you can see, this class allows you to implement custom methods that can be executed in a scenario. Don't forget that annotation is required for both methods and parameters.

Next, we need to deploy the data provider to the server, where it can be used in a scenario. See Section 67.5, "Deploying the Custom Data Provider."

### 67.5 Deploying the Custom Data Provider

After you create a new data provider class, you need to compile it and deploy it properly. After you accomplish these tasks successfully, the provider will show up in the scenario editor in the Invoke Data Provider Properties dialog, as explained in Section 67.6, "Invoking the Data Provider in a Scenario from JDeveloper."

---

**Note:** Data provider deployment can be tricky, because it's largely a manual procedure. You must follow these instructions carefully—even a simple typo at any point will prevent the provider from being deployed successfully.

To summarize the steps, you must deploy a JAR file containing:

- your compiled data provider classes
- a resource properties file (which must be in the same parent directory as the classes)
- a configuration file listing the classname(s) of your data provider(s)

### 67.5.1 Compile Your Data Provider Classes

Data providers are written in Java, and must be compiled before they are deployed.
1. In Application Navigator, right-click the project containing your source code and select either the Make or Rebuild options, as shown in Figure 67–5.

Figure 67–5  Compiling a Project

67.5.2 Create a Configuration File in META-INF/services

The next step in preparing your data provider for deployment is to create a simple configuration file that lists the full class name(s) of your provider(s).

1. In the Application Navigator, create a new folder called META-INF in the Application Sources folder.
2. In the META-INF folder, create a folder called services.
3. In the services folder, create a text file called oracle.wcps.conductor.provider.IDataProvider. You must use this exact name.
4. Open the oracle.wcps.conductor.provider.IDataProvider file and enter the full class name for each data provider you wish to deploy. If you have more than one provider, place each one on a separate line in the file.

   For example:
   
   providers.data.SayHelloProvider

67.5.3 Setting Up the JAR File Structure

If you are unfamiliar with JAR file creation in JDeveloper, this section explains how to set up your JAR file structure (a deployment profile) to ensure that all the required elements are included.

1. Right-click the project containing your data provider source code in the Application Navigator and select Project Properties.
2. In the Project Properties dialog, select Deployment.
3. Click New.
4. Select JAR File as the Archive Type and give the JAR a name.
5. Click OK.
6. In the JAR Options dialog, select **Contributors**.
7. Be sure that **Project Output Directory**, **Project Source Path**, and **Project Dependencies** are selected, as shown in Figure 67–6.

**Figure 67–6 Selecting Project Output Contributors**

8. Select **Filters**.
9. Select the **Patterns** tab.
10. In the Patterns tab, add the following file patterns to include and exclude specific types of files from the JAR.
- **Include**: **.class**
- **Include**: **.resources**
- **Include**: **META-INF**
- **Exclude**: **.java**
- **Exclude**: **.txt**
- **Exclude**: **.html**

**Figure 67–7** shows these patterns after they have been added to the list.
11. Click OK, and then OK in the Project Properties dialog.

12. Save your project.

### 67.5.4 Create a JAR File

After the JAR deployment profile is set, you are ready to generate a JAR file from your project.

1. Right-click the project (in this case, the Portal project) in the Application Navigator and select **Deploy**.

2. Select the JAR file you want to deploy. This is the name you gave the file in Section 67.5.3, "Setting Up the JAR File Structure."

3. In the Deployment Action dialog, select **Deploy to JAR** file and click **Finish**.

The JAR file is placed in your project directory. You can view it on the file system in: 
`<Workspace_Directory>/Portal/deploy`. For example, if your Workspace is called MyProviders, you might find the JAR here:
`C:\JDeveloper\mywork\MyProviders\Portal\deploy`

### 67.5.5 Copy the JAR File to the Server

You must copy the JAR file to the server.

1. Copy the JAR file that you created in the previous step. As noted previously, the file is located in `<Workspace_Directory>/Portal\deploy`.

2. Paste the JAR file the following location on the server:

   `DOMAIN_HOME/conductor-extensions-library/WEB-INF/lib`

   For example, for an Integrated WebLogic Server environment:

   `C:\JDeveloper\system11.1.7.40.63.82\DefaultDomain\conductor-extensions-library\WEB-INF\lib`
Or, in a managed server environment:

ORACLE_HOME/oracle/user_projects/applications/wc_domain/conductor-extensions-library/WEB-INF/lib

3. Restart the server.

67.5.6 Updating the WCPS Connections Configuration File

Finally, you need to update `wcps-connections.xml`. This file exists on the server, in a configuration directory in the default domain directory. In your own development environment, the simplest way to update this file is to edit it manually. The file is located in your default domain, here:

`DOMAIN_HOME/config/fmwconfig/wcps-connections.xml`

---

**Note:** After editing and saving this file, redeploy the `wcps-services` application using the WebLogic Server Console. To do this, go to the Deployments section of the console and locate `wcps-services` in the Deployments table. Select `wcps-services` and click **Update** to redeploy it.

---

Add the `<connection>` element shown in Example 67–5 to the `wcps-connections.xml` file.

**Example 67–5 Sample wcps-connection.xml Entry**

```
<connection>
    <connection-name>SayHelloConnection</connection-name>
    <connection-type>client.simple.connection</connection-type>
    <namespace>*</namespace>
    <properties>
        <property>
            <name>isDefault</name>
            <value>true</value>
        </property>
    </properties>
</connection>
```

The `namespace` attribute specifies the namespace in which the scenario exists. By default, the namespace is the name of your portal application. Or, `namespace` can be an asterisk (*), which specifies all namespaces.

**Tip:** You can discover the namespace through REST commands. From the `resourcemIndex`, follow the namespaces link. For example:

http://localhost:7101/wcps/api/conductor/resourceIndex

For more on the `resourcemIndex`, see Section 66.4, "Tutorial: Creating, Testing and Deploying a Simple Application."

The connection name is an arbitrary name that will appear in the scenario editor user interface. The connection type must match the `connectionType` annotation in your extension of `BaseConnectionConfig` (see Example 67–2). The connection types for the out-of-the-box data providers are:

- properties.provider.connection – Property Service Data Provider
- activity.provider.connection – Activity Graph Data Provider
- cmis.provider.connection – CMIS Data Provider

**Tip:** If you deploy your provider and find that it does not show up in the user interface, it’s possible you specified an incorrect namespace in `wcps-connections.xml`. Try using the asterisk (*) if you are in doubt about which namespace to use.

If you do not have direct access to the server file system, you can update the connection information using Fusion Middleware Control. For more details using Fusion Middleware Control, see the "Configuring Connections Using Fusion Middleware Control" section in *Administering Oracle WebCenter Portal*.

### 67.5.7 Testing the Deployment

To test the deployment, simply verify that the custom data provider shows up in the Scenario editor user interface. For details, see *Section 67.3, 'Discovering Data Providers.'*

If you’re unable to discover the deployed provider, then go back over your steps and carefully check that there are no obvious errors, like configuration or resource file misspellings. Be sure that you created your JAR file properly as described in *Section 67.5.3, "Setting Up the JAR File Structure."* Also, be sure you copied the JAR file to the correct domain.

These are common mistakes to double-check for:

- The **META-INF/services** contents must be included in the JAR file at the top level. For example, this folder is packaged in the JAR file as `META-INF/services/oracle.wcps.conductor.provider.IDataProvider`, not in any other location.
- The properties file describing the annotations must be included in the JAR file at the top level. See also *Section 67.4.4, "Creating a Resource Properties File."*
- The JAR file must be copied to the folder:
  
  ```properties
  DOMAIN_HOME/conductor-extensions/WEB-INF/lib
  ```

  You must restart the server after copying the file.

- The data provider connection entry must be added to `wcps-connections.xml`. See *Section 67.5.6, "Updating the WCPS Connections Configuration File."*

### 67.6 Invoking the Data Provider in a Scenario from JDeveloper

After your custom data provider is deployed successfully, you can use it in a scenario. Let’s see how this works for our example provider.

1. Right-click your project in the Application Navigator and select **New**.
2. In the New Gallery dialog, select the General category, then click **Personalization**.
3. In the Items list, select **Conductor Scenario** and click **OK**.
4. In the New Conductor Scenario dialog, give the scenario a name and click **OK**.
5. In the scenario editor, right-click the Start node and select **Add Following Statement** and then **Invoke Provider**.
6. Right-click the provider node and select **Invoke Provider Properties**.
7. Drill into the SayHelloProvider node and select the `sayHello(user)` method, as shown in Figure 67–8.

![Selecting the Data Provider Method](image)

8. Enter a variable name in the Variable Name field. You'll reference this variable in the next step, when you create a return statement. For example, simply enter `result`.

---

**Note:** It's easy to forget to enter the variable name. The variable becomes available within the scenario context, and therefore, can be reused by other scenario components and operations, as we will see.

---

9. In the Parameters section of the dialog, enter a value for the `user` parameter. This value will be a static variable that will print out when the scenario is run.

**Tip:** Another approach is to add an EL expression that gets the name of the logged in user and uses it in the scenario. For example: `${ScenarioContext.scenarioRequest.userPrincipal.name}`.

10. Click OK.

11. Right-click the Invoke Provider node and select **Add Following Statement**, and then select **Return**.

12. Right-click the Return node and select **Set Expression**.

13. In the Scenario Expression Builder dialog, simply enter the variable name you created in the Invoke Expression Provider dialog. You must use proper Expression Language syntax. For example, if you called the variable `result`, enter: `${result}`.
14. Click OK.
15. Right-click the Start node and select Run.
16. In the Scenario Input dialog, click OK.
17. After the server starts and the scenario is deployed, check the results in the JDeveloper log window, as shown in Figure 67–10.

For more information on using the scenario editor and the Expression Builder, see Chapter 66.2.2, "Authoring Personalized Scenarios in JDeveloper."

### 67.7 Designing Custom Data Providers: Best Practices

This section discusses best practices to consider in the design of a custom data provider:

- Section 67.7.1, "Create Domain-Specific Providers"
- Section 67.7.2, "Extend the Existing Data Provider Infrastructure"
- Section 67.7.3, "Reuse Scenarios and Data Providers"

#### 67.7.1 Create Domain-Specific Providers

Create individual data providers that are specific to your use cases or problem domains. For example, if you want to access data from a business services, like PeopleSoft and Siebel, the best practice is to create separate data providers to clearly delineate the two.
Also keep in mind that a single data provider could have multiple executable resources, each representing a specific service. For example, a PeopleSoftProvider could extend the base ExternalDataProvider, and have several executable resources: one for retrieving a person's basic information, another for retrieving the job description, and so on.

### 67.7.2 Extend the Existing Data Provider Infrastructure

It's a best practice to extend existing data provider classes rather than implementing your own from scratch.

- **Section 67.7.2.1, "Extend Base Classes"**
- **Section 67.7.2.2, "Use Connection Configuration"**
- **Section 67.7.2.3, "Reuse Connections"**
- **Section 67.7.2.4, "Use Caching"**

#### 67.7.2.1 Extend Base Classes

The Conductor provides a base infrastructure for data providers, and it is strongly recommended that you extend one of these providers instead of starting from scratch. As an example, an AbstractSOAPProvider could extend the ExternalDataProvider. More specific providers, such as a PeopleSoftProvider, could extend the SoapProvider.

#### 67.7.2.2 Use Connection Configuration

For external systems, make use of the Conductor connection configuration files. Information that is rather static, such as host, port, and WSDL, are good attributes to associate with the data provider connection. You can manage conductor connections using Fusion Middleware Control, meaning you can add or change the values of connection attributes at runtime. "Configuring Connections Using Fusion Middleware Control" in *Administering Oracle WebCenter Portal*.

#### 67.7.2.3 Reuse Connections

Because the Conductor's use of data providers is REST-ful, the data provider is instantiated every time a scenario is invoked. Sometimes, connections to external resources can be expensive to create. The base infrastructure for data providers has the ability to cache these connections.

#### 67.7.2.4 Use Caching

Personalization will provide APIs to leverage Coherence caching.

For details, see *Chapter 73, "Cache Management for Personalization."*

### 67.7.3 Reuse Scenarios and Data Providers

One of the most powerful aspects of Personalization is that you can reuse both data providers and scenarios in multiple contexts and with multiple clients. For instance, a single scenario can be reused to deliver data to a mobile application, a transactional web site, a portal, and a tablet.

Scenarios may invoke multiple data (and function) providers and encapsulate a particular business process. Scenarios may also be chained together: the output of one scenario can be used as input to another.
This chapter introduces function providers, discusses their important role in data integration and the Data Integration architecture, and presents a basic example that you can build and run in JDeveloper. After reading this chapter, you will have a basic understanding of how to implement a function provider.

This chapter includes the following chapters:

- Section 68.1, "Introduction to Function Providers"
- Section 68.2, "Do I Need to Write a Function Provider?"
- Section 68.3, "Discovering Function Providers"
- Section 68.4, "Implementing a Simple Custom Function Provider"
- Section 68.5, "Deploying a Custom Function Provider"
- Section 68.6, "Calling a Function Provider in a Scenario"

### 68.1 Introduction to Function Providers

Where a data provider connects to and retrieves data from a data source, a function provider operates on data within the context of a scenario. WebCenter Portal provides a number of function providers out-of-the-box, like providers for working with List objects, Date objects, SQL data, and more. Or, you can implement custom function providers to address specific use cases.

### 68.2 Do I Need to Write a Function Provider?

WebCenter Portal includes a set of out-of-the-box function providers that perform certain commonly needed tasks. WebCenter Portal provides function providers that address general operations, like List and Date creation and manipulation. In addition, out-of-the-box function providers are included that address the data returned by specific data providers, like the SQL and REST data providers.

---

**Note:** The out-of-the-box function providers are described in Chapter 69, "Function Provider Reference."

For some use cases, the set of standard function providers may be insufficient, and you then have the option of writing and deploying your own custom function providers. For example, if you need to work with a specific data set returned from a custom data
provider, you may need to write custom function providers to handle that data. See Section 68.4, "Implementing a Simple Custom Function Provider."

68.3 Discovering Function Providers

One way to discover the list of out-of-the-box and custom function providers is to bring up a Scenario Expression Builder dialog, as shown in Figure 68–1.

1. Create an Execute scenario node.
2. Right-click the Execute node and select Set Expression.

For an example, see Section 68.6, "Calling a Function Provider in a Scenario."

Figure 68–1 List of Function Providers

The function providers allow you to perform operations on data. For example, Figure 68–1 shows the Scenario Expression Builder dialog. The "collections" function provider is selected. The drop-down list lets you select a function provider method for handling a collection of data. The collections function provider is one of several out-of-the-box function providers.

After you select a function provider, you need to fill in values for its parameters, if there are any. You supply parameter values to function providers by editing an Expression Language statement within the Expression Builder dialog. The dialog also provides a list of any variables that are available in the current scenario context as well as the Contextual Services, which are variables that exist within the scope of the current application.
Implementing a Simple Custom Function Provider

Another way to "discover" the function providers available to you is to view REST responses in a tool that lets you make REST requests and view the responses.

1. In a browser, enter this URL, where host and port are the host and port name where the WCPS server is running. For example:

   http://example.com:8891/wcps/api/conductor/resourceIndex

2. Locate the namespace URI and enter its URL in the browser. For example:

   http://example.com:8891/wcps/api/conductor/namespaces

3. Locate the `functionProviders` URI for your application and enter its URL in the browser. Set the value of `projection` to `summary`. For example:

   http://example.com:8891/wcps/api/conductor/namespaces/MyApplication/functionProviders?projection=summary

   This URL returns a REST response containing all of the function providers that are deployed to the server. This is also a handy test to run to test the configuration for a new function provider.

68.4 Implementing a Simple Custom Function Provider

This section describes how to implement a simple custom function provider, configure it, and invoke it inside a scenario. Even though this is a very basic example, it illustrates many important aspects of custom provider development and configuration.

68.4.1 Overview of the Example and Prerequisites

Before you begin developing a function provider in JDeveloper, be sure you have things set up properly. There are a few configurations that you need to make or verify for your environment. For details, see Section 2.5, "Setting Up JDeveloper for Personalization."

As general guidance, Figure 68-2 shows a sample project structure for a custom function provider implementation. You can develop function providers an any type of JDeveloper project that supports Java development. For details on creating Java classes in JDeveloper, refer to JDeveloper Online Help.

Note: Function providers appear in other statement contexts within the scenario editor that use the Expression Builder, such as Conditional and Return statements; however, the Expression Builder user interface associated with each statement is the same.

Note: In this example, the provider is created in a Fusion Web Application in JDeveloper. You can also create function providers in WebCenter Portal applications or any other type of JDeveloper application that includes Java capabilities.
Implementing a Simple Custom Function Provider

**Note:** Your project structure will not be identical to the sample shown in Figure 68–2; however, the basic organization of components (source files, resource properties, and META-INF/services configuration file) must follow this pattern for successful deployment. For example, it is required that your resources directory be in the same directory as your class files. Again, by following the general pattern shown in Figure 68–2, your deployment will succeed. The individual parts of the project structure are discussed in the following sections.

Figure 68–2  Sample Function Provider Project Structure

68.4.2 Example Function Provider

All function providers are implementations of the IFunctionProvider interface. As with data providers, Java annotations allow function provider methods and parameters to appear in the scenario editor user interface.

Example 68–1 shows an example function provider. It provides a method for formatting numbers. In the following sections, we'll examine the parts of the class in detail, including the required Java annotations that appear at the top of the class and around methods and parameters.

**Tip:** The general instructions for creating a class are to select New from the File menu and choose to create a new Java class. Copy the example code into the file and save it. If you explicitly enter the package model (for this example) in the Create Java Class dialog, the package directory will be created for you and your class file will be placed in that package directory.

**Note:** Because function providers are simply Java components, you can follow best practices for exception handling as you would for any server-side Java components.

---

**Example 68–1  Example Function Provider**

```java
package model;

import oracle.wcps.conductor.annotation.FunctionProvider;
import oracle.wcps.conductor.provider.IFunctionProvider;
```
import oracle.wcps.conductor.annotation.PublicFunction;
import oracle.wcps.conductor.annotation.PublicParameter;
import java.text.DecimalFormat;
import java.lang.NumberFormatException;
import oracle.wcps.conductor.exception.ConductorRuntimeException;

@FunctionProvider
(prefix="number",
resourceBundle="resources.Providers",
nameBundleKey="function.provider.name",
descriptionBundleKey="function.provider.description"
)
public class FormatNumberProvider implements IFunctionProvider {
    @PublicFunction
    (functionName="format",
     descriptionBundleKey="format.description"
    )
    public static String getNumber(@PublicParameter(parameterName="pattern",
                                                      descriptionBundleKey="pattern.description") String pattern,
                              @PublicParameter(parameterName="number", descriptionBundleKey="number.description") String number) {
            Double dd;
            dd = new Double("1.0");
            try {
                dd = new Double(number);
            } catch (NumberFormatException e) {
                return "{You must enter a valid number.}";
            }
            DecimalFormat df = new DecimalFormat(pattern);
            number = df.format(dd);
            return number;
        }

Figure 68–3 shows our example Java source file, FormatNumProvider.java, as it appears in the Application Navigator in JDeveloper. The file is in a package called model.

Tip: Your source files may be organized differently, but Figure 68–3 at least shows a complete project structure for a function provider. This general pattern of project organization is the best practice.
Any object can be returned from a function provider. As a best practice, be sure that your return objects are marshalable, adhering to JAXB standards. Return objects that adhere to this criteria can be used anywhere within a scenario, like as input to a data provider.

### 68.4.3 Adding Java Class Annotations

Every function provider class requires the following Java annotation stanza immediately before the class definition:

```java
@FunctionProvider
{
    prefix="number",
    resourceBundle="resources.Providers",
    nameBundleKey="function.provider.name",
    descriptionBundleKey="function.provider.description"
}
```

The prefix element specifies the name that will appear in the scenario editor user interface and lets you select the function provider. For example, when this function provider is referenced in a scenario, the method is called with an Expression Language expression like this:`${number:getNumber(format, number)}`.

---

**Note:** Prefix names and variables must not contain hyphens or spaces. If hyphens or spaces appear in the prefix names or variables, they cannot be referenced by Expression Language expressions.

---

The `resourceBundle` element refers to a resource file that exists in the class path of the function provider. The other annotations are resource bundle keys used to identify strings in the resource file, which primarily support localization.

Annotations are also required around your function provider's methods and method parameters, as explained in the next section. Note that to use annotations, you must import the following classes:

```java
import oracle.wcps.conductor.annotation.PublicFunction;
import oracle.wcps.conductor.annotation.PublicParameter;
```
68.4.4 Writing Function Provider Methods

Methods and their parameters must be annotated, following the pattern shown in Example 68–2. The \texttt{functionName} value appears in the user interface, and the \texttt{descriptionBundleKey} value allows a description to be coded into the resource file specified at the top of the class. The same pattern applies to method parameter annotations.

\textbf{Tip:} If you create methods with a Java Bean style "get" syntax, you can access object attributes in EL using convenient dot notation. For example, if your provider has a "foo" attribute and a "getFoo()" method, the attribute can be accessed through dot notation in EL in a scenario with a statement like this: \texttt{$\{myReturnObject.foo\}.

\textbf{Example 68–2 Java Annotations for Methods and Parameters}

\begin{verbatim}
@PublicFunction
{
    functionName="format",
    descriptionBundleKey="format.description"
}
public static String getNumber(@PublicParameter(parameterName="pattern",
    descriptionBundleKey="pattern.description") String pattern,
    @PublicParameter(parameterName="number", descriptionBundleKey="number.description") String number) {

    Double dd;
    dd = new Double("1.0");
    try {
        dd = new Double(number);
    } catch (NumberFormatException e){
        return "You must enter a valid number."
    }
    DecimalFormat df = new DecimalFormat(pattern);
    return number = df.format(dd);
}
\end{verbatim}

68.5 Deploying a Custom Function Provider

After you create a new function provider class, you need to compile it and deploy it properly. After you accomplish these tasks successfully, the provider will show up in the scenario editor’s Expression Builder user interface. This section explains the required steps.

\textbf{Note:} Function provider deployment can be tricky, because it’s largely a manual procedure. You must follow these instructions carefully—even a simple typo at any point will prevent the provider from being deployed.

To summarize the steps, you must deploy a JAR file containing:

- your compiled function provider class(es)
- a resource properties file (which must be in the same directory as the classes)
- a configuration file listing the classname(s) of your function provider(s)
The JAR file must then be manually copied to the server and the server must be restarted.

68.5.1 Compile Your Function Provider Class(es)

Function providers are written in Java, and must be compiled before they are deployed.

1. In Application Navigator, right-click the project containing your source code and select either the Make or Rebuild options, as shown in Figure 68–4.

![Figure 68–4 Compiling a Project](image)

68.5.2 Create a Resource Properties File

The Java annotations placed in the function provider class file point to a resource properties file. This file must exist and must include the required keys and values. The basic steps for creating the file are:

1. Right-click the project containing your Java files and select New.
2. In the New Gallery dialog, select General.
3. Create a Java package. In this example, the package is called resources.
4. Create a property file in the package. In this example, the file is called Providers.properties.

Figure 68–5 shows a properties file called Providers.properties in the resources package.
For example, the following resource keys were listed in the NumberFormatProvider class annotations (see the listing in Example 68–1). The first annotation, format.description, describes the provider's purpose. The others describe each of the input parameters.

\[
\begin{align*}
\text{format.description} &= \text{Format a number.} \\
\text{pattern.description} &= \text{A number format pattern like #.##.} \\
\text{number.description} &= \text{The number to format.}
\end{align*}
\]

68.5.3 Create a Configuration File in META-INF/services

The last step in preparing your function provider for deployment is to create a simple configuration file that lists the full class name(s) of your provider(s).

1. In the Application Navigator, create a new folder called META-INF in the Application Sources folder.
2. In the META-INF folder, create a folder called services.
3. In the services folder, create a file called oracle.wcps.conductor.provider.IFunctionProvider. You must use this exact name. See Figure 68–6.
4. Open the `oracle.wcps.conductor.provider.IFunctionProvider` file and enter the full classname for each function provider you wish to deploy.

   For example:

   ```java
   model.FormatNumProvider
   ```

68.5.4 Set Up the JAR File Structure

If you are unfamiliar with JAR file creation in JDeveloper, this section explains how to set up your JAR file structure (a deployment profile) to ensure that all the required elements are included.

1. Right-click your project (in this example, the project called Portal) in the Application Navigator and select **Project Properties**.
2. In the Project Properties dialog, select **Deployment**.
3. Click **New**.
4. Select JAR File as the Archive Type and give the JAR a name.
5. Click OK.
6. In the JAR Options dialog, select **Contributors**.
7. Be sure that Project Output Directory, Project Source Path, and Project Dependencies are selected, as shown in Figure 68–7.
8. Select Filters.
9. In the Files tab, select only the files you want to be added to the JAR. The JAR should include the META-INF/services directory contents, Java classes, and resources folder, as shown in Figure 68–8.
10. Click OK, and then OK in the Project Properties dialog.

11. Save your project.

### 68.5.5 Create a JAR File

After the JAR deployment profile is set, you are ready to generate a JAR file from your project.

1. Right-click the project (in this case, the Portal project) in the Application Navigator and select **Deploy**.

2. Select the JAR file you want to deploy. This is the name you gave the file in **Section 68.5.4, “Set Up the JAR File Structure.”**

3. In the Deployment Action dialog, select **Deploy to JAR file** and click **Finish**.

The JAR file is placed in your project directory. You can view it on the file system in: `<Workspace_Directory>/Portal/deploy`. For example, if your Workspace is called MyProviders, you might find the JAR here: C:/JDeveloper/mywork/MyProviders/Portal/deploy.

### 68.5.6 Copy the JAR File to the Server

You must copy the JAR file to the server.

1. Copy the JAR file that you created in the previous step. As noted previously, the file is located in `<Workspace_Directory>/Portal/deploy`.

2. Paste the JAR file into the following location on the server:
For example, for an Integrated WebLogic Server environment:

C:\JDeveloper\system11.1.7.40.63.82\DefaultDomain\conductor-extensions-library\WEB-INF\lib

Or, in a managed server environment:

ORACLE_HOME/oracle/user_projects/applications/wc_domain/conductor-extensions-library/WEB-INF/lib

3. Restart the server.

68.5.7 Test the Deployment

To test the deployment, simply verify that the custom data provider shows up in the UI or that you can discover it through the REST API. For details, see Section 68.3, "Discovering Function Providers."

If you're unable to discover the deployed function provider, then go back over your steps and carefully check that there are no obvious errors, like configuration or resource file misspellings. Be sure that you created your JAR file properly as described in Section 68.5.4, "Set Up the JAR File Structure." Also, be sure you copied the JAR file to the correct domain.

These are common mistakes to double-check for:

- The META-INF/services contents must be included in the JAR file
- The JAR file must be copied to your DOMAIN_HOME/conductor-extensions/WEB-INF/lib folder. You must restart the server after copying the file.

68.6 Calling a Function Provider in a Scenario

The scenario editor allows you to call function providers wherever you can perform an operation on a variable, like when you set a variable, create a return statement, create an input parameter to a data provider, or set up a conditional branch statement.

You always call a function provider from within the Expression Builder dialog. This dialog lets you select a function provider and select the specific method you wish to execute. If the provider takes input parameters, you must specify them in an Expression Language (EL) statement. An EL statement is automatically generated for you when you select your method, and all you need to do then is select or type in the appropriate variables.

For example, let's look at our example function provider, NumberFormatProvider (listed in Example 68–1).

The provider class had an annotation called "prefix." In our example, this annotation was set to the value "number." In the Expression Builder dialog, you can see that a provider called "number" appears in the drop down list of function providers. Figure 68–9 shows our example function provider called "number" is selected in the Functions drop down list. Under "number" is a list of methods in the provider. Our example contains just one method, called "format". Recall that "format" was provided in the method's functionName annotation.
The Expression field in the dialog is automatically filled in with the EL statement that allows you to pass variables to the function provider. The expression is:

${number:format(format,myNumber)}

To parse the expression:

- **numbers** – The name of the function provider as provided in the function provider class annotation.
- **format** – The name of the method, as provided in the method's annotation.
- **pattern** – The name of the first parameter, as provided in the method's annotation. In this case, the parameter is of type String, as indicated in the Description. As you can see in the function parameter's implementation, this string is a pattern that represents how you want the number to be formatted. This parameter comes from the Java class DecimalNumber. For more information, refer to the Javadoc for that class.
- **number** – The name of the second parameter, as provided in the method's annotation. In this case, the parameter is also of type String. It's just a number represented as a string.

For more information on using the scenario editor and the Expression Builder, see Chapter 66.2.2, "Authoring Personalized Scenarios in JDeveloper."
This chapter discusses the set of standard, out-of-the-box function providers and their associated methods.

Function provider methods can be invoked from a Scenario to manipulate data returned by a data provider. The set of standard function providers described in this chapter are provided with WebCenter Portal out-of-the-box and perform commonly required operations upon results returned from data providers.

For information on implementing a custom function provider, see Chapter 68, "Implementing Custom Function Providers: Introduction."

Table 69–1 lists the different categories of standard function provider methods.

<table>
<thead>
<tr>
<th>Function Provider Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cache Methods</td>
<td>Perform operations on a named cache.</td>
</tr>
<tr>
<td>Maps Methods</td>
<td>Create and update a Map object.</td>
</tr>
<tr>
<td>List Methods</td>
<td>Create and update a List object.</td>
</tr>
<tr>
<td>String Methods</td>
<td>Perform common String manipulations.</td>
</tr>
<tr>
<td>Date Methods</td>
<td>Create and operate upon Date objects.</td>
</tr>
<tr>
<td>System Methods</td>
<td>Perform print operations.</td>
</tr>
<tr>
<td>Collections Methods</td>
<td>Create, update, and transform Collections objects.</td>
</tr>
<tr>
<td>Security Methods</td>
<td>Determines if a user is in a specified role.</td>
</tr>
<tr>
<td>CMIS Methods</td>
<td>Supports the CMIS provider.</td>
</tr>
<tr>
<td>Activity Graph Methods</td>
<td>Supports the Activity Graph provider.</td>
</tr>
<tr>
<td>Security Methods</td>
<td>Lets you get information about users and roles.</td>
</tr>
</tbody>
</table>
Cache Methods

Perform operations on a named Oracle Coherence cache object.

<table>
<thead>
<tr>
<th>Use this method...</th>
<th>To...</th>
</tr>
</thead>
<tbody>
<tr>
<td>getCache</td>
<td>Return a named cache using the entry in the specified cache configuration file.</td>
</tr>
<tr>
<td>get</td>
<td>Return the value for the address key in a named cache.</td>
</tr>
<tr>
<td>clear</td>
<td>Clear a named cache.</td>
</tr>
<tr>
<td>getCustomCache</td>
<td>Gets the custom cache configured in wcps-dataprovider-cache-config.xml.</td>
</tr>
<tr>
<td>getLargeCache</td>
<td>Gets a large cache configured in wcps-dataprovider-cache-config.xml.</td>
</tr>
<tr>
<td>getMediumCache</td>
<td>Gets a medium cache configured in wcps-dataprovider-cache-config.xml.</td>
</tr>
<tr>
<td>getSmallCache</td>
<td>Gets a medium cache configured in wcps-dataprovider-cache-config.xml.</td>
</tr>
<tr>
<td>put</td>
<td>Puts a value into a named cache.</td>
</tr>
</tbody>
</table>
getCache

Returns a named cache using the entry in the specified cache configuration file.

**Arguments**

*cacheConfigFile*
(String) The name of a cache configuration XML file. For custom function providers, this configuration file must be in the application classpath. Otherwise, it can be anywhere in the classpath of the caller.

*cacheName*
(String) The name of the cache returned by this method.

**Expression Language Format**

${cache:getCache(configFile,cacheName)}$

**Returns**

com.tangosol.net.NamedCache

**Example**

${cache:getCache('my-config.xml','mycache')}$

**Usage Notes**

None.
get

Returns the value to which the cache maps the specified key.

Arguments

**cacheName**

**key**
(Object) The key to the address value in the named cache.

**Expression Language Format**

${cache:getCache(cache, key)}

**Returns**

(Object) The value to which the cache maps the specified key.

**Example**

${cache:get(mycache, aKey)}

**Usage Notes**

None.
clear

Clears the named cache.

**Arguments**

cacheName

**Expression Language Format**

`$\{cache:getCache(cacheName)\}`

**Example**

`$\{cache:clear(aCache)\}`

**Usage Notes**

None.
getCustomCache

Gets the custom cache configured in wcps-dataprovider-cache-config.xml.

Arguments

**cacheName**
(String) The name of a cache configured in wcps-dataprovider-cache-config.xml.

Expression Language Format

```java
${cache:getCache("cacheName")}
```

(Object) The value to which the cache maps the specified key.

Returns


Example

```java
${cache:clear("aCache")}
```

Usage Notes

None.
getLargeCache

Gets a large cache configured in wcps-dataprovider-cache-config.xml. The default cache parameters are:

- 10000 entries
- 1 hour TTL

Arguments

cacheName
(String) The name of a large cache configured in wcps-dataprovider-cache-config.xml.

Expression Language Format

`${cache:getLargeCache("cacheName")}`

Returns

(com.tangosol.net.NamedCache) A cache object named wcps-dp-large-<cacheName>.

Example

`${cache:getLargeCache("aLargeCache")}`

Usage Notes

None.
getMediumCache

Gets a medium cache configured in `wcps-dataprovider-cache-config.xml`. The default cache parameters are:

- 1000 entries
- 1 hour TTL

**Arguments**

- **cacheName** (String) The name of a medium cache configured in `wcps-dataprovider-cache-config.xml`.

**Expression Language Format**

```
${cache:getMediumCache('cacheName')}
```

**Returns**


**Example**

```
${cache:getMediumCache('aMediumCache')}
```

**Usage Notes**

None.
getSmallCache

Gets a small cache configured in wcps-dataprovider-cache-config.xml. The default cache parameters are:

- 100 entries
- 1 hour TTL

Arguments

cacheName
(String) The name of a small cache configured in wcps-dataprovider-cache-config.xml.

Expression Language Format

${cache:getSmallCache("cacheName")}

Returns


Example

${cache:getSmallCache("aSmallCache")}

Usage Notes

None.
put

Puts an value into a named cache.

**Arguments**

**cache**

**key**
(java.lang.Object) A key to identify the saved object.

**value**
(java.lang.Object) An object to cache.

**Expression Language Format**

`${cache:put(aCache, aKey, aValue)}`

**Returns**

void

**Example**

`${cache:put(myCache, myKey, myValue)}`

**Usage Notes**

None.
Maps Methods

Utility methods that operate on Map objects.

**Table 69–3  Maps Methods**

<table>
<thead>
<tr>
<th>Use this method...</th>
<th>To...</th>
</tr>
</thead>
<tbody>
<tr>
<td>new</td>
<td>Create and return a new HashMap object.</td>
</tr>
<tr>
<td>put</td>
<td>Puts an entry in an existing map of key/value pairs.</td>
</tr>
<tr>
<td>size</td>
<td>Returns the number of entries in a Map.</td>
</tr>
<tr>
<td>keyset</td>
<td>Returns a collection of the keys in the Map.</td>
</tr>
<tr>
<td>values</td>
<td>Returns a collection of the values in a Map object.</td>
</tr>
<tr>
<td>get</td>
<td>Return a value from a Map object that is associated with a key.</td>
</tr>
</tbody>
</table>
Create and return a new HashMap object.

**Arguments**
None.

**Expression Language Format**
`${maps:new()}`

**Returns**
java.util.HashMap

**Example**
`${maps:new()}`

**Usage Notes**
None.
put

Puts an entry in an existing map of key/value pairs.

**Arguments**

*map*
The name of the HashMap.

*key*
A key object.

*value*
A value mapped to the key.

**Expression Language Format**

\${maps:put(map,key,value)}

**Returns**

Returns the new Map object (java.util.Map).

**Example**

\${(maps:put(existingMap, keyObj, aValue))}

**Usage Notes**

None.
size

Returns the number of entries in a Map.

**Arguments**

map
(java.util.Map) A map object.

**Expression Language Format**

${maps:size(aMap)}$

**Returns**

int

**Example**

${maps:size(aMap)}$

**Usage Notes**

None.
keyset

Returns a collection of the keys in the Map.

**Arguments**

**map**
(java.util.Map) A Map object.

**Expression Language Format**

${maps:put(map)}$

**Returns**

java.util.Set

**Example**

${maps:put(aMap)}$

**Usage Notes**

None.
values

Returns a collection of the values in a Map object.

Arguments

map
(java.util.Map) A Map object.

key
(java.lang.Object) A key object.

value<br/>(<V>) A value mapped to the key.

Expression Language Format

${maps:values(map)}

Returns

java.util.Collection

Example

${maps:values(aMap)}

Usage Notes

None.
get

Return a value from a Map object that is associated with a key.

Arguments

map
(java.util.Map) A Map object.

key
(java.lang.Object) A key object.

Expression Language Format

${maps:put(map, key)}$

Returns

A value of type <V>.

Example

${maps:put(aMap, aKey)}$

Usage Notes

None.
withEntries

Creates a HashMap populated with the given key/value pairs.

**Arguments**

key – (String) A key.

tvalue – (String) A value.

**Expression Language Format**

${maps:withEntries(keyvalues)}$e

**Returns**

t.java.util.Map

**Usage Notes**

None.
List Methods

Constructs and adds to a List of objects.

Table 69–4  List Methods

<table>
<thead>
<tr>
<th>Use this method...</th>
<th>To...</th>
</tr>
</thead>
<tbody>
<tr>
<td>new</td>
<td>Create and return a new ArrayList object.</td>
</tr>
<tr>
<td>add</td>
<td>Adds an object to a List.</td>
</tr>
<tr>
<td>size</td>
<td>Returns the number of elements in a List object.</td>
</tr>
</tbody>
</table>
Create and return a new ArrayList object.

**Arguments**
None.

**Expression Language Format**
${list:new()}

**Returns**
java.util.ArrayList

**Example**
${list:new()}

**Usage Notes**
None.
add

Adds an object to a List.

Arguments

list
(java.util.List) An object of type List.

value
(java.lang.Object) A value to add to the List.

Expression Language Format

${list:add(list,value)}

Returns

java.lang.Boolean

Example

${list.add(existingList, aValue)}

Usage Notes

None.
size

Returns the number of elements in a List object.

Arguments

list
(java.util.List) A List object.

Expression Language Format

${list:size(list)}

Returns

int

Example

${list:size(mylist)}

Usage Notes

None.
String Methods

Performs basic utility functions on strings.

Table 69–5  String Methods

<table>
<thead>
<tr>
<th>Use this method...</th>
<th>To...</th>
</tr>
</thead>
<tbody>
<tr>
<td>concat</td>
<td>Concatenates two strings and returns a new String object.</td>
</tr>
<tr>
<td>contains</td>
<td>Returns true if this string contains the specified string.</td>
</tr>
<tr>
<td>indexOf</td>
<td>Returns the index within this string of the first occurrence of the specified substring.</td>
</tr>
<tr>
<td>join</td>
<td>Join together String objects in a given Collection, separating them by a specified delimiter, and returning a new String.</td>
</tr>
<tr>
<td>replaceAll</td>
<td>Replaces each substring of this string that matches the given regular expression with the given replacement string.</td>
</tr>
<tr>
<td>split</td>
<td>Splits this string around matches of the given regular expression.</td>
</tr>
<tr>
<td>startsWith</td>
<td>Tests if this string starts with the specified prefix.</td>
</tr>
<tr>
<td>endsWith</td>
<td>Tests if this string ends with the specified prefix.</td>
</tr>
<tr>
<td>substring</td>
<td>Returns a new string that is a substring of this string.</td>
</tr>
</tbody>
</table>
concat

Concatenates two strings and returns a new String object.

**Arguments**

- **stringA**  
  (java.lang.String) The String to concatenate to.

- **stringB**  
  (java.lang.String) The String to concatenate onto the end of stringA.

**Expression Language Format**

`${string:concat(stringA, stringB)}`

**Returns**

java.lang.String

**Example**

`${cache:getCache('Hello, ','world!')}`

**Usage Notes**

None.
contains

Returns true if this string contains the specified string.

Arguments

string
(java.lang.String) The String to search.

containsString
(java.lang.String) A string to search for.

Expression Language Format

${string:contains(string,containsString)}

Returns

java.lang.Boolean

Example

${cache:getCache("the quick brown fox","fox")}

Usage Notes

None.
indexOf

Returns the index within this string of the first occurrence of the specified substring.

Arguments

string
(java.lang.String) A string to search for a substring.

substring
(java.lang.String) The substring to search for.

Expression Language Format

${string:indexOf(string,substring)}

Returns

int

Example

${cache:getCache('the quick brown fox', "quick")}

Usage Notes

None.
Join together String objects in a given Collection, separating them by a specified delimiter, and returning a new String.

**Arguments**

*collection*
A Collection object containing strings.

*delimiter*
A string token used to split the input string values.

**Expression Language Format**

```
${string:join(collection,delimiter)}
```

**Returns**

java.lang.String

**Example**

```
${cache:getCache(aCollection,', ', '')}
```

**Usage Notes**

None.
replaceAll

Replaces each substring of this string that matches the given regular expression with the given replacement string.

**Arguments**

**string**
The original String object.

**regex**
A regular expression used to identify sub-strings.

**replacementString**
A String to replace the original String input variable.

**Expression Language Format**

`${string:replaceAll(string,regex,replacementString)}`

**Returns**

`java.lang.String`

**Example**

`${cache:getCache("SomeSimpleString, 'Simple', 'Complex")}`

**Usage Notes**

None.
split

Splits this string around matches of the given regular expression.

**Arguments**

**string**
A String object to split.

**splitToken**
A token String used to split the input String parameter.

**Expression Language Format**

`${string:split(string,splitToken)}`

**Returns**

java.lang.String

**Example**

`${cache:getCache("one, two, three"), ",")}`

**Usage Notes**

None.
startsWith

Tests if this string starts with the specified prefix.

**Arguments**

string
The String to test.

start
The test string.

**Expression Language Format**

`${string:startsWith(string,start)}`

**Returns**

(java.lang.Boolean) Returns true if the input string starts with the test string.

**Example**

`${cache:getCache("BeginString", "Begin")}`

**Usage Notes**

None.
endsWith

Tests if this string ends with the specified prefix.

**Arguments**

**string**
The String to test.

**start**
The test string.

**Expression Language Format**

```
${string:startsWith(string,end)}
```

**Returns**

(java.lang.Boolean) Returns true if the input string starts with the test string.

**Example**

```
${cache:getCache("StringEnd", "End")}
```

**Usage Notes**

None.
substring

Returns a new string that is a substring of this string.

**Arguments**

**inputString**
The String subset.

**startIndex**
The beginning index, inclusive.

**endIndex**
The ending index, inclusive.

**Expression Language Format**

```
${string:startsWith(string,end)}
```

**Returns**

(java.lang.Boolean) Returns true if the input string starts with the test string.

**Example**

```
${cache:getCache('The quick brown fox', 1, 10)}
```

**Usage Notes**

None.
Date Methods

Utility methods for handling Date objects.

<table>
<thead>
<tr>
<th>Use this method...</th>
<th>To...</th>
</tr>
</thead>
<tbody>
<tr>
<td>after</td>
<td>Tests if this date is after the specified date.</td>
</tr>
<tr>
<td>before</td>
<td>Tests if this date is after the specified date.</td>
</tr>
<tr>
<td>diffDates</td>
<td>Returns the difference between two dates (date1 – date2) by the specified unit of measure.</td>
</tr>
<tr>
<td>diffFromNow</td>
<td>Returns the difference between today’s date and the specified date, by the specified unit of measure.</td>
</tr>
<tr>
<td>format</td>
<td>Formats the provided date with the specified format string.</td>
</tr>
<tr>
<td>new</td>
<td>Returns a new date with the current date/time.</td>
</tr>
</tbody>
</table>
after

Tests if this date is after the specified date.

**Arguments**

*date1*  
(java.util.Date) The first date to test.

*date2*  
(java.util.Date) The second date to test.

**Expression Language Format**

`${date:after(date1,date2)}`

**Returns**

(java.lang.Boolean) Returns true if the second date is after the first date.

**Example**

`${date:after(aDate, bDate)}`

**Usage Notes**

None.
Date Methods

before

Tests if this date is before the specified date.

Arguments

date1
(java.util.Date) The first date to test.

date2
(java.util.Date) The second date to test.

Expression Language Format

${date:after(date1,date2)}$

Returns

(java.lang.Boolean) Returns true if the second date is before the first date.

Example

${date:before(aDate, bDate)}$

Usage Notes

None.
diffDates

Returns the difference between two dates (date1 – date2) by the specified unit of measure.

Arguments

date1
The first Date to test.

date
The second Date to test.

units
(String) Valid units are:

<table>
<thead>
<tr>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>Year</td>
</tr>
<tr>
<td>M</td>
<td>Month of Year</td>
</tr>
<tr>
<td>w</td>
<td>Week of Year</td>
</tr>
<tr>
<td>d</td>
<td>Day of Year</td>
</tr>
<tr>
<td>m</td>
<td>Minutes</td>
</tr>
<tr>
<td>s</td>
<td>Seconds</td>
</tr>
<tr>
<td>l</td>
<td>Milliseconds</td>
</tr>
</tbody>
</table>

Expression Language Format

${date:diffDates(date1, date2, units)}

Returns

(int) The calculated difference between the two dates, as specified by the input units. For example, if the input units is "y", the result will be the number of years separating the two dates.

Example

${date:diffDates(aDate, bDate, 'M')}

Usage Notes

None.
**diffFromNow**

Returns the difference between today’s date and the specified date, by the specified unit of measure.

**Arguments**

*date1*  
(java.util.Date) The Date to test.

*units*  
(String) Valid units are:

**Table 69–8 Valid Values for the units Parameter**

<table>
<thead>
<tr>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>Year</td>
</tr>
<tr>
<td>M</td>
<td>Month of Year</td>
</tr>
<tr>
<td>w</td>
<td>Week of Year</td>
</tr>
<tr>
<td>d</td>
<td>Day of Year</td>
</tr>
<tr>
<td>m</td>
<td>Minutes</td>
</tr>
<tr>
<td>s</td>
<td>Seconds</td>
</tr>
<tr>
<td>l</td>
<td>Milliseconds</td>
</tr>
</tbody>
</table>

**Expression Language Format**

\$\{date:diffFromNow(date1, date2, units)\}$

**Returns**

(int) The calculated difference between the two dates, as specified by the input units. For example, if the input units is "y", the result will be the number of years separating the two dates.

**Example**

\$\{date:diffFromNow(aDate, bDate, "M")\}$

**Usage Notes**

None.
format

Formats the provided date with the specified format string.

Arguments

date
(java.util.Date) The Date object to format.

pattern
The Date pattern to format the date with.

Expression Language Format

${date:format(date,pattern)}

Returns

java.lang.String

Example

${date:format(aDate, "yyyy-MM-dd")}

Usage Notes

None.
new

Returns a new date with the current date/time.

**Arguments**
None.

**Expression Language Format**
${date:new()}$

**Returns**
java.lang.Date

**Example**
${date:new()}$

**Usage Notes**
None.
System Methods

Provides standard print methods.

<table>
<thead>
<tr>
<th>Use this method...</th>
<th>To...</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>print</code></td>
<td>Print an object to either standard error output or the standard console.</td>
</tr>
<tr>
<td><code>println</code></td>
<td>Print an object to either standard error output or the standard console with a newline.</td>
</tr>
</tbody>
</table>
**print**

Prints an object.

**Arguments**

*stream*
(String) Specify *err* to print to the standard error output stream or *out* to print to the standard console.

*object*
(java.lang.Object) The Object to print.

**Expression Language Format**

`${system:print(stream,object)}`

**Returns**

void

**Example**

`${cache:getCache("err", "Hello, world!")} `

**Usage Notes**

Output goes to the WC_Utilities.out file for the "out" option and WC_Utilities.err file for the "err" option. These files are located in the server domain directory.
println

Prints an object with a newline.

**Arguments**

**stream**
(String) Specify `err` to print to the standard error output stream or `out` to print to the standard console.

**object**
(java.lang.Object) The Object to print.

**Expression Language Format**

```
${system:println(stream, object)}
```

**Returns**

void

**Example**

```
${cache:println('out', 'Hello, world!')}
```

**Usage Notes**

Output goes to the WC_Utilities.out file for the "out" option and WC_Utilities.err file for the "err" option. These files are located in the server domain directory.
Security Methods

Let's you discover information about users and roles.

<table>
<thead>
<tr>
<th>Use this method...</th>
<th>To...</th>
</tr>
</thead>
<tbody>
<tr>
<td>getUserRoles</td>
<td>Returns an array of roles/groups that is part of the current subject.</td>
</tr>
<tr>
<td>isUserInRole</td>
<td>Determines if the current user is a member of the specified role or group name.</td>
</tr>
</tbody>
</table>
getUserRoles

Returns an array of roles/groups that is part of the current subject.

**Arguments**
None.

**Expression Language Format**
${security:getUserRoles()}

**Returns**
java.util.List

**Example**
${security:getUserRoles()}

**Usage Notes**
None.
isUserInRole

Determines if the current user is a member in the specified role or group name.

Arguments

role
(String) The name of a role or group.

Expression Language Format

\${security:isUserInRole(role)}

Returns

Boolean

Example

\${security:isUserInRole("Moderator")}

Usage Notes

None.
Collections Methods

Perform operations on Collections objects.

<table>
<thead>
<tr>
<th>Table 69–11  Collections Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Use this method...</strong></td>
</tr>
<tr>
<td>append</td>
</tr>
<tr>
<td>fromArray</td>
</tr>
<tr>
<td>size</td>
</tr>
<tr>
<td>sort</td>
</tr>
<tr>
<td>toArray</td>
</tr>
<tr>
<td>fromArray</td>
</tr>
</tbody>
</table>
append

Appends an object to a collection of objects.

**Arguments**

- **collection**

- **item**
  (java.lang.Object) An Object to append to the Collection.

**Expression Language Format**

${collections:append(collection,item)}$

**Returns**

java.util.Collection

**Example**

${collections:append(aCollection, anObject)}$

**Usage Notes**

None.
toArray

Returns an array of Objects from a collection.

Arguments

collection (java.util.Collection) A Collection object to turn into an array of Objects.

Expression Language Format

\${collections:toArray(collection)}

Returns

java.lang.Object

Example

\${cache:getCache(aCollection)}

Usage Notes

None.
fromArray

Returns a List object populated with objects from an array of Objects.

Arguments
array
(java.lang.Object[]) An array of Objects.

Expression Language Format
${collections:fromArray(array)}

Returns
java.util.Collection

Example
${collections:fromArray(anArray)}

Usage Notes
None.
size

Determines the size of a collection. If the collection is null, zero is returned.

**Arguments**

**collection**

**Expression Language Format**

${collections:size(collection)}

**Returns**

long

**Example**

${collections:size(aCollection)}

**Usage Notes**

None.
sort

Sorts the specified list into ascending order, according to the natural ordering of its elements.

**Arguments**

**list**

(java.util.List) A List object to sort.

**Expression Language Format**

${collections:sort(list)}$

**Returns**

void

**Example**

${collections:sort(myList)}$

**Usage Notes**

None.
new

Returns a new Collection object.

Arguments
None.

Expression Language Format
${collections:new()}

Returns
java.lang.Collection

Example
${collections:new()}

Usage Notes
None.
Activity Graph Methods

Perform operations on Activity Graphs.

**Table 69–12  Activity Graph Methods**

<table>
<thead>
<tr>
<th>Use this method...</th>
<th>To...</th>
</tr>
</thead>
<tbody>
<tr>
<td>filterRecsByScore</td>
<td>Filter out Recommendations whose score is &gt;= the input score.</td>
</tr>
<tr>
<td>getCMISLinksFromCommonItems</td>
<td>Return a List of URLs representing the common items as CMIS objects.</td>
</tr>
<tr>
<td>getCMISLinksFromRecommendations</td>
<td>Return a List of URLs representing the CMIS objects.</td>
</tr>
<tr>
<td>getContentIDs</td>
<td>Extract content IDs from input objects. Includes content of all types (WC.document, WC.wiki).</td>
</tr>
<tr>
<td>getContentIDsExclude</td>
<td>Extract content IDs from input objects.</td>
</tr>
<tr>
<td>getContentIDsFiltered</td>
<td>Extract content IDs from input objects.</td>
</tr>
</tbody>
</table>
filterRecsByScore

Filter out Recommendations whose score is >= the input score. Return only those Recommendations that are equal to or above the cutoffScore.

Arguments

recommendations

cutoffScore
(float) The cutoff score.

Expression Language Format

${agfunction:filterRecsByScore{recommendations,cutoffScore}}$

Returns

oracle.wcps.activity.agrest.spy.jaxb.Recommendations

Usage Notes

None.
getCMISLinksFromCommonItems

Return a List of clickable URLs representing the actual content item in common items as CMIS objects.

**Arguments**

*results*

(oracle.wcps.activity.agrest.spy.jaxb.Results)

**Expression Language Format**

`${agfunction:getCMISLinksFromCommonItems(results)}`

**Returns**

java.util.List

**Usage Notes**

None.
getCMISLinksFromRecommendations

Return a List of clickable URLs representing actual content item in the recommendations.

Arguments

recommendation
(oracle.wcps.activity.agrest.spy.jaxb.Recommendations)

Expression Language Format

${agfunction:getCMISLinksFromRecommendations(recommendations)}

Returns

java.util.List

Usage Notes

None.
**getContentIDs**

Extract content IDs from input objects. Includes content of all types (WC.document, WC.wiki).

Returns the short-version content identifier from the Recommendations results. The parameter 'agResults' can be one of:

- Recommendations
- RecommendedItems
- List<Recommendation>

**Arguments**

**agResults**
(java.lang.Object)

**Expression Language Format**

${agfunction:getContentIDs(agResults)}

**Returns**

java.util.List

**Usage Notes**

None.
getContentIDsExclude

Returns the short-version content identifier from the Recommendations results. excludes the 'excludeClassURN' from the results. The parameter 'agResults' can be one of:
- Recommendations
- RecommendedItems
- List<Recommendations>

Arguments

agResults
(java.lang.Object)

Expression Language Format

${agfunction:getContentIDsExclude(agResults,excludeClassURN)}

Returns

java.util.List

Usage Notes

None.
**getContentIDsFiltered**

Returns the short-version content identifier from the Recommendations results. returns only those IDs corresponding to the classURN. See the nomenclature section in this blog for more details on ClassURN. The parameter 'agResults' can be one of:

- Recommendations
- RecommendedItems
- List<Recommendations>

**Arguments**

- **agResults**
  java.lang.Object

- **filterClassURN**
  java.lang.String

**Expression Language Format**

`${agfunction:getContentIDsFiltered(agResults,filterClassURN)}`

**Returns**

java.util.List

**Usage Notes**

None.
**CMIS Methods**

Perform operations on CMIS.

**Table 69–13  CMIS Methods**

<table>
<thead>
<tr>
<th>Use this method...</th>
<th>To...</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>getCMISLinksFromCommonItems</code></td>
<td>Return a List of URLs representing the common items as CMIS objects.</td>
</tr>
<tr>
<td><code>getCMISLinksFromRecommendations</code></td>
<td>Return a List of URLs representing the CMIS objects.</td>
</tr>
<tr>
<td><code>getContentIDs</code></td>
<td>Extract content IDs from input objects. Includes content of all types (WC.document, WC.wiki).</td>
</tr>
<tr>
<td><code>getContentIDsExclude</code></td>
<td>Extract content IDs from input objects.</td>
</tr>
<tr>
<td><code>getContentIDsFiltered</code></td>
<td>Extract content IDs from input objects.</td>
</tr>
<tr>
<td><code>atomAsCMISObjects</code></td>
<td>ATOM feed as a List&lt;CMISObject&gt;.</td>
</tr>
<tr>
<td><code>atomAsEntries</code></td>
<td>ATOM feed as List&lt;org.apache.abdera.model.Entry&gt;</td>
</tr>
<tr>
<td><code>atomAsFeed</code></td>
<td>ATOM feed as org.apache.abdera.model.Feed</td>
</tr>
<tr>
<td><code>atomAsStreamUrls</code></td>
<td>ATOM feed as List&lt;String&gt; of URLs</td>
</tr>
<tr>
<td><code>getCMISQueryForDocIDs</code></td>
<td>Construct a CMIS query in the form of ‘IN’ query syntax that will retrieve the documents for the array of input doc IDs.</td>
</tr>
<tr>
<td><code>getCMISQueryForDocIDsFromFullID</code></td>
<td>Construct a CMIS query in the form of ‘IN’ query syntax that will retrieve the documents for the array of input doc IDs.</td>
</tr>
</tbody>
</table>
atomAsCMISObjects

Because CMISObject is not marshalable, use this method in your Java client code to convert the ATOM query response to List<CMISObject>.

Arguments

atomFeed
(java.lang.String) A Collection Object.

cutoffScore
(float) The cutoff score.

Expression Language Format

${cmisfunction:atomAsCMISObjects(atomFeed)}$

Returns

java.util.List

Usage Notes

None.
atomAsEntries

Because org.apache.abdera.model.Entry (an attribute on an Abdera feed) is not marshalable, use this method in your Java client code to convert the ATOM query response to List<org.apache.abdera.model.Entry>.

**Arguments**

**atomFeed**
(String)

**Expression Language Format**

`{cmisfunction:atomAsEntries(atomFeed)}`

**Returns**

`java.util.List`

**Usage Notes**

None.
atomAsFeed

Because org.apache.abdera.model.Feed is not marshalable, use this method in your Java client code to convert the ATOM query response to List<org.apache.abdera.model.Feed>.

Arguments

atomFeed
(String)

Expression Language Format

${cmisfunction:atomAsFeed(atomFeed)}

Returns

java.util.List

Usage Notes

None.
atomAsStreamUrls

Returns an ATOM feed as List<String> of URLs.

Arguments

atomFeed
(String)

Expression Language Format

${agfunction:getContentIDs(argResults)}

Returns

java.util.List

Usage Notes
None.
getCMISQueryForDocIDs

Construct a CMIS query in the form of 'IN' query syntax that will retrieve the documents for the array of input doc IDs.

**Arguments**

**repository**
(String)

**ids**
(java.util.List)

**Expression Language Format**

$\{agfunction:getContentIDsExclude(agResults,excludeClassURN)\}$

**Returns**

String

**Usage Notes**

None.
getCMISQueryForDocIDsFromFullID

Construct a CMIS query in the form of 'IN' query syntax that will retrieve the documents for the array of input doc IDs.

This method converts a list of IDs (perhaps coming from the Activity Graph Provider) to a CMIS query string to retrieve multiple content items. The ID is in the form of a WebCenter ID, such as 'my-ucm11g#dDocName:MOUNTAINS'.

**Arguments**

*ids*  
(java.util.List)

**Expression Language Format**

```
${cmisfunction:getCMISQueryForDocIDsFromFullID(ids)}
```

**Returns**

String

**Usage Notes**

None.
This chapter describes the Service Access Layer's property service, which provides a simple, scalable way for developers to access data and make it available to client applications.

This chapter includes the following topics:

- Section 70.1, "Introduction to the Property Service"
- Section 70.2, "Creating a Property Set Definition in JDeveloper"
- Section 70.3, "Deploying the Property Set Definition"
- Section 70.4, "Testing the Property Set Definition Deployment"
- Section 70.5, "Using the Property Service Data Provider"
- Section 70.6, "Using the Property Service Caching Service"
- Section 70.7, "Implementing a Custom Locator"

70.1 Introduction to the Property Service

The *property service* is a mechanism for retrieving and updating data through REST or Java APIs. The data can be stored in any data source, such as relational databases, Oracle Metadata Service (MDS) repositories, and LDAP repositories. Unlike a data provider, which fetches data into the context of a scenario (Section 67.1, "Introduction to Data Providers"), the property service can be used to fetch and update data outside the context of a scenario. In addition, the property service lets you create organized representations of data called *property sets*, each uniquely associated with a namespace. To find a property set, you simply look in its namespace using a REST request or Java method call. For information on scenarios, see Section 66.1.1, "Personalization Architecture."

---

**Note:** As a convenience, WebCenter Portal provides a property service data provider that allows you to fetch data from a property set into the context of a scenario. For more information, see Section 70.5, "Using the Property Service Data Provider."

---

This section provides conceptual information about the property service:

- Section 70.1.1, "What Is the Property Service Used For?"
- Section 70.1.2, "What Are Property Sets?"
- Section 70.1.3, "What Is a Locator?"
70.1.1 What Is the Property Service Used For?

Many use cases for the property service and its features exist. An important use case is the ability to aggregate data from multiple sources and make the data available to scenarios. As you will see, the property service is integrated with the Conductor and scenarios through a data provider. The data provider provides access to data supplied from the property service inside the context of a scenario. For example, you could create a user profile that consists of data from several sources and then use that profile in multiple scenarios.

70.1.2 What Are Property Sets?

A property set is simply a collection of related properties. For example, a property set might represent a collection of books and include properties like ISBN number, title, author, number of pages, and so on.

Each property set has a schema, called a property set definition. This definition specifies zero or more properties (members of the set), which can have names, descriptions, and data types. Think of a property set as an instantiation of a property set definition.

Figure 70–1 shows a property set definition in the visual editor in JDeveloper. Let’s look at the parts of a property set definition one by one.

![Modeling a Property Set Definition in JDeveloper](image)

First, the property set definition is modeled as a hierarchy. The visual editor in JDeveloper makes property set definition creation easy. First, at the top of the hierarchy, a namespace element is defined. All property sets exist in the context of a namespace. You can use the namespace name to find your property set definition with REST requests or Java method calls. In Figure 70–1, the namespace is called
myProperties.

Next, the property set definition itself is created and named. In our example, the property set definition is called **Employee**. The next step is to populate the Employee property set definition with property mappings. Each property mapping has a name, a definition (like a data type), and a locator class associated with it. **Figure 70–2** shows the dialog used to create a new property mapping, and the dropdown menu that lets you select a definition for the property mapping. Definitions include String, StringArray, Number, Int, DateTime, and several others. For example, a "name" property mapping might have a definition type of String. As you can see, property sets are particularly useful for modeling well-structured data such as user profiles.

![Figure 70–2 Creating a Property Mapping](image)

### 70.1.3 What Is a Locator?

Each property set is associated with a Java component called a **locator**. A locator is a custom implementation of the ILocator class, and includes methods for retrieving, filtering, and updating property set data. In this sense, a locator is similar to a data provider (**Section 67.1, "Introduction to Data Providers"**). You can use the REST API (or Java API) to retrieve data directly to a client application, like a mobile application or a portal.

**Note:** You can attach a locator class at the namespace level, the property set definition level, or at the individual property level. Locators are resolved in a specific order of precedence. Locators attached to individual properties take precedence over a property set definition-level locator, and so on. If no custom locator is specified, the default locator is used.

A default property set locator is provided with WebCenter Portal 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. If the default locator is insufficient for your use case, you can develop a custom locator. See also **Section 70.7, "Implementing a Custom Locator."**
70.1.4 Caching for the Property Service

The property service also includes an integrated caching service and an interface for writing data validation code. See Section 70.6, "Using the Property Service Caching Service."

70.1.5 Using the Property Service with Scenarios

If you are not familiar with concepts like the Conductor and scenarios, refer to Chapter 66, "Personalizing Oracle WebCenter Portal Applications."

It is important to note that the property service is not directly tied to the Conductor. This means you can use the property service to retrieve and update data outside the context of a scenario. However, WebCenter Portal provides a property service data provider that lets you retrieve data from the property service into a scenario, where you can further filter and process the data, batch it with data from other scenarios, and take advantage of other data integration features, like business objects.

70.1.6 Architectural Overview

Figure 70–3 shows the overall architecture of the property service, as described in the previous sections.

Figure 70–3  Property Service Architecture

70.2 Creating a Property Set Definition in JDeveloper

This section explains how to create a new property set definition.
Creating a Property Set Definition in JDeveloper

To create a property set definition in JDeveloper, follow these steps:

1. Be sure the data integration components are included in your project classpaths. See Section 2.5, "Setting Up JDeveloper for Personalization."

2. In the Application navigator, right-click your project and select New.

3. In the New Gallery dialog, under the General category, select Personalization.

4. In the Items part of the dialog, select Properties Namespace and click OK.

5. In the Create Properties Namespace dialog, give the namespace a name.

6. The property service editor opens in design view. Right-click the Property Namespace Definitions node and select Create New Property Set Definition, as shown in Figure 70–4.

Figure 70–4  Creating a New Property Set Definition

![Creating a New Property Set Definition](image)

7. In the Create New Property Set Definition dialog, enter a name and click OK.

8. Right-click the new property set definition node and select Create New Property Set Definition Mapping.

Note: Generally, it's best to create clearly defined property definitions with one or more property mappings. However, a zero-element property set definition permits property sets to be created dynamically, and may be useful in some cases.

Note: Property set definitions created in JDeveloper are imported into MDS during deployment. All property sets created at run-time that are based on those property set definitions are stored in the default database (or other persistence storage as configured through custom locators).

Note: This dialog lets you define individual property mappings to include in the property set definition. Note that each property mapping can have a Locator Class. The default locator class looks in the default database for property set data. To obtain data from another source, you need to implement a custom locator. For more information, see Section 70.7, "Implementing a Custom Locator."
9. In the Create/Edit Property Mapping dialog, give the property a name. Pick a Definition (like a data type) for the property. For example, a name might be type StringDef and an employee ID might be an IntDef (integer). You can also specify a Locator Class for the property. If you don’t specify a Locator Class, the property uses the default Locator. See Section 70.1.3, "What Is a Locator?.”

10. To create more properties, right-click the Property Set Definition node and follow these same steps.

11. Save your work.

The property set definition must next be deployed to the server, where it can be used to instantiate new property sets.

### 70.3 Deploying the Property Set Definition

After you create a property set definition, you need to deploy it to a server. You can do this from JDeveloper, and deploy to either the Integrated WebLogic Server or to an external managed server. After you deploy the property set definition, you can access it through REST APIs, Java APIs, or in a scenario through the property service data provider.

In this example, we will deploy to the Integrated WebLogic Server instance.

1. Create one or more property set definitions, as explained in Section 70.2, "Creating a Property Set Definition in JDeveloper.”

2. Decide where you intend to deploy to. The target server must be running.

3. From the application menu, select Deploy Personalization Files.

4. Select the deployment target:
   - **To Last Selected Server** – Deploys to the last selected server.
   - **To Server** – This option lets you select the server to deploy to.
   - **To Filesystem** – Deploys a Metadata Archive (MAR) file to the filesystem. This file can later be deployed to the server at a later time.

5. For this example, select **To Server**.

6. In the Deploy to Server dialog, select **IntegratedWebLogicServer** and **DefaultServer** as the server instance.
7. Click OK. JDeveloper deploys the set definition files to the server. Technically, they are placed in a Metadata Archive (MAR) file, which is then deployed to the server. You can view the results of the deployment in the Deployment Log window, as shown in Figure 70–6.

Now that the property set definition is deployed, you can quickly test the deployment.

### 70.4 Testing the Property Set Definition Deployment

This section explores two ways that you can test the deployment of a property set definition:

- Section 70.4.1, "Locating the Property Set Definition Using REST"
- Section 70.4.2, "Locating the Property Set Definition Using the Data Provider"

#### 70.4.1 Locating the Property Set Definition Using REST

One way to locate the property sets available to you is to use a tool that lets you send REST requests and view responses. Many such tools and browser plugins exist. For information about using REST APIs, see Chapter 53, "Using Oracle WebCenter Portal REST APIs."

1. In a REST client tool, enter this URL, where host and port are the host and port name where the WCPS server is running.
Using the Property Service Data Provider

http://host:port/wcps/api/property/resourceIndex

2. Locate the namespace URI and enter its URL in the browser. For example:
   http://host:port/wcps/api/conductor/namespaces

   For example: http://myhost:7101/wcps/api/conductor/namespaces

3. Look in the response for the property set namespace(s) you created. For example, if you created a property set namespace called Employee_Namespace (see Section 70.2, "Creating a Property Set Definition in JDeveloper"), the response will look similar to the response shown in Figure 70–7.

![Figure 70–7  REST Response Showing the Deployed Namespace Employee_Namespace](image)

70.4.2 Locating the Property Set Definition Using the Data Provider

You can access deployed property set definitions in a scenario through the property service data provider. For details, see Section 70.5, "Using the Property Service Data Provider."

70.5 Using the Property Service Data Provider

The property service data provider (oracle.PropertiesServiceProvider) allows you to integrate property service data into a scenario. This provider lets you:

- retrieve property sets for a specified property set definition
- retrieve a property set by name
- return a specific property from a property set

70.5.1 Accessing the Property Service Data Provider in a Scenario

You use the property service data provider like any other provider, by selecting the Invoke Data Provider function in a scenario:

1. In the scenario editor, create a new scenario. For information on using the scenario editor, see Chapter 66.2.2, “Authoring Personalized Scenarios in JDeveloper.”

2. If asked, create a URL Connection to the server. Enter the following information in the URL Connection dialog.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Conductor</td>
</tr>
<tr>
<td>URL Endpoint</td>
<td><a href="http://host:port/wcps/api/conductor/resourceIndex">http://host:port/wcps/api/conductor/resourceIndex</a> for example: <a href="http://example.com:7101/wcps/api/conductor/resourceIndex">http://example.com:7101/wcps/api/conductor/resourceIndex</a></td>
</tr>
<tr>
<td>Authentication Type</td>
<td>Basic</td>
</tr>
</tbody>
</table>
3. Click the Start node and select **Add Following Statement** and then **Invoke Data Provider**.

4. Click the Invoke Provider node and select **Invoke Provider Properties**.

5. In the Invoke Provider dialog, open the `oracle.PropertiesServiceProvider` node, then open LocalServerConnection and finally open GetPropertiesResource nodes. See Figure 70–8.

**Figure 70–8 Exposing the Property Service Data Provider Methods**

Select a method to execute and enter parameters at the bottom of the dialog. The parameters let you specify with the property set or property to retrieve. For details on the methods, see Section 70.5.2, "Property Service Data Provider Method Reference."

### 70.5.2 Property Service Data Provider Method Reference

This section describes each of the property service data provider methods. These methods let you gain access to property sets and their properties in a specified namespace:

- Section 70.5.2.1, "getProperty(definition, set, property)"
- Section 70.5.2.2, "getProperty(namespace, definition, set, property)"

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Username</td>
<td>weblogic</td>
</tr>
<tr>
<td>Password</td>
<td>weblogic1</td>
</tr>
<tr>
<td>Realm</td>
<td>WCPS</td>
</tr>
</tbody>
</table>
70.5.2.1  getProperty(definition, set, property)
Retrieves a property from the current, default namespace.

Parameters:
- definition – The name of a property set definition that exists in the default namespace.
- set – The name of an instance of the property set definition.
- property – The name of the property you wish to retrieve.

Returns:
An object of the property’s type. Returns null if no property is found.

Example:
getProperty("EmployeeSetDef", "kjones", "address")

70.5.2.2  getProperty(namespace, definition, set, property)
Retrieves a property from the specified namespace.

Parameters:
- namespace – The name of a namespace.
- definition – The name of a property set definition that exists in the default namespace.
- set – The name of an instance of the property set definition.
- property – The name of the property you wish to retrieve.

Returns:
An object of the property’s type.

Example:
getProperty("NS1", "EmployeeSetDef", "kjones", "address")

70.5.2.3  getPropertySet(definition, set)
Retrieves a property set in the current or default namespace.

Parameters:
- definition – The name of a property set definition that exists in the default namespace.
- set – The name of an instance of the property set definition.

Returns:
A Map object containing the property set.

Example:
getPropertySet("EmployeeSetDef", "kjones")
70.5.2.4 getPropertySet(namespace, definition, set)
Retrieves a property set in the specified namespace.

Parameters:
- namespace – The name of a namespace.
- definition – The name of a property set definition that exists in the default namespace.
- set – The name of an instance of the property set definition.

Returns:
A Map object containing the property set.

Example:
getPropertySet("NS1", "EmployeeSetDef", "kjones")

70.5.2.5 getPropertySets(definition, search, restrictions)
Retrieves a list of property sets that are instances of the specified property set definition. The list can be managed with optional search and restrictions parameters. The method looks in the current or default namespace.

Parameters:
- definition – The name of a property set definition that exists in the default namespace.
- search – (Optional) Apply the provided search expression (a string) to retrieve the specified property sets. If null, all property sets for the provided property sets will be retrieved (within the pagination context).
- restrictions – (Optional) A List object specifying a list of properties. The method will only retrieve the provided in this list. If null, all properties are retrieved.

Returns:
An object of the property’s type.

Example:
getProperty("EmployeeSetDef", "tmyProp:equals:someValue")
Other search string examples:
myFavoriteHobbies:contains:camping
tps-created-on:gt:1980-01-01T00:00:00.000-070

70.5.2.6 getPropertySets(namespace, definition, search, restrictions)
Retrieves a list of property sets that are instances of the specified property set definition. The list can be managed with optional search and restrictions parameters. The method looks in the specified namespace only.

Parameters:
- namespace – The name of a namespace.
- definition – The name of a property set definition that exists in the default namespace.
search – (Optional) Apply the provided search expression to retrieve the specified property sets. If null, all property sets for the provided property sets will be retrieved (within the pagination context).

restrictions – (Optional) A List object specifying a list of properties. The method will only retrieve the provided in this list. If null, all properties are retrieved.

Returns:
An object of the property’s type.

Example:
getProperty("NS1", "EmployeeSetDef", "tmyProp:equals:someValue")

Other search string examples:
myFavoriteHobbies:contains:camping
tps-created-on:gt:1980-01-01T00:00:00.000-070

70.6 Using the Property Service Caching Service

The property service includes a built-in caching layer that automatically caches property sets when they are returned. The property service uses the Oracle Coherence, an in-memory data grid caching solution that provides fast access to frequently used data. For detailed information on configuring Oracle Coherence, see the Oracle Coherence Documentation.

The property service cache configuration file is located in:
$ORACLE_HOME/user_projects/applications/<domain>/conductor-extensions-library/WEB-INF/classes.

You are free to modify the default cache configuration file, or add other cache configuration files if desired. All cache factories instantiated by the property service are specific to the property service classloader. The cache configuration file is located in:
$ORACLE_HOME/user_projects/applications/<domain>/conductor-extensions-library/WEB-INF/classes.

The configurations can be changed to suit the end-user needs, but the names of the caches must remain unchanged.

70.7 Implementing a Custom Locator

Each property set is associated with a Java component called a locator. A locator is a custom implementation of the ILocator class, and includes methods for retrieving, filtering, and updating property set data. If the default locator is insufficient for your use case, you can develop a custom locator.

This section describes how to implement a simple custom locator, configure it, and invoke it. Even though this is a basic example, it illustrates many important aspects of custom locator development and use.

For a general introduction to locators, see Section 70.1.3, "What Is a Locator?.”

- Section 70.7.1, "Overview of the Example and Prerequisites"
- Section 70.7.2, "Creating a Property Set Definition"
- Section 70.7.3, "Making Up Data: Implement a Simple Library Class"
Implementing a Custom Locator

Section 70.7.4, "Backing the Data with a JavaBean"
Section 70.7.5, "Implementing a Java Data Access Object"
Section 70.7.6, "Implementing the ILocator Class"
Section 70.7.7, "Updating the PropertySetDefinition with the Custom Locator"
Section 70.7.8, "Compiling and Deploying the Example"
Section 70.7.9, "Creating a Scenario to Use the Locator"
Section 70.7.10, "Viewing the Scenario XML Source"

70.7.1 Overview of the Example and Prerequisites

Before you begin developing a custom locator in JDeveloper, be sure you have things set up properly. There are a few configurations that you need to make or verify for your environment. For details, see Section 2.5, "Setting Up JDeveloper for Personalization."

Note: In this example, the locator is created in an Fusion Web Application in JDeveloper. You can also create function providers in WebCenter Portal applications or any other type of JDeveloper application that includes Java capabilities.

Our example describes how to create a locator that accesses book-related data, like author, title, and IBSN number. Here, the raw data is simply hardcoded into a Java class, but the locator could be written similarly to access data in a database.

We also illustrate using the Data Access Object (DAO) pattern to delegate data access from the locator, a commonly used approach. Whether the data is a library of books, a user database, or an inventory system, the same pattern applies.

The main steps include:

- Define a Book property set definition.
- Let the property service know about the Book property set definition.
- Implement the IPropertyLocator interface to load properties from a Book object.

70.7.2 Creating a Property Set Definition

Following the general instructions in Section 70.2, "Creating a Property Set Definition in JDeveloper," create a new property set definition with these parts:

- Call the namespace book.property.locator.
- Call the property set definition BookPropertySetDefinition.
- Add three property mappings of type StringDef called title, author, and ISBN.

Figure 70–9 shows this simple definition in the visual editor in JDeveloper.
For now, we will use the default locator class. Later, we will change the default to use our new custom locator implementation.

### 70.7.3 Making Up Data: Implement a Simple Library Class

The class in Example 70–1 simply creates a Map from hardcoded, static book data. The book’s titles are used as the Map keys. In a real use case, data would come from an external data source, like a library database.

**Example 70–1  A Class to Create Book Data**

```java
package locator;

import java.util.HashMap;
import java.util.Map;

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);
    }
}
```

### 70.7.4 Backing the Data with a JavaBean

The JavaBean class in Example 70–2 represents the book attributes hardcoded into our Library class: title, author, and ISBN number. To simplify the example code, we add a
implementing a custom locator
using the property service

getProperty(String name) method. In practice, JavaBean introspection is a better practice.

Example 70–2  A Simple JavaBean Representing Book Data

package locator;

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) {
        if (propName.equals("ISBN")) {
            return getISBN();
        }
        if (propName.equals("title")) {
            return getTitle();
        }
        if (propName.equals("author")) {
            return getAuthor();
        }
        return null;
    }
}
70.7.5 Implementing a Java Data Access Object

The DAO class is more interesting than the previous classes. Here, the DAO does the hard work of fetching data from a source. In this case, the source is our Library object, but it could be a database, REST service, or other source.

Example 70–3  A Data Access Object Implementation

```java
package model;

import java.util.ArrayList;
import java.util.List;
import java.util.Map;
import oracle.wcps.property.IContext;
import oracle.wcps.property.IProperty;
import oracle.wcps.property.IPropertyDefinition;
import oracle.wcps.property.IPropertyDefinitionName;
import oracle.wcps.property.IPropertyMapping;
import oracle.wcps.property.IPropertyName;
import oracle.wcps.property.IPropertyService;
import oracle.wcps.property.IPropertySet;
import oracle.wcps.property.IPropertySetDefinition;
import oracle.wcps.property.ServiceFactory;
import oracle.wcps.property.Type;
import oracle.wcps.property.model.Property;

class BookPropertyDAO {
    IPropertyService service;

    // Generate the library that will provide our backing store.
    static
    {
        Library.generateLibrary();
    }

    /**
     * Assume the name of the PropertySet is the title of the book.
     */
    void loadProperties(IPropertySet propertySet,
                        IPropertySetDefinition iPropertySetDefinition,
                        List<IPropertyName> propertyNames, IContext context) throws Exception
    {
        String title = propertySet.getName().getName();
        Book book = Library.getBook(title);
        loadProperties(book, propertySet, iPropertySetDefinition,
                        propertyNames, context);
    }

    private void loadProperties(Book book, IPropertySet propertySet,
                                 IPropertySetDefinition iPropertySetDefinition,
                                 List<IPropertyName> propertyNames, IContext context) throws Exception
    {
        Map<IPropertyName, IPropertyMapping> mappings =
            iPropertySetDefinition.getPropertyMappings();

        // Null propertyNames is valid, and indicates to get all properties
        if (propertyNames==null)
        {
            propertyNames = new ArrayList<IPropertyName>();
        }
```
Implementing a Custom Locator

Using the Property Service

```java
propertyNames.addAll(mappings.keySet());

// Loop over all property names, create a Property value, and add it to the Property set
for (IPropertyName pname : propertyNames)
{
    String value = book.getProperty(pname.getName());
    if (value != null)
    {
        IPropertyMapping mapping = mappings.get(pname);
        addProperty(value, pname, propertySet, mapping, context);
    }
}

// Add typed properties according to the information in our mappings
private void addProperty(String value, IPropertyName pname,
                        IPropertySet propertySet, IPropertyMapping mapping, IContext context)
{
    // The property mappings could be useful if we wanted to get the individual
    // PropertyDefinitions and inspect their type.
    IPropertyDefinitionName pdefName = mapping.getPropertyDefinitionName();

    // We need the namespace to look up our property definitions
    if (service==null)
    {
        String ns = propertySet.getNamespaceName().getName();
        System.err.println("Using namespace: " + ns);
        service = ServiceFactory.getPropertyService(IContext.NULL, ns);
        service = ServiceFactory.getPropertyService(context, ns);
    }

    IPropertyDefinition pdef =
        service.getPropertyDefinition(context, pdefName);
    //service.getPropertyDefinition(IContext.NULL, pdefName);

    Type type = pdef.getType();
    IProperty ps = Property.builder().withProperty(pname, value, type).build();
    System.err.println("Adding " + ps.getName() + " of type " + ps.getType());
    propertySet.putProperty(pname, ps);
}
```

### 70.7.6 Implementing the ILocator Class

Finally, Example 70–4 lists the locator class itself. The first thing to notice is that we are implementing ILocator. The interface includes several methods that enable a locator to do full CRUD (create, read, update, delete) operations. However, our example only implement the loadProperties(), which corresponds to the "read" operation. For more information on this interface, you can refer to the WebCenter Portal API Reference (Javadoc).

Another important point is that the loadProperties() method delegates to the DAO. The DAO class has an identical method called loadProperties() that does the work of fetching data from the Library data store.
**Example 70–4  Example Locator Class**

```java
package locator;

import java.util.List;
import java.util.Map;
import oracle.wcps.property.IContext;
import oracle.wcps.property.INamespaceName;
import oracle.wcps.property.IPagedList;
import oracle.wcps.property.IPropertyLocator;
import oracle.wcps.property.IPropertyMapping;
import oracle.wcps.property.IPropertyName;
import oracle.wcps.property.IPropertySet;
import oracle.wcps.property.IPropertySetDefinition;
import oracle.wcps.property.IPropertySetDefinitionName;
import oracle.wcps.property.filter.PropertySetFilterContext;
import oracle.wcps.property.filter.expression.PropertySetExpression;

public class BookPropertyLocator implements IPropertyLocator
{
    private BookPropertyDAO dao;

    // Must have a public default constructor, since the Property Service instantiates
    // this

    public BookPropertyLocator()
    {
        dao = new BookPropertyDAO();
    }

    /**
     * Load property values, optionally returning only a subset of those.
     * @param context the IContext containing parameters passed in from the PropertyService
     * @param iPropertySet The PropertySet into which the retrieved values will be loaded
     * @param iPropertySetDefinition The PropertySetDefinition defining the 'shape' of the
     * properties
     * @param propertyNames List of PropertyNames to load. If null, all will be loaded.
     */
    public void loadProperties(IContext context, IPropertySet iPropertySet,
                                IPropertySetDefinition iPropertySetDefinition,
                                List<IPropertyName> propertyNames)
    {
        try
        {
            // Delegate to BookProperDAO
            dao.loadProperties(iPropertySet, iPropertySetDefinition, propertyNames, context);
        }
        catch (Exception e) {
            throw new RuntimeException("Cannot load properties ", e);
        }
    }

    /**
     * Filter properties; load/return those properties that only match the filter criteria.
     * @param context the IContext containing parameters passed in from the PropertyService
     * @param iNamespaceName
     * @param iPropertySetDefinitionName
     * @param propertySetFilterContext
     */
```
Let’s look at the loadProperties() method of the ILocator interface in a little more detail. This method takes these parameter types:

- **IContext** – The IContext object is the vehicle for authentication. An object of this type is passed in from the property service. It includes data like the username and password credentials of the logged in user. For example, you could retrieve the user name and password like this:

  ```java
  String u = context.getProperty(IContext.USERNAME_KEY);
  String p = context.getProperty(IContext.PASSWORD_KEY);
  ```

  You can also retrieve the request object and use it to obtain the credentials, or other request information:

  ```java
  HttpServletRequest request = context.getProperty(IContext.HTTP_REQUEST_KEY)
  ```

- **IPropertySet** – The retrieved property set values are loaded into this object.

- **IPropertySetDefinition** – The object that defines the "shape" of the data. This shape is based on the PropertySetDefinition you created previously, in Section 70.7.2, "Creating a Property Set Definition."
- List<IPROPERTYNAME> – This object provides a list of PropertyName objects to load. If null, all PropertyName objects are loaded.

Now that the example classes are finished, we can compile, deploy, and use the locator.

### 70.7.7 Updating the PropertySetDefinition with the Custom Locator

When we created the property set definition previously, we used the default Locator. Now that we have implemented a custom locator, we need to add it to the definition.

1. Locate the `ns_book.property.locator.xml` file in the Resources folder of your project.
2. Right-click the Locators node in the design view and select **Edit Locator Info**.
3. Click the **Add** icon (a plus symbol) and enter BookPropertyLocator in the New Locator dialog, as shown in Figure 70–10, and click **OK**.

![Figure 70–10 Adding the Custom Locator](image)

4. In the Locator Map Configuration dialog, add the Unassigned Properties to the Locator Assigned Properties list. When you are finished, author, title, and ISBN should be listed as assigned properties.
5. Click **OK**.
6. In the design view, you will see the custom locator now appears in the Locator Info node.
7. Save your work.

### 70.7.8 Compiling and Deploying the Example

Before deploying, compile your project by whatever method is customary to you. For example, in JDeveloper, you can select **Make** from the Build menu.
To deploy your property namespace definition to the server, follow the instruction in Section 70.3, "Deploying the Property Set Definition."

You also need to deploy your custom locator classes to the server, follow the steps in:

- Section 70.7.8.1, "Setting Up the JAR File Structure"
- Section 70.7.8.2, "Create a JAR File"
- Section 70.7.8.3, "Copy the JAR File to the Server"

### 70.7.8.1 Setting Up the JAR File Structure

If you are unfamiliar with JAR file creation in JDeveloper, this section explains how to set up your JAR file structure (a deployment profile) to ensure that all the required elements are included.

1. Right-click your project (in this example, the project called Model) in the Application Navigator and select **Project Properties**.
2. In the Project Properties dialog, select **Deployment**.
3. Click **New**.
4. Select JAR File as the Archive Type and give the JAR a name.
5. Click **OK**.
6. In the JAR Options dialog, select **Contributors**.
7. Be sure that Project Output Directory, Project Source Path, and Project Dependencies are selected, as shown in **Figure 70–11**.

**Figure 70–11  Selecting Project Output Contributors**

8. Select **Filters**.
9. Select the **Patterns** tab.
10. In the Patterns tab, add the following file patterns to include and exclude specific types of files from the JAR.

Figure 70–12 shows these patterns after they have been added to the list.

- Include **.class
- Exclude **.java
- Exclude **.txt
- Exclude **.html

Figure 70–12 Selecting JAR File Contents

11. Click OK, and then OK in the Project Properties dialog.

12. Save your project.

70.7.8.2 Create a JAR File

After the JAR deployment profile is set, you are ready to generate a JAR file from your project.

1. Right-click the project (in this case, the Model project) in the Application Navigator and select Deploy.

2. Select the JAR file you want to deploy. This is the name you gave the file in Section 70.7.8.1, "Setting Up the JAR File Structure."

3. In the Deployment Action dialog, select Deploy to JAR file and click Finish.

The JAR file is placed in your project directory. You can view it on the file system in: <Workspace_Directory>/Model/deploy. For example, if your Workspace is called MyProviders, you might find the JAR here: C:\JDeveloper\mywork\MyProviders\Model\deploy.
### 70.7.8.3 Copy the JAR File to the Server
You must copy the JAR file to the server.

1. Copy the JAR file that you created in the previous step. As noted previously, the file is located in `<Workspace_Directory\Model\deploy`.

2. Paste the JAR file into the DefaultDomain of your server, here:
   
   $DOMAIN_HOME/locator-extensions-library/WEB-INF/lib

   For example:

   C:\JDeveloper\system11.1.7.40.63.82\DefaultDomain\locator-extensions-library\WEB-INF\lib

3. Restart the server.

### 70.7.9 Creating a Scenario to Use the Locator
Follow these basic steps to create a scenario. For more information on creating scenarios, see Section 66.2.2, "Authoring Personalized Scenarios in JDeveloper."

1. Create a new scenario called BookPropertyScenario.
2. Add an Invoke Provider node to the scenario.
3. Edit the provider properties.
4. In the list of known providers, open the oracle.PropertiesServiceProvider node and drill down to the provider methods.
5. Select the GetProperty(namespace, definition, set, property) method.
6. Add the following parameter values:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>namespace</td>
<td>book.property.locator</td>
</tr>
<tr>
<td>definition</td>
<td>BookPropertySetDefinition</td>
</tr>
<tr>
<td>set</td>
<td>Into Thin Air</td>
</tr>
<tr>
<td>property</td>
<td>author</td>
</tr>
</tbody>
</table>

7. Remember to specify a return variable in the Edit Provider Properties dialog. For this example, call the variable `property`.
8. Create a return statement to return the property value. Remember, you must use expression language (EL) format for the return variable: `${property}`.
9. Save the scenario and run it.

### 70.7.10 Viewing the Scenario XML Source
The XML source code for the scenario created in the previous section looks like this:

```xml
<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>
  </call-provider>
</body>
```
```
<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>
</scenario:scenario>
```
This chapter describes the REST APIs for the property service.
This chapter includes the following topics:

- Section 71.1, "Introduction to Property Service REST APIs"
- Section 71.2, "Property Service REST APIs"

### 71.1 Introduction to Property Service REST APIs

This chapter provides examples of how you can use the property service’s REST APIs to perform create, read, update, and delete (CRUD) operations on property sets and properties.

- Creating a namespace
- Viewing and creating property set definitions
- Viewing and creating property definitions
- Viewing, creating, and executing scenarios
- Viewing data provider details

### 71.2 Property Service REST APIs

This section provides examples of how you can use the property service’s REST APIs to perform create, read, update, and delete (CRUD) operations on property sets and properties.

- 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 includes the following subsections:

- Section 71.2.1, "Property Service Resource Index"
- Section 71.2.2, "Namespace CRUD"
- Section 71.2.3, "Property Definition CRUD"
- Section 71.2.4, "Property Set Definition CRUD"
Section 71.2.5, "Property Set CRUD"

Section 71.2.6, "Property CRUD"

Section 71.2.7, "JSON Payload"

Section 71.2.8, "Exceptions"

Section 71.2.9, "Query Parameters"

Section 71.2.10, "Search Expressions"

71.2.1 Property Service Resource Index

The property service resource index's response contains links and templates to access other resources of the property service.

71.2.1.1 ResourceIndex

This section describes the request and response for accessing the property service's resource index.

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=t0FKMnYDSzjR1shhs0sFX20vnvTZLQY8vnxNLkplyfYPCwgf89y6!-1714998420;
pather=wcs; 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={startIndex}&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}"/>
  </links>
</resourceIndex>
<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;data
  Fields={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: F]exc1lmTFgxuer9Zq2JpsXXn12y_**
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie: JSESSIONID=WFGvMnyKP7nDBW1RQySnllbQwWLDjLbFJy2MQ1g6nFH2VQlgJ4!-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}/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"
    }, {
        "template": "http://localhost:8891/wcps/api/property/namespaces/{namespaceName}/propertydefinitions/{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}/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"
    }, {
        "template": "http://localhost:8891/wcps/api/property/namespaces/{namespaceName}/propertysetdefinitions/{setDefinitionName}",
        "resourceType": "urn:oracle:wcps:property:propertysetdefinition",
        "capabilities": "urn:oracle:restframework:read urn:oracle:restframework:delete"
    }
}
71.2.2 Namespace CRUD

71.2.2.1 Create Namespace
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. This section describes the request and response for creating a namespace using the property service.

CreateNamespace

XML Request:
POST http://localhost:8891/wcps/api/property/namespaces
X-Oracle-RF-Token: FC59cJknjJPhD0PwKsStfD0Xjm8_w**
Cookie: JSESSIONID=t0FKMnYDSjRlshhs0sFX20nvnT2LQY8vnxNLkplyfYPCw9ey6!-1714998420;Version=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: FC59cJhknJPoIzPwkStD0Dxjm8_ww
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">
  <name>Oracle</name>
  <createdOn>2010-10-07T17:31:16.437-0600</createdOn>
</namespace>

JSON
Request:
POST http://localhost:8891/wcps/api/property/namespaces
X-Oracle-RF-Token: F1xecllmTbgxer9ZqZJpsXSnL2y_ww
Cookie:
Property Service REST APIs

JSESSIONID=WFGvMnyKP7nDBW1KqynhhphbQvWLDjLSbFJj2MQ1g6nFH2VQ1gJ4!-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: FIxec1lmTFguxer9ZqZJpsXSnl2y_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",
  "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",
    "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",
  }, {
    "template" :
    "http://localhost:8891/wcps/api/property/namespaces/Oracle/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",
  }, {
    "template" :
(setDefinitionName)/propertysets?
startIndex=(startIndex)&dataFields={dataFields}&q=(q)&itemsPerPage={itemsPerPage}",
    "resourceType" : "urn:oracle:wcps:property:propertysets",
    "capabilities" : "urn:oracle:restframework:read"
} ]
71.2.2.2 Retrieve Namespace

RetrieveNamespace

XML

Request:
GET http://localhost:8891/wcps/api/property/namespaces/Oracle
X-Oracle-RF-Token: FC59cJkhnJp0izPwKsD0Dxjm8_w**
Cookie:
JSESSIONID=t0FKMnVDSzjRlxshs0eFX20nvnT2ZQY8vnxNLkplIfYPCwgf9y96T-1714998420;Versi
on=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: FC59cJkhnJp0izPwKsD0Dxjm8_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/propertysetdefin
itions" resourceType="urn:oracle:wcps:property:propertysetdefinitions"
      template="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdef
itions?startIndex={startIndex}&q={q}&amp;itemsPerPage={itemsPerPage}"/>
    <link
      capabilities="urn:oracle:restframework:create urn:oracle:restframework:read
urn:oracle:restframework:delete"
s" resourceType="urn:oracle:wcps:property:propertydefinitions"
      template="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefini
tions?startIndex={startIndex}&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
itions/{setDefinitionName}/propertysets?startIndex={startIndex}&amp;dataFields={
```
<name>Oracle</name>
<createdOn>2010-10-07T17:31:16.437-0600</createdOn>

<propertyLocatorClassName>com.oracle.custom.MyPropertyLocator</propertyLocatorClassName>
</namespace>

**JSON Request:**
GET http://localhost:8891/wcps/api/property/namespaces/Oracle
X-Oracle-RF-Token: F1xec1lmTFguxer9ZqZJpsXSn12y_w**
Cookie: 
JSESSIONID=WFGvMnykPK7nDBW1RQynhhphbQvWLdJLSbfJj2MQ1g6nPH2VQlQgJ4!=609606431;Version =1
Content-Type: application/json
Accept: application/json

**Response:**
Status Code: 200
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: F1xec1lmTFguxer9ZqZJpsXSn12y_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",
  "resourceType": "urn:oracle:wcps:property:namespace",
  "links": [
    {
      "resourceType": "urn:oracle:wcps:property:namespace",
      "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?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/propertydefinitions"
    }
  ]
}
Property Service REST APIs

71.2.2.3 Update Namespace

UpdateNamespace

XML

Request:
PUT http://localhost:8891/wcps/api/property/namespaces/Oracle
X-Oracle-RF-Token: FC59cJkhnJPolzPwKSfD0Dxjm8_w**
Cookie: JSESSIONID=t0FKMnYDSzjR1sShs0aFX20vnT2LQY8vmxNLkplyfYPCwWgfb9y6!-1714998420;Version=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>
  <propertyLocatorClassName>com.oracle.custom.MyPropertyLocator</propertyLocatorClassName>
</namespace>

Response:
Status Code: 200
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKSfD0Dxjm8_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: FIxec1lmTFguxer9ZqZJpsXSnl2y_w**
Cookie: JSESSIONID=WFGvMnyKP7nDBW1RQynhhphbQwWLDjLSeBFjJ2M016nFH2VQ1gJ4!-609606431;Version=1
Content-Type: application/json
Accept: application/json
If-Match: 1286501066698
71.2.2.4 Delete Namespace

XML

**Request:**
DELETE http://localhost:8891/wcps/api/property/namespaces/Oracle
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0DxjmB_w**
Cookie: JSESSIONID=t0FKMnYDSjRlshhs0sFX20vnT2LQY8vnxNLkp1fyYPCwGfb9y6!-1714998420;Version=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: FC59cJkhnJPolzPwKsStfD0DxjmB_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: FIxec1lmTFguxer9ZqZjpsXSn12y_w**
Cookie: JSESSIONID=WFGvMnyKF7nDBW1RQynhhphbQvWLDjLSbFJj2MQ1g6nFH2VQ1gJ4!-609606431;Version=1
Content-Type: application/json
Accept: application/json
If-Match: 1286501066808

**Response:**
Status Code: 204

```json
{
   "name": "Oracle",
   "propertyLocatorClassName": "com.oracle.custom.MyPropertyLocator",
   "createdOn": "2010-10-07T19:24:26.698-0600",
   "resourceType": "urn:oracle:wcps:property:namespace"
}
```
71.2.2.5 Retrieve all Namespaces

RetrieveAllNamespaces

XML

Request:
GET http://localhost:8891/wcps/api/property/namespaces

Response:
Status Code:200
<?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={startIndex}&q={q}&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&itemsPerPage=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:create urn:oracle:restframework:read
          urn:oracle:restframework:delete"
          href="http://localhost:8891/wcps/api/property/namespaces"
          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?startIndex=10&itemsPerPage=10"
          rel="next" resourceType="urn:oracle:wcps:property:namespace"/>
      </links>
    </namespace>
  </items>
</namespaces>
<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.propertysetdefinitions.MDSScenarioPropertyLocator</propertyLocatorClassName>
<definitionLocatorClassName>oracle.wcps.property.persistence.mds.MDSDefinitionsLocator</definitionLocatorClassName>
</namespace>
</nameresourceType="urn:oracle:wcps:property:namespace">
<link capabilities="urn:oracle:restframework:read urn:oracle:restframework:update urn:oracle:restframework:delete"
    href="http://localhost:8891/wcps/api/property/namespaces/Oracle_0/propertysetdefinitions"
    rel="self" resourceType="urn:oracle:wcps:property:namespaces"/>
<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?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
</link>
</links>
</namespace>
</nameresourceType="urn:oracle:wcps:property:namespaces">
<link capabilities="urn:oracle:restframework:read urn:oracle:restframework:update urn:oracle:restframework:delete"
    href="http://localhost:8891/wcps/api/property/namespaces/JDEV/propertysetdefinitions"
    rel="self" resourceType="urn:oracle:wcps:property:namespaces"/>
<link capabilities="urn:oracle:restframework:create urn:oracle:restframework:read urn:oracle:restframework:delete"
    href="http://localhost:8891/wcps/api/property/namespaces/JDEV/propertysetdefinitions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
</link>
</links>
</namespace>
</nameresourceType="urn:oracle:wcps:property:namespaces">
<link capabilities="urn:oracle:restframework:read urn:oracle:restframework:update urn:oracle:restframework:delete"
    href="http://localhost:8891/wcps/api/property/namespaces/Oracle_0/propertysetdefinitions"
    rel="self" resourceType="urn:oracle:wcps:property:namespaces"/>
<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?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
</link>
</links>
</namespace>
</nameresourceType="urn:oracle:wcps:property:namespaces">
<link capabilities="urn:oracle:restframework:read urn:oracle:restframework:update urn:oracle:restframework:delete"
    href="http://localhost:8891/wcps/api/property/namespaces/Oracle_0/propertysetdefinitions"
    rel="self" resourceType="urn:oracle:wcps:property:namespaces"/>
<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?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
</link>
</links>
</namespace>
</nameresourceType="urn:oracle:wcps:property:namespaces">
<link capabilities="urn:oracle:restframework:read urn:oracle:restframework:update urn:oracle:restframework:delete"
    href="http://localhost:8891/wcps/api/property/namespaces/JDEV/propertysetdefinitions"
    rel="self" resourceType="urn:oracle:wcps:property:namespaces"/>
<link capabilities="urn:oracle:restframework:create urn:oracle:restframework:read urn:oracle:restframework:delete"
    href="http://localhost:8891/wcps/api/property/namespaces/JDEV/propertysetdefinitions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
</link>
</links>
</namespace>
</nameresourceType="urn:oracle:wcps:property:namespaces">
<link capabilities="urn:oracle:restframework:read urn:oracle:restframework:update urn:oracle:restframework:delete"
    href="http://localhost:8891/wcps/api/property/namespaces/Oracle_0/propertysetdefinitions"
    rel="self" resourceType="urn:oracle:wcps:property:namespaces"/>
<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?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
</link>
</links>
</namespace>
</nameresourceType="urn:oracle:wcps:property:namespaces">
<link capabilities="urn:oracle:restframework:read urn:oracle:restframework:update urn:oracle:restframework:delete"
    href="http://localhost:8891/wcps/api/property/namespaces/Oracle_0/propertysetdefinitions"
    rel="self" resourceType="urn:oracle:wcps:property:namespaces"/>
<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?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
</link>
</links>
</namespace>
</nameresourceType="urn:oracle:wcps:property:namespaces">
<link capabilities="urn:oracle:restframework:read urn:oracle:restframework:update urn:oracle:restframework:delete"
    href="http://localhost:8891/wcps/api/property/namespaces/Oracle_0/propertysetdefinitions"
    rel="self" resourceType="urn:oracle:wcps:property:namespaces"/>
<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?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
</link>
</links>
</namespace>
</nameresourceType="urn:oracle:wcps:property:namespaces">
<namespace resourceType="urn:oracle:wcps:property:namespace">
  <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">
  <name>Oracle_5</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"
<link capabilities="urn:oracle:restframework:create urn:oracle:restframework:read urn:oracle:restframework:delete"
<link capabilities="urn:oracle:restframework:read urn:oracle:restframework:delete"
<link capabilities="urn:oracle:restframework:read urn:oracle:restframework:update urn:oracle:restframework:delete"
<link capabilities="urn:oracle:restframework:create urn:oracle:restframework:read urn:oracle:restframework:delete"
<link capabilities="urn:oracle:restframework:read urn:oracle:restframework:delete" href="http://localhost:8891/wcps/api/property/namespaces/Oracle_6/propertysetdefinitions/{setDefinitionName}/propertysets?startIndex={startIndex}&amp;dataFields={dataFields}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
</links>
</namespace>
urn:oracle:restframework:read urn:oracle:restframework:delete*
    resourceType="urn:oracle:wcps:property:propertydefinitions"
    template="http://localhost:8891/wcps/api/property/namespaces/Oracle_7/propertydefinitions?startIndex={startIndex}&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_7/propertysetdefinitions/{setDefinitionName}/propertysets?startIndex={startIndex}&amp;dataFields={dataFields}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>

    <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*
        resourceType="urn:oracle:wcps:property:propertysets"

        <link capabilities="urn:oracle:restframework:create*
        resourceType="urn:oracle:wcps:property:propertysetdefinitions"
        template="http://localhost:8891/wcps/api/property/namespaces/Oracle_8/propertysetdefinitions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>

        <link capabilities="urn:oracle:restframework:create*
        resourceType="urn:oracle:wcps:property:propertysetdefinitions"
        template="http://localhost:8891/wcps/api/property/namespaces/Oracle_8/propertysetdefinitions?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_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: FIxec11m7TFguxer9ZqZJpsXSn12y_w**

Cookie:
JSESSIONID=WFGvMnyKP7nDBW1KqynhhphbQvWLDjLSbFJj2MQl6nFH2VQlJ4!=609606431;Version=1
Content-Type: application/json
Accept: application/json

Response:
Status Code: 200
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnl2y_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",
    "propertyLocatorClassName" : "oracle.wcps.conductor.scenario.persistence.impl.properties.MDSScenarioPropertyLocator",
    "definitionLocatorClassName" : "oracle.wcps.property.persistence.mds.MDSDefinitionsLocator",
    "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",
      "rel" : "self"
    } ],
    "template" : "http://localhost:8891/wcps/api/property/namespaces/JDEV/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",
  } ]
}

Property Service REST APIs 71-19
"name": "Oracle_0",
"createdOn": "2010-10-07T19:24:26.907-0600",
"resourceType": "urn:oracle:wcps:property:namespace",
"links": [{
  "resourceType": "urn:oracle:wcps:property:namespace",
  "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"
},

"name": "Oracle_1",
"createdOn": "2010-10-07T19:24:26.954-0600",
"resourceType": "urn:oracle:wcps:property:namespace",
"links": [{
  "resourceType": "urn:oracle:wcps:property:namespace",
  "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"
Property Service REST APIs

- "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_1/propertydefinitions"
- "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_1/propertysetdefinitions?setDefinitionName={setDefinitionName}&startIndex={startIndex}&dataFields={dataFields}&q={q}&itemsPerPage={itemsPerPage}",
  "resourceType" : "urn:oracle:wcps:property:propertysetdefinitions",
  "capabilities" : "urn:oracle:restframework:read"
]
],

- "name" : 'Oracle_2',
  "createdOn" : '2010-10-07T19:24:27.034-0600",
  "resourceType" : "urn:oracle:wcps:property:namespace",
  "links" : [ {
    "resourceType" : "urn:oracle:wcps:property:namespace",
    "rel" : "self"
  }, {
    "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_2/propertysetdefinitions?setDefinitionName={setDefinitionName}&startIndex={startIndex}&dataFields={dataFields}&q={q}&itemsPerPage={itemsPerPage}",
    "resourceType" : "urn:oracle:wcps:property:propertysetdefinitions",
  }
],

- "name" : 'Oracle_3',
  "createdOn" : '2010-10-07T19:24:27.191-0600",
  "resourceType" : "urn:oracle:wcps:property:namespace",
  "links" : [ {
    "resourceType" : "urn:oracle:wcps:property:namespace",
    "rel" : "self"
  }, {
    "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_3/propertysetdefinitions?setDefinitionName={setDefinitionName}&startIndex={startIndex}&dataFields={dataFields}&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/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",
},

"name" : "Oracle_4",
"createdOn" : "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",
"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",
},

"name" : "Oracle_5",
"createdOn" : "2010-10-07T21:24:27.359-0600",
"resourceType" : "urn:oracle:wcps:property:namespace",
"links" : [{
"resourceType" : "urn:oracle:wcps:property:namespace",
"capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:delete urn:oracle:restframework:delete",
"rel" : "self"
}]
}
"resourceType": "urn:oracle:wcps:property:namespace",
"capabilities": "urn:oracle:restframework:read
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",
  "rel": "self"
}, {
  "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",
  "rel": "self"
}
],

{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': []
}]
],

'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:read
urn:oracle:restframework:delete",
'rel': "self"
}, {
'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:read
urn:oracle:restframework:delete",
'rel': "self"
}]
}]
)

'template': "http://localhost:8891/wcps/api/property/namespaces/Oracle_6/propertysetdefinitions/(setName)/propertysets?startIndex={startIndex}&dataFields={dataFields}&q={q}&itemsPerPage={itemsPerPage}",
'resourceType': "urn:oracle:wcps:property:propertyset definitions",
'capabilities': "urn:oracle:restframework:read
urn:oracle:restframework:delete",
'rel': "self"
}]
)
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",
  "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",
    "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",
  }, {
    "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",
  }, {
    "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",
  }, {
    "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",
}
'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'
}
]
)


71.2.2.6 Delete all Namespaces

DeleteAllNamespaces

XML

Request:
DELETE http://localhost:8891/wcps/api/property/namespaces
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0DxjmB_w**
Cookie:
JSESSIONID=t0FKMnYDszjlshhs0sFX20nnvTv2LQy8vnxNLkplyfVPCwgbf9y6i-1714998420;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: FC59cJkhnJPolzPwKsStfD0DxjmB_w**
71.2.3 Property Definition CRUD

71.2.3.1 Create Property Definition - Create Integer Property Definition

This section describes the request and response for creating a property definition within a namespace using the property service.

CreateIntPD

XML

Request:
POST http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=ZfhzMnYQz7pg4gzv5w2V1srvy3yDTsG4J4090X10HhrQpCSTkG2!-1714998420;Version =1
Content-Type: application/xml
Accept: application/xml

Response:
Status Code:201
Content-Length: 678

Property Service REST APIs

X-Oracle-RF-Token: FC59cJkhnJPnlzPwKsfD0Dxjm8_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/Integer_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"
      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: F1xec1lmTFguxhr92gZJpsXSn12y_w**
Cookie: JSESSIONID=GSw4MnyYTtw87Q6yCbb2H9F0JL6kLjwrqYvJb72LrtD9Qg0LtKPGc!-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: F1xec1lmTFguxhr92gZJpsXSn12y_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:
71.2.3.2 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: FC59cJkhnJPo1zPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=ZfhzMnYQz7pg4gsv5ZV1sryv3yDTsG4J4090X1GnHhrQpCSTkG2!=-1714998420;Version=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: FC59cJkhnIPolzPwKsStfDfDxjmB_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/Integer_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/propertydefinitions/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: FIxec1lmTFguxer9ZqZJpsXSnl2y_w**
Cookie:
JSESSIONID=GSw4MnyYTty87Q6yCBb2H9F0JL6kJLwjerqYv3b72LtDMQ6dLKPGRc!=609606431; Version
=1
Content-Type: application/json
Accept: application/json

{
  "name" : "Integer_Array_Def",
  "type" : "INT_ARRAY",
  "namespaceName" : "Oracle",
  "restricted" : true,
"restrictedValues" : [ {
  "concreteType" :
  "value" : 1
 }, {
  "concreteType" :
  "value" : 2
 }, {
  "concreteType" :
  "value" : 3
 }, {
  "concreteType" :
  "value" : 4
 }, {
  "concreteType" :
  "value" : 5
 }],
"defaultValues" : [ {
  "concreteType" :
  "value" : 1
 }, {
  "concreteType" :
  "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/Integer_Array_Def
Date: Fri, 08 Oct 2010 01:24:41 GMT
ETag: "1286501081123"

{  
  "name" : "Integer_Array_Def",
  "type" : "INT_ARRAY",
  "namespaceName" : "Oracle",
  "defaultValues" : [ {
    "concreteType" : 
    "value" : 1
  }, {
    "concreteType" : 
    "value" : 2
  }
}
71.2.3.3 Retrieve Property Definition

RetrievePD

XML

Request:
GET
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions/Integer_Def
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=ZfhzMnYQz7pg4gzv5wZv1srv3yD7w4J4090X1GhHhrQPpCSTTkG2!-1714998420;Version=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**
Property Service REST APIs

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
      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/Integer_Def
X-Oracle-RF-Token: FIxec1lmTFguxer9zqZJpsXSnI2Y_w**
Cookie: JSESSIONID=GSw4MnyYTtw87q6yCbb2Ht6J6klb5wjrqYv3b721tDMQqGdLKPGRc!-609606431;Version=1
Content-Type: application/json
Accept: application/json

Response:
Status Code: 200
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9zqZJpsXSnI2Y_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",
}
Property Service REST APIs

71.2.3.4 Update Property Definition

UpdatePD

**XML**

**Request:**
PUT
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions/Integer_Def
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=ZfhzMnYQz7pg4gzv5wZV1srvv3yDTsG4J4090X1GnHhrQpCSTkG2!-1714998420;Version=1
Content-Type: application/xml
Accept: application/xml
If-Match: 1286494289263

```xml
<?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/Integer_Def
71.2.3.5 Delete Property Definition

DeletePD

XML

Request:
DELETE http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions/Integer_Def
X-Oracle-RF-Token: FC59cJkhmJP1zPwKsTfD0Dxj8mw**
Cookie:
JSESSIONID=2fhzMoN0r7pg4gzv52V1srvy3yDtsG4J4090X10nHhrQpCSTnG21-1714998420;Version=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: FC59cJkhmJP1zPwKsTfD0Dxj8mw**
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
**71.2.3.6 Retrieve all Property Definitions**

This section describes the request and response for viewing property definitions within a namespace using the property service.

**RetrieveAllPDs**

**XML**

**Request:**
- POST
- URL: http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions
- X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0DxjmB_w**
- Cookie: JSESSIONID=ZfhzMnYQz7pg4gzv5w2V1srvvy3yDTeG4J4090X1GnHrgrqCSTkwG2!-1714998420;Version=s1
- Content-Type: application/xml
- Accept: application/xml

**Response:**
- Status Code: 200
- Content-Length: 1586
- X-Oracle-RF-Token-Scheme: utoken
- X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0DxjmB_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

```xml
<?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"
  </links>
</propertyDefinitions>
```
<rel="self" resourceType="urn:oracle:wcps:property:propertydefinitions">
<template>http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
</links>
<startIndex>0</startIndex>
<totalCount>1</totalCount>
<namespaceName>Oracle</namespaceName>
<items>
<propertyDefinition resourceType="urn:oracle:wcps:property:propertydefinition">
<links>
</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: FIxec1lmTfguexr9ZqZJpsXSn12y_w**
Cookie: JSESSIONID=GSw4MnyYTtw87Q6yCBb2H9F0J6kJwjrqYv3b72LcDn6QGdL3PGGc!-609606431; Version =1
Content-Type: application/json
Accept: application/json

**Response:**
Status Code:200
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTfguexr9ZqZJpsXSn12y_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",
      "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",
          "href": "http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions/Integer_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?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",
d-name:like:%25&startIndex=0&itemsPerPage=10",
"rel": "self"
]
}

71.2.3.7 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=ZfhzMnYQ27pg4gzv5W1srvy3yDTsG4J4090X10nHhrQpCSTkg21-1714998420;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: 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: F1xecllm7FguXer92q2JpsXSnl2y_w**
Cookie:
JSESSIONID=GSw4MnyYTww7Q6yCBB2H9F0UL6kLJwjrqFv3b72LcDMMQdLKPGRc1-609606431;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: F1xecllm7FguXer92q2JpsXSnl2y_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
71.2.4 Property Set Definition CRUD

71.2.4.1 Create Property Set Definition

This section describes the request and response for creating a property set definition within a namespace using the property service.

CreatePSD

XML

Request:

POST
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions
X-Oracle-RF-Token: FC59cJkhnPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=6G93MnYT5xcTQJ3t5PLn3v77sCPDNFLy9QGRSRnXlfHp81STbR8NH!-1714998420;Version=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: FC59cJkhnPwKsStfD0Dxjm8_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/partnersinfo
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"
            rel="self" resourceType="urn:oracle:wcps:property:propertysetdefinition"/>
        <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/Oracle/propertysetdefinitions/partnersinfo/propertysets?startIndex={startIndex}&dataFields={dataFields}&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"
                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/propertydefinitions/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/propertydefinitions/String_Def"
                        resourceType="urn:oracle:wcps:property:propertydefinition"/>
                </links>
                <propertyName>address</propertyName>
                <propertyDefinitionName>String_Def</propertyDefinitionName>
                </propertyMapping>
        </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/propertydefinitions/String_Def"
                    resourceType="urn:oracle:wcps:property:propertydefinition"/>
            </links>
            <propertyName>company_name</propertyName>
            <propertyDefinitionName>String_Def</propertyDefinitionName>
            </propertyMapping>
    </propertyMapping>
</propertySetDefinition>
<propertyDefinitionName>String_Def</propertyDefinitionName>
</propertyMapping>
<propertyMapping>
<link>
</link>
</propertyMapping>
</propertySetDefinition>

**JSON Request:**

**POST**

http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions

X-Oracle-RF-Token: F1xec1lmTFguxer9ZqZJpsXSnlZy_w**

Cookie: JSESSIONID=dpwQMyDkSNwv1TZ20JSSYyzJpKJn1JwBQdRtN0qNPPHwpqvs8Pc2!-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: F1xec1lmTFguxer9ZqZJpsXSnlZy_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:


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",
    "href": "http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions/Integer_Def"
  } ]
}, {
  "propertyName": "address",
  "propertyDefinitionName": "String_Def",
  "links": [ {
    "resourceType": "urn:oracle:wcps:property:propertydefinition",
    "href": "http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions/String_Def"
  } ]
}, {
  "propertyName": "company_name",
  "propertyDefinitionName": "String_Def",
  "links": [ {
    "resourceType": "urn:oracle:wcps:property:propertydefinition",
    "href": "http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions/String_Def"
  } ]
}, {
  "propertyName": "previous_ranking_and_current_ranking",
  "propertyDefinitionName": "Integer_Array_Def",
  "links": [ {
    "resourceType": "urn:oracle:wcps:property:propertydefinition",
    "href": "http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions/Integer_Array_Def"
  } ]
}, {
  "resourceType": "urn:oracle:wcps:property:propertysetdefinition",
  "links": [ {
    "resourceType": "urn:oracle:wcps:property:propertysetdefinition",
    "rel": "self"}]}
71.2.4.2 Retrieve Property Set Definition

**RetrievePSD**

**XML**

**Request:**
DELETE
X-Oracle-RF-Token: FC59cJkhnJP0lzPwKsStfD0Jxm8_w**
Cookie:
JSESSIONID=6Q93MnYT5xcTQJ3tPLn3vT7aCPDNFLyqQR85RnX1fHp81STbR8NH!:~17149986420;Version=1
Content-Type: application/xml
Accept: application/xml
If-Match: 128649292658

**Response:**
Status Code: 204
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJP0lzPwKsStfD0Jxm8_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
X-Oracle-RF-Token: F1xec1lmTFguxer9ZqZpsXSnJ2y_w**
Cookie:
JSESSIONID=dpwQMnyDkSNwv1T220JYSYzJpKJnJwB0dRtNLQhNPnhppgvs8Pc2!:~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
71.2.4.3 Retrieve All Property Set Definitions

This section describes the request and response for viewing property set definitions within a namespace using the property service.

**RetrieveAllPSDs**

**XML**

Request:
GET
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=6Q93MnYT5xcTQJ3tPLn3vT7sCPDNFyqzQGRSRnX1fHp81STbR8NH!-1714998420;Version=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
<?xml version="1.0" encoding="UTF-8"?>
<propertySetDefinitions
    resourceType="urn:oracle:wcps:property:propertysetdefinitions">
    <links>
        <link
            href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions?startIndex=0&itemsPerPage=10"
            rel="self"
            resourceType="urn:oracle:wcps:property:propertysetdefinitions"/>
    </links>
    <itemsPerPage>10</itemsPerPage>
    <startIndex>0</startIndex>
    <totalCount>0</totalCount>
    <namespaceName>Oracle</namespaceName>
    <items/>
</propertySetDefinitions>
```

**JSON**

Request:
GET

```json

```

---

71-44 Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSn1Zy_w**
Cookie:
JSESSIONID=dpvQmyDkSNwv1TZ20JSYyzJpKJnIjWQdRtNLoHQPgvs8Pc2!-609606431;Version =1
Content-Type: application/json
Accept: application/json

Response:
Status Code:200
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSn1Zy_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"
  } ]
}

71.2.4.4 Delete all Property Set Definitions

DeleteAllPSDs

XML

Request:
DELETE
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions
X-Oracle-RF-Token: FC59cJkhnJPo1zPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=6Q93MnYT5xcTQj3tPLn3v77sCPDNFLyZQGSRnX1fHp81STbR8NH!=1714998420;Version =1
Content-Type: application/xml
Accept: application/xml

Response:
Status Code:204
Content-Length: 0
71.2.5 Property Set CRUD

71.2.5.1 Create Property Set
This section describes the request and response for creating a property set within a namespace using the property service.

CreatePS

XML

Request:
POST
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie: JSESSIONID=vc1pMnYWRZysgqhXV=5XrGKhvfy0vv2zc7xQ3RysDsvG0mG2221J!-1714998420;Version=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>
    </property>
  </items>
</propertySet>
<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>

Response:

Status Code: 201
Content-Length: 3709
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhmJPolzPwKsStfD0Djxjm8_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/partnersinfo/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
      resourceType="urn:oracle:wcps:property:property"
      template="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/partnersinfo/propertysets/infosys/properties/{propertyName}"
    />
    <link
      href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/partnersinfo/propertysets/infosys/properties/{propertyName}"/>
  </links>
</propertySet>
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"
        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/propertysetdefinitions/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/propertysetdefinitions/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>
  </property>
</items>


```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": """
        }
      ],
      "multiValue": false,
      "resourceType": "urn:oracle:wcps:property:property"
    }
  ]
}
```
"value" : '1 foo loop, San Francisco, CA'
},
"multiValue" : false,
"resourceType" : "urn:oracle:wcps:property:property"
},
"name" : "number_of_employees",
"type" : "INT",
"values" : [ {
  "concreteType" :
  "value" : 10000
} ],
"multiValue" : false,
"resourceType" : "urn:oracle:wcps:property:property"
},
"name" : "previous_ranking_and_current_ranking",
"type" : "INT_ARRAY",
"values" : [ {
  "concreteType" :
  "value" : 3
}, {
  "concreteType" :
  "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: FIxec1lmTFguexer9ZqZJpsXSn12y_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/partnersinfo/propertysets/infosys
Date: Fri, 08 Oct 2010 01:24:58 GMT
ETag: "1286501098924"

{
  "name" : "infosys",
  "properties" : [ {
    "name" : "number_of_employees",
    "type" : "INT",
    "values" : [ {
      "concreteType" :
      "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:delete",
        "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:delete",
        "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:delete",
        "rel": "self"
    }
],

{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",
"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:delete",
        "rel": "self"
    }
],

{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",
"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:delete",
        "rel": "self"
    }
],

{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",
"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:delete",
        "rel": "self"
    }
],

{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",
"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:delete",
        "rel": "self"
    }
],

{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",
"multiValue": false,
"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",
    "href" :
    "rel" : "self"
  }]
},

"namespaceName" : 'Oracle",
"createdOn" : '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",
  }, {
    "resourceType" : "urn:oracle:wcps:property:propertyset",
    "href" :
    "rel" : "self"
  }]
}

### 71.2.5.2 Retrieve Property Set

**RetrievePS**

**XML**

**Request:**

<?xml version="1.0" encoding="UTF-8"?>
<propertySet resourceType="urn:oracle:wcps:property:propertyset">
    <links>
    </links>
    <namespaceName>Oracle</namespaceName>
    <propertySetDefinitionName>partnersinfo</propertySetDefinitionName>
    <name>infosys</name>
    <items>
        <property multiValue="false" resourceType="urn:oracle:wcps:property:property">
            <links>
            </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>
    </items>
</propertySet>
<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/propertysetdefinitions/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">
<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/propertysetdefinitions/partnersinfo/propertysets/infosys/properties/address"
rel="self" resourceType="urn:oracle:wcps:property:property"/>
</link>

<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">
<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/propertysetdefinitions/partnersinfo/propertysets/infosys/properties/previous_ranking_and_current_ranking"
rel="self" resourceType="urn:oracle:wcps:property:property"/>
</link>

<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>
Property Service REST APIs

JSON Request:
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSn1Zy_w**
Cookie: JSESSIONID=h72xMnyKQQSw9Lwmg2HMGDnNQnK1H2JpGJmLgZ2PlvHPvT2bX2k+-609606431;Version=1
Content-Type: application/json
Accept: application/json

Response:
Status Code: 200
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSn1Zy_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",
    "multiValue" : false,
    "resourceType" : "urn:oracle:wcps:property:property",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:property:property",
      "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",
    "multiValue" : false,
    "resourceType" : "urn:oracle:wcps:property:property",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:property:property",
      "rel" : "self"
    } ]
  } ]
"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" :
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",
"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" :
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",
"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" :
partnersinfo/propertysets/infosys/properties/previous_ranking_and_current_ranking",
  "rel" : "self"
} ]
71.2.5.3 Update Property Set

**UpdatePS**

**XML Request:**

```xml
PUT http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/partnersinfo/propertysets/infosys/properties/{propertyName}
X-Oracle-RF-Token: FC59cJkhmJFolxPwKsStfDODxjmB_w**
Cookie: JSESSIONID=vc1pMnYWRZysgqhXVz5XRgKqhVfy0vv2zc7xQ3RyDSvG0mG221LJ!-1714998420;Version=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>
    </property>
  </items>
</propertySet>
```


<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>
  <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>
  <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>

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"

Request:
PUT
X-Oracle-RF-Token: FIxec11mTPguxe92q2JPsXSn12y_w**
Cookie:
JSESSIONID=h72xMnykQQSW9Lwmc52HM5DnQnK1HZupGzMsLgZ2P1vHFpvT2bX2k!-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: FIxec11mTFguxer92qJpsXSnl2y_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"

71.2.5.4 Delete Property Set

DeletePS

Request:
DELETE
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfDOXjm8_w**
Cookie: JSESSIONID=vc1pMnYWR2ysgghXVz5XrGQhvfy0vv2zc7xQ3RyDSvG0mG22z1J!-1714998420;Version=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: FC59cJkhnJPolzPwKsStfDOXjm8_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
X-Oracle-RF-Token: FIxec11mTFguxer92qJpsXSnl2y_w**
Cookie: JSESSIONID=h72xmNhyKQQSW9Iwgc2HMGDnNQnK1HZJpGJmLgZ2P1vHPpvTz2X2k!-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: FIxec11mTFguxer92qJpsXSnl2y_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
71.2.5.5 Retrieve all Property Sets

This section describes the request and response for viewing a property set within a namespace using the property service.

RetrieveAllPSs

XML

Request:
GET
X-Oracle-RF-Token: FC59cJkhmJPolzPwKsStfD0DxjmB_w**
Cookie: JSESSIONID=vcipMnYWR2ysqwhXVz5XrwGkhvfy0vV2zz7xQ3rDSvG0mG222LJ!-1714998420;Version="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: FC59cJkhmJPolzPwKsStfD0DxjmB_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/propertysetdefinitions/partnersinfo/propertysets?startIndex=0&itemsPerPage=10"
rel="self" resourceType="urn:oracle:wcps:property:propertysets"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/partnersinfo/propertysets?startIndex={startIndex}&dataFields={dataFields}&q={q}&itemsPerPage={itemsPerPage}"/>
  </links>
  <itemsPerPage>10</itemsPerPage>
  <startIndex>0</startIndex>
  <totalCount>0</totalCount>
  <namespaceName>Oracle</namespaceName>
  <propertySetDefinitionName>partnersinfo</propertySetDefinitionName>
  <items/>
</propertySets>

JSON

Request:
GET
X-Oracle-RF-Token: FIxec11mTFguxer9ZqZJpsXSnL3y_w**
Cookie: JSESSIONID=h72xMnyKQSSW9Lwqg2HMMDnMq0nK1H2JpGJmLg22P1vHPvT2bX2k!-609606431;Version="1"
Property Service REST APIs

Response:
Status Code: 200
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnl2y_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" :
      partnersinfo/propertysets?startIndex={startIndex}&dataFields={dataFields}&q={q}&itemsPerPage={itemsPerPage},
      "resourceType" : "urn:oracle:wcps:property:propertysets",
      "href" :
        partnersinfo/propertysets?startIndex=0&itemsPerPage=10",
      "rel" : "self"
  } ]
}

71.2.5.6 Delete all Property Sets

DeleteAllPSs

XML

Request:
DELETE
X-Oracle-RF-Token: FC59cJkhnJPo1zPwKaStfD0Dxjm8_w**
Cookie:
JSESSIONID=vc1pMnYWRZysgqhXVz5XrGKh4fy0v+v2zc7xQ3RyD8vG0mG221J!-1714998420;Version=1
Content-Type: application/xml
Accept: application/xml

Response:
Status Code: 204
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
71.2.6 Property CRUD

71.2.6.1 Create Property

XML

Request:
POST
X-Oracle-RF-Token: FC59cJkhnJPo1zPwKsStfD0Dxjm8_w**
Cookie: JSESSIONID=vc1pMnYWRZysgqhXVz5XrgKqhvfy0vV2zc7xQ3RyDSvG0mG22ZfJ!-1714998420;Version=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: FC59cJkhnPJo1zPwKsStfD0Dxjm8_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/partnersinfo/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/propertysetdefinitions/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
X-Oracle-RF-Token: F1xec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=h72xDMyYQQSW9Lwmcg2HM3DnNQnKlHZjPgJmLgZ2P1vHFpvT2bX2k!-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: F1xecl1mTFguxer9ZqZJpsXSn12y_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/partnersinfo/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",
    "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",
        "rel" : "self"
    } ]
}

71.2.6.2 Retrieve Property

Retrieve Property

XML

Request:
GET
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/partnersinfo/propertysets/infosys/properties/ceo_name
X-Oracle-RF-Token: FC59cJkhmJPo1zPwKsStfD0DxjmB_w**
Cookie: JSESSIONID=vc1pMnYWR5yagqhXvz5XrGKgbyy0vv2zc7xQ3hyDsvG0mG2221J!:1-714998420;Versio
ns=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: FC59cJkhmJPo1zPwKsStfD0DxjmB_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Property Service REST APIs

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/propertysetdefinitions/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/partnersinfo/propertysets/infosys/properties/ceo_name
X-Oracle-RF-Token: FIxecllmT9guxer9Zq2psXSN12y_w**
Cookie: JSESSIONID=h72xMnyKQ0Sw9Kwcg2iHMDnWQnK1H2LpGJmLgZ2P1vHFPvT2bX2k!-609606431;Version =1
Content-Type: application/json
Accept: application/json

Response:
Status Code:200
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxecllmT9guxer9Zq2psXSN12y_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",
  "multiValue" : false,
'resourceType' : "urn:oracle:wcps:property:property",
'links' : [ {
    'resourceType' : "urn:oracle:wcps:property:property",
    'href' :
        'http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/partnersinfo/propertysets/infosys/properties/ceo_name",
    'rel' : 'self'
} ]
}

71.2.6.3 Update Property

UpdateP

XML

Request:
PUT
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/partnersinfo/propertysets/infosys/properties/ceo_name
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0DxjmB_w**
Cookie: JSESSIONID=vc1pMnYWRZyqghXVz5XrGKhvfy0vvv2zc7xQ3RyDSvG0mG22ZlJ!-1714998420;Version=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: FC59cJkhnJPolzPwKsStfD0DxjmB_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/partnersinfo/propertysets/infosys/properties/ceo_name
X-Oracle-RF-Token: FIxec1lMTFguxer92qZjpsXSnl2y_w**
Cookie: JSESSIONID=h72xMnyKQcSW9LMwcg2HMGDnNqNk1HZJpQjMlGz2P1vHFpvT2bX2k!=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: F1xec1lm7Fguxer9Zq2JpsXSnl2y_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Connection: close
Cache-Control: no-cache
Date: Fri, 08 Oct 2010 01:24:59 GMT
ETag: "1286501099162"

71.2.6.4 Delete Property

Delete

XML

Request:
DELETE
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/partnersinfo/propertysets/infosys/properties/ceo_name
X-Oracle-RF-Token: F1xec1lm7Fguxer9Zq2JpsXSnl2y_w**
Cookie: JSESSIONID=VvThMyG0gy9PprkF4VtrjTDHJ3SL894V0glWzqf24L0B!=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: F1xec1lm7Fguxer9Zq2JpsXSnl2y_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Connection: close
Date: Fri, 08 Oct 2010 01:24:58 GMT
JSON

Request:
DELETE
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/partnersinfo/propertysets/infosys/properties/ceo_name
X-Oracle-RF-Token: F1xec1lmTFguxer92qZJpsXSn1Zy_w**
Cookie:
JSESSIONID=h72xMnyKQQSw9Lwcg2HMGDnNQnKiHzJpGJmLg22PlvHPvpT2b2k!-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: F1xec1lmTFguxer92qZJpsXSn1Zy_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

71.2.6.5 Delete all Properties

XML

Request:
DELETE
X-Oracle-RF-Token: F1xec1lmTFguxer92qZJpsXSn1Zy_w**
Cookie:
JSESSIONID=VyThMnyG0gyp97J3vKPrkfF4VvtrjTDHLJ3SL894VOglWzqJ24L0B!-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: F1xec1lmTFguxer92qZJpsXSn1Zy_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
X-Oracle-RF-Token: F1xec1lmTFguxer92qZJpsXSn1Zy_w**
Cookie:
JSESSIONID=h72xMnyKQQSw9Lwcg2HMGDnNQnKiHzJpGJmLg22PlvHPvpT2b2k!-609606431;Version =1
Content-Type: application/json
Accept: application/json
If-Match: 1286501099162
71.2.7 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>
<thead>
<tr>
<th>Property Definition Type</th>
<th>Java Data Type</th>
<th>concreteType Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLOB</td>
<td>oracle.wcps.property.BlobData</td>
<td>urn:oracle:wcps:property:blob-data</td>
</tr>
<tr>
<td>CLOB</td>
<td>oracle.wcps.property.ClobData</td>
<td>urn:oracle:wcps:property:clob-data</td>
</tr>
<tr>
<td>DATETIME</td>
<td>oracle.wcps.property.DateTime</td>
<td>urn:oracle:wcps:property:date-time</td>
</tr>
</tbody>
</table>

71.2.8 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 /resourceIndex request.
- Attempting to delete an "in use" Namespace, Property Definition, or Property Set Definition results in a 412 response status.

71.2.9 Query Parameters

Table 71–2 shows the supported query parameters and their default values.
71.2.10 Search Expressions

71.2.10.1 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]

71.2.10.2 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 Operation
  - ; ===> and

71.2.10.3 Supported Attribute Names

Table 71–3, Table 71–4, Table 71–5, and Table 71–6 show the supported attribute names that can be used in a search expressions to search for property service entities.

Namespace Attributes

Table 71–3 Namespace Attributes

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ns-name</td>
<td>A namespace’s name</td>
</tr>
</tbody>
</table>
Examples:

/wcps/api/property/namespaces?q=ns-name:like:Oracle_%25

/wcps/api/property/namespaces?q=ns-definition-locator-class-name:like:%25Locator%25;ns-name:like:Oracle_%25

Property Definition Attributes

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pd-name</td>
<td>A property definition's name</td>
</tr>
<tr>
<td>pd-description</td>
<td>A property definition's description</td>
</tr>
<tr>
<td>pd-type</td>
<td>A property definition's type</td>
</tr>
<tr>
<td>pd-restricted</td>
<td>A property definition's restricted attribute</td>
</tr>
<tr>
<td>pd-validator-class-name</td>
<td>A property definition's validator class name</td>
</tr>
<tr>
<td>pd-created-on</td>
<td>A property definition's created on date time</td>
</tr>
<tr>
<td>pd-updated-on</td>
<td>A property definition's updated on date time</td>
</tr>
</tbody>
</table>

Example:


Property Set Definition Attributes

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>psd-name</td>
<td>A property set definition's name</td>
</tr>
<tr>
<td>psd-description</td>
<td>A property set definition's description</td>
</tr>
<tr>
<td>psd-pd-name</td>
<td>A property definition name referred by a property set definition</td>
</tr>
<tr>
<td>psd-property-locator-class-name</td>
<td>A property locator class name associated with a property set definition</td>
</tr>
<tr>
<td>psd-created-on</td>
<td>A property set definition's created on date time</td>
</tr>
<tr>
<td>psd-updated-on</td>
<td>A property set definition's updated on date time</td>
</tr>
</tbody>
</table>

Example:

/wcps/api/property/namespaces/Oracle/propertysetdefinitions?q=psd-created-on:gt:1980-01-01T00:00:00.000-0700;psd-name:like:%255%25

Table 71–3  (Cont.) Namespace Attributes

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ns-property-locator-class-name</td>
<td>A namespace's property locator class name</td>
</tr>
<tr>
<td>ns-definition-locator-class-name</td>
<td>A namespace's definition locator class name</td>
</tr>
<tr>
<td>ns-created-on</td>
<td>A namespace's created on date time</td>
</tr>
<tr>
<td>ns-updated-on</td>
<td>A namespace's updated on date time</td>
</tr>
</tbody>
</table>
## Property Set Attributes

### Table 71–6  Property Set Attributes

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ps-name</td>
<td>A property set's name</td>
</tr>
<tr>
<td>ps-description</td>
<td>A property set's description</td>
</tr>
<tr>
<td>psd-created-on</td>
<td>A property set's created on date time</td>
</tr>
<tr>
<td>psd-updated-on</td>
<td>A property set's updated on date time</td>
</tr>
<tr>
<td>a property’s name</td>
<td>A property's name attribute (for example, firstName, zipCode, etc.)</td>
</tr>
</tbody>
</table>

**Example:**

```
/wcps/api/property/namespaces/Oracle/propertysetdefinitions/partners_info/propertysets?q=ps-created-on:gt:1980-01-01T00:00:00.000-0700
```
This chapter describes the public APIs for the Conductor, the component that orchestrates data integration components and activities for WebCenter Portal.

This chapter includes the following topics:

- Section 72.1, "Using Java APIs"
- Section 72.2, "Using the Conductor REST APIs"
- Section 72.3, "Calling Data Integration Client Services Using ELs"

72.1 Using Java APIs

The service interfaces shown in Table 72–1 define the remote Conductor API.

<table>
<thead>
<tr>
<th>Interface/Service</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>oracle.wcps.conductor.services.IScenariosService</td>
<td>General management, retrieval, and execution of stored Scenarios.</td>
</tr>
<tr>
<td>oracle.wcps.conductor.services.INamespaceService</td>
<td>General management of stored namespaces known to the Conductor.</td>
</tr>
<tr>
<td>oracle.wcps.conductor.services.IProviderService</td>
<td>Retrieval of provider metadata known to the Conductor server.</td>
</tr>
</tbody>
</table>

72.2 Using the Conductor REST APIs

This section provides examples of how you can use the Conductor's REST APIs to programatically carry out basic data integration 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 data integration ELs to call them from within your WebCenter Portal Framework application or WebCenter Portal.

This section includes the following subsections:

- Section 72.2.1, "Using the Data Provider Management REST APIs"
- Section 72.2.2, "Using the Function Provider Management APIs"
72.2.1 Using the Data Provider Management REST APIs

This section provides examples of using the Conductor's data provider management APIs, and contains the following subsections:

- Section 72.2.1.1, "Getting a Collection of Data Providers (GET)"
- Section 72.2.1.2, "Getting a Single Data Provider's Metadata (GET)"
- Section 72.2.1.3, "Getting Information About a Data Provider Connection (GET)"
- Section 72.2.1.4, "Getting Information about an Executable Resource for a Data Provider and Connection (GET)"
- Section 72.2.1.5, "Getting Method Information for an Executable Resource for a Data Provider and Connection (GET)"
- Section 72.2.1.6, "Getting Entire Data Provider Information with Details Projection (GET)"

72.2.1.1 Getting a Collection of Data Providers (GET)

XML

Sample Request:
accept-language: en
content-type: application/xml
accept: application/xml

Sample Response:
Status Code: 200
Content-Length: 4469
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie: JSESSIONID=YxYmM25GLpCy7BSND7vF6BnsQCh20RQsCr92n1SLXmxpccVkpX1822326988;
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>
  </links>
  <items>
    <dataProvider resourceType="urn:oracle:wcps:conductor:provider:dataProvider">
      <links>
        ...
      </links>
    </dataProvider>
    ...
  </items>
</dataProviders>
<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/CMISProvider"
    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/SampleProvider"
    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/ActivityGraphProvider"
    rel="self"
    resourceType="urn:oracle:wcps:conductor:provider:dataProvider"
    template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ActivityGraphProvider?
    projection={projection}"/>
    </links>
</dataProvider>
Using the Conductor REST APIs


class ActivityGraphProvider

<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"
rel="self"
resourceType="urn:oracle:wcps:conductor:provider:dataProvider"
</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:
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=ylp2M27pSyLJPv2X4cn9RYy7NRyfTzz2vs7VCyJ1NQXKJ1XQnpjXdl!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",
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/Activity
GraphProvider?projection={projection}",
    'capabilities' : "urn:oracle:restframework:read",
    'href' :
'http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/Activity
GraphProvider",
    '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'
  ]
}
Using the Conductor REST APIs

72.2.1.2 Getting a Single Data Provider's Metadata (GET)

**XML**

**Request:**

```
```

```
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=L3b2MN25L67PvznfYfQnHcyPh2J11rP51MYL1h8QvWmPxlHPyY!82326988;
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>
  </links>
</dataProvider>
Using the Conductor REST APIs

iders/{provider-name}/connections/{connection-name}/resources/{resource-name}/methods/{method-name}?projection={projection}' />

<link>
<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">
<link capabilities="urn:oracle:restframework:read"
rel="urn:oracle:webcenter:parent"
resourceType="urn:oracle:wcps:conductor:provider:dataProvider"
<link capabilities="urn:oracle:restframework:read"
rel="self"
resourceType="urn:oracle:wcps:conductor:provider:providerConnections"
<link>
<items
resourceType="urn:oracle:wcps:conductor:provider:providerConnection">
<link capabilities="urn:oracle:restframework:read"
rel="urn:oracle:wcps:conductor:provider:connectionMetadata"
resourceType="urn:oracle:wcps:conductor:provider:connectionMetadata"
<link capabilities="urn:oracle:restframework:read"
rel="self"
resourceType="urn:oracle:wcps:conductor:provider:providerConnection"
<link>
</items>
<providerConnection
resourceType="urn:oracle:wcps:conductor:provider:providerConnection">
<link capabilities="urn:oracle:restframework:read"
rel="urn:oracle:wcps:conductor:provider:connectionMetadata"
resourceType="urn:oracle:wcps:conductor:provider:connectionMetadata"
<link capabilities="urn:oracle:restframework:read"
rel="self"
resourceType="urn:oracle:wcps:conductor:provider:providerConnection"
<link>
</providerConnection>
Using the Conductor REST APIs

JSON

Request:
GET  
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=R1P2M27QdO9wh86s5rG0DFkBcbKfKftPKvdpDrM2gcpLR7XP0nQ18223263988; 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 implemenation to integrate the properties service for use in scenarios.",
  "providerConnections" : [ { 
    "name" : "LocalServerConnection",
    "connectionResources" : [ { 
      "name" : "GetPropertyResource",
      "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."
        } ]
      }, { 
        "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."
        } ]
      } ],
      "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
      "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, null will be returned."
    } ]
  } ]
}
Using the Conductor REST APIs

```json
    "links": [ {
        "resourceType": "urn:oracle:wcps:conductor:provider:resourceMethods",
        "capabilities": "urn:oracle:restframework:read",
        "rel": "urn:oracle:webcenter:parent"
    }, {
        "resourceType": "urn:oracle:wcps:conductor:provider:resourceMethod",
        "capabilities": "urn:oracle:restframework:read",
        "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, null will be returned.",
    "resourceType": "urn:oracle:wcps:conductor:provider:resourceMethod",
    "links": [ {
        "resourceType": "urn:oracle:wcps:conductor:provider:resourceMethods",
        "capabilities": "urn:oracle:restframework:read",
    } ]
```

72-10  Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper
Conductor API Reference

72-11

Using the Conductor REST APIs

```
}, {
    "resourceType": "urn:oracle:wcps:conductor:provider:resourceMethod",
    "capabilities": "urn:oracle:restframework:read",
    "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",
        "capabilities": "urn:oracle:restframework:read",
        "rel": "urn:oracle:webcenter:parent"
    }, {
        "resourceType": "urn:oracle:wcps:conductor:provider:resourceMethod",
        "capabilities": "urn:oracle:restframework:read",
        "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",
    "capabilities" : "urn:oracle:restframework:read",
    "rel" : "urn:oracle:webcenter:parent"
    } ], { "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
    "capabilities" : "urn:oracle:restframework:read",
    "capabilities" : "urn:oracle:restframework:read",
    "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",
    "capabilities" : "urn:oracle:restframework:read",
    "capabilities" : "urn:oracle:restframework:read",
    "rel" : "self"
    } ]
)"
```
Using the Conductor REST APIs


- "rel" : "urn:oracle:webcenter:parent"
- "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
  - "capabilities" : "urn:oracle:restframework:read",
- "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",
  - "capabilities" : "urn:oracle:restframework:read",
- "rel" : "urn:oracle:webcenter:parent"
}
],
- "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
  - "capabilities" : "urn:oracle:restframework:read",
- "rel" : "self"
}]
"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",
  "capabilities" : "urn:oracle:restframework:read",
  "rel" : "urn:oracle:webcenter:parent"
}, {
  "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
  "capabilities" : "urn:oracle:restframework:read",
  "rel" : "self",
} ]
},
"resourceType" : "urn:oracle:wcps:conductor:provider:connectionResource",
"capabilities" : "urn:oracle:restframework:read",
"rel" : "urn:oracle:webcenter:parent"
}, {
  "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
  "capabilities" : "urn:oracle:restframework:read",
  "rel" : "self",
} ]}
propertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties Resource/methods?projection={projection}
,  
'capabilities' : "urn:oracle:restframework:read",
'href' : 
'rel' : 'self'  
}),  
'resourceType' : "urn:oracle:wcps:conductor:provider:connectionResource",
'links' : [ 
  
'resourceType' : "urn:oracle:wcps:conductor:provider:connectionResources",
'template' :  
'capabilities' : "urn:oracle:restframework:read",
'href' : 
'rel' : 'urn:oracle:webcenter:parent'
},  
'resourceType' : "urn:oracle:wcps:conductor:provider:connectionResource",
'template' :  
'capabilities' : "urn:oracle:restframework:read",
'href' : 
'rel' : 'urn:oracle:wcps:conductor:provider:connectionMetadata'  
},  
'resourceType' : "urn:oracle:wcps:conductor:provider:providerConnection",
'template' :  
'capabilities' : "urn:oracle:restframework:read",
'href' : 
'rel' : 'urn:oracle:webcenter:parent'
},  
'resourceType' : "urn:oracle:wcps:conductor:provider:providerConnection",
'template' :  
'capabilities' : "urn:oracle:restframework:read",
'href' : 
'rel' : 'urn:oracle:wcps:conductor:provider:connectionMetadata"  
},  
'resourceType' : "urn:oracle:wcps:conductor:provider:providerConnection",
'template' :  
'capabilities' : "urn:oracle:restframework:read",
'href' : 
'rel' : 'urn:oracle:wcps:conductor:provider:connectionResource"
"capabilities" : "urn:oracle:restframework:read",
"rel" : "self"
}]
]

"resourceType" : "urn:oracle:wcps:conductor:provider:dataProvider",
"links" : [
{
"resourceType" : "urn:oracle:wcps:conductor:provider:dataProviders",
"capabilities" : "urn:oracle:restframework:read",
"rel" : "urn:oracle:webcenter:parent"
},
{
"resourceType" : "urn:oracle:wcps:conductor:provider:dataProvider",
"capabilities" : "urn:oracle:restframework:read",
"rel" : "self"
}],
{
"resourceType" : "urn:oracle:wcps:conductor:provider:providerConnections",
"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/{provider-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/{provider-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/{provider-name}/connections/{connection-name}/resources/{resource-name}?projection={projection}",
"capabilities" : "urn:oracle:restframework:read",
"rel" : "urn:oracle:wcps:conductor:provider:connectionResource"
72.2.1.3 Getting Information About a Data Provider Connection (GET)

**XML**

**Request:**
GET
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=QGswM25H1GLC61Qvbt08HLfJhfJdN50TSV81RzWhpvbJG0vFJtSN!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>
  </links>
</connection>
Using the Conductor REST APIs

```
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/metadata?projection={projection}"
  <link capabilities="urn:oracle:restframework:read"
  rel="self"
resourceType="urn:oracle:wcps:conductor:provider:providerConnection"
</links>
<name>LocalServerConnection</name>
<connectionResources
resourceType="urn:oracle:wcps:conductor:provider:connectionResources">
<link capabilities="urn:oracle:restframework:read"
  rel="urn:oracle:webcenter:parent"
resourceType="urn:oracle:wcps:conductor:provider:providerConnection"
</link>
<link capabilities="urn:oracle:restframework:read"
href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources"
  rel="self"
resourceType="urn:oracle:wcps:conductor:provider:connectionResources"
</link>
<items>
<connectionResource
resourceType="urn:oracle:wcps:conductor:provider:connectionResource">
<link capabilities="urn:oracle:restframework:read"
href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources"
  rel="urn:oracle:webcenter:parent"
resourceType="urn:oracle:wcps:conductor:provider:connectionResources"
</link>
<link capabilities="urn:oracle:restframework:read"
  rel="self"
resourceType="urn:oracle:wcps:conductor:provider:connectionResource"
</link>
</items>
</connectionResources>
```
Using the Conductor REST APIs

**JSON Request:**

GET


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=tLqRM27fHTGT9nxHFwIyHJv3522rhw84Z5Rcymb1dy3pLMFgp6!822326988;

path=/wcps; HttpOnly

Transfer-Encoding: chunked

Content-Type: application/json

Date: Wed, 13 Oct 2010 23:07:11 GMT

```json
{
    "name" : "LocalServerConnection",
    "connectionResources" : [ {
        "name" : "GetPropertiesResource",
        "resourceType" : "urn:oracle:wcps:conductor:provider:connectionResource",
        "links" : [ {
            "resourceType" : "urn:oracle:wcps:conductor:provider:connectionResources",
            "capabilities" : "urn:oracle:restframework:read",
            "rel" : "urn:oracle:webcenter:parent"
        }, {
            "resourceType" : "urn:oracle:wcps:conductor:provider:connectionResource",
            "capabilities" : "urn:oracle:restframework:read",
            "rel" : "self"
        } ]
    } ],
    "resourceType" : "urn:oracle:wcps:conductor:provider:providerConnection",
    "links" : [ {
        "resourceType" : "urn:oracle:wcps:conductor:provider:providerConnections",
        "rel" : "self"
    } ]
}
```
Using the Conductor REST APIs

propertiesServiceProvider/connections?projection={projection}
"capabilities" : "urn:oracle:restframework:read",
"rel" : "urn:oracle:webcenter:parent"
},
{
"resourceType" : "urn:oracle:wcps:conductor:provider:connectionMetadata",
"template" :
",
"capabilities" : "urn:oracle:restframework:read",
"rel" : "urn:oracle:wcps:conductor:provider:connectionMetadata"
},
{
"resourceType" : "urn:oracle:wcps:conductor:provider:providerConnection",
"template" :
",
"capabilities" : "urn:oracle:restframework:read",
"rel" : "self"
} ]

72.2.1.4 Getting Information about an Executable Resource for a Data Provider and Connection (GET)

XML

Request:
GET
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=6NMkM25LjbjyWPMTScyYksJhItVVRK9BhTv5MvGYjDHCV6yHQGQ!822326988;
persistent=true; path=/wcps; HttpOnly

<?xml version="1.0" encoding="UTF-8"?>
<connectionResource
resourceType="urn:oracle:wcps:conductor:provider:connectionResource"
<links>

72-20 Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper
Using the Conductor REST APIs

Conductor API Reference 72-21

Using the Conductor REST APIs

Conductor API Reference 72-21
  resourceType="urn:oracle:wcps:conductor:provider:resourceMethod"
</links>
<name>GetProperty</name>
<uniqueName>GetProperty(namespace,definition,set,property)</uniqueName>
<parameters/>
</resourceMethod>
<resourceMethod resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
<link capabilities="urn:oracle:restframework:read"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod"
</links>
<name>GetProperty</name>
<uniqueName>GetProperty(definition,set,property)</uniqueName>
<parameters/>
</resourceMethod>
<resourceMethod resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
<link capabilities="urn:oracle:restframework:read"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod"
</links>
<name>GetProperty</name>
<uniqueName>GetProperty(set,property)</uniqueName>
<parameters/>
</resourceMethod>
<resourceMethod resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
<link capabilities="urn:oracle:restframework:read"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod"
</links>
<name>GetProperty</name>
<uniqueName>GetProperty(property)</uniqueName>
<parameters/>
</resourceMethod>
Using the Conductor REST APIs
Using the Conductor REST APIs

resourceType="urn:oracle:wcps:conductor:provider:resourceMethod"
</links>
<name>GetPropertySet</name>
<uniqueName>GetPropertySet(namespace,definition,set)</uniqueName>
<parameters/>
</resourceMethod>

resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
<links>
<link capabilities="urn:oracle:restframework:read"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
<link capabilities="urn:oracle:restframework:read"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod"
</links>
<name>GetPropertySets</name>
<uniqueName>GetPropertySets(definition)</uniqueName>
<parameters/>
</resourceMethod>

resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
<links>
<link capabilities="urn:oracle:restframework:read"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
<link capabilities="urn:oracle:restframework:read"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod"
</links>

Using the Conductor REST APIs

PropertiesResource/methods/GetPropertySets(namespace, definition)?projection={projection} />

</links>

<name>GetPropertySets</name>
<uniqueName>GetPropertySets(namespace, definition)</uniqueName>
<parameters/>
</resourceMethod>
</items>
</resourceMethods>
</connectionResource>

**JSON**

**Request:**

GET


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=lP2dM27QwpNKslGK8xd818TLMmBhpdBzcLmzyncLXvlv6BrNjkq8!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',
        'capabilities': 'urn:oracle:restframework:read",
        'rel': 'urn:oracle:webcenter:parent"
    ), 
    'resourceType': 'urn:oracle:wcps:conductor:provider:resourceMethod',
    'capabilities': 'urn:oracle:restframework:read",
    'href': 

}
Using the Conductor REST APIs

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",
  ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
 Resource/methods?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
  ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
 Resource/methods/GetProperty(definition,set,property)?projection={projection}",
      "rel" : "urn:oracle:webcenter:parent"
   },
   { "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
  ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
 Resource/methods/execute()?projection={projection}",
     "capabilities" : "urn:oracle:restframework:read",
  ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
 Resource/methods/execute()"
   ] ]
   },
   {
      "name" : "execute",
      "uniqueName" : "execute()",
      "parameters" : [ ],
      "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
      "links" : [ {
         "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
            ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
 Resource/methods/execute()?projection={projection}",
         "capabilities" : "urn:oracle:restframework:read",
            ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
 Resource/methods/execute()"
      ] ]
   }
Using the Conductor REST APIs

```
  "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" : 
    "capabilities" : "urn:oracle:restframework:read",
    "href" : 
    "rel" : "urn:oracle:webcenter:parent"
  },
  {
    "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
    "template" : 
    "capabilities" : "urn:oracle:restframework:read",
    "href" : 
    "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" : 
    "capabilities" : "urn:oracle:restframework:read",
    "href" : 
    "capabilities" : "urn:oracle:restframework:read",
    "href" : 
    "rel" : "urn:oracle:webcenter:parent"
  },
  {
    "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
    "template" : 
    "capabilities" : "urn:oracle:restframework:read",
    "href" : 
```
Using the Conductor REST APIs

"href" :
"rel" : "self"
}

"name" : "GetPropertySets",
"uniqueName" : "GetPropertySets(definition)",
"parameters" : [ ],
"resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
"links" : [ { "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
"capabilities" : "urn:oracle:restframework:read",
"rel" : "urn:oracle:webcenter:parent"
}, { "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
"capabilities" : "urn:oracle:restframework:read",
"rel" : "urn:oracle:webcenter:parent"
}, { "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
"capabilities" : "urn:oracle:restframework:read",
"rel" : "urn:oracle:webcenter:parent"}
72.2.1.5 Getting Method Information for an Executable Resource for a Data Provider and Connection (GET)

XML Request:

GET
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=rnBkM27d8dH MuhammadT4WdT0CLGythB2hRh02yjTmg6r7BDfB1JTQjL!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>
  </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>
    </parameter>
  </parameters>
</resourceMethod>
Using the Conductor REST APIs

**Conductor API Reference**

```xml
<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
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=MD62M27BQLv9jxFR8K1SSXXyL70DwLGd7gpHvWb21xGM9gfpLR2!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."
}
72.2.1.6 Getting Entire Data Provider Information with Details Projection (GET)

XML

Request:
GET
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=T5nQM27dHT1yGdzP7yMy251oQGKDc95kYmmtN83tfhT72n1gkcQ!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"
      rel="urn:oracle:webcenter:parent"
      resourceType="urn:oracle:wcps:conductor:provider:dataProvider">
      ...
    </link>
    ...
  </links>
...
Using the Conductor REST APIs

            <link capabilities="urn:oracle:restframework:read"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:dataProvider"
            <link capabilities="urn:oracle:restframework:read"
          rel="urn:oracle:wcps:conductor:provider:providerConnections"
          resourceType="urn:oracle:wcps:conductor:provider:providerConnections"
            <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}/dataProviders/{provider-name}/connections/{connection-name}/resources?projection={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}/dataProviders/{provider-name}/connections/{connection-name}/resources/{resource-name}/methods?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}/dataProviders/{provider-name}/connections/{connection-name}/resources/{resource-name}/methods/{method-name}?projection={projection}"/>
        </links>
      </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">
        <link capabilities="urn:oracle:restframework:read"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:dataProvider"
    </providerConnections>
  </providerConnections>
</providerConnections>
</providerConnections>
</providerConnections>
</providerConnections>
</providerConnections>
</providerConnections>
</providerConnections>
</providerConnections>
</providerConnections>
</providerConnections>
</providerConnections>
</providerConnections>
</providerConnections>
</providerConnections>
</providerConnections>
</providerConnections>
</providerConnections>
</providerConnections>
</providerConnections>
</providerConnections>
</providerConnections>
</providerConnections>
</providerConnections>
</providerConnections>
</providerConnections>
</providerConnections>
</providerConnections>

Conductor API Reference  72-33
Using the Conductor REST APIs

```
<link capabilities="urn:oracle:restframework:read"
rel="self"
resourceType="urn:oracle:wcps:conductor:provider:providerConnections"
</links>
<items>
  <providerConnection
resourceType="urn:oracle:wcps:conductor:provider:providerConnection">
    <links>
      <link capabilities="urn:oracle:restframework:read"
rel="urn:oracle:webcenter:parent"
resourceType="urn:oracle:wcps:conductor:provider:providerConnections"
      <link capabilities="urn:oracle:restframework:read"
rel="urn:oracle:wcps:conductor:provider:connectionMetadata"
resourceType="urn:oracle:wcps:conductor:provider:connectionMetadata"
      <link capabilities="urn:oracle:restframework:read"
rel="self"
resourceType="urn:oracle:wcps:conductor:provider:providerConnection"
    </links>
    <name>LocalServerConnection</name>
    <connectionResources
resourceType="urn:oracle:wcps:conductor:provider:connectionResources">
      <links>
        <link capabilities="urn:oracle:restframework:read"
rel="urn:oracle:webcenter:parent"
resourceType="urn:oracle:wcps:conductor:provider:providerConnection"
        <link capabilities="urn:oracle:restframework:read"
href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources"
rel="self"
resourceType="urn:oracle:wcps:conductor:provider:connectionResources"/>
      </links>
    </connectionResources>
  </providerConnection>
</items>
```
Using the Conductor REST APIs

</links>
<items>
<connectionResource
resourceType="urn:oracle:wcps:conductor:provider:connectionResource">
<link capabilities="urn:oracle:restframework:read"
href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources"
rel="urn:oracle:webcenter:parent"
resourceType="urn:oracle:wcps:conductor:provider:connectionResources"
<link capabilities="urn:oracle:restframework:read"
rel='self'
resourceType="urn:oracle:wcps:conductor:provider:resourceMethods">
<link capabilities="urn:oracle:restframework:read"
rel="urn:oracle:webcenter:parent"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
</links>
<items>
<resourceMethod
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
Using the Conductor REST APIs

<links>
<link capabilities="urn:oracle:restframework:read"
rel="urn:oracle:webcenter:parent"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
<link capabilities="urn:oracle:restframework:read"
rel="self"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod"
</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>
  </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 exist, 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>
    <parameter>
      <localizedDescription>The property name.</localizedDescription>
      <name>property</name>
      <type>java.lang.String</type>
    </parameter>
  </parameters>
  <returnType>java.io.Serializable</returnType>
</resourceMethod>
Using the Conductor REST APIs

  rel="urn:oracle:webcenter:parent"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
<link capabilities="urn:oracle:restframework:read"
rel="self"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
<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">
<link capabilities="urn:oracle:restframework:read"
<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
Using the Conductor REST APIs

Conductor API Reference

72-39

<resourceMethod resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
</links>

<nombre>GetPropertySet</nombre>
<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>
Using the Conductor REST APIs

```xml
<name>getPropertySets</name>
<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>

resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
  <links>
    <link capabilities="urn:oracle:restframework:read"
    <link capabilities="urn:oracle:restframework:read"
  </links>
  <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>
```

72-40  Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper
Using the Conductor REST APIs

```xml
<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>
</connectionResource>
</items>
</providerConnections>
</dataProvider>

**JSON Request:**

GET
```
Using the Conductor REST APIs

Response:
Status Code: 200
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie: JSESSIONID=GvbXM27B2xRJQqvn9Qmsn1j2GIvzhc67hpLg4gnjhjsjynTTm1822326988;
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" : [{
        "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."
        }]
      }, {
        "type" : "java.lang.String",
        "name" : "namespace",
        "localizedDescription" : "The namespace in which the property set exists\nnull will be returned."
      }]
    }]
  }]
  "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\nnull will be returned."
}
  "rel" : "urn:oracle:webcenter:parent"
}, {
  "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
  "capabilities" : "urn:oracle:restframework:read",
  "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\nnull will be returned."
, resourceType : "urn:oracle:wcps:conductor:provider:resourceMethod",
links : [
  {
    resourceType : "urn:oracle:wcps:conductor:provider:resourceMethods",
    capabilities : "urn:oracle:restframework:read",
    rel : "urn:oracle:webcenter:parent"
  },
  {
    resourceType : "urn:oracle:wcps:conductor:provider:resourceMethod",
    capabilities : "urn:oracle:restframework:read",
    href :
   "rel" : "self"
 ]
 },
 "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",
   "capabilities" : "urn:oracle:restframework:read",
   "rel" : "urn:oracle:webcenter:parent"
 ],
 "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",
   "capabilities" : "urn:oracle:restframework:read",
   "rel" : "self"
 } ]
},

Using the Conductor REST APIs

propertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods?projection={projection}
  
  "capabilities": "urn:oracle:restframework:read",
  "href":
  propertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
  Resource/methods",
  "rel": "urn:oracle:webcenter:parent"
},
  
},
  
"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."
  
],
"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":
  propertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
  Resource/methods?projection={projection}",
  "capabilities": "urn:oracle:restframework:read",
  "href":
  propertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
  Resource/methods",
  "rel": "urn:oracle:webcenter:parent"
},
  
},
  
"resourceType": "urn:oracle:wcps:conductor:provider:resourceMethod",
"template":
propertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods?projection={projection}",
"capabilities": "urn:oracle:restframework:read",
"href":
propertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods",
"rel": "urn:oracle:webcenter:parent"
}
Using the Conductor REST APIs

```json
{
  "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."
    }
  ],
  "localizedDescription" : "Retrieves the 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."
}
```
"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",
   "capabilities" : "urn:oracle:restframework:read",
   "rel" : "urn:oracle:webcenter:parent"
}, {
   "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
   "capabilities" : "urn:oracle:restframework:read",
   "rel" : "self"
} ]
},
"resourceType" : "urn:oracle:wcps:conductor:provider:connectionResource",
"capabilities" : "urn:oracle:restframework:read",
"rel" : "urn:oracle:webcenter:parent"
}, {
   "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
   "capabilities" : "urn:oracle:restframework:read",
   "rel" : "self"
Using the Conductor REST APIs

...
Using the Conductor REST APIs

'rel' : 'self'
} ]
],
'resourceType' : 'urn:oracle:wcps:conductor:provider:dataProvider",
'links' : [ {
'resourceType' : 'urn:oracle:wcps:conductor:provider:dataProviders",
'capabilities' : 'urn:oracle:restframework:read",
'rel' : 'urn:oracle:webcenter:parent'
}, {
'resourceType' : 'urn:oracle:wcps:conductor:provider:dataProvider",
'capabilities' : 'urn:oracle:restframework:read",
'rel' : 'self'
}, {
'resourceType' : 'urn:oracle:wcps:conductor:provider:providerConnections",
'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/{provider-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/{provider-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/{provider-name}/connections/{connection-name}/resources/{resource-name}?projection={projection}',
'capabilities' : 'urn:oracle:restframework:read",
}, {
'resourceType' : 'urn:oracle:wcps:conductor:provider:resourceMethods",
'template' : 'http://localhost:7001/wcps/api/conductor/namespaces/{namespace}/dataProviders/{provider-name}/connections/{connection-name}/resources/{resource-name}/methods?projection={projection}',
'capabilities' : 'urn:oracle:restframework:read",
'rel' : 'urn:oracle:wcps:conductor:provider:resourceMethods'
72.2.2 Using the Function Provider Management APIs

This section provides examples of using the Conductor's function provider APIs, and contains the following subsections:

- Section 72.2.2.1, "Getting a Collection of Function Providers (GET)"
- Section 72.2.2.2, "Getting Metadata for a Single Function Provider (GET)"

72.2.2.1 Getting a Collection of Function Providers (GET)

XML

Request:

GET
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=1NxqM2GHF76VydXnQc.trgCJ7FmxMLD3pPrRCgPrCLGRJ0k5!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"
resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
</links>
<items>
<functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
<link>
</link>
</functionProvider>
</items>
</functionProviders>
Using the Conductor REST APIs

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders"
   rel="urn:oracle:webcenter:parent"
   resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
   <link capabilities="urn:oracle:restframework:read"
   rel="self"
   resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
   <links>
   <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/functionProviders/agfunction:getContentIDs?projection={projection}/">
   <links>
   <link capabilities="urn:oracle:restframework:read"
   <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>
</links>
</links>
</functionProvider>
</category>
</example>
</localizedDescription>
Using the Conductor REST APIs

Conductor API Reference

72-53

resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
</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/functionProviders/agfunction:getContentIDsFiltered?projection={projection}"/>
</links>
<category>agfunction</category>
<example>${agfunction:getContentIDsFiltered(applyResults,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">
<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"
</link>
<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">
<link capabilities="urn:oracle:restframework:read"
href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/agfunction:getContentIDsFiltered"/>
</link>
<category>agfunction</category>
<example>${agfunction:getContentIDsFiltered(applyResults,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">
<link capabilities="urn:oracle:restframework:read"
</link>
<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>
Conductor API Reference

Using the Conductor REST APIs

```xml
<functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
    <links>
        <link capabilities="urn:oracle:restframework:read"
        <link capabilities="urn:oracle:webcenter:parent"
        rel="self"
        resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
    </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"
        <link capabilities="urn:oracle:webcenter:parent"
        rel="self"
        resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
    </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"
        <link capabilities="urn:oracle:webcenter:parent"
        rel="self"
        resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
    </links>
    <category>cmisfunction</category>
    <example>${cmisfunction:getCMISQueryForDocIDsFromFullID(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>getCMISQueryForDocIDsFromFullID</name>
    <parameters/>
    <returnType>java.lang.String</returnType>
</functionProvider>
```
Construct a CMIS query in the form of 'IN' query syntax that will retrieve the documents for the array of input doc IDs.

```xml
<functionProvider
    resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
    <links>
      <link capabilities="urn:oracle:restframework:read"
      <link capabilities="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
    </links>
    <category>collections</category>
    <example>${collections:append(collection,item)}</example>
    <localizedDescription>Appends </localizedDescription>
    <name>append</name>
    <parameters/>
    <returnType>java.util.Collection</returnType>
</functionProvider>
```

```xml
<functionProvider
    resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
    <links>
      <link capabilities="urn:oracle:restframework:read"
      <link capabilities="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
    </links>
    <category>collections</category>
    <example>${collections:new()}</example>
    </localisedDescription>
    <name>new</name>
    <parameters/>
    <returnType>java.util.Collection</returnType>
</functionProvider>
```
<localizedDescription>Creates and returns a new collection.</localizedDescription>

<name>new</name>
<parameters/>
<returnType>java.util.Collection</returnType>

<functionProvider>
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">


<example>${collections:sort(list)}</example>
<localizedDescription>Sorts the provided collection.</localizedDescription>

<name>sort</name>
<parameters/>
<returnType>void</returnType>

<functionProvider>
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">

<example>${connectionsQA:getSampleConfig(scenarioContext,connectionName)}</example>
<localizedDescription>Retrieves a sample configuration by name.</localizedDescription>

<name>getSampleConfig</name>
Using the Conductor REST APIs

<parameters/>

<returnType>oracle.wcps.test.providers.sample.SampleConnectionConfigImpl</returnType>
</functionProvider>
</functionProvider>

<functionProvider resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
<links>
</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>
</links>
<category>connectionsQA</category>
<example>${connectionsQA:removeSampleConfig(scenarioContext,connectionName)}</example>
<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"
   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>
<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"
   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>${magiceightball:response(scenarioContext,responses)}</example>
<localizedDescription>Provides clairvoyant responses to all your
questions.</localizedDescription>
</name>
<name>response</name>
<functionProvider>
  resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
  <links>
    <link capabilities="urn:oracle:restframework:read"
    <link rel="urn:oracle:webcenter:parent"
      resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
  </links>
  <category>security</category>
  <example>${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"
    <link rel="urn:oracle:webcenter:parent"
      resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
  </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>
Using the Conductor REST APIs

```
<functionProvider resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
  <links>
  </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>
  </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>
```

Conductor API Reference  72-61
Using the Conductor REST APIs

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
rel="urn:oracle:webcenter:parent"
resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
<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"
<link capabilities="urn:oracle:restframework:read">
<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>

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
test:concat"
rel="self"
resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
<link capabilities="urn:oracle:restframework:read">
<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>

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
test:concat"
Using the Conductor REST APIs

Conductor API Reference 72-63

  <link capabilities="urn:oracle:restframework:read"
    rel="self"
    resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
</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
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=Q9VFM2GLn2ny2CzbxKvxxXYYbYTMT4zvJ3MQP2DxxH0GSkfct76cC!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",
        'capabilities' : "urn:oracle:restframework:read",
        'rel' : "urn:oracle:webcenter:parent"}]
    }]
}
Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper

Using the Conductor REST APIs

```
],
  "parameters" : [ ],
  "name" : "getCMISLinksFromCommonItems",
  "returnType" : "java.util.List",
  "category" : "agfunction",
  "example" : "${agfunction:getCMISLinksFromCommonItems(results)}",
  "localizedDescription" : "Return a List of URLs representing the common items
as CMIS objects",
  "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
  "links" : [ {
    "resourceType" : "urn:oracle:wcps:conductor:provider:allfunctionProviders",
    "capabilities" : "urn:oracle:restframework:read",
    "rel" : "urn:oracle:webcenter:parent"
  },
  "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
  "capabilities" : "urn:oracle:restframework:read",
  "rel" : "self"
} ]
},
  "parameters" : [ ],
  "name" : "getCMISLinksFromRecommendations",
  "returnType" : "java.util.List",
  "category" : "agfunction",
  "example" : "${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",
    "capabilities" : "urn:oracle:restframework:read",
    "rel" : "urn:oracle:webcenter:parent"
  },
  "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
  "capabilities" : "urn:oracle:restframework:read",
  "rel" : "self"
} ]
```
Using the Conductor REST APIs

Conductor API Reference 72-65

'parameters' : [ ],
'name' : 'getContentTypeIDs",
'returnType' : 'java.util.List",
'category' : 'agfunction",
'example' : '($agfunction:getContentTypeIDs(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' : [ ]
],
'parameters' : [ ],
'name' : 'getContentTypeIDsFiltered",
'returnType' : 'java.util.List",
'category' : 'agfunction",
'example' : '$(agfunction:getContentTypeIDsFiltered(agResults,excludeClassURN))",
'localizedDescription' : 'Extract content IDs from input objects'

'resourceType' : 'urn:oracle:wcps:conductor:provider:functionProvider",
'links' : [ ]
]
"capabilities": "urn:oracle:restframework:read",
"href":
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/agfunction:getContentIDsFiltered",
"rel": "self"
}]
}, {
"parameters": [],
"name": "atomAsCMISObjects",
"returnType": "java.util.List",
"category": "cmisfunction",
"example": "$\{\text{cmisfunction:atomAsCMISObjects\{atomFeed\}}\}$",
"localizedDescription": "ATOM feed as List\<CMISObject\>",
"resourceType": "urn:oracle:wcps:conductor:provider:functionProvider",
"links": [
"resourceType": "urn:oracle:wcps:conductor:provider:functionProviders",
"template":
"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":
"capabilities": "urn:oracle:restframework:read",
"href":
"rel": "self"
}]
}, {
"parameters": [],
"name": "atomAsEntries",
"returnType": "java.util.List",
"category": "cmisfunction",
"example": "$\{\text{cmisfunction:atomAsEntries\{atomFeed\}}\}$",
"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":
"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":
"capabilities": "urn:oracle:restframework:read",
"href":
Using the Conductor REST APIs

Conductor API Reference

72-67

'rel' : 'self'
}
],
], {
'parameters' : [ ],
'name' : 'atomAsFeed',
'returnType' : 'org.apache.abdera.model.Feed',
'category' : 'cmisfunction',
'exaple' : '${cmisfunction:atomAsFeed(atomFeed)}',
'localizedDescription' : 'ATOM feed as org.apache.abdera.model.Feed',
'resourceType' : 'urn:oracle:wcps:conductor:provider:functionProvider',
'links' : [ {
'resourceType' : 'urn:oracle:wcps:conductor:provider:functionProviders',
'capabilities' : 'urn:oracle:restframework:read',
'rel' : 'urn:oracle:webcenter:parent'
}, {
'resourceType' : 'urn:oracle:wcps:conductor:provider:functionProvider',
'capabilities' : 'urn:oracle:restframework:read',
'rel' : 'self'
}
],
], {
'parameters' : [ ],
'name' : 'atomAsStreamUrls',
'returnType' : 'java.util.List',
'category' : 'cmisfunction',
'exaple' : '${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',
'capabilities' : 'urn:oracle:restframework:read',
'rel' : 'urn:oracle:webcenter:parent'
}, {
'resourceType' : 'urn:oracle:wcps:conductor:provider:functionProvider',
'capabilities' : 'urn:oracle:restframework:read',
'rel' : 'self'
}
]
Using the Conductor REST APIs

"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?projection={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",
"capabilities": "urn:oracle:restframework:read",
"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?projection={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",
"capabilities": "urn:oracle:restframework:read",
"rel": "self"
}]

"parameters": []
"category": "collections",
"example": "${collections:append(collection,item)}",
"localizedDescription": "Appends ",
"resourceType": "urn:oracle:wcps:conductor:provider:functionProvider",
"links": [ {
  "resourceType": "urn:oracle:wcps:conductor:provider:functionProviders",
  "template": "http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?projection={projection}",
  "capabilities": "urn:oracle:restframework:read",
  "href": "http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
  "rel": "urn:oracle:webcenter:parent"
}, {
  "resourceType": "urn:oracle:wcps:conductor:provider:functionProvider",
  "capabilities": "urn:oracle:restframework:read",
  "rel": "self"
} ]
}, {
  "parameters": [ ],
  "name": "new",
  "returnType": "java.util.Collection",
  "category": "collections",
  "example": "${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?projection={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",
    "capabilities": "urn:oracle:restframework:read",
    "rel": "self"
  } ]
}, {
  "parameters": [ ],
  "name": "sort",
  "returnType": "void",
  "category": "collections",
  "example": "${collections:sort(list)}",
  "localizedDescription": "Sorts the provided collection.",
  "resourceType": "urn:oracle:wcps:conductor:provider:functionProvider",
"links" : [ {
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
    "capabilities" : "urn:oracle:restframework:read",
    "rel" : "urn:oracle:webcenter:parent"
}, {
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "capabilities" : "urn:oracle:restframework:read",
    "rel" : "self"
}
}, {
    "parameters" : [ ],
    "name" : "getSampleConfig",
    "returnType" : "oracle.wcps.test.providers.sample.SampleConnectionConfigImpl",
    "category" : "connectionsQA",
    "example" : "${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",
        "capabilities" : "urn:oracle:restframework:read",
        "rel" : "urn:oracle:webcenter:parent"
    }, {
        "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
        "capabilities" : "urn:oracle:restframework:read",
        "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",
        "capabilities" : "urn:oracle:restframework:read",
        "rel" : "urn:oracle:webcenter:parent"
    }, {
        "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
        "capabilities" : "urn:oracle:restframework:read",
        "rel" : "self"
    }
} ]
}
"template": "http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?projection={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",
  "capabilities": "urn:oracle:restframework:read",
  "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?projection={projection}",
    "capabilities": "urn:oracle:restframework:read",
    "href": "http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
    "rel": "urn:oracle:webcenter:parent"
  }
]

"parameters": [],

"name": "saveSampleConfig",

"returnType": "oracle.wcps.test.providers.sample.SampleConnectionConfigImpl",

"category": "connectionsQA",

"example": "$\{connectionsQA:saveSampleConfig(scenarioContext,connectionName,samplePropertyValue)\}"

"localizedDescription": "Creates or saves a sample configuration."

"resourceType": "urn:oracle:wcps:conductor:provider:functionProvider",

"links": [
  {
    "resourceType": "urn:oracle:wcps:conductor:provider:functionProviders",

\["a providers\].
Using the Conductor REST APIs

```
section={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/security:isUserInRole?projection={projection}",
   'capabilities' : "urn:oracle:restframework:read",
   'href' :
     'http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/security:isUserInRole",
   'rel' : 'self'
 }
 }, {
   'parameters' : [ ],
   'name' : 'isUserInRole',
   'returnType' : 'boolean',
   'category' : "security",
   'example' : '${security:isUserInRole(role)}",
   'localizedDescription' : "Determines if the current user is a member in the specified role or group name (role).",
   'resourceType' : "urn:oracle:wcps:conductor:provider:functionProvider",
   'links' : [ {
     'resourceType' : "urn:oracle:wcps:conductor:provider:functionProvider",
     'template' :
       'http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/security:isUserInRole?projection={projection}",
     'capabilities' : "urn:oracle:restframework:read",
     'href' :
       'http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/security:isUserInRole",
     'rel' : 'self'
   }]
 }, {
   'parameters' : [ ],
   'name' : 'complexType',
   'returnType' : 'oracle.wcps.test.providers.functions.ComplexType",
   'category' : "test",
   'example' : '${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:functionProvider",
     'template' :
       'http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?projection={projection}",
     'capabilities' : "urn:oracle:restframework:read",
     'href' :
       'http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders',
     'rel' : 'urn:oracle:webcenter:parent'
   }]
 }, {
   'parameters' : [ ],
   'name' : 'complexType',
   'returnType' : 'oracle.wcps.test.providers.functions.ComplexType",
   'category' : "test",
   'example' : '${test:complexType(arg0,arg1)}",
   'localizedDescription' : "Demonstrates how to return a complex type.",
   'resourceType' : "urn:oracle:wcps:conductor:provider:functionProvider",
   'links' : [ {
     'resourceType' : "urn:oracle:wcps:conductor:provider:functionProvider",
     'template' :
       'http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?projection={projection}",
     'capabilities' : "urn:oracle:restframework:read",
     'href' :
       'http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders',
     'rel' : 'urn:oracle:webcenter:parent'
   }]
 }, {
   'parameters' : [ ],
   'name' : 'complexType',
   'returnType' : 'oracle.wcps.test.providers.functions.ComplexType",
   'category' : "test",
   'example' : '${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:functionProvider",
     'template' :
       'http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?projection={projection}",
     'capabilities' : "urn:oracle:restframework:read",
     'href' :
       'http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders',
     'rel' : 'urn:oracle:webcenter:parent'
   }]
 }
```

"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" :
ection={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
    "rel" : "urn:oracle:webcenter:parent"
  }
  
), {
  "parameters" : [ ],
  "name" : "complexTypeMap",
  "returnType" : "java.util.Map",
  "category" : "test",
  "example" : "${test:complexTypeMap{arg0}}",
  "localizedDescription" : "Demonstrates how to return a map of complex types.",
  "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
  "links" : [ {
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/test:
complexTypeList?projection={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/test:
complexTypeList",
    "rel" : "self"
  } ]
), {
  "parameters" : [ ],
  "name" : "complexTypeMap",
  "returnType" : "java.util.Map",
  "category" : "test",
  "example" : "${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" :
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",
'capabilities' : "urn:oracle:restframework:read",
'rel' : 'self'
} ]
],
'parameters' : [ ],
'name' : 'concat',
'returnType' : 'java.lang.String',
'category' : 'test',
'exmaple' : '${test:concat(arg0,arg1)}',
'localizedDescription' : "Demonstrates how to return a complex type.",
'resourceType' : "urn:oracle:wcps:conductor:provider:functionProvider",
'links' : [ {
'resourceType' : "urn:oracle:wcps:conductor:provider:functionProviders",
'capabilities' : "urn:oracle:restframework:read",
'rel' : 'urn:oracle:webcenter:parent'
} ],
'parameters' : [ ],
'name' : 'sleep',
'returnType' : 'void',
'category' : 'test',
'exampel' : '${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",
'capabilities' : "urn:oracle:restframework:read",
'href' : "http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?
'rel' : 'urn:oracle:webcenter:parent'
} ]
},
'}
72.2.2.2 Getting Metadata for a Single Function Provider (GET)

**XML Request:**
GET
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=xf4NM2GL2yrTY3QPCZ591Ht8h7MvTSW9GGzR6Tm12WtFLvD0vmng!12098846330; 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">
<link capabilities="urn:oracle:restframework:read">
<link capabilities="urn:oracle:restframework:read">
<link capabilities="urn:oracle:restframework:read">
</link>
</link>
</functionProvider>
Using the Conductor REST APIs

\collections:append?projection=details*

rel='self'
resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
</links>
</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>
</parameter>
<parameter>
<localizedDescription>The item or object to append.</localizedDescription>
</name>item</name>
</parameter>
</parameters>
</returnType>java.util.Collection</returnType>
</functionProvider>

**JSON**

**Request:**

GET


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=BCz3M2GJWQnS3jQV1DvMsJynJXGnlJ6MrmrST4qpom8L7NxbgJh9T!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"
}
Using the Conductor REST APIs

72.2.3 Using the Namespace Management APIs

This section provides examples of using the Conductor's namespace management APIs, and contains the following subsections:

- Section 72.2.3.1, "Getting a Collection of Namespaces (GET)"
- Section 72.2.3.2, "Retrieving a Single Namespace (GET)"
- Section 72.2.3.3, "Creating a Namespace (POST)"
- Section 72.2.3.4, "Deleting a Namespace (DELETE)"

72.2.3.1 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=JVqNM0HGTxxrXJJsay2cLCPjRTqJzDQ6pFytM0By3JQTph671rTC21-1838429496; 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">
<link capabilities="urn:oracle:restframework:read">
</link>
</namespaces>
Using the Conductor REST APIs

**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=3yvMM0HLChkXHS7R2VGn4F61FKHw7GPP2NhjC2Ll7J2jpvLTW9GvV!-1838429496;
```

Transfer-Encoding: chunked

Content-Type: application/json

Date: Tue, 12 Oct 2010 20:40:11 GMT

```json
{
  "namespaces" : [
    {
      "name" : "JDEV",
      "resourceType" : "urn:oracle:wcps:conductor:namespace",
      "links" : [
        {
          "resourceType" : "urn:oracle:wcps:conductor:namespaces",
          "capabilities" : "urn:oracle:restframework:read",
          "rel" : "urn:oracle:webcenter:parent"
        }
      ]
    }
  ]
}
```
Using the Conductor REST APIs

Conductor API Reference 72-81

- "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}&projection={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?projection={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}&projection={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",
"capabilities" : "urn:oracle:restframework:read",
"rel" : "urn:oracle:wcps:conductor:provider:dataProviders"
},
{"resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
"capabilities" : "urn:oracle:restframework:read",
"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}&itemsPerPage={itemsPerPage}",
"capabilities" : "urn:oracle:restframework:read urn:oracle:restframework:create",
"href" : "http://localhost:7001/wcps/api/conductor/namespaces?startIndex=0&itemsPerPage=10",
"rel" : "self"
} ]

72.2.3.2 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=qzMqM0HMPYB1KQGMJvqQMS4rX54Jn1QnBjW850G1vnSjt41HbLy!-1838429496;
pth=/wcps; HttpOnly
Transfer-Encoding: chunked
Content-Type: application/json
Date: Tue, 12 Oct 2010 20:40:12 GMT
Using the Conductor REST APIs

**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=qzMqM0HMPYBlKQGMJvnQMS4rxXS4Jn1QnBjw850G1vmSjt41HbLy!-1838429496; path=/wcps; HttpOnly
Transfer-Encoding: chunked
Content-Type: application/json
Using the Conductor REST APIs

72.2.3.3 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=RW6QM0HF5x2X1vSmN3JVncSpSXbl1lrKY3tTffxH94yY6Qv6ddGJyY!-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:namespaces"/>
    <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={itemsPerPage}"/>
    <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"
    <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"
  </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
Using the Conductor REST APIs

```json
{
  "name" : "Oracle",
  "resourceType" : "urn:oracle:wcps:conductor:namespace",
  "links" : [
    {
      "resourceType" : "urn:oracle:wcps:conductor:namespaces",
      "capabilities" : "urn:oracle:restframework:read",
      "rel" : "urn:oracle:webcenter:parent"
    },
    {
      "resourceType" : "urn:oracle:wcps:conductor:namespace",
      "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:delete",
      "rel" : "self"
    },
    {
      "resourceType" : "urn:oracle:wcps:conductor:scenarioMetadatas",
      "template" : "http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios?tag={tag}&projection={projection}&startIndex={startIndex}&itemsPerPage={itemsPerPage}",
      "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:create",
      "rel" : "urn:oracle:wcps:conductor:scenarioMetadatas"
    },
    {
      "resourceType" : "urn:oracle:wcps:conductor:provider:dataProviders",
      "capabilities" : "urn:oracle:restframework:read",
      "rel" : "urn:oracle:wcps:conductor:provider:dataProviders"
    },
    {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "capabilities" : "urn:oracle:restframework:read",
      "rel" : "urn:oracle:wcps:conductor:provider:functionProviders"
  ]
}
```

Response:
Status Code:201
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie: JSESSIONID=xh818M0HKT86f4y1Wq28mzHXvNnhv0p7y9RLTmQfRQC3fh79gx!-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
72.2.3.4 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=SMctM0HJxYxsVkfMrzjrfp15zJ3Q0v25NhB7QPRtnJFJmsdtK1hYC!=1838429496; path=/wcps; HttpOnly
Date: Tue, 12 Oct 2010 20:40:09 GMT

72.2.4 Using the Scenario Management APIs

This section provides examples of using the Conductor's scenario management APIs, and contains the following subsections:

- Section 72.2.4.1, "Getting a Collection of Scenarios (GET)"
- Section 72.2.4.2, "Retrieving a Single Scenario (GET)"
- Section 72.2.4.3, "Creating a Scenario (POST)"
- Section 72.2.4.4, "Updating a Scenario (PUT)"
- Section 72.2.4.5, "Deleting a Scenario (DELETE)"
- Section 72.2.4.6, "Execute a Scenario (POST)"

72.2.4.1 Getting a Collection of Scenarios (GET)

XML

Request:
GET
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios?projection=summary&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=LJq1M1TGzp7fClLyTJzwwgGczyv2r8N0X1jsPY1g&kpGFWwGMvrQeM121637120; path=/wcps; HttpOnly
Content-Type: application/xml
<?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?projection=summary&startIndex=0&itemsPerPage=10"
            rel="self"
            resourceType="urn:oracle:wcps:conductor:scenarioMetadatas"
            template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios?tag={tag}&projection={projection}&startIndex={startIndex}&itemsPerPage={itemsPerPage}"/>
    </links>
    <items>
        <scenarioMetadata resourceType="urn:oracle:wcps:conductor:scenarioMetadata">
            <links>
                <link
                    href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/HelloWorld"
                    rel="self"
                    resourceType="urn:oracle:wcps:conductor:scenarioMetadata"
                    template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/HelloWorld?projection={projection}&startIndex={startIndex}&itemsPerPage={itemsPerPage}"/>
            </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=summary&startIndex=0&itemsPerPage=10
Using the Conductor REST APIs

Response:

Status Code: 200
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie: JSESSIONID=0VBvM2ZMwfWC6bPn28QDv7q4h7WhmTMPvb0Lyc9n8t0lnyTtGly!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",
    'rel' : 'self'
    },
    'resourceType' : "urn:oracle:wcps:conductor:scenario",
    'capabilities' : "urn:oracle:restframework:read",
    'rel' : 'alternate'
  } ]
} ]

'resourceType' : "urn:oracle:wcps:conductor:scenarioMetadatas",
'links' : [ { 
  'resourceType' : "urn:oracle:wcps:conductor:scenarioMetadatas",
ojection={projection}&startIndex={startIndex}&itemsPerPage={itemsPerPage}",
  'capabilities' : "urn:oracle:restframework:read
urn:oracle:restframework:create",
} ]

Conductor API Reference  72-89
72.2.4.2 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: JSESSID=1mS7LM2hYWdd8xvMKGXGlXJ2qYD7XNL3hy3lqknyp8fYSt1BQNT!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%20World?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/Hello%20World?projection={projection}&amp;startIndex={startIndex}&amp;itemsPerPage={itemsPerPage}"/>
    <link capabilities="urn:oracle:restframework:read"
      href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%20World/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>
  </scenario>
</scenarioMetadata>
Using the Conductor REST APIs

Conductor API Reference

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=LJnKM2hpVQL9phB8JWCymz8M1TQqcQjSbD03zJQ9z6yyT314Zy4!822326988;
path=/wcps; HttpOnly
Transfer-Encoding: chunked
Content-Type: application/json
Date: Wed, 13 Oct 2010 20:45:50 GMT

```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."
    },
    "status" : 'PUBLISHED',
    "createdAuthor" : "weblogic",
    "createdDateTime" : "2010-10-13T14:45:45.8681-0600",
    "updatedAuthor" : "weblogic",
    "updatedDateTime" : "2010-10-13T14:45:45.8681-0600",
}
```
72.2.4.3 Creating a Scenario (POST)

XML Request:

POST
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios?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>
      <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>
Using the Conductor REST APIs

Response:
Status Code: 201
Content-Length: 1669
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie: JSESSIONID=L41NM2hW6p8TvJmJrx9kZNF2IJoynY4JhGJCTx3DG4prwfl1Tg3D!823226988; 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
      href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%20World"
      rel='self'
      resourceType="urn:oracle:wcps:conductor:scenarioMetadata"
      template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%20World?projection={projection}&startIndex={startIndex}&itemsPerPage={itemsPerPage}"/>
    <link capabilities="urn:oracle:restframework:read"
      href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%20World/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>
Using the Conductor REST APIs

JSON

Request:
POST
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios?projection=details&startIndex=0&itemsPerPage=10
accept-language: en
content-type: application/json
accept: application/json

```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=XrT7M2hcNRzpMcK7SxPm9phBvPlsnfFy98nRLDL8G52WhWd1y!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

```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."
      } ]
    }
  }
}
72.2.4.4 Updating a Scenario (PUT)

**XML Request:**
```
PUT http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%20World
```

```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>
            </return>
        </body>
    </scenario>
</scenarioMetadata>
```
Using the Conductor REST APIs

```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">
    <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%20World"
rel="self"
resourceType="urn:oracle:wcps:conductor:scenarioMetadata"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello\nWorld?projection={projection}&startIndex={startIndex}&itemsPerPage={itemsPerPage}"/>
        <link capabilities="urn:oracle:restframework:read"
href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%20World/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>
        </body>
    </scenario>
</scenarioMetadata>
```

Response:
Status Code: 200
Content-Length: 1777
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie: JSESSIONID=CnGYM2hYBBJ2hNPGLkcdG21WvhP41gBDvBsCwmKpdHvtCJtQL67L!822326988;
path=/wcps; HttpOnly
Content-Type: application/xml
Date: Wed, 13 Oct 2010 20:45:44 GMT

```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">
    <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%20World"
rel="self"
resourceType="urn:oracle:wcps:conductor:scenarioMetadata"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello\nWorld?projection={projection}&startIndex={startIndex}&itemsPerPage={itemsPerPage}"/>
        <link capabilities="urn:oracle:restframework:read"
href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%20World/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 '${input}'!</expression>
            </assign-variable>
        </body>
    </scenario>
</scenarioMetadata>
```
<return>
  <comments>This is what is returned by the scenario. A static string with the input parameter embedded.</comments>
  <expression>${initVariable}</expression>
</return>

<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=sJBGM2hpTwwW713q7HNnQ81DpjkgQvkJy4h1Wrl1KJFxlgd2XDSY1822326988; path=/wcps; HttpOnly
Transfer-Encoding: chunked
Content-Type: application/json
Using the Conductor REST APIs

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 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/HelloWorld?projection={projection}&startIndex={startIndex}&itemsPerPage={itemsPerPage}"
        }, { 
            "resourceType": "urn:oracle:wcps:conductor:scenario",
            "href": "http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/HelloWorld/definition",
            "rel": "self"
        }, { 
            "resourceType": "urn:oracle:wcps:conductor:scenario",
            "capabilities": "urn:oracle:restframework:read",
            "href": "http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/HelloWorld/definition",
            "rel": "alternate"
    } ]
}

72.2.4.5 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=nyKyM2WdGL6v6Qagpn335Vgzs5b7vmS417p9dcTGr1v3736mmm!822326988;
path=/wcps; HttpOnly
Date: Wed, 13 Oct 2010 20:28:45 GMT

72.2.4.6 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=FlFzM2WchrvP59qBJdbSQldb7TvPQPyRj1HsvvJPC0hKrj7z6zs8!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
Using Expression Language (EL) expressions, you can call data integration services from remote clients such as web pages and Java Server Faces environments. You can, for example, use ELs to apply data integration scenarios to pages in WebCenter Portal. For more information about data integration ELs, see also Section G.11, "ELs Related to Personalization."

This section contains the following subsections:

- Section 72.3.1, "Calling Data Integration Client Services from JSF Pages"
- Section 72.3.2, "Calling Data Integration Client Services from JSP Pages"
- Section 72.3.3, "Data Integration Context Object Method Reference"
- Section 72.3.4, "Enabling Single Sign-on"

### 72.3.1 Calling Data Integration Client Services from JSF Pages

The P13NContext 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:

```el
#{p13nContext}
```

**Accessing the Conductor context**

Access to remote Conductor services can be accessed in the following way:

```el
#{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['MyPropertySetDefinition'].sets['MyPropertySet'].results['MyPropertyName']}

**Configuring the client services for request scope**

You may sometimes need to have the data integration 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:

```xml
<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>
```

### 72.3.2 Calling Data Integration Client Services from JSP Pages

The data integration client can be used with JSP pages similarly to the way it is used in Java Server Faces pages (see Section 72.3.1, "Calling Data Integration Client Services from JSF Pages"). You must, however, register the bean using `<jsp:useBean>` as shown in the following example:

```jsp
<jsp:useBean id="p13nContext" class="oracle.wcps.client.PersonalizationContext" scope="session"/>
<c:out value="\${p13nContext.conductor[\"http://weblogic:webl0gic@localhost:7001/wcps/api/conductor\"].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
```

### 72.3.3 Data Integration Context Object Method Reference

Table 72–2 shows the data integration context object methods.
### Table 72–2  Data Integration Context Object Method Reference

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
</table>
| reset       | Clears the state of the context object in which it is invoked. | `#{p13nContext.reset}`
|             |             | `#{p13nContext.conductor['ConductorConnection'].reset}`
|             |             | `#{p13nContext.conductor['ConductorConnection'].namespace['MyNamespace'].reset}`
|             |             | `#{p13nContext.properties['PropertiesConnection'].namespaces['MyNamespace'].setDefinitions['MyPropertySetDefinition'].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'].namespace['MyNamespace'].scenario['MyScenario'].results}`
|             |             | `#{p13nContext.conductor['ConductorConnection'].namespace['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'].namespace['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'].namespace['MyManagedBean'].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'].namespace['MyManagedBean'].scenario['myManagedBean.scenario'].errorMessage}`
|             |             | `#{p13nContext.properties['PropertiesConnection'].namespaces['MyNamespace'].setDefinitions['MyPropertySetDefinition'].set['MyPropertySet'].errorMessage}` |
72.3.4 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:

```xml
<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.
73 Cache Management for Personalization

This chapter discusses cache management for Oracle WebCenter Personalization.

This chapter includes the following topics:

- Section 73.1, "Introduction to Cache Management for Personalization"
- Section 73.2, "Locating and Updating Cache Configuration Files"
- Section 73.3, "Using Caching with Personalization Components"
- Section 73.4, "Configuring Caches for Personalization Components"
- Section 73.5, "Configuring a Cache on a Data Provider Connection"
- Section 73.6, "Monitoring Oracle Coherence"

73.1 Introduction to Cache Management for Personalization

WebCenter Portal uses Oracle Coherence to manage server-side caching for personalization features. A local Coherence license is provided with all WebCenter installations. Personalization supports other more complex modes of Coherence use, although appropriate Coherence licenses must be purchased for those cases.

This chapter explains how to use caching with WebCenter Portal personalization features, like scenarios and data providers.

Note: It is assumed that you have a basic understanding of caching concepts and techniques. In general, no attempt will be made in this chapter to explain these concepts. It is also assumed that you are familiar with Oracle Coherence. To learn about Coherence, see Getting Started with Oracle Coherence.

It is also assumed that you are familiar with the WebCenter Portal personalization concepts and features described in Chapter 66, "Personalizing Oracle WebCenter Portal Applications."

73.2 Locating and Updating Cache Configuration Files

WebCenter Portal provides two Coherence cache configuration files for use with personalization. The use of these files is described in this section.

Note: All cache factories instantiated by WebCenter Personalization Server (WCPS) are specific to the WCPS classloader.
73.2.1 Where are the Cache Configuration Files Located?

Cache configuration files are located in:

$ORACLE_HOME/user_projects/applications/<domain>/conductor-extensions-library/WEB-INF/classes

If you create any custom cache configuration files, the best practice is to place them in this same directory. On the other hand, you can place them in any location that is in the WCPS web application classpath. You need to redeploy the WCPS web application or restart the server for the new class to be recognized. See also Section 73.2.2.3, "Creating Custom Cache Configuration Files."

73.2.2 Using the Cache Configuration Files

This section discusses each of the cache configuration files used by personalization features and the creation of custom cache configuration files.

73.2.2.1 Internal Cache Configuration File

The cache configuration file `wcps-coherence-cache-config.xml` supports internal integration components like the Property Service, Conductor, and the core data provider functionality.

---

**Note:** Do not change the names of the caches in this configuration file. You can, however, change the cache configuration parameters in the file.

---

73.2.2.2 Data Provider Cache File

The cache configuration file `wcps-dataprovider-cache-config.xml` addresses caching associated with data providers. This file uses the wildcard pattern so that clients can have their specific caches, either by using a pattern like `wcps-dp-small-content` or `my-custom-cache`. For example, a wildcard pattern such as `wcps-dp-medium-*` will resolved when specifying cache names such as `wcps-dp-small-content` or `wcps-dp-small-ebs`.

The out-of-the-box data provider configurations may be modified to suit your needs, or you may add new entries to support your custom data providers. Please note that any changes to existing out-of-the-box configurations may affect other custom data providers deployed to your environment.

Default data provider cache configurations include the following:

- Local cache implemented in memory
- LRU eviction policy
- Small: 100 entries, default TTL of 1 hour
- Medium: 1000 entries, default TTL of 1 hour
- Large: 10000 entries, default TTL of 1 hour

---

**Note:** These configurations may be changed to support upgraded Coherence license features.
This cache configuration file takes advantage of Coherence parameter macros and can easily be modified to fit specific needs. Links to Coherence documentation specific to the parameters are provided in the configuration file itself.

Clients can configure additional entries in this configuration file. Those entries will be available to the personalization cache APIs. See Section 73.3, "Using Caching with Personalization Components."

### 73.2.2.3 Creating Custom Cache Configuration Files

You can create custom Coherence cache configuration files to address specific use cases. Oracle recommends that all Coherence cache configuration files adhere to these general rules:

- Place cache configuration files in:
  
  ```
  $ORACLE_HOME/user_projects/applications/<domain>/conductor-extensions-library/WEB-INF/classes
  ```

- Only refer to caches that are used by server-side personalization components, like data and function providers. For details on these components, see Chapter 67, "Implementing Custom Data Providers: Introduction" and Chapter 68, "Implementing Custom Function Providers: Introduction."

  **Note:** In reality, cache configuration files can be placed wherever they are visible to the WCPS web application classloader; placing it in the above directory keeps it consistent with other WCPS cache configuration files. However, there may be times when you wish to define and package your cache configuration file differently.

You can package cache configuration files as part of the data provider's JAR file, or anywhere on the file system, because the CoherenceCacheFactory loads files using a URI. CoherenceCacheFactory is used programmatically by data providers or any other class deployed to the WCPS server. See Section 73.3.2, "Using CoherenceCacheFactory."

  **Note:** Data provider JAR files are located in:

  ```
  $DOMAIN_HOME/conductor-extensions-library/WEB-INF/lib
  ```

  See Chapter 67, "Implementing Custom Data Providers: Introduction" for more information on packaging and deploying data providers.

You must redeploy the WCPS web application or restart the server for the new classes to be recognized.

### 73.3 Using Caching with Personalization Components

Personalization components like data and function providers, and custom data providers, interact with Coherence in one of two ways, depending on the context.

- **CacheFunctionProvider** – Includes methods that can be used within the context of a scenario or used programmatically by any data provider. Function providers perform operations on data within the context of a scenario. See also Section 68.1, "Introduction to Function Providers."

- **CoherenceCacheFactory** – Data providers and other classes deployed to the WCPS use this factory class to implement caching programmatically.
73.3.1 The Cache Function Provider

The CacheFunctionProvider provides methods for managing caches from within a scenario. This function provider delegates to the CoherenceCacheFactory class and includes several convenience methods.

The methods of the Cache Function Provider are described in "Cache Methods" in Chapter 69, "Function Provider Reference."

The prefix for the CacheFunctionProvider is cache (the name that appears in the Expression Builder dialogs in the scenario editor). Figure 73–1 shows the cache function provider in the Expression Builder dialog.

Figure 73–1  The Cache Function Provider

Example 73–1 lists the XML structure of a scenario that uses the Cache Function Provider method getMediumCache(). First, a cache is returned and stored in a variable. Then, that variable is used to return a specific value from the cache.

Example 73–1  Using the Cache Function Provider in a Scenario

```xml
<!-- Check cache first, using CacheFunctionProvider -->
<assign-variable>
  <variable>userIdCache</variable>
  <expression>${cache:getMediumCache('userIdCache')}</expression>
</assign-variable>

<assign-variable>
  <variable>userId</variable>
  <expression>${cache:get(userIdCache,username)}</expression>
</assign-variable>
```
Configuring Caches for Personalization Components

Example 73–2 shows how the CacheFunctionProvider methods can be called programmatically. Be sure to import:
oracle.wcps.dataprovider.base.CacheFunctionProvider.

**Example 73–2 Using the Cache Function Provider Methods in a Java Class**

```java
NamedCache myCache = CacheFunctionProvider.getMediumCache("userIdCache");
String userId = (String)myCache.get("jpalmer");
```

**Note:** Both examples Example 73–1 and Example 73–2 return a cache named wcps-cp-medium-userIdCache. If this cache does not already exist, it will be created.

Example 73–3 shows an example where a custom cache is used. In this case, the returned cache will not be the full cache name without a prefix.

**Example 73–3 Using the Cache Function Provider Methods in a Java Class**

```java
NamedCache myCache = CacheFunctionProvider.getCustomCache("my-custom-cache");
```

### 73.3.2 Using CoherenceCacheFactory

Data providers and other classes deployed to the WCPS use this factory class to implement caching programmatically. This class is scoped to the web application with which it is deployed.

The CoherenceCacheFactory has a single method to call:

```java
public static NamedCache getCache(String cacheConfigFile, String cacheName)
```

**Example 73–4 shows how to call getCache() when the cache configuration file is in the WCPS classpath, according to the best practice described previously in Section 73.2.2.3, "Creating Custom Cache Configuration Files."

**Example 73–4 Calling getCache() with Configuration File in the Classpath**

```java
NamedCache myCache = CoherenceCacheFactory.getCache("content-cache-config.xml", "userIdCache");
String userId = (String)myCache.get("jpalmer");
```

**Example 73–5 shows how to call getCache() when the cache configuration file is located elsewhere on the file system.**

**Example 73–5 Calling getCache() with Configuration File on the Filesystem**

```java
NamedCache myCache = CoherenceCacheFactory.getCache("file:///myconfig/content-cache-config.xml", "userIdCache");
String userId = (String)myCache.get("jpalmer");
```

### 73.4 Configuring Caches for Personalization Components

This section explains how caches are configured in configuration files. Pre-configured and custom caches configured in wcps-dataprovider-cache-config.xml are discussed. Then, cases where a cache has its own configuration file is explained.

- Section 73.4.1, "Pre-configured Caches"
73.4.1 Pre-configured Caches

The configuration file includes pre-configured caches (small, medium, and large). These caches are accessible through the Cache Function Provider (see Section 73.3.1, "The Cache Function Provider"). The access methods are described in "Cache Methods" in Chapter 69, "Function Provider Reference." They include:

- `getSmallCache(String cacheName)`
- `getMediumCache(String cacheName)`
- `getLargeCache(String cacheName)`

**Note:** The "cacheName" parameter is appended to an internal prefix to match patterns in the `wcps-dataprovider-cache-config.xml`. For example: `wcps-dp-large-<cacheName>`. For more details, see Section 73.2.2.2, "Data Provider Cache File."

73.4.2 Custom Caches

This assumes you have added a new entry in `wcps-dataprovider-cache-config.xml`. It returns a cache with that cacheName (no prefix), configured as the configuration file has specified it.

The Cache Function Provider method used to return a custom cache that is configured in `wcps-dataprovider-cache-config.xml` is:

`getCustomCache(String cacheName)`

For more details, see "Cache Methods" in Chapter 69, "Function Provider Reference."

73.4.3 When a Cache Has Its Own Configuration File

The configuration file, "configFile" must be available to the WCPS web application classloader. See also Section 73.2.2.3, "Creating Custom Cache Configuration Files."

The function provider method used to return a cache configured in a custom cache configuration file is:

`getCache(String myConfigFile.xml, String cacheName)`

**Note:** The cache configuration file must be in the application classpath. Otherwise, it can be anywhere in the classpath of the caller.

For more details, see "Cache Methods" in Chapter 69, "Function Provider Reference."

73.5 Configuring a Cache on a Data Provider Connection

You can hard code a cache name into a data provider. Or, you can set the cache name as a parameter in the `wcps-connections.xml` provider configuration file. This file is explained in detail in Section 67.5.6, "Updating the WCPS Connections Configuration File."
Tip: The advantage of setting the cache name as a connection parameter is that the cache name can be changed at runtime.

Example 73–6 illustrates how to configure a cache into wcps-connnections.xml. You need to add two properties to the <connection> element.

- <cacheName> – Specifies a name for the cache. In the example, the name is demo-cache.
- <configFileUri> – Specifies the location of the cache configuration file on the filesystem. In the example, the value is file:///mydir/demo-cache.xml.

In this example, the <connection> element is given a <cache-name> property with a value of demo-cache.

Example 73–6 Data Provider Connection Configuration

```xml
<connection>
  <connection-name>TestConnection</connection-name>
  <connection-type>my.test.connection</connection-type>
  <namespace>default</namespace>
  <properties>
    <property>
      <name>cacheName</name>
      <value>demo-cache</value>
    </property>
    <property>
      <name>configFileUri</name>
      <value>file:///Users/jpalmer/demo-cache.xml</value>
    </property>
  </properties>
</connection>
```

73.6 Monitoring Oracle Coherence

Coherence provides an MBean console to monitor its caches. For information on how to set server properties to enable this feature, see Using JMX to Manage Oracle Coherence.
Figure 73–2  Coherence Monitoring Console
Part XII

Completing Your WebCenter Portal Application

Part XII contains the following chapters:

- Chapter 74, "Securing Your WebCenter Portal Framework Application"
- Chapter 75, "Building Multilanguage Portals"
Securing Your WebCenter Portal Framework Application

This chapter describes the security features and mechanisms 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 topics:

- Section 74.1, "Introduction to Portal Framework Application Security"
- Section 74.2, "Creating an Application Role"
- Section 74.3, "Configuring ADF Security"
- Section 74.4, "Using the Role Manager Task Flow"
- Section 74.5, "Using the Enterprise Role Member Task Flows"
- Section 74.6, "Using the Page Hierarchy Security Editor"
- Section 74.7, "Creating Login Pages and a Login Component"
- Section 74.8, "Creating a Login Portlet"
- Section 74.9, "Adding Portlets to a Login Page"
- Section 74.10, "Creating a Self-Registration Page"
- Section 74.11, "Creating a Reset Password Page"
- Section 74.12, "Configuring Basic Authentication for Testing Portlet Personalization"
- Section 74.13, "Working with External Applications"
- Section 74.14, "Registering Custom Certificates with the Keystore"
- Section 74.15, "Overriding Inherited Security on Portlets and Customizable Components"
- Section 74.16, "Identity Propagation Mechanisms"
- Section 74.17, "Securing Identity Propagation Through WSRP Producers with WS-Security"
- Section 74.18, "Implementing PDK-Java Portlet Security"
- Section 74.19, "Using WebCenter Portal Impersonation ELs and APIs"
- Section 74.20, "Troubleshooting Security Issues"
74.1 Introduction to Portal Framework Application Security

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, see the "Enabling ADF Security in a Fusion Web Application" section in Fusion Developer’s Guide for Oracle Application Development Framework.

By default, a 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. For more information about configuring ADF security, see Section 74.3, "Configuring ADF Security." Default login and logout pages are also provided with the Portal Framework application template.

Portal 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 74.4, "Using the Role Manager Task Flow," and Section 74.6, "Using the Page Hierarchy Security Editor."

74.2 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, which has administrator privileges for the domain, and whose default member is weblogic

- **AppConnectionManager** - the default member for the AppConnectionManager role is the application Administrator role, which has administration privileges for the application

- **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 taskflow 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 the "Understanding Application Roles" section in Administering Oracle WebCenter Portal.

You can add members at runtime for these roles using the Administrator page (or the Role Manager taskflow 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.

### 74.3 Configuring ADF Security

ADF security is configured by default if you created your application using the Portal Framework application template. This section describes the Configure ADF security wizard, which you can use to override the default settings for a Portal Framework application, or if your application does not use the Portal 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 Portal Framework application, and when components are consumed.

This section contains the following subsections:

- Section 74.3.1, "Configuring ADF Security Settings"
- Section 74.3.2, "Automated Security Grants for Portal Framework Applications"

### 74.3.1 Configuring ADF Security Settings

This section describes the Configure ADF security wizard, which you can use to override the default settings for a Portal Framework application, or if your application does not use the Portal 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 Portal Framework applications.
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 74–2.

**Note:** Portal Framework applications provide default login and error pages. You should only select the **Generate Default Pages** option for applications not built using the Portal 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 74.7, “Creating Login Pages and a Login Component.”
6. Click Next.

7. On the Enable automatic policy grants page, shown in Figure 74–3, 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 74–2 Authentication Page of ADF Security Wizard

Figure 74–3 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 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 74.3.2, "Automated Security Grants for Portal Framework Applications."

**Note:** By default, anonymous-user is granted permission to access 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 74–4.

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 74–4 Specify an Authenticated Welcome Page

10. The Summary page opens, as shown in Figure 74–5.

Figure 74–5 Summary Page

11. Click Finish.

   It may take several moments for the wizard to complete.

12. On the Security Infrastructure Created dialog, click OK.
### 74.3.2 Automated Security Grants for Portal Framework Applications

Table 74–1 lists the security grants that are provided automatically in a Portal Portal Framework application configured with standard portal features.

<table>
<thead>
<tr>
<th>Role</th>
<th>Grant</th>
<th>Actions</th>
<th>Resource Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>anonymous-role</td>
<td>TaskflowPermission</td>
<td>View</td>
<td>/oracle/webcenter/./:*</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/siteresources/scopedMD/*</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/portalapp/*</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/navigationtaskflows/view/pagebreadcrumb-definition.xml#pagebreadcrumb-definition</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/navigationtaskflows/view/pagemeenu-definition.xml#pagemenu-definition</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/navigationtaskflows/view/pagetree-definition.xml#pagetree-definition</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/navigationtaskflows/view/.*</td>
</tr>
<tr>
<td></td>
<td>RegionPermission</td>
<td>View</td>
<td>oracle.webcenter.siteresources.scopedMD.*</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>oracle.webcenter.taskflow.view.ContainerPageDef</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>oracle.webcenter.taskflow.view.ViewerPageDef</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>oracle.webcenter.taskflow.view.PortletPageDef</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>oracle.webcenter.page.pstemplates.*</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>oracle.webcenter.page.pstemplates.*</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>oracle.webcenter.portalapp.pagetemplates.*</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>oracle.webcenter.portalapp.pages.loginPageDef</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>oracle.webcenter.portalapp.pages.errorPageDef</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>oracle.webcenter.portalapp.pages.navigation_rendererPageDef</td>
</tr>
<tr>
<td></td>
<td>HierarchicalResourcePermission</td>
<td></td>
<td>serviceID=oracle.webcenter:page,scopeID=s8bba98ff_4cb40b8_beece_296c916a23ed,resourceID=s8bba98ff_4cb40b8_beece_296c916a23ed</td>
</tr>
<tr>
<td>authenticated-role</td>
<td>RegionPermission</td>
<td>View</td>
<td>oracle.webcenter.portalwebapp.pages.adminPageDef</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>oracle.webcenter.collab.*</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>oracle.webcenter.doclib.*</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>oracle.webcenter.tagging.*</td>
</tr>
<tr>
<td></td>
<td>RegionPermission</td>
<td>Grant</td>
<td>oracle.webcenter.taskflow.view.ContainerPageDef</td>
</tr>
<tr>
<td>HierarchicalResourcePermission</td>
<td></td>
<td>View</td>
<td>serviceID=oracle.webcenter:page,scopeID=s8bba98ff_4cb40b8_beece_296c916a23ed,resourceID=s8bba98ff_4cb40b8_beece_296c916a23ed</td>
</tr>
</tbody>
</table>

**Note:** If you 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:

```bash
revokePermission(appStripe="papp1#V2.0",principalClass="oracle.security.jps.internal.core.principals.JpsAnonymousRoleImpl", principalName="anonymous-role", permClass="oracle.adf.controller.security.TaskFlowPermission", permActions="view", permTarget="/oracle/webcenter/.*")
```
Configuring ADF Security

Securing Your WebCenter Portal Framework Application

74-9

Table 74–2 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 74.3.1, "Configuring ADF Security Settings."

<table>
<thead>
<tr>
<th>Role</th>
<th>Grant</th>
<th>Actions</th>
<th>Resource Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CatalogPermission</td>
<td>Create, Edit</td>
<td>/oracle/webcenter/quicklinks/scopedMD/*.</td>
</tr>
<tr>
<td>Administrator</td>
<td>HierarchicalResource</td>
<td>Create, Delete, Grant,</td>
<td>serviceID=oracle.webcenter.page,scopeID=s8bba98ff_4cbb_40b8_beece_296c916a23ed,resourceID=s8bba98ff_4cbb_40b8_beece_296c916a23ed</td>
</tr>
<tr>
<td></td>
<td>Permission</td>
<td>Manage, Personalize,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Update, View</td>
<td></td>
</tr>
<tr>
<td>WebCenterResource</td>
<td>Manage</td>
<td></td>
<td>oracle_webcenter_siteresource_s8bba98ff_4cbb_40b8_beece_296c916a23ed_contentpresenter_*.</td>
</tr>
<tr>
<td></td>
<td>Permission</td>
<td></td>
<td>oracle_webcenter_siteresource_s8bba98ff_4cbb_40b8_beece_296c916a23ed_datacontrol_*.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>oracle_webcenter_siteresource_s8bba98ff_4cbb_40b8_beece_296c916a23ed_navigation_*.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>oracle_webcenter_siteresource_s8bba98ff_4cbb_40b8_beece_296c916a23ed_pagestyle_*.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>oracle_webcenter_siteresource_s8bba98ff_4cbb_40b8_beece_296c916a23ed_catalog_*.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>oracle_webcenter_siteresource_s8bba98ff_4cbb_40b8_beece_296c916a23ed_pagetemplate_*.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>oracle_webcenter_siteresource_s8bba98ff_4cbb_40b8_beece_296c916a23ed_skin_*.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>oracle_webcenter_siteresource_s8bba98ff_4cbb_40b8_beece_296c916a23ed_taskflow_*.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>oracle_webcenter_siteresource_s8bba98ff_4cbb_40b8_beece_296c916a23ed_taskflowstyle_*.</td>
</tr>
</tbody>
</table>

Table 74–1 (Cont.) Automated Security Grants for Portal Framework applications

Note: When a 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 components that have been consumed in the application. Common grants are granted whenever any WebCenter Portal service is consumed.
### Table 74–2  Automated Security Grants for Portal Framework Applications

<table>
<thead>
<tr>
<th>Service</th>
<th>Role</th>
<th>Grant</th>
<th>Actions</th>
<th>Resource Name</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Common</td>
<td>anonymous-role</td>
<td>TaskFlow</td>
<td>View</td>
<td>/oracle/webcenter/framework/service/controller/taskflows/resourceViewer.xml#resourceViewer</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Permission</td>
<td></td>
<td>/oracle/webcenter/security/peoplepicker/jsf/taskflows/peoplepicker.xml#peoplepicker</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Region</td>
<td></td>
<td>/oracle/webcenter/peopleconnections/personalweb/view/jsf/regions/profile-gallery.xml#profile-gallery</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Permission</td>
<td></td>
<td>/oracle/webcenter/peopleconnections/profile/view/jsf/regions/extended/extended-profile.xml#extended-profile</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Region</td>
<td></td>
<td>/oracle/webcenter/peopleconnections/view/jsf/regions/profile-snapshot.xml#profile-snapshot</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Permission</td>
<td></td>
<td>/oracle/webcenter/peopleconnections/connections/controller/taskflows/connections-main-view-taskflow.xml#connections-main-view-taskflow</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Region</td>
<td></td>
<td>/oracle/webcenter/peopleconnections/connections/controller/taskflows/connections-mini-view-taskflow.xml#connections-mini-view-taskflow</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Permission</td>
<td></td>
<td>/oracle/webcenter/peopleconnections/connections/controller/taskflows/table-of-connections-taskflow.xml#table-of-connections-taskflow</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Region</td>
<td></td>
<td>/oracle/webcenter/peopleconnections/wall/controller/taskflows/WallDetailViewer.xml#WallDetailViewer</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Permission</td>
<td></td>
<td>/oracle/webcenter/peopleconnections/wall/controller/taskflows/WallViewer.xml#WallViewer</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Region</td>
<td></td>
<td>/oracle/webcenter/peopleconnections/kudos/controller/taskflows/KudosDetailViewer.xml#KudosDetailViewer</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Permission</td>
<td></td>
<td>/oracle/webcenter/peopleconnections/kudos/controller/taskflows/KudosMiniViewer.xml#KudosMiniViewer</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Permission</td>
<td></td>
<td>/oracle/webcenter/activitystreaming/controller/taskflows/activity-streaming-mainview.xml#activity-streaming-mainview</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Permission</td>
<td></td>
<td>/oracle/webcenter/activitystreaming/controller/taskflows/activity-streaming-miniview.xml#activity-streaming-miniview</td>
<td></td>
</tr>
<tr>
<td>Administrator</td>
<td>WCApp</td>
<td>Manage</td>
<td></td>
<td>/oracle/webcenter/peopleconnections/profile/s8bba98ff_4cbb_40b8_beee_296c916a23ed/*</td>
<td></td>
</tr>
<tr>
<td>Administrator</td>
<td>Profile</td>
<td>Manage</td>
<td></td>
<td>/oracle/webcenter/peopleconnections/profile/s8bba98ff_4cbb_40b8_beee_296c916a23ed/*</td>
<td></td>
</tr>
</tbody>
</table>
### Table 74–2 (Cont.) Automated Security Grants for Portal Framework Applications

<table>
<thead>
<tr>
<th>Service</th>
<th>Role</th>
<th>Grant</th>
<th>Actions</th>
<th>Resource Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Announcement</td>
<td>anonymous-role</td>
<td>TaskFlow Permission</td>
<td>View</td>
<td>/oracle/webcenter/collab/announcement/view/taskflows/main-view-definition.xml#announcement-main-view</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/announcement/view/taskflows/mini-view-definition.xml#announcement-mini-view</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/announcement/view/taskflows/link-existing-view-definition.xml#link-existing-view-definition</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/announcement/view/taskflows/config-view-definition.xml#announcement-config-view</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/announcement/view/taskflows/announcement-view-definition.xml#announcement-resource-view</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/announcement/view/taskflows/+</td>
</tr>
<tr>
<td>authenticated-role</td>
<td></td>
<td>Region Permission</td>
<td>View</td>
<td>oracle.webcenter.collab.announcement.view.pageDefs.oracle_webcenter_collab_announcement_view_jsf_pages.AnnouncementLinkExistingViewPageDef</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>oracle.webcenter.collab.announcement.view.pageDefs.oracle_webcenter_collab_announcement_view_jsf_pages.AnnouncementLinkResourceViewPageDef</td>
</tr>
<tr>
<td>Composer</td>
<td>anonymous-role</td>
<td>Taskflow Permission</td>
<td>View</td>
<td>/oracle/adfinternal/pageeditor/+</td>
</tr>
<tr>
<td>Discussions</td>
<td>anonymous-role</td>
<td>Taskflow Permission</td>
<td>View</td>
<td>/oracle/webcenter/collab/forum/view/taskflows/main-task-flow.xml#forum-main</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/forum/view/taskflows/miniview-task-flow.xml#forum-miniview</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/forum/view/taskflows/popularTopic-task-flow.xml#forum-popularTopic</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/forum/view/taskflows/recentTopic-task-flow.xml#forum-recentTopic</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/forum/view/taskflows/watchedTopic-task-flow.xml#forum-watchedTopic</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/forum/view/taskflows/watchedForum-task-flow.xml#forum-watchedForum</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/forum/view/taskflows/config-task-flow.xml#forum-config-view</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/forum/view/taskflows/link-existing-task-flow.xml#forum-link-existing</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/forum/view/taskflows/link-new-task-flow.xml#forum-link-new</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/forum/view/taskflows/message-task-flow.xml#forum-message</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/forum/view/taskflows/resource-view-task-flow.xml#forum-resource-view</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/forum/view/taskflows/scope-config-task-flow.xml#forum-scope-config-view</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/forum/view/taskflows/+</td>
</tr>
</tbody>
</table>
### Table 74–2 (Cont.) Automated Security Grants for Portal Framework Applications

<table>
<thead>
<tr>
<th>Service</th>
<th>Role</th>
<th>Grant</th>
<th>Actions</th>
<th>Resource Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Documents</td>
<td>anonymous-role</td>
<td>Taskflow</td>
<td>View</td>
<td>/oracle/webcenter/doclib/view/jsf/taskflows/contentPresenter.xml#doclib-content-presenter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Permission</td>
<td></td>
<td>/oracle/webcenter/doclib/view/jsf/taskflows/docListViewer.xml#doclib-document-list-viewer</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/doclib/view/jsf/taskflows/docViewer/docInfo.xml#doclib-document-information</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/doclib/view/jsf/taskflows/explore/explorer.xml#doclib-explorer</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/doclib/view/jsf/taskflows/treeNavigator.xml#doclib-navigator</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/doclib/view/jsf/taskflows/documentViewer.xml#doclib-document-viewer</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/doclib/view/jsf/taskflows/folderViewer/folderView.xml#doclib-folder-viewer</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/doclib/view/jsf/taskflows/mainView.xml#doclib-document-library</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/doclib/view/jsf/taskflows/miniProperties/miniProperties.xml#doclib-mini-properties</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/doclib/view/jsf/taskflows/recentDocuments.xml#doclib-recent-documents</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/doclib/view/jsf/taskflows/richtextViewEditor/editor.xml#doclib-richtext-editor</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/doclib/view/jsf/taskflows/uploadable.xml#doclib-upload-document</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/doclib/view/jsf/taskflows/versionHistory/history.xml#doclib-version-history</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/blog/view/jsf/taskflows/blogDigestViewer/blog-main-view.xml#blog-main-view</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/doclib/view/jsf/taskflows/*</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/blog/view/jsf/taskflows/*</td>
</tr>
<tr>
<td>External Application</td>
<td>authenticated-role</td>
<td>Taskflow</td>
<td>View</td>
<td>/oracle/adfinternal/extapp/view/fragments/extapp-credential-provisioning-taskflow.xml#extapp-credential-provisioning-taskflow</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Permission</td>
<td></td>
<td>/oracle/adfinternal/extapp/view/fragments/extapp-change-password-taskflow.xml#extapp-change-password-taskflow</td>
</tr>
</tbody>
</table>
### Table 74–2 (Cont.) Automated Security Grants for Portal Framework Applications

<table>
<thead>
<tr>
<th>Service</th>
<th>Role</th>
<th>Grant</th>
<th>Actions</th>
<th>Resource Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Links</td>
<td>anonymous-role</td>
<td>Taskflow</td>
<td>View</td>
<td>/oracle/webcenter/relationship/view/jsf/resources/links-detail.xml#links-detail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Permission</td>
<td></td>
<td>/oracle/webcenter/relationship/view/jsf/resources/links-detail-popup.xml#links-detail-popup</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/relationship/view/jsf/resources/links-to-service.xml#link-to-service</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/relationship/view/jsf/resources/links-picker-popup.xml#links-picker-popup</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/relationship/url/view/jsf/taskflows/linkToUrl.xml#link-to-url</td>
</tr>
<tr>
<td>Lists</td>
<td>anonymous-role</td>
<td>Taskflow</td>
<td>View</td>
<td><em>s8bba98ff_4cbb_40b8_beee_296c916a23ed/.</em></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Permission</td>
<td></td>
<td>/oracle/webcenter/list/view/jsf/regions/main-view-task-flow.xml#main-view-task-flow</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/list/view/jsf/regions/list-instance-view-task-flow.xml#list-instance-view-task-flow</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/list/view/jsf/regions/*/</td>
</tr>
<tr>
<td>Mail</td>
<td>anonymous-role</td>
<td>Taskflow</td>
<td>View</td>
<td><em>s8bba98ff_4cbb_40b8_beee_296c916a23ed/.</em></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Permission</td>
<td></td>
<td>/oracle/webcenter/collab/mail/view/jsf/regions/compose-task-flow.xml#mail-compose-view</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/mail/view/jsf/regions/content-view-definition.xml#mail-content-view</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/mail/view/jsf/regions/dl-config-definition.xml#dl-config-view</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/mail/view/jsf/regions/mini-view-definition.xml#mail-mini-view</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/mail/view/jsf/regions/*/</td>
</tr>
<tr>
<td>Notifications</td>
<td>anonymous-role</td>
<td>Taskflow</td>
<td>View</td>
<td>oracle.webcenter.collab.mail.view.pageDefs.oracle_webcenter_collab_mail_view_jsf_pages_ComposeviewPageDef</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Permission</td>
<td></td>
<td>oracle.webcenter.collab.mail.view.pageDefs.oracle_webcenter_collab_mail_view_jsf_pages_ContentViewPageDef</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/notification/view/jsf/regions/SubscriptionPreferences.xml#SubscriptionPreferences</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/notification/view/jsf/regions/SubscriptionViewer.xml#SubscriptionViewer</td>
</tr>
</tbody>
</table>

Securing Your WebCenter Portal Framework Application  74-13
### Table 74-2 (Cont.) Automated Security Grants for Portal Framework Applications

<table>
<thead>
<tr>
<th>Service</th>
<th>Role</th>
<th>Grant</th>
<th>Actions</th>
<th>Resource Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Page</td>
<td>anonymous-role</td>
<td>TaskFlow Permission</td>
<td>View</td>
<td>/oracle/webcenter/page/view/jsf/fragments/page-create-page</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/page/view/jsf/fragments/page-doc-prop-panel-definition.xml#page-doc-prop-panel-definition</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/page/view/jsf/fragments/page-doc-sec-panel-definition.xml#page-doc-sec-panel-definition</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/page/view/jsf/fragments/golink-prop-panel-definition.xml#golink-prop-panel-definition</td>
</tr>
<tr>
<td></td>
<td>authenticated-role</td>
<td>Region Permission</td>
<td>Create</td>
<td>oracle_webcenter_pageScopedMD_s8bba98ff_4cbb_40b8_beee_296c916a23ed_.*</td>
</tr>
<tr>
<td></td>
<td>Administrator</td>
<td>Region Permission</td>
<td>Customize, Edit, Grant, Personalize, View</td>
<td>oracle_webcenter_pageScopedMD_s8bba98ff_4cbb_40b8_beee_296c916a23ed_.*</td>
</tr>
<tr>
<td>Polls</td>
<td>anonymous-role</td>
<td>TaskFlow Permission</td>
<td>View</td>
<td>/oracle/webcenter/collab/survey/view/jsf/taskflows/list-surveys-definition.xml#list-surveys</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/survey/view/jsf/taskflows/take-survey-definition.xml#take-survey</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/survey/view/jsf/taskflows/quick-poll-definition.xml#quick-poll</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/survey/view/jsf/taskflows/view-results-definition.xml#view-results</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/survey/view/jsf/taskflows/take-polls-definition.xml#take-polls</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/survey/view/jsf/taskflows/empty-definition.xml#empty-definition</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/survey/view/jsf/taskflows/*</td>
</tr>
<tr>
<td>Recent Activities</td>
<td>anonymous-role</td>
<td>TaskFlow Permission</td>
<td>View</td>
<td>/oracle/webcenter/recentactivity/controller/taskflows/recent-activities.xml#recent-activities</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/recentactivity/controller/taskflows/*</td>
</tr>
<tr>
<td>RSS</td>
<td>anonymous-role</td>
<td>TaskFlow Permission</td>
<td>View</td>
<td>/oracle/webcenter/rssviewer/view/jsf/fragments/RSSViewerTaskFlow.xml#RSSViewerTaskFlow</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/rssviewer/view/jsf/fragments/*</td>
</tr>
</tbody>
</table>
### Table 74–2 (Cont.) Automated Security Grants for Portal Framework Applications

<table>
<thead>
<tr>
<th>Service</th>
<th>Role</th>
<th>Grant</th>
<th>Actions</th>
<th>Resource Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search</td>
<td>anonymous-role</td>
<td>TaskFlow</td>
<td>View</td>
<td>/oracle/webcenter/search/controller/taskflows/searchResults.xml#search-view</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Permission</td>
<td></td>
<td>/oracle/webcenter/search/controller/taskflows/localToolbarSearch.xml#search-toolbar</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/search/controller/taskflows/preferences.xml#search-preferences</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/search/controller/taskflows/allSavedSearches.xml#all-saved-searches</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/search/controller/taskflows/simpleSearchResults.xml#search-simple-view</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/search/controller/taskflows/customize.xml#search-customize</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/search/controller/taskflows/*.xml#search-view</td>
</tr>
<tr>
<td>Tags</td>
<td>anonymous-role</td>
<td>TaskFlow</td>
<td>View</td>
<td>/oracle/webcenter/tagging/controller/taskflows/related-links.xml#tagging-related-links</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Permission</td>
<td></td>
<td>/oracle/webcenter/tagging/controller/taskflows/launch-dialog.xml#tagging-launch-dialog</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/tagging/controller/taskflows/tagging-personal-view.xml#tagging-personal-view</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/tagging/controller/taskflows/tag-selection.xml#tag-selection</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/tagging/controller/taskflows/related-resources.xml#related-resources</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/tagging/controller/taskflows/tag-center-task-flow.xml#tag-center</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/tagging/controller/taskflows/tag-center-related-tags.xml#tag-center-related-tags</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/tagging/controller/taskflows/tag-center-related-users.xml#tag-center-related-users</td>
</tr>
<tr>
<td>Worklist</td>
<td>authenticated-role</td>
<td>Region</td>
<td>View</td>
<td>oracle.webcenter.tagging.view.pageDefs.oracle_webcenter_tagging_view.jsf_fragments_launch_dialogPageDef</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Permission</td>
<td></td>
<td>oracle.webcenter.tagging.view.pageDefs.oracle_webcenter_tagging_view.jsf_fragments_tag_centerPageDef</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>oracle.webcenter.tagging.view.pageDefs.oracle_webcenter_tagging_view.jsf_fragments_tag_center_related_resourcesPageDef</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>oracle.webcenter.tagging.view.pageDefs.oracle_webcenter_tagging_view.jsf_fragments_tag_center_related_tagsPageDef</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>oracle.webcenter.tagging.view.pageDefs.oracle_webcenter_tagging_view.jsf_fragments_tag_center_related_usersPageDef</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>oracle.webcenter.tagging.view.pageDefs.oracle_webcenter_tagging_view.jsf_fragments_tag_center_tag_selectionPageDef</td>
</tr>
</tbody>
</table>

| Worklist         | anonymous-role        | TaskFlow      | View                                                                    | /oracle/webcenter/worklist/view/jsf/taskFlowDefs/worklist.xml#worklist         |
|                  |                       | Permission    |                                                                         | /oracle/webcenter/worklist/view/jsf/taskFlowDefs/*.xml#worklist               |
74.4 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 Portal Framework application template. For information on how to use the Role Manager runtime to manage and configure users and application roles, see the "Managing Users and Application Roles" section in Administering 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 Portal Framework application:

1. Create a new application choosing Portal 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.

To add the Role Manager task flow onto the page you created as a region.
Using the Page Hierarchy Security Editor

6. Build the application, log in as an administrator, and navigate to the page containing the role Manager task flow.

74.5 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 74.4, "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.

74.6 Using the Page Hierarchy Security Editor

This section describes how to use page hierarchy security to provide inherited and delegated security for Portal Framework application pages. It includes the following sections:

- Section 74.6.1, "Introduction to Page Hierarchy Security"
- Section 74.6.2, "Building a Page Hierarchy"

74.6.1 Introduction to Page Hierarchy Security

This section describes the page hierarchy model and how page hierarchy security works within a Portal Framework application.

This section includes the following subsections:
Section 74.6.1.1, "Page Hierarchy Model"

Section 74.6.1.2, "Page-level and Page Hierarchy Security"

Section 74.6.1.3, "Inherited and Delegated Security"

74.6.1.1 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 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 74–8 shows the default navigation model (default-navigation-model.xml), which includes a node for the default page hierarchy (pages.xml).

Figure 74–8 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 10.2, "Creating a Navigation Model." For more information about adding page hierarchies to a navigation, see Section 10.3.5, "How to Add Other Resources 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.
Using the Page Hierarchy Security Editor

Figure 74–9  Page Hierarchy Root Node Permissions

![Page Hierarchy Root Node Permissions](image)

Figure 74–9 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.

74.6.1.2 Page-level and Page Hierarchy Security
A page hierarchy provides an alternative security mechanism to page-level security for your Portal 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.

74.6.1.3 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

---

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.
Section 74.6.2, "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.

---

### 74.6.2 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 74.6.2.1, "Adding a Page Hierarchy to a Navigation"
- Section 74.6.2.2, "Adding Pages to a Page Hierarchy"
- Section 74.6.2.3, "Delegating Security"

#### 74.6.2.1 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 10.3.5, "How to Add Other Resources to a Navigation Model."

#### 74.6.2.2 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.
74.6.2.3 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 the "Managing Application Roles and Permissions" section in *Administering 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 74–10).

**Figure 74–10  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 the "Managing Application Roles and Permissions" section in *Administering Oracle WebCenter Portal*.
74.7 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:

   ```xml
   <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 Portal Framework application:

   ```xml
   <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-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 the "ADF Faces Configuration" section in Fusion Developer’s Guide for Oracle Application Development Framework.

74.8 Creating a Login Portlet

Refer to Fusion Developer’s Guide for Oracle Application Development Framework for information about creating a login portlet for your application.

74.9 Adding Portlets to a Login Page

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 Framework application.

Make sure your portlet producers have been registered before proceeding. See Chapter 63, "Consuming Portlets" 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 74–11.

**Figure 74–11  Create Web Application Filter Mapping Dialog Box**

4. Click OK.

Follow the steps in *Fusion Developer’s Guide for Oracle Application Development Framework* to complete the creation of the login page.

### 74.10 Creating a Self-Registration Page

You can create a self-registration page using the WebCenter Portal 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 WebCenter Portal public invitation task flow. For more information about these two types of self-registration pages and their runtime behavior, see the "Enabling Self-Registration" section in *Administering Oracle WebCenter Portal*.

This section contains the following subsections:

- Section 74.10.1, "Integrating the WeBCenter Portal Self-Registration Task Flow"
- Section 74.10.2, "Building a Custom Self-Registration Page"

#### 74.10.1 Integrating the WeBCenter Portal Self-Registration Task Flow

You can integrate the self-registration page provided in WebCenter Portal in a Framework application as described below. For information about the runtime behavior of the task flow, see the "Enabling Anyone to Self-Register" section in *Administering Oracle WebCenter Portal*.

To consume the WebCenter Portal self-registration task flow:

1. Open or create a Portal 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. For information on how to set up a mail connection, see the "Managing Mail" chapter in *Administering Oracle WebCenter Portal*.

5. Run the page with the self-registration task flow to see the runtime self-registration page.

---

**Caution:** If you are using a login attribute other than *cn*, such as *uid* or *mail*, then your *jps-config.xml* file must contain the properties *user.login.attr* and *username.attr*, and these must be set to the corresponding login attribute (mail, for example). Modify *jps-config.xml* file as shown in the example below:

```xml
<property name="user.login.attr" value="mail"/>
<property name="username.attr" value="mail"/>
```

After making the change, restart the WebCenter Portal server (*WC_Spaces*).

---

### 74.10.2 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.

### 74.10.3 Creating a Self-Registration Invitation Page

You can create a public invitation to self-register page in your Portal Framework application using the WebCenter Portal public invitation task flow. For information about the runtime behavior of the task flow, see the "Enabling Self-Registration By Invitation-Only" section in *Administering Oracle WebCenter Portal*.

To consume the self-registration invitation task flow:

1. Open or create a Portal 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 the following:

   ```xml
   <permission>
   <class>oracle.security.jps.service.credstore.CredentialAccessPermission</class>
   <name>context=SYSTEM,mapName=o.webcenter.security.selfreg,keyName=o.webcenter.security.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.

### 74.11 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 Portal 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.

### 74.12 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 (Omnportlet, JPS, and PDK-Java).

---

1. Create a user sking and the role manager. See *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 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.

74.13 Working with External Applications

The 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 Portal 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 Portal 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.
The external applications that are to be used by the 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 Administrator’s Guide for more information.

This section contains the following:

- Section 74.13.1, "Using External Applications"
- Section 74.13.2, "Supplying User Credentials"
- Section 74.13.3, "Managing External Applications"

### 74.13.1 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 74.13.1.1, "Secured Service Connections"
- Section 74.13.1.2, "Automated Single Sign-On"

### 74.13.1.1 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 74–12.
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 74–13.

For more information about the identity propagation mechanisms used by WebCenter Portal, see Section 74.16, “Identity Propagation Mechanisms”.

Note: The following WebCenter Portal components must be configured with external applications for credential mapping support to be available:

- Instant Messaging and Presence (IMP)
- Mail
- Documents
- RSS
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 the WebCenter Portal 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.

### 74.13.1.2 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 ($\text{adfextapplogin}$). 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.

/\text{adfextapplogin}\?\text{extappid}=<\text{extappid}>

For example, if you had a Framework application in which you defined an external application that represented the $\text{myoracle.com}$ web site (external application identifier is "myoracle"), the proxy URL would look like this:

/\text{adfextapplogin}\?\text{extappid=myoracle}

The link's target attribute should also be set appropriately. For example, if you use $<\text{a href}=\rangle$ 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.

/\text{adfextapplogin}\?\text{extappid=<extappid>} \ [\text{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.

---

### 74.13.2 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 components, the credential provisioning screen is incorporated into the task flow that the component 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 components 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 74.13.3, "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 34.2.2, "Adding IMP Functionality 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.

74.13.3 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 74.13.3.1, "Adding External Application Task Flows"
- Section 74.13.3.2, "Working with External Applications in Oracle JDeveloper"
- Section 74.13.3.3, "Working with External Applications in Enterprise Manager"
- Section 74.13.3.4, "Working with External Applications Using WLST"

74.13.3.1 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 Tools and 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 Portal Framework application template, both these task flows are available out-of-the-box through Portal Framework application Administration page (Tools and Services tab). For details, see the "Configuring Services, Portlet Producers, and External Applications for Portal Framework Applications" section in Administering 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 Portal 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 15.2, "Creating Pages in a WebCenter Portal Framework Application").

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 - # {bindings.govapp.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 "Creating Test Users" section in **Fusion Developer’s Guide for Oracle Application Development Framework**.

6. Save and run your page. Log in as an administrator or the `TEST_EXTAPP` user defined in the previous step. The screen shown in **Figure 74–14** appears.

**Figure 74–14  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 "Adding Members to Application Roles" section in **Administering Oracle WebCenter Portal**). Alternatively, system administrators can grant AppConnectionManager and AppConnectionViewer roles through Fusion Middleware Control (see the "Granting Application Roles Using Fusion Middleware Control" section in **Administering Oracle WebCenter Portal**).
74.13.3.2 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 74.13.3.2.1, "Registering an External Application in Oracle JDeveloper"
- Section 74.13.3.2.2, "Editing External Application Registration Details in Oracle JDeveloper"
- Section 74.13.3.2.3, "Deleting External Application Registration Details in Oracle JDeveloper"

74.13.3.2.1 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 Portal Framework applications. Select Application Resources to make the external application available only in the Portal Framework application in which it is registered, or select Resource Palette to make the external application available from the Resource Palette to any new Portal 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 74.13.3.2.2, "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.
9. In the **User Name/ID FieldName** field, enter the label that the application uses for the user name field, for example, *Username*.

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 components.
Working with External Applications

(such as Document Library, or Instant Messaging and Presence), and the taskflow 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 taskflow is accessible only to authenticated users

74.13.3.2.2 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 74.13.3.2.1, "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.

74.13.3.2.3 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 components, 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.

### 74.13.3.3 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 "Working with External Applications" chapter in the Using Oracle WebCenter Portal.

### 74.13.3.4 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 "Working with External Applications" chapter in the Using Oracle WebCenter Portal.

### 74.14 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 74–15, click **View Certificate**.
5. In the Certificate dialog box, click the Certification Path tab.

6. The dummy child certificate is selected by default as shown in Figure 74–16. Select the root certificate and click View Certificate.

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 74–17.
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.

### 74.15 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 74–1 and Example 74–2 and set the `enableSecurity` element in those sections to `true`.

**Example 74–1 enableSecurity Element in the Portlet Security Section in adf-config.xml**

```
<!--
```
PORTLETS ACTIONS SECURITY

--> 
  <adfp:enableSecurity value="true"/>
  <adfp:actionsCategory>
    ..........................................
  </adfp:actionsCategory>
</adfp:adf-config-child>

Example 74–2 enableSecurity Element in the Customizable Components Security Section in adf-config.xml

<!--
CUSTOMIZABLE COMPONENTS ACTIONS SECURITY

--> 
  <cust:enableSecurity value="true"/>
  <cust:actionsCategory>
    ..........................................
  </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 74.15.1, "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 22.5, "Applying Action-Level Restrictions on Panel Customizable and Show Detail Component Actions."**

### 74.15.1 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**

### 74.15.1.1 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 74–3 describes the different actionsCategory attributes and the portlet actions they support by default.

**Table 74–3 actionsCategory Attributes and Portlets Actions Mapping**

<table>
<thead>
<tr>
<th>Attribute Value</th>
<th>Actions Supported</th>
</tr>
</thead>
<tbody>
<tr>
<td>viewActionsCategory</td>
<td>Render, isHelpModeAvailable, isNormalModeAvailable, isAboutModeAvailable, isPreviewModeAvailable, isDetailModeAvailable, isLinkModeAvailable, isPrintModeAvailable</td>
</tr>
<tr>
<td>customizeActionsCategory</td>
<td>showMoveAction, showRemoveAction, isCustomizeModeAvailable, showMinimizeAction, showMaximizeAction, isConfigModeAvailable</td>
</tr>
<tr>
<td>personalizeActionsCategory</td>
<td>isPersonalizeModeAvailable</td>
</tr>
</tbody>
</table>

Example 74–3 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 74–3 actionsCategory Element in the Portlets Security Section**

```
<!--
==============================================================================
PORTLETS ACTIONS SECURITY
==============================================================================
-->
  <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>
</adf:adf-config-child>
```
74.15.1.2 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 74–4 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 74–4 actions Element in the Portlets Security Section

```xml
<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>
```

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 74–5.

Example 74–5 InsideBizHours Method Defined in appBusinessRules Managed Bean

```java
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 74–6.
Example 74–6  InsideBizHours Method Referenced in the adf-config.xml File

```
  <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 74–7 to the managed bean:

Example 74–7  InsideCorpNetwork Method Defined in appBusinessRules Managed Bean

```java
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 74–8.

Example 74–8  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.

### 74.16 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 Portal 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 Security and Administrator's Guide for Web Services. For more information about External Applications, see Section 74.13, "Working with External Applications".

<table>
<thead>
<tr>
<th>Service</th>
<th>Connection Type</th>
<th>Identity Propagation Mechanism</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discussions</td>
<td>Jive</td>
<td>Oracle WS-Security</td>
</tr>
<tr>
<td>Announcements</td>
<td>see Discussion Forum</td>
<td></td>
</tr>
<tr>
<td>Documents</td>
<td>Content Server</td>
<td>Proprietary ID propagation mechanism through socket connection. Can use SSL with mutual authentication, or clear socket with IP authorization (or use External Application option)</td>
</tr>
<tr>
<td></td>
<td>File System</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>Portal 10g/11g</td>
<td>JSR-170 (External Application)</td>
</tr>
<tr>
<td></td>
<td>Day Adapters</td>
<td>JSR-170 (External Application)</td>
</tr>
<tr>
<td>EMail</td>
<td>EMail Connection</td>
<td>External Application</td>
</tr>
<tr>
<td></td>
<td>LDAP Connection</td>
<td>External Application</td>
</tr>
<tr>
<td>Events - Personal</td>
<td>Exchange Server</td>
<td>External Application</td>
</tr>
<tr>
<td></td>
<td>Connection</td>
<td></td>
</tr>
<tr>
<td>External Application</td>
<td>HTTP Request from Browser</td>
<td>External Application</td>
</tr>
<tr>
<td>IMP</td>
<td>Microsoft Live</td>
<td>SOAP, Web Services calls.</td>
</tr>
<tr>
<td></td>
<td>Communication Server</td>
<td>External Application</td>
</tr>
<tr>
<td></td>
<td>(LCS)</td>
<td></td>
</tr>
<tr>
<td>Portlet Producers</td>
<td>WSRP Producer (Secure)</td>
<td>Oracle WS-Security (Recommended by WSRP Specification)</td>
</tr>
<tr>
<td></td>
<td>WSRP Producer (Non-Secure)</td>
<td>WSRP userContext in SOAP message (WSRP Specification)</td>
</tr>
</tbody>
</table>
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:
- Section 74.17.1, "Identity Propagation Without WS-Security"
- Section 74.17.2, "Identity Propagation with WS-Security"
- Section 74.17.3, "Configuring Security for WSRP Portlets"

### 74.17 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:
- Section 74.17.1, "Identity Propagation Without WS-Security"
- Section 74.17.2, "Identity Propagation with WS-Security"
- Section 74.17.3, "Configuring Security for WSRP Portlets"

### 74.17.1 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 74–9 shows how a sample portlet's markup rendering code uses the isUserInRole API to decide what content to display.

**Example 74–9 isUserInRole (role) API**

```java
private void doViewHtml(RenderRequest request, RenderResponse response)
```

<table>
<thead>
<tr>
<th>Service</th>
<th>Connection Type</th>
<th>Identity Propagation Mechanism</th>
</tr>
</thead>
<tbody>
<tr>
<td>JPDK Producer</td>
<td>External Application</td>
<td></td>
</tr>
<tr>
<td>(External App)</td>
<td>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)</td>
<td></td>
</tr>
<tr>
<td>JPDK Producer</td>
<td>Username in render request (Proprietary, Legacy)</td>
<td></td>
</tr>
<tr>
<td>(Non-Secure)</td>
<td>Search</td>
<td>Secure Enterprise Search (SES)</td>
</tr>
<tr>
<td>Wiki</td>
<td>Browser Connection</td>
<td>SSO mechanisms - OSSO/OAM or other WLS supported SSO mechanism</td>
</tr>
<tr>
<td>Worklist Service</td>
<td>BPEL Connection</td>
<td>Web Service call</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service</th>
<th>Connection Type</th>
<th>Identity Propagation Mechanism</th>
</tr>
</thead>
<tbody>
<tr>
<td>JPDK Producer (Non-Secure)</td>
<td>Security</td>
<td>Secure Enterprise Search (SES)</td>
</tr>
<tr>
<td>Search</td>
<td>Secure Enterprise Search (SES)</td>
<td>Web Service Call</td>
</tr>
<tr>
<td>Wiki</td>
<td>Browser Connection</td>
<td>SSO mechanisms - OSSO/OAM or other WLS supported SSO mechanism</td>
</tr>
<tr>
<td>Worklist Service</td>
<td>BPEL Connection</td>
<td>Web Service call</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service</th>
<th>Connection Type</th>
<th>Identity Propagation Mechanism</th>
</tr>
</thead>
<tbody>
<tr>
<td>JPDK Producer (Non-Secure)</td>
<td>Security</td>
<td>Secure Enterprise Search (SES)</td>
</tr>
<tr>
<td>Search</td>
<td>Secure Enterprise Search (SES)</td>
<td>Web Service Call</td>
</tr>
<tr>
<td>Wiki</td>
<td>Browser Connection</td>
<td>SSO mechanisms - OSSO/OAM or other WLS supported SSO mechanism</td>
</tr>
<tr>
<td>Worklist Service</td>
<td>BPEL Connection</td>
<td>Web Service call</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service</th>
<th>Connection Type</th>
<th>Identity Propagation Mechanism</th>
</tr>
</thead>
<tbody>
<tr>
<td>JPDK Producer (Non-Secure)</td>
<td>Security</td>
<td>Secure Enterprise Search (SES)</td>
</tr>
<tr>
<td>Search</td>
<td>Secure Enterprise Search (SES)</td>
<td>Web Service Call</td>
</tr>
<tr>
<td>Wiki</td>
<td>Browser Connection</td>
<td>SSO mechanisms - OSSO/OAM or other WLS supported SSO mechanism</td>
</tr>
<tr>
<td>Worklist Service</td>
<td>BPEL Connection</td>
<td>Web Service call</td>
</tr>
</tbody>
</table>
throws PortletException, IOException
{
  // To do: markup the required content.
  PrintWriter out = response.getWriter();
  out.print('<p>Welcome</p>);
  out.println('<p></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>');
}

74.17.2 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 the "Registering Oracle PDK-Java Producers" section in the *Administering Oracle WebCenter Portal*.

The use of WS-Security adds some complexity to the configuration and management of the 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 Framework application.

The 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 (the 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 74–18 summarizes the overall WSRP portlet security architecture.
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 61, "Deploying Portlet Producers."

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 the "Configuring WS-Security" chapter in Administering Oracle WebCenter Portal.

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
74.18.1 Assumptions
The following assumptions are made to perform the tasks in this section:

1. You have followed through and understood Section 60.2, "Creating a PDK-Java Portlet."
2. You built a portlet using the wizard and successfully added it to a page.

74.18.2 Introduction to PDK-Java Portlet Security Features
This section introduces the major features that are available to secure your PDK-Java portlet producers.

74.18.2.1 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:

```java
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.

74.18.2.2 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 74.18.4, "Portlet Security Managers."

74.18.2.3 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 (Portal Framework application) and the producer to digitally sign messages. See Section 74.18.5, "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 74.18.6, "User Input Escape" for more information.

### 74.18.3 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 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.

#### 74.18.3.1 External Application

An external application uses a different authentication server than the Framework application. This means that when a user is logged into the 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 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).

### 74.18.3.2 No Application Authentication

The producer trusts the 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.

### 74.18.4 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 Portal 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`:

```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:

74.18.4.1 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 your new class name and configure child elements appropriately.

74.18.5 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 be 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 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.

74.18.6 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:
74.18.6.1 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 60.3.5.2, "Setting JNDI Variable Values" for more information.

74.18.6.2 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());
```

74.19 Using WebCenter Portal Impersonation ELs and APIs

WebCenter Portal Impersonation offers a set of Expression Language expressions (ELs) and matching Java APIs that can be used to customize impersonation sessions. For more information about managing WebCenter Portal Impersonation, see the "Managing Impersonation" chapter in Administering Oracle WebCenter Portal. For instructions on how to initiate an impersonation session (by the impersonator) and how to allow an Impersonation session (by the impersonatee), see the "Using WebCenter Portal Impersonation" chapter in Using Oracle WebCenter Portal.

The following ELs are exposed:

- #{WCSecurityContext.impersonationConfigured} - returns whether or not impersonation has been enabled for the current domain.

  This EL can be useful when determining if an error was caused by an impersonation session ending prematurely, or to provide an additional indicator that a session has ended.

- #{WCSecurityContext.userInImpersonationSession} - returns whether the current user is in an impersonation session or not.
You can use this EL to protect content and render it inaccessible during an impersonation session. For example, you could map the rendered attribute of an administration taskflow on a page to this EL only rendering the taskflow if the user is not viewing the taskflow in an impersonation session.

■ #{WCSecurityContext.currentImpersonator} - returns the current impersonator, if any.

This EL could be used to modify the page template to display the impersonator or render content accessible only to a particular impersonator.

For more information about impersonation and other ELs, refer to Appendix G, "Expression Language Expressions."

The following public APIs are exposed in oracle.webcenter.security.common.WCSecurityUtility:

■ isImpersonationConfigured() - returns whether or not impersonation has been enabled for the current domain.

This API can be useful to determine if an error was caused by an impersonation session ending prematurely, or to provide an additional indicator that a session has ended.

■ isUserInImpersonationSession() - returns whether the current user is in an impersonation session or not.

This API is recommended for use to protect content and render it inaccessible during an impersonation session. For example, you could map the rendered attribute of an administration taskflow on a page to this API throwing an authorization exception or returning an empty list if the user is viewing the taskflow in an impersonation session.

■ getCurrentImpersonatorId() - returns the current impersonator, if any.

This API could be used to modify the page template to display the impersonator (as shown in Example 74–10), or render some content accessible only to a particular impersonator.

For more information about these and other APIs, refer to Java API Reference for Oracle WebCenter Portal.

Example 74–10  getCurrentImpersonatorId API

```java
import oracle.webcenter.security.common.WCSecurityUtility;
if (WCSecurityUtility.isUserInImpersonationSession())
{
    String impersonator = WCSecurityUtility.getCurrentImpersonatorId();
    String currentUser = ADFContext.getCurrent().getSecurityContext().getUserName();
    //Code to be executed when the user is in an impersonation session.
    ..log("User " + impersonator + " is impersonating as user " + currentUser);
}
```

74.20 Troubleshooting Security Issues

This section provides troubleshooting information to help diagnose security related issues for Portal Framework applications.

This section includes the following troubleshooting notes:

■ Section 74.20.1, "Error Message Appears When Running a Page with a Content Repository Data Control Method Being Consumed"
74.20.1 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:

```java
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);
}
This chapter describes how to configure your application to display text in the correct language of a user’s browser.

This chapter includes the following topics:

- Section 75.1, "Guidelines for Building Multilanguage Portals"
- Section 75.2, "Adding Support for a New Language"
- Section 75.3, "Language Support in ADF Faces Components"
- Section 75.4, "Using Resource Bundles to Support Multiple Languages"

### 75.1 Guidelines for 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 tools and services. For more information, see Section 75.3, "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 75.4, "Using Resource Bundles to Support Multiple Languages."

- Make sure your database character set supports all required languages.

  For more information, see the "Choosing a Character Set" chapter in Oracle Database Globalization Support Guide.

- Use number, date, and time formatting that supports all required locales.

  For more information, see the "Setting Up a Globalization Support Environment" chapter in Oracle Database Globalization Support Guide.
75.2 Adding Support for a New Language

Oracle WebCenter Portal provides run-time translations for 28 languages and 100 different locales. If you want to support a language that is not supported out-of-the-box, you need to provide string files for the new language, and add a <language> tag for the new language in the supported-languages.xml configuration file.

For information about adding support for a new language, see the "Using WebCenter Spaces Extension Samples" white paper on the Oracle WebCenter Portal White Papers and Technical Notes page on Oracle Technology Network.

For a WebCenter Portal Framework application, there may be a need to verify or add the entry for the new language, if not present, in the web.xml file of the application.

For example, to add portuguese as a language (Portugal & Brazil), ensure the following entry is present in web.xml:

```xml
    ...
    <locale-config>
      ...
      <supported-locale>pt</supported-locale>
      <supported-locale>pt-BR</supported-locale>
      <supported-locale>pt-PT</supported-locale>
      ...
    </locale-config>
    ...
```

75.3 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.

75.4 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 about automatic resource bundles, see the "Using Automatic Resource Bundle Integration in JDeveloper" section in Web User Interface Developer’s Guide for Oracle Application Development Framework.

For more information about manually defining resource bundles, see the "Manually Defining Resource Bundles and Locales" section in Web User Interface Developer’s Guide for Oracle Application Development Framework.
Part XIII contains the following appendixes:

- Appendix A, "WebCenter Portal Files"
- 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 F, "Reuse of Oracle Portal Components"
- Appendix G, "Expression Language Expressions"
- Appendix H, "WebCenter Portal Analytics Database Schema"
- Appendix I, "WebCenter Portal Accessibility Features"
A

WebCenter Portal Files

This appendix describes the files that are created and modified as you build up your WebCenter Portal Framework application.

This appendix includes the following topics:

- 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 WebCenter Portal Tools and 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 Fusion Developer’s Guide for Oracle Application Development Framework.

A.1 About Files

When you use Oracle WebCenter Portal 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 Oracle WebCenter Portal 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 Portal Framework applications. Those modifications are described in this appendix, but for more complete descriptions of these common files, see the Fusion Developer’s Guide for Oracle Application Development Framework.

- Files that are unique to Portal 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 Portal Framework application. Hence, the easiest way to discuss these files is by object:

- Files Related to WebCenter Portal Framework Applications
- Files Related to JSR 286 Portlets
- Files Related to PDK-Java Portlets
- System Files
- Files Related to Security
- Oracle JDeveloper Files
- Files Related to WebCenter Portal Tools and Services

A.3 Files Related to WebCenter Portal Framework Applications

This section describes the files that are created for you when you create an application using WebCenter Portal's Portal Framework application template. It includes the following topics:

- 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.3.10, "navigation-renderer.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 runtime administration console 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 10, "Developing a Navigation Model."

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 5.11, "Changing the Default Home Page and Login/Logout Target Pages."

A.3.5 navigation-registry.xml

navigation-registry.xml is used by the runtime administration console 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 10, "Developing a Navigation Model."

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 5.3.2, "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 Oracle 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, links, tags, and discussions.

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 74.9, "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 10.8, "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
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.jspx but with different graphics. For more information, see Chapter 11, "Developing Page Templates."

A.4 Files Related to JSR 286 Portlets

This section describes the files that are created for you when you build a JSR 286 portlet. It includes the following topics:

- 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:


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>
  <portlet-info>
    <title>Map Portlet</title>
    <short-title>Map</short-title>
    <keywords/>
    <portlet-preferences>
      <preference>
        <name>portletTitle</name>
```
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:

  ```xml
  <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:

  ```xml
  <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`
A.4.4 portlet_name.java

del.epte 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_nameBundle.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 Fusion Developer’s Guide for Oracle Application Development Framework.

A.5 Files Related to PDK-Java Portlets

This section describes the files that are created for you when you build a PDK-Java portlet. It includes the following topics:

- Section A.5.1, "producer_name.properties"
- Section A.5.2, "_default.properties"
- Section A.5.3, "index.jsp"
- Section A.5.4, "portlet_name_modePage.jsp"
- 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 57.4.2, "Portlet Modes".

A.5.5 provider.xml

provider.xml is the definition file for your PDK-Java producer.

Example A–2 provides a sample provider.xml file.

Example A–2 provider.xml Sample

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<providerDefinition version="3.1">
  <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>
```
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 *Fusion Developer’s Guide for Oracle Application Development Framework*.

A.6  **System Files**

This section describes the files that are created or modified when you create or modify pages, deploy your application, or consume JSR 268 portlets. It includes the following topics:

- 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, "trinidad-skins.xml"
- Section A.6.8, "web.xml"
- Section A.6.9, "weblogic-application.xml"
- Section A.6.10, "mds Subdirectory"
- Section A.6.11, "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 tools and services, including pages, portlets, and all the tools and services such as search, worklist, discussions, and so on, to set configuration information. This file, with `connections.xml`, contains the configuration information required for most of the tools and services to function. For more information, see:

- Section 19.2.1.3, "Registering Add-Ons in adf-config.xml"
A.6.2 DataBindings.cpx

DataBindings.cpx is a file common to web applications. For more information about DataBindings.cpx, see the 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 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 Fusion Developer’s Guide for Oracle Application Development Framework.

PageDef.xml Sample

Example A–3 provides a sample PageDef.xml file.

Example A–3 PageDef.xml Sample

```xml
<?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>
</pageDefinition>
```
<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>
A.6.6 trinidad-config.xml

In JDeveloper, when you create a Portal Framework application, a trinidad-config.xml file is automatically created in the /WEB-INF directory. Example A–4 shows this seeded trinidad-config.xml file.

Example A–4 Seeded trinidad-config.xml File Created by JDeveloper

```xml
<?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>, which 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.
Additionally, trinidad-config.xml 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 the "Configuration in trinidad-config.xml" section in the Web User Interface Developer’s Guide for Oracle Application Development Framework.

A.6.7 trinidad-skins.xml

trinidad-skins.xml 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 Portal 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, see the "Customizing the Appearance Using Styles and Skins" chapter in the Web User Interface Developer’s Guide for Oracle Application Development Framework.

A.6.8 web.xml

web.xml is a Java EE standard descriptor that contains details about web applications. For more information about web.xml, see the Fusion Developer’s Guide for Oracle Application Development Framework.

A.6.9 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.6.10 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.6.11 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 describes the files that are created or modified when you implement or modify security. It includes the following topics:

- 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
<?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
                    xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                    <input name="setPortletProperties"/>
                  </operation>
                </operation>
                <operation name="getPortletProperties">
                  <soap
                    xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                    <input name="getPortletProperties"/>
                  </operation>
                </operation>
                <operation name="importPortlets">
                  <soap
                    soapAction="urn:oasis:names:tc:wsrp:v2:importPortlet"
                    xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                    <input name="importPortlets"/>
                  </operation>
                </operation>
                <operation name="destroyPortlets">
                  <soap
                    soapAction="urn:oasis:names:tc:wsrp:v2:destroyPortlets"
                    xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                    <input name="destroyPortlets"/>
                  </operation>
                </operation>
              </port>
            </service>
          </model>
        </wsconnection>
      </Contents>
    </XmlRefAddr>
  </Reference>
</References>
```
Files Related to Security

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"/>
<operation 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/soap/
operation name="initCookie">
<soap
soapAction="urn:oasis:names:tc:wsrp:v2:initCookie"
xmlns="http://schemas.xmlsoap.org/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/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/soap/
<input name="releaseSessions"/>
<output name="releaseSessionsResponse"/>
</operation>
<operation name="performBlockingInteraction">
<soap
xmlns="http://schemas.xmlsoap.org/soap/
<input name="performBlockingInteraction"/>
<output name="performBlockingInteractionResponse"/>
</operation>
<operation name="getResource">
<soap
xmlns="http://schemas.xmlsoap.org/soap/
<input name="getResource"/>
<output name="getResourceResponse"/>
</operation>
<operation name="handleEvents">
<soap
soapAction="urn:oasis:names:tc:wsrp:v2:handleEvents"
xmlns="http://schemas.xmlsoap.org/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/soap/
operation name="register">
<soap
soapAction="urn:oasis:names:tc:wsrp:v2:register"
xmlns="http://schemas.xmlsoap.org/soap/
}
<operation name="register">  
<soap  
soapAction="urn:oasis:names:tc:wsrp:v2:register"  
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"  
<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"/>
</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 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">
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 Fusion Developer's Guide for Oracle Application Development Framework.

**jazn-data.xml Sample**

Example A–6 provides a sample `jazn-data.xml` file.

```
<?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>
          </app-role>
          <app-role>
            <name>authenticated-role</name>
            <display-name>authenticated-users</display-name>
          </app-role>
          <class>oracle.security.jps.service.policystore.ApplicationRole</class>
        </app-roles>
      </application>
    </applications>
  </policy-store>
</jazn-data>
```

A.7.2 jazn-data.xml
<name>aprole</name>
<display-name>aprole</display-name>
<description>aprole</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</class>
            <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>
<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 "OPSS Configuration File Reference" appendix in the Securing Applications with Oracle Platform Security Services.

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 "Migrating Identities Manually" section in the Securing Applications with Oracle Platform Security Services.

The policy store is migrated in the following ways: entire policy store, application-specific policies, and global polices. For details, see the "Migrating Policies Manually" section in the Securing Applications with Oracle Platform Security Services.

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 "Migrating Credentials Manually" section in the Securing Applications with Oracle Platform Security Services.

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>
A.8 Oracle JDeveloper Files
This section describes the pagetemplate-metadata.xml file that is created or modified when you create a page template.

pagetemplate-metadata.xml keeps track of all the page templates you create in a project.

A.9 Files Related to WebCenter Portal Tools and Services
This section includes sample connections.xml and adf-config.xml files for some WebCenter Portal tools and 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.SesConnectionFactory" 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>
            <SoapUrl>http://host:port/search/query/OracleSearch</SoapUrl>
          </SesConfig>
        </Contents>
      </XmlRefAddr>
      <StringRefAddr addrType="appUsername">wpadmin</StringRefAddr>
      <SecureRefAddr addrType="appPassword"/>
    </RefAddresses>
  </Reference>

Example A–10 shows a sample adf-config.xml for documents.

Example A–10  ADF Configuration for the Documents
<doclibC:adf-doclib-config xmlns="http://xmlns.oracle.com/webcenter/doclib/config" primaryConnectionName="Oracle_Content_Server">
  <doclibC:spaces-repository

Note: The credentialStoreLocation entry in the sample is valid at design time only.
Example A–11 shows a sample adf-config.xml for worklists.

**Example A–11  ADF Configuration for Worklists**

<worklistC:adf-worklist-config>
  <worklistC:ConnectionName>WebCenter-Worklist</worklistC:ConnectionName>
</worklistC:adf-worklist-config>
This appendix provides reference information about Composer-specific tags, configuration files, and style properties.

This appendix includes the following topics:

- 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 Components Style-Specific Properties"
- Section B.5, "Customizable Components (HTML) Properties"

The information provided in this chapter is useful while performing the tasks described in Chapter 18, "Enabling Runtime Editing of Pages Using Composer" and Chapter 19, "Extending Runtime Editing Capabilities Using Composer."

**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 18.1, "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 "Extended Metadata Properties" section in *Fusion Developer’s Guide for Oracle Application Development Framework*.

---

This section contains the following subsections:

- Section B.1.1, "Page Customizable Component"
- 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"
B.1.1 Page Customizable Component

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 runtime, 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>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>No</td>
<td>The identifier for the component. The identifier must follow a subset of the syntax allowed in HTML:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ Must not be a zero-length String.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ First character must be an ASCII letter (A-Za-z) or an underscore (’_’).</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ Subsequent characters must be an ASCII letter or digit (A-Za-z0-9), an underscore (’_’), or a dash (’-’).</td>
</tr>
<tr>
<td>rendered</td>
<td>Boolean</td>
<td>Yes</td>
<td>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.</td>
</tr>
</tbody>
</table>
### Appearance Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sourceViewPosition</td>
<td>String</td>
<td>Yes</td>
<td>The position of the Structure view pane displaying components in a tree structure. Valid values are bottom, end, start, and top. Default value is top. For more information, see the section, Section 16.6, “Editing Capabilities in Structure View in Page Edit Mode.”.</td>
</tr>
<tr>
<td>sourceViewSize</td>
<td>String</td>
<td>Yes</td>
<td>The height or width of the Structure 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 Structure view pane enables users to alter the height or width of the pane at runtime.</td>
</tr>
<tr>
<td>editModeView</td>
<td></td>
<td></td>
<td>Specifies the default view in which to open a page in Composer. Valid options are design and source. Default value is design. If you select source, the page is opened in Structure view in Composer. If you select design, the page is opened in Design view in Composer if the value design or all is specified for the designViews attribute. Else the page is displayed in Add Content view.</td>
</tr>
<tr>
<td>rendered</td>
<td>Boolean</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>toolbarLayout</td>
<td>String</td>
<td>Yes</td>
<td>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 19.10, “Customizing the Composer Toolbar.”</td>
</tr>
</tbody>
</table>

### Style Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>styleClass</td>
<td>String</td>
<td>Yes</td>
<td>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.</td>
</tr>
</tbody>
</table>
### Table B–1 (Cont.) Attributes of a Page Customizable Component

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inlineStyle</td>
<td>String</td>
<td>Yes</td>
<td>The CSS styles to use for this component. This is intended for basic style changes. The <code>inlineStyle</code> is a set of CSS styles that are applied to the root DOM element of the component. If the <code>inlineStyle</code>'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 <code>::label</code> or <code>::icon-style</code>.</td>
</tr>
<tr>
<td>binding</td>
<td><code>oracle.adf.view.page.editor.compon.ent.PageCustomizable</code></td>
<td>Supports only EL</td>
<td>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.</td>
</tr>
<tr>
<td>sourceViewNodeActions</td>
<td>String</td>
<td>Yes</td>
<td>The name of the JSPX file to be called if you want to display custom actions against task flows in the component navigator in Composer Structure view. The <code>sourceViewNodeActions</code> 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 20.4, &quot;Enabling Custom Actions that Display on Task Flows in the Component Navigator.&quot;</td>
</tr>
<tr>
<td>allowLabel</td>
<td>Boolean</td>
<td>Yes</td>
<td>Specifies whether label creation must be enabled in the Customization Manager. When <code>allowLabel</code> is set to <code>true</code>, a Save and Label button is displayed on the Composer toolbar. For more information, see Section 16.5.15, &quot;Manage Application Customizations.&quot; This attribute is relevant only if you have configured sandbox support in your application.</td>
</tr>
<tr>
<td>catalog</td>
<td>String</td>
<td>Yes</td>
<td>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 14, &quot;Developing Resource Catalogs.&quot;</td>
</tr>
<tr>
<td>allowEL</td>
<td>Boolean</td>
<td>No</td>
<td>Specifies whether or not EL can be run in application pages. Default is true. This setting overrides the <code>&lt;pe:allow-el&gt;</code> setting in <code>adf-config.xml</code>. For more information, see Section 19.4.2, &quot;How to Protect Expression Language.&quot;</td>
</tr>
<tr>
<td>protectEL</td>
<td>Boolean</td>
<td>No</td>
<td>Specifies whether or not EL in form elements is read-only. Default is false. This setting overrides the <code>&lt;pe:protect-el&gt;</code> setting in <code>adf-config.xml</code>. For more information, see Section 19.4.2, &quot;How to Protect Expression Language.&quot;</td>
</tr>
</tbody>
</table>
Composer Component Properties

### 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–1 (Cont.) Attributes of a Page Customizable Component

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>designViews</td>
<td>String</td>
<td>Yes</td>
<td>Specifies the views available to a user in Composer. By default, Add Content and Structure views are available for backward compatibility. This attribute can take a space-separated list of elements. Available options are design, add-content, source, select, preview, and all. If you set the value as all, all the views are rendered except Add Content, and Design view becomes the default view in Composer. If the designViews attribute is set, it overrides the designViews setting in adf-config.xml. However, if this attribute is set to an EL that returns null, then the adf-config.xml setting for designViews is used. For more information see Section 19.11, &quot;Enabling Direct Select in Design or Add Content View.&quot;</td>
</tr>
</tbody>
</table>

**Customization Attributes**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customizationAllowed</td>
<td>Boolean</td>
<td>Specifies whether customizations are allowed on this component. Available values are true and false. The default value is true.</td>
</tr>
<tr>
<td>customizationAllowedBy</td>
<td>String</td>
<td>Specifies the roles for which customization is enabled.</td>
</tr>
</tbody>
</table>

---

**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

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Common Attributes</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The identifier for the component. The identifier must follow a subset of the syntax allowed in HTML:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ Must not be a zero-length String.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ First character must be an ASCII letter (A-Za-z) or an underscore (&quot;_&quot; ).</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ Subsequent characters must be an ASCII letter or digit (A-Za-z0-9), an underscore (&quot;_&quot;), or a dash (&quot;-&quot;).</td>
</tr>
<tr>
<td>rendered</td>
<td>Boolean</td>
<td>Yes</td>
<td>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).</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The default value is true.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> 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.</td>
</tr>
<tr>
<td><strong>Behavior Attribute</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>immediate</td>
<td>Boolean</td>
<td>Yes</td>
<td>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.</td>
</tr>
</tbody>
</table>

### Advanced Attributes
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>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>binding</td>
<td>oracle.adf.view.page.editor.component.ChangeModeLink or oracle.adf.view.page.editor.component.ChangeModeButton</td>
<td>Supports only EL</td>
<td>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.</td>
</tr>
</tbody>
</table>

Customization Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customizationAllowed</td>
<td>Boolean</td>
<td>Specifies whether customizations are allowed on this component. Available values are true and false. The default value is true.</td>
</tr>
<tr>
<td>customizationAllowedBy</td>
<td>String</td>
<td>Specifies the roles for which customization is enabled.</td>
</tr>
</tbody>
</table>

Other Attribute

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customizationId</td>
<td>This attribute has been deprecated. Use the id attribute.</td>
</tr>
</tbody>
</table>

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.
### Table B–3 Attributes of the Layout Customizable Component

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Common Attributes</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| 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 22, “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' |
Composer Component Properties

**Table B–3 (Cont.) Attributes of the Layout Customizable Component**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>accessKey</td>
<td>Char</td>
<td>Yes</td>
<td>A character used to gain quick access to this button. For accessibility reasons, this functionality is not supported in screen reader mode.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>This attribute is sometimes referred to as the “mnemonic”.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The character specified by this attribute must exist in the <code>Text</code> attribute of this button instance. If it does not, the user receives no visual indication of the existence of the <code>accessKey</code>. The easiest, and most convenient way to specify both the text and the mnemonic together is to use <code>textAndAccessKey</code>.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Note that the <code>accessKey</code> 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+&lt;accessKey&gt;. Firefox 2.0 on some operating systems sets focus when you press Alt+Shift+&lt;accessKey&gt;. Firefox 2.0 on other operating systems set focus when you press Control+&lt;accessKey&gt;. Refer to your browser's documentation for how it treats accessKey.</td>
</tr>
<tr>
<td>showIcon</td>
<td>Boolean</td>
<td>Yes</td>
<td>Specifies whether the default Change Layout icon is visible or not. The default value is true.</td>
</tr>
<tr>
<td>showLayoutChanger</td>
<td>Boolean</td>
<td>Yes</td>
<td>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:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Displayed in View mode.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Displayed in Edit mode.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>It is always displayed in Edit mode, even if the value of the showLayoutChanger attribute is false.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>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.</td>
</tr>
</tbody>
</table>
Table B–3 (Cont.) Attributes of the Layout Customizable Component

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>showTypes</td>
<td>String</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>shortDesc</td>
<td>String</td>
<td>Yes</td>
<td>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.</td>
</tr>
</tbody>
</table>

**Style Attributes**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>styleClass</td>
<td>String</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>inlineStyle</td>
<td>String</td>
<td>Yes</td>
<td>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.</td>
</tr>
</tbody>
</table>

**Advanced Attribute**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports only EL</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>binding</td>
<td>oracle.adf.view.page.editor.component.LayoutCustomizable</td>
<td>Supports only EL</td>
<td>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=&quot;#{yourManagedBean.Binding}&quot;</td>
</tr>
</tbody>
</table>

**Customization Attributes**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customizationAllowed</td>
<td>Boolean</td>
<td>Specifies whether customizations are allowed on this component. Available values are true and false. The default value is true.</td>
</tr>
<tr>
<td>customizationAllowedBy</td>
<td>String</td>
<td>Specifies the roles for which customization is enabled.</td>
</tr>
</tbody>
</table>

**Other Attribute**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customizationId</td>
<td>This attribute has been deprecated. Use the id attribute.</td>
</tr>
</tbody>
</table>

**Supported Facets**

Table B–4 describes the facets provided for a Layout Customizable component.
Table B–4  Facets of the Layout Customizable Component

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contentA</td>
<td>Prepopulated with a Panel Customizable component. This Panel Customizable</td>
</tr>
<tr>
<td></td>
<td>contains content to be rendered in area A in the layouts shown in Table B–5.</td>
</tr>
<tr>
<td>contentB</td>
<td>Prepopulated with a Panel Customizable component. This Panel Customizable</td>
</tr>
<tr>
<td></td>
<td>contains content to be rendered in area B in the layouts shown in Table B–5.</td>
</tr>
<tr>
<td>facetSeparator</td>
<td>Content to be rendered once between each facet.</td>
</tr>
<tr>
<td>separator</td>
<td>Content to be rendered once between each of the other children.</td>
</tr>
</tbody>
</table>

Predefined Layout Types

When you add a Layout Customizable component to the page, three child Panel Customizable components are added by default; one direct child and two Panel Customizable components 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

<table>
<thead>
<tr>
<th>Layout Type</th>
<th>Image Showing Content Arrangement</th>
<th>Note: In a oneColumn layout, A and B are rendered only if they contain child components.</th>
</tr>
</thead>
<tbody>
<tr>
<td>oneColumn</td>
<td><img src="image" alt="oneColumn" /></td>
<td></td>
</tr>
<tr>
<td>threeColumn</td>
<td><img src="image" alt="threeColumn" /></td>
<td></td>
</tr>
<tr>
<td>threeColumnNarrow</td>
<td><img src="image" alt="threeColumnNarrow" /></td>
<td></td>
</tr>
<tr>
<td>twoColumn</td>
<td><img src="image" alt="twoColumn" /></td>
<td></td>
</tr>
<tr>
<td>twoColumnBottom</td>
<td><img src="image" alt="twoColumnBottom" /></td>
<td></td>
</tr>
<tr>
<td>twoColumnNarrowLeft</td>
<td><img src="image" alt="twoColumnNarrowLeft" /></td>
<td></td>
</tr>
</tbody>
</table>
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>
<thead>
<tr>
<th>Layout Type</th>
<th>Image Showing Content Arrangement</th>
</tr>
</thead>
<tbody>
<tr>
<td>twoColumnNarrowRight</td>
<td><img src="imagename" alt="Image" /></td>
</tr>
<tr>
<td>twoColumnTop</td>
<td><img src="imagename" alt="Image" /></td>
</tr>
</tbody>
</table>

Table B–5 (Cont.) Values for Type Attribute of Layout Customizable
## Table B–6  Attributes of a Panel Customizable Component

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Common Attributes</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>id</td>
<td>String</td>
<td>No</td>
<td>The identifier for the component. The identifier must follow a subset of the syntax allowed in HTML:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Must not be a zero-length String.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- First character must be an ASCII letter (A-Za-z) or an underscore (‘_’).</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Subsequent characters must be an ASCII letter or digit (A-Za-z0-9), an underscore (‘_’), or a dash (‘-’).</td>
</tr>
<tr>
<td>rendered</td>
<td>Boolean</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>halign</td>
<td>String</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>valign</td>
<td>String</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>shortDesc</td>
<td>String</td>
<td>Yes</td>
<td>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.</td>
</tr>
</tbody>
</table>
Table B–6 (Cont.) Attributes of a Panel Customizable Component

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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. |

<table>
<thead>
<tr>
<th>Style Attributes</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>styleClass</td>
<td>String</td>
<td>Yes</td>
<td>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.</td>
</tr>
</tbody>
</table>
Table B–6  (Cont.) Attributes of a Panel Customizable Component

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inlineStyle</td>
<td>String</td>
<td>Yes</td>
<td>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.</td>
</tr>
</tbody>
</table>

Behavior Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>allowAction</td>
<td>Boolean</td>
<td>Yes</td>
<td>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 22.5, &quot;Applying Action-Level Restrictions on Panel Customizable and Show Detail Component Actions.&quot;</td>
</tr>
<tr>
<td>showEditAction</td>
<td>Boolean</td>
<td>Yes</td>
<td>While in Edit mode, in Add Content or 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.</td>
</tr>
</tbody>
</table>
### Table B–6  (Cont.) Attributes of a Panel Customizable Component

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>showSplitAction</td>
<td>Boolean</td>
<td>Yes</td>
<td>In Add Content or 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.</td>
</tr>
<tr>
<td>showTabAction</td>
<td>Boolean</td>
<td>Yes</td>
<td>In Add Content or 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.</td>
</tr>
</tbody>
</table>

#### 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. |

#### Customization Attributes

<table>
<thead>
<tr>
<th>customizationAllowed</th>
<th>Boolean</th>
<th>Specifies whether customizations are allowed on this component. Available values are true and false. The default value is true.</th>
</tr>
</thead>
<tbody>
<tr>
<td>customizationAllowed</td>
<td>String</td>
<td>Specifies the roles for which customization is enabled.</td>
</tr>
</tbody>
</table>

#### Other Attribute

| customizationId     | This attribute has been deprecated. Use the id attribute.                                                                       |
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 Add Content or 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 Add Content or 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 Add Content or Design view. However, users can still select such a component in Structure 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

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Common Attributes</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>id</td>
<td>String</td>
<td>No</td>
<td>The identifier for the component. The identifier must follow a subset of the syntax allowed in HTML:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ Must not be a zero-length String.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ First character must be an ASCII letter (A-Za-z) or an underscore ('_').</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ Subsequent characters must be an ASCII letter or digit (A-Za-z0-9), an underscore ('_'), or a dash ('-').</td>
</tr>
<tr>
<td>rendered</td>
<td>Boolean</td>
<td>Yes</td>
<td>Specifies whether the component is rendered. When set to <code>false</code>, 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 <code>true</code>. The rendering of a component can also be defined at runtime using the Show Component and Hide Component options.</td>
</tr>
<tr>
<td>text</td>
<td>String</td>
<td>Yes</td>
<td>A title for the Show Detail Frame component. You can change the style of this text using the <code>headerStyle</code> attribute.</td>
</tr>
<tr>
<td>icon</td>
<td>String</td>
<td>Yes</td>
<td>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: <code>icon=&quot;http://source-pc/images/accessability.gif&quot;</code> <strong>Note:</strong> An image that is stored at the document root does not require a full path. For example: <code>icon=&quot;detail.gif&quot;</code></td>
</tr>
<tr>
<td><strong>Appearance Attributes</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>text</td>
<td>String</td>
<td>Yes</td>
<td>A title for the Show Detail Frame component. You can change the style of this text using the <code>headerStyle</code> attribute.</td>
</tr>
</tbody>
</table>
### Table B–7 (Cont.) Attributes of a Show Detail Frame Component

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>shortDesc</td>
<td>String</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>icon</td>
<td>String</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>icon=&quot;<a href="http://source-pc/images/accessability.gif">http://source-pc/images/accessability.gif</a>&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Note: An image that is stored at the document root does not require a full path. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>icon=&quot;detail.gif&quot;</td>
</tr>
<tr>
<td>background</td>
<td>String</td>
<td>Yes</td>
<td>Working with the skin CSS, provides a means of applying a different look and feel for this Show Detail Frame instance.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Available values are light, medium, dark, and coreDefault. The default value is medium.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>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.</td>
</tr>
<tr>
<td>displayHeader</td>
<td>Boolean</td>
<td>Yes</td>
<td>Indicates whether the header of the Show Detail Frame is displayed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The default value is true.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>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.</td>
</tr>
<tr>
<td>displayShadow</td>
<td>Boolean</td>
<td>Yes</td>
<td>Specifies whether a shadow is cast by the Show Detail Frame component.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The default value is false.</td>
</tr>
</tbody>
</table>
Table B–7 (Cont.) Attributes of a Show Detail Frame Component

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>expansionMode</td>
<td>String</td>
<td>Yes</td>
<td>The default state of the Show Detail Frame. Available values are minimized and normal. The default display mode is normal.</td>
</tr>
<tr>
<td>rendered</td>
<td>Boolean</td>
<td>Yes</td>
<td>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.</td>
</tr>
</tbody>
</table>

Actions Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>displayActions</td>
<td>String</td>
<td>Yes</td>
<td>Specifies when seeded interactions are displayed. Available options are onHover and always. The default is always.</td>
</tr>
<tr>
<td>inheritGlobalActions</td>
<td>Boolean</td>
<td>Yes</td>
<td>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 20.3.2.2, &quot;Defining Custom Actions at the Global Level.&quot;</td>
</tr>
<tr>
<td>showMoveAction</td>
<td>String</td>
<td>Yes</td>
<td>Specifies whether the Move action is available for the component. Available values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- menu—Move action is displayed in the Actions menu, and the action to drag and drop the header is enabled</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- none—User can move the component by clicking the header and using drag and drop</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- disabled—Component cannot be moved</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The default value is menu.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> The disabled option is applicable only in View mode.</td>
</tr>
</tbody>
</table>
### Table B–7 (Cont.) Attributes of a Show Detail Frame Component

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>showRemoveAction</td>
<td>String</td>
<td>Yes</td>
<td>Renders a <strong>Remove</strong> icon on the Show Detail Frame header. Clicking this icon removes the component from the page. Available values are <strong>chrome</strong> and <strong>none</strong>. The default value is <strong>none</strong>. Note: In page Edit mode, the Remove icon appears regardless of this setting, provided other restrictions have not been applied.</td>
</tr>
<tr>
<td>showResizer</td>
<td>String</td>
<td>Yes</td>
<td>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 <strong>never</strong> and <strong>always</strong>. The default value is <strong>always</strong>.</td>
</tr>
<tr>
<td>showMinimizeAction</td>
<td>String</td>
<td>Yes</td>
<td>Specifies whether the minimize action is displayed on the header. Available values are <strong>chrome</strong> and <strong>none</strong>. The default is <strong>chrome</strong>.</td>
</tr>
<tr>
<td>showEditAction</td>
<td>Boolean</td>
<td>Yes</td>
<td>While in Edit mode, in Add Content or Design view, renders an <strong>Edit</strong> icon on the Show Detail Frame header that enables you to edit component properties at runtime. Default value is <strong>true</strong>. This is relevant only in a Composer-enabled page, that is, if the Show Detail Frame component is nested within a Page Customizable component.</td>
</tr>
</tbody>
</table>

### Style Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contentStyle</td>
<td>String</td>
<td>Yes</td>
<td>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. The CSS styles in this attribute are commonly used to define the height and background color of the Show Detail Frame. For example, <code>height:200px; background-color:green;</code> 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.</td>
</tr>
</tbody>
</table>
### Table B-7 Attributes of a Show Detail Frame Component

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>styleClass</td>
<td>String</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>inlineStyle</td>
<td>String</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>headerStyle</td>
<td>String</td>
<td>Yes</td>
<td>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. The CSS styles in this attribute are commonly used to define the font, size, and color of the title text.</td>
</tr>
</tbody>
</table>

#### Behavior Attributes

<table>
<thead>
<tr>
<th>partialTriggers</th>
<th>String</th>
<th>Yes</th>
<th>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 - &quot;::&quot; pop out of the component's naming container (or itself if the component is a naming container) and begin the search from there, &quot;:::&quot; pop out of two naming containers (including itself if the component is a naming container) and begin the search from there.</th>
</tr>
</thead>
<tbody>
<tr>
<td>disclosureListener</td>
<td>javax.el.MethodExpression</td>
<td>Yes</td>
<td>A method reference to a disclosure listener. A disclosure event is fired when the user expands or collapses the Show Detail Frame component.</td>
</tr>
</tbody>
</table>

Composer Component Properties
#### 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

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>binding</td>
<td>oracle.adf.view.page.editor.component.ShowDetailFrame</td>
<td>Supports only EL</td>
<td>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.</td>
</tr>
<tr>
<td>attributeChangeListener</td>
<td>javax.el.MethodExpression</td>
<td>Yes</td>
<td>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.</td>
</tr>
</tbody>
</table>
Table B–7 (Cont.) Attributes of a Show Detail Frame Component

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>selectChild</td>
<td>String</td>
<td>Yes</td>
<td>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.</td>
</tr>
</tbody>
</table>

Customization Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customizationAllowed</td>
<td>Boolean</td>
<td>Specifies whether customizations are allowed on this component. Available values are true and false. The default value is true.</td>
</tr>
<tr>
<td>customizationAlloweBy</td>
<td>String</td>
<td>Specifies the roles for which customization is enabled.</td>
</tr>
</tbody>
</table>
**Table B–7  (Cont.) Attributes of a Show Detail Frame Component**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>childCreation</td>
<td>String</td>
<td>Yes</td>
<td>Specifies when the children (contents) of the Show Detail Frame component are created. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- immediate (default): All children are immediately created when the Show Detail Frame component is rendered, even when it is collapsed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- lazy: Children are created when the Show Detail Frame component is likely to be expanded. This behavior optimizes performance by postponing the creation of children until the Show Detail Frame parent component is expanded. The children remain in cached memory for subsequent display.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- lazyUncached: Children are created when the Show Detail Frame parent component is likely to be expanded, and children are deleted when no longer visible. This means children do not reside in cached memory when not visible so must be re-created each time the parent component is expanded.</td>
</tr>
<tr>
<td>customizationId</td>
<td></td>
<td></td>
<td>This attribute has been deprecated. Use the id attribute.</td>
</tr>
<tr>
<td>helpTopicID</td>
<td>String</td>
<td>Yes</td>
<td>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.</td>
</tr>
</tbody>
</table>

**Supported Facets**

Table B–8 describes the facets available on a Show Detail Frame component.

**Table B–8  Show Detail Frame Facets**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>titleBarAction</td>
<td>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.</td>
</tr>
</tbody>
</table>
Displaying Child Component Properties Along with Show Detail Frame Properties

When you switch to the Edit mode of a page at runtime, in Add Content or 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 Add Content or 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 Add Content or 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 Structure 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.
Table B–9 describes the attributes of a Custom Action component.

Table B–9  Attributes of a Custom Action Component

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Common Attributes</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| 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

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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. |

### Customization Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customizationAllowed</td>
<td>Boolean</td>
<td>Specifies whether customizations are allowed on this component. Available values are true and false. The default value is true.</td>
</tr>
</tbody>
</table>
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 19.3.6, "How to Display Properties and Parameters in a Custom Property Panel."

Table B–9 describes the new attributes of a Show Property component.

**Table B–9 (Cont.) Attributes of a Custom Action Component**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>No</td>
<td>The identifier for the component. The identifier must follow a subset of the syntax allowed in HTML:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ Must not be a zero-length String.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ First character must be an ASCII letter (A-Za-z) or an underscore (_).</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ Subsequent characters must be an ASCII letter or digit (A-Za-z0-9), an underscore (_), or a dash (-).</td>
</tr>
<tr>
<td>rendered</td>
<td>Boolean</td>
<td>Yes</td>
<td>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).</td>
</tr>
<tr>
<td>attributeType</td>
<td>String</td>
<td>Yes</td>
<td>Specifies the property type. Supported values are text, cbox, date, lov.</td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
<td>Supports EL?</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>---------</td>
<td>--------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>label</td>
<td>String</td>
<td>Yes</td>
<td>Specifies the label of the component. If you want the label to appear above the control, use a panelFormLayout.</td>
</tr>
<tr>
<td>selectItem</td>
<td></td>
<td></td>
<td>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.</td>
</tr>
<tr>
<td>columns</td>
<td>Int</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>rows</td>
<td>Int</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>label</td>
<td>String</td>
<td>Yes</td>
<td>Specifies the label of the component. If you want the label to appear above the control, use a panelFormLayout.</td>
</tr>
<tr>
<td>shortDesc</td>
<td>String</td>
<td>Yes</td>
<td>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.</td>
</tr>
</tbody>
</table>
### Table B–10 (Cont.) Attributes of a Show Property Component

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>simple</td>
<td>Boolean</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>unselectedLabel</td>
<td>String</td>
<td>Yes</td>
<td>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.</td>
</tr>
</tbody>
</table>

#### Behavior Attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>required</td>
<td>Boolean</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>autoSubmit</td>
<td>Boolean</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>disabled</td>
<td>Boolean</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>validator</td>
<td>String</td>
<td>Yes</td>
<td>Specifies a method reference to a validator method.</td>
</tr>
<tr>
<td>valueChangeListene</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>r</td>
<td>javax.faces.el.MethodBinder</td>
<td>EL only</td>
<td>Specifies a method reference to a value change listener.</td>
</tr>
</tbody>
</table>

#### Advanced Attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>binding</td>
<td>oracle.adf.view.rich.component.fragment.UIXDeclarativeComponent</td>
<td>Supports only EL</td>
<td>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.</td>
</tr>
</tbody>
</table>
### Attributes of a Show Property Component

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>clientComponent</td>
<td>Boolean</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>componentId</td>
<td></td>
<td></td>
<td>If you use the <strong>Show Property</strong> tag in Composer’s property panel, componentId specifies the ID of the component for which the property must be rendered.</td>
</tr>
<tr>
<td>showELBuilder</td>
<td>Boolean</td>
<td>Yes</td>
<td>Specifies whether to display the EL builder for the property.</td>
</tr>
<tr>
<td>showResourceEditor</td>
<td>Boolean</td>
<td>Yes</td>
<td>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 20.12, &quot;Configuring Runtime Resource String Editing.&quot;</td>
</tr>
<tr>
<td>taskFlowId</td>
<td>String</td>
<td>Yes</td>
<td>If <strong>Show Property</strong> is used to display task flow parameters, taskFlowId specifies the ID of the task flow.</td>
</tr>
<tr>
<td>taskFlowParam</td>
<td>String</td>
<td>Yes</td>
<td>If <strong>Show Property</strong> is used to display task flow parameters, taskFlowParam specifies the task flow parameter that the input field must display.</td>
</tr>
<tr>
<td>customizationId</td>
<td></td>
<td></td>
<td>This attribute has been deprecated. Use the <strong>id</strong> attribute.</td>
</tr>
</tbody>
</table>

### Customization Attributes

- **customizationAllowed** (Boolean): Specifies whether customizations are allowed on this component. Available values are **true** and **false**. The default value is **true**.
- **customizationAllowedBy** (String): Specifies the roles for which customization is enabled.

### Other Attribute

- **text** (String): Specifies the text used for the check box. This is used only when **type** is set to **boolean**.
Supported Facets

The Show Property component supports a customLink facet, which can be used to include a Command 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>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>An identifier that is used while referencing the new add-on in <code>adf-config.xml</code>, or a property panel in the <code>property-panels</code> section. This must be unique in the application.</td>
</tr>
<tr>
<td>title</td>
<td>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.</td>
</tr>
<tr>
<td>icon</td>
<td>The icon that appears next to the title on the button. This is optional.</td>
</tr>
<tr>
<td>taskflow-id</td>
<td>The ID of the task flow that defines the add-on or property panel.</td>
</tr>
<tr>
<td>help-topic-id</td>
<td>A help topic ID that links to a custom help topic from the add-on or property panel. This is optional.</td>
</tr>
</tbody>
</table>

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:
For more information about add-ons, see Section 19.2, "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 19.3, "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>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>An identifier for the custom property panel you want to display in the Component Properties dialog.</td>
</tr>
<tr>
<td>rendered</td>
<td>Whether this panel is displayed to users in the Component Properties dialog. This can take an EL value.</td>
</tr>
</tbody>
</table>

**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>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name with which you declared the panel in the addon-config section.</td>
</tr>
<tr>
<td>rendered</td>
<td>Whether the custom property panel must be rendered or not.</td>
</tr>
<tr>
<td>parameter</td>
<td>Parameters to be passed to the implementing task flow. Takes an EL only and must return a Map&lt;String, String&gt;.</td>
</tr>
</tbody>
</table>

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:

```xml
<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:

```xml
<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:

```xml
<lov-config>
  <task-flow-definition taskflow-id="/WEB-INF/employee-details.xml#employee-details">
    <input-parameter-definition>
      <name>Designation</name>
      <enumeration>
        <value>Manager</value>
        <value>Supervisor</value>
      </enumeration>
    </input-parameter-definition>
  </task-flow-definition>
</lov-config>
```
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 19.6, "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:

```xml
<drop-handlers>
  <drop-handler>test.TestDropHandler</drop-handler>
</drop-handlers>
```

For more information, see [Section 19.7, "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 19.8, "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>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>namespace</td>
<td>Namespace of the tag library containing the component whose properties you want to filter.</td>
</tr>
</tbody>
</table>

**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>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the component whose properties you want to filter, for example, commandButton, portlet.</td>
</tr>
</tbody>
</table>

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>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the property you want to filter.</td>
</tr>
<tr>
<td>filtered</td>
<td>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.</td>
</tr>
<tr>
<td>label</td>
<td>The label for the property in the UI. To retrieve localized strings or show a different label depending on the view, use EL.</td>
</tr>
<tr>
<td>view</td>
<td>(Optional) The view in which to show/hide this property. Takes a value of design or source.</td>
</tr>
</tbody>
</table>

The following example shows how you can define global and component-level property filters:

```xml
<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 19.4, "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:

```xml
<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 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 19.11, "Enabling Direct Select in Design or Add Content 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 direct select 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.

```xml
<selection-config>
  <global-filter>
    <selection view="design" enabled="false"/>
  </global-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"/>
    </tag>
  </taglib-filter>
</filter-config>
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 Portal Framework application:

```xml
<?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 Structure 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:

```xml
<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 19.2, "Creating Composer Add-Ons" and Section 19.3, "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:

```xml
<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:

```xml
<pe:add-on-panels>
  <pe:add-on-panel
    name="oracle.adf.pageeditor.addonpanels.customization-manager"
    parameters="#{AppUtilBean.customizationManagerParams}"/>
</pe:add-on-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 21.2.1, "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:

```xml
<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 21.3, "Adding Customization Layers to View and Edit Modes: Example."

The following example shows how to register a class named AppsSessionOptionsFactoryImpl:

```xml
<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 Select view. For more information on direct select functionality, see Section 19.11, "Enabling Direct Select in Design or Add Content View." Add the `<pe:enable-design-views>` tag inside the `<pe:page-editor-config>` tag to define the available views. The possible values are: design, add-content, select, source, preview, and all. The all value will show design, select, source, and preview only since add-content is a subset of design.
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 19.4.2, “How to Protect Expression Language.”

The following example disables EL:

```xml
<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 14.2, “Creating a 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:

```xml
<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:

```xml
<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 22.5.2, "Defining Security at the Actions Category Level."

The following example shows how `<actionsCategory>` can be used:

```xml
<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:

```xml
<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:actions>
        . . .
      </cust:actions>
    </cust:actionsCategory>
  </cust:customizableComponentsSecurity>
</adf-config>
```
For more information about applying action-level restrictions, see Section 22.5, "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 20.3.2, "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 22.1.1, "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:

```xml
<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:

```xml
<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 20.12, "Configuring Runtime Resource String Editing."

**B.2.2.11 enable-source-view**
Use the `<pe:enable-source-view>` element to enable or disable Structure view in Composer. Insert the `<pe:enable-source-view>` element within the `<pe:page-editor-config>` section. Structure 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:

```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>
```

For more information, see Section 20.8, "Disabling Structure View for the 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:

```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>
```

For more information, see Section 20.9, "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:

```xml
<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>
```

For more information about change persistence in Composer, see Section 20.11, "Configuring the Persistence Change Manager."

For more information about the `<persistent-change-manager>` element, see 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 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:

```xml
<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 20.11, "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.
Composer-Specific Files and Configurations

**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 22.6, "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
<?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-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>
    </mds-config>
  </mdsC:adf-mds-config>
</adf-config>
```
<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>
</mds:C: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:enableSecurity values="true"/>
<cust:actionsCategory>
<cust:actionCategory name="personalizeActionsCategory" value="false"/>
<cust:actionCategory name="editActionsCategory" value="true"/>
</cust:actionsCategory>
</cust:customizableComponentsSecurity>
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 20.3, "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:

```xml
<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:

```xml
<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 Portal 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 20.11, "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:

```xml
<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 21.3, "Adding Customization Layers to View and Edit Modes: Example."
- Enabling sandbox creation in Composer. For more information, see Section 21.2.1, "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 19.2, "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 Add Content or Design view; for details, see Section 19.11, "Enabling Direct Select in Design or Add Content View."

For information about creating property panels, see Section 19.3, "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 Parameters
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 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.4.1, "Style Selectors for Composer Components"
■ Section B.4.2, "Style Attributes"
B.4.1 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.4.1.1 Global Style Selectors

Use global style selectors to define styles for multiple components within the application. Table B–17 lists and describes the global style selectors relevant to Composer components.

Table B–17  Global Style Selectors

<table>
<thead>
<tr>
<th>Style Selector</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>.ComposerDark:alias</td>
<td>Specifies the default color for toolbars, headers, and so on in Composer, with a color scheme of dark.</td>
</tr>
<tr>
<td>.ComposerLight:alias</td>
<td>Specifies the default color for toolbars, headers, and so on in Composer, with a color scheme of light.</td>
</tr>
<tr>
<td>.ComposerBackground</td>
<td>Specifies the color scheme for Composer dialogs, such as the Component Properties and Page Properties dialogs.</td>
</tr>
<tr>
<td>.ComposerBorder</td>
<td>Specifies the border, width, and so on for Composer dialogs, such as the Component Properties and Page Properties dialogs.</td>
</tr>
<tr>
<td>.PEClickableImageAnchor:alias</td>
<td>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 &quot;display: block&quot; with height, width attributes to display in block mode.</td>
</tr>
<tr>
<td>.ChildPanelFacetWarning</td>
<td>Specifies the style for the warning message displayed in the child navigator in the Component Properties dialog.</td>
</tr>
</tbody>
</table>

B.4.1.2 Composer Status Indicator Style Selectors

Use the style selectors in Table B–18 to skin the status indicator component in the Composer toolbar.
### Table B–18 Status Indicator Style Selectors

<table>
<thead>
<tr>
<th>Style Selector</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>af</td>
<td>statusIndicator.ComposerStatus</td>
</tr>
<tr>
<td>af</td>
<td>statusIndicator.ComposerStatus::idle-icon</td>
</tr>
<tr>
<td>af</td>
<td>statusIndicator.ComposerStatus::processing-icon</td>
</tr>
<tr>
<td>af</td>
<td>statusIndicator.ComposerStatus::connecting-icon</td>
</tr>
<tr>
<td>af</td>
<td>statusIndicator.ComposerStatus::connected-icon</td>
</tr>
<tr>
<td>af</td>
<td>statusIndicator.ComposerStatus::reconnecting-icon</td>
</tr>
<tr>
<td>af</td>
<td>statusIndicator.ComposerStatus::disconnected-icon</td>
</tr>
</tbody>
</table>

### B.4.1.3 Page Customizable Style Selectors

Use the style selectors in Table B–19 to skin Page Customizable components.

### Table B–19 Page Customizable Style Selectors

<table>
<thead>
<tr>
<th>Style Selector</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>af</td>
<td>pageCustomizable</td>
</tr>
<tr>
<td>af</td>
<td>pageCustomizable af</td>
</tr>
<tr>
<td>af</td>
<td>pageCustomizable af</td>
</tr>
<tr>
<td>af</td>
<td>pageCustomizable af</td>
</tr>
</tbody>
</table>
### Table B–19 (Cont.) Page Customizable Style Selectors

<table>
<thead>
<tr>
<th>Style Selector</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>`af</td>
<td>pageCustomizable af</td>
</tr>
<tr>
<td>`af</td>
<td>pageCustomizable af</td>
</tr>
<tr>
<td>`af</td>
<td>pageCustomizable af</td>
</tr>
<tr>
<td>`af</td>
<td>pageCustomizable af</td>
</tr>
<tr>
<td>`af</td>
<td>pageCustomizable af</td>
</tr>
<tr>
<td>`af</td>
<td>pageCustomizable af</td>
</tr>
<tr>
<td>`af</td>
<td>pageCustomizable af</td>
</tr>
<tr>
<td>`af</td>
<td>pageCustomizable af</td>
</tr>
<tr>
<td>`af</td>
<td>pageCustomizable af</td>
</tr>
<tr>
<td>`af</td>
<td>dialog.ComposerDialog::content-start menuBar`</td>
</tr>
<tr>
<td>`af</td>
<td>dialog.ComposerDialog::content-start:rtl menuBar`</td>
</tr>
<tr>
<td>`af</td>
<td>pageCustomizable af</td>
</tr>
<tr>
<td>`af</td>
<td>pageCustomizable af</td>
</tr>
</tbody>
</table>
B.4.1.4 Layout Customizable Style Selectors

Use the style selectors in Table B–20 to skin Layout Customizable components.

<table>
<thead>
<tr>
<th>Style Selector</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>af/layoutCustomizable::menu:edit</td>
<td>Specifies the style for the Change Layout button in Edit mode.</td>
</tr>
<tr>
<td>af/commandImageLink::text</td>
<td>Specify the styles for the text on the Change Layout button in Edit mode.</td>
</tr>
<tr>
<td>af/layoutCustomizable::AFStretchWidth</td>
<td>Specifies the width to which the component can stretch horizontally.</td>
</tr>
</tbody>
</table>

B.4.1.5 Panel Customizable Style Selectors

Use the style selectors listed in Table B–21 to skin Panel Customizable components.

Table B–21 (Cont.) Page Customizable Style Selectors

<table>
<thead>
<tr>
<th>Style Selector</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>af</td>
<td>pageCustomizable:edit, af</td>
</tr>
<tr>
<td>af</td>
<td>pageCustomizable:printable</td>
</tr>
<tr>
<td>af</td>
<td>pageCustomizable af</td>
</tr>
</tbody>
</table>

Note: Some style selectors in Table B–21 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.
Use the style selectors listed in Table B–22 to skin portlets and Show Detail Frame components.
In Portal Framework applications, each portlet is rendered with portlet *chrome*. 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–22 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>
<thead>
<tr>
<th>Style Selector</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>`af</td>
<td>showDetailFrame`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::container:light`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::container:medium`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::container:dark`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::container:core:default`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame.p_AFStretched`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::content`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::content:light`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::content:medium`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::content:dark`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::content:core:default`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::header`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::header:light`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::header:medium`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::header:dark`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::header:core:default`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::header-center`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::header-center:light`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::header-center:medium`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::header-center:dark`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::header-center:core:default`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::header-start`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::header-start:rtl`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::header-start:light`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::header-start:light:rtl`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::header-start:medium`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::header-start:medium:rtl`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::header-start:dark`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::header-start:dark:rtl`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::header-start:core:default`</td>
</tr>
<tr>
<td>`af</td>
<td>showDetailFrame::header-start:core:default:rtl`</td>
</tr>
</tbody>
</table>
Composer Components Style-Specific Properties

**Table B–22 (Cont.) Show Detail Frame Style Selectors**

<table>
<thead>
<tr>
<th>Style Selector</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>af</td>
<td>showDetailFrame::header-end</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::header-end:rtl</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::header-end:light</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::header-end:light:rtl</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::header-end:medium</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::header-end:dark</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::header-end:dark:rtl</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::header-end:core:default</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::header-end:core:default:rtl</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::footer-start</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::footer-start:rtl</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::footer-start:light</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::footer-start:light:rtl</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::footer-start:medium</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::footer-start:dark</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::footer-start:dark:rtl</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::footer-start:core:default</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::footer-start:core:default:rtl</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::footer-center</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::footer-center:light</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::footer-center:medium</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::footer-center:dark</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::footer-center:core:default</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::footer-center:core:default:rtl</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::footer-end</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::footer-end:rtl</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::footer-end:light</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::footer-end:light:rtl</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::footer-end:medium</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::footer-end:dark</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::footer-end:dark:rtl</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::footer-end:core:default</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::footer-end:core:default:rtl</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::shadowbox</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::shadowbox:rtl</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::shadowbox_div1</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::shadowbox_div1:rtl</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::shadowbox_div2</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::shadowbox_div2:rtl</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::toolbar</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::toolbar:rtl</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::icon-style</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::header-text</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::header-text:rtl</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::header-actions</td>
</tr>
</tbody>
</table>

Composer Component Properties and Files  B-63
## Composer Components Style-Specific Properties

### Table B–22  (Cont.) Show Detail Frame Style Selectors

<table>
<thead>
<tr>
<th>Style Selector</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>af</td>
<td>showDetailFrame::toolbar-container</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::toolbar-container:light</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::toolbar-container:medium</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::toolbar-container:dark</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::toolbar-container:core:default</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::collapse-icon-style</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::collapse-icon-style:active</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::collapse-icon-style:hover</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::collapse-icon-style.PEDisplayNone</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::disclose-icon-style</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::disclose-icon-style:active</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::disclose-icon-style:hover</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::disclose-icon-style.PEDisplayNone</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::actionmenu-icon-style</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::actionmenu-icon-style:active</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::actionmenu-icon-style:hover</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::remove-icon-style</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::remove-icon-style:active</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::remove-icon-style:hover</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::edit-icon-style</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::edit-icon-style:active</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::edit-icon-style:hover</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::preview-icon-style</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::preview-icon-style:active</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::preview-icon-style:hover</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::edit-mode-content-style</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::edit-mode-content-style:light</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::edit-mode-edit-icon-style</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::edit-mode-edit-icon-style:active</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::edit-mode-edit-icon-style:hover</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::edit-mode-remove-icon-style</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::edit-mode-remove-icon-style:active</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::edit-mode-remove-icon-style:hover</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::edit-mode-remove-icon-style:active</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::edit-mode-remove-icon-style:hover</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::edit-mode-remove-icon-style:active</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::edit-mode-remove-icon-style:hover</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::edit-mode-remove-icon-style:hover:active</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::edit-mode-remove-icon-style:hover:active</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::edit-mode-remove-icon-style:hover:active</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::edit-text-link</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::edit-text-link:rtl</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::edit-text-link:rtl:rtl</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::edit-text-link:hover</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::resize</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::resize:rtl</td>
</tr>
</tbody>
</table>
### Table B–22 (Cont.) Show Detail Frame Style Selectors

<table>
<thead>
<tr>
<th>Style Selector</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>af</td>
<td>showDetailFrame::content.PEWrapsRTE &gt; af</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::content.PEStretched</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::content.PEStretched:printable</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::dynamic-help-icon-container af</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::dynamic-help-icon-style af</td>
</tr>
<tr>
<td>.PEShowDetailFrameAnimIcon</td>
<td>Specifies the animation icon to be used to show busy status on the component, for example, when you expand it.</td>
</tr>
<tr>
<td>.PEIconAndTextInHeading:alias .PEIconAndTextInHeading:alias:rtl</td>
<td>Specify the styles for rendering an icon with text in the component header.</td>
</tr>
</tbody>
</table>

### B.4.2 Style Attributes

Table B–23 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–23  Style Attributes of Composer and WebCenter Portal Customizable Components

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>background-color</td>
<td>The background color of the component. Choose from a list of colors, or click the Edit icon to select from a color palette.</td>
</tr>
<tr>
<td>background-image</td>
<td>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.</td>
</tr>
<tr>
<td>background-repeat</td>
<td>Specify whether and how the background image should repeat. Choose from:</td>
</tr>
<tr>
<td></td>
<td>■ &lt;none&gt;: Forgo repeating the image.</td>
</tr>
<tr>
<td></td>
<td>■ no-repeat: Forgo repeating the image.</td>
</tr>
<tr>
<td></td>
<td>■ repeat: Repeat the image to fill the container.</td>
</tr>
<tr>
<td></td>
<td>■ repeat-x: Repeat the image horizontally but not vertically.</td>
</tr>
<tr>
<td></td>
<td>■ repeat-y: Repeat the image vertically but not horizontally.</td>
</tr>
<tr>
<td></td>
<td>■ inherit: Inherit this setting from a parent component.</td>
</tr>
<tr>
<td>border-color</td>
<td>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.</td>
</tr>
<tr>
<td>border-style</td>
<td>The style of border to draw around the component. Choose from:</td>
</tr>
<tr>
<td></td>
<td>■ &lt;none&gt;: Apply no style to the border.</td>
</tr>
<tr>
<td></td>
<td>■ 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.)</td>
</tr>
<tr>
<td></td>
<td>■ hidden: Hide the border.</td>
</tr>
<tr>
<td></td>
<td>■ dotted: Display the border as a line of dots.</td>
</tr>
<tr>
<td></td>
<td>■ double: Display the border as a double line.</td>
</tr>
<tr>
<td></td>
<td>■ groove: Display the border as a grooved line.</td>
</tr>
<tr>
<td></td>
<td>■ ridge: Display the border as a ridged line.</td>
</tr>
<tr>
<td></td>
<td>■ inset: Display the border as a concave line.</td>
</tr>
<tr>
<td></td>
<td>■ outset: Display the border as a convex line.</td>
</tr>
<tr>
<td></td>
<td>■ dashed: Display the border as a line of dashes.</td>
</tr>
<tr>
<td></td>
<td>■ solid: Display the border as a solid line.</td>
</tr>
<tr>
<td></td>
<td>■ inherit: Inherit the border style from a parent component.</td>
</tr>
<tr>
<td>border-width</td>
<td>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.</td>
</tr>
<tr>
<td>color</td>
<td>The color of the component text. Select from a list, or click the Edit icon to select from a color palette.</td>
</tr>
<tr>
<td>font-family</td>
<td>The font family (such as Arial, Helvetica, or sans-serif) for the component text. Enter this value manually.</td>
</tr>
</tbody>
</table>
Table B–23  (Cont.) Style Attributes of Composer and WebCenter Portal Customizable Components

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>font-size</td>
<td>The size of component text. Choose from:</td>
</tr>
<tr>
<td></td>
<td>■ Choose Length: Specify an absolute value for the font-size in your preferred unit of measurement.</td>
</tr>
<tr>
<td></td>
<td>■ Choose Percentage: Specify font size as a percentage of the normal text size defined for the browser.</td>
</tr>
<tr>
<td></td>
<td>■ inherit: Inherit font size from a parent component.</td>
</tr>
<tr>
<td></td>
<td>■ large: Select the next size larger than the font associated with the parent component.</td>
</tr>
<tr>
<td></td>
<td>■ larger: Increase the font size to larger than the font associated with the parent component (larger is larger than large).</td>
</tr>
<tr>
<td></td>
<td>■ medium: Set the font size to the default font size of the parent component.</td>
</tr>
<tr>
<td></td>
<td>■ small: Set the font size to smaller than the default font size of the parent component.</td>
</tr>
<tr>
<td></td>
<td>■ smaller: Set the font size to smaller than the default font size of the parent component (smaller is smaller than small).</td>
</tr>
<tr>
<td>font-style</td>
<td>The font style of the component text. Choose from:</td>
</tr>
<tr>
<td></td>
<td>■ &lt;none&gt;: Apply no style to the font.</td>
</tr>
<tr>
<td></td>
<td>■ normal: The font is not italic.</td>
</tr>
<tr>
<td></td>
<td>■ italic: The font is italic.</td>
</tr>
<tr>
<td></td>
<td>■ oblique: The font is slanted to look italic, though the font family may not have a formal italic member.</td>
</tr>
<tr>
<td></td>
<td>■ inherit: Inherit font style from a parent component.</td>
</tr>
<tr>
<td>font-weight</td>
<td>The thickness of the component text. Choose from:</td>
</tr>
<tr>
<td></td>
<td>■ &lt;none&gt;: Specify no weight for the component text.</td>
</tr>
<tr>
<td></td>
<td>■ normal: Apply same weight as the parent component’s default font weight.</td>
</tr>
<tr>
<td></td>
<td>■ bold: Set component text in bold.</td>
</tr>
<tr>
<td></td>
<td>■ bolder: Set component text darker than bold.</td>
</tr>
<tr>
<td></td>
<td>■ light: Set component text lighter than normal.</td>
</tr>
<tr>
<td></td>
<td>■ lighter: Set component text lighter than light.</td>
</tr>
<tr>
<td></td>
<td>■ 100–900: Specify a value for text darkness—100 is lightest and 900 is darkest—400 is normal weight; 700 is bold.</td>
</tr>
<tr>
<td></td>
<td>■ inherit: Inherit font weight from a parent component.</td>
</tr>
<tr>
<td>height</td>
<td>The spacing to apply between lines of continuous text (also known as leading). Specify a height by choosing from:</td>
</tr>
<tr>
<td></td>
<td>■ Choose Length: Specify an absolute value in your preferred unit of measurement.</td>
</tr>
<tr>
<td></td>
<td>■ Choose Percentage: Specify a value as a percentage of the value set for a parent component.</td>
</tr>
<tr>
<td></td>
<td>■ auto: Set line height automatically, usually about 2 points larger than font size.</td>
</tr>
<tr>
<td></td>
<td>■ inherit: Inherit line height from the parent component.</td>
</tr>
<tr>
<td>list-style-image</td>
<td>An image to use as an indicator of a list item. Specify an image instead of defining a list-style-type. Choose from:</td>
</tr>
<tr>
<td></td>
<td>■ inherit: Inherit a list style image from a parent component.</td>
</tr>
<tr>
<td></td>
<td>■ none: Use no image as an indicator of a list item.</td>
</tr>
<tr>
<td></td>
<td>Or click the Edit icon and select an image.</td>
</tr>
</tbody>
</table>
### Table B–23 (Cont.) Style Attributes of Composer and WebCenter Portal Customizable Components

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>list-style-type</td>
<td>The type of indicator to use for items on a list. Use this instead of specifying an image through <code>list-style-image</code>. Select from among many styles offered on the list, including <code>&lt;none&gt;</code>, which means the browser's default style is used, and <code>inherit</code>, which means the <code>list-style-type</code> is inherited from a parent component.</td>
</tr>
</tbody>
</table>
| 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.                                                                                 |
**B.5 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.

---

**Table B–23 (Cont.) Style Attributes of Composer and WebCenter Portal Customizable Components**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>text-decoration</strong></td>
<td>The decorative value to apply to component text. Choose from:</td>
</tr>
<tr>
<td>&lt;none&gt;</td>
<td>Use the browser default for hyperlinks; otherwise, no text decoration.</td>
</tr>
<tr>
<td>none</td>
<td>No text decoration, including no underscore for hyperlinks.</td>
</tr>
<tr>
<td>underline</td>
<td>Draw a line under all component text.</td>
</tr>
<tr>
<td>overline</td>
<td>Draw a line over all component text.</td>
</tr>
<tr>
<td>line-through</td>
<td>Draw a line through the middle of all component text.</td>
</tr>
<tr>
<td>blink</td>
<td>Make component text blink.</td>
</tr>
<tr>
<td>inherit</td>
<td>Inherit text decoration from a parent component.</td>
</tr>
<tr>
<td><strong>vertical-align</strong></td>
<td>The vertical alignment of component text. Choose from:</td>
</tr>
<tr>
<td>Choose Length</td>
<td>Specify an absolute value in your preferred unit of measurement.</td>
</tr>
<tr>
<td>Choose Percentage</td>
<td>Specify a value as a percentage of the value set for a parent component.</td>
</tr>
<tr>
<td>inherit</td>
<td>Inherit the value from a parent component.</td>
</tr>
<tr>
<td>baseline</td>
<td>Set the text on the baseline of the text row.</td>
</tr>
<tr>
<td>middle</td>
<td>Set the text in the middle of the text row.</td>
</tr>
<tr>
<td>text-bottom</td>
<td>Set the text on the bottom of the text row, lower than the baseline.</td>
</tr>
<tr>
<td>text-top</td>
<td>Set the text at the top of the text row, above the baseline.</td>
</tr>
<tr>
<td>sub</td>
<td>Set the text as subscript.</td>
</tr>
<tr>
<td>super</td>
<td>Set the text as superscript.</td>
</tr>
<tr>
<td><strong>width</strong></td>
<td>The spacing between text. Specify a width by choosing from:</td>
</tr>
<tr>
<td>Choose Length</td>
<td>Specify an absolute value in your preferred unit of measurement.</td>
</tr>
<tr>
<td>Choose Percentage</td>
<td>Specify a value as a percentage of the value set for a parent component.</td>
</tr>
<tr>
<td>auto</td>
<td>Set line height automatically, usually about 2 points larger than font size.</td>
</tr>
<tr>
<td>inherit</td>
<td>Inherit line height from the parent component.</td>
</tr>
</tbody>
</table>
B.5.1 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–24 describes the attributes of a Panel Customizable (HTML) component.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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–24  (Cont.) Attributes of a Panel Customizable (HTML) Component

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>layout</td>
<td>List of values</td>
<td>Yes</td>
<td>Specifies how the children of the Panel Customizable (HTML) are laid out.</td>
</tr>
<tr>
<td></td>
<td>horizontal/vertical</td>
<td></td>
<td>The default value is vertical.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Depending on the value of the layout attribute, child components are laid out as follows:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>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.</td>
</tr>
</tbody>
</table>

#### Style Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>styleClass</td>
<td>String</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>inlineStyle</td>
<td>String</td>
<td>Yes</td>
<td>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.</td>
</tr>
</tbody>
</table>

#### Advanced Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>binding</td>
<td>String</td>
<td>Yes</td>
<td>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: bindings='#{yourManagedBean.Binding)'</td>
</tr>
</tbody>
</table>
Table B–24  (Cont.) Attributes of a Panel Customizable (HTML) Component

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customization</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Attributes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>customizationAllowed</td>
<td>Boolean</td>
<td></td>
<td>Specifies whether customizations are allowed on this component. Available values are true and false. The default value is true.</td>
</tr>
<tr>
<td>customizationAllowed</td>
<td>String</td>
<td></td>
<td>Specifies the roles for which customization is enabled.</td>
</tr>
<tr>
<td>Other Attribute</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>customizationId</td>
<td></td>
<td></td>
<td>This attribute has been deprecated. Use the id attribute.</td>
</tr>
</tbody>
</table>

B.5.2 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–25 describes the attributes of a Show Detail Frame (HTML) component.

Table B–25 Attributes of a Show Detail Frame (HTML) Component

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Common Attributes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>id</td>
<td>String</td>
<td>No</td>
<td>The identifier for the component. The identifier must follow a subset of the syntax allowed in HTML:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ Must not be a zero-length String.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ First character must be an ASCII letter (A-Za-z) or an underscore (‘_’).</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ Subsequent characters must be an ASCII letter or digit (A-Za-z0-9), an underscore (‘_’), or a dash (‘-’).</td>
</tr>
<tr>
<td>rendered</td>
<td>Boolean</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>text</td>
<td>String</td>
<td>Yes</td>
<td>A title for the Show Detail Frame (HTML) component.</td>
</tr>
</tbody>
</table>
### Table B–25  (Cont.) Attributes of a Show Detail Frame (HTML) Component

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>icon</td>
<td>String</td>
<td>Yes</td>
<td>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=&quot;<a href="http://source-pc/images/accessability.gif">http://source-pc/images/accessability.gif</a>&quot; Note: An image that is stored at the document root does not require a full path. For example: icon=&quot;detail.gif&quot;</td>
</tr>
<tr>
<td>text</td>
<td>String</td>
<td>Yes</td>
<td>A title for the Show Detail Frame (HTML) component.</td>
</tr>
<tr>
<td>shortDesc</td>
<td>String</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>icon</td>
<td>String</td>
<td>Yes</td>
<td>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=&quot;<a href="http://source-pc/images/accessability.gif">http://source-pc/images/accessability.gif</a>&quot; Note: An image that is stored at the document root does not require a full path. For example: icon=&quot;detail.gif&quot;</td>
</tr>
<tr>
<td>background</td>
<td>List of values</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td>light/medium/dark</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table B–25  (Cont.) Attributes of a Show Detail Frame (HTML) Component

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>displayHeader</td>
<td>Boolean</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>expansionMode</td>
<td>List of values</td>
<td>Yes</td>
<td>The default state of the Show Detail Frame (HTML). The default display mode is normal.</td>
</tr>
<tr>
<td>rendered</td>
<td>Boolean</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>Actions Attributes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>showMoveAction</td>
<td>List of values</td>
<td>Yes</td>
<td>Specifies whether the Move action is displayed in the Actions menu. The default value is menu.</td>
</tr>
<tr>
<td></td>
<td>menu/none</td>
<td></td>
<td></td>
</tr>
<tr>
<td>showMinimizeAction</td>
<td>List of values</td>
<td>Yes</td>
<td>Specifies whether the minimize action is displayed on the header. The default is chrome.</td>
</tr>
<tr>
<td></td>
<td>chrome/none</td>
<td></td>
<td></td>
</tr>
<tr>
<td>showMaximizeAction</td>
<td>List of values</td>
<td>Yes</td>
<td>Specifies whether the minimize action is displayed on the header. The default value is menu.</td>
</tr>
<tr>
<td></td>
<td>menu/none</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Style Attributes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>contentStyle</td>
<td>String</td>
<td>Yes</td>
<td>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.</td>
</tr>
</tbody>
</table>
### Customizable Components (HTML) Properties

#### Composer Component Properties and Files

**B-75**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>styleClass</strong></td>
<td>String</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td><strong>inlineStyle</strong></td>
<td>String</td>
<td>Yes</td>
<td>The CSS styles to use for this component. This is intended for basic style changes. The <code>inlineStyle</code> is a set of CSS styles that are applied to the root DOM element of the component. If the <code>inlineStyle</code>'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 <code>::label</code> or <code>::icon-style</code>.</td>
</tr>
</tbody>
</table>

**Behavior Attributes**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>partialTriggers</strong></td>
<td>String</td>
<td>Yes</td>
<td>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 - <code>::</code> pops out of the component's naming container (or itself if the component is a naming container) and begin the search from there, <code>:::</code> pops out of two naming containers (including itself if the component is a naming container) and begin the search from there.</td>
</tr>
<tr>
<td><strong>disclosureListener</strong></td>
<td>javax.el.MethodExpression</td>
<td>Yes</td>
<td>A method reference to a disclosure listener. A disclosure event is fired when the disclosure state changes.</td>
</tr>
</tbody>
</table>

**Advanced Attributes**
Customizable Components (HTML) Properties

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–26 describes the facets that enable you to display custom actions supported by the Show Detail Frame (HTML) component.

<table>
<thead>
<tr>
<th>Facet</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>titleBarAction</td>
<td>Used if an action is to be associated with title of the Show Detail Frame (HTML) component.</td>
</tr>
<tr>
<td>additionalActions</td>
<td>Used if some additional actions are to be added to the list of actions available on the Show Detail Frame (HTML) header.</td>
</tr>
</tbody>
</table>

**Table B–25 (Cont.) Attributes of a Show Detail Frame (HTML) Component**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>binding</td>
<td>String</td>
<td>Yes</td>
<td>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=&quot;#{yourManagedBean.Bin ding}&quot;</td>
</tr>
<tr>
<td>attributeChangeListe</td>
<td>javax.el.MethodExpression</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>customizationAllowed</td>
<td>Boolean</td>
<td></td>
<td>Specifies whether customizations are allowed on this component. Available values are true and false. The default value is true.</td>
</tr>
<tr>
<td>customizationAllowe</td>
<td>String</td>
<td></td>
<td>Specifies the roles for which customization is enabled.</td>
</tr>
<tr>
<td>customizationId</td>
<td></td>
<td></td>
<td>This attribute has been deprecated. Use the id attribute.</td>
</tr>
</tbody>
</table>

Table B–26 (Cont.) Attributes of a Show Detail Frame (HTML) Component

**Customization Attributes**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customizationAllowed</td>
<td>Boolean</td>
<td>Specifies whether customizations are allowed on this component. Available values are true and false. The default value is true.</td>
</tr>
<tr>
<td>customizationAllowe</td>
<td>String</td>
<td>Specifies the roles for which customization is enabled.</td>
</tr>
</tbody>
</table>

**Other Attribute**

customizationId
This attribute has been deprecated. Use the id attribute.
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.5.3 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.5.3.1 Panel Customizable (HTML) Style Selectors

Use the style selectors listed in Table B–27 to skin Panel Customizable (HTML) components. For icon selectors relevant to the Panel Customizable component, see Section B.5.3.4, "Icon Selectors."

<table>
<thead>
<tr>
<th>Style Selector</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>afh</td>
<td>panelCustomizable::no-header-content</td>
</tr>
</tbody>
</table>

#### B.5.3.2 Show Detail Frame (HTML) Style Selectors

Use the style selectors listed in Table B–28 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>
<thead>
<tr>
<th>Style Selector</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>afh</td>
<td>showDetailFrame::header-top-border-light</td>
</tr>
<tr>
<td>afh</td>
<td>showDetailFrame::header-top-border-medium</td>
</tr>
<tr>
<td>afh</td>
<td>showDetailFrame::header-top-border-dark</td>
</tr>
<tr>
<td>afh</td>
<td>showDetailFrame::container</td>
</tr>
<tr>
<td>afh</td>
<td>showDetailFrame::header-light</td>
</tr>
</tbody>
</table>
### B.5.3.3 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–29 lists and describes property keys relevant to Panel Customizable and Show Detail Frame components.
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.5.3.3, "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 Portal Framework application's root folder.

The selectors described in Table B–30 apply to the icons used with Panel Customizable (HTML) and Show Detail Frame (HTML) components.

### Table B–29  Property Keys of Panel Customizable and Show Detail Frame Components

<table>
<thead>
<tr>
<th>Property Key</th>
<th>Description</th>
</tr>
</thead>
</table>
| `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.5.3.4, "Icon Selectors." |

### B.5.3.4 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.5.3.3, "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 Portal Framework application's root folder.

The selectors described in Table B–30 apply to the icons used with Panel Customizable (HTML) and Show Detail Frame (HTML) components.
### Table B–30  Icon Selectors for WebCenter Portal Customizable Components

<table>
<thead>
<tr>
<th>Selector</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>showDetailFrame</strong></td>
<td>This icon represents the Actions menu. The Actions menu lists the actions a user can perform on the component. In a Portal Framework application, the Actions icon is rendered on the right corner of the component header.</td>
</tr>
<tr>
<td><code>::light-ActionsIcon:alias</code></td>
<td></td>
</tr>
<tr>
<td><code>::medium-ActionsIcon:alias</code></td>
<td></td>
</tr>
<tr>
<td><code>::dark-ActionsIcon:alias</code></td>
<td></td>
</tr>
<tr>
<td><strong>panelCustomizable</strong></td>
<td>This icon represents the Minimize option. Minimize collapses the view of the component like a window shade. In a Portal Framework application, the Minimize icon is rendered on the left side of the component header. See also, <code>::light-ExpandIcon:alias</code>.</td>
</tr>
<tr>
<td><code>::light-MinimizeIcon:alias</code></td>
<td></td>
</tr>
<tr>
<td><code>::medium-MinimizeIcon:alias</code></td>
<td></td>
</tr>
<tr>
<td><code>::dark-MinimizeIcon:alias</code></td>
<td></td>
</tr>
<tr>
<td><strong>showDetailFrame</strong></td>
<td>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 Portal Framework application, the Maximize icon is displayed to the left of the Maximize command on the component's Actions menu.</td>
</tr>
<tr>
<td><code>::light-MaximizeIcon:alias</code></td>
<td></td>
</tr>
<tr>
<td><code>::medium-MaximizeIcon:alias</code></td>
<td></td>
</tr>
<tr>
<td><code>::dark-MaximizeIcon:alias</code></td>
<td></td>
</tr>
<tr>
<td><strong>panelCustomizable</strong></td>
<td>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 Portal Framework application, the Maximize icon is displayed to the left of the Maximize command on the component's Actions menu.</td>
</tr>
<tr>
<td><code>::light-MaximizeIcon:alias</code></td>
<td></td>
</tr>
<tr>
<td><code>::medium-MaximizeIcon:alias</code></td>
<td></td>
</tr>
<tr>
<td><code>::dark-MaximizeIcon:alias</code></td>
<td></td>
</tr>
<tr>
<td><strong>showDetailFrame</strong></td>
<td>This icon represents the Restore option, which restores maximized views to their default display modes. In a Portal Framework application, the Restore icon is rendered to the left of the Restore command on the component's Actions menu.</td>
</tr>
<tr>
<td><code>::light-RestoreIcon:alias</code></td>
<td></td>
</tr>
<tr>
<td><code>::medium-RestoreIcon:alias</code></td>
<td></td>
</tr>
<tr>
<td><code>::dark-RestoreIcon:alias</code></td>
<td></td>
</tr>
<tr>
<td><strong>panelCustomizable</strong></td>
<td></td>
</tr>
<tr>
<td><code>::light-RestoreIcon:alias</code></td>
<td></td>
</tr>
<tr>
<td><code>::medium-RestoreIcon:alias</code></td>
<td></td>
</tr>
<tr>
<td><code>::dark-RestoreIcon:alias</code></td>
<td></td>
</tr>
</tbody>
</table>
### Table B–30 (Cont.) Icon Selectors for WebCenter Portal Customizable Components

<table>
<thead>
<tr>
<th>Selector</th>
<th>Description</th>
</tr>
</thead>
</table>
| **showDetailFrame** | showDetailFrame::light-ExpandIcon:alias  
|               | showDetailFrame::medium-ExpandIcon:alias  
|               | showDetailFrame::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 Portal Framework application, the Expand icon is displayed on the left side of the component header.  |
| **panelCustomizable** | panelCustomizable::light-ExpandIcon:alias  
|               | panelCustomizable::medium-ExpandIcon:alias  
|               | panelCustomizable::dark-ExpandIcon: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 Portal Framework application, the Move icon is displayed to the left of the Move command on the component's Actions menu.  |
| **showDetailFrame** | showDetailFrame::light-MoveIcon:alias  
|               | showDetailFrame::medium-MoveIcon:alias  
|               | showDetailFrame::dark-MoveIcon: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 Portal Framework application, the Move Left icon is displayed to the left of the Move Left submenu item on the component's Actions menu.  |
| **panelCustomizable** | panelCustomizable::light-MoveLeftIcon:alias  
|               | panelCustomizable::medium-MoveLeftIcon:alias  
|               | panelCustomizable::dark-MoveLeftIcon: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 Portal 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  
|               | 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 Portal 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–30 (Cont.) Icon Selectors for WebCenter Portal Customizable Components

<table>
<thead>
<tr>
<th>Selector</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>showDetailFrame</code></td>
<td>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.</td>
</tr>
<tr>
<td><code>panelCustomizable</code></td>
<td>In a Portal Framework application, the Move Down icon is displayed to the left of the Move Down submenu item on the component’s Actions menu.</td>
</tr>
<tr>
<td><code>showDetailFrame</code></td>
<td>This icon provides an image for the top-left corner of the component header.</td>
</tr>
<tr>
<td><code>panelCustomizable</code></td>
<td>This icon provides the image for the top-right corner of the component header.</td>
</tr>
<tr>
<td><code>showDetailFrame</code></td>
<td>This icon provides the left portion of the component’s FadeIn-FadeOut toolbar.</td>
</tr>
<tr>
<td><code>panelCustomizable</code></td>
<td>The FadeIn-FadeOut toolbar comes into play when the <code>adfp:portlet</code> tag attribute <code>isSeededInteractionAvailable</code> is set to true and <code>displayHeader</code> is set to false.</td>
</tr>
<tr>
<td><code>showDetailFrame</code></td>
<td>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.</td>
</tr>
<tr>
<td><code>panelCustomizable</code></td>
<td>If the page design is very simple, then the FadeIn-FadeOut toolbar may not display, even when <code>displayHeader</code> is set to false and <code>isSeededInteractionAvailable</code> is set to true.</td>
</tr>
<tr>
<td><code>showDetailFrame</code></td>
<td>This icon provides the right portion of the component’s FadeIn-FadeOut toolbar.</td>
</tr>
<tr>
<td><code>panelCustomizable</code></td>
<td>See the description for <code>showDetailFrame::light-ToolbarLeftIcon:alias</code> for an explanation of the FadeIn-FadeOut toolbar.</td>
</tr>
</tbody>
</table>
### Table B–30 (Cont.) Icon Selectors for WebCenter Portal Customizable Components

<table>
<thead>
<tr>
<th>Selector</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>showDetailFrame</td>
<td>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.</td>
</tr>
<tr>
<td>showDetailFrame::light-ToolbarCenterIcon:alias</td>
<td></td>
</tr>
<tr>
<td>showDetailFrame::medium-ToolbarCenterIcon:alias</td>
<td></td>
</tr>
<tr>
<td>showDetailFrame::dark-ToolbarCenterIcon:alias</td>
<td></td>
</tr>
<tr>
<td>panelCustomizable</td>
<td>This icon represents the Edit option on the component menu. In a Portal Framework application, the Edit icon is displayed to the left of the Edit menu item on the component’s Actions menu.</td>
</tr>
<tr>
<td>panelCustomizable::light-ToolbarCenterIcon:alias</td>
<td></td>
</tr>
<tr>
<td>panelCustomizable::medium-ToolbarCenterIcon:alias</td>
<td></td>
</tr>
<tr>
<td>panelCustomizable::dark-ToolbarCenterIcon:alias</td>
<td></td>
</tr>
<tr>
<td>showDetailFrame</td>
<td>This icon represents the Help option on the component menu. In a Portal Framework application, the Help icon is displayed to the left of the Help menu item on the component’s Actions menu.</td>
</tr>
<tr>
<td>showDetailFrame::light-HelperIcon:alias</td>
<td></td>
</tr>
<tr>
<td>showDetailFrame::medium-HelperIcon:alias</td>
<td></td>
</tr>
<tr>
<td>showDetailFrame::dark-HelperIcon:alias</td>
<td></td>
</tr>
<tr>
<td>panelCustomizable</td>
<td></td>
</tr>
<tr>
<td>panelCustomizable::light-HelperIcon:alias</td>
<td></td>
</tr>
<tr>
<td>panelCustomizable::medium-HelperIcon:alias</td>
<td></td>
</tr>
<tr>
<td>panelCustomizable::dark-HelperIcon:alias</td>
<td></td>
</tr>
</tbody>
</table>
This appendix describes the configuration, location, and attributes of the default catalog definition file.

This appendix includes the following topics:

- 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"

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 14, "Developing Resource Catalogs."

### 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.

```xml
<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:

```xml
<rsc:adf-rcs-config>
  <rsc:rcs-config>
    <rsc:catalog-config default-scope="/" default-registry="/oracle/webcenter/portalapp/catalogs/catalog-registry.xml"/>
  </rsc:rcs-config>
</rsc:adf-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

```xml
<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>
```

C-2 Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper
<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>
      </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>
      </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: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:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:sequence>
    <xsd:element ref="rcs:source" minOccurs="1" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="expression" type="xsd:string"/>
</xsd:extension>
</xsd:complexType>
</xsd:complexContent>
</xsd:complexType>
</xsd:complexType>
</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>
<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:complexType>
</xsd:group>
<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:choice>
<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 name="includeNavigation" type="rcs:includeNavigationType"/>
        <xsd:element name="separator" type="rcs:separatorType"/>
      </xsd:all>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
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 includes the following topics:

- 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 14.2, "Creating a 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

```xml
<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, such as title, subject, and description for elements, see Section C.3.8, "attributes."

Table C–1  catalogDefinition Keywords

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xmlns</td>
<td><a href="http://xmlns.oracle.com/adf/rcs/catalog">http://xmlns.oracle.com/adf/rcs/catalog</a></td>
</tr>
<tr>
<td>id</td>
<td>The identifier used when referring to this catalog definition.</td>
</tr>
<tr>
<td>visible</td>
<td>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.</td>
</tr>
<tr>
<td>resourceBundle</td>
<td>The name of the resource bundle to be used for obtaining translated attribute values, where &lt;attribute... isKey=&quot;true&quot;/&gt;. 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.</td>
</tr>
<tr>
<td>definitionFilter</td>
<td>The fully qualified name of a Java class that implements the oracle.adf.rc.spi.plugin.catalog.CatalogDefinitionFilte r. 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 uses a definition filter to hide entries for services that the administrator has disabled.</td>
</tr>
</tbody>
</table>
```

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

```xml
<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>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The unique identifier used when referring to this folder.</td>
</tr>
<tr>
<td>visible</td>
<td>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.</td>
</tr>
</tbody>
</table>

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 14.3.4, "How to Add a Folder to a Resource Catalog."

### 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 tools and services task flow or a custom task flow created in 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**

```xml
<url visible="#{true}'

factoryClass="oracle.webcenter.portalframework.sitestructure.rc.TaskFlowResourceFactory"
    id="tagCloud"

url="taskflow://oracle/webcenter/tagging/controller/taskflows/tag-selection.xml#tag-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>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The unique identifier used when referring to this link.</td>
</tr>
<tr>
<td>visible</td>
<td>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.</td>
</tr>
</tbody>
</table>
| 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:  
  - 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 14-1. |

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 14.3.3, "How to Add a Link to a Resource Catalog."

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

```xml
<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 14.3.8, "How to Add a Resource Catalog to Another 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` elements 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**

```xml
<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>
</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**

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The unique identifier for the component in the catalog definition.</td>
</tr>
<tr>
<td>visible</td>
<td>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 <code>#{true}</code>, which means that the resource is displayed in the catalog at all times.</td>
</tr>
</tbody>
</table>
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 14.3.5, "How to Add a Custom Component to a Resource Catalog."

### 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**

```xml
<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']}'/>
    </attributes>
    <parameters/>
</customFolder>
```

**Table C–4 (Cont.) component Keywords**

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>factoryClass</code></td>
<td>The name of a Java class that implements the <code>oracle.adf.rc.component.ComponentFactory</code> 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, <code>XmlComponentFactory</code> requires a single parameter named <code>xml</code> and the value of the parameter is a block of XML representing a component or components that can be inserted into a page.</td>
</tr>
</tbody>
</table>

**Table C–5** describes the keywords for the customFolder element.
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 14.3.6, "How to Add a Custom Folder to a Resource Catalog."

### 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**

```xml
<customContent
  contentProviderClass="com.oracle.ensemble.interop.adf.EnsembleContentProvider"
  id="ensembleContentProvider" visible='true'>
  <attributes
    resourceBundle="oracle.webcenter.spaces.resource.DefaultGroupSpaceCatalogBundle"/>
</customContent>
```

Table C–6 describes the keywords for the **customContent** element.

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The unique identifier used when referring to this folder.</td>
</tr>
<tr>
<td>visible</td>
<td>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 <code>#{true}</code>, which means that the resource is displayed in the catalog at all times.</td>
</tr>
<tr>
<td>contentProviderClass</td>
<td>The fully qualified name of a class that implements the interface <code>oracle.adf.rc.spi.plugin.catalog.CustomContentProviderV2</code>.</td>
</tr>
</tbody>
</table>

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 14.3.7, "How to Add a Custom Content Provider to a Resource Catalog."

**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>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>resourceBundle</td>
<td>The name of the resource bundle to be used for obtaining translated attribute values.</td>
</tr>
</tbody>
</table>

**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**

```xml
<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**

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributeId</td>
<td>The identifier of the attribute as declared in the schema element.</td>
</tr>
<tr>
<td>value</td>
<td>The value of the attribute.</td>
</tr>
<tr>
<td>isKey</td>
<td>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.</td>
</tr>
<tr>
<td>resourceBundle</td>
<td>The name of the resource bundle to be used for obtaining translated attribute values.</td>
</tr>
</tbody>
</table>

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:

```xml
<attributes>
  <attribute value="dark" attributeId="attr.background" isKey="false"/>
  <attribute value="#{myBean.myParam1}" attributeId="parameter.myParam1" isKey="false"/>
</attributes>
```

For more information, see Section 14.7, "Defining 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

```xml
<parameters>
  <parameter id="xml">&lt;cust:panelCustomizable id="#" xmlns:cust="http://xmlns.oracle.com/adf/faces/customizable"/>
</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

```xml
<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"
```
Table C–9 describes the keywords of the descriptor element.

Table C–9  descriptor Keywords

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributeId</td>
<td>The identifier of the attribute being described.</td>
</tr>
<tr>
<td>searchable</td>
<td>Indicates whether the attribute is included in resource catalog searches (true) or not (false).</td>
</tr>
<tr>
<td>multivalue</td>
<td>Indicates whether the attribute can have multiple values (true) or not (false).</td>
</tr>
<tr>
<td>endUserVisible</td>
<td>Indicates whether the attribute should be exposed to end users (true) or not (false).</td>
</tr>
<tr>
<td>resourceBundle</td>
<td>The name of the resource bundle used to translate the label keys.</td>
</tr>
<tr>
<td>labelKey</td>
<td>The resource bundle key for the attribute's label.</td>
</tr>
<tr>
<td>shortLabelKey</td>
<td>The resource bundle key for the attribute's short label.</td>
</tr>
</tbody>
</table>

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

```xml
<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" isKeys="false" attributeId="Description"/>
        <attribute value="Weather widget" isKeys="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."
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 topics:

- **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:

```xml
<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_`
```

### Table C–10  resource Keywords

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The identifier for the element in the catalog. Identifiers are required and must be unique at the folder level.</td>
</tr>
<tr>
<td>repository</td>
<td>The path to access ADF libraries included in your classpath.</td>
</tr>
<tr>
<td>path</td>
<td>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 &quot;/&quot; with &quot;+&quot;. 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</td>
</tr>
</tbody>
</table>

The identifier for the element in the catalog. Identifiers are required and must be unique at the folder level.

The path to access ADF libraries included in your classpath.

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

Resource Catalog Properties and Files  C-21
Factory Classes Available for Adding Dynamic Resources to the Catalog

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:

```xml
<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 14.3.3, "How to Add a Link to a Resource Catalog."

C.4.3 CustomFolderContextFactory

The CustomFolderContextFactory 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:

```xml
<customFolder id="AllConnections"
    factoryClass="oracle.webcenter.content.model.rc.CustomFolderContextFactory" />
```

For more information about custom folders, see Section 14.3.6, "How to Add a Custom Folder to a Resource Catalog."

C.4.4 DefaultFolderContextFactory

In the WebCenter Portal context, the DefaultFolderContextFactory class is used to render the folder for the current portal. If the user is in the Home portal, 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:

```xml
<customFolder id="GroupDocuments"
    factoryClass="oracle.webcenter.content.model.rc.DefaultFolderContextFactory" />
```
For more information about custom folders, see Section 14.3.6, "How to Add a Custom Folder to a Resource Catalog."

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:

```xml
<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 29, "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:

```xml
<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 14.3.7, "How to Add a Custom Content Provider to a Resource Catalog."
This appendix describes the guidelines for creating and consuming task flows effectively in your applications by implementing task flow parameters and customization and personalization options.

This appendix includes 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 Portal Framework Application Containing a Task Flow Created By Following the Guidelines"
- Section D.4, "Conclusion"

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 Portal 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

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 Portal Framework applications. Also, be sure to adhere to these guidelines when you edit existing Portal 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**

```html
<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 Portal 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 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 Portal 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.

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.

```xml
<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'
```
Guidelines for Creating Task Flows to Be Used in Composer-Enabled Pages

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](image)

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 WebCenter Portal 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

Both task flows have N=5, meaning at most 5 records are fetched and shown at once.

Watched Topics would have pagination because you want to be able to access every one of your watched topics, and the order may be by name.

Popular Topics, on the other hand, would not necessarily be paginated, because the data set is essentially "all topics". A task flow that shows the N most popular topics, without the ability to see the N+1st most popular, is still a useful task flow.

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 Portal 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>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
<th>When to Use</th>
<th>Who can See It</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameters</td>
<td>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).</td>
<td>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 WebCenter Portal, you can use a parameter for a context like portal name. Precedence: 3 This value overrides built-in values in the task flow.</td>
<td>Users with Edit or Customize permission can see parameters in Composer’s Component Properties dialog while editing a page. In WebCenter Portal, moderators can see and edit task flow parameters. When a task flow is exposed as a portlet, page designers can see these parameters while setting up the portlet.</td>
</tr>
</tbody>
</table>
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

---

**Table D–1 (Cont.) Options Used to Define Task Flow Appearance and Behavior**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
<th>When to Use</th>
<th>Who can See It</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customization/Preferences</td>
<td>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.</td>
<td>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, portal name), customization overrides that value.</td>
<td>Users with 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 can also see these options on the chrome in View mode of the page. When a task flow is exposed as a portlet, the page designer sees these as Edit Defaults options while setting up the portlet.</td>
</tr>
<tr>
<td>Personalization</td>
<td>A subset of the customization and preferences options. You cannot have personalization options that are not also customization options.</td>
<td>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.</td>
<td>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, the Personalization options allow end users to tweak the same visual aspects.</td>
</tr>
<tr>
<td>Implicit Personalization</td>
<td>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.</td>
<td>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.</td>
<td>All end users with View permission on the page can perform personalizations.</td>
</tr>
</tbody>
</table>
Guidelines for Efficient Use of Task Flow Parameters and Customization and Personalization Options

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:

```xml
<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:

```xml
<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:

```xml
<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 Portal Framework Application"
**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:

```xml
<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:

```xml
<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 Section D.3, "Example of a Portal 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 Portal Framework Application Containing a Task Flow Created By Following the Guidelines

As a companion to this document, a sample Portal Framework application was developed. 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:

- Fusion Developer’s Guide for Oracle Application Development Framework, for information about creating task flows.
- Building Portals with Oracle WebCenter Portal, 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. Click the link named "Sample Application with Optimal Task Flows" under Oracle Composer Samples.

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.

Table D–2 describes significant sections of code in the recent-pages-task-flow-definition.xml file.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;default-activity&gt;mainView&lt;/default-activity&gt;</td>
<td>Defining the default view for the task flow: mainView</td>
</tr>
<tr>
<td>&lt;managed-bean&gt;</td>
<td>Referencing the recentPagesBean Java bean that is associated with the task flow definition. This bean was created as part of the view package.</td>
</tr>
<tr>
<td>&lt;managed-bean-name&gt;recentPagesBean&lt;/managed-bean-name&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;managed-bean-class&gt;view.RecentPagesBean&lt;/managed-bean-class&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;managed-bean-scope&gt;pageFlow&lt;/managed-bean-scope&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;/managed-bean&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;view id=&quot;mainView&quot;&gt;</td>
<td>Mapping task flow views to pages or page fragments. For example, mainView is associated with mainView.jsff.</td>
</tr>
<tr>
<td>&lt;page&gt;/mainView.jsff&lt;/page&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;/view&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;view id=&quot;simpleEditPopupView&quot;&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;page&gt;/simpleEditPopupView.jspx&lt;/page&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;/view&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;view id=&quot;advancedEditPopupView&quot;&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;page&gt;/advancedEditPopupView.jspx&lt;/page&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;/view&gt;</td>
<td></td>
</tr>
</tbody>
</table>
Table D–2 (Cont.) Sections Of Code in the Task Flow Definition File

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;control-flow-rule&gt;</code></td>
<td>Control flow rules from mainView. Each control flow case defines an outcome and a corresponding target view.</td>
</tr>
<tr>
<td><code>&lt;from-activity-id&gt;mainView&lt;/from-activity-id&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;control-flow-case&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;from-outcome&gt;refresh&lt;/from-outcome&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;to-activity-id&gt;mainView&lt;/to-activity-id&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;control-flow-case&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;from-outcome&gt;dialog:simpleEditPopup&lt;/from-outcome&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;to-activity-id&gt;simpleEditPopupView&lt;/to-activity-id&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;control-flow-case&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;from-outcome&gt;next&lt;/from-outcome&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;to-activity-id&gt;mainView&lt;/to-activity-id&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;control-flow-case&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;from-outcome&gt;previous&lt;/from-outcome&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;to-activity-id&gt;mainView&lt;/to-activity-id&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;control-flow-case&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;from-outcome&gt;dialog:advancedEditPopup&lt;/from-outcome&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;to-activity-id&gt;advancedEditPopupView&lt;/to-activity-id&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;control-flow-case&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;from-outcome&gt;save&lt;/from-outcome&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;to-activity-id&gt;mainView&lt;/to-activity-id&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;control-flow-case&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;from-outcome&gt;cancel&lt;/from-outcome&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;to-activity-id&gt;mainView&lt;/to-activity-id&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;control-flow-case&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;from-outcome&gt;save&lt;/from-outcome&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;to-activity-id&gt;mainView&lt;/to-activity-id&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;control-flow-case&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;from-outcome&gt;cancel&lt;/from-outcome&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;to-activity-id&gt;mainView&lt;/to-activity-id&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;control-flow-rule&gt;</code></td>
<td></td>
</tr>
</tbody>
</table>

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:
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

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;af:panelGroupLayout id=&quot;pgl1&quot; layout=&quot;vertical&quot;</code></td>
<td>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.</td>
</tr>
<tr>
<td><code>      &lt;af:outputText value=&quot;Shows recent pages in the application&quot; id='ot1'/&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;af:table value=&quot;#{pageFlowScope.recentPagesBean.pages}&quot; var=&quot;row&quot;</code></td>
<td>A table component that is a child of the Panel Group Layout.</td>
</tr>
<tr>
<td><code>      rowBandInterval=&quot;0&quot; id='t1' rowSelection=&quot;none&quot;</code></td>
<td>Table attributes:</td>
</tr>
<tr>
<td><code>      columnBandInterval=&quot;0&quot; width=&quot;100%&quot;</code></td>
<td><code>autoHeightRows=&quot;#{pageFlowScope.recentPagesBean.number_of_items_to_show_at_a_time}&quot;</code></td>
</tr>
<tr>
<td><code>      inlineStyle=&quot;border:none;width:100%;&quot;</code></td>
<td></td>
</tr>
<tr>
<td><code>      columnStretching=&quot;last&quot;</code></td>
<td></td>
</tr>
<tr>
<td><code>      horizontalGridVisible=&quot;true&quot;</code></td>
<td></td>
</tr>
<tr>
<td><code>      verticalGridVisible=&quot;false&quot;</code></td>
<td></td>
</tr>
<tr>
<td><code>      autoHeightRows=&quot;#{pageFlowScope.recentPagesBean.number_of_items_to_show_at_a_time}&quot;</code></td>
<td></td>
</tr>
<tr>
<td><code>      contentDelivery='immediate'</code></td>
<td></td>
</tr>
<tr>
<td><code>      fetchSize=&quot;#{pageFlowScope.recentPagesBean.number_of_items_to_query}&quot;</code></td>
<td></td>
</tr>
</tbody>
</table>
Example of a Portal Framework Application Containing a Task Flow Created By Following the Guidelines

fetchSize (0): 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 $N$ is greater than $M$, pagination controls appear on the task flow to enable users to view the remaining items.

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>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;af:column sortable=&quot;false&quot; headerText='#{null}'' align='start' id='c1'&gt;</code></td>
<td>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.</td>
</tr>
<tr>
<td><code>&lt;af:panelGroupLayout id='pgl11' styleClass='nowrap' inlineStyle='padding-top:2px; font-weight:bold;'&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;af:outputText value='#{row.title}' id='ot4'/&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;af:goLink destination='''/faces#{row.contentMRef}' text=''' targetFrame='_blank' id='gl1'/&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;af:panelGroupLayout id='pgl10' styleClass='AFFieldTextDisabled nowrap' inlineStyle='padding-top:2px'&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;af:outputText value='Created on ' id='ot7'/&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;af:outputText value='#{row.createDate}' id='ot8'/&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;af:outputText value=' by #{row.createdBy}' id='ot9'/&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;/af:panelGroupLayout&gt;</code></td>
<td></td>
</tr>
</tbody>
</table>

Table D–3 (Cont.) The mainView Fragment Code
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

Table D–4 describes significant sections of code in the simpleEditPopupView.jspx page.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;af:panelStretchLayout id=&quot;ps11&quot;&gt;</code></td>
<td>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.</td>
</tr>
<tr>
<td><code>&lt;f:facet name=&quot;center&quot;&gt;</code></td>
<td>The Panel Form Layout contains an Input Text component, which is populated with a value returned by recentPagesBean.</td>
</tr>
<tr>
<td><code>  &lt;af:group id=&quot;group1&quot;&gt;</code></td>
<td>The Panel Group Layout contains two Command Button components called Save and Close, which are wired to events defined in recentPagesBean.</td>
</tr>
<tr>
<td><code>    &lt;af:panelFormLayout id=&quot;pfl1&quot;&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>      &lt;af:inputText label=&quot;Number of Items to show at a time&quot;</code></td>
<td></td>
</tr>
<tr>
<td><code>        id=&quot;it1&quot;</code></td>
<td></td>
</tr>
<tr>
<td><code>          value=&quot;#{pageFlowScope.recentPagesBean.number_of_items_to_show_at_a_time}&quot;</code></td>
<td></td>
</tr>
<tr>
<td><code>          columns=&quot;3&quot;/&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>    &lt;/af:panelFormLayout&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>    &lt;af:panelGroupLayout id=&quot;pgl2&quot;</code></td>
<td></td>
</tr>
<tr>
<td><code>      layout='horizontal'</code></td>
<td></td>
</tr>
<tr>
<td><code>      halign='right'</code></td>
<td></td>
</tr>
<tr>
<td><code>      inlineStyle='width:500px;'</code></td>
<td></td>
</tr>
<tr>
<td><code>      &lt;af:commandButton text='Save' id='cb1'</code></td>
<td></td>
</tr>
<tr>
<td><code>        action='#{pageFlowScope.recentPagesBean.saveSettings}'</code></td>
<td></td>
</tr>
<tr>
<td><code>        actionListener='#{pageFlowScope.recentPagesBean.cancel}'</code></td>
<td></td>
</tr>
<tr>
<td><code>    &lt;/af:commandButton&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>    &lt;/af:panelGroupLayout&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;/af:group&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;/f:facet&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;/af:panelStretchLayout&gt;</code></td>
<td></td>
</tr>
</tbody>
</table>

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.
**Figure D–7** The `advancedEditPopup.jspx` Page

Table D–5 describes significant sections of code in the `advancedPopupView.jspx` page.
Example of a Portal Framework Application Containing a Task Flow Created By Following the Guidelines

Table D–5  Code in the Customization Page

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;af:panelStretchLayout id=&quot;psl1&quot;&gt;</code></td>
<td>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.</td>
</tr>
<tr>
<td><code>&lt;f:facet name=&quot;center&quot;&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;af:group id=&quot;group1&quot;&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;af:panelFormLayout id=&quot;pfl1&quot;&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;af:inputText label=&quot;Number of Items to show at a time&quot;</code></td>
<td></td>
</tr>
<tr>
<td>id=&quot;it1&quot;</td>
<td></td>
</tr>
<tr>
<td>value=&quot;#{pageFlowScope.recentPagesBean.number_of_items_to_show_at_a_time}&quot;</td>
<td></td>
</tr>
<tr>
<td>columns=&quot;3&quot;</td>
<td></td>
</tr>
<tr>
<td><code>&lt;af:inputText label=&quot;Number of Items to Query&quot;</code></td>
<td></td>
</tr>
<tr>
<td>id=&quot;it2&quot;</td>
<td></td>
</tr>
<tr>
<td>value=&quot;#{pageFlowScope.recentPagesBean.number_of_items_to_query}&quot;</td>
<td></td>
</tr>
<tr>
<td>columns=&quot;3&quot;</td>
<td></td>
</tr>
<tr>
<td><code>&lt;af:selectBooleanCheckbox label=&quot;Show Page Control&quot;</code></td>
<td></td>
</tr>
<tr>
<td>id=&quot;sbc1&quot;</td>
<td></td>
</tr>
<tr>
<td>value=&quot;#{pageFlowScope.recentPagesBean.showPageControl}&quot;</td>
<td></td>
</tr>
<tr>
<td><code>&lt;af:panelFormLayout&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;af:panelGroupLayout id=&quot;pgl2&quot; layout=&quot;horizontal&quot;</code></td>
<td></td>
</tr>
<tr>
<td>halign=&quot;right&quot;</td>
<td></td>
</tr>
<tr>
<td>inlineStyle=&quot;width:500px;&quot;</td>
<td></td>
</tr>
<tr>
<td><code>&lt;af:commandButton text=&quot;Save&quot; id=&quot;cb1&quot; action=&quot;#{pageFlowScope.recentPagesBean.saveSettings}&quot;</code></td>
<td></td>
</tr>
<tr>
<td>actionListener=&quot;#{pageFlowScope.recentPagesBean.cancel}&quot;</td>
<td></td>
</tr>
</tbody>
</table>

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.

Table D–6 describes significant sections of code in the recentPagesBean.java file.
### Table D–6  The recentPagesBean Java Bean Code

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
</table>
| private String _getPreferenceFileName()  
  { 
    String taskflowId =  
    {  
      ControllerContextFwk.getInstanceFwk().getCurrentViewPortFwk().getClientId();  
      if (taskflowId == null)  
      {  
        taskflowId =  
      }  
  
  ControllerContextFwk.getInstanceFwk().getCurrentViewPortFwk().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(beanName);  
  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. |
| Logic for identifying the task flow and retrieving the instance-level preference file for the currently running task flow. |
| Logic for identifying the task flow and retrieving the instance-level preference file for the currently running task flow. |
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().returnFromDialog(null, null);
    return "save";
}

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;
}

Table D–6 (Cont.) The recentPagesBean Java Bean Code

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>public String saveSettings() throws Exception</td>
<td>Logic to define behavior of Save button in the personalization or customization dialog. Settings are saved in the task flow preference file.</td>
</tr>
<tr>
<td>public void setNumber_of_items_to_show_at_a_time(String number_of</td>
<td>Logic to populate the Number of Items to show at a time field in the personalization or customization dialog.</td>
</tr>
<tr>
<td>items_to_show_at_a_time)</td>
<td></td>
</tr>
<tr>
<td>public String getNumber_of_items_to_show_at_a_time()</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
</tbody>
</table>
Table D-6 (Cont.) The recentPagesBean Java Bean Code

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>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; }</td>
<td>Logic to populate the Number of Items to Query field in the customization dialog.</td>
</tr>
<tr>
<td>public boolean getShowPageControl() { boolean showPageControl = Preference.getShowPageControl(_getPreferenceFileName()); return showPageControl; } public void setShowPageControl(boolean showPageControl) { _showPageControl = showPageControl; }</td>
<td>Logic to set the Show Page Control check box in the customization dialog.</td>
</tr>
<tr>
<td>public List&lt;PageDef&gt; getPages() { PageServiceBean psb = getPageServiceBean(); List&lt;PageDef&gt; pages = psb.getPages(); int start = getStart(); int end = getEnd(); if (start &lt; 0</td>
<td></td>
</tr>
</tbody>
</table>
Example of a Portal Framework Application Containing a Task Flow Created By Following the Guidelines

Table D–6 (Cont.) The recentPagesBean Java Bean Code

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>public int getTableSize()</td>
<td>Logic to set the table size to the number of records retrieved.</td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>int tableSize = -1;</td>
<td></td>
</tr>
<tr>
<td>try</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>PageServiceBean psb = getPageServiceBean();</td>
<td></td>
</tr>
<tr>
<td>List&lt;PageDef&gt; pages = psb.getPages();</td>
<td></td>
</tr>
<tr>
<td>tableSize = pages.size();</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>catch (Exception e)</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>System.out.println(e.toString());</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>return tableSize;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>public void cancel(ActionEvent actionEvent)</td>
<td>Logic to define behavior of Cancel button in the personalization or customiz</td>
</tr>
<tr>
<td>{</td>
<td>tion dialog.</td>
</tr>
<tr>
<td>AdfFacesContext.getCurrentInstance().returnFromDialog(null, null);</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>public void previous(ActionEvent actionEvent)</td>
<td>Logic to define behavior of Previous action in the task flow’s pagination</td>
</tr>
<tr>
<td>{</td>
<td>control.</td>
</tr>
<tr>
<td>int page_num = getPageNumber();</td>
<td></td>
</tr>
<tr>
<td>page_num--;</td>
<td></td>
</tr>
<tr>
<td>if (page_num &lt;= 0)</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>page_num = 1;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>RequestContext.getCurrentInstance().getPageFlowScope().put(&quot;page_num&quot;, page_num);</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>public void next(ActionEvent actionEvent)</td>
<td>Logic to define behavior of Next action in the task flow’s pagination con</td>
</tr>
<tr>
<td>{</td>
<td>trol.</td>
</tr>
<tr>
<td>int page_num = getPageNumber();</td>
<td></td>
</tr>
<tr>
<td>page_num++;</td>
<td></td>
</tr>
<tr>
<td>String items_to_query =</td>
<td></td>
</tr>
<tr>
<td>this.getNumber_of_items_to_query();</td>
<td></td>
</tr>
<tr>
<td>int offset = Integer.parseInt(items_to_query);</td>
<td></td>
</tr>
<tr>
<td>int total_page_num = getTotalPageNumber();</td>
<td></td>
</tr>
<tr>
<td>if (page_num &gt; total_page_num)</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>page_num = this.getTableSize()</td>
<td></td>
</tr>
<tr>
<td>/ offset;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>RequestContext.getCurrentInstance().getPageFlowScope().put(&quot;page_num&quot;, page_num);</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
</tbody>
</table>
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, portal-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 sample application.
Table D-7 describes significant sections of code in the `Preference.java` file.

### Table D-7  The Preference Java Bean Code

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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. |
| 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);  
  } | Logic to read the preference file in the MDS session or create one, if required. |
### Table D–7 (Cont.) The Preference Java Bean Code

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
</table>
| ```java
// 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();
    }
    catch (Exception e)
    {
        throw new RuntimeException(e);
    }
}
return mo;
``` | Logic to read the preference file in the MDS session or create one, if required. |
| ```java
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);
}
``` |
| ```java
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().getElement().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().getElement().getAttribute("numberOfItemsToQuery");
    return Integer.parseInt(query);
}

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logic to get and set the number of items to query from the data source.</td>
<td></td>
</tr>
</tbody>
</table>
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
<?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>
                    </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.recentpages-taskflowdefinition1.regionModel}" id="r2"/>
                                </cust:showDetailFrame>
                            </cust:panelCustomizable>
                        </pe:pageCustomizable>
                    </f:facet>
                    <f:facet name="end"/>
                </af:panelStretchLayout>
            </af:form>
        </af:document>
    </f:view>
```
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:

```xml
<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 Portal 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 Portal Framework application users, follow the guidelines in this document while implementing task flows to be consumed in the application.
Conclusion
This appendix describes how to get optimal performance when using JavaScript with WebCenter Portal pages.

This appendix contains the following topics:

- Section E.1, "Using JavaScript Partitioning"
- Section E.2, "Using Partitioning for Performance"

### E.1 Using JavaScript Partitioning

ADF UI components are complex entities, each containing one or more HTML, CSS, and JavaScript (JS) elements. When a component is rendered, the HTML tags are stamped on the page with right CSS style classes attached. ADF is smart enough to download only the JS files belonging to the components rendered on the page. If individual JS files are downloaded, however, it will cause a lot of network traffic per page, slowing down the page performance. The solution is to use JavaScript Partitions.

ADF’s JavaScript Partitioning feature helps combine individual JS files from ADF components together into buckets, thereby reducing the number of downloads. By default, all ADF pages need the `boot.js` and `core.js` files to operate. Additionally, JS files are downloaded either combined as partitions or one for each component (if no partition is defined). For more information about JavaScript partitioning, see "Using JavaScript Library Partitioning" in *Web User Interface Developer’s Guide for Oracle Application Development Framework*.

### E.2 Using Partitioning for Performance

The general rules to get optimal JS performance are:

- Keep the number of static file downloads less to avoid network latency
- Keep individual file size to be around the same else the largest file will delay the page load.
- Construct the home page carefully so that subsequent browser pages need fewer file downloads

Keeping these general rules in mind, follow these steps to tune the JS partitions for optimal performance:

2. Remove all the features from the core partition that start with Adf*.

Note: Remember to redeploy the application and clear the browser cache every time you test the adf-js-partitions.xml file.

3. Test the home page and note the JS files downloaded. Figure E–1 shows the results using Firebug.

Figure E–1 Initial results for downloaded files for the home page

4. To create partitions, examine the individual JS files downloaded by your page. The files starting with Adf don’t have partitions defined, while files such as dnd-11.1.1.7 are partitions. Remove (or comment out) all those partitions as well in the adf-js-partitions.xml file.

5. To find the feature name, open the JS file in Firebug or similar tool. The createComponentClass usually contains the feature name as shown in Figure E–2.

Figure E–2 Finding the feature name

6. Create a new partition (for example, custom-webcenter), and using the technique above, add all the downloaded features to the new partition as shown in Example E–1. Note that this is only an example, and that your partition will be different depending on the content of the home page.

Example E–1 Sample JavaScript partition

```xml
<partition>
  <partition-name>custom-webcenter</partition-name>
  <feature>AdfRichDialog</feature>
</partition>
```
7. Remove the features which you added to custom from the existing partitions and re-run the application to see the difference in number of downloads and (more importantly) the time to render.

8. Repeat the process for any other complex pages. Note that since the home page will have downloaded most of the JS files they will be cached by the browser and this time there will be a lot less features to partition.
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 topics:

- 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.
  - **WSRP producers** serve as the containers for standards-based portlets.
  - **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 Portal 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 Portal Framework applications.

This section includes the following topics:

- 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 Portal 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 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 Portal Framework application, see Chapter 63, "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 a 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 Portal Framework application. In a Portal 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 Portal 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 Portal 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 Portal Framework application.

For more information about the Federated Portal Adapter, see the "Using the Federated Portal Adapter" chapter in the *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 "Setting Up the Environment to Use the Federated Portal Adapter" section in the *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 Portal 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 "Placing One Page Onto Another" section in the User's Guide for Oracle Portal.

4. Confirm that the users of the Portal 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 Portal 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 Portal 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 check box 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 63.2.2, "Registering an Oracle PDK-Java Portlet Producer with a WebCenter Portal Framework Application."

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 Portal 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 Portal 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 topics:

- 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 Portal 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 the "Oracle Portal - Limitations" section in Administering 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 Portal 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 topics:

■ 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

```
<init-param>
  <param-name>resourceServletMapping</param-name>
  <param-value>/pdkresource</param-value>
</init-param>
```

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 63.2.2, "Registering an Oracle PDK-Java Portlet Producer with a WebCenter Portal Framework Application."

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 63.2.2, "Registering an Oracle PDK-Java Portlet Producer with a WebCenter Portal Framework Application."

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 25, "Configuring Content Repository Connections."
This appendix describes the Expression Language (EL) expressions that you can use in your portals built with WebCenter Portal and Portal Framework applications built with JDeveloper.

This appendix includes the following topics:

- Section G.1, "Introduction to Expression Language (EL) Expressions"
- Section G.2, "Expression Language Example Use Case"
- Section G.3, "ELs Related to Applications"
- Section G.4, "ELs Related to Security"
- Section G.5, "ELs Related to General Settings"
- Section G.6, "ELs Related to Portal Resources"
- Section G.7, "ELs Related to Navigation"
- Section G.8, "ELs Related to Tools and Services"
- Section G.9, "ELs Related to Documents"
- Section G.10, "ELs Related to People Connections"
- Section G.11, "ELs Related to Personalization"
- Section G.12, "ELs Related to Impersonation"
- Section G.13, "EL Expressions Relevant to Composer"
- Section G.14, "EL Expressions Relevant Only to WebCenter Portal"
- Section G.15, "EL Expressions Related to Device Settings"

See Also: For additional information about EL APIs, see Java API Reference for Oracle WebCenter Portal.

G.1 Introduction to Expression Language (EL) Expressions

When configuring page components or assets, you can express values as variables that take advantage of the current application context. For example, you can use variables to obtain the name of the current user, the user's assigned role, the state of the current page, and so on. Such flexibility is made possible by using EL expressions.

This section includes the following subsections:

- Section G.1.1, "Introducing the Expression Builder"
- Section G.1.2, "Introducing the Expression Editor in WebCenter Portal"
G.1.1 Introducing the Expression Builder

You use EL expressions throughout a Portal Framework application to bind attributes to object values determined at runtime. At runtime, the value of certain components is determined by their value attribute. While a component can have static text as its value, typically the value attribute will contain an EL expression that the runtime infrastructure evaluates to determine what data to display.

You can create EL expressions using the JDeveloper Expression Builder. You can access the builder from the Property Inspector. In the Property Inspector, locate the attribute you wish to modify and click the Property Menu icon, and select choose Expression Builder from the popup (Figure G–1).

Figure G–1 Accessing the Expression Builder

In the Expression Builder dialog (Figure G–2), you can directly type your EL expression in the Expression box. You can also use the Variables drop-down list to select items that you want to include in the expression. Use the operator buttons to add logical or mathematical operators to the expression.

For more information about Expression Builder, see the "Getting Started with ADF Faces" chapter in Web User Interface Developer’s Guide for Oracle Application Development Framework.

Figure G–2 Expression Builder Dialog
G.1.2 Introducing the Expression Editor in WebCenter Portal

WebCenter Portal provides a simple Expression Language (EL) editor, called the Expression Editor. Use the Expression Editor when you want to formulate a dynamic computation for an otherwise unknown property value, for example, to specify the current user, the current page mode, and so on.

The Expression Editor is available through the administration and editing screens in WebCenter Portal. You can open the Expression Editor by clicking the Advanced Edit Options icon next to a field, check box or drop-down list, and then clicking Expression Builder. (Figure G–3).

Figure G–3  Advanced Edit Options Icon Next to a Parameter Value Field and the Resulting Editor

The Expression Editor provides two options for entering expressions:

- **Choose a value**, for selecting a seeded EL expression
- **Type a value or expression**, for entering an expression manually

The options under **Choose a value** are categorized according to the type of information an expression returns. Select a category from the first menu, and then select the type of value you want returned from the second menu (Figure G–4).
Many expressions in the tables in this appendix are building blocks for narrowing down the object or objects 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 asset-related ELs to form a single query that returns a particular portal template. It retrieves the template storesMasterTemplate from the parent portal stores:

```el
#{srmContext.resourceScope['stores'].resourceType['siteTemplate'].displayName['storesMasterTemplate'].singleResult}
```

In the Expression Editor, you can either:

- Select **Choose a value**, then select predefined values from the drop-down lists.
- Select **Type a value or expression**, then enter a value or EL expression. Use the following formats:
  - a literal number: #{123}
  - a literal string: #{'string'}
  - a literal Boolean: #{true}
  - a Java Bean called to return a value:
    
    ```el
    #{generalSettings.preferredTimeStyle}
    ```

The editor provides a **Test** button for validating your EL entry. The **Test** button is available for expressions entered in the text box under **Type a value or expression**.
Validation checks the EL syntax and evaluates the expression. Because expression values vary according to the context in which they are executed, the resulting value that appears in the editor may differ from the value returned during actual use.

Note, however, that only EL syntax is validated when you click Test; other types of values are not validated. Test results are shown in a popup.

---

**Note:** When you enter EL in the generic Display Options tab in the Component Properties dialog, the parser reports an error only if it detects invalid syntax, such as a missing closing bracket. Validation is performed only on syntax, not on the expression value. Generic Display Options are those cataloged in the "Display Options Properties" table in Building Portals with Oracle WebCenter Portal. EL validation is not performed on non-generic display options.

---

### G.2 Expression Language Example Use Case

This section provides an example of using EL expressions to display something based on a user's role, permissions, and so on.

In this example, we will use Expression Language to reveal or hide an item in the navigation model based on the user's role. Because there is Expression Language for almost every piece of metadata maintained by Oracle WebCenter Portal, using ELs to personalize the user's experience is an extremely powerful tool.

You can use the following EL to determine a user's role:

```el
#{WCSecurityContext.userInScopedRole['Role 1'] or WCSecurityContext.userInScopedRole['Role 2']}
```

This EL compares the current user's credentials (in this case, the role), against what is specified in the EL. If the result is true, that is, if the user is assigned one of the specified roles then the navigation item is visible to the user. If the result is false, the navigation item is hidden from the user.

The roles referred to in EL can be a system role (such as Administrator, Moderator, or Participant) or a custom role.

Notice that this EL joins two conditions with the use of an OR clause; that is, the user must have either Role 1 OR Role 2 to be able to see the navigational item.

These steps demonstrate how to add a navigational item (a page query for another portal) that is visible only to user assigned the role Moderator.

---

**Note:** This exercise assumes you have applied a custom navigation model to your portal. You cannot edit a seeded navigation model.

---

To personalize navigation in a portal:

1. In the portal administration, go to the Assets page, then click Navigations in the left pane.

   You can also enter the following URL in your browser to navigate directly to the Assets page:

   `http://host:port/webcenter/portal/portalName/admin/assets`
2. Click the **Edit** quick link for the navigation model that the portal is using.

   **Tip:** To find out which navigation model the current portal is using, go to the portal’s administration settings, and look for the value in the **Navigation** field.

3. In the **Edit - navigation model** dialog, expand the **Add** menu, and select **Pages Query**.

4. On the **Target** tab:
   - In the **Name** field, enter a name for your query, for example, **moderatorQuery**.
   - Next to **Find Page in**, select **Portal**, and ensure that the current portal name appears in the field.
   - Click the **Advanced Edit Options** icon next to the **Visible** check box, and select **Expression Builder**.

5. Select **Choose a value**, and then select **Portal Info** from the first drop-down list, and **CurrentPortal Role - Moderator** from the second list.

   The expression `#{WCSecurityContext.userInScopedRole['Moderator']}` appears in the text field.

   **Tip:** If you are working with a custom role, select **CurrentPortal Role - Custom**. In the text area, highlight `CustomRole`, then type in the name of your custom role.

6. Click **OK** until you return to the **Assets** tab.

7. Click **Save**.

   To test this, add a member to the portal, and assign this user any role other than **Moderator**. Log off and log back in as that user. You should not see the Pages Query navigational item.

---

### G.3 ELs Related to Applications

**Table G–1** lists the EL expression relevant to an application and describes the type of functionality it provides. The listed EL applies to both portals built with WebCenter Portal and Portal Framework applications built with JDeveloper.

For information about EL expressions relevant only to WebCenter Portal, see Section G.14.3, “EL Expressions Relevant to WebCenter Portal Information.”

**Table G–1  EL Expression Relevant to an Application**

<table>
<thead>
<tr>
<th>Expression</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>#{requestContext.skinFamily}</code></td>
<td>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 <code>#{adfFacesContext.skinFamily}</code>.</td>
</tr>
</tbody>
</table>

---

### G.4 ELs Related to Security

**Table G.4** lists EL expressions relevant to application security and describes the types of functionality they provide. Some of these expressions can be used in both Portal Framework applications and portals built with WebCenter Portal. The **Context** column
indicates whether the EL is exclusive to Framework applications, exclusive to WebCenter Portal, or can be used with both (Both).

### Table G–2  EL Expressions Relevant to Security

<table>
<thead>
<tr>
<th>Expression</th>
<th>Function</th>
<th>Context</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{security.authenticated}</td>
<td>Returns true when the current user is authenticated in the context in which the EL is being invoked, otherwise false.</td>
<td>Both</td>
</tr>
<tr>
<td>#{securityContext.userName}</td>
<td>Returns the user name of the currently logged in user. If the current user is not logged in, this expression returns no value.</td>
<td>Both</td>
</tr>
<tr>
<td>#{WCSecurityContext.currentUser['userName']}</td>
<td>Returns the value true if the current user is the specified user, otherwise false.</td>
<td>Both</td>
</tr>
<tr>
<td>#{WCSecurityContext.userInAppRole['role']}</td>
<td>Returns the value true if the current user is assigned the specified OPSS application role. This expression is useful in Portal Framework applications that are scope unaware (that is, that do not support portals). Example: #{WCSecurityContext.userInAppRole['AppConnectManager']}</td>
<td>Framework applications</td>
</tr>
<tr>
<td>#{WCSecurityContext.userInGroup['group']}</td>
<td>Returns the value true if the current user is assigned the specified group, for example: #{WCSecurityContext.userInGroup['Administrators']}</td>
<td>Both</td>
</tr>
<tr>
<td>#{security.pageContextCommunityModerator}</td>
<td>Returns the value true if the current user is a moderator of the current portal.</td>
<td>WebCenter Portal</td>
</tr>
<tr>
<td>#{WCSecurityContext.userInScopedRole['role']}</td>
<td>Returns the value true if the current user is assigned the specified role in the current scope. Role can be Moderator, Participant, Viewer, or a custom role defined in that scope. The scope is implicitly resolved to be the current scope. If you use this EL in the Home portal, it resolves to Home portal GUID and roles defined at the application level. If you use this EL in a portal scope, it resolves to roles defined for the portal.</td>
<td>WebCenter Portal</td>
</tr>
</tbody>
</table>

### G.4.1 Example: Restricting Access to Structure View in Composer Using EL Expressions

This section provides an example of using EL expressions to display something based on a user’s role. In the example, you will create a page style and edit its source code to add an EL expression that allows only users granted the AdvancedEdit role to be able to access Structure view of Composer in the Home portal.

To restrict access to Structure view in Composer in the Home portal:

1. Log on to WebCenter Portal with administrative privileges, and navigate to the Security tab in WebCenter Portal Administration.
2. Create a new role called AdvancedEdit, and assign it to the user, weblogic.

For information about application-level roles and permissions in WebCenter Portal, see the “Managing Security Across Portals” chapter in *Administering Oracle WebCenter Portal*. 

Expression Language Expressions  G-7
The user `weblogic` will now have two roles: Administrator and AdvancedEdit.

3. Create a new page style, say Blank Page (For information about creating page styles, see the "Creating a Page Style" section in Building Portals with Oracle WebCenter Portal.)

4. To grant access to Composer’s Structure view only to the users with the AdvancedEdit role, edit the source code of the page style and add the following attribute to the `<pe: pageCustomizable>` tag (Figure G–6):

```
toolbarLayout="message stretch statusindicator newline designViews="design
#WCSecurityContext.userInScopedRole('AdvancedEdit') ? 'source' : ")
addPanels stretch help button"
```

For information about editing the source code, see the "Editing the Source Code of an Asset" section in Building Portals with Oracle WebCenter Portal.

**Figure G–6  Editing Source Code of a Page Style**

5. Log in as a user other than `weblogic`, and in the Home portal create a page using the Blank Page page style. When you open the page in Composer, the page appears in Design view, and no Structure tab is available because the user is not granted the AdvancedEdit role.

Applying this technique to all page styles available to users in the Home portal completely controls access to Structure view. You can follow the same technique for other portals.

### G.5 ELs Related to General Settings

Table G.5 lists EL expressions relevant to general application settings and describes the types of functionality they provide. All listed ELs apply to both portals built with WebCenter Portal and Portal Framework applications built with JDeveloper.
Use the expressions in this section to query for portal resources. Querying for a portal 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 so on, 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 portal resource. When no matching resource is found, null is returned.
- **resultList**—Returns a list of portal resources. When no matching portal resources are found, an empty list is returned.

### Table G–3  EL Expressions Relevant to General Settings

<table>
<thead>
<tr>
<th>Expression</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{generalSettings.userTimeZone}</td>
<td>Returns the time zone the current user has selected in application preferences.</td>
</tr>
<tr>
<td>#{generalSettings.preferredDateStyle}</td>
<td>Returns the date format the current user has selected in application preferences.</td>
</tr>
<tr>
<td>#{generalSettings.preferredDatePattern}</td>
<td>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 Portal Framework applications, where they take precedence over date + style if they are set.</td>
</tr>
<tr>
<td>#{generalSettings.preferredTimeStyle}</td>
<td>Returns the time format the current user has selected in application preferences.</td>
</tr>
<tr>
<td>#{generalSettings.preferredTimePattern}</td>
<td>Returns the current user's preferred time format pattern if it has been set, else, returns null.</td>
</tr>
<tr>
<td>#{generalSettings.preferredDateTimePattern}</td>
<td>Returns the current user's preferred date and time format pattern if it has been set, else, returns null.</td>
</tr>
<tr>
<td>#{generalSettings.preferredAccessibilityMode}</td>
<td>Returns the current user's preferred accessibility mode (either default, inaccessible, or screenReader)</td>
</tr>
<tr>
<td>#{generalSettings.preferredSkinName}</td>
<td>The current user's preferred skin name if one is specified, otherwise the default value.</td>
</tr>
<tr>
<td>#{generalSettings.formattedCurrentDate}</td>
<td>Returns the current date in the current user's selected locale.</td>
</tr>
<tr>
<td>#{generalSettings.formattedCurrentTime}</td>
<td>Returns the current time in the current user's selected locale.</td>
</tr>
<tr>
<td>#{generalSettings.formattedCurrentDateTime}</td>
<td>Returns the current date and time in the current user's selected locale.</td>
</tr>
<tr>
<td>#{requestContext.skinFamily}</td>
<td>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.</td>
</tr>
</tbody>
</table>
The following example returns the first page template portal resource found with a display name that contains myPage:

#{srmContext.resourceType['siteTemplate'].displayName['myPage'].singleResult}

The following example returns a list of page template portal resources residing in the directory resourceDir, with a description that contains sampleDesc:

#{srmContext.resourceType['siteTemplate'].description['sampleDesc'].contentDirectory['resourceDir'].resultList}

A property of this class includes any attribute of this class. Example properties include: Id, DisplayName, iconURI, contentDirectory, and so on.

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'].singleResult}

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. Many of the expressions in Table G–4 are building blocks for narrowing down the portal 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 portal resource-related ELs to form a single query that returns a particular portal template. It retrieves the template storesMasterTemplate from the parent portal stores:

#{srmContext.resourceScope['stores'].resourceType['siteTemplate'].displayName['storesMasterTemplate'].singleResult}

For information about EL expressions relevant only to WebCenter Portal assets, see Section G.14.4, "EL Expressions Relevant to Assets."
Table G–4  ELs Relevant to Portal Resources

<table>
<thead>
<tr>
<th>EL</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{srmContext.id['resourceGUID']}</td>
<td>Returns the portal resource with the specified ID</td>
</tr>
<tr>
<td>#{srmContext.displayName['resourceDisplayName']}</td>
<td>Returns any portal resources with the specified display name</td>
</tr>
<tr>
<td>#{srmContext.displayNameKey['displayNameKey']}</td>
<td>Returns any portal resources with the specified display name key</td>
</tr>
<tr>
<td>#{srmContext.description['resourceDescription'].singleResult}</td>
<td>Returns one result of a portal resource that contains the specified value in its description. To get multiple results, use resultList in lieu of singleResult.</td>
</tr>
<tr>
<td>#{srmContext.descriptionKey[descriptionKey].singleResult}</td>
<td>Returns one result of a portal resource with the specified description key. The description key is the key in the xsrt file for the portal resource description. To get multiple results, use resultList in lieu of singleResult.</td>
</tr>
<tr>
<td>#{srmContext.createdDate['resourceCreationDate']}</td>
<td>Returns any portal resource with the specified creation date</td>
</tr>
<tr>
<td>#{srmContext.modifiedBy['resourceLastModifiedBy']}</td>
<td>Returns any portal resource that was last modified by the user with the specified user name</td>
</tr>
<tr>
<td>#{srmContext.modifiedDate['resourceLastModifiedDate']}</td>
<td>Returns any portal resource that was last modified by the specified date</td>
</tr>
<tr>
<td>#{srmContext.resourceScope['resourceScope']}</td>
<td>Returns any portal resource that falls within the specified scope</td>
</tr>
</tbody>
</table>
| #{srmContext.category['categoryName']}  | Returns any portal resource that falls within the specified category. For example:  
|  |  
|  |  
| #{srmContext.contentDirectory['contentDirectory']}  | Returns any portal resource that is stored within the specified directory |
| #{srmContext.metadataFile['metadataFileLocation']}  | Returns portal resource metadata from the specified metadata file. For example, the following expression returns resource metadata from the following file /home/metadata/data.xml:  
|  |  
|  |  
|  |  
| #{srmContext.jspx['jspxFileLocation']}  | Returns any jspx file in the specified location. For example, the following expression returns the page.jspx file:  
|  |  
|  |  
| #{srmContext.pageDef['pageDefinition']}  | Returns any jspx file with the specified page definition |
| #{srmContext.iconURI['iconURI']}  | Returns the icon at the specified icon URI |
### Table G–4  (Cont.) ELs Relevant to Portal Resources

<table>
<thead>
<tr>
<th>EL</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{srmContext.excludeFrom['excludeFromScopes']}</td>
<td>In a larger expression, returns all specified portal resources except those available in the excluded scopes</td>
</tr>
<tr>
<td>#{srmContext.usesCustomSecurity['usesCustomSecurity']}</td>
<td>In a larger expression, returns any portal resource that either does or does not use custom security. Set <code>usesCustomSecurity</code> to TRUE or FALSE. For example: &lt;br&gt;#{srmContext.usesCustomSecurity['TRUE']}</td>
</tr>
<tr>
<td>#{srmContext.seeded['seeded']}</td>
<td>In a larger expression, returns any portal resource that is or is not seeded, depending on the provided value. Set <code>seeded</code> to TRUE or FALSE. For example: &lt;br&gt;#{srmContext.seeded['TRUE']}</td>
</tr>
<tr>
<td>#{srmContext.visibleType['visibleType']}</td>
<td>In a larger expression, returns any portal resource that is of the specified type of visibility. Set <code>visibleType</code> to TRUE, FALSE, or NEVER to indicate that the portal resource is never shown. For example, the following expression returns portal resources that are visible, that is, portal resources that are set to Show: &lt;br&gt;#{srmContext.visibleType['TRUE']}</td>
</tr>
<tr>
<td>#{srmContext.visible['visible']}</td>
<td>In a larger expression, returns one or multiple portal resources with visibility set to either TRUE or FALSE</td>
</tr>
<tr>
<td>#{srmContext.createdBy['resourceCreator']}</td>
<td>In a larger expression, returns any portal resource created by the specified user</td>
</tr>
<tr>
<td>#{srmContext.resourceType['resourceType']}</td>
<td>In a larger expression, returns one or multiple portal resources of the specified type. For example, the following expression searches for a portal resource of the type SITE_TEMPLATE: &lt;br&gt;#{srmContext.resourceType['SITE_TEMPLATE']}</td>
</tr>
</tbody>
</table>

Note that all resource types are listed in *Java API Reference for Oracle WebCenter Portal*, in the `GenericSiteResourceTypes` class.
Table G–4  (Cont.) ELs Relevant to Portal Resources

<table>
<thead>
<tr>
<th>EL</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{srmContext.version['version']}</td>
<td>In a larger expression, returns one or multiple portal resources available in the application of the specified version</td>
</tr>
</tbody>
</table>
| #{srmContext.resourceScope['scopeName']} | In a larger expression, returns one or multiple portal resources available in the specified scope. For example, the following expression searches for portal resources in the scope (in this instance, the portal) MyPortal:

#{srmContext.resourceScope['MyPortal']}

To search in the default scope, that is, the application scope, use defaultScope. |
| #{srmContext.searchType['searchType']} | In a larger expression, returns one or multiple portal resources that contain or equal the values set by other included expressions. Set searchType to CONTAINS or EQUALS. |

G.7 ELs Related to Navigation

Table G.7 lists EL expressions relevant to application navigation and describes the types of functionality they provide. All listed navigation ELs apply to both portals built with WebCenter Portal and Portal Framework applications built with JDeveloper.

Table G–5  EL Expressions Relevant to Navigation

<table>
<thead>
<tr>
<th>Expression</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Navigation Context Expressions</td>
<td></td>
</tr>
</tbody>
</table>
| #{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}. |
Table G-5  (Cont.) EL Expressions Relevant to Navigation

<table>
<thead>
<tr>
<th>Expression</th>
<th>Function</th>
</tr>
</thead>
</table>
| `#{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/navigations
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

<table>
<thead>
<tr>
<th>Expression</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{navigationContext.processAction}</td>
<td>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:</td>
</tr>
</tbody>
</table>
| | `<af:forEach var='node' varStatus='vs' items='#{navigationContext.defaultNavigationModel.rooTreeNode.children}'>
<af:subform id='pt_sfml'>
<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' triggerTypes='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>` |
| #{navigationContext.defaultNavigationModel.defaultTreeModel} | 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 |
| | ■ ListNavigationResource—used in `<af:foreach>` |
### Table G–5 (Cont.) EL Expressions Relevant to Navigation

<table>
<thead>
<tr>
<th>Expression</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{navigationContext.defaultNavigationModel.treeModel['parameters']}</td>
<td>Returns tree/menu/list model based on the specified comma-separated parameters. Available parameters (with default values as examples) are:</td>
</tr>
<tr>
<td>■ startNode=/&gt;—specify the starting node of the model (do not need &quot;/&quot; prefix unless requesting the root node, for example, home).</td>
<td></td>
</tr>
<tr>
<td>■ 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.</td>
<td></td>
</tr>
<tr>
<td>■ depth=0—defines the initial depth of fetching. &quot;0&quot; means fetch the entire tree, which may take a long time. In which case, use &quot;1&quot; to fetch on demand (when users click the Expand icon).</td>
<td></td>
</tr>
<tr>
<td>■ 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.</td>
<td></td>
</tr>
</tbody>
</table>

For siteMap, the available parameters are startNode and includeStartNode.

Example:

#{navigationContext.defaultNavigationModel.treeModel['startNode=home,includeStartNode=false,depth=2']} |

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 portal resource. |
### 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:

```jsp
<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">
    ...
  </af:forEach>
</c:if>
```

#{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 Portal Resource Expressions

#{node.attributes['Description']} | Returns the value of the specified attribute of the navigation portal resource.

#{node.parameters['StockSymbol']} | Returns the value of the specified parameter of the navigation portal resource.

#{node.parametersRaw['StockSymbol']} | Returns the raw value of the specified parameter of the navigation portal resource (before it is evaluated).

#{node.title} | Returns the title of the navigation portal resource.

#{node.path} | Returns the path to the navigation portal resource.

#{node.externalURL} | Returns the URL for the navigation portal resource of type `External Link`.

#{node.prettyUrl} | Returns the identifying path for this navigation portal resource.

#{node.prettyUrlPath} | 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`:

```jsp
#{node.prettyUrlPath[3]}
```
<table>
<thead>
<tr>
<th>Expression</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{node.goLinkPretty Url}</td>
<td>Returns the identifying path for this navigation portal resource suitable for goLink destination.</td>
</tr>
<tr>
<td>&lt;af:goLink id=&quot;pt_gl2&quot; text=&quot;#{node.title}&quot; destination=&quot;#{node.goLinkPrettyUrl}&quot; targetFrame=&quot;#{node.attributes['Target']}&quot; inlineStyle=&quot;font-size:small;#{node.selected ? 'font-weight:bold;' : ''}&quot;/&gt;</td>
<td></td>
</tr>
<tr>
<td>#{node.separator ? ... : ...}</td>
<td>Returns whether this portal resource is a separator item.</td>
</tr>
<tr>
<td>#{node.navigable ? ... : ...}</td>
<td>Returns whether it is possible to navigate to this portal resource.</td>
</tr>
<tr>
<td>#{node.selected ? ... : ...}</td>
<td>Returns whether this portal resource is currently selected.</td>
</tr>
<tr>
<td>#{node.currentlySelected ? ... : ...}</td>
<td>Returns whether the portal resource is the currently selected portal resource and the model is the currently selected model.</td>
</tr>
<tr>
<td>#{node.onSelectedPath ? ... : ...}</td>
<td>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.</td>
</tr>
</tbody>
</table>
| `<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

<table>
<thead>
<tr>
<th>Expression</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{$node.nextSibling}</td>
<td>Returns the next sibling in the list (of type NavigationResource) of this node.</td>
</tr>
<tr>
<td></td>
<td>For example, the following code ensures that you do not output two separators in a row:</td>
</tr>
<tr>
<td></td>
<td>`&lt;c:if test=&quot;${(node.separator) &amp;&amp; (!node.nextSibling.separator)}&quot;&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;af:separator id=&quot;s166&quot;/&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/c:if&gt;</td>
</tr>
<tr>
<td>#{$node.previousSibling}</td>
<td>Returns the previous sibling in the list (of type NavigationResource) of this node.</td>
</tr>
<tr>
<td>#{$node.index}</td>
<td>Returns the zero-relative index of this node relative to its siblings.</td>
</tr>
<tr>
<td>#{$node.children}</td>
<td>Returns a collection of child resources.</td>
</tr>
<tr>
<td>#{$node.children[N].title}</td>
<td>Returns the number of children of this node.</td>
</tr>
<tr>
<td>#{$node.childCount}</td>
<td>Returns the child node (of type NavigationResource) specified by the zero-relative index.</td>
</tr>
<tr>
<td></td>
<td>If not found, this returns null.</td>
</tr>
<tr>
<td>#{$node.childByPath['admin']}</td>
<td>Returns the child node (of type NavigationResource) specified by the path relative to this node.</td>
</tr>
<tr>
<td></td>
<td>The path can be deeper than one level and does not need the '/' prefix.</td>
</tr>
<tr>
<td></td>
<td>If not found, this returns null.</td>
</tr>
<tr>
<td>#{$node.properties['propertyName']}</td>
<td>Returns the value of the specified property of the navigation portal resource, where propertyName is one of the following: separator, title, prettyUrl, prettyUrlPath, depth, leaf, childCount, navigable, selected, or currentlySelected.</td>
</tr>
</tbody>
</table>

G.7.1 Example: Using EL Expressions for Navigation

This section provides an example of using a navigation model to monitor activities of two subportals from a shared page in the parent portal. In this example, you use a page containing the Activity Stream task flow. The content of the task flow is determined by a parameter passed to the page. The navigation model includes two links that link to the page. The first link passes subportal1 as the parameter so that the Activity Stream task flow shows activities of subportal1. The second link passes subportal2 as the parameter. So, the content of the page is different depending on which link you click in the navigation.

To build a navigation model that uses ELs to monitor subportal activity from a parent portal:

1. In WebCenter Portal, create a portal, and two subportals. While creating subportals, note down the internal name used (for example, subportal1 and subportal2).

2. In your portal, create a page called Subportal Activity, and add the Activity Stream task flow to the page. For information about adding a task flow, see the "Adding Components to a Page" section in Building Portals with Oracle WebCenter Portal.
3. Create a page parameter, say subportalName, for the newly created page. For information, see the "Adding or Modifying Page Parameters" section in Building Portals with Oracle WebCenter Portal.

Use the following EL as the value of the parameter:

```el
#{empty
navigationContext.currentNavigationModel.currentSelection.parameters.subspaceName ? 'subportal1' :
navigationContext.currentNavigationModel.currentSelection.parameters.subspaceName}
```

where 'subportal1' is the value to be used for the page parameter if none is passed to it from the navigation.

4. Edit the settings of the Activity Stream task flow. In the Component Properties dialog, on the Parameters tab, for the Spaces parameter, open the Expression Builder, and for Page Parameter, select subportalName.

5. On the Display Options tab, for the Text field, specify a value that shows the name of the subportal whose activity is shown. For example, use the following EL:

```el
#{bindings.subspaceName} Activity
```

6. Build a navigation that passes the portal to show your page:
   a. Create a navigation based on Portal Default Navigation Model.
   b. Edit the navigation by adding a Link navigation item.
   c. On the Target tab, in the Name field, specify the name of the Link item as Sub Portal 1.
   d. In the Path field, select the Subportal Activity page.
   e. On the Parameters tab, for the subportalName field, specify the internal name of the first subportal, subportal1.

7. Copy the newly created link, and edit it to make a link for the second subportal, subportal2.
   While editing, on the Target tab you only need to change the Name of the link. On the Parameters tab, you only need to change the value of the subportalName parameter to subportal2 (the internal name of the second subportal).

8. Make sure the newly created navigation is set as the current navigation for your portal.

9. You now have two links that take you to the shared Subportal Activity page, passing different values for the portal depending on the link you click.

G.8 ELs Related to Tools and Services

Table G.8 lists EL expressions relevant to tools and services and describes the types of functionality they provide. All listed ELs apply to both portals built with WebCenter Portal and Portal Framework applications built with JDeveloper.
**Table G–6  EL Expressions Relevant to Tools and Services**

<table>
<thead>
<tr>
<th>Expression</th>
<th>Function</th>
</tr>
</thead>
</table>
| #{webcenterService['serviceId']} | An `oracle.webcenter.framework.service.Service` object representing the WebCenter Portal tool or service represented by the service ID `serviceId`. For example, the following EL returns Documents Service:  
#{webcenterService['oracle.webcenter.doclib']} |
| #{webcenterService['serviceId'].configured} | Returns a Boolean value that indicates whether the WebCenter Portal tool or service represented by the service ID `serviceId` is configured for use in the current Portal Framework application. For example, the following EL returns `true` if discussions can be used in the application, `false` otherwise:  
#{webcenterService['oracle.webcenter.collab.forum'].configured} |
| #{sessionContext['oracle.webcenter.collab.forum'].groupInfo['portalName'].forumId} | Returns the forum ID of the specified portal discussion forum. Enter the portal name in lieu of `portalName`. |
| #{sessionContext['oracle.webcenter.collab.forum'].groupInfo['portalName'].categoryId} | Returns the category ID of the specified portal discussion forums. Enter the portal name in lieu of `portalName`. |

For service IDs, see **Table G–7**.

---

**Table G–7  Service and Tool IDs**

<table>
<thead>
<tr>
<th>Service/Tool</th>
<th>ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Announcements</td>
<td>oracle.webcenter.collab.announcement</td>
</tr>
<tr>
<td>Discussions</td>
<td>oracle.webcenter.collab.forum</td>
</tr>
<tr>
<td>Documents and Wikis</td>
<td>oracle.webcenter.doclib</td>
</tr>
<tr>
<td>Events</td>
<td>oracle.webcenter.collab.calendar.community</td>
</tr>
<tr>
<td>Instant Messaging and Presence (IMP)</td>
<td>oracle.webcenter.collab.rtc</td>
</tr>
<tr>
<td>Links</td>
<td>oracle.webcenter.relationship</td>
</tr>
<tr>
<td>Lists</td>
<td>oracle.webcenter.list</td>
</tr>
<tr>
<td>Mail</td>
<td>oracle.webcenter.collab.mail</td>
</tr>
<tr>
<td>Notifications</td>
<td>oracle.webcenter.notification</td>
</tr>
<tr>
<td>Page</td>
<td>oracle.webcenter.page</td>
</tr>
<tr>
<td>People Connections: Activity Stream</td>
<td>oracle.webcenter.activitystreaming</td>
</tr>
<tr>
<td>People Connections: Connections</td>
<td>oracle.webcenter.peopleconnections.connections</td>
</tr>
<tr>
<td>People Connections: Feedback</td>
<td>oracle.webcenter.peopleconnections.kudos</td>
</tr>
<tr>
<td>People Connections: Message Board</td>
<td>oracle.webcenter.peopleconnections.wall</td>
</tr>
<tr>
<td>People Connections: Profile</td>
<td>oracle.webcenter.peopleconnections.profile</td>
</tr>
<tr>
<td>Polls</td>
<td>oracle.webcenter.collab.survey</td>
</tr>
<tr>
<td>Recent Activities</td>
<td>oracle.webcenter.recentactivity</td>
</tr>
</tbody>
</table>
G.9 ELs Related to Documents

Table G–8 lists EL expressions relevant to documents and describes the types of functionality they provide. All listed ELs apply to both portals built with WebCenter Portal and Portal Framework applications built with JDeveloper.

<table>
<thead>
<tr>
<th>Expression</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{documentsService.defaultConnectionName}</td>
<td>Gets the default content repository connection name. Returns null if no connection name is defined.</td>
</tr>
<tr>
<td>#{documentsService.configured}</td>
<td>Checks whether the documents feature is configured. Returns true if the documents feature is configured, otherwise false.</td>
</tr>
</tbody>
</table>

G.10 ELs Related to People Connections

Table G–9 lists EL expressions relevant to the Profile feature of the people connections and describes the types of functionality they provide. All listed ELs apply to both portals built with WebCenter Portal and Portal Framework applications built with JDeveloper.

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>
<thead>
<tr>
<th>Expression</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{webCenterProfile[userName].ManagerDisplayName}</td>
<td>The display name of the specified user's manager if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].ManagerDisplayName}</td>
<td>The display name of the specified user's manager if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[userName].employeeNumber}</td>
<td>The specified user's employee number if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].employeeNumber}</td>
<td>The specified user's employee number if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[userName].businessPOBox}</td>
<td>The post office box number associated with the specified user if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].businessPOBox}</td>
<td>The post office box number associated with the specified user if the current user is allowed to know it.</td>
</tr>
<tr>
<td>Expression</td>
<td>Function</td>
</tr>
<tr>
<td>------------</td>
<td>----------</td>
</tr>
<tr>
<td>#{webCenterProfile[userName].timeZone}</td>
<td>The time zone in which the specified user's home office is located if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].timeZone}</td>
<td></td>
</tr>
<tr>
<td>#{webCenterProfile[userName].description}</td>
<td>A description of the specified user (from Profile &quot;About Me&quot;) if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].description}</td>
<td></td>
</tr>
<tr>
<td>#{webCenterProfile[userName].department}</td>
<td>The department to which the specified user belongs if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].department}</td>
<td></td>
</tr>
<tr>
<td>#{webCenterProfile[userName].businessPager}</td>
<td>The specified user's business pager number if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].businessPager}</td>
<td></td>
</tr>
<tr>
<td>#{webCenterProfile[userName].businessCity}</td>
<td>The city in which the specified user is located if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].businessCity}</td>
<td></td>
</tr>
<tr>
<td>#{webCenterProfile[userName].maidenName}</td>
<td>The specified user's surname or last name before marriage if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].maidenName}</td>
<td></td>
</tr>
<tr>
<td>#{webCenterProfile[userName].businessFax}</td>
<td>The specified user's business fax number if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].businessFax}</td>
<td></td>
</tr>
<tr>
<td>#{webCenterProfile[userName].dateofHire}</td>
<td>The specified user's date of hire if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].dateofHire}</td>
<td></td>
</tr>
<tr>
<td>#{webCenterProfile[userName].nameSuffix}</td>
<td>Additional information appended to the user's name, such as Esq., Jr., and so on if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].nameSuffix}</td>
<td></td>
</tr>
<tr>
<td>#{webCenterProfile[userName].middleName}</td>
<td>The specified user's middle name if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].middleName}</td>
<td></td>
</tr>
<tr>
<td>#{webCenterProfile[userName].homePhone}</td>
<td>The specified user's home phone number if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].homePhone}</td>
<td></td>
</tr>
<tr>
<td>#{webCenterProfile[userName].employeeType}</td>
<td>The specified user's employee type classification, for example, IC4 if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].employeeType}</td>
<td></td>
</tr>
<tr>
<td>#{webCenterProfile[userName].lastName}</td>
<td>The specified user's surname or last name if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].lastName}</td>
<td></td>
</tr>
<tr>
<td>#{webCenterProfile[userName].dateofBirth}</td>
<td>The specified user's birthday if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].dateofBirth}</td>
<td></td>
</tr>
<tr>
<td>#{webCenterProfile[userName].businessState}</td>
<td>The state in which the specified user's home office is located if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].businessState}</td>
<td></td>
</tr>
</tbody>
</table>
### EL Expressions Relevant to People Connections (Profile)

<table>
<thead>
<tr>
<th>Expression</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{webCenterProfile[userName].homeAddress}</td>
<td>The specified user's home street address if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].homeAddress}</td>
<td>The specified user's home street address if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[userName].businessStreet}</td>
<td>The street on which the specified user's home office is located if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].businessStreet}</td>
<td>The street on which the specified user's home office is located if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[userName].businessPostalCode}</td>
<td>The specified user's postal or ZIP code if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].businessPostalCode}</td>
<td>The specified user's postal or ZIP code if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[userName].initials}</td>
<td>The specified user's initials if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].initials}</td>
<td>The specified user's initials if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[userName].firstName}</td>
<td>The specified user's first name if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].firstName}</td>
<td>The specified user's first name if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[userName].organizationalUnit}</td>
<td>The organizational unit to which the specified user belongs, for example, D10, if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].organizationalUnit}</td>
<td>The organizational unit to which the specified user belongs, for example, D10, if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[userName].wirelessAcctNumber}</td>
<td>The specified user's wireless account number if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].wirelessAcctNumber}</td>
<td>The specified user's wireless account number if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[userName].businessPhone}</td>
<td>The specified user's business telephone number if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].businessPhone}</td>
<td>The specified user's business telephone number if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[userName].businessCountry}</td>
<td>The country in to which the specified user is assigned if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].businessCountry}</td>
<td>The country in to which the specified user is assigned if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[userName].preferredLanguage}</td>
<td>The specified user's preferred language if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].preferredLanguage}</td>
<td>The specified user's preferred language if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[userName].displayName}</td>
<td>The specified user's display name if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].displayName}</td>
<td>The specified user's display name if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[userName].userName}</td>
<td>The specified user's actual name if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].userName}</td>
<td>The specified user's actual name if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[userName].title}</td>
<td>The specified user's job title if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].title}</td>
<td>The specified user's job title if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[userName].businessEmail}</td>
<td>The specified user's business email address if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].businessEmail}</td>
<td>The specified user's business email address if the current user is allowed to know it.</td>
</tr>
</tbody>
</table>
Table G–9  (Cont.) EL Expressions Relevant to People Connections (Profile)

<table>
<thead>
<tr>
<th>Expression</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{webCenterProfile[userName].organization}</td>
<td>The organization to which the specified user belongs, for example, Sales, if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].organization}</td>
<td></td>
</tr>
<tr>
<td>#{webCenterProfile[userName].businessMobile}</td>
<td>The specified user’s cell or mobile phone number if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].businessMobile}</td>
<td></td>
</tr>
<tr>
<td>#{webCenterProfile[userName].expertise}</td>
<td>The specified user’s business expertise if the current user is allowed to know it.</td>
</tr>
<tr>
<td>#{webCenterProfile[securityContext.userName].expertise}</td>
<td></td>
</tr>
</tbody>
</table>

G.11 ELs Related to Personalization

Table G–10 lists EL expressions relevant to personalization and describes the types of functionality they provide. All listed ELs apply to both portals built with WebCenter Portal and Portal Framework applications built with JDeveloper.
## Table G–10  EL Expressions Relevant to Personalization

<table>
<thead>
<tr>
<th>Expression</th>
<th>Function</th>
</tr>
</thead>
</table>
| `{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. |
### G.12 ELs Related to Impersonation

Table G–10 lists EL expressions relevant to WebCenter Portal Impersonation and describes the types of functionality they provide. All listed ELs apply to both portals built with WebCenter Portal and Portal Framework applications built with JDeveloper.

For more information about using these ELs, see Section 74.19, "Using WebCenter Portal Impersonation ELs and APIs."

<table>
<thead>
<tr>
<th>Expression</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{p13nContext.conductor['ConductorConnection'].namespaces['MyNamespace'].scenario['MyScenario'].withInput['input1=val1;input2=value2;input3=value3'].results}</td>
<td>Executes a scenario by name with input parameters. Input parameters must be in the following format: <code>&lt;paramOne&gt;=&lt;valueOne&gt;;&lt;paramTwo&gt;=&lt;valueTwo&gt;;&lt;paramThree&gt;=&lt;valueThree&gt;</code> Where each parameter name and value is separated by semicolon. The withInput method applies only to the context ScenarioExecutionContext.</td>
</tr>
<tr>
<td>#{p13nContext.conductor['ConductorConnection'].namespaces['myManagedBean.namespace'].scenario['myManagedBean.scenario'].isError}</td>
<td>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'].isErro}r</td>
</tr>
<tr>
<td>#{p13nContext.conductor['ConductorConnection'].namespaces['myManagedBean.namespace'].scenario['myManagedBean.scenario'].errorMessage}</td>
<td>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}</td>
</tr>
</tbody>
</table>
Table G–11  EL Expressions Relevant to Impersonation

<table>
<thead>
<tr>
<th>Expression</th>
<th>Function</th>
</tr>
</thead>
</table>
| #{WCSecurityContext.impersonationConfigured}    | Returns whether or not impersonation has been enabled for the current domain.  
This EL can be useful when determining if an error was caused by an impersonation session ending prematurely, or to provide an additional indicator that a session has ended. |
| #{WCSecurityContext.userInImpersonationSession} | Returns whether the current user is in an impersonation session or not.  
You can use this EL to protect content and render it inaccessible during an impersonation session. For example, you could map the rendered attribute of an administration task flow on a page to this EL only rendering the task flow if the user is not viewing the task flow in an impersonation session. |
| #{WCSecurityContext.currentImpersonator}       | This EL could be used to modify the page template to display the impersonator or render content accessible only to a particular impersonator. |

Table G–12  EL Expressions Relevant to Composer

<table>
<thead>
<tr>
<th>Expression</th>
<th>Function</th>
</tr>
</thead>
</table>
| #{composerContext.inEditMode}                  | Determines whether the current page is in View mode or Edit mode.  
For example:  
<af:outputText id="ot1"  
value="#{composerContext.inEditMode ne true ? 'View mode' : 'Edit mode'}" />  
This example shows an ADF Faces outputText component that returns View mode when the page is in View mode and Edit mode when the page is in Edit mode. |
| #{changeModeBean.inEditMode}                   | The childCreation attribute determines whether the children of popup dialogs can be viewed in Composer. By default, this attribute is set to deferred and users are unable to see the children of popup dialogs. Setting the value to immediate allows users to view the children of popup dialogs.  
For example:  
<af:popup childCreation="#{changeModeBean.inEditMode ne 'immediate' : 'deferred'}" />  
There is a performance impact by setting the childrenCreation attribute to immediate. If you do not need to use the popup on the page, using the default setting of deferred will avoid the penalty of CPU and memory usage when it is not needed. Note that the popup will be shown non-modally, thus, the application must be able to handle that fact without error. |
G.13.1 Example: Using EL Expressions for Composer

This section provides an example of using an EL expression to display different pages in View mode and Edit mode. Consider that you want one of your pages, Page1, to display another page, Page2, in View mode. However, while editing Page1, instead of Page2, you want to display a website, www.example.com. You can use the EL of the following format:

```java
```

This EL displays the specified website when your page is in Edit mode, and shows Page2 when in View mode.

The following steps demonstrate how to display different pages depending on the page mode.

1. Open a page named Page1 in the page editor (Composer).
2. In Design view, in the resource catalog, click Web Development.
3. Click Add displayed next to Web Page.
4. Click the Edit icon for the newly added Web Page.
5. In the Component Properties dialog, on the Display Options tab, in the Source field, enter the value as per the following format:
   ```java
   #{ composerContext.inEditMode ? 'website_url' : 'url_of_Page2' }
   ```
6. Click OK.

In Composer, Page1 should now display the website you specified. Save and close the page. Page1 should now show Page2.

G.14 EL Expressions Relevant Only to WebCenter Portal

This section contains the following subsections:

- Section G.14.2, "Seeded Expressions in the Expression Editor"
- Section G.14.3, "EL Expressions Relevant to WebCenter Portal Information"
- Section G.14.4, "EL Expressions Relevant to Assets"
- Section G.14.5, "EL Expressions Relevant to Specific Portals"
- Section G.14.6, "EL Expressions Relevant to Specific Pages"
- Section G.14.7, "EL Expressions Relevant to Portal Event Contexts"
- Section G.14.8, "Utilitarian EL Expressions"

G.14.1 Desupport of Freeform JPQL WHERE and SORT Clauses in Portal Queries

In WebCenter Portal, if you have introduced an EL expression for querying portals based on a JPQL WHERE clause, you should know that the freeform JPQL WHERE clause is desupported. Instead, use a structured WHERE clause. Freeform SORT clauses are also desupported. Instead, specify sort criteria on a portal query in a structured way.

For example, suppose that you have used the following syntax to form a WHERE clause on a portal query:

```java
#{spaceContext.spacesQuery.unionOf['ALL_QUERIABLE'].
    whereClause['sp.keywords like \'forquery%\' and not
    (sp.discoverable is null or sp.discoverable = \
    \'N\')'].listSpaces}
```

The freeform JPQL WHERE clause is desupported in WebCenter Portal.
Going forward, you must use the following syntax to form a WHERE clause:

```java
#{spaceContext.spacesQuery.unionOf['ALL_QUERIABLE'].
  where[wCond['sp.keywords']['like']['forquery']['and'][wCond['sp.discoverable']
  ['is']][['null']['or'][wCond['sp.discoverable']['=']['N']['not']]].listSpaces}
```

Where:

- `wCond` is an atomic condition that can be expressed as `wCond[p][op][v]`.

  Where:
  - `p` is a portal attribute (a field in `WcSpaceHeader`, see the following bullet list).
  - `op` is one among a restricted set of JPQL comparison operators (`'='`, `'like'`, `'='`, `'<'`, `'>', '<=', '>=', `'is'`).
  - `v` is a string whose interpretation depends on the operator `op`.

    If the operator is `is`, `v` can be the string `null` or `not null`, and the condition is understood as the attribute being checked for null or not null respectively in JPQL. If the operator is anything else, `v` is interpreted as a literal, and the condition is understood to be the attribute being compared with the literal value.

You can use any field defined in `WcSpaceHeader` in the WHERE clause, including:

- `closed`
- `createDate`
- `createdBy`
- `description`
- `discoverable`
- `displayName`
- `keywords`
- `icon`
- `logo`
- `lastUpdateDate`
- `selfSubEnabled`
- `rssEnabled`
- `spaceGuid`
- `spaceId`
- `spaceOffline`
- `spacePublic`
- `spacesType`
- `spacesUser`
- `updatedBy`
- `version`
- `parentGuid`
- `securityParentGuid`
- ancestorPath
- seeded
- groupSpaceMemCount

- **sp** represents the JPA entity for a portal, `WcSpaceHeader`.

For example, you can use the following condition to create a filter for all portals created by users having user names starting with `monty` on which self subscription is enabled:

```
sp.createdBy like 'monty%' and not (sp.selfSubEnabled is null or sp.selfSubEnabled = 'N')
```

- **and**, **or**, **not** (along with `wCond`) are types of conditions.

  Where:

  - **and** represents a conjunction of two conditions. You can express a conjunction condition as `cond1['and'] [cond2]` where `cond1` and `cond2` are the expressions for the two conditions being conjoined.
  
  - **or** represents a disjunction condition. You can express a disjunction condition as `cond1['or'] [cond2]` where `cond1` and `cond2` are the expressions for the two conditions being disjoined.
  
  - **not** represents a negation condition. You can express a negation condition as `cond['not']` where `cond` is the expression for the condition being conjoined.

For example, the following EL returns a list of all portals (to which the current user has access) that are created by users having names starting with `monty` and on which self subscription is enabled:

```
#{spaceContext.spacesQuery.unionOf.ALL_QUERIABLE.where[wCond['sp.createdBy']
['like'] ['monty%'] ['and'] [wCond['sp.selfSubEnabled'] ['is'] ['null'] ['or']
[wCond['sp.selfSubEnabled'] ['='] ['N'] ['not']]].listSpaces}
```

Now imagine that you have used the following syntax to specify **SORT** criteria on a portal query:

```
#{spaceContext.spacesQuery.unionOf['ALL_QUERIABLE'].sortCriteria
['sp.discoverabledesc', 'sp.keywords'].listSpaces}
```

Going forward, you must use the following syntax to specify **SORT** criteria on a portal query:

```
#{spaceContext.spacesQuery.unionOf['ALL_QUERIABLE'].sort['sp.discoverable']
['desc'].sort['sp.keywords']['asc'].listSpaces}
```

Where:

- **sp** represents the JPA entity for a portal, `WcSpaceHeader` (see previous bullet list for fields defined in `WcSpaceHeader`).

- **desc** and **asc** define sort order, descending and ascending, respectively.

For example, the following query returns a list of portals to which the user has access and sorts the result set on the fields `discoverable` and `lastUpdateDate`.

```
#{spaceContext.spacesQuery.unionOf['ALL_QUERIABLE'].sort['sp.discoverable']
['asc'].sort['sp.lastUpdateDate']['desc'].listSpaces}
```

Oracle **strongly** recommends that you convert all existing freeform JPQL **WHERE** and **SORT** clauses to the new syntax.
If for any reason you must retain the existing syntax, your WebCenter Portal administrator must explicitly enable support for freeform JPQL in the webcenter-config.xml.xml file by setting the following global flag:

```java
oracle.webcenter.spaces.query.DIRECT_JPQL_CLAUSE_SUPPORTED
```

**Note:** The webcenter-config.xml.xml file is the customization document for webcenter-config.xml.

Set this flag to `true` for continued support of legacy freeform `WHERE` and `SORT` clauses. When this option is set to `true`, WebCenter Portal log files will contain warnings.

To set this flag in webcenter-config.xml.xml:

1. Export the latest webcenter-config.xml.xml from MDS.
   
   For example:
   ```bash
   exportMetadata(application='webcenter', server='WC_Spaces',
   toLocation='/tmp/mydata', docs='/oracle/webcenter/webcenterapp/metadata/mdssys/cust/site/webcenter/webcenter-config.xml.xml')
   ```

2. Open the file in a text editor and modify settings, as required.
   
   For example:
   ```xml
   <mds:insert parent="webcenter(xmlns(webcenter=http://xmlns.oracle.com/webcenter/webcenterapp))/webcenter:custom-attributes" position="last">
   <attribute name="oracle.webcenter.spaces.query.DIRECT_JPQL_CLAUSE_SUPPORTED" xmlns="http://xmlns.oracle.com/webcenter/webcenterapp">
     <description/>
     <type>java.lang.String</type>
     <value>true</value>
     <visible/>
   </attribute>
   </mds:insert>
   ```

3. Save and close webcenter-config.xml.xml.

4. Import the updated webcenter-config.xml.xml file to WebCenter Portal.
   
   For example:
   ```bash
   importMetadata(application='webcenter', server='WC_Spaces',
   fromLocation='/tmp/mydata',
   docs='/oracle/webcenter/webcenterapp/metadata/mdssys/cust/site/webcenter/webcenter-config.xml.xml')
   ```

**G.14.2 Seeded Expressions in the Expression Editor**

In the Expression Editor, when you select the option **Choose a value** you can populate the text box in the dialog with a seeded EL expression. First, select an expression category, and then select the type of information you want the expression to return. Your selection's associated EL appears in the text box at the bottom of the dialog (Figure G–7).
This section is broken into subsections for each category. Each subsection lists and describes the ELs associated with that category.

This section includes the following subsections:

- Section G.14.2.1, "Application Info Seeded ELs"
- Section G.14.2.2, "Asset Info Seeded ELs"
- Section G.14.2.3, "Page Info Seeded ELs"
- Section G.14.2.4, "Page Parameter Seeded ELs"
- Section G.14.2.5, "Portal Info Seeded ELs"
- Section G.14.2.6, "Portal Page Info Seeded Paths"
- Section G.14.2.7, "System Seeded ELs"
- Section G.14.2.8, "User Info Seeded ELs"
- Section G.14.2.9, "WebCenter Events Seeded ELs"

G.14.2.1 Application Info Seeded ELs

Application Info seeded EL expressions return values related to WebCenter Portal. For example, the expression #{WCAppContext.application.applicationConfig.title}, returns the application name specified on the General page in WebCenter Portal administration.

See Also: For additional EL expressions related to the application, see Section G.14.3, "EL Expressions Relevant to WebCenter Portal Information."

Table G–13 lists the seeded EL expressions available under the category Application Info and provides an example of the types of values they return.

<table>
<thead>
<tr>
<th>Option</th>
<th>Expression</th>
<th>Example of Returned Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current Scope</td>
<td>#{serviceCtx.scope}</td>
<td>Scope[name=standards, guid=sa78185d3_9d65_49ab_ac1d_4809380d0b30]</td>
</tr>
<tr>
<td>Current Page Template Scope</td>
<td>#{WCAppContext.currentScope}</td>
<td>Scope[name=standards, guid=sa78185d3_9d65_49ab_ac1d_4809380d0b30]</td>
</tr>
<tr>
<td>Previously Set Page Template</td>
<td>#{WCAppContext.previouslySetPageTemplate}</td>
<td>Fusion Side Navigation Template</td>
</tr>
</tbody>
</table>
For example, consider that you want a link to appear in all pages of a portal that lets you copy and share the current page's URL. You can add an HTML Markup item to a page template, and add the EL that returns page URL.

These steps demonstrate how to use ELs to retrieve page URLs:

1. Open a custom page template in Edit mode.
2. In Design view, in the resource catalog, click Web Development.
3. Click Add next to HTML Markup.
4. Click the Edit icon for the newly added HTML Markup.
5. In the Component Properties dialog, on the Display Options tab, in the Value field, enter the following EL:

```
<a href="javascript:
var returnvalue = prompt( '#{pageDocBean.title} URL','#{WCAppContext.applicationURL}/faces#{WCAppContext.currentWebCenterURI}')
}">Share Link</a>
```
6. Click OK.
7. Apply the page template to your portal.

### G.14.2.2 Asset Info Seeded ELs

Asset Info seeded EL expressions return values related to WebCenter Portal resources, such as templates, styles, resource catalogs, and so on. For example, the expression 

```
#{srmContext.id['ResourceID'].singleResult}
```

returns the asset with the specified GUID.
**See Also:** For additional asset-related EL expressions, see Section G.14.4, "EL Expressions Relevant to Assets."

Table G–14 lists the seeded EL expressions available under the category Asset Info and provides an example of the types of values each returns.

### Table G–14 Seeded EL Expressions Under Asset Info

<table>
<thead>
<tr>
<th>Option</th>
<th>Expression</th>
<th>Example of Returned Value</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Display Name</th>
<th>#{srmContext[wCond['displayName']['like']['DisplayName']].singleResult}</th>
<th>See Asset ID for an example of a returned value.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>#{srmContext[wCond['description']['like']['Description']].singleResult}</td>
<td>See Asset ID for an example of a returned value.</td>
</tr>
<tr>
<td>Creation Date</td>
<td>#{srmContext[wCond['createdDate']['like']['CreationDate']].singleResult}</td>
<td>See Asset ID for an example of a returned value.</td>
</tr>
</tbody>
</table>
EL Expressions Relevant Only to WebCenter Portal

G.14.2.3 Page Info Seeded ELs

Page Info seeded EL expressions return values related to WebCenter Portal pages. For example, the expression `#{pageDocBean.createdBy}`, returns the user name of the person who created the current page.

Table G–15 lists the seeded EL expressions available under the category Page Info and provides an example of the types of values each returns.

### Table G–15 Seeded EL Expressions Under Page Info

<table>
<thead>
<tr>
<th>Option</th>
<th>Expression</th>
<th>Example of Returned Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td><code>#{pageDocBean.title}</code></td>
<td>Welcome Page (BRP)</td>
</tr>
<tr>
<td>Created By</td>
<td><code>#{pageDocBean.createdBy}</code></td>
<td>j.doe</td>
</tr>
<tr>
<td>Create Date String</td>
<td><code>#{pageDocBean.createDateString}</code></td>
<td>2010-11-17T13:27:52</td>
</tr>
<tr>
<td>Last Updated By</td>
<td><code>#{pageDocBean.lastUpdatedBy}</code></td>
<td>j.smith</td>
</tr>
<tr>
<td>Last Update Date String</td>
<td><code>#{pageDocBean.lastUpdateDateString}</code></td>
<td>2010-11-17T13:27:52</td>
</tr>
<tr>
<td>Page Path</td>
<td><code>#{pageDocBean.pagePath}</code></td>
<td>/oracle/webcenter/page/scopedMD/GUID/businessRolePages/Page3.jspx</td>
</tr>
<tr>
<td>Name</td>
<td><code>#{pageDocBean.name}</code></td>
<td>Page3.jspx</td>
</tr>
<tr>
<td>UI CSS Style</td>
<td><code>#{pageDocBean.UICSSStyle}</code></td>
<td>WCSchemeCustom</td>
</tr>
<tr>
<td>UI Scheme BG Image</td>
<td><code>#{pageDocBean.UISchemeBGImage}</code></td>
<td><code>(facesContext.externalContext.requestContextPath/adf/oracle/webcenter/page/images/clouds.png</code></td>
</tr>
<tr>
<td>UI Scheme BG Color String</td>
<td><code>#{pageDocBean.UISchemeBGColorString}</code></td>
<td>null</td>
</tr>
<tr>
<td>Page Permission</td>
<td><code>#{pageDocBean.pagePermission}</code></td>
<td>oracle.webcenter.page.model.security.CustomPagePermission</td>
</tr>
<tr>
<td>Page Security Target</td>
<td><code>#{pageDocBean.pageSecurityTarget}</code></td>
<td>oracle_webcenter_page ScopedMD_sa78185dd3_9d65_49ab_acld_4809380db30_C0IHomePageDef</td>
</tr>
</tbody>
</table>

G.14.2.4 Page Parameter Seeded ELs

Page Parameter seeded EL expressions return values related to WebCenter Portal page parameters. The editor lists all parameters that are defined for the current page. If no parameters are defined, the Page Parameter category is not available.

For example, the seeded page style **Three Column** provides five seeded page parameters (see Table G–16). The `width` parameters describe the percentage of total width allotted to each column. The `show` parameters specify whether a page header
and footer are rendered. Of the seeded page styles, Blank, Home Page, Left Narrow, and Right Narrow all provide the same seeded page parameters (see Table G–16). Additionally, any page may have its own custom page parameters. Whether seeded or custom, available page parameters are exposed in the Expression Editor under the Page Parameter category.

**See Also:** For more information about creating and using page parameters, see the "Wiring Components and Page Parameters" section in Building Portals with Oracle WebCenter Portal.

Table G–16 lists the EL expressions available under the Page Parameter category for pages based on the styles Three Column, Blank, Home Page, Left Narrow, and Right Narrow. It provides an example of the types of values each returns.

<table>
<thead>
<tr>
<th>Option</th>
<th>Expression</th>
<th>Example of Returned Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>leftWidth</td>
<td>#{bindings.leftWidth}</td>
<td>25%</td>
</tr>
<tr>
<td>centerWidth</td>
<td>#{bindings.centerWidth}</td>
<td>50%</td>
</tr>
<tr>
<td>rightWidth</td>
<td>#{bindings.rightWidth}</td>
<td>25%</td>
</tr>
<tr>
<td>showHeader</td>
<td>#{bindings.showHeader}</td>
<td>false</td>
</tr>
<tr>
<td>showFooter</td>
<td>#{bindings.showFooter}</td>
<td>false</td>
</tr>
</tbody>
</table>

**G.14.2.5 Portal Info Seeded ELs**

Portal Info seeded EL expressions return values related to portals (other than the Home portal). For example, the expression

`#{spaceContext.space['portalName'].children}`

returns a list of the subportals under the named parent portal.

**See Also:** For additional portal-related EL expressions, see Section G.14.5, "EL Expressions Relevant to Specific Portals."

Table G–17 lists the seeded EL expressions available under the category Portal Info and provides an example of the types of values each returns.

**Note:** Use the EL expressions listed in Table G–17 only in the context of portals other than the Home portal.

<table>
<thead>
<tr>
<th>Option</th>
<th>Expression</th>
<th>Example of Returned Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>NamedPortal</td>
<td>#{spaceContext.space['portalName']}</td>
<td>oracle.webcenter.spaces.internal.model.SpaceImpl@15fc839</td>
</tr>
<tr>
<td>CurrentPortal</td>
<td>#{spaceContext.currentSpace}</td>
<td>oracle.webcenter.spaces.internal.model.SpaceImpl@d5e197</td>
</tr>
<tr>
<td>CurrentPortal Name</td>
<td>#{spaceContext.currentSpaceName}</td>
<td>standards</td>
</tr>
<tr>
<td>NamedPortal Metadata Path</td>
<td>#{spaceContext.space['portalName'].metadataPath}</td>
<td>/oracle/webcenter/space/metadata/spaces/portalName/space.xml</td>
</tr>
</tbody>
</table>
| CurrentPortal Metadata Path | #{spaceContext.currentSpace.metadataPath}                  | /oracle/webcenter/space/metadata/spaces/sta

---

**Table G–16** Seeded EL Expressions Under Page Parameter

**Table G–17** Seeded EL Expressions Under Portal Info
<table>
<thead>
<tr>
<th>Option</th>
<th>Expression</th>
<th>Example of Returned Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>NamedPortal Metadata</td>
<td>#{spaceContext.space['portalName'].metadata}</td>
<td>oracle.webcenter.spaces.beans.SpaceMetadataImpl@1d7320a</td>
</tr>
<tr>
<td>CurrentPortal Metadata</td>
<td>#{spaceContext.currentSpace.metadata}</td>
<td>oracle.webcenter.spaces.beans.SpaceMetadataImpl@1096c64</td>
</tr>
<tr>
<td>NamedPortal Metadata DisplayName</td>
<td>#{spaceContext.space['portalName'].metadata.displayName}</td>
<td>Standards</td>
</tr>
<tr>
<td>CurrentPortal Metadata DisplayName</td>
<td>#{spaceContext.currentSpace.GSMetadata.displayName}</td>
<td>Sales</td>
</tr>
<tr>
<td>NamedPortal Metadata Icon</td>
<td>#{WCPrepareImageURL[spaceContext.space['portalName'].metadata.icon]['']}</td>
<td>/webcenter-spaces-resources/oracle/webcenter/spaces/metadata/spaceTemplate/PortalSite/images/portal_icon.png</td>
</tr>
<tr>
<td>CurrentPortal Metadata Icon</td>
<td>#{WCPrepareImageURL[spaceContext.currentSpace.metadata.icon]['']}</td>
<td>/webcenter-spaces-resources/oracle/webcenter/spaces/metadata/Test_Space/images/wc_prj_icon.png</td>
</tr>
<tr>
<td>NamedPortal Metadata Description</td>
<td>#{spaceContext.space['portalName'].metadata.description}</td>
<td>Central point for all standards, templates, and stylesheets for company collateral</td>
</tr>
<tr>
<td>CurrentPortal Metadata Description</td>
<td>#{spaceContext.currentSpace.GSMetadata.description}</td>
<td>Sales Force Team Site</td>
</tr>
<tr>
<td>NamedPortal Metadata CreationDate</td>
<td>#{spaceContext.space['portalName'].metadata.creationDate}</td>
<td>java.util.GregorianCalendar[time=1290103432242, areFieldsSet=true, areAllFieldsSet=false, lenient=false, zoneId=, offset=-28800000, dstSavings=3600000, useDaylight=false, startYear=0, startTime=0, startDayOfWeek=0, startMonth=0, startDay=0, startMode=0, endMonth=0, endDay=0, endDayOfWeek=0, endTime=0, endTimeMode=0, firstDayOfWeek=1, minimalDaysInFirstWeek=1, ERA=1, YEAR=2010, MONTH=10, WEEK_OF_YEAR=?, WEEK_OF_MONTH=?, DAY_OF_MONTH=18, DAY_OF_YEAR=?, DAY_OF_WEEK=?, AM_PM=0, HOUR=10, HOUR_OF_DAY=10, MINUTE=3, SECOND=52, MILLISECOND=242, ZONE_OFFSET=?, DST_OFFSET=?]</td>
</tr>
</tbody>
</table>
### Table G–17 (Cont.) Seeded EL Expressions Under Portal Info

<table>
<thead>
<tr>
<th>Option</th>
<th>Expression</th>
<th>Example of Returned Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CurrentPortal Metadata</td>
<td>#{spaceContext.currentSpace.metadata.creationDate}</td>
<td>java.util.GregorianCalendar{time=1290103432242,areFieldsSet=true,areAllFieldsSet=false,lenient=false,zone=javutil.SimpleTimeZone[id=,offset=-28800000,dstSavings=3600000,useDaylight=false,startYear=0,startMode=0,startMonth=0,startDay=0,startDayOfWeek=0,startTime=0,startTimeMode=0,endMode=0,endMonth=0,endDay=0,endDayOfWeek=0,endTime=0,endTimeMode=0,firstDayOfWeek=1,minimalDaysInFirstWeek=1,ERA=1,YEAR=2010,MONTH=10,WEEK_OF_YEAR=?,WEEK_OF_MONTH=?,DAY_OF_MONTH=18,DAY_OF_YEAR=?,DAY_OF_WEEK=?,DAY_OF_WEEK_IN_MONTH=?,AM_PM=0,HOUR=10,HOUR_OF_DAY=10,MINUTE=3,SECOND=52,MILLISECOND=242,ZONE_OFFSET=?,DST_OFFSET=?}</td>
</tr>
<tr>
<td>CreatedBy</td>
<td>#{spaceContext.space['portalName'].metadata.createdBy}</td>
<td>j.doe</td>
</tr>
<tr>
<td>Keywords</td>
<td>#{spaceContext.space['portalName'].metadata.keywords}</td>
<td>standards templates style sheets collateral</td>
</tr>
<tr>
<td>Offline</td>
<td>#{spaceContext.space['portalName'].metadata.offline}</td>
<td>false or true, depending on whether the named portal is offline</td>
</tr>
<tr>
<td>Offline</td>
<td>#{spaceContext.currentSpace.metadata.offline}</td>
<td>false or true, depending on whether the current portal is offline</td>
</tr>
<tr>
<td>Closed</td>
<td>#{spaceContext.space['portalName'].metadata.closed}</td>
<td>false or true, depending on whether the named portal is closed</td>
</tr>
<tr>
<td>Closed</td>
<td>#{spaceContext.currentSpace.metadata.closed}</td>
<td>false or true, depending on whether the current portal is closed</td>
</tr>
<tr>
<td>SelfRegistration</td>
<td>#{spaceContext.space['portalName'].metadata.selfRegistration}</td>
<td>false or true, depending on whether self registration is enabled for the named portal</td>
</tr>
<tr>
<td>SelfRegistration</td>
<td>#{spaceContext.currentSpace.metadata.selfRegistration}</td>
<td>false or true, depending on whether self registration is enabled for the current portal</td>
</tr>
<tr>
<td>Discoverable</td>
<td>#{spaceContext.space['portalName'].metadata.discoverable}</td>
<td>false or true, depending on whether the named portal is discoverable</td>
</tr>
<tr>
<td>Discoverable</td>
<td>#{spaceContext.currentSpace.metadata.discoverable}</td>
<td>false or true, depending on whether the current portal is discoverable</td>
</tr>
<tr>
<td>PublishRSS</td>
<td>#{spaceContext.space['portalName'].metadata.publishRSS}</td>
<td>false or true, depending on whether the named portal can be published in an RSS reader</td>
</tr>
<tr>
<td>PublishRSS</td>
<td>#{spaceContext.currentSpace.metadata.publishRSS}</td>
<td>false or true, depending on whether the current portal can be published in an RSS reader</td>
</tr>
<tr>
<td>Distribution List</td>
<td>#{spaceContext.space['portalName'].distributionList}</td>
<td><a href="mailto:standardsGroup@myCompany.com">standardsGroup@myCompany.com</a></td>
</tr>
</tbody>
</table>

Expression Language Expressions  G-39
Table G–17  (Cont.) Seeded EL Expressions Under Portal Info

<table>
<thead>
<tr>
<th>Option</th>
<th>Expression</th>
<th>Example of Returned Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CurrentPortal Distribution</td>
<td>#{spaceContext.currentSpace.distributionList}</td>
<td><a href="mailto:salesTeam@myCompany.com">salesTeam@myCompany.com</a></td>
</tr>
<tr>
<td>List</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NamedPortal Metadata</td>
<td>#{spaceContext.space['portalName'].metadata.customAttributes['attributeName']}</td>
<td>Returns the value provided for the named custom attribute</td>
</tr>
<tr>
<td>CustomAttributes</td>
<td></td>
<td>Note: If you use this EL to provide a value for a field in WebCenter Portal Administration, clicking the Test button may return a blank value. However, the value will resolve correctly in the running portal.</td>
</tr>
<tr>
<td>CurrentPortal Metadata</td>
<td>#{spaceContext.currentSpace.metadata.customAttributes['attributeName']}</td>
<td>Returns the value provided for the named custom attribute</td>
</tr>
<tr>
<td>CustomAttributes</td>
<td></td>
<td>Note: If you use this EL to provide a value for a field in WebCenter Portal Administration, clicking the Test button may return a blank value. However, the value will resolve correctly in the running portal.</td>
</tr>
<tr>
<td>Logo</td>
<td>#{WCPrepareImageURL[spaceContext.currentSpace.metadata.logo]['']}</td>
<td>/webcenter-spaces-resources/oracle/webcenter/space/metadata/space/template/PortalSite/images/portal_logo.png</td>
</tr>
<tr>
<td>CurrentPortal Role -</td>
<td>#{WCSecurityContext.userInScopedRole('Moderator')}</td>
<td>false or true, depending whether any users in the current scope are assigned the role Moderator</td>
</tr>
<tr>
<td>Moderator</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CurrentPortal Role -</td>
<td>#{WCSecurityContext.userInScopedRole('Participant')}</td>
<td>false or true, depending whether any users in the current scope are assigned the role Participant</td>
</tr>
<tr>
<td>Participant</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CurrentPortal Role -</td>
<td>#{WCSecurityContext.userInScopedRole('Viewer')}</td>
<td>false or true, depending whether any users in the current scope are assigned the role Viewer</td>
</tr>
<tr>
<td>Viewer</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CurrentPortal Role - Custom</td>
<td>#{WCSecurityContext.userInScopedRole('CustomRole')}</td>
<td>false or true, depending on if any users in the current scope are assigned the named custom role</td>
</tr>
<tr>
<td>NamedPortal Children</td>
<td>#{spaceContext.space['portalName'].children}</td>
<td>[oracle.webcenter.spaces.internal.model.SpaceImpl@5705f4, oracle.webcenter.spaces.internal.model.SpaceImpl@147d658]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Note: The braces ([ ]) denote a set.</td>
</tr>
<tr>
<td>CurrentPortal Children</td>
<td>#{spaceContext.currentSpace.children}</td>
<td>[oracle.webcenter.spaces.internal.model.SpaceImpl@5705f4, oracle.webcenter.spaces.internal.model.SpaceImpl@147d658]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Note: The braces ([ ]) denote a set.</td>
</tr>
<tr>
<td>NamedPortal Parent</td>
<td>#{spaceContext.space['portalName'].parent}</td>
<td>oracle.webcenter.spaces.internal.model.SpaceImpl@15fc839</td>
</tr>
<tr>
<td>CurrentPortal Parent</td>
<td>#{spaceContext.currentSpace.parent}</td>
<td>oracle.webcenter.spaces.internal.model.SpaceImpl@15fc839</td>
</tr>
<tr>
<td>NamedPortal Security</td>
<td>#{spaceContext.space['portalName'].securityParent}</td>
<td>oracle.webcenter.spaces.internal.model.SpaceImpl@1392793</td>
</tr>
<tr>
<td>Parent</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table G–17 (Cont.) Seeded EL Expressions Under Portal Info

<table>
<thead>
<tr>
<th>Option</th>
<th>Expression</th>
<th>Example of Returned Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CurrentPortal Security Parent</td>
<td>#{spaceContext.currentSpace.securityParent}</td>
<td>oracle.webcenter.spaces.internal.model.SpaceImpl@1392793</td>
</tr>
<tr>
<td>All visible root level spaces</td>
<td>#{spaceContext.spacesQuery.unionOf('ALL_QUERIABLE,DISCOVERABLE,PUBLIC_ACCESSIBLE,ALLUSER_ACCESSIBLE,USER_JOINED,USER_MODERATED').shape['ROOT_LEVEL']}</td>
<td>oracle.webcenter.spaces.query.SpaceandWhereParameters@ae34c8</td>
</tr>
<tr>
<td>All children and descendant portals for specified Parent Portal GUID</td>
<td>#{spaceContext.spacesQuery.parentSpaceGuid['portalGuid'].shape['RECURSIVE_FLATTENED']}</td>
<td>oracle.webcenter.spaces.query.SpaceandWhereParameters@4da22f</td>
</tr>
<tr>
<td>Immediate Sub Portals for specified Parent Portal name</td>
<td>#{spaceContext.spacesQuery.parentSpaceName['portalGuid'].shape['ROOT_LEVEL']}</td>
<td>oracle.webcenter.spaces.query.SpaceandWhereParameters@1089971</td>
</tr>
<tr>
<td>List of Portals matching given Portal name</td>
<td>#{spaceContext.spacesQuery.parentSpaceName['portal1', 'portal2', 'portal3']}</td>
<td>oracle.webcenter.spaces.query.SpaceandWhereParameters@68e467</td>
</tr>
<tr>
<td>List of Portals matching given Portal GUID</td>
<td>#{spaceContext.spacesQuery.parentSpaceGuid['portalGuid1', 'portalGuid2']}</td>
<td>oracle.webcenter.spaces.query.SpaceandWhereParameters@1c56009</td>
</tr>
<tr>
<td>List of Portals sorted based on criteria</td>
<td>#{spaceContext.spacesQuery.unionOf.ALL_QUERIABLE.sort['sp.lastUpdateDate']['desc']}</td>
<td>oracle.webcenter.spaces.query.SpaceandWhereParameters@6c3def</td>
</tr>
<tr>
<td>List of Portals based on search pattern</td>
<td>#{spaceContext.spacesQuery.unionOf.ALL_QUERIABLE.search['searchKeyword']}</td>
<td>oracle.webcenter.spaces.query.SpaceandWhereParameters@19b22cf3</td>
</tr>
<tr>
<td>List of Portals with limited page size</td>
<td>#{spaceContext.spacesQuery.unionOf.USER_JOINED.itemsPerPage[n].pageNumber[n]}</td>
<td>oracle.webcenter.spaces.query.SpaceandWhereParameters@4f8818</td>
</tr>
<tr>
<td>List of portals matching the where clause</td>
<td>#{spaceContext.spacesQuery.unionOf.USER_JOINED.where[wCond['sp.createdBy']['=', 'userName']]}</td>
<td>oracle.webcenter.spaces.query.SpaceandWhereParameters@1ba816d</td>
</tr>
</tbody>
</table>

### G.14.2.6 Portal Page Info Seeded Paths

Rather than EL expressions, the selections under Portal Page Info are directory paths to different application pages. Use them, for example, to provide navigation information to a specified page.

Table G–18 lists the seeded directory paths available under the category Portal Page Info and provides an example of the types of values each returns. The number of values in the dropdown list depends on the template used and the pages created by users.
Table G–18  Seeded Paths Under Portal Page Info

<table>
<thead>
<tr>
<th>Option</th>
<th>Path</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>PortalSiteHome.jspx</td>
<td>/faces/oracle/webcenter/page/scoPedMD/se2dff4fb_203a4f4c_b56d_c7cf5ad1c04/PortalSiteHome.jspx</td>
<td>Path to the Home page of the current portal</td>
</tr>
<tr>
<td>GroupSpaceDocLibMainView.jspx</td>
<td>/faces/oracle/webcenter/page/scoPedMD/s8bba98ff_4cbb_40b8_beee_296c916a23ed/businessRolePages/GroupSpaceDocLibMainView.jspx</td>
<td>Path to the Documents page in the current portal</td>
</tr>
<tr>
<td>ForumMainView.jspx</td>
<td>/faces/oracle/webcenter/page/scoPedMD/s8bba98ff_4cbb_40b8_beee_296c916a23ed/businessRolePages/ForumMainView.jspx</td>
<td>Path to Discussions page in the current portal</td>
</tr>
<tr>
<td>AnnouncementsMainView.jspx</td>
<td>/faces/oracle/webcenter/page/scoPedMD/s8bba98ff_4cbb_40b8_beee_296c916a23ed/businessRolePages/AnnouncementsMainView.jspx</td>
<td>Path to Announcements page in current portal</td>
</tr>
<tr>
<td>ListsMainView.jspx</td>
<td>/faces/oracle/webcenter/page/scoPedMD/s8bba98ff_4cbb_40b8_beee_296c916a23ed/businessRolePages/ListsMainView.jspx</td>
<td>Path to Lists page in current portal</td>
</tr>
<tr>
<td>EventsMainView.jspx</td>
<td>/faces/oracle/webcenter/page/scoPedMD/s8bba98ff_4cbb_40b8_beee_296c916a23ed/businessRolePages/EventsMainView.jspx</td>
<td>Path to Events page in current portal</td>
</tr>
<tr>
<td>GroupSpaceActivityStreamMainView.jspx</td>
<td>/faces/oracle/webcenter/page/scoPedMD/s8bba98ff_4cbb_40b8_beee_296c916a23ed/businessRolePages/GroupSpaceActivityStreamMainView.jspx</td>
<td>Path to Activity Stream page in current portal</td>
</tr>
</tbody>
</table>

G.14.2.7  System Seeded ELs

System seeded EL expressions return values related to system settings, such as the user name of the current user or the current user’s specified locale. For example, the expression #{securityContext.userName}, returns the name of the currently logged in user. For example, if user j.doe is viewing a page where this expression is used, the returned value in j.doe’s view is j.doe. If user j.smith is looking at the same page, the value returned in j.smith’s view is j.smith.

See Also: For additional utilitarian EL expressions, see Section G.14.8, “Utilitarian EL Expressions.”

Table G–19 lists the seeded EL expressions available under the category System and provides an example of the types of values each returns.
G.14.2.8 User Info Seeded ELs

User Info seeded EL expressions return values related to a given user's Profile. For example, the expression

\#{webCenterProfile[securityContext.userName].description}

returns the content of the About Me attribute of the current user's Profile.

See Also: For additional user-related expressions, see Section G.10, "ELs Related to People Connections."

Table G–20 lists the seeded EL expressions available under the category User Info and provides an example of the types of values each returns.

<table>
<thead>
<tr>
<th>Option</th>
<th>Expression</th>
<th>Example of Returned Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>About Me</td>
<td>#{webCenterProfile[securityContext.userName].description}</td>
<td>Returns the content of the About Me attribute of the current user's Profile</td>
</tr>
<tr>
<td>Username</td>
<td>#{webCenterProfile[securityContext.userName].userName}</td>
<td>The user name specified in the current user's Profile</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Compare this with the current user's display name.</td>
</tr>
<tr>
<td>First Name</td>
<td>#{webCenterProfile[securityContext.userName].firstName}</td>
<td>The current user's first name</td>
</tr>
<tr>
<td>Middle Name</td>
<td>#{webCenterProfile[securityContext.userName].middleName}</td>
<td>The current user's middle name</td>
</tr>
<tr>
<td>Last Name</td>
<td>#{webCenterProfile[securityContext.userName].lastName}</td>
<td>The current user's last name</td>
</tr>
<tr>
<td>Initials used</td>
<td>#{webCenterProfile[securityContext.userName].initials}</td>
<td>The current user's initials</td>
</tr>
<tr>
<td>Display Name</td>
<td>#{webCenterProfile[securityContext.userName].displayName}</td>
<td>The current user's display name</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Compare this with the current user's user name.</td>
</tr>
<tr>
<td>Email</td>
<td>#{webCenterProfile[securityContext.userName].businessEmail}</td>
<td>The current user's business email address</td>
</tr>
<tr>
<td>Department</td>
<td>#{webCenterProfile[securityContext.userName].department}</td>
<td>The current user's department</td>
</tr>
<tr>
<td>Job Title/Designation</td>
<td>#{webCenterProfile[securityContext.userName].title}</td>
<td>The current user's job title or designation</td>
</tr>
</tbody>
</table>
### Table G–20 (Cont.) Seeded EL Expressions Under User Info

<table>
<thead>
<tr>
<th>Option</th>
<th>Expression</th>
<th>Example of Returned Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manager</td>
<td>#{webCenterProfile[securityContext.userName].managerDisplayName}</td>
<td>The current user’s manager’s display name</td>
</tr>
<tr>
<td>Time zone</td>
<td>#{webCenterProfile[securityContext.userName].timeZone}</td>
<td>The time zone associated with the current user</td>
</tr>
<tr>
<td>Employee type</td>
<td>#{webCenterProfile[securityContext.userName].employeeType}</td>
<td>The current user’s employee type</td>
</tr>
<tr>
<td>Employee Number</td>
<td>#{webCenterProfile[securityContext.userName].employeeNumber}</td>
<td>The current user’s employee number</td>
</tr>
<tr>
<td>Preferred Language Code</td>
<td>#{webCenterProfile[securityContext.userName].preferredLanguage}</td>
<td>The current user’s preferred language</td>
</tr>
<tr>
<td>Organization</td>
<td>#{webCenterProfile[securityContext.userName].organization}</td>
<td>The company organization to which the current user belongs</td>
</tr>
<tr>
<td>Organizational Unit</td>
<td>#{webCenterProfile[securityContext.userName].organizationalUnit}</td>
<td>The unit of the company organization to which the current user belongs</td>
</tr>
<tr>
<td>Expertise</td>
<td>#{webCenterProfile[securityContext.userName].expertise}</td>
<td>The current user’s expertise, for example, Application, system, and database administration</td>
</tr>
<tr>
<td>Business Fax</td>
<td>#{webCenterProfile[securityContext.userName].businessFax}</td>
<td>The current user’s business fax number</td>
</tr>
<tr>
<td>Business Mobile Number</td>
<td>#{webCenterProfile[securityContext.userName].businessMobile}</td>
<td>The current user’s business mobile or cell phone number</td>
</tr>
<tr>
<td>Business Phone</td>
<td>#{webCenterProfile[securityContext.userName].businessPhone}</td>
<td>The current user’s business phone number</td>
</tr>
<tr>
<td>Business Pager Number</td>
<td>#{webCenterProfile[securityContext.userName].businessPager}</td>
<td>The current user’s business pager number</td>
</tr>
<tr>
<td>Business Street</td>
<td>#{webCenterProfile[securityContext.userName].businessStreet}</td>
<td>The street address of the current user’s business office</td>
</tr>
<tr>
<td>Business City</td>
<td>#{webCenterProfile[securityContext.userName].businessCity}</td>
<td>The city where the current user’s business office is located</td>
</tr>
<tr>
<td>Business State</td>
<td>#{webCenterProfile[securityContext.userName].businessState}</td>
<td>The state where the current user’s business office is located</td>
</tr>
<tr>
<td>Business PO Box</td>
<td>#{webCenterProfile[securityContext.userName].businessPOBox}</td>
<td>The current user’s business Post Office Box number</td>
</tr>
<tr>
<td>Business Postal Code</td>
<td>#{webCenterProfile[securityContext.userName].businessPostalCode}</td>
<td>The postal or ZIP code of the current user’s business office</td>
</tr>
<tr>
<td>Business Country</td>
<td>#{webCenterProfile[securityContext.userName].businessCountry}</td>
<td>The country where the current user’s business office is located</td>
</tr>
<tr>
<td>Home Address</td>
<td>#{webCenterProfile[securityContext.userName].homeAddress}</td>
<td>The current user’s home address</td>
</tr>
<tr>
<td>Home phone</td>
<td>#{webCenterProfile[securityContext.userName].homePhone}</td>
<td>The current user’s home phone number</td>
</tr>
<tr>
<td>Date of Birth</td>
<td>#{webCenterProfile[securityContext.userName].dateOfBirth}</td>
<td>The current user’s date of birth</td>
</tr>
<tr>
<td>Maiden Name</td>
<td>#{webCenterProfile[securityContext.userName].maidenName}</td>
<td>The current user’s maiden name</td>
</tr>
</tbody>
</table>
WebCenter Events seeded EL expressions return values related to document-related events, such as the name of the selected document, the document’s creator, the date the document was last modified, and so on. For example, the expression 
\#{wcEventContext.events.WebCenterResourceSelected.name}
returns the name of the currently selected document.

Table G–21 lists the seeded EL expressions available under the category WebCenter Events and provides an example of the types of values each returns.

### Table G–21  Seeded EL Expressions Under WebCenter Events

<table>
<thead>
<tr>
<th>Option</th>
<th>Expression</th>
<th>Example of Returned Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selected User Name</td>
<td>#{wcEventContext.events.WebCenterUserSelected.UserName}</td>
<td>The user name of the currently selected user</td>
</tr>
<tr>
<td>Selected Document Name</td>
<td>#{wcEventContext.events.WebCenterResourceSelected.name}</td>
<td>The name of the currently selected document</td>
</tr>
<tr>
<td>Selected Document Creator</td>
<td>#{wcEventContext.events.WebCenterResourceSelected.createdBy}</td>
<td>The name of the user who created the currently selected document</td>
</tr>
<tr>
<td>Selected Document Last Modified By</td>
<td>#{wcEventContext.events.WebCenterResourceSelected.lastModifiedBy}</td>
<td>The date the selected document was last modified</td>
</tr>
</tbody>
</table>

### G.14.3 EL Expressions Relevant to WebCenter Portal Information

Table G–22 lists EL expressions relevant to WebCenter Portal information and describes the types of functionality it provides.
### Table G–22  **EL Expressions Relevant to WebCenter Portal Information**

<table>
<thead>
<tr>
<th>Expression</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{WCAppContext}</td>
<td>An oracle.webcenter.webcenterapp.context.WCAClass object that provides an access point in the current web request for all WebCenter Portal-related information.</td>
</tr>
</tbody>
</table>
| #{WCAppContext.currentWebCenterURI}             | Returns a URL representing the current web request with bookmarkable WebCenter Portal URL parameters of the request appended to the end (parameters are not necessarily in a fixed order).  
Example:  
Use the expressions in this section to query for assets. Querying for an asset 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']['like']['value']`, where `property` is the name of the property, for example, `id`, `resourceScope`, and so on, 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 asset. When no matching asset is found, null is returned.
- **resultList**—Returns a list of assets. When no matching assets 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 page template asset found with a display name that contains `myPage`:

```
#{srmContext[wCond['resourceType']['like']['siteTemplate']]['and'][wCond['displayName']['like']['myPage']].singleResult}
```

### Table G–22 (Cont.) EL Expressions Relevant to WebCenter Portal Information

<table>
<thead>
<tr>
<th>Expression</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{WCAppContext.application.applicationConfig.skin}</td>
<td>Returns the name of the default ADF Faces skin family to use for rendering pages in the application (as configured through WebCenter Portal Administration settings). This expression represents only the application-level setting that may not necessarily be used in all web requests. For example, you cannot use it successfully if a user has chosen to override the skin through application Preferences.</td>
</tr>
<tr>
<td>#{requestContext.skinFamily}</td>
<td>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}. This EL can be used both in portals built using WebCenter Portal and Portal Framework applications built with Oracle JDeveloper.</td>
</tr>
</tbody>
</table>

**G.14.4 EL Expressions Relevant to Assets**

Use the expressions in this section to query for assets. Querying for an asset 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']['like']['value']`, where `property` is the name of the property, for example, `id`, `resourceScope`, and so on, 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 asset. When no matching asset is found, null is returned.
- **resultList**—Returns a list of assets. When no matching assets 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 a list of page template assets residing in the directory `resourceDir`, with a description that contains `sampleDesc`:

```el
#{srmContext[wCond['resourceType']['like']['siteTemplate']]['and'][wCond['description']['like']['sampleDesc']]['and'][wCond['contentDirectory']['like']['resourceDir']].resultList}
```

A property of this class includes any attribute of this class. Example properties include: `Id`, `DisplayName`, `iconURI`, `contentDirectory`, and so on.

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:

```el
#{srmContext[wCond['id']['like']['resourceId']].singleResult}
#{srmContext[wCond['id']['like']['spacesContext.currentSpace.uiMetadata.siteTemplateId']].singleResult}
```

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:

```el
#{srmContext[wCond['resourceScope']['like']['scopeName']]['and'][wCond['id']['like']['resourceId']].singleResult}
#{srmContext[wCond['id']['like']['resourceId']]['and'][wCond['resourceScope']['like']['scopeName']].singleResult}
```

Table G–23 lists EL expressions relevant to assets and describes the types of functionality each provides.

<table>
<thead>
<tr>
<th>EL</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{srmContext[wCond['id']['like']['resourceGUID']]}</td>
<td>Returns the asset with the specified ID</td>
</tr>
<tr>
<td>#{srmContext[wCond['displayName']['like']['resourceDisplayName']]}</td>
<td>Returns any assets with the specified display name</td>
</tr>
<tr>
<td>#{srmContext[wCond['displayNameKey']['like']['displayNameKey']]}</td>
<td>Returns any assets with the specified display name key</td>
</tr>
<tr>
<td>#{srmContext[wCond['description']['like']['resourceDescription']].singleResult}</td>
<td>Returns one result of an asset that contains the specified value in its description</td>
</tr>
<tr>
<td></td>
<td>To get multiple results, use resultList in lieu of singleResult.</td>
</tr>
<tr>
<td>#{srmContext[wCond['descriptionKey']['like']['descriptionKey']].singleResult}</td>
<td>Returns one result of an asset with the specified description key</td>
</tr>
<tr>
<td></td>
<td>The description key is the key in the xsrt file for the asset description.</td>
</tr>
<tr>
<td></td>
<td>To get multiple results, use resultList in lieu of singleResult.</td>
</tr>
<tr>
<td>#{srmContext[wCond['createdDate']['like']['resourceCreationDate']]}</td>
<td>Returns any asset with the specified creation date</td>
</tr>
<tr>
<td>#{srmContext[wCond['modifiedBy']['like']['resourceLastModifiedBy']]}</td>
<td>Returns any asset that was last modified by the user with the specified user name</td>
</tr>
<tr>
<td>#{srmContext[wCond['modifiedDate']['like']['resourceLastModifiedDate']]}</td>
<td>Returns any asset that was last modified by the specified date</td>
</tr>
<tr>
<td>EL</td>
<td>Function</td>
</tr>
<tr>
<td>----</td>
<td>----------</td>
</tr>
<tr>
<td>#{srmContext[wCond['resourceScope']['like']['resource Scope']]}</td>
<td>Returns any asset that falls within the specified scope</td>
</tr>
</tbody>
</table>
| #{srmContext[wCond['category']['like']['categoryName']]} | Returns any asset that falls within the specified category. For example:  
  #{srmContext[wCond['category']['like']['siteTemplates']]}  
  List of possible category names includes: siteTemplate, pageStyle, dataPresenter, contentPresenter, resourceCatalog, navigation, taskFlow, dataControl, taskFlowStyle, and skin. |
| #{srmContext[wCond['contentDirectory']['like']['contentDirectory']]} | Returns any asset that is stored within the specified directory |
| #{srmContext[wCond['metadataFile']['like']['metadataFileLocation']]} | Returns asset metadata from the specified metadata file  
  For example, the following expression returns asset metadata from the following file  
  /home/metadata/data.xml:  
  #{srmContext[wCond['metadataFile']['like']['/home/metadata/data.xml']]}  
| #{srmContext[wCond['jspx']['like']['jspxFileLocation']]} | Returns any jspx file in the specified location  
  For example, the following expression returns the page.jspx file:  
  #{srmContext[wCond['jspx']['like']['/home/web/page.jspx']]}  
| #{srmContext[wCond['pageDef']['like']['pageDefinition']]} | Returns any jspx file with the specified page definition |
| #{srmContext[wCond['iconURI']['like']['iconURI']]} | Returns the icon at the specified icon URI |
| #{srmContext[wCond['excludeFrom']['like']['excludeFromScopes']]} | In a larger expression, returns all specified resources except those available in the excluded scopes |
| #{srmContext[wCond['usesCustomSecurity']['like']['useCustomSecurity']]} | In a larger expression, returns any asset the either does or does not use custom security  
  Set usesCustomSecurity to TRUE or FALSE. For example:  
  #{srmContext[wCond['usesCustomSecurity']['like']['TRUE']]}  
| #{srmContext[wCond['seeded']['like']['seeded']]} | In a larger expression, returns any asset that is or is not seeded, depending on the provided value  
  Set seeded to TRUE or FALSE. For example:  
  #{srmContext[wCond['seeded']['like']['TRUE']]}  

Example: Using EL Expressions for Assets

This section provides an example of using an EL expression to display a page template based on a user’s role.

In this example, you will use an EL expression to enable a specific page template to be displayed for moderators so that only they can access the features or links available in that page template.

To display a page template based on a user role:

1. Go to the desired portal’s administration settings.

Table G–23 (Cont.) ELs Relevant to Assets

<table>
<thead>
<tr>
<th>EL</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{srmContext[wCond['visibleType']['like']['visibleType']]}</td>
<td>In a larger expression, returns any asset that is of the specified type of visibility. Set visibleType to TRUE, FALSE, or NEVER to indicate that the asset is never shown. For example, the following expression returns assets that are visible, that is, assets that are set to Show: #{srmContext[wCond['visibleType']['like']['TRUE']]}</td>
</tr>
<tr>
<td>#{srmContext[wCond['visible']['like']['visible']]}</td>
<td>In a larger expression, returns one or multiple assets with visibility set to either TRUE or FALSE.</td>
</tr>
<tr>
<td>#{srmContext[wCond['createdBy']['like']['resourceCreator']]}</td>
<td>In a larger expression, returns any asset created by the specified user.</td>
</tr>
<tr>
<td>#{srmContext[wCond['resourceType']['like']['resourceType']]}</td>
<td>In a larger expression, returns one or multiple assets of the specified type. For example, the following expression searches for an asset of the type SITE_TEMPLATE: #{srmContext[wCond['resourceType']['like']['SITE_TEMPLATE']]} Note that all asset types are listed in Java API Reference for Oracle WebCenter Portal, in the GenericSiteResourceTypes class.</td>
</tr>
<tr>
<td>#{srmContext[wCond['version']['like']['version']]}</td>
<td>In a larger expression, returns one or multiple assets available in the application of the specified version.</td>
</tr>
<tr>
<td>#{srmContext[wCond['resourceScope']['like']['scopeName']]}</td>
<td>In a larger expression, returns one or multiple assets available in the specified scope. For example, the following expression searches for assets in the scope (in this instance, the portal) MyPortal: #{srmContext[wCond['resourceScope']['like']['MyPortal']]} To search in the default scope, that is, the application scope, use defaultScope.</td>
</tr>
<tr>
<td>#{srmContext[wCond['searchType']['like']['searchType']]}</td>
<td>In a larger expression, returns one or multiple assets that contain or equal the values set by other included expressions. Set searchType to CONTAINS or EQUALS.</td>
</tr>
</tbody>
</table>
2. In the **Assets** section, for **Page Template** click the Advanced Edit Options icon, and click **Expression Builder**.

3. In the **Type a value or expression**, enter the following expression:

```java
#{srmContext[wCond['resourceType']['like']['siteTemplate']][wCond['displayName']['like'][WCSecurityContext.userInScopedRole['Moderator'] ? 'Fusion Side Navigation' : 'WebCenter Spaces Top Navigation']].singleResult.id}
```

**Where:**

- **srmContext[wCond['resourceType']['like']['siteTemplate']]**
  Gets a list of all page templates in the scope

- **[wCond['displayName']['like'][WCSecurityContext.userInScopedRole['Moderator'] ? 'Fusion Side Navigation' : 'WebCenter Portal Top Navigation']]**
  Reduces the list down to one depending on whether the current user is a moderator. If the user is a moderator, the Fusion Side Navigation page template is applied. For all other user roles, the portal is rendered using the WebCenter Portal Top Navigation page template.

- **singleResult.id**
  Returns the GUID of the required entry.

4. Click **OK**.

### G.14.5 EL Expressions Relevant to Specific Portals

**Table G–24** lists EL expressions relevant to portals and describes the types of functionality each provides.

---

**Note:** The portal **internal name** is the name specified for **Portal URL** on the **Overview** page of a portal's administration settings. The portal **display name** is the name specified for **Name**. Many of the EL expressions in **Table G–24** call for the portal internal name.
### Table G–24  EL Expressions Relevant to Specific Portals

<table>
<thead>
<tr>
<th>Expression</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{spaceContext}</td>
<td>An oracle.webcenter.spaces.context.SpacesContext object that provides an access point in the current web request for all portal-related information.</td>
</tr>
<tr>
<td></td>
<td>The value of this expression is whatever is returned on invoking the Java API: SpacesContext.getCurrentInstance()</td>
</tr>
<tr>
<td>#{spaceContext.currentSpace}</td>
<td>An oracle.webcenter.spaces.Space object that represents the portal associated with the current web request. If the current web request is in the Home portal context, it returns a value of null.</td>
</tr>
<tr>
<td></td>
<td>The value of this expression is whatever is returned on invoking the Java API: SpacesContext.getCurrentInstance().getCurrentSpace()</td>
</tr>
<tr>
<td>#{spaceContext.currentSpaceName}</td>
<td>The name of the portal associated with the current web request. If the current web request is in the Home portal context, it returns a value of null.</td>
</tr>
<tr>
<td></td>
<td>The value of this expression is whatever is returned on invoking the Java API: SpacesContext.getCurrentInstance().getCurrentSpace()</td>
</tr>
<tr>
<td>#{spaceContext.space['portalName']}.metadataPath</td>
<td>An oracle.webcenter.spaces.Space object that represents the portal that is named portalName or the current portal (currentSpace). For example,</td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadataPath}</td>
<td>#{spaceContext.space['FinanceProject']}.metadataPath evaluates to /oracle/webcenter/space/metadata/spaces/FinanceProject/space.xml</td>
</tr>
<tr>
<td></td>
<td>The value of this expression is whatever is returned in Java on invoking .getSpace(...) on the portal object for the portal.</td>
</tr>
<tr>
<td>#{spaceContext.space['portalName'].metadata}</td>
<td>An oracle.webcenter.spaces.beans.SpaceType bean that carries metadata about the portal that is named portalName or the current portal (currentSpace).</td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata}</td>
<td>The value of this expression is whatever is returned in Java on invoking .getMetadata() on the portal object for the portal passing in the MDSSession of the current ADFContext.</td>
</tr>
<tr>
<td>#{spaceContext.space['portalName'].metadata.displayName}</td>
<td>The display name of the portal that is named portalName or the current portal (currentSpace). For example, if a portal called Web20Portal has the display name web 2.0 Portal, then</td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.metadata.displayName}</td>
<td>#{spaceContext.space['Web20Portal'].metadata.displayName} evaluates to web 2.0 Portal.</td>
</tr>
<tr>
<td>#{spaceContext.space['portalName'].GSMetadata.groupSpaceURI}</td>
<td>The URL of the portal that is named portalName or the current portal (currentSpace).</td>
</tr>
<tr>
<td>#{spaceContext.currentSpace.GSMetadata.groupSpaceURI}</td>
<td></td>
</tr>
<tr>
<td>Expression</td>
<td>Function</td>
</tr>
<tr>
<td>------------</td>
<td>----------</td>
</tr>
<tr>
<td><code>#{WCPrepareImageURL[spaceContext.space['portalName'].metadata.icon]['']}</code></td>
<td>A URL to the icon associated with the portal that is named <code>portalName</code> or the current portal (<code>currentSpace</code>).</td>
</tr>
<tr>
<td><code>#{WCPrepareImageURL[spaceContext.currentSpace.metadata.icon]['']}</code></td>
<td></td>
</tr>
<tr>
<td><code>{spaceContext.space['portalName'].metadata.description}</code></td>
<td>The description of the portal that is named <code>portalName</code> or the current portal (<code>currentSpace</code>). For example, <code>{spaceContext.space['FinanceProject'].metadata.description}</code> evaluates to <em>Conglomeration of all teams involved in financial activities.</em></td>
</tr>
<tr>
<td><code>{spaceContext.currentSpace.metadata.description}</code></td>
<td></td>
</tr>
<tr>
<td><code>{spaceContext.space['portalName'].metadata.creationDate}</code></td>
<td>A <em>java.util.Calendar</em> object representing the date-time on which the portal with specified name <code>portalName</code> or the current portal (<code>currentSpace</code>) was created.</td>
</tr>
<tr>
<td><code>{spaceContext.currentSpace.metadata.creationDate}</code></td>
<td></td>
</tr>
<tr>
<td><code>{spaceContext.space['portalName'].metadata.createdBy}</code></td>
<td>The user-name of the person who created the portal that is named <code>portalName</code> or the current portal (<code>currentSpace</code>).</td>
</tr>
<tr>
<td><code>{spaceContext.currentSpace.metadata.createdBy}</code></td>
<td></td>
</tr>
<tr>
<td><code>{spaceContext.space['portalName'].metadata.keywords}</code></td>
<td>A comma-delimited list of searchable keywords associated with the portal with the name <code>portalName</code> or the current portal (<code>currentSpace</code>). For example, if the portal <code>FinanceProject</code> has the keywords finance, project, money, then <code>{spaceContext.space['FinanceProject'].metadata.keywords}</code> evaluates to <em>finance, project, money.</em></td>
</tr>
<tr>
<td><code>{spaceContext.currentSpace.metadata.keywords}</code></td>
<td></td>
</tr>
<tr>
<td><code>{spaceContext.space['portalName'].metadata.offline}</code></td>
<td>Boolean value that indicates whether the portal that is named <code>portalName</code> or the current portal (<code>currentSpace</code>) is offline.</td>
</tr>
<tr>
<td><code>{spaceContext.currentSpace.metadata.offline}</code></td>
<td></td>
</tr>
<tr>
<td><code>{spaceContext.space['portalName'].metadata.closed}</code></td>
<td>Boolean value that indicates whether the portal that is named <code>portalName</code> or the current portal (<code>currentSpace</code>) is closed.</td>
</tr>
<tr>
<td><code>{spaceContext.currentSpace.metadata.closed}</code></td>
<td></td>
</tr>
<tr>
<td><code>{spaceContext.space['portalName'].metadata.selfRegistration}</code></td>
<td>Boolean value that indicates whether users are allowed to register themselves with the portal that is named <code>portalName</code> or the current portal (<code>currentSpace</code>).</td>
</tr>
<tr>
<td><code>{spaceContext.currentSpace.metadata.selfRegistration}</code></td>
<td></td>
</tr>
<tr>
<td><code>{spaceContext.space['portalName'].metadata.discoverable}</code></td>
<td>Boolean value that indicates whether users can discover the existence of the portal that is named <code>portalName</code> or the current portal (<code>currentSpace</code>) by searching for it or seeing it listed on the My portals page.</td>
</tr>
<tr>
<td><code>{spaceContext.currentSpace.metadata.discoverable}</code></td>
<td></td>
</tr>
<tr>
<td><code>{spaceContext.space['portalName'].metadata.publishRSS}</code></td>
<td>Boolean value indicating whether the portal that is named <code>portalName</code> or the current portal (<code>currentSpace</code>) publishes RSS feeds.</td>
</tr>
<tr>
<td><code>{spaceContext.currentSpace.metadata.publishRSS}</code></td>
<td></td>
</tr>
<tr>
<td><code>{spaceContext.space['portalName'].distributionList}</code></td>
<td>The email address of the mailing list associated with the portal that is named <code>portalName</code> or the current portal (<code>currentSpace</code>).</td>
</tr>
<tr>
<td><code>{spaceContext.currentSpace.distributionList}</code></td>
<td></td>
</tr>
</tbody>
</table>
### EL Expressions Relevant to Specific Portals

<table>
<thead>
<tr>
<th>Expression</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>{spaceContext.space['portalName'].metadata.customAttributes['attributeName']}</code></td>
<td>The value of a specific custom attribute of the name <code>attributeName</code> for the portal that is named <code>portalName</code> or the current portal (currentSpace). For example, if the <code>FinanceProject</code> portal has a custom attribute called <code>stockPrice</code> with a value of 13.9, then <code>{spaceContext.space['FinanceProject'].metadata.customAttributes['stockPrice']}</code> evaluates to 13.9. <strong>Note:</strong> If you use these ELs to provide a value for a field in WebCenter Portal Administration, clicking the Test button may return a blank value. However, the value will resolve correctly in the running portal.</td>
</tr>
<tr>
<td><code>{WCPrepareImageURL[spaceContext.space['portalName'].metadata.logo]['']}</code></td>
<td>Returns the logo URL for the portal named <code>portalName</code> or the current portal (currentSpace).</td>
</tr>
<tr>
<td><code>{WCPrepareImageURL[spaceContext.currentSpace.metadata.logo]['']}</code></td>
<td></td>
</tr>
<tr>
<td><code>{spaceContext.spacesQuery.property['value'].listSpaces}</code></td>
<td>A means of querying a portal using a query parameter in the form of <code>property['value']</code>, where <code>property</code> means the name of the property, for example, <code>unionOf</code>, <code>shape</code>, and so on; and <code>value</code> means the criteria to use in fetching the list of all portals, discoverable portals, and so on. If <code>listSpaces</code> is appended to the expression, the query returns the list of portals of type <code>GSMetadata</code>. (For more information, see Interface <code>GSMetadata</code> in Java API Reference for Oracle WebCenter Portal.) For example, the following EL expression returns a list of all discoverable portals. <code>{spaceContext.spacesQuery.unionOf['DISCOVERABLE'].listSpaces}</code> If <code>listPortals</code> is not appended to the EL, then the EL evaluates to an object of type <code>SpacesQueryParameter</code>. This object type must be evaluated using <code>SpacesManager.getSpaces(SpacesQueryParameters)</code>, which in turn returns a list of portals of type <code>GSMetadata</code>. For example, the following EL expression returns an instance of type <code>SpacesQueryParameters</code> with all the query conditions populated. <code>{spaceContext.spacesQuery.unionOf['DISCOVERABLE'].listSpaces}</code></td>
</tr>
<tr>
<td><code>{spaceContext.spacesQuery.unionOf['USER_JOINED'].shape['ROOT_LEVEL'].listSpaces}</code></td>
<td>Returns a list of all top-level portals of which the current user is a member. This EL returns only those portals that do not have a parent, that is, it does not return subportals.</td>
</tr>
<tr>
<td><code>{spaceContext.spacesQuery.unionOf['USER_JOINED'].shape['RECURSIVE_FLATTENED'].listSpaces}</code></td>
<td>Returns a list of all portals of which the current user is a member. This EL also returns all the subportals under each of the parent portals to which the current user has access. To see an example, refer to Section G.14.5.1, &quot;Example: Using EL Expressions for Various Portals&quot;.</td>
</tr>
</tbody>
</table>
Example: Using EL Expressions for Various Portals

This section provides an example of using EL expressions to query various portals.

In this example, you will use the following EL to display a list of all portals of which you are a member:

"#{spaceContext.spacesQuery.unionOf(['USER_JOINED']).shape('RECURSIVE_FLATTENED')}.listSpaces"

You will also use the following methods to display the logo, name, description, and number of members of a portal:

- iconURI - to display the portal logo
- displayName - to display the portal name
- description - to display the portal description
- memberCount - to display the number of portal members
To query a portal:

1. Create a new task flow, and mark it as available for use, as described in the “Working with Task Flows” section in Building Portals with Oracle WebCenter Portal.

2. Edit the source code of the task flow. On the Assets tab, select the task flow. From the Actions menu, select Edit Source.

3. In the Edit Source dialog, on the Fragment tab, replace the existing code with the following code:

```xml
<?xml version='1.0' encoding='UTF-8'?>
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version='2.1'
xmllns:pe="http://xmlns.oracle.com/adf/pageeditor"
xmllns:cust="http://xmlns.oracle.com/adf/faces/customizable"
xmllns:f="http://java.sun.com/jsf/core"
xmllns:af="http://xmlns.oracle.com/adf/faces/rich"
xmllns:sc="http://java.sun.com/jsf/jstl/core"
xmllns:fn="http://java.sun.com/jsp/jstl/functions">
  <af:panelGroupLayout id="pgl1">
    <cust:panelCustomizable id="pc1">
      <table border="0" width="100%">
        <af:iterator id="it1" var="row" value="#{spaceContext.spacesQuery.unionOf['USER_JOINED'].shape['RECURSIVE_FLATTENED'].listSpaces}"
          var="iconURI" value="#{fn:split(row.iconURI,'}')\[1\]}"
          var="displayName" value="#{row.displayName}" var="description" value="#{row.description}" var="numMembers" value="#{row.memberCount}"/>
        <tr class="PortletText1">
          <td width="16px" rowspan="3" valign="top">
            <af:image id="img1" source="/webcenter-spaces-resources/oracle/webcenter/siteresources/scopedMD/shared#/iconURI}/ICON16"/>
          </td>
          <td>
            <af:outputText id="otCol1" value="#{displayName}" inlineStyle="color:#333333; font-size:12px; font-weight:bold;"/>
          </td>
        </tr>
        <tr>
          <td>
            <af:outputText id="otCol3" value="#{description}" inlineStyle="color:#666666; font-size:12px;"/>
          </td>
        </tr>
        <tr>
          <td>
            <af:outputText id="otCol4" value="members (#{numMembers}) -" inlineStyle="color:#666666; font-size:10px;"/>
            <af:Spacer id="spacer1" width="5px"/>
            <af:goLink id="gl1" destination="/spaces/#{row.name}"/>
          </td>
        </tr>
      </table>
    </cust:panelCustomizable>
  </af:panelGroupLayout>
</jsp:root>
```
4. Add the task flow to a page. In the resource catalog, the custom task flow is available under **UI Components > Task Flows**.

The page should now show a list of all the portals of which you are a member.

### G.14.6 EL Expressions Relevant to Specific Pages

Table G–25 lists EL expressions relevant to portal pages and describes the types of functionality it provides.

<table>
<thead>
<tr>
<th>Expression</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{pageDocBean.title}</td>
<td>Returns the page display name, for example:</td>
</tr>
<tr>
<td></td>
<td>FinanceProject</td>
</tr>
<tr>
<td>#{pageDocBean.createdBy}</td>
<td>Returns the user name of the person who created the page, for example:</td>
</tr>
<tr>
<td></td>
<td>monty</td>
</tr>
<tr>
<td>#{pageDocBean.createDateString}</td>
<td>Returns the date and time the page was created, for example:</td>
</tr>
<tr>
<td></td>
<td>2008-11-19T10:18:36</td>
</tr>
<tr>
<td>#{pageDocBean.lastUpdatedBy}</td>
<td>Returns the user name of the person who last updated the page, for example:</td>
</tr>
<tr>
<td></td>
<td>monty</td>
</tr>
<tr>
<td>#{pageDocBean.lastUpdateDateString}</td>
<td>Returns the date and time the page was last updated, for example:</td>
</tr>
<tr>
<td></td>
<td>2008-11-19T10:18:36</td>
</tr>
<tr>
<td>#{pageDocBean.pagePath}</td>
<td>Returns the file directory path to the page relative to the application root directory, for example:</td>
</tr>
<tr>
<td></td>
<td>/oracle/webcenter/page/scopedMD/s8bba98ff_4eb_40b8_beeec_296c916a23ed/user/Umonty/Page4.jspx</td>
</tr>
<tr>
<td>#{pageDocBean.name}</td>
<td>Returns the file name of the page, for example:</td>
</tr>
<tr>
<td></td>
<td>Page4.jspx</td>
</tr>
<tr>
<td>#{pageDocBean.UICSSStyle}</td>
<td>Returns the name of the style scheme used on the page, for example:</td>
</tr>
<tr>
<td></td>
<td>WCSchemeEggShell</td>
</tr>
<tr>
<td>#{pageDocBean.UISchemeBGImage}</td>
<td>Returns the directory path and file name of the page scheme background image.</td>
</tr>
<tr>
<td>#{pageDocBean.UISchemeBGColorString}</td>
<td>Returns the hexadecimal value of the page scheme background color, for example:</td>
</tr>
<tr>
<td></td>
<td>#ffa500</td>
</tr>
</tbody>
</table>
G.14.7 EL Expressions Relevant to Portal Event Contexts

Table G–26 lists EL expressions relevant to the types of portal events that can trigger the passing of payloads (rather than events relevant to the events feature).

You can access all or part of the event payloads provided in Table G–26 once they have been raised.

For example, for the whole payload, use the following EL:

\#{wcEventContext.events.eventName}

For example:

\#{wcEventContext.events.WebCenterUserSelected}

All of the payloads for the ELs in Table G–26 are Maps. To dereference a map entry, use the standard EL for Maps:

\#{wcEventContext.events.WebCenterUserSelected.UserName}

See Also: For more information about event wiring, see the "Wiring Pages, Task Flows, Portlets, and ADF Components" chapter in Building Portals with Oracle WebCenter Portal.
### Table G–26  EL Expressions Relevant to Event Contexts

<table>
<thead>
<tr>
<th>Expression</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Event Context</strong></td>
<td>Use in context wiring between producer and consumer task flows. Returns a map that relates some piece of metadata from the producer to some piece of metadata from the consumer, for example, from a document creator to the creator’s Profile.</td>
</tr>
<tr>
<td>#{wcEventContext.events.WebCenterResourceSelected}</td>
<td></td>
</tr>
<tr>
<td><strong>Document Name</strong></td>
<td>Producer task flows include:</td>
</tr>
<tr>
<td>#{wcEventContext.events.WebCenterResourceSelected.name}</td>
<td>■ Document Manager</td>
</tr>
<tr>
<td><strong>Document Creator</strong></td>
<td>■ Recent Documents</td>
</tr>
<tr>
<td>#{wcEventContext.events.WebCenterResourceSelected.createdBy}</td>
<td>■ Document List Viewer</td>
</tr>
<tr>
<td><strong>Document Last Modified By</strong></td>
<td>Consumer task flows include:</td>
</tr>
<tr>
<td>#{wcEventContext.events.WebCenterResourceSelected.lastModifiedBy}</td>
<td>■ Activity Graph - Similar Items (show other documents that may be of interest to the user)</td>
</tr>
<tr>
<td></td>
<td>■ Links (show the items that are linked to the selected document)</td>
</tr>
<tr>
<td></td>
<td>■ Profile (view the Profile of the document’s author)</td>
</tr>
<tr>
<td></td>
<td>■ Activity Stream (view activities related to this document)</td>
</tr>
<tr>
<td></td>
<td>■ Tags (tags on this document and a means of accessing the Tag Center)</td>
</tr>
<tr>
<td><strong>User Name</strong></td>
<td>Producer task flows include:</td>
</tr>
<tr>
<td>#{wcEventContext.events.WebCenterUserSelected.userName}</td>
<td>■ Connections</td>
</tr>
<tr>
<td><strong>User GUID</strong></td>
<td>■ Portal Members</td>
</tr>
<tr>
<td>#{wcEventContext.events.WebCenterUserSelected.userId}</td>
<td>Consumer task flows include:</td>
</tr>
<tr>
<td><strong>portal Name</strong></td>
<td>■ Connections (show connections of the selected user)</td>
</tr>
<tr>
<td>#{wcEventContext.events.WebCenterUserSelected.portalName}</td>
<td>■ Documents task flows (show documents created by the selected user)</td>
</tr>
<tr>
<td><strong>portal GUID</strong></td>
<td>■ Activity Stream (view this user’s activities)</td>
</tr>
<tr>
<td>#{wcEventContext.events.WebCenterUserSelected.portalGUID}</td>
<td>■ Tags (tags created by this user)</td>
</tr>
<tr>
<td></td>
<td>■ Profile (show this user’s Profile)</td>
</tr>
</tbody>
</table>
### Table G–26 (Cont.) EL Expressions Relevant to Event Contexts

<table>
<thead>
<tr>
<th>Expression</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>${wcEventContext.events.WebCenterConnectionSelected}</td>
<td>Event Context --</td>
</tr>
<tr>
<td>${wcEventContext.events.WebCenterConnectionSelected.profileUserName}</td>
<td>Profile User Name</td>
</tr>
<tr>
<td>${wcEventContext.events.WebCenterConnectionSelected.profileUserGUID}</td>
<td>Profile User GUID</td>
</tr>
<tr>
<td>${wcEventContext.events.WebCenterConnectionSelected.userName}</td>
<td>Connected User Name</td>
</tr>
<tr>
<td>${wcEventContext.events.WebCenterConnectionSelected.userGUID}</td>
<td>Connected User GUID</td>
</tr>
<tr>
<td>${wcEventContext.events.WebCenterSpaceSelected.SpaceName}</td>
<td>Returns the name of the selected portal</td>
</tr>
<tr>
<td>${wcEventContext.events.WebCenterSpaceSelected.SpaceGUID}</td>
<td>Returns the GUID of the selected portal</td>
</tr>
</tbody>
</table>

### G.14.8 Utilitarian EL Expressions

Table G–27 lists utilitarian EL expressions and describes the types of functionality they provide.

### Table G–27 Utilitarian EL Expressions

<table>
<thead>
<tr>
<th>Expression</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>${userPreferences.currentDate}</td>
<td>Returns the current date in the format specified in the current user's preferences.</td>
</tr>
<tr>
<td>${WCTruncator['text'] ['numberOfChars']}</td>
<td>Returns a truncation of the string specified as text to the number of characters specified as numberOfChars, followed by a trailing ellipsis, for example: ${WCTruncator['abracadabra'] ['5']} evaluates to abrac...</td>
</tr>
<tr>
<td>${facesContext.viewRoot.locale}</td>
<td>Both of these expressions return the request locale (that is, the browser locale setting).</td>
</tr>
<tr>
<td>${facesContext.externalContext.requestLocale}</td>
<td></td>
</tr>
<tr>
<td>${adfFacesContext.skinFamily}</td>
<td>Returns the current application skin family. Returns the same value as ${requestContext.skinFamily}.</td>
</tr>
</tbody>
</table>
G.15 EL Expressions Related to Device Settings

This section lists ELs for obtaining information about device settings. For information about device settings, see the "Administering Device Settings" chapter in Administering Oracle WebCenter Portal.

You can use EL expressions to set assets (skins and page templates) targeted to specific devices. For information on using EL with device settings, see the "Administering Device Settings" chapter in Administering Oracle WebCenter Portal.

- **Table G–28** lists EL expressions for obtaining information about device group mappings.
- **Table G–29** lists ELs for obtaining information about devices.
- **Table G–30** lists ELs for obtaining information about skins and page templates associated with device groups.
- **Section G.15.1, "Sample Task Flow Code for Discovering Device Attributes"** presents sample code that can be used in a task flow for discovering the attributes of the device used to access the portal.

For example, a page designer can use the device's attribute to control the width of the content area of a page by using the following EL:

```el
#{DeviceAgent.device.attributes['display_resolution_width']}
```

### Table G–28  EL Expressions Related to Device Agents

<table>
<thead>
<tr>
<th>Expression</th>
<th>Related Java API</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{DeviceAgent}</td>
<td>DeviceAgent.getInstance()</td>
<td>The root entry point to all device settings EL.</td>
</tr>
<tr>
<td>#{DeviceAgent.device}</td>
<td>DeviceAgent.getInstance().getDevice()</td>
<td>The device to which the current request is mapped.</td>
</tr>
<tr>
<td>#{DeviceAgent.deviceGroups}</td>
<td>DeviceAgent.getInstance().getDeviceGroups()</td>
<td>A list of device groups to which the current device maps.</td>
</tr>
<tr>
<td>#{DeviceAgent.currentScopeDeviceGroup}</td>
<td>DeviceAgent.getInstance().getCurrentScopeDeviceGroup()</td>
<td>Gets the device group that the incoming request was mapped to for the given portal. This EL can help the administrator diagnose issues about why a page variant is not getting displayed for a particular device.</td>
</tr>
</tbody>
</table>

### Table G–29  EL Expressions Relevant to Devices

<table>
<thead>
<tr>
<th>Expression</th>
<th>Related Java API</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{DeviceAgent.device}</td>
<td>DeviceAgent.getInstance().getDevice()</td>
<td>The device to which the current request is mapped.</td>
</tr>
</tbody>
</table>
Table G–29  (Cont.) EL Expressions Relevant to Devices

<table>
<thead>
<tr>
<th>Expression</th>
<th>Related Java API</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{DeviceAgent.device.name}</td>
<td>DeviceAgent.getInstance().getDevice().getName()</td>
<td>The name of the device. This is a system friendly identifier that can be used for internal linking.</td>
</tr>
<tr>
<td>#{DeviceAgent.device.displayName}</td>
<td>DeviceAgent.getInstance().getDevice().getDisplayName()</td>
<td>The display name of the device.</td>
</tr>
</tbody>
</table>
| #{DeviceAgent.device.attributes} | DeviceAgent.getInstance().getDevice().getAttributes() | A map of attributes associated with a device - like the screen size, OS version, and so on. For example, the following EL returns the display resolution width that was set when the device was created: #{DeviceAgent.device.attributes['display_resolution_width']}

Table G–30  EL Expressions Relevant to Device Groups

<table>
<thead>
<tr>
<th>Expression</th>
<th>Related Java API</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{DeviceAgent.currentScopeDeviceGroup}</td>
<td>DeviceAgent.getInstance().getCurrentScopeDeviceGroup()</td>
<td>The device group to which the current request has been mapped. The typical flow for identifying the appropriate device group is as follows: 1. Identify the device to which the current request belongs. 2. Get all the device groups that have been enabled for the current portal. 3. Identify the first device group (from the above list) to which the device belongs.</td>
</tr>
<tr>
<td>#{DeviceAgent.currentScopeDeviceGroup.name}</td>
<td>DeviceAgent.getInstance().getCurrentScopeDeviceGroup().getName()</td>
<td>The name of the device group. For example: #{DeviceAgent.currentScopeDeviceGroup.name == 'androidTablets'}</td>
</tr>
<tr>
<td>#{DeviceAgent.currentScopeDeviceGroup.portalPageTemplate}</td>
<td>DeviceAgent.getInstance().getCurrentScopeDeviceGroup().getPortalPageTemplate()</td>
<td>Returns the GUID of the portal page template associated with this device group object. If the method returns NULL, the default page template setting for the portal is in effect. <strong>Note:</strong> For information on the resource ELs used to work with the obtained GUID, see Section G.6, &quot;ELs Related to Portal Resources.”</td>
</tr>
</tbody>
</table>
Sample Task Flow Code for Discovering Device Attributes

Example G–1 lists a JSF page fragment that can be used to create a task flow for discovering device attributes. EL is used to return the attributes. The task flow may be useful for troubleshooting situations where the portal is not rendering properly on a device. By discovering the device attributes, you can take action to correct the rendering problem. Figure G–8 shows output from a task flow created with this sample code.

Example G–1 Task Flow Code for Discovering Device Attributes

```xml
<?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/pageditor'
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 layout='scroll' id='pgl1'>
    <cust:panelCustomizable id='pc1'/>
    <f:facet name='separator'>
      <af:separator id='s1'/>
    </f:facet>
    <af:panelFormLayout id='pfl1'>
      <af:group id='g1'>
        <af:panelLabelAndMessage id='it0' label='Current Device'
labelStyle='color:#0000ff;font-weight:bold;'/>
        <af:outputText id='it1' value='#{DeviceAgent.device.name}'/>
        <af:panelLabelAndMessage id='it1plm' label='Internal Name'>
          <af:outputText id='it1' value='#{DeviceAgent.device.name}'/>
        </af:panelLabelAndMessage>
        <af:panelLabelAndMessage id='it2plm' label='Display Name'>
          <af:outputText id='it2' value='#{DeviceAgent.device.displayName}'/>
        </af:panelLabelAndMessage>
      </af:group>
    </af:panelGroupLayout>
</jsp:root>
```

Table G–30 (Cont.) EL Expressions Relevant to Device Groups

<table>
<thead>
<tr>
<th>Expression</th>
<th>Related Java API</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>#{DeviceAgent.currentScopeDeviceGroup.portalSkin}</td>
<td>DeviceAgent.getInstance().getCurrentScopeDeviceGroup().getPortalSkin()</td>
<td>Returns the GUID of the portal skin associated with this device group. If the method returns NULL, the default skin setting for the portal is in effect. Note: For information on the resource ELs used to work with the obtained GUID, see Section G.6, &quot;ELs Related to Portal Resources.&quot;</td>
</tr>
<tr>
<td>#{DeviceAgent.currentScopeDeviceGroup.default}</td>
<td>DeviceAgent.getInstance().getCurrentScopeDeviceGroup().isDefault()</td>
<td>Returns true if the device group is the default for the current device. Returns false if the device group is not the default for the device. For example, if the default device group is &quot;Desktop Browsers&quot; and you access the portal on an iPad, then the device group will be iPad, but it is not the default device group. In this case, the method returns false.</td>
</tr>
<tr>
<td>#{DeviceAgent.currentScopeDeviceGroup.enabled}</td>
<td>DeviceAgent.getInstance().getCurrentScopeDeviceGroup().isEnabled()</td>
<td>Identifies whether or not this device group has been marked as enabled for the current portal. Note: In this invocation, the device group will always state true. This EL is useful when working with all the device groups as obtained from the EL #{DeviceAgent.deviceGroups}.</td>
</tr>
</tbody>
</table>
<af:panelLabelAndMessage id="it01" label="Current Device Group"
 labelStyle="color:#0000ff;font-weight:bold;"/>
<af:panelLabelAndMessage id="it3plm" label="Internal Name">
<af:outputText id="it3" value="#{DeviceAgent.currentScopeDeviceGroup.name}" />
</af:panelLabelAndMessage>
<af:panelLabelAndMessage id="it4plm" label="Display Name">
<af:outputText id="it4" value="#{DeviceAgent.currentScopeDeviceGroup.displayName}" />
</af:panelLabelAndMessage>
<af:panelLabelAndMessage id="it5plm" label="PageTemplate">
<af:outputText id="it5" value="#{srmContext.id[DeviceAgent.currentScopeDeviceGroup.portalPageTemplate].singleResult.displayName}" />
</af:panelLabelAndMessage>
<af:panelLabelAndMessage id="it6plm" label="Skin">
<af:outputText id="it6" value="#{srmContext.id[DeviceAgent.currentScopeDeviceGroup.portalSkin].singleResult.displayName}" />
</af:panelLabelAndMessage>
<af:panelLabelAndMessage id="it7plm" label="Is Default">
<af:outputText id="it7" value="#{DeviceAgent.currentScopeDeviceGroup.default}" />
</af:panelLabelAndMessage>
<af:panelLabelAndMessage id="it8plm" label="Is Enabled">
<af:outputText id="it8" value="#{DeviceAgent.currentScopeDeviceGroup.enabled}" />
</af:panelLabelAndMessage>
<af:panelLabelAndMessage id="it9plm" label="Current Browser User Agent"
 labelStyle="color:#0000ff;font-weight:bold;"/>
<af:outputText id="it9" value="#{DeviceAgent.userAgent}" />
</af:panelLabelAndMessage>
<af:panelLabelAndMessage id="it010plm" label="Expected PageTemplate:GUID">
<af:outputText id="it010" value="#{DeviceAgent.currentScopeDeviceGroup.portalPageTemplate}" />
</af:panelLabelAndMessage>
<af:panelLabelAndMessage id="it0010plm" label="Expected PageTemplate:Name">
<af:outputText id="it0010" value="#{srmContext.id[DeviceAgent.currentScopeDeviceGroup.portalPageTemplate].singleResult.displayName}" />
</af:panelLabelAndMessage>
<af:panelLabelAndMessage id="it10plm" label="Current PageTemplate:GUID">
<af:outputText id="it10" value="#{WCAppContext.application.siteTemplatesManager.currentSiteTemplateId}" />
</af:panelLabelAndMessage>
<af:panelLabelAndMessage id="it10aplm" label="Current PageTemplate:Name">
<af:outputText id="it10a" value="#{srmContext.id[WCAppContext.application.siteTemplatesManager.currentSiteTemplateId].singleResult.displayName}" />
</af:panelLabelAndMessage>
<af:panelLabelAndMessage id="it9plm" label="Current Browser User Agent"
 labelStyle="color:#0000ff;font-weight:bold;"/>
<af:outputText id="it9" value="#{DeviceAgent.userAgent}" />
</af:panelLabelAndMessage>
<af:panelLabelAndMessage id="it0111plm" label="Expected Skin:GUID">
<af:outputText id="it0111" value="#{DeviceAgent.currentScopeDeviceGroup.portalSkin}" />
</af:panelLabelAndMessage>
<af:panelLabelAndMessage id="it0112plm" label="Expected Skin:Name">
<af:outputText id="it0111" value="#{DeviceAgent.currentScopeDeviceGroup.portalSkin}" />
</af:panelLabelAndMessage>
<af:outputText id="it0112" value="#{srmContext.id[DeviceAgent.currentScopeDeviceGroup.portalSkin].singleResult.displayName}"/>
</af:panelLabelAndMessage>
<af:panelLabelAndMessage id="it111plm" label="Current Skin:GUID">
<af:outputText id="it111" value=""/>
</af:panelLabelAndMessage>
<af:panelLabelAndMessage id="it112plm" label="Current Skin:Name">
<af:outputText id="it112" value="#{requestScope['oracle.webcenter.webcenterapp.view.shell.WebCenterShellManager.SHELLHANDLER'].adfFacesSkin}"/>
</af:panelLabelAndMessage>
</af:group>
<af:group>
<af:panelLabelAndMessage id="pg0" label="Page Info" labelStyle="color:#0000ff;font-weight:bold;"/>
<af:panelLabelAndMessage id="pg10" label="Page Path">
<af:outputText id="pg1" value="#{pageDocBean.pagePath}"/>
</af:panelLabelAndMessage>
<af:panelLabelAndMessage id="pg20" label="Page Style">
<af:outputText id="pg2" value=""/>
</af:panelLabelAndMessage>
</af:group>
<af:group>
<af:panelLabelAndMessage id="ap90" label="Optional Attributes" labelStyle="color:#0000ff;font-weight:bold;"/>
<af:panelLabelAndMessage id="ap1opta" label="brand-name">
<af:outputText id="ap1" value="#{DeviceAgent.device.attributes['brand-name']}"/>
</af:panelLabelAndMessage>
<af:panelLabelAndMessage id="ap2opta" label="device-os">
<af:outputText id="ap2" value="#{DeviceAgent.device.attributes['device-os']}"/>
</af:panelLabelAndMessage>
<af:panelLabelAndMessage id="ap3opta" label="device-type">
<af:outputText id="ap3" value="#{DeviceAgent.device.attributes['device-type']}"/>
</af:panelLabelAndMessage>
<af:panelLabelAndMessage id="ap4opta" label="device_default_aspect_ratio">
<af:outputText id="ap4" value="#{DeviceAgent.device.attributes['device_default_aspect_ratio']}"/>
</af:panelLabelAndMessage>
<af:panelLabelAndMessage id="ap5opta" label="device_os-version">
<af:outputText id="ap5" value="#{DeviceAgent.device.attributes['device_os-version']}"/>
</af:panelLabelAndMessage>
<af:panelLabelAndMessage id="ap6opta" label="device_preview_css">
<af:outputText id="ap6" value="#{DeviceAgent.device.attributes['device_preview_css']}"/>
</af:panelLabelAndMessage>
<af:panelLabelAndMessage id="ap7opta" label="device_preview_horizontal_css">
<af:outputText id="ap7" value="#{DeviceAgent.device.attributes['device_preview_horizontal_css']}"/>
</af:panelLabelAndMessage>
<af:panelLabelAndMessage id="ap8opta" label="device_preview_viewport_css">
<af:outputText id="ap8" value="#{DeviceAgent.device.attributes['device_preview_viewport_css']}"/>
</af:panelLabelAndMessage>
<af:panelLabelAndMessage id="ap9opta" label="device_preview_viewport_horizontal_css">
<af:outputText id="ap9" value="#{DeviceAgent.device.attributes['device_preview_viewport_horizontal_css']}"/>
</af:panelLabelAndMessage>
<af:panelLabelAndMessage id="ap10opta" label="device_streaming_flv">
EL Expressions Related to Device Settings

<af:outputText id="ap10" value="#{DeviceAgent.device.attributes['device_streaming_flv']}"/>
</af:panelLabelAndMessage>
<af:outputText id="ap11" value="#{DeviceAgent.device.attributes['device_streaming_preferred_http_protocol']}"/>
</af:panelLabelAndMessage>
<af:outputText id="ap12" value="#{DeviceAgent.device.attributes['device_streaming_preferred_protocol']}"/>
</af:panelLabelAndMessage>
<af:outputText id="ap13" value="#{DeviceAgent.device.attributes['device_video_streaming']}"/>
</af:panelLabelAndMessage>
<af:outputText id="ap14" value="#{DeviceAgent.device.attributes['display_orientation_support']}"/>
</af:panelLabelAndMessage>
<af:outputText id="ap15" value="#{DeviceAgent.device.attributes['display_resolution_height']}"/>
</af:panelLabelAndMessage>
<af:outputText id="ap16" value="#{DeviceAgent.device.attributes['display_resolution_width']}"/>
</af:panelLabelAndMessage>
<af:outputText id="ap17" value="#{DeviceAgent.device.attributes['is-wireless_device']}"/>
</af:panelLabelAndMessage>
<af:outputText id="ap18" value="#{DeviceAgent.device.attributes['marketing-name']}"/>
</af:panelLabelAndMessage>
<af:outputText id="ap19" value="#{DeviceAgent.device.attributes['model-name']}"/>
</af:panelLabelAndMessage>
<af:outputText id="ap20" value="#{DeviceAgent.device.attributes['model-version']}"/>
</af:panelLabelAndMessage>
<af:outputText id="ap21" value="#{DeviceAgent.device.attributes['icon']}"/>
</af:panelLabelAndMessage>
</af:group>
</af:panelFormLayout>
</af:panelGroupLayout>
</jsp:root>
### Figure G-8  Output from Sample Task Flow

<table>
<thead>
<tr>
<th>Mobile EL Taskflow</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Current Device</strong></td>
</tr>
<tr>
<td>Internal Name</td>
</tr>
<tr>
<td>Display Name</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Current Device Group</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal Name</td>
</tr>
<tr>
<td>Display Name</td>
</tr>
<tr>
<td>PageTemplate</td>
</tr>
<tr>
<td>IsDefault</td>
</tr>
<tr>
<td>IsEnabled</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Current Browser User Agent</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Mozilla/5.0 (iPad; CPU OS 5.1; U; en) AppleWebKit/6531.22.15 (KHTML, like Gecko) Version/5.1 Mobile Safari/6531.22.15</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Page Template Info</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Expected PageTemplate GUId</td>
</tr>
<tr>
<td>Current PageTemplate GUId</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Skin Info</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Expected Skin GUId</td>
</tr>
<tr>
<td>Current Skin GUId</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Page Info</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>PagePath</td>
</tr>
<tr>
<td>PageName</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Optional Attributes</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>brand-name</td>
</tr>
<tr>
<td>device-os</td>
</tr>
<tr>
<td>device-type</td>
</tr>
<tr>
<td>device_default_aspect-ratio</td>
</tr>
</tbody>
</table>
EL Expressions Related to Device Settings
This appendix describes the underlying database design for Oracle WebCenter Portal's Analytics for developers building custom analytics data queries.

This appendix contains the following topics:

- Section H.1, "Analytics Database Schema Overview"
- Section H.2, "Analytics Fact Table Descriptions"
- Section H.3, "Analytics Dimension Table Descriptions"
- Section H.4, "Analytics System Table Descriptions"
- Section H.5, "Analytics User Properties"
- Section H.6, "Analytics Event and Dimension Data Types"

For more information about building custom analytics data queries, see Section 47.3, "Building Analytics Reports."

H.1 Analytics Database Schema Overview

This section contains the following topics:

- Section H.1.1, "Analytics Database Model Overview"
- Section H.1.2, "Analytics Database Table Overview"

H.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, portal 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 H–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, portal name, and so on.

Table H–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 H–1 shows the relationship between predefined fact, dimension, and application tables in the analytics database schema.

**Figure H–1 Analytics Database Model**

![Analytics Database Model Diagram]

Tables shown in Figure H–1 store the following analytics data:

- Application configuration
- Event metadata
- Report metadata

**Application Configuration**

Figure H–2 shows which entities are used to store system parameters for Analytics.

**Figure H–2 Application Configuration Entities**

![Application Configuration Entities Diagram]

**Event Metadata**

Figure H–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.
Report Metadata

Figure H–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 Oracle 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.

H.1.2 Analytics Database Table Overview

This section lists and describes database tables in the analytics schema:

- Table H–1, "Analytics Facts Tables (Prefixed ASFACT_WC)"
- Table H–2, "Analytics Dimension Tables (Prefixed ASDIM_WC)"
- Table H–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.

Some table names include double underscores, for example, `ASFACT_WC_DOCLIB__0`. This is because all the table names include a component value that is 7 characters long, and in some cases one of those characters is an underscore (for example, `DOCLIB_`).

For an example of how to use a query to return the table names associated with a specific event, see Section H.1.2.1, "Sample SQL Query: Finding Table Names Associated with Specific Events".

<table>
<thead>
<tr>
<th>Event Object</th>
<th>Table Name</th>
<th>Event Table Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pages</td>
<td>ASFACT_WC_PAGECRE_0</td>
<td>Stores event information captured by Analytics when a page is created.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_PAGEDEL_0</td>
<td>Stores event information captured by Analytics when a page is deleted.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_PAGEEDI_0</td>
<td>Stores event information captured by Analytics when a page is edited.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_PAGETAG_0</td>
<td>Stores event information captured by Analytics when a page is tagged.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_PAGEVIE_0</td>
<td>Stores event information captured by Analytics when a page is viewed.</td>
</tr>
<tr>
<td>Discussions</td>
<td>ASFACT_WC_DISCUSS_0</td>
<td>Stores event information captured by Analytics when a discussion topic is viewed.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_DISCUSS_1</td>
<td>Stores event information captured by Analytics when an announcement is viewed.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_DISCUSS_2</td>
<td>Stores event information captured by Analytics when a discussion forum is created.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_DISCUSS_3</td>
<td>Stores event information captured by Analytics when a discussion forum is deleted.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_DISCUSS_4</td>
<td>Stores event information captured by Analytics when a discussion topic is created.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_DISCUSS_5</td>
<td>Stores event information captured by Analytics when a discussion topic is edited.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_DISCUSS_6</td>
<td>Stores event information captured by Analytics when a discussion topic is deleted.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_DISCUSS_7</td>
<td>Stores event information captured by Analytics when a discussion topic is tagged.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_DISCUSS_8</td>
<td>Stores event information captured by Analytics when someone replies to a discussion topic message.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_DISCUSS_9</td>
<td>Stores event information captured by Analytics when a discussion topic message is liked.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_DISCUSS_A</td>
<td>Stores event information captured by Analytics when an announcement is created.</td>
</tr>
</tbody>
</table>
### Table H-1 (Cont.) Analytics Facts Tables (Prefixed ASFACT_WC)

<table>
<thead>
<tr>
<th>Event Object</th>
<th>Table Name</th>
<th>Event Table Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Documents -</td>
<td>ASFACT_WC_DISCUSS_B</td>
<td>Stores event information captured by Analytics when an announcement is edited.</td>
</tr>
<tr>
<td>Blog - Wikis</td>
<td>ASFACT_WC_DISCUSS_C</td>
<td>Stores event information captured by Analytics when an announcement is deleted.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_DOCLIB__0</td>
<td>Stores event information captured by Analytics when a document is viewed.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_DOCLIB__1</td>
<td>Stores event information captured by Analytics when a document is downloaded.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_DOCLIB__2</td>
<td>Stores event information captured by Analytics when a document is created.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_DOCLIB__3</td>
<td>Stores event information captured by Analytics when a document is edited.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_DOCLIB__4</td>
<td>Stores event information captured by Analytics when a document is tagged.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_DOCLIB__5</td>
<td>Stores event information captured by Analytics when a document is liked.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_DOCLIB__6</td>
<td>Stores event information captured by Analytics when a document is commented.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_DOCLIB__7</td>
<td>Stores event information captured by Analytics when a document is deleted.</td>
</tr>
<tr>
<td>Portal Events</td>
<td>ASFACT_WC_EVENT_C_0</td>
<td>Stores event information captured by Analytics when a portal event is created.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_EVENT_D_0</td>
<td>Stores event information captured by Analytics when a portal event is deleted.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_EVENT_E_0</td>
<td>Stores event information captured by Analytics when a portal event is edited.</td>
</tr>
<tr>
<td>Portals</td>
<td>ASFACT_WC_GROUPSP_0</td>
<td>Stores event information captured by Analytics when a portal is viewed.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_GROUPSP_1</td>
<td>Stores event information captured by Analytics when a portal is created.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_GROUPSP_2</td>
<td>Stores event information captured by Analytics when someone joins a portal.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_GROUPSP_3</td>
<td>Stores event information captured by Analytics when a portal is deleted.</td>
</tr>
<tr>
<td>Lists</td>
<td>ASFACT_WC_LIST_CR_0</td>
<td>Stores event information captured by Analytics when a list is created.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_LIST_DE_0</td>
<td>Stores event information captured by Analytics when a list is deleted.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_LIST_ED_0</td>
<td>Stores event information captured by Analytics when a list is edited.</td>
</tr>
<tr>
<td>Logins</td>
<td>ASFACT_WC_LOGINS_0</td>
<td>Stores event information captured by Analytics when a user logs in to the site.</td>
</tr>
<tr>
<td>People</td>
<td>ASFACT_WC_PEOPLEC_0</td>
<td>Stores event information captured by Analytics when someone adds a connection.</td>
</tr>
<tr>
<td></td>
<td>ASFACT_WC_PEOPLEC_1</td>
<td>Stores event information captured by Analytics when someone posts on a wall.</td>
</tr>
</tbody>
</table>
**Table H–1 (Cont.) Analytics Facts Tables (Prefixed ASFACT_WC)**

<table>
<thead>
<tr>
<th>Event Object</th>
<th>Table Name</th>
<th>Event Table Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Portlets</td>
<td>ASFACT_WC_PORTLET_0</td>
<td>Stores event information captured by Analytics when a portlet is viewed.</td>
</tr>
<tr>
<td>Searches</td>
<td>ASFACT_WC_SEARCHCE_0</td>
<td>Stores event information captured by Analytics when a search is performed.</td>
</tr>
</tbody>
</table>

**Table H–2 Analytics Dimension Tables (Prefixed ASDIM_WC)**

<table>
<thead>
<tr>
<th>Dimension Name</th>
<th>Table Name</th>
<th>Dimension Table Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Applications</td>
<td>ASDIM_WC_APPLICA_0</td>
<td>Stores names of WebCenter Portal applications accessing analytics data.</td>
</tr>
<tr>
<td>IP Clients</td>
<td>ASDIM_WC_CLIENT_0</td>
<td>Stores client IP addresses received in analytics events.</td>
</tr>
<tr>
<td>Discussion Topics</td>
<td>ASDIM_WC_DISCUSS_0</td>
<td>Stores discussion topic details received in analytics events.</td>
</tr>
<tr>
<td>Discussion Forums</td>
<td>ASDIM_WC_DISCUSS_1</td>
<td>Stores discussion forum details received in analytics events.</td>
</tr>
<tr>
<td>Discussion Messages</td>
<td>ASDIM_WC_DISCUSS_2</td>
<td>Stores discussion message details received in analytics events.</td>
</tr>
<tr>
<td>Announcements</td>
<td>ASDIM_WC_DISCUSS_3</td>
<td>Stores announcement details received in analytics events.</td>
</tr>
<tr>
<td>Documents</td>
<td>ASDIM_WC_DOCUMENT_0</td>
<td>Stores document details received in analytics events.</td>
</tr>
<tr>
<td>Lists</td>
<td>ASDIM_WC_LISTS_0</td>
<td>Stores list details received in analytics events.</td>
</tr>
<tr>
<td>Pages</td>
<td>ASDIM_WC_PAGES_0</td>
<td>Stores WebCenter Portal page details received in analytics events.</td>
</tr>
<tr>
<td>Portlets</td>
<td>ASDIM_WC_PORTLET_0</td>
<td>Stores portlet information received in analytics events.</td>
</tr>
<tr>
<td>Portlet Instances</td>
<td>ASDIM_WC_PORTLET_1</td>
<td>Stores portlet instance information received in analytics events.</td>
</tr>
<tr>
<td>Portlet Producers</td>
<td>ASDIM_WC_PRODUCE_0</td>
<td>Stores producers associated with portlet.</td>
</tr>
<tr>
<td>Referrers</td>
<td>ASDIM_WC_REFERER_0</td>
<td>Stores referred URLs where analytics events come from.</td>
</tr>
<tr>
<td>Searches</td>
<td>ASDIM_WC_SEARCHCE_0</td>
<td>Stores search phrases received in analytics events.</td>
</tr>
<tr>
<td>Portal Events</td>
<td>ASDIM_WC_EVENTS_0</td>
<td>Stores portal event details, received in analytics events.</td>
</tr>
<tr>
<td>Portals</td>
<td>ASDIM_WC_GROUPSP_0</td>
<td>Stores portal information received in analytics events.</td>
</tr>
<tr>
<td>Tags</td>
<td>ASDIM_WC_TAGS_0</td>
<td>Stores tags received in analytics events.</td>
</tr>
<tr>
<td>Browsers</td>
<td>ASDIM_WC_USER_AG_0</td>
<td>Stores information about which browsers are used to access analytics.</td>
</tr>
<tr>
<td>Times</td>
<td>ASDIM_TIME</td>
<td>Stores times used to group analytics reports.</td>
</tr>
</tbody>
</table>
Table H-2 (Cont.) Analytics Dimension Tables (Prefixed ASDIM_WC)

<table>
<thead>
<tr>
<th>Dimension Name</th>
<th>Table Name</th>
<th>Dimension Table Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Users</td>
<td>ASDIM_USERS</td>
<td>Stores user details received in analytics events.</td>
</tr>
<tr>
<td>User Properties</td>
<td>ASDIM_USERPROPERTIES</td>
<td>Stores user properties.</td>
</tr>
<tr>
<td>User Property Values</td>
<td>ASDIM_USERPROPERTYVALUES</td>
<td>Stores user property values.</td>
</tr>
</tbody>
</table>

Table H-3 Analytics System Tables (Prefixed ASSYS)

<table>
<thead>
<tr>
<th>Entity Name</th>
<th>Table Name</th>
<th>Table Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration</td>
<td>ASSYS_CONFIG</td>
<td>Stores system configuration details.</td>
</tr>
<tr>
<td>Dimension</td>
<td>ASSYS_DIMENSIONPROPS</td>
<td>Stores configuration details for each dimension.</td>
</tr>
<tr>
<td>Event Dimensions</td>
<td>ASSYS_EVENTDIMENSIONS</td>
<td>Stores dimensions associated with each event.</td>
</tr>
<tr>
<td>Event Facts</td>
<td>ASSYS_EVENTFACTS</td>
<td>Stores facts associated with each event.</td>
</tr>
<tr>
<td>Events</td>
<td>ASSYS_EVENTS</td>
<td>Stores the different types of analytics events that are collected by Analytics. An analytics event represents a single user action, such as document view, user login, and so on.</td>
</tr>
<tr>
<td>Namespaces</td>
<td>ASSYS_NAMESPACES</td>
<td>Stores the namespace in which analytics events are registered.</td>
</tr>
<tr>
<td>Report Composition</td>
<td>ASSYS_REPORTCOMPOSITION</td>
<td>Stores out-of-the-box composite analytics reports.</td>
</tr>
<tr>
<td>Report Group</td>
<td>ASSYSREPORTGROUP</td>
<td>Stores out-of-the-box analytics report groups, such as portlets, services, and so on.</td>
</tr>
<tr>
<td>Report Items</td>
<td>ASSYSREPORTITEMS</td>
<td>Stores items included in each analytics report.</td>
</tr>
<tr>
<td>Report Item Values</td>
<td>ASSYSREPORTITEMVALUES</td>
<td>Stores values associated with each item in an analytics report.</td>
</tr>
<tr>
<td>Reports</td>
<td>ASSYSREPORTS</td>
<td>Stores out-of-the-box analytics reports.</td>
</tr>
</tbody>
</table>

H.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 documents.

```sql
SELECT tablename "Table Name",
       displayname 'Event Display Name'
FROM assys_events
WHERE tablename like 'ASFACT_WC_DOCLIB%'
ORDER BY 1;
```

Sample Report Output

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Event Display Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASFACT_WC_DOCLIB__0</td>
<td>(<a href="HTTP://WWW.ORACLE.COM/ANALYTICS/WC)DOCLIB_DOCUMENTVIEWS">HTTP://WWW.ORACLE.COM/ANALYTICS/WC)DOCLIB_DOCUMENTVIEWS</a></td>
</tr>
<tr>
<td>ASFACT_WC_DOCLIB__1</td>
<td>(<a href="HTTP://WWW.ORACLE.COM/ANALYTICS/WC)DOCLIB_DOCUMENTDOWNLOADS">HTTP://WWW.ORACLE.COM/ANALYTICS/WC)DOCLIB_DOCUMENTDOWNLOADS</a></td>
</tr>
<tr>
<td>ASFACT_WC_DOCLIB__2</td>
<td>(<a href="HTTP://WWW.ORACLE.COM/ANALYTICS/WC)DOCLIB_DOCUMENTCREATE">HTTP://WWW.ORACLE.COM/ANALYTICS/WC)DOCLIB_DOCUMENTCREATE</a></td>
</tr>
<tr>
<td>ASFACT_WC_DOCLIB__3</td>
<td>(<a href="HTTP://WWW.ORACLE.COM/ANALYTICS/WC)DOCLIB_DOCUMENTEDIT">HTTP://WWW.ORACLE.COM/ANALYTICS/WC)DOCLIB_DOCUMENTEDIT</a></td>
</tr>
</tbody>
</table>
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, portal 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 H–1, "Analytics Facts Tables (Prefixed ASFACT_WC)".

H.2.1 ASFACT_WC_PAGECRE_0

Fact table that stores event information captured by Analytics when someone creates a page.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time the dimension event occurred.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>PAGE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Page ID dimension.</td>
<td>ASDIM_WC_PAGES_0.ID</td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WC_GROUPSP_0.ID</td>
</tr>
<tr>
<td>APPLICATION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID.</td>
<td>ASDIM_WC_APPLICA_0.ID</td>
</tr>
</tbody>
</table>

H.2.2 ASFACT_WC_PAGEDEL_0

Fact table that stores event information captured by Analytics when someone deletes a page.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time the dimension event occurred.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WC_GROUPSP_0.ID</td>
</tr>
</tbody>
</table>

Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper
### H.2.3 ASFACT_WC_PAGEDEII_0

Fact table that stores event information captured by Analytics when someone edits a page.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>APPLICATION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_WC_APPLICA_0.ID</td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WC_GROUPSP_0.ID</td>
</tr>
<tr>
<td>PAGE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Page ID dimension.</td>
<td>ASDIM_WC_PAGES_0.ID</td>
</tr>
</tbody>
</table>

### H.2.4 ASFACT_WC_PAGETAG_0

Fact table that stores event information captured by Analytics when someone tags a page.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WC_GROUPSP_0.ID</td>
</tr>
<tr>
<td>TAG_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Tag ID dimension.</td>
<td>ASDIM_WC_USER_AG_0.ID</td>
</tr>
<tr>
<td>PAGE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Page ID dimension.</td>
<td>ASDIM_WC_PAGES_0.ID</td>
</tr>
<tr>
<td>APPLICATION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_WC_APPLICA_0.ID</td>
</tr>
</tbody>
</table>
H.2.5  ASFACT_WC_PAGEVIE_0

Fact table that stores event information captured by Analytics when someone views a page.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>RESPONSE_TIME_</td>
<td>NULL</td>
<td>NUMBER(16)</td>
<td>Response time (ms).</td>
<td></td>
</tr>
<tr>
<td>CLIENT_IP_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Client IP ID dimension.</td>
<td>ASDIM_WC_CLIENT_.0.ID</td>
</tr>
<tr>
<td>SESSION_ID_</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>Session ID.</td>
<td></td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WC_GROUPSP_0.ID</td>
</tr>
<tr>
<td>REFERRE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Referrer ID dimension.</td>
<td>ASDIM_WC_REFERER_0.ID</td>
</tr>
<tr>
<td>PAGE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Page ID dimension.</td>
<td>ASDIM_WC_PAGES_0.ID</td>
</tr>
<tr>
<td>USER_AGENT_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User agent.</td>
<td>ASDIM_WC_USER_AG_0.ID</td>
</tr>
<tr>
<td>APPLICATION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_WC_APPLICA_0.ID</td>
</tr>
</tbody>
</table>

H.2.6  ASFACT_WC_DISCUSS_0

Fact table that stores event information captured by Analytics when someone views a discussion topic.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>CLIENT_IP_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Client IP ID dimension.</td>
<td>ASDIM_WC_CLIENT_.0.ID</td>
</tr>
<tr>
<td>APPLICATION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_WC_APPLICA_0.ID</td>
</tr>
<tr>
<td>USER_AGENT_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User agent.</td>
<td>ASDIM_WC_USER_AG_0.ID</td>
</tr>
</tbody>
</table>
### H.2.7 ASFACT_WC_DISCUSS_1
Fact table that stores event information captured by Analytics when someone views an announcement.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>CLIENT_IP_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Client IP dimension.</td>
<td>ASDIM_WC_CLIENT_0.ID</td>
</tr>
<tr>
<td>APPLICATION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_WC_APPLICA_0.ID</td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WC_GROUPSP_0.ID</td>
</tr>
<tr>
<td>REFERRE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Referrer ID dimension.</td>
<td>ASDIM_WCREFERRE_0.ID</td>
</tr>
<tr>
<td>USER_AGENT_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User agent.</td>
<td>ASDIM_WC_USER_AG_0.ID</td>
</tr>
<tr>
<td>SESSION_ID_</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>Session ID.</td>
<td></td>
</tr>
<tr>
<td>ANNOUNCEMENT_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Announcement associated with the event.</td>
<td>ASDIM_WC_DISCUSS_3.ID</td>
</tr>
</tbody>
</table>

### H.2.8 ASFACT_WC_DISCUSS_2
Fact table that stores event information captured by Analytics when someone creates a discussion forum.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
</tbody>
</table>
### H.2.9 ASFACT_WC_DISCUSS_3

Fact table that stores event information captured by Analytics when someone deletes a discussion forum.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WC_GROUPSP_0.ID</td>
</tr>
<tr>
<td>APPLICATION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_WC_APPLICA_0.ID</td>
</tr>
<tr>
<td>DISCUSSION_FORUM_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Discussion forum ID dimension.</td>
<td>ASDIM_WC_DISCUSS_0.ID</td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred.</td>
<td>ASDIM_TIME.ID</td>
</tr>
</tbody>
</table>

### H.2.10 ASFACT_WC_DISCUSS_4

Fact table that stores event information captured by Analytics when someone creates a discussion topic.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WC_GROUPSP_0.ID</td>
</tr>
<tr>
<td>DISCUSSION_TOPIC_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Discussion topic ID dimension.</td>
<td>ASDIM_WC_DISCUSS_0.ID</td>
</tr>
<tr>
<td>DISCUSSION_MESSAGE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Discussion message ID dimension.</td>
<td>ASDIM_WC_DISCUSS_2.ID</td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred.</td>
<td>ASDIM_TIME.ID</td>
</tr>
</tbody>
</table>
H.2.11 ASFACT_WC_DISCUSS_5
Fact table that stores event information captured by Analytics when someone edits a discussion topic.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>APPLICATION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_WC_APPLICATION_0.ID</td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WCGROUPSP_0.ID</td>
</tr>
<tr>
<td>DISCUSSION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Discussion topic ID dimension.</td>
<td>ASDIM_WCDISCUSS_0.ID</td>
</tr>
</tbody>
</table>

H.2.12 ASFACT_WC_DISCUSS_6
Fact table that stores event information captured by Analytics when someone deletes a discussion topic.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WCGROUPSP_0.ID</td>
</tr>
<tr>
<td>APPLICATION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_WC_APPLICATION_0.ID</td>
</tr>
<tr>
<td>DISCUSSION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Discussion topic ID dimension.</td>
<td>ASDIM_WCDISCUSS_0.ID</td>
</tr>
</tbody>
</table>

H.2.13 ASFACT_WC_DISCUSS_7
Fact table that stores event information captured by Analytics when someone tags a discussion topic.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WCGROUPSP_0.ID</td>
</tr>
<tr>
<td>APPLICATION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_WC_APPLICATION_0.ID</td>
</tr>
<tr>
<td>DISCUSSION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Discussion topic ID dimension.</td>
<td>ASDIM_WCDISCUSS_0.ID</td>
</tr>
</tbody>
</table>
### H.2.14 ASFACT_WC_DISCUSS_8

Fact table that stores event information captured by Analytics when someone replies to a discussion topic message.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>TAG_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Tag ID dimension.</td>
<td>ASDIM_WC_TAGS_0.ID</td>
</tr>
<tr>
<td>APPLICATION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_WC_APPLICA_0.ID</td>
</tr>
<tr>
<td>DISCUSSION_TOPIC_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Discussion topic ID dimension.</td>
<td>ASDIM_WC_DISCUSS_0.ID</td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WC_GROUPSP_0.ID</td>
</tr>
</tbody>
</table>

### H.2.15 ASFACT_WC_DISCUSS_9

Fact table that stores event information captured by Analytics when someone likes a discussion topic message.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred.</td>
<td></td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td></td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td></td>
</tr>
<tr>
<td>DISCUSSION_MESSAGE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Discussion message ID dimension.</td>
<td>ASDIM_WC_DISCUSS_2.ID</td>
</tr>
<tr>
<td>DISCUSSION_TOPIC_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Discussion topic ID dimension.</td>
<td>ASDIM_WC_DISCUSS_0.ID</td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WC_GROUPSP_0.ID</td>
</tr>
<tr>
<td>APPLICATION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_WC_APPLICA_0.ID</td>
</tr>
</tbody>
</table>
### H.2.16 ASFACT_WC_DISCUSS_A
Fact table that stores event information captured by Analytics when someone creates an announcement.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>DISCUSSION_MESSAGE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Discussion message ID dimension.</td>
<td>ASDIM_WC_DISCUSS_0.ID</td>
</tr>
<tr>
<td>APPLICATION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_WC_APPLICA_0.ID</td>
</tr>
<tr>
<td>DISCUSSION_TOPIC_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Discussion topic ID dimension.</td>
<td>ASDIM_WC_DISCUSS_3.ID</td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WC_GROUPSP_0.ID</td>
</tr>
</tbody>
</table>

### H.2.17 ASFACT_WC_DISCUSS_B
Fact table that stores event information captured by Analytics when someone edits an announcement.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>APPLICATION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_WC_APPLICA_0.ID</td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WC_GROUPSP_0.ID</td>
</tr>
<tr>
<td>DISCUSSION_ANNOUNCEM</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Announcement ID dimension.</td>
<td>ASDIM_WC_DISCUSS_3.ID</td>
</tr>
</tbody>
</table>
### H.2.18 ASFACT_WC_DISCUSS_C

Fact table that stores event information captured by Analytics when someone deletes an announcement.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>DISCUSSION_ANNOUNCEMENT_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Announcement ID dimension.</td>
<td>ASDIM_WC_DISCUSS_3.ID</td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WC_GROUPSP_0.ID</td>
</tr>
<tr>
<td>APPLICATION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_WC_APPLICA_0.ID</td>
</tr>
</tbody>
</table>

### H.2.19 ASFACT_WC_DOCLIB__0

Fact table that stores event information captured by Analytics when someone views a document.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>APPLICATION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_WC_APPLICA_0.ID</td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WC_GROUPSP_0.ID</td>
</tr>
<tr>
<td>DISCUSSION_FORUM_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Discussion forum ID dimension.</td>
<td>ASDIM_WC_DISCUSS_1.ID</td>
</tr>
</tbody>
</table>
### H.2.20 ASFACT_WC_DOCLIB__1

Fact table that stores event information captured by Analytics when someone downloads a document.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>RESPONSE_TIME_</td>
<td>NULL</td>
<td>NUMBER(16)</td>
<td>Response Time (ms).</td>
<td></td>
</tr>
<tr>
<td>CLIENT_IP_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Client IP ID dimension.</td>
<td>ASDIM_WC_CLIENT_.ID</td>
</tr>
<tr>
<td>SESSION_ID_</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>Session ID.</td>
<td></td>
</tr>
<tr>
<td>GROUPSPACE</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WC_GROUPSP_.ID</td>
</tr>
<tr>
<td>REFFERER</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Reffer ID dimension.</td>
<td>ASDIM_WC_REFFERRE_.ID</td>
</tr>
<tr>
<td>DOCUMENT</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Document ID dimension.</td>
<td>ASDIM_WC_DOCUMENT_.ID</td>
</tr>
<tr>
<td>USER_AGENT</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User agent.</td>
<td>ASDIM_WC_USER_AG_.ID</td>
</tr>
<tr>
<td>APPLICATION</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_WC_APPLICA_.ID</td>
</tr>
</tbody>
</table>
### H.2.21 ASFACT_WC_DOCLIB__2
Fact table that stores event information captured by Analytics when someone creates document.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td>ASDIM_WC.Document_0.ID</td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM.TIME_ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM.USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS.EVENTFACTS.ID</td>
</tr>
<tr>
<td>APPLICATION</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM.WC.APPLICA_0.ID</td>
</tr>
<tr>
<td>WEBCENTER_RESOURCE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>WebCenter Portal resource ID.</td>
<td></td>
</tr>
<tr>
<td>GROUPSPACE</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM.WC.GROUPSP_0.ID</td>
</tr>
</tbody>
</table>

### H.2.22 ASFACT_WC_DOCLIB__3
Fact table that stores event information captured by Analytics when someone edits a document.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>DOCUMENT</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Document ID dimension.</td>
<td>ASDIM_WC_DOCUMENT_0.ID</td>
</tr>
<tr>
<td>REFERRER_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Referrer ID dimension.</td>
<td>ASDIM_WC_REFERRE_0.ID</td>
</tr>
<tr>
<td>CLIENT_IP_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Client IP ID dimension.</td>
<td>ASDIM_WC_CLIENT_0.ID</td>
</tr>
<tr>
<td>GROUPSPACE</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WC_GROUPSP_0.ID</td>
</tr>
<tr>
<td>USER_AGENT_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User agent.</td>
<td>ASDIM_WC_USER_AG_0.ID</td>
</tr>
<tr>
<td>SESSION_ID_</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>Session ID.</td>
<td></td>
</tr>
<tr>
<td>APPLICATION</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_WC_APPLICA_0.ID</td>
</tr>
<tr>
<td>WEBCENTER_RESOURCE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>WebCenter Portal resource ID.</td>
<td></td>
</tr>
<tr>
<td>GROUPSPACE</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WC_GROUPSP_0.ID</td>
</tr>
</tbody>
</table>
### H.2.23 ASFACT_WC_DOCLIB__4

Fact table that stores event information captured by Analytics when someone tags a document.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS EVENTFACTS.ID</td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM WC GROUPSP_0.ID</td>
</tr>
<tr>
<td>WEBCENTER_ RESOURCE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>WebCenter Portal resource ID.</td>
<td></td>
</tr>
<tr>
<td>APPLICATION</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM WC APPLICA_0.ID</td>
</tr>
<tr>
<td>TAGS</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Tag added to the document.</td>
<td></td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td></td>
</tr>
<tr>
<td>WEBCENTER_ RESOURCE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>WebCenter Portal resource ID.</td>
<td></td>
</tr>
<tr>
<td>APPLICATION</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td></td>
</tr>
</tbody>
</table>

### H.2.24 ASFACT_WC_DOCLIB__5

Fact table that stores event information captured by Analytics when someone "likes" a document.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS EVENTFACTS.ID</td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM WC GROUPSP_0.ID</td>
</tr>
<tr>
<td>WEBCENTER_ RESOURCE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>WebCenter Portal resource ID.</td>
<td></td>
</tr>
<tr>
<td>APPLICATION</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM WC APPLICA_0.ID</td>
</tr>
</tbody>
</table>
### H.2.25 ASFACT_WC_DOCLIB__6

Fact table that stores event information captured by Analytics when someone comments on a document.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>APPLICATION</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_WC_APPLICA_0.ID</td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WC_GROUPSP_0.ID</td>
</tr>
<tr>
<td>WEBCENTER_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>WebCenter Portal resource ID.</td>
<td></td>
</tr>
</tbody>
</table>

### H.2.26 ASFACT_WC_DOCLIB__7

Fact table that stores event information captured by Analytics when someone deletes a document.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
</tbody>
</table>

---

H-20 Developing Portals with Oracle WebCenter Portal and Oracle JDeveloper
### H.2.27 ASFACT_WC_EVENT_C_0

Fact table that stores event information captured by Analytics when someone creates an event in a portal.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>APPLICATION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_WC_APPLICATION_0.ID</td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WC_GROUPSP_0.ID</td>
</tr>
<tr>
<td>WEBCENTER_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>WebCenter Portal resource ID.</td>
<td></td>
</tr>
</tbody>
</table>

### H.2.28 ASFACT_WC_EVENT_D_0

Fact table that stores event information captured by Analytics when someone deletes an event from a portal.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>EVENT_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event Identifier.</td>
<td>ASDIM_WC_EVENTS_0.ID</td>
</tr>
<tr>
<td>APPLICATION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_WC_APPLICATION_0.ID</td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WC_GROUPSP_0.ID</td>
</tr>
</tbody>
</table>
### H.2.29 ASFACT_WC_EVENT_E_0

Fact table that stores event information captured by Analytics when someone edits and event in a portal.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.IID</td>
</tr>
<tr>
<td>APPLICATION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_WC_APPLICATION_0.ID</td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WC_GROUPSP_0.ID</td>
</tr>
<tr>
<td>EVENT_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event Identifier.</td>
<td>ASDIM_WC_EVENTS_0.ID</td>
</tr>
</tbody>
</table>

### H.2.30 ASFACT_WC_GROUPSP_0

Fact table that stores event information captured by Analytics when someone views a portal.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td></td>
</tr>
<tr>
<td>APPLICATION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td></td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td></td>
</tr>
<tr>
<td>EVENT_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event Identifier.</td>
<td></td>
</tr>
</tbody>
</table>
### H.2.31 ASFACT_WC_GROUPSP_1

Fact table that stores event information captured by Analytics when someone creates a portal.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WC_GROUPSP_0.ID</td>
</tr>
<tr>
<td>APPLICATION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_WC_APPLICATION_0.ID</td>
</tr>
</tbody>
</table>

### H.2.32 ASFACT_WC_GROUPSP_2

Fact table that stores event information captured by Analytics when someone joins a portal.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
</tbody>
</table>

WebCenter Portal Analytics Database Schema  H-23
### H.2.33 ASFACT_WC_GROUPSP_3

Fact table that stores event information captured by Analytics when someone deletes a portal.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WC_GROUPSP_0.ID</td>
</tr>
<tr>
<td>APPLICATION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_WC_APPICA_0.ID</td>
</tr>
</tbody>
</table>

### H.2.34 ASFACT_WC_LIST_CR_0

Fact table that stores event information captured by Analytics when someone creates a list.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
</tbody>
</table>
### H.2.35 ASFACT_WC_LIST_DE_0

Fact table that stores event information captured by Analytics when someone deletes a list.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>List ID.</td>
<td>ASDIM_WC_LISTS_0.ID</td>
</tr>
<tr>
<td>GROUPSPACE</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WC_GROUPSP_0.ID</td>
</tr>
<tr>
<td>APPLICATION</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_WC_APPLICA_0.ID</td>
</tr>
</tbody>
</table>

### H.2.36 ASFACT_WC_LIST_ED_0

Fact table that stores event information captured by Analytics when someone edits a list.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS_ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>APPLICATION</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_WC_APPLICA_0.ID</td>
</tr>
<tr>
<td>LIST_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>List ID.</td>
<td>ASDIM_WC_LISTS_0.ID</td>
</tr>
<tr>
<td>GROUPSPACE</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WC_GROUPSP_0.ID</td>
</tr>
</tbody>
</table>
**Analytics Fact Table Descriptions**

**H.2.37 ASFACT_WC_LOGINS_0**

Fact table that stores login event information captured by Analytics with related data for each event.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>APPLICATION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_WC_APPLICATION_0.ID</td>
</tr>
<tr>
<td>SESSION_ID_</td>
<td>NULL</td>
<td>NVARCHAR 2(254)</td>
<td>Session ID</td>
<td></td>
</tr>
<tr>
<td>USER_AGENT_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User agent.</td>
<td>ASDIM_WC_USER_AGENT_0.ID</td>
</tr>
<tr>
<td>REFERRE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Referrer ID dimension.</td>
<td>ASDIM_WC_REFERRE_0.ID</td>
</tr>
<tr>
<td>CLIENT_IP_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Client IP ID dimension.</td>
<td>ASDIM_WC_CLIENT_IP_0.ID</td>
</tr>
</tbody>
</table>

**H.2.38 ASFACT_WC_PEOPLEC_0**

Fact table that stores event information captured by Analytics when someone adds a people connection.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>TARGET_USER</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID</td>
<td></td>
</tr>
</tbody>
</table>
H.2.39 ASFACT_WC_PEOPLEC_1

Fact table that stores event information captured by Analytics when someone posts on a wall.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>TARGET_USER</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID.</td>
<td></td>
</tr>
</tbody>
</table>

H.2.40 ASFACT_WC_PEOPLEC_2

Fact table that stores event information captured by Analytics when someone edits a profile.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
</tbody>
</table>

H.2.41 ASFACT_WC_PEOPLEC_3

Fact table that stores event information captured by Analytics when someone edits status information.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
</tbody>
</table>
**H.2.42 ASFACT_WC_PORTLET_0**

Fact table that stores information captured by Analytics when portlets are accessed by end users.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>CACHED_</td>
<td>NULL</td>
<td>NUMBER(5)</td>
<td>Indicates whether the portlet was displayed from</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>PORTLET_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portlet.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>REFERRER_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Referrer ID dimension.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>PRODUCER_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portlet producer.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>PAGE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Page ID dimension.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>SESSION_ID_</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>Session ID.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>CLIENT_IP</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Client IP ID dimension.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>PORTLET_INSTANCE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portlet instance.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>RESPONSE_TIME_</td>
<td>NULL</td>
<td>NUMBER(16)</td>
<td>Response time (ms).</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>APPLICATION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USER_AGENT_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User agent.</td>
<td>ASDIM_TIME.ID</td>
</tr>
</tbody>
</table>
H.2.43 ASFACT_WC_SEARCHE_0

Fact table that stores event information captured by Analytics when someone performs a search.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>OCURRED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date event occurred.</td>
<td></td>
</tr>
<tr>
<td>TIMEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Time dimension event occurred. This column is used for partitioning.</td>
<td>ASDIM_TIME.ID</td>
</tr>
<tr>
<td>USERID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASSYS_EVENTFACTS.ID</td>
</tr>
<tr>
<td>PAGE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Page ID dimension.</td>
<td>ASDIM_WC_PAGES_0.ID</td>
</tr>
<tr>
<td>USER_AGENT_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User agent.</td>
<td>ASDIM_WC_USER_AG_0.ID</td>
</tr>
<tr>
<td>CLIENT_IP_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Client IP ID dimension.</td>
<td>ASDIM_WC_CLIENT_0.ID</td>
</tr>
<tr>
<td>APPLICATION_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Application ID dimension.</td>
<td>ASDIM_WC_APPLICA_0.ID</td>
</tr>
<tr>
<td>SESSION_ID_</td>
<td>NULL</td>
<td>NVARCHAR2 (254)</td>
<td>Session ID.</td>
<td></td>
</tr>
<tr>
<td>REFERREER_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Referrer ID dimension.</td>
<td>ASDIM_WC_REFERRE_0.ID</td>
</tr>
<tr>
<td>SEARCHED_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Search phrase.</td>
<td></td>
</tr>
<tr>
<td>GROUPSPACE_</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Portal ID dimension.</td>
<td>ASDIM_WC_GROUPSP_0.ID</td>
</tr>
</tbody>
</table>

H.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, portal name, and so on.

This section describes each of the tables listed in Table H–2, "Analytics Dimension Tables (Prefixed ASDIM_WC)".

H.3.1 ASDIM_WC_APPLICA_0

Dimension table that stores the names of WebCenter Portal applications accessing analytics data, for example webcenter (a portal).
### H.3.2 ASDIM_WC_CLIENT_0

Dimension table that stores client IP address details captured by Analytics when an analytics event is received.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
<td></td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record last modified.</td>
<td></td>
</tr>
<tr>
<td>IP</td>
<td>NOT NULL</td>
<td>NVARCHAR2(15)</td>
<td>Client IP address</td>
<td></td>
</tr>
</tbody>
</table>

### H.3.3 ASDIM_WC_DISCUSS_0

Dimension table that stores discussion topic details captured by Analytics when an analytics event is received.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
<td></td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record last modified.</td>
<td></td>
</tr>
<tr>
<td>ICONURL_</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>Icon URL</td>
<td></td>
</tr>
<tr>
<td>NAME_</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>Discussion topic name.</td>
<td></td>
</tr>
<tr>
<td>DESCRIPTION_</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>Discussion topic description.</td>
<td></td>
</tr>
<tr>
<td>RESOURCEID_</td>
<td>NOT NULL</td>
<td>NVARCHAR2(254)</td>
<td>WebCenter Portal resource ID.</td>
<td></td>
</tr>
</tbody>
</table>

### H.3.4 ASDIM_WC_DISCUSS_1

Dimension table that stores discussion forum details captured by Analytics when an analytics event is received.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
<td></td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record last modified.</td>
<td></td>
</tr>
<tr>
<td>ICONURL_</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>Icon URL</td>
<td></td>
</tr>
</tbody>
</table>
### H.3.5 ASDIM_WC_DISCUSS_2

Dimension table that stores discussion message details captured by Analytics when an analytics event is received.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESCRIPTION_</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>Discussion forum description.</td>
<td></td>
</tr>
<tr>
<td>RESOURCEID_</td>
<td>NOT NULL</td>
<td>NVARCHAR2(254)</td>
<td>WebCenter Portal resource ID.</td>
<td></td>
</tr>
<tr>
<td>NAME_</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>WebCenter Portal resource name, that is, the name of the discussion forum.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
<td></td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date records last modified.</td>
<td></td>
</tr>
<tr>
<td>NAME_</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>WebCenter Portal resource name, that is, the name of the discussion message.</td>
<td></td>
</tr>
<tr>
<td>RESOURCEID_</td>
<td>NOT NULL</td>
<td>NVARCHAR2(254)</td>
<td>WebCenter Portal resource ID.</td>
<td></td>
</tr>
<tr>
<td>DESCRIPTION_</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>Discussion message description.</td>
<td></td>
</tr>
<tr>
<td>ICONURL_</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>Icon URL</td>
<td></td>
</tr>
</tbody>
</table>

### H.3.6 ASDIM_WC_DISCUSS_3

Dimension table that stores announcement details captured by Analytics when an analytics event is received.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
<td></td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record last modified.</td>
<td></td>
</tr>
<tr>
<td>DESCRIPTION_</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>Announcement description.</td>
<td></td>
</tr>
<tr>
<td>ICONURL_</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>Icon URL</td>
<td></td>
</tr>
<tr>
<td>RESOURCEID_</td>
<td>NOT NULL</td>
<td>NVARCHAR2(254)</td>
<td>WebCenter Portal resource ID.</td>
<td></td>
</tr>
</tbody>
</table>
### H.3.7 ASDIM_WC_DOCUMENT_0

Dimension table that stores information captured by Analytics when documents are accessed by end users.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record last modified.</td>
</tr>
<tr>
<td>CONTENTTYPE</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>Document content type.</td>
</tr>
<tr>
<td>CONTEXT</td>
<td>NULL</td>
<td>NVARCHAR2(512)</td>
<td>Document prefix (for identification purposes):</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Document prefix - Folder URL for parent portal</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Blog prefix - Blog title name</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Wiki prefix - Wiki title name</td>
</tr>
<tr>
<td>NAME_</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>Document name.</td>
</tr>
<tr>
<td>RESOURCEID_</td>
<td>NOT NULL</td>
<td>NVARCHAR2(254)</td>
<td>WebCenter Portal resource ID.</td>
</tr>
<tr>
<td>ICONURL_</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>Icon URL</td>
</tr>
<tr>
<td>PATH_</td>
<td>NULL</td>
<td>NVARCHAR2(512)</td>
<td>Document path.</td>
</tr>
<tr>
<td>OBJECTTYPE_</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>Object type.</td>
</tr>
</tbody>
</table>

### H.3.8 ASDIM_WC_EVENTS_0

Dimension table that stores information about portal events captured by Analytics when a user creates an event in his or her calendar.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record last modified.</td>
</tr>
<tr>
<td>RESOURCEID_</td>
<td>NOT NULL</td>
<td>NVARCHAR2(254)</td>
<td>WebCenter Portal resource ID.</td>
</tr>
<tr>
<td>Column Name</td>
<td>NOT NULL?</td>
<td>Data Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------</td>
<td>-----------------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>NAME_</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>WebCenter Portal resource name, that is, the name of the portal event.</td>
</tr>
</tbody>
</table>

**H.3.9 ASDIM_WC_GROUPSP_0**

Dimension table that stores information captured by Analytics when a portal is accessed by an end user (WebCenter Portal only).

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record last modified.</td>
</tr>
<tr>
<td>ICONURL_</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>Icon URL</td>
</tr>
<tr>
<td>DESCRIPTION_</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>Description.</td>
</tr>
<tr>
<td>PERSONAL_</td>
<td>NULL</td>
<td>NUMBER(5)</td>
<td>Indicates whether the portal is the Home portal (0) or a group space (1).</td>
</tr>
<tr>
<td>RESOURCEID_</td>
<td>NOT NULL</td>
<td>NVARCHAR2(254)</td>
<td>WebCenter Portal resource ID.</td>
</tr>
<tr>
<td>NAME_</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>Name of the portal.</td>
</tr>
</tbody>
</table>

**H.3.10 ASDIM_WC_LISTS_0**

Dimension table that stores list details captured by Analytics when an analytics event is received.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record last modified.</td>
</tr>
<tr>
<td>NAME_</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>WebCenter Portal resource name, that is, the name of the list.</td>
</tr>
<tr>
<td>RESOURCEID_</td>
<td>NOT NULL</td>
<td>NVARCHAR2(254)</td>
<td>WebCenter Portal resource ID.</td>
</tr>
</tbody>
</table>

**H.3.11 ASDIM_WC_PAGES_0**

Dimension table that stores information captured by Analytics when pages are accessed by end users.
<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
<td></td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record last modified.</td>
<td></td>
</tr>
<tr>
<td>PERSONAL_</td>
<td>NULL</td>
<td>NUMBER(5)</td>
<td>Indicates whether the page is a personal page (in the Home portal) or belongs to a portal.</td>
<td></td>
</tr>
<tr>
<td>RESOURCEID_</td>
<td>NOT NULL</td>
<td>NVARCHAR2(512)</td>
<td>WebCenter Portal resource ID. The full page path, for example, /oracle/webcenter/page/sda254765_f950/Home.jspx</td>
<td></td>
</tr>
<tr>
<td>NAME_</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>Name of the page.</td>
<td></td>
</tr>
</tbody>
</table>

**H.3.12 ASDIM_WC_PORTLET_0**
Dimension table that stores information captured by Analytics when portlets are accessed by end users.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
<td></td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record last modified.</td>
<td></td>
</tr>
<tr>
<td>DEFAULT__TITLE__</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>Default portlet title.</td>
<td></td>
</tr>
<tr>
<td>RESOURCEID_</td>
<td>NOT NULL</td>
<td>NVARCHAR2(512)</td>
<td>WebCenter Portal resource ID.</td>
<td></td>
</tr>
</tbody>
</table>

**H.3.13 ASDIM_WC_PORTLET_1**
Dimension table that stores portlet instance information captured by Analytics when an analytics event is received.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
<td></td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record last modified.</td>
<td></td>
</tr>
<tr>
<td>TITLE_</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>Portlet title.</td>
<td></td>
</tr>
<tr>
<td>RESOURCEID_</td>
<td>NOT NULL</td>
<td>NVARCHAR2(512)</td>
<td>WebCenter Portal resource ID.</td>
<td></td>
</tr>
</tbody>
</table>
H.3.14 ASDIM_WC_PRODUCE_0
Dimension table that stores portlet producer information captured by Analytics when an analytics event is received.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
<td></td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record modified.</td>
<td></td>
</tr>
<tr>
<td>RESOURCEID_</td>
<td>NOT NULL</td>
<td>NVARCHAR2(512)</td>
<td>WebCenter Portal resource ID.</td>
<td></td>
</tr>
<tr>
<td>NAME_</td>
<td>NULL</td>
<td>NVARCHAR2(254)</td>
<td>Producer name.</td>
<td></td>
</tr>
</tbody>
</table>

H.3.15 ASDIM_WC_REFERRE_0
Dimension table that stores information captured by Analytics related to the location (URL) to which the user was navigating when the analytics event was captured.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
<td></td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record last modified.</td>
<td></td>
</tr>
<tr>
<td>URL</td>
<td>NOT NULL</td>
<td>NVARCHAR2(2000)</td>
<td>Referred URL.</td>
<td></td>
</tr>
</tbody>
</table>

H.3.16 ASDIM_WC_SEARCHE_0
Dimension table that stores search phrases captured by Analytics when an analytics event is received.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
<td></td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record last modified.</td>
<td></td>
</tr>
<tr>
<td>PHRASE</td>
<td>NOT NULL</td>
<td>NVARCHAR2(254)</td>
<td>Search phrase.</td>
<td></td>
</tr>
</tbody>
</table>

H.3.17 ASDIM_WC_TAGS_0
Dimension table that stores tag information captured by Analytics when an analytics event is received.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
<td></td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record last modified.</td>
<td></td>
</tr>
</tbody>
</table>
### H.3.18 ASDIM_WC_USER_AG_0

Dimension table that stores information about which browsers are used to access analytics data.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>NOT NULL</td>
<td>NVARCHAR2(254)</td>
<td>Tag value.</td>
<td></td>
</tr>
</tbody>
</table>

### H.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.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
<td></td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record last modified.</td>
<td></td>
</tr>
<tr>
<td>OS</td>
<td>NULL</td>
<td>NVARCHAR2(100)</td>
<td>Operating system.</td>
<td></td>
</tr>
<tr>
<td>AGENT</td>
<td>NOT NULL</td>
<td>NVARCHAR2(512)</td>
<td>User agent ID.</td>
<td></td>
</tr>
<tr>
<td>BROWSER</td>
<td>NULL</td>
<td>NVARCHAR2(255)</td>
<td>Name of the browser.</td>
<td></td>
</tr>
<tr>
<td>BROWSER_VERSION</td>
<td>NULL</td>
<td>NVARCHAR2(30)</td>
<td>Browser version.</td>
<td></td>
</tr>
</tbody>
</table>
### H.3.20 ASDIM_USERS
Dimension table that stores user details captured by Analytics when an analytics event is received.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>USERID</td>
<td>NOT NULL</td>
<td>NVARCHAR2(255)</td>
<td>User ID.</td>
<td></td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
<td></td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record last modified.</td>
<td></td>
</tr>
</tbody>
</table>

For example: `weblogic`

### H.3.21 ASDIM_USERPROPERTIES
Dimension table that stores user properties, such as Title, Department, and so on.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>PROPERTYID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Property ID.</td>
<td></td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
<td></td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NULL</td>
<td>DATE</td>
<td>Date record last modified.</td>
<td></td>
</tr>
<tr>
<td>NAME</td>
<td>NOT NULL</td>
<td>NVARCHAR2(255)</td>
<td>Property name.</td>
<td></td>
</tr>
</tbody>
</table>

### H.3.22 ASDIM_USERPROPERTYVALUES
Dimension table that stores values for user properties.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
<td></td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record last modified.</td>
<td></td>
</tr>
<tr>
<td>USERID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>User ID dimension.</td>
<td>ASDIM_USERS.ID</td>
</tr>
<tr>
<td>PROPERTYID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Property ID.</td>
<td>ASDIM_USERPROPERTIES.ID</td>
</tr>
<tr>
<td>VALUE</td>
<td>NULL</td>
<td>NVARCHAR2(255)</td>
<td>Property ID value.</td>
<td></td>
</tr>
<tr>
<td>TYPE</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Property type.</td>
<td></td>
</tr>
</tbody>
</table>

### H.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.
H.4.1 ASSYS_CONFIG

System table that stores system configuration information. Each row contains value pairs—configuration type/value.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record last modified.</td>
</tr>
<tr>
<td>NAME</td>
<td>NULL</td>
<td>VARCHAR2(510)</td>
<td>Name of configuration parameter, for example,</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>collector_port.</td>
</tr>
<tr>
<td>CONFIGTYPEID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Configuration type ID.</td>
</tr>
<tr>
<td>INTVAL</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Integer value associated with the configuration.</td>
</tr>
<tr>
<td>STRVAL</td>
<td>NULL</td>
<td>VARCHAR2(2000)</td>
<td>String value associated with the configuration.</td>
</tr>
</tbody>
</table>

H.4.2 ASSYS_DIMENSIONPROPS

System table that stores dimension configuration information. Each row represents property values for a dimension.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record last modified.</td>
</tr>
<tr>
<td>DIMENSIONID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Dimension ID.</td>
</tr>
<tr>
<td>DISPLAYNAME</td>
<td>NOT NULL</td>
<td>VARCHAR2(510)</td>
<td>Dimension property display name.</td>
</tr>
<tr>
<td>COLUMNNAME</td>
<td>NOT NULL</td>
<td>VARCHAR2(510)</td>
<td>Column name.</td>
</tr>
<tr>
<td>COLUMNTYPE</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Column type.</td>
</tr>
<tr>
<td>COLUMNLENGTH</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Column length.</td>
</tr>
<tr>
<td>ISKEY</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Indicates whether this is a key dimension property.</td>
</tr>
</tbody>
</table>

H.4.3 ASSYS_EVENTDIMENSIONS

System table that stores dimensions associated with each analytics event.
### H.4.4 ASSYS_EVENTFACTS

System table that stores facts associated with each analytics event.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
<td></td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record last modified.</td>
<td></td>
</tr>
<tr>
<td>DISPLAYNAME</td>
<td>NOT NULL</td>
<td>NVARCHAR2(510)</td>
<td>Dimension event display name.</td>
<td></td>
</tr>
<tr>
<td>TABLENAME</td>
<td>NOT NULL</td>
<td>NVARCHAR2(510)</td>
<td>Table name associated with the dimension.</td>
<td></td>
</tr>
<tr>
<td>ISUNIQUE</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Deprecated.</td>
<td></td>
</tr>
<tr>
<td>IDENTIFYINGUSER</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>User ID.</td>
<td>ASDIM_USERS.ID</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
<td></td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record last modified.</td>
<td></td>
</tr>
<tr>
<td>EVENTID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Event ID dimension.</td>
<td>ASDIM_WC_EVENTS_0.ID</td>
</tr>
<tr>
<td>DISPLAYNAME</td>
<td>NOT NULL</td>
<td>NVARCHAR2(510)</td>
<td>Event fact display name.</td>
<td></td>
</tr>
<tr>
<td>COLUMNNAME</td>
<td>NOT NULL</td>
<td>NVARCHAR2(510)</td>
<td>Event fact column name.</td>
<td></td>
</tr>
<tr>
<td>COLUMNTYPEID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>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.</td>
<td></td>
</tr>
<tr>
<td>COLUMNLNGTH</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>User ID.</td>
<td></td>
</tr>
<tr>
<td>EVENTDIMENSIONID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Dimension ID.</td>
<td>ASSYS_EVENTDIMENSIONS.ID</td>
</tr>
</tbody>
</table>
H.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.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
<td></td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record last modified.</td>
<td></td>
</tr>
<tr>
<td>DISPLAYNAME</td>
<td>NOT NULL</td>
<td>NVARCHAR2(510)</td>
<td>Event fact display name.</td>
<td></td>
</tr>
<tr>
<td>DESCRIPTION</td>
<td>NULL</td>
<td>NVARCHAR2(510)</td>
<td>Event description.</td>
<td></td>
</tr>
<tr>
<td>TABLENAME</td>
<td>NOT NULL</td>
<td>NVARCHAR2(510)</td>
<td>Table name.</td>
<td></td>
</tr>
<tr>
<td>ISCUSTOM</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Indicates whether or not the event is a custom event. (0/1)</td>
<td></td>
</tr>
<tr>
<td>ISACTIVE</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Deprecated.</td>
<td></td>
</tr>
<tr>
<td>ISPERSISTENT</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Indicates whether or not to create a fact table for the event. (0/1)</td>
<td></td>
</tr>
<tr>
<td>ISACTION</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Not used.</td>
<td></td>
</tr>
</tbody>
</table>

H.4.6 ASSYS_NAMESPACES

System table that stores the different namespaces in which analytics events are registered.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
<td></td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record last modified.</td>
<td></td>
</tr>
<tr>
<td>PREFIX</td>
<td>NOT NULL</td>
<td>NVARCHAR2(510)</td>
<td>Namespace prefix.</td>
<td></td>
</tr>
<tr>
<td>URI</td>
<td>NOT NULL</td>
<td>NVARCHAR2(510)</td>
<td>Universal resource identifier for the namespace.</td>
<td></td>
</tr>
</tbody>
</table>

H.4.7 ASSYS_REPORTCOMPOSITION

System table that stores the composition for reports provided by Analytics.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
<td></td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record last modified.</td>
<td></td>
</tr>
</tbody>
</table>
H.4.8 ASSYS_REPORTGROUP

System table that stores the report groups included for Analytics. Report groups enable end users to display analytics data in a specific way.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
<td></td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record was last modified.</td>
<td></td>
</tr>
<tr>
<td>NAME</td>
<td>NOT NULL</td>
<td>NVARCHAR2(510)</td>
<td>Report group name.</td>
<td></td>
</tr>
<tr>
<td>DESCRIPTION</td>
<td>NULL</td>
<td>NVARCHAR2(510)</td>
<td>Report group description.</td>
<td></td>
</tr>
<tr>
<td>SEQUENCE</td>
<td>NULL</td>
<td>NUMBER(10)</td>
<td>Sequence.</td>
<td></td>
</tr>
</tbody>
</table>

H.4.9 ASSYS_REPORTITEMS

System table that stores item information in each report. Each row represents a specific item belonging to one report.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
<td></td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record last modified.</td>
<td></td>
</tr>
<tr>
<td>REPORTID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Report ID.</td>
<td>ASSYS_REPORTS.ID</td>
</tr>
<tr>
<td>ITEMTYPE</td>
<td>NOT NULL</td>
<td>VARCHAR2(30)</td>
<td>Item type.</td>
<td></td>
</tr>
<tr>
<td>ITEMDESCRIPTION</td>
<td>NULL</td>
<td>NVARCHAR2(510)</td>
<td>Item description.</td>
<td></td>
</tr>
<tr>
<td>ITEMVALUE</td>
<td>NULL</td>
<td>NUMBER(5)</td>
<td>Item value.</td>
<td></td>
</tr>
<tr>
<td>EVENTDIMENSIONID</td>
<td>NULL</td>
<td>NUMBER(38)</td>
<td>Dimension ID.</td>
<td>ASSYS_EVENTDIMENSIONS.ID</td>
</tr>
</tbody>
</table>
### H.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.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NOT NULL</td>
<td>NUMBER(38)</td>
<td>Internal table ID.</td>
<td></td>
</tr>
<tr>
<td>CREATED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record created.</td>
<td></td>
</tr>
<tr>
<td>LASTMODIFIED</td>
<td>NOT NULL</td>
<td>DATE</td>
<td>Date record last modified.</td>
<td></td>
</tr>
<tr>
<td>REPORTITEMID</td>
<td>NOT NULL</td>
<td>NUMBER</td>
<td>Report item ID.</td>
<td>ASSYS_REPORTITEMS.ID</td>
</tr>
<tr>
<td>VALUETYPE</td>
<td>NOT NULL</td>
<td>VARCHAR2(30)</td>
<td>Value type: CVALUE, NVALUE or DVALUE.</td>
<td></td>
</tr>
</tbody>
</table>
## Analytics System Table Descriptions

### H.4.11 ASSYS_REPORTS

System table that stores the different out-of-the-box reports to be provided by Analytics. Each row represents a single report and the report’s configuration.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL?</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>CVALUE</td>
<td>NULL</td>
<td>NVARCHAR2(510)</td>
<td>Char value.*</td>
<td></td>
</tr>
<tr>
<td>NVALUE</td>
<td>NULL</td>
<td>NUMBER</td>
<td>Numeric value.*</td>
<td></td>
</tr>
<tr>
<td>DVALUE</td>
<td>NULL</td>
<td>DATE</td>
<td>Date value.*</td>
<td></td>
</tr>
</tbody>
</table>

*Only one of CVALUE, NVALUE or DVALUE has a value.*

### Column Descriptions

- **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.
- **EVENTTYPEID**: NULL, NUMBER(38) — Associated event ID.
- **EVENTDISPLAYNAME**: NULL, NVARCHAR2(510) — Denormalized. Display name of associated event.
- **GROUPSPACEFILTERENABLED**: NOT NULL, VARCHAR2(1) — Indicates whether users can filter the report 'by portal'. If set to true (1), the report displays the portal 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.
- **DEFAULTGROUPBYTIMEINTERVAL**: NULL, NUMBER(2) — Default group time interval.
- **DEFAULTGROUPBYUSERPROPERTY**: NULL, NUMBER(38) — Default "group by user" property.
<table>
<thead>
<tr>
<th>Column Name</th>
<th>NOT NULL</th>
<th>Data Type</th>
<th>Description</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>SHOWMETRICSELECTOR</td>
<td>NOT NULL</td>
<td>NVARCHAR2(1)</td>
<td>Indicates whether or not to display selector UI for this metric. (Y/N)</td>
<td></td>
</tr>
<tr>
<td>SHOWCHARTYPEOPTIONS</td>
<td>NOT NULL</td>
<td>NVARCHAR2(1)</td>
<td>Indicates whether or not to display chart type options in the UI. (Y/N)</td>
<td></td>
</tr>
<tr>
<td>SHOWDISPLAYOPTIONS</td>
<td>NOT NULL</td>
<td>NVARCHAR2(1)</td>
<td>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.)</td>
<td></td>
</tr>
<tr>
<td>SHOWDISPLAYTOPOPTION</td>
<td>NOT NULL</td>
<td>NVARCHAR2(1)</td>
<td>Indicates whether or not users will be able to select the &quot;Top n&quot; option. (Y/N) If N, the &quot;Top n&quot; option will not be available in the combo box.</td>
<td></td>
</tr>
<tr>
<td>SHOWDISPLAYBOTTOMOPTION</td>
<td>NOT NULL</td>
<td>NVARCHAR2(1)</td>
<td>Indicates whether or not users will be able to select the &quot;Bottom n&quot; option. (Y/N) If N, the &quot;Bottom n&quot; option will not be available in the combo box.</td>
<td></td>
</tr>
<tr>
<td>SHOWDISPLAYALLOPTION</td>
<td>NOT NULL</td>
<td>NVARCHAR2(1)</td>
<td>Indicates whether or not users will be able to select the &quot;All records&quot; option. (Y/N) If N, the &quot;All records&quot; 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.</td>
<td></td>
</tr>
<tr>
<td>SHOWOBJECTPICKER</td>
<td>NOT NULL</td>
<td>NVARCHAR2(1)</td>
<td>Indicates whether or not users will be able to select a list of specific records option. (Y/N) If N, the &quot;Select ...&quot; option will not be available in the combo box. If Y the &quot;Select ...&quot; option will be available and a picker dialog will be displayed to select the list of objects to be included in the report.</td>
<td></td>
</tr>
<tr>
<td>SHOWTIMEFRAMEFILTER</td>
<td>NOT NULL</td>
<td>NVARCHAR2(1)</td>
<td>Indicates whether or not to display time frame filter. (Y/N)</td>
<td></td>
</tr>
</tbody>
</table>
H.5 Analytics User Properties

Analytics uses several tables to store information about users who are triggering analytics events in WebCenter Portal applications. This section contains the following topics:

- Section H.5.1, "User Property Tables - Overview"
- Section H.5.2, "User Property Dimension Tables"
- Section H.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 Analytics."

### H.5.1 User Property Tables - Overview

Figure H–5 shows the database model used by Analytics 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 H–5  User Property Dimensions for Analytics**

The ASDIM_USERPROPERTIES table stores the names of user properties received by Analytics. User properties (attributes of the user including name, position, email, address, etc.) are captured by Analytics as an event when a user logs in to WebCenter Portal, 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.
H.5.2 User Property Dimension Tables
User property dimension tables are described in the following topics:

- Section H.5.2.1, "ASDIM_USERPROPERTIES"
- Section H.5.2.2, "ASDIM_USERPROPERTYVALUES"

H.5.2.1 ASDIM_USERPROPERTIES
Dimension table that stores the names of user properties received by Analytics.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NUMBER(38,0)</td>
<td>Table primary key. Value obtained from the sequence.</td>
</tr>
<tr>
<td>NAME</td>
<td>VARCHAR(255)</td>
<td>Name of the user property. Names are converted to uppercase and whitespace characters are removed.</td>
</tr>
</tbody>
</table>

H.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.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NUMBER(38,0)</td>
<td>Table primary key. Value obtained from the sequence.</td>
</tr>
<tr>
<td>USERID</td>
<td>NUMBER(38,0)</td>
<td>Foreign key to the ASDIM_USERS table.</td>
</tr>
<tr>
<td>PROPERTYID</td>
<td>NUMBER(38,0)</td>
<td>Foreign key to the ASDIM_USERPROPERTIES table.</td>
</tr>
<tr>
<td>VALUE</td>
<td>VARCHAR(255)</td>
<td>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 &quot;CITY&quot; and the user &quot;weblogic&quot;, this column will be storing the city name where that user works (for example &quot;San Francisco&quot;).</td>
</tr>
<tr>
<td>TYPE</td>
<td>NUMBER(38,0)</td>
<td>User property value type. (Property values are always stored as strings.)</td>
</tr>
</tbody>
</table>

H.5.3 Out-of-the-Box User Properties for WebCenter Portal
WebCenter Portal applications send a specific set of user properties to Analytics, as shown in Table H–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 H–4 Out-of-the-Box User Properties for WebCenter Portal

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Property Label (displayed in reports)</th>
<th>Example Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMUSER</td>
<td>IM User</td>
<td>dfrabott</td>
</tr>
<tr>
<td>DISPLAYNAME</td>
<td>Display</td>
<td>Diego Frabotta</td>
</tr>
<tr>
<td>PHONE</td>
<td>Phone</td>
<td>+5411-2222-1234</td>
</tr>
<tr>
<td>TITLE</td>
<td>Title</td>
<td>Software Engineer</td>
</tr>
<tr>
<td>DEPARTMENT</td>
<td>Department</td>
<td>Analytics</td>
</tr>
<tr>
<td>MANAGER</td>
<td>Manager</td>
<td>uid=dsabarisi,cn=users,dc=us,dc=myco,dc=com</td>
</tr>
<tr>
<td>COMPANY</td>
<td>Company</td>
<td>MyCo</td>
</tr>
<tr>
<td>STREET</td>
<td>Street</td>
<td>Parque Austral, Edificio Insignia M3</td>
</tr>
<tr>
<td>ZIPCODE</td>
<td>ZIP Code</td>
<td>12345</td>
</tr>
<tr>
<td>STATEORPROVINCE</td>
<td>State or Province</td>
<td>Buenos Aires</td>
</tr>
<tr>
<td>COUNTRY</td>
<td>Country</td>
<td>Argentina</td>
</tr>
<tr>
<td>EMPLOYEEID</td>
<td>Employee Id</td>
<td>55555</td>
</tr>
<tr>
<td>CITY</td>
<td>City</td>
<td>Pilar</td>
</tr>
</tbody>
</table>

### H.6 Analytics Event and Dimension Data Types

Table H–5 shows how analytics event and dimension data types map to data types in Oracle, Microsoft SQL Server, and IBM DB2 databases.

### Table H–5 Analytics Event and Dimension Data Types

<table>
<thead>
<tr>
<th>Analytics Event and Dimension Type</th>
<th>Oracle</th>
<th>Microsoft SQL Server</th>
<th>IBM DB2</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTEGER</td>
<td>NUMBER</td>
<td>BIGINT</td>
<td>INTEGER</td>
</tr>
<tr>
<td>STRING</td>
<td>NVARCHAR</td>
<td>NVARCHAR</td>
<td>VARGRAPHIC</td>
</tr>
<tr>
<td>DATE</td>
<td>DATE</td>
<td>DATETIME</td>
<td>TIMESTAMP</td>
</tr>
<tr>
<td>FLOAT</td>
<td>FLOAT</td>
<td>FLOAT</td>
<td>DECFLOAT</td>
</tr>
<tr>
<td>BOOLEAN</td>
<td>NUMBER(5)</td>
<td>TINYINT</td>
<td>SMALLINT</td>
</tr>
</tbody>
</table>
This appendix describes the accessibility features of WebCenter Portal.

This appendix contains the following topic:

- Section I.1, "Overview of WebCenter Accessibility Features"

### I.1 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](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 "Specifying Component-Level Accessibility Properties" section in [Web User Interface Developer’s Guide for Oracle Application Development Framework](http://www.oracle.com). 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.

### I.1.1 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 I–1.

<table>
<thead>
<tr>
<th>Component</th>
<th>Accessibility Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>pe:changeModeButton</td>
<td>No accessibility attributes.</td>
</tr>
<tr>
<td>pe:changeModeLink</td>
<td>No accessibility attributes.</td>
</tr>
<tr>
<td>pe:imageLink</td>
<td>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.</td>
</tr>
<tr>
<td>pe:pageCustomizable</td>
<td>No accessibility attributes.</td>
</tr>
</tbody>
</table>
I.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 I–1. For a list of accessibility-related attributes for other components, see the "Specifying Component-Level Accessibility Properties" section in Web User Interface Developer’s Guide for Oracle Application Development Framework.

I.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.

<table>
<thead>
<tr>
<th>Component</th>
<th>Accessibility Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>pe:layoutCustomizable</td>
<td>shortDesc—Mandatory. This attribute transforms into an HTML alt attribute.</td>
</tr>
<tr>
<td></td>
<td>accessKey—Optional. This attribute sets the mnemonic character used to gain quick access to the component.</td>
</tr>
<tr>
<td>cust:panelCustomizable</td>
<td>No accessibility attributes</td>
</tr>
<tr>
<td>cust:showDetailFrame</td>
<td>shortDesc—Mandatory. This attribute transforms into an HTML alt attribute.</td>
</tr>
</tbody>
</table>

**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 63.5.4, "What You May Need to Know About Inline Frames."