Oracle® Fusion Middleware
Developer's Guide for Oracle WebCenter
11g Release 1 (11.1.1)
E10148-07

August 2010
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preface</td>
<td>li</td>
</tr>
<tr>
<td>Audience</td>
<td>li</td>
</tr>
<tr>
<td>Documentation Accessibility</td>
<td>lii</td>
</tr>
<tr>
<td>Related Documents</td>
<td>lii</td>
</tr>
<tr>
<td>Conventions</td>
<td>lii</td>
</tr>
<tr>
<td>Part I Introduction</td>
<td></td>
</tr>
<tr>
<td>1 Understanding Oracle WebCenter</td>
<td></td>
</tr>
<tr>
<td>1.1 Introduction to Oracle WebCenter</td>
<td>1-1</td>
</tr>
<tr>
<td>1.1.1 Introduction to Oracle Application Development Frame</td>
<td>1-2</td>
</tr>
<tr>
<td>1.1.2 Introduction to Oracle JDeveloper and the WebCenter Extension</td>
<td>1-3</td>
</tr>
<tr>
<td>1.1.3 Introduction to Oracle WebCenter Framework and Custom WebCenter Applications...</td>
<td>1-4</td>
</tr>
<tr>
<td>1.1.3.1 Oracle Composer</td>
<td>1-5</td>
</tr>
<tr>
<td>1.1.3.2 Portlets</td>
<td>1-6</td>
</tr>
<tr>
<td>1.1.3.3 Content Integration</td>
<td>1-7</td>
</tr>
<tr>
<td>1.1.3.4 Search Framework</td>
<td>1-7</td>
</tr>
<tr>
<td>1.1.3.5 Resource Catalog</td>
<td>1-7</td>
</tr>
<tr>
<td>1.1.3.6 Customizable Components</td>
<td>1-8</td>
</tr>
<tr>
<td>1.1.3.7 Application Security</td>
<td>1-8</td>
</tr>
<tr>
<td>1.1.3.8 Application Life Cycle</td>
<td>1-9</td>
</tr>
<tr>
<td>1.1.4 Introduction to Oracle WebCenter Services</td>
<td>1-10</td>
</tr>
<tr>
<td>1.2 Design Questions to Consider Before You Start</td>
<td>1-12</td>
</tr>
<tr>
<td>1.2.1 User Considerations</td>
<td>1-12</td>
</tr>
<tr>
<td>1.2.2 Site Administrator Considerations</td>
<td>1-13</td>
</tr>
<tr>
<td>1.2.3 Developer Considerations</td>
<td>1-14</td>
</tr>
<tr>
<td>1.3 Accessibility Features</td>
<td>1-15</td>
</tr>
<tr>
<td>1.3.1 Generating Accessible HTML</td>
<td>1-16</td>
</tr>
<tr>
<td>1.3.2 Accessibility Features at Runtime</td>
<td>1-16</td>
</tr>
<tr>
<td>1.3.3 Accessibility Considerations for Portlets</td>
<td>1-17</td>
</tr>
<tr>
<td>1.3.4 Accessibility Features in WebCenter Spaces</td>
<td>1-17</td>
</tr>
</tbody>
</table>
4 Preparing Your Application for Oracle WebCenter Services

4.1 Preparing Your WebCenter Application to Consume Services ........................................... 4-1
4.1.1 How to Prepare Your Application to Consume Services ............................................. 4-1
4.1.1.1 Implementing Security for Services ........................................................................ 4-2
4.1.1.2 Setting Up SSL-Protected Connections for Services ................................................ 4-3
4.1.2 Setting Up an External Application Connection .......................................................... 4-5
4.1.3 Automated Task Flow Grants ....................................................................................... 4-5
4.2 Extending Your Application with Custom Components ................................................... 4-9
4.2.1 Introducing the Resource Action Handling Framework .............................................. 4-9
4.2.2 Registering a Resource Viewer ................................................................................... 4-10
4.3 Configuring General Settings for Your Services .............................................................. 4-15
4.3.1 Building a Preferences User Interface ......................................................................... 4-17

Part II Using, Extending, and Customizing Your Application with Oracle Composer

5 Introduction to Oracle Composer

5.1 Personalization and Customization .................................................................................. 5-2
5.2 View and Edit Modes of a Page ....................................................................................... 5-2
5.3 Personalizing Capabilities in Page View Mode ................................................................. 5-3
5.3.1 Rearrange Components ............................................................................................... 5-3
5.3.2 Change the Layout ..................................................................................................... 5-4
5.3.3 Expand and Collapse Components ............................................................................. 5-5
5.3.4 Customize Portlets ..................................................................................................... 5-6
5.4 Editing Capabilities in Design View in Page Edit Mode ........................................................ 5-7
5.4.1 Add Content .............................................................................................................. 5-9
5.4.2 Add Box Components Adjacent to Existing Ones ........................................................ 5-12
5.4.2.1 Add Box Above ..................................................................................................... 5-13
5.4.2.2 Add Box Below ................................................................................................... 5-14
5.4.2.3 Add Box Left ....................................................................................................... 5-15
5.4.2.4 Add Box Right .................................................................................................... 5-15
5.4.3 Rearrange Page Content ............................................................................................. 5-16
5.4.4 Edit Component Properties and Parameters ............................................................... 5-17
5.4.5 Reset and Override Component Properties in the Current Layer ................................ 5-20
5.4.6 Edit Resource Strings ............................................................................................... 5-21
5.4.7 Show or Hide Components ....................................................................................... 5-26
5.4.8 Delete Components ................................................................................................... 5-28
5.4.9 Change the Layout..................................................................................................... 5-28
5.4.10 Edit Page Properties ................................................................................................ 5-29
5.4.11 Wire Components to Page Parameters ...................................................................... 5-30
5.4.12 Reset Page ................................................................................................................ 5-30
5.4.13 Reset Customizations on Detecting Errors ............................................................... 5-32
5.4.14 Create Labels On Saving Customizations ................................................................. 5-32
5.4.15 Manage Customizations .......................................................................................... 5-33
5.5 Editing Capabilities in Source View in Page Edit Mode .................................................... 5-37
5.5.1 Rearrange Page Content Using Cut and Paste Options .............................................. 5-40
6 Enabling Runtime Creation and Management of Pages

6.1 Introduction to Page Creation and Management .......................................... 6-1
6.2 Creating Pages and Task Flows ................................................................. 6-2
   6.2.1 How to Create Pages ................................................................. 6-3
       6.2.1.1 How to Add the Page - Create New Task Flow ................. 6-4
       6.2.1.2 Setting Security for the Page Service ............................. 6-4
       6.2.1.3 How to Create Pages at Runtime .................................... 6-6
       6.2.1.4 Structure of Pages Created at Runtime ........................... 6-6
       6.2.1.5 How to Access Pages Created at Runtime ....................... 6-8
   6.2.2 How to Access Page - Create New Task Flow Parameters ................. 6-8
   6.2.3 Setting Scope in a Page - Create New Task Flow ............................ 6-10
   6.2.4 Setting an Outcome Parameter .................................................... 6-11
   6.2.5 Specifying Styles ....................................................................... 6-12
       6.2.5.1 Out-of-the-Box Styles .................................................. 6-12
       6.2.5.2 Custom Styles ............................................................. 6-13
   6.2.6 Specify an ADF Template ............................................................ 6-15
   6.2.7 Showing a Command Link ........................................................... 6-15
   6.2.8 Displaying an Image .................................................................. 6-16
   6.2.9 Customizing the Label ................................................................. 6-16
   6.2.10 How to Create Task Flow View Pages ...................................... 6-17
6.3 Managing Pages .................................................................................... 6-18
   6.3.1 Using the Page Service Data Control to Manage Pages ................ 6-18
   6.3.2 How to Add the Page Service Data Control ................................. 6-18
   6.3.3 How to View, Edit, and Delete Pages at Runtime ...................... 6-22
   6.3.4 Using the Page Service APIs to Manage Pages and Task Flows .... 6-22
6.4 Introduction to Custom Styles and Templates .......................................... 6-22
   6.4.1 How to Create Templates for Pages Created at Runtime .......... 6-23
       6.4.1.1 How to Create a Page or Page Fragment Style ................ 6-23
       6.4.1.2 How to Edit Create Page Dialog Styles ......................... 6-28
   6.4.2 Using ADF Templates ............................................................. 6-28
   6.4.3 Using Create Page Dialog Styles .............................................. 6-28
   6.4.4 ADF Page Dialog Styles ........................................................... 6-28
   6.4.5 ADF Page Dialog Styles ........................................................... 6-28
6.5 Introducing Custom Styles and Templates ............................................. 6-23
7 Enabling Runtime Editing of Pages Using Oracle Composer

7.1 Designing Editable Pages Using Oracle Composer Components ........................................ 7-1
7.1.1 How to Create a Customizable Page ............................................................................ 7-2
7.1.2 How to Enable Runtime Customization Using a Page Customizable ......................... 7-3
7.1.3 How to Enable Switching Between Page Modes Using a Change Mode Link or Change Mode Button 7-4
7.1.4 How to Define Editable Areas of a Page Using Panel Customizable Components... 7-6
7.1.5 How to Enable Layout Customization for a Page Using a Layout Customizable..... 7-7
7.1.6 How to Enable Component Customization Using Show Detail Frame Components....... 7-9
7.1.7 How to Create a Page Template for Creating Customizable Pages ......................... 7-10
7.1.8 How to Enable Customization in a Populated Page .................................................. 7-11
7.1.9 What Happens When You Add Oracle Composer Components .......................... 7-11
7.1.10 What Happens at Runtime .................................................................................... 7-12
7.1.11 What You May Need to Know When Designing Editable Pages ......................... 7-12
7.2 Designing Editable Pages Using Oracle Composer Components: Example ............... 7-13
7.3 Populating Pages with Content ..................................................................................... 7-16
7.3.1 What You May Need to Know When Adding Content to the Page ......................... 7-16
7.3.1.1 Considerations for Adding Task Flows .............................................................. 7-17
7.4 Troubleshooting Oracle Composer Problems................................................................... 7-17

8 Extending Runtime Editing Capabilities Using Oracle Composer

8.1 Overview of Extensibility Options .................................................................................. 8-1
8.1.1 Oracle Composer Add-Ons......................................................................................... 8-2
9 Performing Advanced Oracle Composer Configurations

9.1 Enabling Custom Actions on Show Detail Frame Components by Using Facets .......... 9-1
9.2 Enabling Custom Actions on Show Detail Frame Components By Using Facets: Example ... 9-2
  9.2.1 How to Create an ADF Task Flow ................................................................. 9-2
  9.2.2 How to Include an Additional Actions Facet ................................................. 9-4
  9.2.3 How to Create a Redirection Page ............................................................... 9-4
  9.2.4 How to Create Navigation Rules Between Pages ........................................ 9-5
  9.2.5 What Happens at Runtime .......................................................................... 9-6
9.3 Enabling Custom Actions on a Show Detail Frame Enclosing a Task Flow .......... 9-7
  9.3.1 How to Define Custom Actions at the Instance Level .................................. 9-7
  9.3.2 How to Define Custom Actions at the Global Level ..................................... 9-8
  9.3.3 How to Configure Custom Actions that Display Task Flow Views in a Separate Browser Window 9-10
  9.3.4 What Happens at Runtime .......................................................................... 9-12
9.4 Enabling Custom Actions On a Show Detail Frame Enclosing a Task Flow: Example. 9-12
9.5 Enabling Custom Actions on Task Flows that Display in the Component Navigator .. 9-14
  9.5.1 How to Configure Custom Actions in the Component Navigator .................. 9-15
    9.5.1.1 Defining the Logic for the Custom Action ............................................. 9-15
    9.5.1.2 Creating a JSPX Page Containing the Custom Action ......................... 9-16
    9.5.1.3 Calling the JSPX Page from the Application Page Containing the Task Flow. 9-18
  9.5.2 What Happens at Runtime .......................................................................... 9-19
9.6 Creating Event-Enabled Task Flows ................................................................. 9-19
9.7 Configuring an Application Page to Display in Source View by Default ............ 9-19
  9.7.1 How to Configure an Application Page to Display in Source View by Default .... 9-20
  9.7.2 What Happens at Runtime .......................................................................... 9-20
9.8 Disabling Source View for the Application ....................................................... 9-20
  9.8.1 How to Disable Source View ....................................................................... 9-20
  9.8.2 What Happens at Runtime .......................................................................... 9-21
9.9 Applying Styles to Components ....................................................................... 9-21
9.10 Configuring the Persistence Change Manager .................................................. 9-23
  9.10.1 Overview of the Default Change Manager Configuration .......................... 9-23
  9.10.2 How to Configure ComposerChangeManager in Existing 11.1.1.1 Applications... 9-25
    9.10.2.1 Updating CHANGE_PERSISTENCE Context Parameter in the web.xml File 9-25
    9.10.2.2 Adding Relevant Entries to the adf-config.xml File ............................. 9-26
9.11 Configuring Runtime Resource String Editing .................................................. 9-27
  9.11.1 Overview of the Resource String Editor ..................................................... 9-27
  9.11.2 How to Enable the Resource String Editor in Your Application ................ 9-28
  9.11.3 How to Configure the Override Bundle ..................................................... 9-29
10 Performing Oracle Composer-Specific MDS Configurations

10.1 Introduction to MDS............................................................................................................ 10-1
10.2 Using Oracle Composer Sandbox..................................................................................... 10-2
10.2.1 How to Enable Oracle Composer Sandbox Creation .................................................. 10-3
10.2.1.1 Updating Your Application's adf-config.xml File ................................................ 10-3
10.2.1.2 Updating Your Application's web.xml File ......................................................... 10-4
10.2.1.3 Selecting a Database Store for Your Application................................................... 10-5
10.2.2 What Happens at Runtime.............................................................................................. 10-5
10.2.3 How to Disable Sandbox for an Application............................................................... 10-7
10.2.4 How to Destroy Stale Sandboxes .................................................................................... 10-7
10.2.5 How to Enable Application Sandbox Creation ........................................................... 10-8
10.2.6 What Happens When You Enable Application Sandbox Creation ........................... 10-8
10.3 Adding Customization Layers to View and Edit Modes: Example ..................................... 10-8
10.3.1 How to Add Oracle Composer to a JSF Page ............................................................... 10-9
10.3.2 How to Create a Custom SiteCC Tip Layer ................................................................ 10-10
10.3.3 How to Create a Custom UserCC Tip Layer ................................................................ 10-11
10.3.4 How to Implement the ComposerSessionOptionsFactory Class ............................... 10-13
10.3.5 How to Register the Implementation with Oracle Composer ....................................... 10-14
10.3.6 How to Configure WebCenterComposerFilter .......................................................... 10-14
10.3.7 How to Redirect the Servlet to Enable Switch Between MDS Customization Layers...... 10-15
10.3.7.1 How to Create the AppNavigationUtils Class.................................................... 10-16
10.3.7.2 How to Create AppCloseHandler ........................................................................ 10-16
10.3.7.3 How to Register the AppCloseHandler ................................................................ 10-17
10.3.7.4 How to Create AppModeBean .............................................................................. 10-17
10.3.8 What Happens at Runtime............................................................................................ 10-18
10.4 Troubleshooting MDS-Related Problems with Oracle Composer................................. 10-18

11 Modifying Default Security Behavior of Oracle Composer Components

11.1 Applying Component-Level Restrictions by Defined Customization Policies .......... 11-1
11.1.1 How to Define Type-Level Customization Policies..................................................... 11-2
11.1.2 How to Define Instance-Level Customization Policies............................................ 11-3
11.2 Applying a Component Instance-Level Customization Restriction By Using Security Roles: Example 11-4
11.2.1 How to Configure ADF Security ................................................................................ 11-5
11.2.2 How to Define Roles and Grant Privileges in the jazn-data.xml File .................... 11-5
11.2.3 How to Customize the SessionOptions Object to Include Customization Policy... 11-8
11.2.4 How to Register the Implementation with Oracle Composer .................................... 11-10
11.2.5 How to Configure WebCenterComposerFilter ....................................................... 11-10
11.2.6 How to Apply an Instance-Level Customization Restriction .................................. 11-10
11.2.7 What Happens at Runtime......................................................................................... 11-11
11.3 Applying Tag-Level Security Using the customizationAllowed Attribute.................. 11-13
11.3.1 How to Enable Customization on an Image Component......................................... 11-14
11.3.2 How to Restrict Customization on an Image Component....................................... 11-14
13 Integrating Content

13.1 Introduction to Content Integration................................................................. 13-2
13.1.1 Overview of Content Adapters........................................................................ 13-2
13.1.2 Overview of Content Presenter Display Templates and the Templates Registry... 13-3
13.1.3 Overview of Content Data Controls .............................................................. 13-4
13.1.3.1 The getItems Method.................................................................................. 13-4
13.1.3.2 The search Method ..................................................................................... 13-5
13.1.3.3 The advancedSearch Method....................................................................... 13-5
13.1.3.4 The getURI Method.................................................................................... 13-6
13.1.3.5 The getAttributes Method.......................................................................... 13-6
13.1.3.6 The getEntity Method............................................................................... 13-7
13.1.3.7 The getNodes Method............................................................................... 13-8
13.2 Configuring Content Repository Connections.................................................. 13-6
13.2.1 How to Create a Content Repository Connection Based on the Oracle Content Server Adapter 13-7
13.2.2 How to Create a Content Repository Connection Based on the Oracle Portal Adapter... 13-11
13.2.3 How to Create a Content Repository Connection Based on the File System Adapter...... 13-14
13.2.4 How to Create a Content Repository Connection Based on the Oracle WebCenter Adapter for Microsoft SharePoint 13-16
13.2.4.1 Creating a Content Repository Connection Based on the Oracle WebCenter Adapter for Microsoft SharePoint 13-16
13.2.4.2 Mapping Microsoft SharePoint Content and Services.................................. 13-19
13.2.5 What Happens When You Create a Repository Connection ......................... 13-20
13.2.6 What You May Need to Know When Creating a Repository Connection........... 13-21
13.2.6.1 What You Should Know About Using Identity Propagation and External Application Authentication Methods 13-21
13.2.6.2 What You Should Know About Oracle Portal .............................................. 13-22
13.2.6.3 What You Should Know About Oracle WebCenter Adapter for Microsoft SharePoint 13-23
13.2.7 How to Edit a Common Repository Connection............................................. 13-25
13.2.8 How to Edit a WebCenter Application-Specific Content Repository Connection 13-26
13.2.9 How to Use an Existing Repository Connection for a New WebCenter Application...... 13-27
13.3 Adding Content to a JSF Page........................................................................... 13-28
13.4 Creating Custom Templates for Displaying Content Using Content Presenter........ 13-28
13.4.1 How To Define Custom Display Templates...................................................... 13-28
13.4.1.1 EL Expressions for Content Item Nodes....................................................... 13-29
13.4.1.2 Defining Single Content Item Display Templates........................................... 13-33
13.4.1.3 Defining Multiple Content Item Display Templates....................................... 13-38
13.4.1.4 Registering Templates in the Templates Registry.......................................... 13-41
13.4.1.5 Importing Custom Display Templates for WebCenter Spaces Applications. 13-46
13.4.1.6 Testing Custom Display Templates............................................................... 13-46
13.4.2 Performance Considerations for Custom Display Templates........................... 13-46
13.4.3 What Happens at Runtime............................................................................ 13-47
13.4.3.1 Identifying Display Templates for Selected Content Items.......................... 13-48
13.5 Configuring Content Data Controls for JCR Adapters....................................... 13-49
13.5.1 How to Configure a Content Repository Data Control ..................................... 13-49
13.5.2 What Happens When You Configure a Content Repository Data Control........... 13-50
18 Integrating the Instant Messaging and Presence Service

18.1 Introduction to the IMP Service ................................................................. 18-1
18.1.1 Understanding the IMP Service ............................................................. 18-1
18.1.2 Requirements for IMP ........................................................................... 18-2
18.2 Basic Configuration for the IMP Service .................................................... 18-2
18.2.1 Setting up Connections for the IMP Service .......................................... 18-3
18.2.1.1 IMP Service Connections ................................................................. 18-3
18.2.1.2 How to Set Up Microsoft LCS Connections for the IMP Service ....... 18-3
18.2.1.3 How to Set Up Microsoft OCS Connections for the IMP Service ...... 18-6
18.2.1.4 How to Set Up OWLCS Connections for the IMP Service ............... 18-11
18.2.2 Adding the IMP Service at Design Time .............................................. 18-15
18.2.2.1 IMP Service Task Flows .................................................................... 18-15
18.2.2.2 How to Add the IMP Service to your Application .............................. 18-15
18.2.2.3 Adding the Buddies Task Flow ........................................................... 18-16
18.2.3 Setting Security for the IMP Service ..................................................... 18-17
18.3 Advanced Information for the IMP Service .............................................. 18-20
18.3.1 Customizing IMP Views ........................................................................ 18-20
18.3.2 Troubleshooting the IMP Service .......................................................... 18-21
## 19 Integrating the Mail Service

19.1 Introduction to the Mail Service ................................................................. 19-1  
19.1.1 Understanding the Mail Service .............................................................. 19-1  
19.1.2 Requirements for the Mail Service .......................................................... 19-3  
19.2 Basic Configuration for the Mail Service .................................................... 19-3  
19.2.1 Setting up Connections for the Mail Service ............................................ 19-4  
19.2.2 Adding the Mail Service at Design Time .................................................. 19-10  
19.2.2.1 Mail Service Task Flows ...................................................................... 19-10  
19.2.2.2 How to Add the Mail Service to your Application ................................. 19-10  
19.2.3 Setting Security for the Mail Service ..................................................... 19-11  
19.3 Advanced Information for the Mail Service .............................................. 19-12  
19.3.1 Invoking the Mail Compose Page ............................................................ 19-12  
19.3.2 Configuring the Number of Mails Displayed ........................................... 19-13  
19.3.3 Troubleshooting the Mail Service ............................................................ 19-13

## 20 Integrating the People Connections Service

20.1 Introduction to the People Connections Service ......................................... 20-1  
20.1.1 Understanding the People Connections Service ...................................... 20-2  
20.1.1.1 Activity Stream .................................................................................... 20-2  
20.1.1.2 Connections ....................................................................................... 20-4  
20.1.1.3 Feedback ............................................................................................ 20-4  
20.1.1.4 Message Board .................................................................................. 20-4  
20.1.1.5 Profile ................................................................................................. 20-4  
20.1.2 Requirements for the People Connections Service ............................... 20-4  
20.2 Basic Configuration for the People Connections Service ......................... 20-5  
20.2.1 How to Set Up a Database Connection for the People Connections Service 20-5  
20.2.2 Adding the People Connections Service at Design Time .................... 20-7  
20.2.2.1 People Connections Service Task Flows ........................................... 20-7  
20.2.2.1.1 Activity Stream Task Flows ............................................................ 20-8  
20.2.2.1.2 Connections Task Flows ................................................................. 20-9  
20.2.2.1.3 Feedback Task Flows .................................................................... 20-11  
20.2.2.1.4 Message Board Task Flows ......................................................... 20-13  
20.2.2.1.5 Profile Task Flows ........................................................................ 20-14  
20.2.2.2 How to Add People Connections Service Task Flows to a Page ........ 20-16  
20.2.3 Setting Security for the People Connections Service ............................. 20-17  
20.3 Advanced Information for the People Connections Service .................... 20-18  
20.3.1 Establishing Site-Level Settings for People Connections Features ............ 20-18  
20.3.1.1 Understanding the People Connections Site-Level Setting Scripts .......... 20-18  
20.3.1.1.1 Preparing and Running settings-insert.sql .................................. 20-18  
20.3.1.1.2 Preparing and Running settings-update.sql ................................. 20-19  
20.3.1.2 Supported Site-Level Settings for People Connections Features .......... 20-20  
20.3.1.2.1 Activity Stream Site-Level Settings ............................................. 20-20  
20.3.1.2.2 Connections Site-Level Settings .................................................. 20-22  
20.3.1.2.3 Feedback Site-Level Settings ...................................................... 20-23  
20.3.1.2.4 Message Board Site-Level Settings ............................................. 20-23  
20.3.1.2.5 Profile Site-Level Settings ......................................................... 20-24  
20.3.1.3 Example: Configuring Connections to Accept Invitations Automatically 20-27
20.3.2 Customizing People Connections Task Flows Using Task Flow Binding Parameters.............. 20-28
20.3.2.1 How to Revise People Connections Task Flow Binding Parameters............................... 20-29
20.3.2.2 People Connections Task Flow Binding Parameters...................................................... 20-30
20.3.3 Using the People Connections Service REST APIs............................................................. 20-35
20.3.3.1 Activity Stream REST APIs......................................................................................... 20-36
20.3.3.1.1 Activity Stream Entry Point.................................................................................... 20-36
20.3.3.1.2 Activity Stream Resource Type Taxonomy.............................................................. 20-36
20.3.3.1.3 Activity Stream Security Considerations................................................................. 20-36
20.3.3.1.4 Activity Stream Resource Types.............................................................................. 20-37
20.3.3.2 Connections and Profile REST APIs............................................................................... 20-40
20.3.3.2.1 Connections and Profile Entry Point........................................................................ 20-41
20.3.3.2.2 Connections and Profile Resource Type Taxonomy................................................ 20-41
20.3.3.2.3 Connections and Profile Security Considerations.................................................. 20-41
20.3.3.2.4 Connections and Profile Resource Types................................................................. 20-42
20.3.3.3 Feedback REST APIs.................................................................................................... 20-51
20.3.3.3.1 Feedback Entry Point............................................................................................. 20-51
20.3.3.3.2 Feedback Resource Type Taxonomy......................................................................... 20-52
20.3.3.3.3 Feedback Security Considerations............................................................................. 20-52
20.3.3.3.4 Feedback Resource Types........................................................................................ 20-52
20.3.3.4 Message Board REST APIs.......................................................................................... 20-54
20.3.3.4.1 Message Board Entry Point.................................................................................... 20-54
20.3.3.4.2 Message Board Resource Type Taxonomy............................................................... 20-55
20.3.3.4.3 Message Board Security Considerations................................................................. 20-55
20.3.3.4.4 Message Board Resource Types.............................................................................. 20-55
20.4 Troubleshooting the People Connections Service................................................................. 20-57

21 Integrating Oracle WebCenter Wiki and Blog Server
21.1 Introduction to the Wiki and Blog Services........................................................................... 21-1
21.1.1 Understanding the Wiki and Blog Services....................................................................... 21-1
21.1.2 Requirements for the Wiki and Blog Services................................................................. 21-1
21.1.3 What Happens at Runtime............................................................................................... 21-2
21.2 Basic Configuration for the Wiki and Blog Services............................................................. 21-2
21.2.1 Adding Wikis or Blogs to Your Application or Portal......................................................... 21-3
21.2.1.1 Adding Wikis or Blogs by Using a Portlet...................................................................... 21-3
21.2.1.2 Adding Wikis or Blogs by Using an IFRAME............................................................... 21-4
21.2.1.3 Adding Wikis or Blogs by Using Web Services............................................................ 21-5
21.2.2 Oracle WebCenter Wiki and Blog Server URL Endpoints and Query String Parameters 21-5
21.2.3 Oracle WebCenter Wiki and Blog Server Web Services Interface.................................... 21-8
21.2.3.1 Definition of the Interface............................................................................................ 21-8
21.2.3.2 Oracle WebCenter Wiki and Blog Server Web Services Security............................. 21-13
21.2.3.3 Generating the Web Services Client Stub...................................................................... 21-14
21.2.3.4 Sample Code for Accessing the Web Services Interface............................................. 21-15
21.3 Advanced Information for the Wiki and Blog Services....................................................... 21-16

Part V Helping Users Find Content
# Integrating the Links Service

22.1 Introduction to the Links Service .......................................................... 22-1
22.1.1 Understanding the Links Service .................................................. 22-1
22.1.2 Requirements for the Links Service ......................................... 22-3
22.1.3 What Happens at Runtime ..................................................... 22-3
22.2 Basic Configuration for the Links Service ....................................... 22-4
22.2.1 Setting up Connections for the Links Service ......................... 22-4
22.2.2 Adding the Links Service at Design Time ............................ 22-6
22.2.2.1 Links Service Task Flows .................................................. 22-7
22.2.2.2 How to Add the Links Service to your Application .......... 22-7
22.2.3 Setting Security for the Links Service .................................... 22-8
22.2.4 Troubleshooting the Links Service ......................................... 22-8

# Integrating the Tags Service

23.1 Introduction to the Tags Service ...................................................... 23-1
23.1.1 Understanding the Tags Service ............................................. 23-1
23.1.2 Requirements for the Tags Service ....................................... 23-2
23.1.3 What Happens at Runtime .................................................... 23-3
23.2 Basic Configuration for the Tags Service ....................................... 23-4
23.2.1 Setting up Connections for the Tags Service ......................... 23-4
23.2.1.1 How to Set Up Connections for the Tags Service ............. 23-4
23.2.2 Adding the Tags Service at Design Time ............................ 23-5
23.2.2.1 Tags Service Components ............................................... 23-6
23.2.2.2 Tags Service Task Flows .................................................. 23-6
23.2.2.3 How to Add the Tags Service to your Application .......... 23-7
23.2.3 Setting Security for the Tags Service .................................... 23-9
23.3 Advanced Information for the Tags Service .................................. 23-9
23.3.1 Optional Way to Show Tags on Pages .................................. 23-9
23.3.2 Using the Resource Action Handling Framework to Tag Custom Objects ...... 23-13
23.3.3 Using Tagged Cloud and Tagged Items Together .................... 23-14

# Integrating the Search Service

24.1 Introduction to Search ..................................................................... 24-1
24.1.1 Understanding the Search Service ....................................... 24-2
24.1.2 What Happens at Runtime ................................................... 24-2
24.2 Basic Configuration for the Search Service .................................. 24-3
24.2.1 Setting up Connections for the Search Service ..................... 24-3
24.2.2 Adding the Search Service at Design Time ......................... 24-3
24.2.2.1 Search Service Task Flows ............................................... 24-4
24.2.2.2 How to Add the Search Service to Your Application ......... 24-4
24.2.3 Setting Security for the Search Service ................................. 24-5
24.3 Advanced Information for the Search Service ............................. 24-5
24.3.1 Adding the Search Task Flow ............................................. 24-5
24.3.2 Adding the Search - Saved Searches Task Flow ................. 24-7
24.3.3 Adding the Search Preferences Task Flow ......................... 24-7
24.3.4 Including Oracle SES Results in WebCenter Search Results .... 24-8
26. Integrating the Worklist Service

26.1 Introduction to the Worklist Service
26.1.1 Understanding Worklists
26.1.2 Requirements for the Worklist Service
26.1.3 What Happens at Runtime
26.2 Basic Configuration for the Worklist Service
26.2.1 Setting up Connections for the Worklist Service
26.2.2 Adding the Worklist Service at Design Time
26.2.2.1 Worklist Service Task Flows
26.2.2.2 How to Add the Worklist Service to your Application
26.3 Advanced Information for the Worklist Service

25. Integrating the Recent Activities Service

25.1 Introduction to the Recent Activities Service
25.1.1 Understanding the Recent Activities Service
25.1.2 What Happens at Runtime
25.2 Basic Configuration for the Recent Activities Service
25.2.1 Setting up Connections for the Recent Activities Service
25.2.2 Adding the Recent Activities Service at Design Time
25.2.3 Setting Security for the Recent Activities Service
25.3 Advanced Information for the Recent Activities Service
25.3.1 Refining the Behavior of the Recent Activities Service
25.3.2 Obtaining RSS News Feed URL for the Recent Activities Service

Part VI  Helping Users Keep Track
Part VII  Extending Your Application

27  Integrating with Oracle WebCenter Spaces

27.1  Introduction to WebCenter Spaces ................................................................. 27-1
27.2  Exposing WebCenter Spaces in Custom WebCenter Applications Using APIs ... 27-4
27.2.1 Introduction to WebCenter Spaces APIs .................................................... 27-4
27.2.2 Case Study 1: Purchasing Application Uses a Group Space to Evaluate Suppliers 27-5
27.2.3 Case Study 2: Customer Support Center Application Uses a Group Space to Discuss Customer Escalations 27-8
27.2.4 How to Set Up Your Custom WebCenter Application to Use the WebCenter Spaces APIs 27-9
27.2.4.1 Verifying That WebCenter Spaces Is Up and Running ......................... 27-9
27.2.4.2 Setting Up the Custom WebCenter Application to Use WebCenter Spaces APIs ... 27-10
27.2.4.3 Securing the Connection Between the Application and WebCenter Spaces .. 27-12
27.2.4.4 Setting Up the Group Space Client Context ........................................... 27-14
27.2.5 How to Provide Group Space Functionality in Custom WebCenter Applications ........ 27-14
27.2.5.1 Managing Group Spaces and Templates ................................................. 27-16
27.2.5.1.1 Creating a Group Space .................................................................. 27-16
27.2.5.1.2 Creating a Custom Attribute ......................................................... 27-17
27.2.5.1.3 Creating a Group Space Template .................................................. 27-17
27.2.5.1.4 Deleting a Group Space .................................................................. 27-18
27.2.5.2 Managing Group Space Membership .................................................... 27-18
27.2.5.2.1 Adding Members to a Group Space ................................................ 27-18
27.2.5.2.2 Inviting Users to Join a Group Space ............................................. 27-19
27.2.5.2.3 Removing Members from a Group Space ......................................... 27-20
27.2.5.2.4 Retrieving Role Information .......................................................... 27-20
27.2.5.3 Retrieving Group Space and Template Information .................................. 27-21
27.2.5.3.1 Retrieving a List of Group Spaces .................................................. 27-21
27.2.5.3.2 Retrieving a List of Public Group Spaces ........................................ 27-22
27.2.5.3.3 Retrieving Group Space Metadata .................................................. 27-22
27.2.5.3.4 Retrieving a List of Group Space Templates .................................... 27-24
27.2.5.3.5 Retrieving Group Space Template Metadata .................................... 27-25
27.2.5.3.6 Retrieving the WebCenter Spaces URL ......................................... 27-26
27.2.5.3.7 Retrieving a Group Space URL ...................................................... 27-26
27.2.5.3.8 Retrieving RSS Feed URLs for Group Space Services ..................... 27-26
27.2.6 How to Handle Exceptions Raised by WebCenter Spaces APIs .................... 27-28
27.2.6.1 Providing Localized Error Messages .................................................. 27-28
27.2.6.2 Listing the Error Stack ...................................................................... 27-29
27.2.7 Finding More Information on WebCenter Spaces APIs ............................. 27-29
27.2.8 Troubleshooting Issues with WebCenter Spaces APIs .............................. 27-29
27.3 Using the WebCenter Spaces REST APIs .................................................... 27-30
27.3.1 WebCenter Spaces Entry Point ............................................................... 27-31
27.3.2 WebCenter Spaces Resource Type Taxonomy ......................................... 27-31
27.3.3 WebCenter Spaces Security Considerations ........................................... 27-31
27.3.4 WebCenter Spaces Resource Types ....................................................... 27-32
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>27.3.4.1</td>
<td><code>urn:oracle:webcenter:spaces</code></td>
<td>27-32</td>
</tr>
<tr>
<td>27.3.4.2</td>
<td><code>urn:oracle:webcenter:space</code></td>
<td>27-33</td>
</tr>
<tr>
<td>27.3.4.3</td>
<td><code>urn:oracle:webcenter:space:members</code></td>
<td>27-34</td>
</tr>
<tr>
<td>27.3.4.4</td>
<td><code>urn:oracle:webcenter:space:member</code></td>
<td>27-35</td>
</tr>
<tr>
<td>27.3.4.5</td>
<td><code>urn:oracle:webcenter:space:attributes</code></td>
<td>27-36</td>
</tr>
<tr>
<td>27.3.4.6</td>
<td><code>urn:oracle:webcenter:space:attribute</code></td>
<td>27-36</td>
</tr>
<tr>
<td>27.3.4.7</td>
<td><code>urn:oracle:webcenter:space:resourceindex</code></td>
<td>27-37</td>
</tr>
<tr>
<td>27.3.4.8</td>
<td><code>urn:oracle:webcenter:space:lists</code></td>
<td>27-38</td>
</tr>
<tr>
<td>27.3.4.9</td>
<td><code>urn:oracle:webcenter:space:list</code></td>
<td>27-39</td>
</tr>
<tr>
<td>27.3.4.10</td>
<td><code>urn:oracle:webcenter:space:list:rows</code></td>
<td>27-41</td>
</tr>
<tr>
<td>27.3.4.11</td>
<td><code>urn:oracle:webcenter:space:list:row</code></td>
<td>27-42</td>
</tr>
<tr>
<td>27.3.4.12</td>
<td><code>urn:oracle:webcenter:space:list:columns</code></td>
<td>27-43</td>
</tr>
<tr>
<td>27.3.4.13</td>
<td><code>urn:oracle:webcenter:space:list:column</code></td>
<td>27-44</td>
</tr>
<tr>
<td>27.4</td>
<td>Exposing Enterprise Applications in WebCenter Spaces</td>
<td>27-46</td>
</tr>
<tr>
<td>27.4.1</td>
<td>Exposing Custom WebCenter Applications in WebCenter Spaces</td>
<td>27-46</td>
</tr>
<tr>
<td>27.4.2</td>
<td>Exposing Oracle Applications in WebCenter Spaces</td>
<td>27-46</td>
</tr>
<tr>
<td>27.4.3</td>
<td>Exposing Non-Oracle Applications in WebCenter Spaces</td>
<td>27-46</td>
</tr>
</tbody>
</table>

### 28 Using Oracle WebCenter REST APIs

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>28.1</td>
<td>Introduction to REST</td>
<td>28-1</td>
</tr>
<tr>
<td>28.2</td>
<td>Introduction to WebCenter REST APIs</td>
<td>28-2</td>
</tr>
<tr>
<td>28.3</td>
<td>Understanding the Link Model</td>
<td>28-2</td>
</tr>
<tr>
<td>28.3.2.1</td>
<td>Resource Type</td>
<td>28-4</td>
</tr>
<tr>
<td>28.3.2.2</td>
<td>Relationship</td>
<td>28-6</td>
</tr>
<tr>
<td>28.3.2.3</td>
<td>Capabilities</td>
<td>28-6</td>
</tr>
<tr>
<td>28.3.2.4</td>
<td>Media Type</td>
<td>28-7</td>
</tr>
<tr>
<td>28.3.2.5</td>
<td>Templates</td>
<td>28-7</td>
</tr>
<tr>
<td>28.4</td>
<td>Understanding Items Hypermedia</td>
<td>28-8</td>
</tr>
<tr>
<td>28.5</td>
<td>Navigating Hypermedia Using HTTP</td>
<td>28-8</td>
</tr>
<tr>
<td>28.6</td>
<td>Security Considerations for WebCenter REST APIs</td>
<td>28-10</td>
</tr>
<tr>
<td>28.7</td>
<td>Understanding Common Types</td>
<td>28-11</td>
</tr>
<tr>
<td>28.7.1</td>
<td>Common Types</td>
<td>28-11</td>
</tr>
<tr>
<td>28.7.1.1</td>
<td><code>personReference</code></td>
<td>28-11</td>
</tr>
<tr>
<td>28.7.1.2</td>
<td><code>groupSpaceReference</code></td>
<td>28-11</td>
</tr>
<tr>
<td>28.7.2</td>
<td>Portable Contact Types</td>
<td>28-12</td>
</tr>
<tr>
<td>28.7.2.1</td>
<td><code>name Portable Contact Type</code></td>
<td>28-12</td>
</tr>
<tr>
<td>28.7.2.2</td>
<td><code>address Portable Contact Type</code></td>
<td>28-12</td>
</tr>
<tr>
<td>28.7.2.3</td>
<td><code>organization Portable Contact Type</code></td>
<td>28-12</td>
</tr>
<tr>
<td>28.7.2.4</td>
<td><code>value Portable Contact Type</code></td>
<td>28-13</td>
</tr>
<tr>
<td>28.8</td>
<td>WebCenter REST API Examples</td>
<td>28-13</td>
</tr>
<tr>
<td>28.8.1</td>
<td>Navigating the Message Board Hypermedia</td>
<td>28-13</td>
</tr>
<tr>
<td>28.8.1.1</td>
<td>Accessing the Resource Index</td>
<td>28-14</td>
</tr>
<tr>
<td>28.8.1.2</td>
<td>Reading Messages</td>
<td>28-14</td>
</tr>
<tr>
<td>28.8.1.3</td>
<td>Creating a New Message</td>
<td>28-16</td>
</tr>
<tr>
<td>28.8.1.4</td>
<td>Updating a Message</td>
<td>28-17</td>
</tr>
</tbody>
</table>
30 Creating Portlets with the Oracle JSF Portlet Bridge

30.1 Introduction to the Oracle JSF Portlet Bridge

30.2 Creating a Portlet from a JSF Application

30.2.1 How to Create a JSF Portlet Based on a Page
31 Creating Portlets with the Portlet Wizard

31.1 Introduction to Java Portlets .............................................................................. 31-1
31.1.1 Introduction to Standards-Based Java Portlets .............................................. 31-1
31.1.2 Introduction to PDK-Java Portlets ............................................................... 31-3
31.2 Creating Java Portlets ....................................................................................... 31-5
31.2.1 How to Create a JSR 168 Java Portlet ........................................................... 31-5
31.2.2 What Happens When You Create a JSR 168 Java Portlet Using the JDeveloper Wizard .......................................................................................... 31-13
31.2.3 How to Create a PDK-Java Portlet ............................................................... 31-14
31.2.4 What Happens When You Create a PDK-Java Portlet ................................... 31-20
31.2.5 What You May Need to Know When Creating Java Portlets ....................... 31-21
31.2.5.1 Guidelines for Portlet Modes .................................................................... 31-21
31.2.5.1.1 Shared Screen Mode (View Mode for JSR 168) .................................... 31-22
31.2.5.1.2 Edit Mode (JSR 168 and PDK-Java) .................................................... 31-24
31.2.5.1.3 Edit Defaults Mode (JSR 168 and PDK-Java) .................................... 31-25
31.2.5.1.4 Preview Mode (JSR 168 and PDK-Java) ............................................ 31-26
31.2.5.1.5 Full Screen Mode (PDK-Java) ............................................................ 31-26
31.2.5.1.6 Help Mode (JSR 168 and PDK-Java) .................................................. 31-26
31.2.5.1.7 About Mode (JSR 168 and PDK-Java) ............................................... 31-27
31.2.5.2 Guidelines for Navigation within a Portlet .............................................. 31-27
31.2.5.2.1 Intraportlet Links .............................................................................. 31-28
31.2.5.2.2 Application Links ............................................................................. 31-28
31.2.5.2.3 External Links .................................................................................. 31-28
31.2.5.2.4 Internal/Resource Links .................................................................... 31-28
31.2.5.3 Guidelines for JavaScript ......................................................................... 31-29
31.2.5.4 Guidelines for PDK-Java Portlets ............................................................ 31-30

32 Coding Portlets

32.1 Enhancing JSR 168 Java Portlets ...................................................................... 32-1
32.1.1 How to Add Personalization ........................................................................ 32-2
33 Testing and Deploying Your Portlets

33.1 Introduction to Portlet Deployment Testing in a Development Environment .......... 33-1
33.2 Testing a Portlet Application on Integrated WebLogic Server ............................ 33-2
33.2.1 How to Test JSR 168 Portlets on Integrated WebLogic Server ..................... 33-3
33.2.2 What Happens When You Test JSR 168 Portlets on Integrated WebLogic Server .. 33-3
33.2.3 How to Test PDK-Java Portlet Applications on Integrated WebLogic Server .......... 33-4
33.2.4 What Happens When You Test PDK-Java Portlet Applications on Integrated WebLogic Server 33-3
33.3 Deploying a Portlet Application to an Oracle WebLogic Managed Server Instance .... 33-7
33.3.1 How to Create Deployment Profiles .............................................................. 33-8
33.3.1.1 Creating a WAR Deployment Profile .......................................................... 33-8
34.7.5  How To Cascade Events Across Multiple Components .............................. 34-44
34.7.6  What Happens at Runtime ........................................................................ 34-47

35  Creating Portlets with OmniPortlet
35.1  Introduction to OmniPortlet................................................................. 35-1
35.2  Adding OmniPortlet to Your Application ............................................ 35-2
35.3  Customizing OmniPortlet ..................................................................... 35-3
35.4  Troubleshooting OmniPortlet Problems .................................................. 35-4

36  Creating Content-Based Portlets with Web Clipping
36.1  Introduction to Web Clipping ............................................................... 36-1
36.2  Adding Web Clipping to Your Application ............................................. 36-3
36.3  Integrating Authenticated Web Content Using Single Sign-On ............... 36-4
36.4  Advanced Features of Web Clipping ..................................................... 36-6
36.4.1  Using the Web Clipping Transport API ........................................... 36-7
36.4.1.1  Implementing the Web Clipping Transport API ............................... 36-7
36.4.1.2  Deploying the Web Clipping Transport API .................................. 36-7
36.4.2  Rewriting Image Links to Use a Resource Proxy ............................... 36-8
36.5  Current Limitations of Web Clipping .................................................... 36-8
36.6  Troubleshooting Web Clipping ............................................................. 36-9

Part IX  Completing Your WebCenter Application

37  Securing Your WebCenter Application
37.1  Introduction to WebCenter Application Security ...................................... 37-1
37.2  Working with External Applications .................................................... 37-1
37.2.1  Using External Applications ............................................................ 37-2
37.2.1.1  Secured Service Connections .......................................................... 37-3
37.2.1.2  Automated Single Sign-On .............................................................. 37-4
37.2.2  Supplying User Credentials ............................................................... 37-5
37.2.3  Managing External Applications ....................................................... 37-6
37.2.3.1  Working with External Applications in Oracle JDeveloper ............... 37-6
37.2.3.1.1  Registering an External Application in Oracle JDeveloper ............ 37-6
37.2.3.1.2  Editing External Application Registration Details in Oracle JDeveloper 37-8
37.2.3.1.3  Deleting External Application Registration Details in Oracle JDeveloper 37-9
37.2.3.2  Working with External Applications in Enterprise Manager .......... 37-9
37.2.3.3  Working with External Applications Using WLST .......................... 37-9
37.3  Setting Up Security for Your Application ............................................... 37-9
37.4  Creating a Login Portlet ......................................................................... 37-10
37.5  Creating Login Pages and a Login Component ....................................... 37-10
37.6  Adding Portlets to a Login Page ............................................................ 37-10
37.7  Creating a Public Welcome Page for Your Application ......................... 37-11
37.8  Configuring Basic Authentication for Testing Portlet Personalization ....... 37-11
37.9  Registering Custom Certificates with the Keystore ............................... 37-12
37.10 Overriding Inherited Security on Portlets and Customizable Components .... 37-14
37.10.1 Portlets Security ................................................................................. 37-16
Testing and Deploying Your WebCenter Application

38.1 Introduction to Oracle WebLogic Servers ........................................ 38-1
38.2 Testing a Custom WebCenter Application on the Integrated WebLogic Server ........ 38-2
38.3 Packaging and Deploying a Custom WebCenter Application to a WebLogic Managed Server 38-4

38.3.1 Packaging a Custom WebCenter Application ........................................... 38-4
38.3.1.1 Packaging Database Connections and Application Security .................. 38-4
38.3.1.1.1 Packaging the Database Connections ........................................ 38-5
38.3.1.1.2 Packaging Application Security Data ........................................ 38-7
38.3.1.2 Creating Deployment Profiles ......................................................... 38-8
38.3.1.2.1 Creating the WAR Deployment Profile ....................................... 38-9
38.3.1.2.2 Creating the EAR Deployment Profile ........................................ 38-9
38.3.2 Preparing the Target Environment for Deployment .............................. 38-10
38.3.2.1 Creating and Provisioning an Oracle WebLogic Managed Server Instance... 38-10
38.3.2.2 Creating and Registering the Metadata Service Repository .................. 38-10
38.3.2.3 Configuring the Target Environment .................................................. 38-11
38.3.2.4 Creating a WebLogic Managed Server Connection ............................ 38-11
38.3.3 Deploying a Custom WebCenter Application to a Managed Server .......... 38-11
38.4 Transporting Customizations Between Environments ............................ 38-13
39 Working Productively in Teams

39.1 Enabling Source Control on WebCenter Applications .................................................. 39-1
39.1.1 Importing Application Files into CVS ................................................................. 39-2
39.1.2 Creating a Subversion Repository ........................................................................ 39-5
39.2 Understanding WebCenter Application Files Affected by Developers ..................... 39-7
39.2.1 Files Associated With Common Objects ............................................................ 39-7
39.2.2 Developer Actions Affecting Metadata Files ....................................................... 39-7
39.3 Implementing Common Requirements One Time ..................................................... 39-14
39.4 Portlet Producer Considerations ............................................................................... 39-14
39.4.1 Portlet Producer Connections .............................................................................. 39-14
39.4.2 Portlet Producer Name Clashes ............................................................................. 39-15
39.4.3 Combining Portlets from Different Portlet Producers ......................................... 39-15

Part X Appendixes

A Files for WebCenter Applications

A.1 About Files ................................................................................................................ A-1
A.2 Files Overview ............................................................................................................ A-2
A.3 Files Related to JSR 168 Portlets .............................................................................. A-2
A.3.1 portlet.xml ............................................................................................................. A-2
A.3.2 oracle-portlet.xml ................................................................................................. A-3
A.3.2.1 oracle-portlet.xml Syntax ................................................................................ A-4
A.3.2.2 oracle-portlet.xml Sample With Navigation Parameters ................................ A-5
A.3.3 oracle-portlet-tags.jar ......................................................................................... A-6
A.3.4 portlet_mode.jsp ................................................................................................... A-6
A.3.5 portlet_name.java .................................................................................................. A-7
A.3.6 portlet_nameBundle.jar ......................................................................................... A-7
A.3.7 web.xml ................................................................................................................ A-7
A.3.8 connections.xml .................................................................................................... A-7
A.4 Files Related to PDK-Java Portlets .......................................................................... A-11
A.4.1 producer_name.properties .................................................................................... A-12
A.4.2 _default.properties ............................................................................................. A-12
A.4.3 index.jsp ............................................................................................................... A-12
A.4.4 portlet_name_modePage.jsp .................................................................................. A-12
A.4.5 provider.xml .......................................................................................................... A-12
A.4.5.1 provider.xml Syntax ........................................................................................ A-13
A.4.5.2 provider.xml Sample ......................................................................................... A-13
A.4.6 web.xml ................................................................................................................ A-14
A.4.7 connections.xml .................................................................................................... A-14
A.5 Files Related to Pages .............................................................................................. A-14
A.5.1 adf-config.xml ..................................................................................................... A-14
A.5.2 DataBindings.cpx ................................................................................................. A-16
A.5.3 faces-config.xml ................................................................................................. A-16
A.5.4 page_name.jspx .................................................................................................... A-16
A.5.5 PageDef.xml .......................................................................................................... A-16
A.5.6 web.xml ................................................................................................................ A-18
### B Oracle Composer Component Properties and Files

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A.5.7 mds Subdirectory</td>
<td>A-18</td>
</tr>
<tr>
<td>A.5.8 wsdl Subdirectory</td>
<td>A-19</td>
</tr>
<tr>
<td>A.6 Files Related to Security</td>
<td>A-19</td>
</tr>
<tr>
<td>A.6.1 jazn-data.xml</td>
<td>A-19</td>
</tr>
<tr>
<td>A.6.2 cwallet.sso</td>
<td>A-21</td>
</tr>
<tr>
<td>A.6.3 jps-config.xml</td>
<td>A-21</td>
</tr>
<tr>
<td>A.6.4 adf-config.xml</td>
<td>A-21</td>
</tr>
<tr>
<td>A.7 Files Related to WebCenter Web 2.0 Services</td>
<td>A-22</td>
</tr>
</tbody>
</table>

### B.1 Oracle Composer Component Properties

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>B.1.1 Page Customizable</td>
<td>B-2</td>
</tr>
<tr>
<td>B.1.2 Change Mode Link and Change Mode Button</td>
<td>B-4</td>
</tr>
<tr>
<td>B.1.3 Layout Customizable Component</td>
<td>B-6</td>
</tr>
<tr>
<td>B.1.4 Panel Customizable Component</td>
<td>B-10</td>
</tr>
<tr>
<td>B.1.5 Show Detail Frame Component</td>
<td>B-14</td>
</tr>
<tr>
<td>B.1.6 Custom Action</td>
<td>B-21</td>
</tr>
<tr>
<td>B.1.7 ImageLink</td>
<td>B-24</td>
</tr>
</tbody>
</table>

### B.2 Oracle Composer-Specific Files and Configurations

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>B.2.1 pe_ext.xml</td>
<td>B-26</td>
</tr>
<tr>
<td>B.2.1.1 addon-config</td>
<td>B-27</td>
</tr>
<tr>
<td>B.2.1.2 property-panels</td>
<td>B-28</td>
</tr>
<tr>
<td>B.2.1.3 event-handlers</td>
<td>B-30</td>
</tr>
<tr>
<td>B.2.1.4 drop-handlers</td>
<td>B-30</td>
</tr>
<tr>
<td>B.2.1.5 filter-config</td>
<td>B-30</td>
</tr>
<tr>
<td>B.2.1.6 elbuilder-config</td>
<td>B-32</td>
</tr>
<tr>
<td>B.2.1.7 Sample pe_ext.xml File</td>
<td>B-32</td>
</tr>
<tr>
<td>B.2.2 adf-config.xml</td>
<td>B-33</td>
</tr>
<tr>
<td>B.2.2.1 page-editor-config</td>
<td>B-34</td>
</tr>
<tr>
<td>B.2.2.2 addon-panels</td>
<td>B-34</td>
</tr>
<tr>
<td>B.2.2.3 sandbox-namespaces</td>
<td>B-35</td>
</tr>
<tr>
<td>B.2.2.4 session-options-factory</td>
<td>B-35</td>
</tr>
<tr>
<td>B.2.2.5 rcv-config</td>
<td>B-35</td>
</tr>
<tr>
<td>B.2.2.6 customizableComponentsSecurity</td>
<td>B-36</td>
</tr>
<tr>
<td>B.2.2.7 mds-config</td>
<td>B-38</td>
</tr>
<tr>
<td>B.2.2.8 resource-string-editor</td>
<td>B-38</td>
</tr>
<tr>
<td>B.2.2.9 enable-source-view</td>
<td>B-38</td>
</tr>
<tr>
<td>B.2.2.10 persistent-change-manager</td>
<td>B-39</td>
</tr>
<tr>
<td>B.2.2.11 taglib-config</td>
<td>B-39</td>
</tr>
<tr>
<td>B.2.2.12 Sample adf-config.xml File</td>
<td>B-40</td>
</tr>
<tr>
<td>B.2.3 web.xml</td>
<td>B-42</td>
</tr>
<tr>
<td>B.2.3.1 CHANGE_PERSISTENCE Context Parameter</td>
<td>B-42</td>
</tr>
<tr>
<td>B.2.3.2 WebCenterComposerFilter</td>
<td>B-43</td>
</tr>
</tbody>
</table>

### B.3 Oracle Composer Default Add-Ons and Property Panels

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>B.3.1 Default Add-Ons</td>
<td>B-43</td>
</tr>
<tr>
<td>B.3.2 Default Property Panels</td>
<td>B-44</td>
</tr>
</tbody>
</table>

### B.4 Oracle Composer Help Topic IDs

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>B.4.1 Oracle Composer Help Topic IDs</td>
<td>B-45</td>
</tr>
</tbody>
</table>
B.5 Oracle Composer Components Style-Specific Properties.............................................. B-46
B.5.1 Style Selectors for Oracle Composer Components.................................................. B-46
B.5.1.1 Global Style Selectors.................................................................................. B-47
B.5.1.2 Page Customizable Style Selectors.............................................................. B-47
B.5.1.3 Layout Customizable Style Selectors............................................................ B-48
B.5.1.4 Panel Customizable Style Selectors.............................................................. B-48
B.5.1.5 Show Detail Frame Style Selectors............................................................... B-49
B.5.2 Style Attributes.................................................................................................... B-51
B.6 Customizable Components (HTML) Properties...................................................... B-54
B.6.1 Panel Customizable (HTML) Component.......................................................... B-55
B.6.2 Show Detail Frame (HTML) Component............................................................ B-57
B.6.3 Customizable Components (HTML) Style Selectors........................................ B-60
B.6.3.1 Panel Customizable (HTML) Style Selectors............................................... B-61
B.6.3.2 Show Detail Frame (HTML) Style Selectors................................................... B-61
B.6.3.3 Property Keys.............................................................................................. B-62
B.6.3.4 Icon Selectors.............................................................................................. B-63

C Resource Catalog Properties and Files
C.1 Configuration and Location of Catalog Definitions................................................... C-1
C.2 Default Catalog Definition..................................................................................... C-2
C.3 XML Schema....................................................................................................... C-3
C.4 Catalog Definition Attributes................................................................................ C-6
C.4.1 catalogDefinition............................................................................................ C-6
C.4.2 dynamicFolder............................................................................................... C-7
C.4.3 folder............................................................................................................... C-8
C.4.4 attributes........................................................................................................ C-9
C.4.5 attribute.......................................................................................................... C-9
C.4.6 resource.......................................................................................................... C-10
C.4.7 component...................................................................................................... C-11
C.4.8 schema........................................................................................................... C-12

D Guidelines for Creating Task Flows to Be Used in Oracle Composer-Enabled Pages
D.1 Guidelines for Effective Geometry Management and Pagination........................... D-1
D.1.1 Guideline 1: Create Task Flows that Flow......................................................... D-2
D.1.1.1 Example 1: Oracle ADF Faces Table Component with Its autoHeightRows 
              Attribute Set to a Specific Value 
              with a Scrolling Layout 
D.1.1.2 Example 2: Oracle ADF Faces Iterator Component Inside a Panel Group Layout 
D.1.2 Guideline 2: Limit the Number of Records Displayed in a Task Flow............... D-3
D.1.3 Guideline 3: Specify a Minimum Height for the Task Flow.............................. D-5
D.1.4 Guideline 4: Include UI for Accessing Data Beyond the Display Limit ............. D-5
D.2 Guidelines for Efficient Use of Task Flow Parameters and Customization and 
              Personalization Options................................................................................. D-7
D.2.1 Guidelines for Implementing Task Flow Parameters.......................................... D-10
D.2.2 Guidelines for Implementing Customization.................................................... D-10
D.2.3 Guidelines for Implementing Personalization................................................... D-11
G  Reuse of Oracle Portal Components

G.1  Introduction to Oracle Portal Components .................................................................  G-1
G.2  Reusing Portlets .............................................................................................................  G-2
G.2.1  How to Reuse JSR 168 and Oracle PDK-Java Portlets ...............................................  G-2
G.2.2  What You May Need to Know About Events .............................................................  G-3
G.2.3  What You May Need to Know About Mobile Portlets ...............................................  G-3
G.2.4  What You May Need to Know About the Portlet Chrome .......................................  G-3
G.2.5  What You May Need to Know About Personalizations and Customizations .......  G-3
G.2.6  What You May Need to Know About Oracle Portal System Resources ..................  G-3
G.2.7  What You May Need to Know About Partner and External Applications .............  G-4
G.2.8  How to Use the Federated Portal Adapter to Reuse Database Portlets ..................  G-4
G.2.9  What You May Need to Know About Troubleshooting the Federated Portal Adapter ...  G-6
G.2.9.1  On Registration .................................................................................................  G-6
G.2.9.2  During Runtime ..................................................................................................  G-7
G.2.10  What You May Need to Know About Limitations of the Federated Portal Adapter ...  G-7
G.2.11  How to Reuse Oracle PDK-Java Producers from Earlier Oracle Application Server Versions G-8
G.2.11.1  Consuming a Portlet from Oracle Portal ............................................................  G-8
G.2.11.2  Redeploying PDK-Java Producers from Oracle Portal ........................................  G-9
G.3  Reusing Items ..............................................................................................................  G-10

Glossary

Index
List of Examples

<table>
<thead>
<tr>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3–1</td>
<td>Default web.xml File Provided Through the WebCenter Application Template .......... 3-8</td>
</tr>
<tr>
<td>3–2</td>
<td>Default faces-config.xml File Provided Through the WebCenter Application Template...... 3-10</td>
</tr>
<tr>
<td>3–3</td>
<td>Default trinidad-config.xml File Provided Through the WebCenter Application Template.. 3-11</td>
</tr>
<tr>
<td>4–1</td>
<td>Simple Resource Viewer ................................................................. 4-11</td>
</tr>
<tr>
<td>4–2</td>
<td>Specifying a Task Flow ID in a resource-view Declaration ........................................ 4-12</td>
</tr>
<tr>
<td>4–3</td>
<td>Specifying a URL Rewriter Class Name (1) ................................................................. 4-12</td>
</tr>
<tr>
<td>4–4</td>
<td>Specifying a URL Rewriter Class Name (2) ................................................................. 4-12</td>
</tr>
<tr>
<td>4–5</td>
<td>Specifying a URL Rewriter Class Name (3) ................................................................. 4-13</td>
</tr>
<tr>
<td>4–6</td>
<td>No Resource Viewer (Resource ID as Page View ID) ......................................................... 4-13</td>
</tr>
<tr>
<td>4–7</td>
<td>Method for Returning Status Against a Unique Resource ID ............................................. 4-13</td>
</tr>
<tr>
<td>4–8</td>
<td>Possible ResourceStatus Values .................................................................................. 4-14</td>
</tr>
<tr>
<td>4–9</td>
<td>Resource URL Rewriter Method .................................................................................. 4-14</td>
</tr>
<tr>
<td>4–10</td>
<td>Sample Service Definition for Resource View .............................................................. 4-15</td>
</tr>
<tr>
<td>4–11</td>
<td>Code for Displaying Current Date and Time in User's Locale........................................ 4-17</td>
</tr>
<tr>
<td>4–12</td>
<td>Code for Displaying Specified Date and Time ............................................................... 4-17</td>
</tr>
<tr>
<td>4–13</td>
<td>Setting the Accessibility Mode in trinidad-config.xml ................................................... 4-17</td>
</tr>
<tr>
<td>5–1</td>
<td>Code to Create a Label Prefixed with SAVE_LABEL_ ..................................................... 5-31</td>
</tr>
<tr>
<td>5–2</td>
<td>Code to Create a Label Prefixed with SAVE_LABEL_ ..................................................... 5-44</td>
</tr>
<tr>
<td>6–1</td>
<td>Page Created Under a Default Scope ........................................................................... 6-7</td>
</tr>
<tr>
<td>6–2</td>
<td>Page Definition Created Under a Default Scope ......................................................... 6-7</td>
</tr>
<tr>
<td>6–3</td>
<td>Page Created Under a Custom Scope .......................................................................... 6-7</td>
</tr>
<tr>
<td>6–4</td>
<td>Page Definition Created Under a Custom Scope ........................................................... 6-7</td>
</tr>
<tr>
<td>6–5</td>
<td>Task Flow View Page Created Under a Default Scope ................................................... 6-7</td>
</tr>
<tr>
<td>6–6</td>
<td>Task Flow Definition Created Under a Default Scope .................................................... 6-7</td>
</tr>
<tr>
<td>6–7</td>
<td>Task Flow View Page Definition Created Under a Default Scope ................................... 6-7</td>
</tr>
<tr>
<td>6–8</td>
<td>Task Flow View Page Created Under a Custom Scope ................................................... 6-7</td>
</tr>
<tr>
<td>6–9</td>
<td>Task Flow Definition Created Under a Custom Scope ..................................................... 6-7</td>
</tr>
<tr>
<td>6–10</td>
<td>Task Flow Page Definition Created Under a Custom Scope ........................................... 6-7</td>
</tr>
<tr>
<td>6–11</td>
<td>Creating a Scope for Application Pages ................................................................. 6-10</td>
</tr>
<tr>
<td>6–12</td>
<td>Sample default_pageservice_styles.xml File ............................................................... 6-13</td>
</tr>
<tr>
<td>6–13</td>
<td>Sample Code of a New Page Style ............................................................................... 6-24</td>
</tr>
<tr>
<td>6–14</td>
<td>Sample Code of a Page Style that uses an ADF Template ............................................... 6-25</td>
</tr>
<tr>
<td>6–15</td>
<td>Sample Code of a Page Style on an ADF Template ......................................................... 6-26</td>
</tr>
<tr>
<td>6–16</td>
<td>templates.xml File ................................................................................................. 6-27</td>
</tr>
<tr>
<td>6–17</td>
<td>Sample Code for the Create Page Dialog Style .............................................................. 6-30</td>
</tr>
<tr>
<td>6–18</td>
<td>Rendering Pages as Tabs Using af:navigationPane ...................................................... 6-33</td>
</tr>
<tr>
<td>6–19</td>
<td>Rendering Pages as Tabs Using af:panelTabbed ............................................................ 6-33</td>
</tr>
<tr>
<td>6–20</td>
<td>Rendering Pages as Links Using af:commandLink ......................................................... 6-34</td>
</tr>
<tr>
<td>6–21</td>
<td>Rendering Pages as Links Using af:goLink ................................................................... 6-34</td>
</tr>
<tr>
<td>6–22</td>
<td>Rendering Thumbnail Views of Pages ........................................................................... 6-35</td>
</tr>
<tr>
<td>6–23</td>
<td>Using Page Service APIs to Create a Task Flow View Page ........................................... 6-37</td>
</tr>
<tr>
<td>6–24</td>
<td>Sample Showing How to Create a Page ....................................................................... 6-40</td>
</tr>
<tr>
<td>7–1</td>
<td>Page Customizable Component in the Page Source View ............................................... 7-4</td>
</tr>
<tr>
<td>7–2</td>
<td>Page Definition File After Adding Page Customizable Component ................................ 7-12</td>
</tr>
<tr>
<td>8–1</td>
<td>Sample code in the JSFF Fragment ............................................................................. 8-5</td>
</tr>
<tr>
<td>8–2</td>
<td>Oracle Composer Extension File .................................................................................. 8-7</td>
</tr>
<tr>
<td>8–3</td>
<td>New Add-On Referenced in adf-config.xml ................................................................... 8-8</td>
</tr>
<tr>
<td>8–4</td>
<td>The adf-config.xml File with the Reset Page Option Excluded ....................................... 8-9</td>
</tr>
<tr>
<td>8–5</td>
<td>rendered Attribute Setting in the adf-config.xml File ...................................................... 8-10</td>
</tr>
<tr>
<td>8–6</td>
<td>Code to Register a Property Panel for a Component ...................................................... 8-12</td>
</tr>
<tr>
<td>8–7</td>
<td>Code to Register a Property Panel for a Task Flow Instance .......................................... 8-13</td>
</tr>
<tr>
<td>8–8</td>
<td>rendered Attribute for a Property Panel ..................................................................... 8-14</td>
</tr>
</tbody>
</table>
# List of Figures

1–1 Overview of Oracle WebCenter Architecture ................................................................. 1-2
1–2 Oracle JDeveloper and the WebCenter Extension ............................................................ 1-3
2–1 Sample Fusion Order Demo for WebCenter Home Page .............................................. 2-3
2–2 Documents tab ............................................................................................................... 2-4
2–3 Links and Tags .................................................................................................................. 2-4
2–4 Links in the Fusion Order Demo for WebCenter .............................................................. 2-5
2–5 Tags in the Fusion Order Demo for WebCenter ............................................................... 2-5
2–6 Reviews Tab of the Fusion Order Demo for WebCenter .................................................. 2-6
2–7 Search Toolbar ............................................................................................................... 2-7
2–8 Search Results ................................................................................................................ 2-8
2–9 My Page Link .................................................................................................................. 2-8
2–10 Embedding a Video ...................................................................................................... 2-8
2–11 Recent Activities Component ....................................................................................... 2-9
2–12 RSS Viewer in the Fusion Order Demo for WebCenter ................................................ 2-9
2–13 People Connections -- Invite Screen .......................................................................... 2-10
2–14 People Connections -- Connections Screen ................................................................. 2-10
2–15 Announcements in the Fusion Order Demo for WebCenter .......................................... 2-11
2–16 Top Five Electronics (Standards-Based Java (JSR 168) Portlet) ................................. 2-12
2–17 Activity Stream Showing Recent Activities to the Documents and People Connections Services 2-12
2–18 Tag Cloud in the Fusion Order Demo for WebCenter .................................................. 2-13
2–19 Edit Link ...................................................................................................................... 2-13
2–20 About FOD Link in the Fusion Order Demo for WebCenter ........................................ 2-13
2–21 Oracle Composer Toolbar ........................................................................................... 2-14
2–22 Customized Resource Catalog in the Fusion Order Demo for WebCenter .................. 2-14
2–23 Adding Product Videos from YouTube ........................................................................ 2-15
3–1 Application from WebCenter Application Template in JDeveloper ............................ 3-7
3–2 Application from WebCenter Application Template in a Windows File System .......... 3-7
3–4 ADF Security Page of the ADF Security Wizard ........................................................... 3-17
3–5 Authentication Page of ADF Security Wizard ............................................................... 3-18
3–6 Automatic Policy Grants Page ....................................................................................... 3-18
3–7 Specify an Authenticated Welcome Page ...................................................................... 3-20
3–8 Summary Page ............................................................................................................. 3-20
4–2 Preferences Dialog ....................................................................................................... 4-4
4–3 HTTPS and Truststore Settings Pane ............................................................................ 4-4
5–1 Actions Menu on a Show Detail Frame ......................................................................... 5-4
5–2 Change Layout Icon ..................................................................................................... 5-5
5–3 Layout Options .............................................................................................................. 5-5
5–4 Collapse Icon on a Show Detail Frame ......................................................................... 5-6
5–5 Expand Icon on a Show Detail Frame .......................................................................... 5-6
5–6 Actions on a Web Clipping Portlet ................................................................................ 5-6
5–7 Edit Link While in View Mode ..................................................................................... 5-7
5–8 View Menu in Oracle Composer .................................................................................. 5-7
5–9 Design View of a Page .................................................................................................. 5-8
5–10 Oracle Composer Catalog ......................................................................................... 5-9
5–12 ADF Faces Components You Can Add to the Page .................................................... 5-11
5–13 Add Box Above Icon on a Box Component ............................................................... 5-13
5–14 Box Component Before Adding Another One Adjacent to It .................................... 5-13
5–15 Box Component Added Above .................................................................................. 5-14
5–16 Add Box Below Icon on a Box Component ............................................................... 5-14
5–17 Box Component Added Below .................................................................................. 5-14
5–18 Add Box Left Icon on a Box Component .................................................................. 5-15
5–19 Box Component Added to the Left ........................................................................... 5-15
5–20 Add Box Right Icon on a Box Component ............................................................... 5-15

xxxix
<table>
<thead>
<tr>
<th>Page</th>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>13-10</td>
<td>Edit Content Repository Connection</td>
</tr>
<tr>
<td>13-11</td>
<td>Application Resources - Properties</td>
</tr>
<tr>
<td>13-12</td>
<td>Edit Content Repository Connection</td>
</tr>
<tr>
<td>13-13</td>
<td>Dragging an Existing Common Repository Connection (IDE Connection) to an Application</td>
</tr>
<tr>
<td>13-14</td>
<td>Content Display Template Tags in Component Palette</td>
</tr>
<tr>
<td>13-15</td>
<td>Identifying Content Type Properties</td>
</tr>
<tr>
<td>13-16</td>
<td>Data Controls Panel - Oracle Content Server</td>
</tr>
<tr>
<td>13-17</td>
<td>Projects Panel - Model Project</td>
</tr>
<tr>
<td>13-18</td>
<td>Edit Data Controls</td>
</tr>
<tr>
<td>13-19</td>
<td>Edit Content Repository Data Control Dialog</td>
</tr>
<tr>
<td>13-20</td>
<td>The URI Attribute of the getURI Method</td>
</tr>
<tr>
<td>13-21</td>
<td>Oracle JDeveloper Context Menu for the getURI method</td>
</tr>
<tr>
<td>13-22</td>
<td>Edit Action Binding</td>
</tr>
<tr>
<td>13-23</td>
<td>Go Link Properties</td>
</tr>
<tr>
<td>13-24</td>
<td>ADF Go Link in a Browser</td>
</tr>
<tr>
<td>13-25</td>
<td>Page Data Binding Definition</td>
</tr>
<tr>
<td>13-26</td>
<td>Oracle JDeveloper Context Menu for the getURI method</td>
</tr>
<tr>
<td>13-27</td>
<td>New Textual Link</td>
</tr>
<tr>
<td>13-28</td>
<td>Insert Inside af:goLink - ADF Faces</td>
</tr>
<tr>
<td>13-29</td>
<td>Insert ADF Faces Item</td>
</tr>
<tr>
<td>13-30</td>
<td>ADF Object Image Link in a Browser</td>
</tr>
<tr>
<td>13-31</td>
<td>The Return node of the getItems method</td>
</tr>
<tr>
<td>13-32</td>
<td>The Create Context Menu for getItems Method</td>
</tr>
<tr>
<td>13-33</td>
<td>Edit Table Columns</td>
</tr>
<tr>
<td>13-34</td>
<td>Edit Action Binding</td>
</tr>
<tr>
<td>13-35</td>
<td>Read-Only Table for Publishing Folder Content</td>
</tr>
<tr>
<td>13-36</td>
<td>Files and Folders Displayed in a Read-Only Table</td>
</tr>
<tr>
<td>13-37</td>
<td>Default Formatting for the Name Column</td>
</tr>
<tr>
<td>13-38</td>
<td>Folder Content Displayed as Hyperlinks</td>
</tr>
<tr>
<td>13-39</td>
<td>Folder Contents</td>
</tr>
<tr>
<td>13-40</td>
<td>Table Properties - Common Tab</td>
</tr>
<tr>
<td>13-41</td>
<td>Files and Folders Displayed in a Single-Column Table</td>
</tr>
<tr>
<td>13-42</td>
<td>Display Files Only</td>
</tr>
<tr>
<td>13-43</td>
<td>Files Displayed in a Single-Column Table</td>
</tr>
<tr>
<td>13-44</td>
<td>Parameters of the getItems Method</td>
</tr>
<tr>
<td>13-45</td>
<td>Oracle JDeveloper Create Menu for getItems</td>
</tr>
<tr>
<td>13-46</td>
<td>The Edit Action Binding Dialog</td>
</tr>
<tr>
<td>13-47</td>
<td>Edit Tree Binding</td>
</tr>
<tr>
<td>13-48</td>
<td>Tree for Navigating Folder Content</td>
</tr>
<tr>
<td>13-49</td>
<td>Folder Content Displayed in a Tree</td>
</tr>
<tr>
<td>13-50</td>
<td>Output Text Converted to a Switcher Component</td>
</tr>
<tr>
<td>13-51</td>
<td>Switcher Component with Two Facets</td>
</tr>
<tr>
<td>13-52</td>
<td>Tree with File Names as Hyperlinks</td>
</tr>
<tr>
<td>13-53</td>
<td>ADF Parameter Form in the Design View</td>
</tr>
<tr>
<td>13-54</td>
<td>Table with Four Columns - search</td>
</tr>
<tr>
<td>13-55</td>
<td>Search Results for .jpg Files</td>
</tr>
<tr>
<td>13-56</td>
<td>Variables Dialog - predicates</td>
</tr>
<tr>
<td>13-57</td>
<td>Advanced Search - Design View</td>
</tr>
<tr>
<td>13-58</td>
<td>Advanced Search Results</td>
</tr>
<tr>
<td>14-1</td>
<td>Adding Content Repository Root Folder to JSF Page</td>
</tr>
<tr>
<td>14-2</td>
<td>Adding Subfolder to JSF Page</td>
</tr>
<tr>
<td>14-3</td>
<td>Adding Documents Service Task Flow to JSF Page</td>
</tr>
<tr>
<td>14-4</td>
<td>Specifying Parameters for Content Presenter Task Flow</td>
</tr>
<tr>
<td>14-5</td>
<td>Document List View Task Flow in the Bindings View</td>
</tr>
</tbody>
</table>
## List of Tables

<table>
<thead>
<tr>
<th>Table No.</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1–1</td>
<td>Oracle WebCenter Services</td>
<td>1-10</td>
</tr>
<tr>
<td>1–2</td>
<td>Accessibility Attributes for Oracle Composer Components</td>
<td>1-16</td>
</tr>
<tr>
<td>2–1</td>
<td>WebCenter Features in the Fusion Order Demo for WebCenter Sample Application.</td>
<td>2-15</td>
</tr>
<tr>
<td>5–1</td>
<td>Customization Restrictions on Oracle Composer Components</td>
<td>5-50</td>
</tr>
<tr>
<td>5–2</td>
<td>Mapping of Page Privileges to Oracle Composer Behavior</td>
<td>5-52</td>
</tr>
<tr>
<td>6–1</td>
<td>Page Service Developer Tools</td>
<td>6-1</td>
</tr>
<tr>
<td>8–2</td>
<td>Customization Manager Task Flow Parameters</td>
<td>8-30</td>
</tr>
<tr>
<td>8–3</td>
<td>Toolbar Areas</td>
<td>8-33</td>
</tr>
<tr>
<td>11–1</td>
<td>Actions Categories and Show Detail Frame Actions Mapping</td>
<td>11-18</td>
</tr>
<tr>
<td>11–2</td>
<td>Actions Settings for Adding Content to a Page</td>
<td>11-20</td>
</tr>
<tr>
<td>11–3</td>
<td>Actions Settings for Rearranging Content on a Page</td>
<td>11-20</td>
</tr>
<tr>
<td>11–4</td>
<td>Actions Settings for Moving Content on a Page</td>
<td>11-20</td>
</tr>
<tr>
<td>13–1</td>
<td>Parameters of the getItem Method</td>
<td>13-4</td>
</tr>
<tr>
<td>13–2</td>
<td>Return Attributes of the getItem Method</td>
<td>13-4</td>
</tr>
<tr>
<td>13–3</td>
<td>Parameters of the search Method</td>
<td>13-5</td>
</tr>
<tr>
<td>13–4</td>
<td>Return Attributes of the search Method</td>
<td>13-5</td>
</tr>
<tr>
<td>13–5</td>
<td>Parameters of the advancedSearch Method</td>
<td>13-5</td>
</tr>
<tr>
<td>13–6</td>
<td>Return Attributes of the advancedSearch Method</td>
<td>13-6</td>
</tr>
<tr>
<td>13–7</td>
<td>Return Attributes of the getAttributes Method</td>
<td>13-6</td>
</tr>
<tr>
<td>13–8</td>
<td>Configuration Parameters for Oracle Content Server</td>
<td>13-8</td>
</tr>
<tr>
<td>13–9</td>
<td>Oracle Content Server Connection Parameters for Each RIDC Socket Type</td>
<td>13-9</td>
</tr>
<tr>
<td>13–10</td>
<td>Parameters for Creating Oracle Portal-based Content Data Control</td>
<td>13-12</td>
</tr>
<tr>
<td>13–11</td>
<td>Adding Content to a JSF Page</td>
<td>13-28</td>
</tr>
<tr>
<td>13–12</td>
<td>EL Expressions for a Content Item Node</td>
<td>13-29</td>
</tr>
<tr>
<td>13–13</td>
<td>EL Expressions for Content Item Node Properties</td>
<td>13-30</td>
</tr>
<tr>
<td>13–14</td>
<td>EL Expressions for Content Item Node Property Values</td>
<td>13-32</td>
</tr>
<tr>
<td>13–15</td>
<td>EL Expressions for Content Item Node or Property Icons</td>
<td>13-33</td>
</tr>
<tr>
<td>13–16</td>
<td>EL Expressions for Content Item Node URLs</td>
<td>13-33</td>
</tr>
<tr>
<td>13–21</td>
<td>Loading of Node Properties By Node Retrieval Mechanism</td>
<td>13-47</td>
</tr>
<tr>
<td>14–1</td>
<td>Documents Service Task Flow Support by Content Types and Content Repository...</td>
<td>14-5</td>
</tr>
<tr>
<td>14–2</td>
<td>Documents Service Task Flows</td>
<td>14-11</td>
</tr>
<tr>
<td>14–3</td>
<td>Content Presenter Task Flow Parameters</td>
<td>14-13</td>
</tr>
<tr>
<td>14–4</td>
<td>Out-of-the-Box Templates for Displaying Single Content Items</td>
<td>14-14</td>
</tr>
<tr>
<td>14–5</td>
<td>Out-of-the-Box Templates for Displaying Multiple Content Items</td>
<td>14-15</td>
</tr>
<tr>
<td>14–6</td>
<td>Document Manager Task Flow Parameters</td>
<td>14-16</td>
</tr>
<tr>
<td>14–7</td>
<td>Document List View Task Flow Parameters</td>
<td>14-16</td>
</tr>
<tr>
<td>14–8</td>
<td>Recent Documents Task Flow Parameters</td>
<td>14-19</td>
</tr>
<tr>
<td>14–9</td>
<td>Apache MyFaces Trinidad Parameters</td>
<td>14-23</td>
</tr>
<tr>
<td>15–1</td>
<td>RSS Viewer Task Flow Binding Parameters</td>
<td>15-5</td>
</tr>
<tr>
<td>16–1</td>
<td>Announcement Task Flows</td>
<td>16-8</td>
</tr>
<tr>
<td>16–2</td>
<td>Announcements - Sidebar View Task Flow Parameters</td>
<td>16-11</td>
</tr>
<tr>
<td>17–1</td>
<td>Discussions Service Task Flows</td>
<td>17-7</td>
</tr>
<tr>
<td>17–2</td>
<td>Parameters for Discussion Forums Task Flow</td>
<td>17-9</td>
</tr>
<tr>
<td>17–3</td>
<td>Parameters for Discussions - Popular Topics Task Flow</td>
<td>17-13</td>
</tr>
<tr>
<td>17–4</td>
<td>Parameters for Discussions - Recent Topics Task Flow</td>
<td>17-14</td>
</tr>
<tr>
<td>17–5</td>
<td>Parameters for Discussions - Watched Forums Task Flow</td>
<td>17-15</td>
</tr>
<tr>
<td>17–6</td>
<td>Parameters for Discussions - Watched Topics Task Flow</td>
<td>17-16</td>
</tr>
<tr>
<td>17–7</td>
<td>Related Resource Types for forums</td>
<td>17-20</td>
</tr>
<tr>
<td>17–8</td>
<td>Writable Elements for forum</td>
<td>17-21</td>
</tr>
<tr>
<td>17–9</td>
<td>Read-only Elements for forum</td>
<td>17-21</td>
</tr>
<tr>
<td>17–10</td>
<td>Related Resource Types for forum</td>
<td>17-21</td>
</tr>
<tr>
<td>17–11</td>
<td>Related Resource Types for topics</td>
<td>17-22</td>
</tr>
<tr>
<td>17–12</td>
<td>Writable Elements for topic</td>
<td>17-23</td>
</tr>
</tbody>
</table>
Preface

This guide provides in-depth information for all of the following tasks:

- How to plan, build, deploy, and manage a custom WebCenter application with Oracle WebCenter Framework.
- How to administer, monitor, and maintain a custom WebCenter application and all of its associated components with Oracle WebCenter Framework.
- How to prepare an application for, configure, and integrate WebCenter services.
- How to plan, build, deploy, and manage portlets for a custom WebCenter application.

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

Audience

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

This manual is written for all of the following developers:

- The JSF application developer, who wants to build a custom WebCenter application or add Oracle WebCenter functionality to their application.
- The portal application developer, who wants to integrate Oracle WebCenter functionality into their application.
- The component developer, who wants to build portlets from Oracle WebCenter services.

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

- 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

Our goal is to make Oracle products, services, and supporting documentation accessible to all users, including users that are disabled. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at http://www.oracle.com/accessibility/.

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/support/contact.html or visit http://www.oracle.com/accessibility/support.html 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) documentation set or on Oracle Technology Network (OTN) at http://www.oracle.com/technology/index.html.

- Oracle Fusion Middleware Tutorial for Oracle WebCenter Developers
- Oracle Fusion Middleware Error Messages Reference
- Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework
- Oracle Fusion Middleware Developer’s Guide for Oracle Portal

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>
Part I

Introduction

Part I contains the following chapters:

- Chapter 1, "Understanding Oracle WebCenter"
- Chapter 2, "Introduction to the WebCenter Sample Application"
- Chapter 3, "Preparing Your Development Environment"
- Chapter 4, "Preparing Your Application for Oracle WebCenter Services"
This chapter introduces you to Oracle WebCenter and helps you understand what you must consider when building a custom application using the Oracle WebCenter Framework.

This chapter includes the following sections:

- Section 1.1, "Introduction to Oracle WebCenter"
- Section 1.2, "Design Questions to Consider Before You Start"
- Section 1.3, "Accessibility Features"

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

1.1 Introduction to Oracle WebCenter

With the advent of portals, users were suddenly able to interact with the information they were viewing and personalize their experience to match their exact requirements. These users are now demanding the same level of interaction with web applications. The line between portals and other web applications has become blurred, and with the proliferation of Web 2.0 technologies such as wiki, RSS, and blogs, this user demand for highly interactive applications can be met.

One way to simplify transactions is to provide everything users need to support a given task within the application itself. Since it is almost impossible to anticipate everything that users need to complete the tasks associated with their particular jobs, the best solution is often to enable users to evolve the application or mash it up with information from other enterprise sources. Oracle WebCenter provides you with a set of features and services (for example, portlets, customization, and content integration) that help you provide such options to your users, and help simplify the transactions users perform through your JSF applications.

Figure 1–1 provides an overview of the Oracle WebCenter architecture, showing the major components that comprise the product and the features and services offered.
The next few sections describe the WebCenter components depicted in Figure 1–1:

- **Introduction to Oracle Application Development Framework**
- **Introduction to Oracle JDeveloper and the WebCenter Extension**
- **Introduction to Oracle WebCenter Framework and Custom WebCenter Applications**
- **Introduction to Oracle WebCenter Services**

### 1.1.1 Introduction to Oracle Application Development Framework

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

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

Oracle ADF provides over 100 rich components, including hierarchical data tables, tree menus, in-page dialogs, accordions, dividers, and sortable tables. Oracle ADF also provides data visualization components, which are Flash and SVG-enabled and capable of rendering dynamic charts, graphs, gauges, and other graphics that can provide a real-time view of underlying data. Each component also supports skinning, along with internationalization and accessibility.

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

Oracle JDeveloper is an integrated development environment (IDE) for building service oriented applications using the latest industry standards for Java, XML, web services, portlets, and SQL. JDeveloper supports the complete software development life cycle, with integrated features for modeling, coding, debugging, testing, profiling, tuning, and deploying applications. JDeveloper’s visual and declarative approach and Oracle ADF work to simplify application development and to reduce mundane coding tasks. For example, code for many standard user interface widgets, such as buttons, lists of values, and navigation bars, are prepackaged for you. All you need to do is select the appropriate widget from the Component Palette and drop it into your application.

The WebCenter Extension packaged with JDeveloper provides all the development functionality you need for building custom WebCenter applications. Oracle WebCenter components are readily accessible from a catalog of resources.

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

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

- **WebCenter Application template.** Prepopulates the application with projects and libraries optimally scaled for the creation of data controls and consumption of portlets and Web 2.0 services.

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

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

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

1.1.3 Introduction to Oracle WebCenter Framework and Custom WebCenter Applications

Oracle WebCenter Framework augments the Oracle ADF environment by providing additional integration and runtime customization options. In essence, it integrates capabilities historically included in portal products, such as portlets, customization, personalization, and integration, directly into the fabric of the JSF environment. This eliminates artificial barriers for the user and provides the foundation for developing context-rich applications.

You can selectively add only desired Oracle WebCenter components or services to your custom WebCenter application. For example, you might only want to add the Instant Messaging and Presence (IMP) service. In this case, you could add just that service without adding all of the other services available with Oracle WebCenter.

An application built using Oracle WebCenter is referred to as a "custom WebCenter application," and can employ some or all of the following elements:

- The Oracle WebCenter Framework to provide declarative security, life cycle management tools, and the ability to consume portlets, WebCenter Services, and content in a JSF application.

- Oracle Composer to provide a runtime customization tool that enables your users to participate in and evolve the application in a managed way. Composer also enables users at runtime to add the WebCenter Services that they need to tailor their applications to meet their specific needs.

- WebCenter Services to provide the foundation of a social network and enable improved communication, content management capabilities, customization, and advanced searching, tagging, and linking support.

- Content repository data controls to integrate content from multiple content repositories into your application.

- Portlets to display, personalize, and reuse dynamic content.

- JSF skins to define a consistent look and feel for your application.

- Policy- and role-based security to control end user access and privileges.

Oracle WebCenter’s commitment to Service Oriented Architecture (SOA), and to the JCR 1.0 Java Content Repository and other community standards, means that front-end labor, for example needing to become familiar with things such as the APIs of the back-end systems of content stores, is no longer necessary. You get a wide range of plug-and-play products, tools, and services that make it easy to build the applications that your users need.
After you have built and tested your custom WebCenter application, you must still deploy it for your end users. When deployed and running, users begin to access the application and administrators to maintain it.

This section includes the following subsections:

- Oracle Composer
- Portlets
- Content Integration
- Search Framework
- Resource Catalog
- Customizable Components
- Application Security
- Application Life Cycle

1.1.3.1 Oracle Composer

Page customization, making updates that affect all users, is very important when branding an application for each deployment or each department. Allowing each user to modify the page to meet their specific needs is commonly referred to as personalization. Having an infrastructure that can handle all these customizations and personalizations requires a core set of components to manage and retrieve all the relevant metadata. More importantly, these customizations should be easy to change when the application has been deployed. JSF provides not only the running application code but also a description of the application. Hence, it is key to save customizations and personalizations as layers. This approach ensures that customizations are not lost because you can carry them over when you deploy a new version of the application. As an application developer, you require a set of JSF components that have this customization architecture built in to minimize the complexity of implementing such a solution.

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

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

The Oracle Composer tag library provides the components that you can add to make a page editable at runtime and define the behavior of content on the page (for example, move, sequence, or hide components).
In Oracle Composer, the Component Catalog dialog box, which is displayed by clicking the Add button on the page, displays the default runtime Resource Catalog Viewer. Users can browse the components in the viewer and then add them to the page. By default, the viewer displays the default Resource Catalog, which contains all of the Oracle ADF components and portlets available to the application. To control what components are visible to users in the viewer, you can modify the default Resource Catalog, or create one or more of your own Resource Catalogs. Resource Catalogs can contain items such as portlets, layout components, task flows, documents, and Oracle ADF Faces components.

For more information about implementing Composer components in your application, see Chapter 7, “Enabling Runtime Editing of Pages Using Oracle Composer.”

### 1.1.3.2 Portlets

Portlets enable users to access a wide range of functions and services. Oracle WebCenter Framework supports deployment and execution of both standards-based portlets (JSR 168, WSRP 1.0 and 2.0) and traditional Oracle PDK-Java based portlets. It also provides the runtime functions necessary to consume those portlets from within JSF applications. Any existing portlet can be leveraged by a custom WebCenter application with no changes. Developers simply register portlet producers within JDeveloper, and then drag and drop portlets directly onto their JSF pages.

You should be aware that all portlets run remotely from the application in the Oracle WebCenter Framework environment, meaning that there are no local portlets. You must always deploy the producer and register it with the application before consuming its portlets.

Several prebuilt portlets are available for use through a sample producer that you can register with your application. You can use the portlets that Oracle or third parties provide you, or you can create your own portlets programmatically. The prebuilt portlets that the Oracle WebCenter Framework provides include:

- **Web Clipping** is a browser-based, declarative tool that enables you to navigate to any web site and clip all or part of the displayed page. For more information, see Chapter 36, “Creating Content-Based Portlets with Web Clipping.”

- **OmniPortlet** is a declarative portlet-building tool through which you can select from any number of data sources, including XML files, character-separated value files (for example, spreadsheets), web services, database tables, RSS, and web pages. For more information, see Chapter 35, “Creating Portlets with OmniPortlet.”

Packaged applications also often come with their own set of portlets that enable you to access particular data or functions of the application. Assuming that they were built with compatible technology (WSRP, JSR 168, or PDK-Java), you can include these portlets in your custom WebCenter application as well.

Following the Oracle-submitted standard JSR-301, you can expose your task flows as standards based portlets. In this way, one application encompasses both the base application functionality and the portlets to be consumed for integration. When you revise your application, the portlets are naturally and automatically updated right along with it, rather than requiring a separate development project. To support JSR-301, the Oracle WebCenter Framework provides the Oracle JSF Portlet Bridge. For more information about the Oracle JSF Portlet Bridge, see Chapter 30, “Creating Portlets with the Oracle JSF Portlet Bridge.”

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

You can link WSRP 2.0 portlets such that parameters are passed between portlets and Faces components, and between portlets and the page. In this fashion, you can create a context-sensitive application, where the data displayed by the portlets changes depending upon the page context. In effect, it enables you to create an enterprise mashup for your users and, if you take advantage of Oracle Composer, your end users can create their own enterprise mashups, too.

For information about the different ways you might create portlets and how you might use them, see Chapter 29, "Overview of Portlets." For information about consuming portlets on pages and linking them, see Chapter 34, "Consuming Portlets."

1.1.3.3 Content Integration

JCR (Java Content Repository API, also known as JSR 170) adapters enable you to make data stored in content management systems, such as Oracle Universal Content Management, Oracle Portal, or even your file system, available to your application. Using JDeveloper, you can use the prebuilt JCR data control to grab the content and drop it onto your page. You can leverage the provided prebuilt user interface to display the content in your custom WebCenter application. This architecture enables you to build your user interface once and then at deployment time or during runtime, switch to whatever back end is required without having to recompile or rebuild the application. In addition, you do not need to learn the intricacies of each content management system’s custom APIs. For example, you could create a data control that selects content from any JCR 1.0 compliant repository or file system. When the data control is created, you can drop it onto a JSP document as a table.

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

For more information, see Chapter 13, "Integrating Content."

1.1.3.4 Search Framework

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

1.1.3.5 Resource Catalog

The Resource Catalog provides a federated view of the contents of one or more otherwise unrelated repositories within a unified search and browse user interface. Resources are created and published in their source repository, then exposed to the developer through the JDeveloper Resource Palette and to the end user through the Resource Catalog Viewer.

Resource Catalogs can contain the following components:

- **Layout components.** The primary layout component is a Box, which is a container that can hold all other types of components. At runtime, you must have
a Box into which you can drag and drop components. You can also add and arrange child components and delete components from a Box.

- **Oracle ADF Faces components.** You can add Text, Image, Page Link, Web Page, and Website Link components to your page. These are analogous to the JDeveloper design time components Rich Text Editor, Image, Command Link, Web Page, and Go Link, respectively.

  The Text component enables you to add rich text on the page. Adding this component invokes a text editor that can be resized and is similar to an HTML editor. You can add text and format it using the options available in the editor.

- **Portlets.** You can add WSRP, JSF, or PDK-Java portlets from any producer that was registered in JDeveloper. For more information about registering portlet producers and adding portlets to a page, see Chapter 34, "Consuming Portlets."

- **Task Flows.** If you have created task flows in JDeveloper, you can add these task flows from JAR files onto your page at runtime. For more information about ADF task flows, see the *Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework*.

- **Documents.** If you have configured the Documents service in your application, you can add documents from this service. For more information about the Documents service, see Chapter 14, "Integrating the Documents Service."

### 1.1.3.6 Customizable Components

To make any JSF JSP document (.jspx) editable at runtime, you must add Oracle Composer components to your page in JDeveloper. Depending on whether you want to enable personalization or customization of a page, there are two separate libraries to consider.

- **Oracle Composer Components.** Use these components if your page contains ADF Faces components. These components also make a page editable at runtime and define the behavior of content on a page at runtime (for example, move, sequence, or hide components).

- **Oracle Composer HTML Components.** Use these component if your page contains Trinidad components. These components are also useful for enabling personalization only on the page at runtime.

  For more information, see Chapter 7, "Enabling Runtime Editing of Pages Using Oracle Composer."

### 1.1.3.7 Application Security

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

Because Oracle WebCenter 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 custom WebCenter application. You need not synchronize roles within the application being built. It just references and uses the defined users and roles directly. Note also that you can use file-based security for the development phase of the application and then easily switch over to enterprise identity management at deployment time.
Oracle WebCenter also provides application roles that you can use to represent the policy of an application. By associating permissions with an application role defined within the policy store, you can keep them self contained within the application. On deployment, you can then associate users and enterprise roles with the application roles to grant those permissions to end users.

In some cases, it is desirable to leverage existing applications that have their own authentication mechanism, such as email. The email system is often on a different authentication system (with different user names and passwords) than the new application. The application must map the email user to the application user such that end users do not have to enter their user names and passwords each time they need information. Oracle WebCenter 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 custom WebCenter application, see Chapter 37, “Securing Your WebCenter Application.”

1.1.3.8 Application Life Cycle

After you have created and tested your custom WebCenter application in the design time environment (JDeveloper), you must deploy it to your production system. When deployed, you must then maintain the system, which includes monitoring performance and availability, editing or refreshing portlet producers, undeploying applications, and perhaps migrating customization data. Inevitably, at some point you will also want to further enhance the application, stage it again, and then redeploy it to your production system.

WebCenter is built on top of JDeveloper and Oracle ADF, which provide several benefits in the lifecycle management of your application:

- **Development framework:** JDeveloper and Oracle ADF provide the tools and framework you can use to build and update your application. Adding portlets, content, and customization capabilities to your custom WebCenter application is simply a matter of dragging and dropping the appropriate objects in either a source or WYSIWYG environment.

- **Enterprise deployment:** When you are ready to deploy your application to a production environment, JDeveloper packages everything that the JSF/Oracle ADF application needs in the EAR file for easy deployment though Enterprise Manager to the production system. For more information, see Chapter 38, "Testing and Deploying Your WebCenter Application."

**Note:** WebCenter Services typically require some back end, such as Oracle Discussions Server or Oracle Wiki Server, to be available in the deployment environment.

- **Standards-based administration:** Browser-based tools enable administrators to deploy, configure, and manage custom WebCenter applications and WebCenter 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. Custom WebCenter application metrics are delivered using the familiar Application Server Control monitoring and management interface. For more information, see Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter.
1.1.4 Introduction to Oracle WebCenter Services

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

Note: Oracle WebCenter Services refers to a specific packaging option when purchasing WebCenter. WebCenter Services refers to the services provided by Oracle WebCenter.

Table 1–1 describes each of the available services.

| Service                        | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     | Chapter                                                                 |
|--------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Announcements                  | Provides the ability to post announcements about important activities and events to all authenticated users.                                                                                                                                                                                                                                                                                                                                                                                   | Chapter 16, "Integrating the Announcements Service"                                                                       |
| Blog                           | Provides easy integration of a blog application within the context of your application.                                                                                                                                                                                                                                                                                                                                                                                                                                                                 | Chapter 21, "Integrating Oracle WebCenter Wiki and Blog Server"                                                            |
| Discussions                    | Provides the ability to create threaded discussions, posing and responding to questions and searching for answers. Also provides an effective group communication mechanism for important activities and events.                                                                                                                                                                                                                                                   | Chapter 17, "Integrating the Discussions Service"                                                                       |
| Documents                      | Provides content management and storage capabilities, including content upload, file and folder creation and management, file check out, versioning, and so on.                                                                                                                                                                                                                                                                                               | Chapter 14, "Integrating the Documents Service"                                                                       |
| Events                         | Provides the ability to create and maintain a schedule of events relevant to a wider group of users. Events are published to all authenticated users.                                                                                                                                                                                                                                                                                              | "Working with the Events Service" in the Oracle Fusion Middleware User’s Guide for Oracle WebCenter |
| Instant Messaging and Presence (IMP) | Provides the ability to observe the status of other authenticated users (whether online, offline, busy, or idle) and to contact them instantly.                                                                                                                                                                                                                                                                                             | Chapter 18, "Integrating the Instant Messaging and Presence Service"                                                        |
| Links                          | Provides the ability to view, access, and associate related information; for example you can link to a solution document from a discussion thread.                                                                                                                                                                                                                                                                                      | Chapter 22, "Integrating the Links Service"                                                                             |
| Lists                          | Provides the ability to create, publish, and manage lists. Users can create lists from prebuilt structures or create their own custom lists.                                                                                                                                                                                                                                                                                         | "Working with the Lists Service" in the Oracle Fusion Middleware User’s Guide for Oracle WebCenter |
|                               | **Note:** This service is available only in Oracle WebCenter Spaces.                                                                                                                                                                                                                                                                                                                                                               |                                                                                                                          |
|                               | **Note:** This service is available only in Oracle WebCenter Spaces.                                                                                                                                                                                                                                                                                                                                                               |                                                                                                                          |
WebCenter provides task flows for these services through the WebCenter Services Catalog. For information about how to prepare your application to consume these task flows, see Chapter 4, "Preparing Your Application for Oracle WebCenter Services."

### How WebCenter Services Integrate with Each Other

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

- The Search service is designed to search any WebCenter service in your application. For example, if you have the Documents service in your application, then the search operation looks for terms in document names and in the documents themselves. The results include tag hits, so any matching tags on your documents and pages are also returned, with relevance based on the quality and frequency of tags.
The Links service lets you view, access, and associate related information across services. For example, in a discussion forum, you can link to a related discussion forum, document, URL, or event. The Links service also supports bidirectional links between objects, such as a link from a discussion topic to a document and back to the discussion topic.

The Presence icon can appear in any WebCenter service where a user is indicated. For example, click the author's name on a document to invoke a context menu and send an instant message or mail to that person.

The Recent Activities service lets you view the most recent additions or changes to the announcements, documents, discussions, and pages in your application.

1.2 Design Questions to Consider Before You Start

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

This section includes the following subsections:

- User Considerations
- Site Administrator Considerations
- Developer Considerations

1.2.1 User Considerations

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

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

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

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

  When you design your application, you must choose where to insert the Oracle Composer components to make those areas of the page editable. Components that allow users to add to the page must appear in a Resource Catalog at runtime. You can also choose to have different catalogs available depending upon the context of the application or the user's privileges.

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

For more information about the Oracle Composer and exposing resource catalogs, see Chapter 7, "Enabling Runtime Editing of Pages Using Oracle Composer."

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

If you choose to implement personalization, then you must also implement some form of security for the application consuming the portlet. The Personalize option appears only to authenticated users.

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

For more information, see Chapter 29, "Overview of Portlets."

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

For more information, see Chapter 4, "Preparing Your Application for Oracle WebCenter Services."

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

1.2.2 Site Administrator Considerations

The following list suggests questions you should ask about the administrators of your application:
Do administrators need to customize pages and portlets for users or user groups? You may want to enable your custom WebCenter application administrator to customize pages and portlets to provide default views of the custom WebCenter application for users.

- **Custom WebCenter application look and feel.** By implementing skins for your custom WebCenter application, you enable administrators to choose the look of the pages in the application.

  For information about how to implement skins, see Chapter 19 "Customizing the Appearance Using Styles and Skins" in Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework.

- **Customizable Pages.** The administrator can customize the layout of the page.

  For information about customizable components, see Chapter 7, "Enabling Runtime Editing of Pages Using Oracle Composer."

- **Portlets.** Portlets can optionally include a customization mode (Edit Defaults mode) that enables the administrator to customize the portlet. The administrator’s customization is treated as the default view of the portlet for all other users.

  For more information about portlets, see Chapter 29, "Overview of Portlets."

### 1.2.3 Developer Considerations

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

- **Should developers build task flows or portlets as reusable components?** Thanks to the Oracle JSF Portlet Bridge, developers can make their task flows into JSR 168 portlets. Hence, they have the option of creating reusable components either as task flows, which can potentially be made into JSR 168 portlets through the bridge, or directly as JSR 168 portlets. If the primary purpose of the component is inclusion in a portal, then developers may prefer to build it directly as a JSR 168 portlet, without going through the bridge. If the primary purpose of the component is local execution in a JSF application, then the developer may prefer to create it as a task flow first and then, optionally, use the bridge to run it as a JSR 168 portlet. Building task flows means that the components can be used in other applications without having to repackage them as portlets first.

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

**Note:** It is critical that the custom WebCenter application administrator and the developers communicate when the application is under construction. At design time, developers must make many choices that determine what the administrator can do to the application at runtime. For example, if the developers choose not to implement skins, then the administrator has no control over the look and feel of the application. Hence, the administrator and the developers should ensure that they consult over these decisions at design time.
For information about including content through JCR data controls and adapters, see Chapter 13, "Integrating Content."

- **Will the visible/editable content of the custom WebCenter application vary depending upon the identity of the user?** If you plan to have content in your custom WebCenter application that is not for everyone, then you must set up a security model with login and user roles that enable you to control access. Components, portlets, and pages can display or not display, depending upon a user's identity. If the content is open to everyone (for example, a Human Resources custom WebCenter application with content for all employees), then you might not need to implement access control.

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

  For more information about implementing a security model for your custom WebCenter application, see Chapter 37, "Securing Your WebCenter Application."

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

  For more information about accessing external applications from your custom WebCenter application, see Chapter 37, "Securing Your WebCenter Application."

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

  For more information about external application portlets, see Section 37.2.3, "Managing External Applications."

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

  For more information about team development, see Chapter 39, "Working Productively in Teams."

### 1.3 Accessibility Features

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

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

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

### 1.3.1 Generating Accessible HTML

WebCenter 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 1–2.

**Table 1–2 Accessibility Attributes for Oracle Composer Components**

<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.</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>pe:pageCustomizable</td>
<td>No accessibility attributes.</td>
</tr>
<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>

### 1.3.2 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-specific components, see Table 1–2. For a list of accessibility-related attributes for other components, see the Developing Accessible ADF Faces Components and Pages section in the Oracle Fusion Middleware Web User Interface Developer’s Guide for Oracle Application Development Framework.
1.3.3 Accessibility Considerations for Portlets

HTML standards do not allow the use of nested form tags in page content. However, JavaServer Faces pages almost always include a form tag so the JSF reference implementation, which is what is used to generate the markup, puts the entire page body inside a form tag. If you then have a portlet on that Faces page that includes a form tag in its content, which is frequently the case, you have a nested form tag. The common resolution for nested form tags is to put the inner form inside an iframe.

However, iframes are not very well accommodated by today’s screen readers and so are not permitted by some accessibility standards.

To address the situation of nested form tags and iframes, WebCenter offers two solutions:

- WebCenter automatically parses the HTML markup from portlets looking for form tags and rewrites them and the corresponding submit tags to be Faces compliant using JSF's subform mechanisms. However, this HTML parsing can fail, especially given the growing use of complex, AJAX style markup.

- WebCenter provides an optional attribute in the adf:portlet tag called renderPortletInIFrame that has three settings:
  - auto (default) - parse the markup and, if the parsing gets an error because it cannot understand it properly, render the content inside an iframe, otherwise render the content inline.
  - true - always render an iframe.
  - false - never render an iframe.

1.3.4 Accessibility Features in WebCenter Spaces

WebCenter Spaces pages has been designed to meet accessibility guidelines.
Introduction to the WebCenter Sample Application

As a companion to this guide, a sample enterprise application was developed using Oracle Fusion Middleware 11g Release 1 (11.1.1), including Oracle WebCenter Framework. This sample application, called the Fusion Order Demo for WebCenter, is used as an example throughout this guide to illustrate points and provide samples.

In this chapter, you will learn about the Fusion Order Demo for WebCenter, as well as take a quick tour of the components built using Oracle WebCenter Framework.

This chapter includes the following sections:

- Section 2.1, "About the Fusion Order Demo for WebCenter"
- Section 2.2, "Browsing the Oracle WebCenter Framework Components in the Fusion Order Demo for WebCenter at Runtime"
- Section 2.3, "Oracle WebCenter Framework Components in the Fusion Order Demo for WebCenter"

2.1 About the Fusion Order Demo for WebCenter

The Fusion Order Demo for WebCenter showcases Oracle Fusion Middleware, including Oracle WebCenter Framework, in an enterprise, integrated application. In this sample application, electronic devices are sold through a storefront-type web application. Customers can browse various products, place orders, and check the status of their orders. In this application, you can see Oracle WebCenter Framework features being used to provide Web 2.0, customization, and personalization capabilities.

In order to view and run the demo, you need to install Oracle JDeveloper 11g (11.1.1) and the Oracle WebCenter extension (11.1.1), as described in Section 3.1, "Installing the WebCenter Extension Bundle." You then need to download the application for this demonstration. Once the application is installed and running, you can view the application at design time in Oracle JDeveloper and at runtime by logging in as an existing customer.

For information on downloading and installing the sample application and the associated schema, refer to Chapter 2, "Introduction to the ADF Sample Application" in the Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework.

2.2 Browsing the Oracle WebCenter Framework Components in the Fusion Order Demo for WebCenter at Runtime

Once you have downloaded and installed the demo, you can run the application and navigate to the components built using Oracle WebCenter. This section will take you on a visual tour of the application and show you the components built using Oracle WebCenter Framework.

This section focuses on how the various Oracle WebCenter components display at runtime. You can examine the application at design time in Oracle JDeveloper. For more information on how you can add a service or feature to your application, refer to the appropriate chapter in this guide. Table 2–1 provides a quick reference to the relationship between the features you will see in this section and the chapters in the guide that show you how to implement them.

2.2.1 Exploring the Home.jspx Page

Let's take a look at the Home page, which showcases the following Oracle WebCenter Framework features:

- Documents service
- Links service
- Tags service
- Discussions service

To view these features in the Fusion Order Demo for WebCenter application:

1. Open the application in Oracle JDeveloper and run the home.jspx page. You'll notice that the center of the page looks like Figure 2–1.
Below one of the product names, click See Full Details to view a few of the features built with Oracle WebCenter Framework.

For example, if you click See Full Details beneath Bluetooth Phone Headset, your screen will look something like this:
Notice that the Documents tab is selected, with a list of contents beneath it. This component was built using the Documents service (Chapter 14, "Integrating the Documents Service"). The Documents service enables you to display content from a content server or file system directly within the application. In this example, a folder named Bluetooth Phone Headset contains a number of product manuals in different formats (PDF, text, GIF, and so on) on an Oracle Content Server, which you can browse directly from the Documents tab.

3. Notice two components above the tabs, called Links and Tags.

These two components display when the Links and Tags services are added to an application.

You can click each of these links to see what displays, but to learn more about each of these features, refer to Chapter 22, "Integrating the Links Service" and Chapter 23, "Integrating the Tags Service."
Figure 2–4  Links in the Fusion Order Demo for WebCenter

The links that display here are sample links we’ve created from the Bluetooth Phone Headset folder to two documents within the folder and to external URLs. You can quickly access other information about the product or even create new links by choosing Link to New to link to a new document or URL or Link to Existing to link to a document shown in the Bluetooth Phone Headset folder. This service enables you to create relationships between content within the WebCenter application and from within the WebCenter application to external sources.

Figure 2–5  Tags in the Fusion Order Demo for WebCenter

The Tag this item dialog shows the item you wish to tag (Bluetooth Phone Headset) and provides a field where you can view or modify existing tags (in this case, bluetooth) and add new tags. You also see a list of possible tags you may want to use, as well as a check box that enables you to let others see and use these tags within the application.

4. Now, let's browse the other tabs. Click the Reviews tab.
Here, you’ll see product reviews by users, which is an example of the Discussions service. The Discussions service enables you to post reviews about a particular topic to a discussion forum, to which other users can respond. In this example, you are viewing a Discussion forum for the Bluetooth Phone Headset, where a user has posted a review of it. You can click the link that displays under Topic to read or reply to a review, or post your own review of the product. To learn more about this feature, see Chapter 17, "Integrating the Discussions Service."

5. At the bottom of the page, click **OK** to exit this portion of the demo.

Now, let's take a look at some of the other portions of this application that illustrate Oracle WebCenter Framework.

### 2.2.2 Exploring MyPage.jspx

Let’s take a look at the personal page, MyPage, which showcases the following Oracle WebCenter Framework features:

- Search service
- Embedding a video by using Oracle Composer
- Recent Activities service
- RSS Viewer service
- People Connections service
- Announcements service
- Standards-Based Java (JSR 168) portlet
- Activity Stream task flow
Tags service (tag cloud)

To view these features in the Fusion Order Demo for WebCenter:

1. Before we look at the personal page, take a quick look at the top of the application. You’ll notice a search toolbar, which was built using the Search service (Chapter 24, "Integrating the Search Service"). Figure 2–7 shows the toolbar with the keyword “playstation” entered.

**Figure 2–7  Search Toolbar**

![Search Toolbar](image)

2. If you enter a keyword, for example phone, a list of results displays. The Search service returns all instances of the keyword across the entire application. **Figure 2–8** shows an example of possible search results.
3. To the left of the Search toolbar, click the My Page link.

4. In the top left corner of My Page, you can see a YouTube video embedded within a customizable component (Figure 2–10). Oracle Composer includes a toolbar that enables you to add a video to your application.

Section 2.2.3, “Editing MyPage.jspx” explains a bit more about Oracle Composer and the Fusion Order Demo. For more information about embedding a video, see Section 12.5, "Adding XML Content to the Resource Catalog: Examples."
5. Below the video component, notice the component called **Recent Activities** (Figure 2–11), which was built using the Recent Activities service (Chapter 25, "Integrating the Recent Activities Service").

**Figure 2–11 Recent Activities Component**

You'll notice in the upper right corner of this component that activities from the last 30 days display. The services listed: Documents, Discussions, and Announcements, are the services that have been updated by any of the application's users over the past 30 days. The number next to each service name indicates the number of updates during the same period of time.

6. Below the Recent Activities component, notice the **FOD News** section (Figure 2–12), which receives an RSS feed local to the application. This component was built using the RSS Viewer service (Chapter 15, "Integrating the RSS Service").

**Figure 2–12 RSS Viewer in the Fusion Order Demo for WebCenter**

7. To the right of the YouTube video component, notice the **My Connections** component, which was built using the People Connections service. People Connections service enables you to provide social networking capabilities in an
application. At runtime, application users can create their own network by inviting other users to join them.

You can use the Search User field to locate a specific user, or click the arrow to see a list of users available to add to your network (Figure 2–13).

Figure 2–13 People Connections -- Invite Screen

![Invite Screen](image)

Once you invite a user and that user accepts your invitation, you see the user in your network. Here, for example, if you’re logged in as user ngreenbe, you see the two users: ahunold and sking in your network (Figure 2–14). In turn, if you log in as ahunold or sking, you also see user ngreenbe in your network.

Figure 2–14 People Connections -- Connections Screen

![Connections Screen](image)

8. Below the My Connections component, notice the Announcements component (Figure 2–15), built using the Announcements service. This service connects to the same back-end Discussions server as the Reviews you saw in Section 2.2.1, "Exploring the Home.jspx Page."
Here, the Announcements service displays sale items on products contained in the Discussions server. The application administrator or developer can modify the Announcement service to display different information that you may want to call out within the application, such as discounts or product recall information.

9. To the right of the My Connections component is a **Standards-Based Java (JSR 168) portlet** (Figure 2–16) that is based on an Oracle ADF task flow. This ADF task flow was turned into a portlet by using the Oracle JSF Portlet Bridge *(Chapter 30, “Creating Portlets with the Oracle JSF Portlet Bridge”)*.
Browsing the Oracle WebCenter Framework Components in the Fusion Order Demo for WebCenter at Runtime

Figure 2–16  Top Five Electronics (Standards-Based Java (JSR 168) Portlet)

<table>
<thead>
<tr>
<th>Rank</th>
<th>Product</th>
<th>Description</th>
<th>Revenue</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>iPod Speakers</td>
<td>330.99</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Plasma 3D Television</td>
<td>3199.99</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Bluetooth Headset</td>
<td>707.99</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Zune 30GB</td>
<td>700.99</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>20-Inch Mac</td>
<td>2999.99</td>
<td></td>
</tr>
</tbody>
</table>

You can also learn more about turning an Oracle ADF task flow into a portlet using the Oracle JSF Portlet in the Oracle Magazine article, "Turn Applications into Portlets" available on the Oracle Technology Network at http://webcenter.oracle.com.

10. Below the Top Five Electronics portlet, notice the Activity Stream module (Figure 2–17). The Activity Stream task flow, which displays in this module, tracks recent activities to certain services, like Documents and People Connections.

Figure 2–17  Activity Stream Showing Recent Activities to the Documents and People Connections Services

11. Below the Activity Stream component is a Tag Cloud (Figure 2–18), which is built using the Tagging - Tag Cloud task flow in the Tags service (Chapter 23, "Integrating the Tags Service").
The Tag Cloud is a visual representation of all the tag keywords used in the application. The more frequently the tag keyword is used, the bigger the font; in this case, "fod" is the most commonly used tag, and thus its font is largest in the Tag Cloud.

Now that we've explored the pages and their components that were built using Oracle WebCenter Framework, let's take a look at some of the other tasks you can do.

2.2.3 Editing MyPage.jsp

You'll notice that, at the top of the window, there is an Edit link. Clicking this link enables you to customize the page. By clicking this link, you can customize your own personal pages by adding published portlets that display information, such as the "Top Electronic Sellers" and "My Recommendations" portlets. You can also view properties of components at runtime. This capability was implemented using Oracle Composer, one of Oracle WebCenter Framework's components.

This section describes the following Oracle WebCenter Framework features:

- Oracle Composer
- Oracle Composer extension
- Oracle Composer Toolbar
- Resource Catalog

To use Oracle Composer in the sample application:

1. Next to the "Welcome username" text, click Edit (Figure 2–19).

2. At the top of the page, click the About FOD link (Figure 2–20) to display an example of how you can use the Oracle Composer extension to add information about the application for your users.
The About FOD panel that displays is a custom panel, where you can view details about the demo itself. This is an example of how you can create and register a custom panel in a custom WebCenter application.

3. When you edit the page, you can also view the Oracle Composer toolbar at the top of the page, as shown in Figure 2–21.

Figure 2–21 Oracle Composer Toolbar

<table>
<thead>
<tr>
<th>[Title: Editing Page: My Personal Page]</th>
<th>[Title: Add On Panel Custom Panel Menu Button]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message</td>
<td>Add On Panel</td>
</tr>
<tr>
<td></td>
<td>Custom Panel Menu Button</td>
</tr>
</tbody>
</table>

4. You can use the Add content link to add new content to your page. When you click Add content, a catalog displays (Figure 2–22). This catalog is an example of a customized Resource Catalog (Chapter 12, "Configuring the Resource Catalog for Oracle Composer"). You can log into the application and create different Resource Catalogs for each user, or view the same Resource Catalog across the application.

Figure 2–22 Customized Resource Catalog in the Fusion Order Demo for WebCenter

You can use the Resource the Product Videos link to add a video from YouTube to your application, as you saw in Section 2.2.2, "Exploring MyPage.jspx.” Figure 2–23 shows the Resource Catalog that displays when you choose to add a Product Video to your application.
Now that you’ve taken the tour of the Fusion Order Demo for WebCenter, you can check out these features in depth and use them yourself by referring to the relevant chapters in this manual. You can find the references to these chapters throughout the tour, or, for a quick reference guide, refer to Table 2–1.

### 2.3 Oracle WebCenter Framework Components in the Fusion Order Demo for WebCenter

Table 2–1 describes at a high level the Oracle WebCenter features are included in the Fusion Order Demo for WebCenter. To learn more about the different features, refer to the relevant chapter.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Location in the Demo</th>
<th>Chapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle Composer</td>
<td>Users can customize their own personal pages by adding published portlets that display information, such as the &quot;Top Electronic Sellers&quot; and &quot;My Recommendations&quot; portlets. Users can view properties of components at runtime. This runtime customization is enabled by the Oracle Composer. See Section 2.2.3, &quot;Editing MyPage.jspx&quot; for the location in the demo.</td>
<td>Chapter 7, “Enabling Runtime Editing of Pages Using Oracle Composer.”</td>
</tr>
</tbody>
</table>
### Table 2–1  (Cont.) WebCenter Features in the Fusion Order Demo for WebCenter Sample Application

<table>
<thead>
<tr>
<th>Feature</th>
<th>Location in the Demo</th>
<th>Chapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle Composer extension</td>
<td>Users can view a custom panel (the &quot;About FOD&quot; panel in the demo), which lets you view details about the demo itself. This is an example of how you can create and register a custom panel in a custom WebCenter application.</td>
<td>Chapter 8, &quot;Extending Runtime Editing Capabilities Using Oracle Composer&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See Section 2.2.3, &quot;Editing MyPage.jspx&quot; for the location in the demo.</td>
</tr>
<tr>
<td>Oracle Composer Toolbar and Product Videos on YouTube</td>
<td>Oracle Composer Toolbar enables you to edit the page at runtime and perform such actions as embedding a video within your application.</td>
<td>Chapter 12, &quot;Configuring the Resource Catalog for Oracle Composer&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See Section 2.2.2, &quot;Exploring MyPage.jspx&quot; and Section 2.2.3, &quot;Editing MyPage.jspx&quot; for the location in the demo.</td>
</tr>
<tr>
<td>Resource Catalog</td>
<td>You can log in as different users and view customized resource catalogs.</td>
<td>Chapter 12, &quot;Configuring the Resource Catalog for Oracle Composer&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See Section 2.2.3, &quot;Editing MyPage.jspx&quot; for the location in the demo.</td>
</tr>
<tr>
<td>Activity Stream</td>
<td>See Section 2.2.1, &quot;Exploring the Home.jspx Page&quot; for the location in the demo.</td>
<td>Chapter 20, &quot;Integrating the People Connections Service&quot;</td>
</tr>
<tr>
<td></td>
<td>Track recent updates to services like People Connections and Documents in this easy-to-read module.</td>
<td></td>
</tr>
<tr>
<td>Discussion Forums and Announcements</td>
<td>Users can read the reviews of a particular product, as well as post their own comments using the Discussions service.</td>
<td>Chapter 17, &quot;Integrating the Discussions Service&quot;</td>
</tr>
<tr>
<td></td>
<td>The application administrator can use the Announcements service to display news about the products, such as discounts or recall information.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>See Section 2.2.1, &quot;Exploring the Home.jspx Page&quot; for the location in the demo.</td>
<td></td>
</tr>
<tr>
<td>Documents</td>
<td>When you view information about a particular product in the application, you can access documents such as product manuals in different formats (PDF, Microsoft Word, GIF, JPG, and so on). These documents are stored in a content management system.</td>
<td>Chapter 14, &quot;Integrating the Documents Service&quot;</td>
</tr>
<tr>
<td></td>
<td>See Section 2.2.1, &quot;Exploring the Home.jspx Page&quot; for the location in the demo.</td>
<td></td>
</tr>
</tbody>
</table>
Using the Links service, you can quickly access, as well as create links to and from related product information from the document library, the discussion forums, and announcements. For example, if you are viewing a PDF that mentions an iPod, you can click a link that takes you directly to more information about iPods and their accessories.

See Section 2.2.1, "Exploring the Home.jspx Page" for the location in the demo.

This service tracks recent updates to some of the services in the application: Discussions, Announcements, and Documents.

See Section 2.2.2, "Exploring MyPage.jspx" for the location in the demo.

This service tracks recent updates to some of the services in the application: Discussions, Announcements, and Documents.

See Section 2.2.2, "Exploring MyPage.jspx" for the location in the demo.

You can view RSS feeds, in this case news about the Fusion Order Demo for WebCenter.

See Section 2.2.2, "Exploring MyPage.jspx" for the location in the demo.

This search feature enables you to search the content from all the services in the Fusion Order Demo for WebCenter, including discussion forums, announcements, the document library, and tagged content.

See Section 2.2.2, "Exploring MyPage.jspx" for the location in the demo.

You can use the Tags service to tag content in the document library, as well as add a Tag Cloud to your application to show the tags used. Custom tagging is used to tag a sample product.

See Section 2.2.1, "Exploring the Home.jspx Page" and Section 2.2.2, "Exploring MyPage.jspx" for the locations in the demo.
<table>
<thead>
<tr>
<th>Feature</th>
<th>Location in the Demo</th>
<th>Chapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle JSF Portlet Bridge</td>
<td>This portlet is based on an ADF task flow and displays the &quot;Top Five Electronics&quot; in the demo.</td>
<td>Chapter 30, &quot;Creating Portlets with the Oracle JSF Portlet Bridge&quot;</td>
</tr>
<tr>
<td></td>
<td>See Section 2.2.2, &quot;Exploring MyPage.jspx&quot; for the location in the demo.</td>
<td></td>
</tr>
<tr>
<td>Standards-Based Java (JSR 168) Portlet</td>
<td>This portlet contains product recommendations in a standards-based JSR 168 portlet, which you can add from a customized Resource Catalog.</td>
<td>Chapter 31, &quot;Creating Portlets with the Portlet Wizard&quot;</td>
</tr>
<tr>
<td></td>
<td>See Section 2.2.2, &quot;Exploring MyPage.jspx&quot; for the location in the demo.</td>
<td></td>
</tr>
</tbody>
</table>
Preparing Your Development Environment

The WebCenter Extension bundle provides tools for adding, managing, and securing content in a JSP application. Such tools include a declarative security tool; a life cycle management tool; and tools for consuming portlets, WebCenter services, and content. All of these are designed to enable you to better organize your information around the task at hand. After following a few preparatory procedures, you can use these tools to simplify the creation of WebCenter applications and projects.

This chapter steps you through the process of installing the WebCenter Extension bundle and getting started on WebCenter and portlet applications.

This chapter includes the following sections:

- Section 3.1, "Installing the WebCenter Extension Bundle"
- Section 3.2, "Creating a WebCenter Application"
- Section 3.3, "Creating WebCenter Application-Enabled Pages"
- Section 3.4, "Creating a Portlet Application"
- Section 3.5, "Implementing Security in Your Application"
- Section 3.6, "Extending Non-WebCenter Applications to Include WebCenter Capabilities"
- Section 3.7, "Installing the Oracle WebCenter Adapter for Microsoft SharePoint"
- Section 3.8, "Creating a WebCenter Application Project By Importing a WAR File"
- Section 3.9, "Using Integrated WLS"
- Section 3.10, "Accessing Connection Wizards"
- Section 3.11, "Integrating Mobile Support"

3.1 Installing the WebCenter Extension Bundle

The WebCenter extension bundle is a JDeveloper add-in that provides all WebCenter capabilities to the JDeveloper Studio Edition. To create a custom WebCenter application, you must first install the WebCenter extension bundle in an instance of Oracle JDeveloper Studio Edition.

To install the WebCenter extension bundle:

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

4. From the Help menu, select Check for Updates.

5. Click Next in the Welcome page of the Check for Updates wizard.

6. On the Source page, under Search Update Centers, search for the WebCenter extension, select it, then click Finish.

7. When prompted, restart JDeveloper.

JDeveloper is now configured to create WebCenter and portlet applications.

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

http://webcenter.oracle.com

3.2 Creating a WebCenter Application

To readily find and use WebCenter components in your application, you must ensure that the right technology scopes are set and tag libraries added. Once you do this, relevant tags are included in the Component Palette and relevant context menus become available in JDeveloper. The easiest way to ensure that scopes are set properly and the right tag libraries are added is to create your application using a template.

Oracle WebCenter provides two templates for creating applications:

- WebCenter Application template—Prepopulates the application with projects optimally scaled for the creation of Java Content Repository (JCR) data controls and consumption of portlets and WebCenter Services.

- Portlet Producer Application template—Prepopulates the application with a project scoped for creation of JSR 168 (standards-based) and Oracle PDK-Java portlets.

It is not required that you use these templates. If you prefer, you can create your own WebCenter applications and portlet applications by manually scoping the application technologies and creating the relevant projects.

If you are working with an existing application that was created with some other template, you can also extend it with WebCenter components. To do this, you must manually add any required projects, technology scopes, and tag libraries. For information about extending existing applications, see Section 3.6, "Extending Non-WebCenter Applications to Include WebCenter Capabilities."

This section describes how to create an application using the WebCenter Application template. It includes the following subsections:

- Section 3.2.1, "How to Create a WebCenter Application Using a Template"

- Section 3.2.2, "What Happens When You Use the WebCenter Application Template"

**Note:** Technology scopes control the options that appear by default in the New Gallery dialog. They can also be used to control the context menus that appear in JDeveloper and to add tag libraries to the project so that tags appear in the Component Palette.
3.2.1 How to Create a WebCenter Application Using a Template

The easiest way to ensure that the right tag libraries are included and the right technology scopes are set for a new WebCenter application is to create the application using the WebCenter Application template. Application templates provide a means of easily creating an application with a pre-defined set of projects and technology scopes. Templates also provide a means of partitioning an application into projects that reflect a logical separation of the overall work.

A project typically represents a group of related components that can be deployed into their own logical—and possibly physical—tier. You can deploy application projects to the same WebLogic Server instance on a single host or to different WebLogic Server instances, potentially running on different hosts.

The WebCenter Application template consists of a Model project for the data model and a ViewController project for consuming portlets, components, and data controls. The WebCenter Application template folds both of these projects into one application for simplicity, but you can arrange your applications and projects the way that best fits your circumstances. For example, if you are either the portlet developer or the application developer—but not both—then it is more likely that you would benefit from a separate application for the WebCenter application and for the portlets the application consumes.

To create an application using the WebCenter Application template:

1. Access the application creation wizard in any of the following ways:
   - From the File menu, choose New. In the New Gallery dialog, expand Applications, select WebCenter Application, and click OK.
   - From the Application Navigator, select New Application.
   - If you have an existing application that is open, then in the Application Navigator, right-click <Application-Name> and choose New. In the New Gallery dialog, expand Applications, select WebCenter Application, and click OK.
   - If you have an existing application that is open, then in the Application Navigator, from the <Application-Name> drop down menu, select New Application.

2. In the Name your application page of the creation wizard, enter a name for the application in the Application Name field.

3. In the Directory field, enter a path to the directory where the application should be stored, 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.

5. From the Application Template list in the creation wizard, select WebCenter Application. Perform this step only if you invoked the wizard by clicking New Application in the Application Navigator.

6. Click OK.
   - The template prepopulates the application with two projects:
Creating a WebCenter Application

- **Model**, scoped for data modeling and content sourcing
- **ViewController**, scoped for creation of WebCenter application pages and page elements and for registering portlet producers

7. To create the new WebCenter application with the default settings for the two projects, click **Finish**.

   Alternatively, configure the projects according to your requirements by navigating through the remaining pages of the wizard. Then click **Finish**.

   For more information about the projects and technology scope, see Section 3.2.2, "What Happens When You Use the WebCenter Application Template."

### 3.2.2 What Happens When You Use the WebCenter Application Template

Using the WebCenter Application template generates default projects with unique technology scopes, libraries, and default files. Additionally, it limits the types of options that are exposed in the JDeveloper user interface to those appropriate to the project type.

---

**Note:** In addition to populating the application with default files and folders, the template populates the Resource Palette with the WebCenter Services Catalog. The WebCenter Services Catalog contains data controls and task flows that you can use to integrate Oracle WebCenter Services in your application. For more information, see Part III, "Integrating, Presenting, and Publishing Content".

---

This section provides an overview of what occurs when you use the WebCenter Application template. It includes the following subsections:

- Section 3.2.2.1, "Template Projects, Technology Scopes, and Libraries"
- Section 3.2.2.2, "WebCenter Application Template Default Files and Folders"

#### 3.2.2.1 Template Projects, Technology Scopes, and Libraries

By default, the WebCenter Application template names its two projects **Model** and **ViewController**. If you prefer, you can rename the projects before or after you create the application. Rename projects before you create the application through the Manage Templates command on the Tools menu. Rename projects after you create the application through the Rename command on the File menu.

When you use the WebCenter Application template, JDeveloper does the following for you:

- Creates **Model** and **ViewController** projects, each with their own relevant technology scopes and libraries.

  **Table 3–1** lists the default technology scopes and libraries that are included with applications created with the WebCenter Application template. As you start building the application and adding components to the project, the list of libraries grows rapidly.
When you work in each project, the New Gallery is filtered to show only choices that are relevant to the selected technologies. For example, the option to create a Content Repository data control is available in the New Gallery while the Model project is selected, but that option is not available while the ViewController project is selected. Similarly, options for creating JSP pages are available while the ViewController project is selected, but those options are not available while the Model project is selected.
- Creates a starter application deployment descriptor `web.xml` file with default settings. You will find the `web.xml` file in the Application Navigator under the ViewController project's `WEB-INF` folder. For more information about `web.xml`, see Section 3.2.2.2, "The WebCenter Application Template Default `web.xml` File."

- Creates an empty `faces-config.xml` file in the ViewController project's `WEB-INF` folder. The `faces-config.xml` file is a JSF configuration file where JSF application resources, such as custom validators and managed beans, are registered and all page-to-page navigation rules are defined. For more information about `faces-config.xml`, see Section 3.2.2.2.3, "The WebCenter Application Template Default `faces-config.xml` File."

- Adds `jsf-impl.jar` in the ViewController project's `WEB-INF\lib` folder. This JAR file is the JSF reference implementation that JDeveloper includes by default.

- Creates an empty `adfc-config.xml` file. This file can be used to set up routers to conditionally control flow between pages.


3.2.2.2 WebCenter Application Template Default Files and Folders

When you use the WebCenter Application template to create an application, it creates a folder hierarchy with an accompanying set of default files. This section provides detailed information about default files and the file hierarchy. It contains the following subsections:

- Section 3.2.2.2.1, "The WebCenter Application Default Folder Hierarchy"
- Section 3.2.2.2.2, "The WebCenter Application Template Default `web.xml` File"
- Section 3.2.2.2.3, "The WebCenter Application Template Default `faces-config.xml` File"
- Section 3.2.2.2.4, "The WebCenter Application Template Default `trinidad-config.xml` File"
- Section 3.2.2.2.5, "The WebCenter Application Template Default `adfc-config.xml` File"

3.2.2.2.1 The WebCenter Application Default Folder Hierarchy

Because there are several possible views of the Oracle JDeveloper Application Navigator, the hierarchy of folders in a WebCenter application may differ when you compare folders in the Application Navigator with the same folders in your file system. For example, Figure 3–1 illustrates one view of a newly created application in the Oracle JDeveloper Application Navigator.
Figure 3–1 Application from WebCenter Application Template in JDeveloper

Figure 3–2 Application from WebCenter Application Template in a Windows File System

If you prefer to view the Windows file system hierarchy in the Application Navigator, click the **Navigator Display Options** icon in the Projects panel and select **Group by Directory**.

3.2.2.2 The WebCenter Application Template Default web.xml File

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

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

Typical runtime settings include initialization parameters, custom tag library location, and security settings. Depending on the technology you use, other settings may be added to `web.xml` as appropriate.
Example 3–1 illustrates the default web.xml file provided through the WebCenter Application template. This file is located in the /public_html/WEB-INF directory relative to the ADF application’s ViewController project.

**Example 3–1  Default web.xml File Provided Through the WebCenter Application Template**

```xml
<?xml version = '1.0' encoding = 'windows-1252'?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
  http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd" version="2.5"
  xmlns="http://java.sun.com/xml/ns/javaee">
  <context-param>
    <param-name>javax.faces.STATE_SAVING_METHOD</param-name>
    <param-value>client</param-value>
  </context-param>
  <context-param>
    <description>If this parameter is true, there will be an automatic check
    of the modification date of your JSPs, and saved state will be discarded when
    JSP’s change. It will also automatically check if your skinning css files
    have changed without you having to restart the server. This makes development
    easier, but adds overhead. For this reason this parameter should be set to
    false when your application is deployed.</description>
    <param-name>org.apache.myfaces.trinidad.CHECK_FILE_MODIFICATION</param-name>
    <param-value>false</param-value>
  </context-param>
  <context-param>
    <description>Whether the ‘Generated by...’ comment at the bottom of ADF
    Faces HTML pages should contain version number information.</description>
    <param-name>oracle.adf.view.rich.versionString.HIDDEN</param-name>
    <param-value>false</param-value>
  </context-param>
  <filter>
    <filter-name>trinidad</filter-name>
    <filter-class>org.apache.myfaces.trinidad.webapp.TrinidadFilter</filter-class>
  </filter>
  <filter>
    <filter-name>ServletADFFilter</filter-name>
    <filter-class>oracle.adf.share.http.ServletADFFilter</filter-class>
  </filter>
  <filter-mapping>
    <filter-name>trinidad</filter-name>
    <servlet-name>Faces Servlet</servlet-name>
    <dispatcher>FORWARD</dispatcher>
    <dispatcher>REQUEST</dispatcher>
  </filter-mapping>
  <filter-mapping>
    <filter-name>ServletADFFilter</filter-name>
    <servlet-name>Faces Servlet</servlet-name>
    <dispatcher>FORWARD</dispatcher>
    <dispatcher>REQUEST</dispatcher>
  </filter-mapping>
  <servlet>
    <servlet-name>Faces Servlet</servlet-name>
  </servlet>
</web-app>
```

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.
<servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
<load-on-startup>1</load-on-startup>
</servlet>
<servlet>
  <servlet-name>resources</servlet-name>
</servlet>
<servlet-mapping>
  <servlet-name>Faces Servlet</servlet-name>
  <url-pattern>/faces/*</url-pattern>
</servlet-mapping>
<servlet-mapping>
  <servlet-name>resources</servlet-name>
  <url-pattern>/adf/*</url-pattern>
</servlet-mapping>
<servlet-mapping>
  <servlet-name>resources</servlet-name>
  <url-pattern>/afr/*</url-pattern>
</servlet-mapping>
</web-app>

When you first create your WebCenter 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 (Example 3–1) 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 (Example 3–1) provides information about the ADF resource servlet used to serve up web application resources (images, style sheets, JavaScript libraries) by delegating to a ResourceLoader.

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

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

---

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

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

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

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

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

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

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

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

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

Example 3–2 illustrates the default faces-config.xml file provided through the WebCenter Application template. This file is located in the /public_html/WEB-INF directory relative to the ADF application's ViewController project.

Example 3–2 Default faces-config.xml File Provided Through the WebCenter Application Template

```xml
<?xml version="1.0" encoding="windows-1252"?>
<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>
  </application>
  <lifecycle>
    <phase-listener>oracle.adf.controller.faces.lifecycle.ADFPhaseListener</phase-listener>
  </lifecycle>
</faces-config>
```

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

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

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

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

3.2.2.2.4 The WebCenter Application Template Default trinidad-config.xml File  The default trinidad-config.xml file, available when you create an 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 3–3 illustrates the default trinidad-config.xml file provided through the WebCenter Application template. This file is located in the /public_html/WEB-INF directory relative to the ADF application’s ViewController project.

Example 3–3  Default trinidad-config.xml File Provided Through the WebCenter Application Template

<?xml version="1.0" encoding="windows-1252"?>
<trinidad-config xmlns="http://myfaces.apache.org/trinidad/config">
  <skin-family>blafplus-rich</skin-family>
</trinidad-config>

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

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


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

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

The adfc-config.xml file is located in the /public_html/WEB-INF directory relative to the ADF application’s ViewController project.
Creating WebCenter Application-Enabled Pages

3.3 Creating WebCenter Application-Enabled Pages

Create customizable WebCenter application pages in JDeveloper as JSP documents (.jspx) rather than JSP pages (.jsp). For page customizations to be stored, the page must be a JSF page represented in XML. By choosing to create an XML representation of the JSP document (.jspx), you ensure that customizations are always possible for the WebCenter application page.

If you are not planning to enable customization, you can create a JSP page; however this choice might cause problems if you later decide that you want to enable customization.

This section explains how to create WebCenter application-enabled pages. It contains the following subsections:

- Section 3.3.1, “How to Create a Page in a WebCenter Application Based on a Template”
- Section 3.3.2, "How to Create a Page in a Manually-Created WebCenter Application”

Note: For detailed information about creating pages in JDeveloper with Oracle ADF, see the Oracle Fusion Middleware Web User Interface Developer’s Guide for Oracle Application Development Framework. It contains all the information you need about Oracle ADF Faces and building basic and complex pages.

3.3.1 How to Create a Page in a WebCenter Application Based on a Template

When you create a WebCenter application using the WebCenter Application template, all the libraries necessary for adding WebCenter Web 2.0 Services and Composer components to your page are available by default. Additionally, runtime page customizations are enabled by default.

To create a customizable WebCenter application-enabled JSPX page:

1. Start JDeveloper.
2. In the Application Navigator, go to the ViewController project of the application in which you want to create a customizable page, right click the ViewController project, and choose New.

Note: For information about how to create a WebCenter application if you have not yet created one, see Section 3.2, “Creating a WebCenter Application.”

3. In the New Gallery, expand Web Tier, select JSF and then JSF Page, and click OK.
4. In the Create JSF Page dialog (Figure 3–3), enter a file name for the page.

**Figure 3–3  The Create JSF Page Dialog**

5. Select Create as XML Document (*.jspx) and click OK.

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

### 3.3.2 How to Create a Page in a Manually-Created WebCenter Application

To create a JSPX page in a manually-created WebCenter application or non-WebCenter application, you must first add the necessary libraries to the page’s home project. Adding the required libraries ensures that Web 2.0 Services and Oracle Composer component tag libraries are available in the Component Palette. You can then drag and drop components from these libraries onto your page.

Add the libraries specified in this section to enable page customization in the following types of applications:

- A WebCenter application created manually rather than through a template. For more information, see Section 3.6, “Extending Non-WebCenter Applications to Include WebCenter Capabilities.”
- A non-WebCenter application that was created using no template or any template other than the WebCenter Application template.
To add the necessary libraries:

1. Right-click your project and choose **Project Properties** from the context menu.
2. In the left panel of the Project Properties dialog, click **JSP Tag Libraries**.
3. Click **Add**.
4. Add the following libraries, and click **OK**:
   - Oracle Composer 1.0
   - WebCenter Customizable Components Rich 11
   - Customizable Components (HTML) 11
   - WebCenter IM and Presence Tags 11.1.1.1.0
   - WebCenter Links Service 11.1.1.1.0
   - WebCenter Tagging 11.1.1.1.0

You can now create a WebCenter Application-enabled JSPX page by performing the steps in Section 3.3.1, "How to Create a Page in a WebCenter Application Based on a Template."

### 3.4 Creating a Portlet Application

Use portlets to provide controlled display of and access to local and remote content sources. You can create portlets as part of your application development effort when you create an application and project scoped for portlets. With the appropriate scoping in place, Oracle JDeveloper provides options that are tailored to creating portlets. For example, in an application scoped for portlets the Web Tier node appears in the New Gallery, and portlet creation is allowed.

You can scope an application and its projects for portlet creation in either of the following ways:

- Use the Portlet Producer Application template
- Extend an application by optimizing one of its projects for portlet creation

**Notes:** Use separate applications to build and consume portlets. Building and consuming portlets within the same application can cause problems during packaging and deployment.

This section describes the steps to create a portlet application using the Portlet Producer Application template. It contains the following subsections:

- **Section 3.4.1,** "How to Create a Portlet Application Using a Template"
- **Section 3.4.2,** "What Happens When You Use a Portlet Producer Application Template"

For information about extending an existing application to enable portlet creation, see Section 3.6, "Extending Non-WebCenter Applications to Include WebCenter Capabilities."

### 3.4.1 How to Create a Portlet Application Using a Template

The Portlet Producer Application template consists of a single project, **Portlets.** Use the Portlets project to create JSR 168 (standards-based) and Oracle PDK-Java portlets.
To create an application using the Portlet Producer Application template:

1. Access the application creation wizard in any of the following ways:
   - From the File menu, choose New. In the New Gallery dialog, expand Applications, select Portlet Producer Application, and click OK.
   - From the Application Navigator, select New Application.
   - If you have an application open, then in the Application Navigator open the application name drop down menu and select New Application.
   - In the Application Navigator, right-click the application name and choose New. In the New Gallery dialog, expand Applications, select Portlet Producer Application, and click OK.

2. In the Name your application page of the creation wizard, enter a name for the application in the Application Name field.

3. In the Directory field, accept the default path or enter a path to the directory where the application should be stored.
   For example:
   
   C:\JDeveloper\mywork\Application1

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

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

5. From the Application Template list, select Portlet Producer Application.

   **Note:** The Application Template list appears only when you invoke the wizard by clicking New Application in the Application Navigator as described in step 1.

6. Click Finish to create the portlet application with default project configurations.

### 3.4.2 What Happens When You Use a Portlet Producer Application Template

Using the Portlet Producer Application template generates the default Portlets project with the Java, JSP and Servlets, and Portlet technology scopes. It limits the types of options that are exposed in the JDeveloper user interface to those appropriate for creating portlets.

When you right-click this new project, options to create JSR 168 and Oracle PDK-Java portlets are provided.

### 3.5 Implementing Security in Your Application

The use of Oracle ADF security enables WebCenter applications to easily adjust to real-world business security requirements because, rather than securing paths, you secure actions with JAAS. JAAS-based Oracle ADF Security provides:

- More granular security. Because Java EE security is URL-based or page-based, it is not possible to have a finer level of access control. With Oracle ADF security, different roles can perform different levels of activity at the same URL.
Implementing Security in Your Application

- The ability to create new roles and their associated access privileges at runtime. This capability flows from the fact that policies storage is external to the application.

- Simplified permission assignment through hierarchical roles, which enable the inheritance of permissions. While the roles used by Java EE security constraints are flat, JAAS permissions reference enterprise roles, which can be nested.

---

**Note:** For details on how to implement a complete security solution for your custom application, see the section on enforcing ADF Security in a Fusion Web application in the Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework.

---

Some services must know the identity of the user. For example, for saving searches the Search service requires the user’s identity. For these services, you must at least configure your application to authenticate users so that they have distinct identities for personalization and preferences. Such a security configuration is useful in testing how user privileges impact runtime customization capabilities.

This section covers the basic security configuration required to help in testing your application in Integrated WebLogic Server (WLS). It contains the following subsections:

- Section 3.5.1, "How to Configure ADF Security"
- Section 3.5.2, "How to Create an Application Role"
- Section 3.5.3, "How to Add Login and Logout Links"

### 3.5.1 How to Configure ADF Security

This section provides information about implementing a basic security solution.

To enforce Oracle ADF Security for the application:

1. Open your application, and select the ViewController project.
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 WebCenter applications.
4. Click Next.

5. Select the authentication type you want your application to use when users submit their login information.

   To make the process of securing your application easy, select Form-Based Authentication and then the Generate Default Pages option, as shown in Figure 3–5.

   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 Chapter 37, "Securing Your WebCenter Application."
6. Click Next.

7. On the Enable automatic policy grants page, shown in Figure 3–6, select whether to make ADF resources public without requiring application developers to first define ADF policy grants. For a basic 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 3–6 Automatic Policy Grants Page
When you enable automatic policy grants, a View grant is made to all members of the test-all application 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. For more information about the specific grants provided through task flows, see Section 4.1.3, "Automated Task Flow Grants."

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 services task flows are consumed in an application, the WebCenter extension automatically adds appropriate grants for these task flows, giving View access to the role authenticated-role.

---

**Note:** Grants made to the role authenticated-role are visible in the source view of the jazn-data.xml file. If the default grant to authenticated-role is not appropriate to the application, you can change it by manually updating the source view of the application’s jazn-data.xml.

---

Select **Grant to Existing Objects Only** when you want JDeveloper to grant View privileges to the test-all 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 3-7.

You can either create a custom welcome page or let the wizard create a default one. To keep the process simple, select **Generate Default**.

For information about creating a custom welcome page, see Chapter 37, "Securing Your WebCenter Application."
Implementing Security in Your Application

10. The Summary page opens, as shown in Figure 3–8.

11. Click Finish.

   It may take several moments for the wizard to complete.

12. On the Security Infrastructure Created dialog, click OK.
3.5.2 How to Create an Application Role

You can test your ADF Security settings by creating a role and a user, and giving the user privileges on your JSF page. Then, as you add services, you can use this user’s credentials to test the service at runtime.

To create an application role:

1. In the Application Resources panel, navigate to the jazn-data.xml file, which is available under the META-INF folder.

2. Access the Edit JPS Identity and Policy Store dialog using any of the following methods:
   - Right-click jazn-data.xml and select Properties from the context menu.
   - Right-click jazn-data.xml and select Open from the context menu. Click Application Roles on the page.
   - Right-click the application name, select Secure and then Users and Groups.

3. In the Edit JPS Identity and Policy Store dialog, click Roles under the jazn.com realm and then click Add.

4. In the Add Role dialog, enter a name for the role, for example, admin.

5. Click OK, then click OK again.
   
   You have now created a sample role for testing purposes.

6. Save the file.

Similarly, you can add users to the application by specifying the user name and password. For example, if you have a user named fmwadmin for your Discussions server, you may want to use the same user ID, so that this user can authenticate with the Discussions server. You can then associate this user with the admin role you created.

3.5.3 How to Add Login and Logout Links

You can add login and logout links to your public welcome page so that users can explicitly log in and out while they are in the application. For information about adding login and logout links, see the chapter titled “Enforcing ADF Security in a Fusion Web application” in the Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework.

3.5.4 What Happens at Runtime

When you run the application, a login page is rendered. Subsequently, a welcome page is rendered for successfully-authenticated users, and an error message is rendered for users without the required privileges.

3.6 Extending Non-WebCenter Applications to Include WebCenter Capabilities

If your business requirement was to create a non-WebCenter application, say a Fusion web application, but you want to also include WebCenter capabilities in that application, you can do so by adding the required technology scopes and libraries. The technology scopes limit UI options to those appropriate to the type of application project you are building. The libraries provide the components and elements useful in constructing application content. You can extend an existing application to integrate
content from various repositories, create portlets, or consume portlets and WebCenter services. You can extend ADF applications from the current release or from a 10.1.x.x release.

The easiest way to understand what to specify for a WebCenter application or portlet application is to review the out-of-the-box template’s characteristics. You can then mimic these in your own applications or templates.

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

To enable WebCenter capabilities in an application:

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

   ![Note](image)
   *Note:* If you migrated a 10.1.3.x WebCenter application, then the application continues to have the Model, ViewController, and Portlets projects.

2. Optionally, create new projects for defining the application data model, consuming content, or creating portlets:
   a. In the Application Navigator, right-click the application and select New Project from the context menu.
   b. In the New Gallery, expand General, select Projects, then Generic Project, and click OK.
   c. In the Create Generic Project wizard, in the Project Name field, enter a name for the project.
      For example View.
   d. In the Directory field, specify the directory path for storing the project or accept the default path.
   e. Click Finish to create the new project.
3. Right-click a project and select Project Properties.
4. In the Project Properties dialog, add the required technologies and libraries depending on which WebCenter-related capabilities you want to enable in the selected project:
   - To define the application data model:
     Include the technologies and libraries listed against the Model project in Table 3–1.
   - To consume content:
     Include the technologies and libraries listed against the ViewController project in Table 3–1.
   - To create portlets:
     Include the WebCenter Portlet Creation Service technology.
5. Save your work.
3.7 Installing the Oracle WebCenter Adapter for Microsoft SharePoint

Before you configure the Oracle WebCenter adapter for Microsoft SharePoint as described in the section "How to Create a Content Repository Connection Based on the Oracle WebCenter Adapter for Microsoft SharePoint", you must install it in JDeveloper. The adapter files are located in the Oracle Fusion Middleware 11g Companion DVD in the This zip is in directory /Disk1/WebCenter/services/content/adapters.

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, and then select oracle.webcenter.content.jcr.sharepoint_bundle.zip in the /Disk1/WebCenter/services/content/adapters directory in Oracle Fusion Middleware 11g Companion DVD.
5. Click Finish. When prompted, restart JDeveloper.

3.8 Creating a WebCenter Application Project By Importing a WAR File

This procedure is included in the event that you have an existing WAR file you want to import into your WebCenter application through JDeveloper.

---

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

---

To import a WAR file into JDeveloper:

1. Right-click your application in the Application Navigator, and choose New Project from the context menu.
2. In the New Gallery, expand General, select Projects, then Project from War File, and click OK.
3. Follow the wizard instructions to complete creating the project.
4. Right-click the newly created project in the Application Navigator, and select Project Properties from the context menu.
5. Add the required technologies and libraries depending on which WebCenter-related capabilities you want to enable in the selected project.

For information about required technologies and libraries, see Table 3–1.
6. Click OK to close the Project Properties dialog.

3.9 Using Integrated WLS

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

- Section 3.9.1, "How to Start and Stop Integrated WLS"
- Section 3.9.2, "What You May Need to Know About Integrated WLS"
- Section 3.9.3, "The WebCenter Preconfigured Server Readme File"
- Section 3.9.4, "What You May Need to Know About Preconfigured Portlet Producers"

### 3.9.1 How to Start and Stop Integrated WLS

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

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

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

There are a couple of ways to determine if Integrated WLS is running:

- From the View menu, select **Log** and look for the following entry:
  
  DefaultServer started

- Access the Integrated WLS console from a browser:

  http://localhost:7101/console

---

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

---

### 3.9.2 What You May Need to Know About Integrated WLS

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

You can use Integrated WLS as a platform for pretesting WebCenter application deployments on your local computer by establishing an application server connection to it from Oracle JDeveloper. When you run the application in Integrated WLS, it is actually deployed as if you were deploying it to a WebLogic Server instance in an application server. For more information about Integrated WLS, see Section 38.3, "Packaging and Deploying a Custom WebCenter Application to a WebLogic Managed Server.”
3.9.3 The WebCenter Preconfigured Server Readme File

The WebCenter Preconfigured Server Readme file contains valuable information about how to use Integrated WLS. Additionally, it contains links to the preconfigured portlet producers. You can access the Preconfigured Server Readme file by selecting WebCenter Preconfigured Server Readme from the Oracle JDeveloper Help menu.

3.9.4 What You May Need to Know About Preconfigured Portlet Producers

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

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

- Section 3.9.4.1, "OmniPortlet and Web Clipping"
- Section 3.9.4.2, "The WSRP Sample Portlet Producers and Portlets"
- Section 3.9.4.3, "The PDK-Java Sample Portlet Producer and Portlets"

For information about registering portlet producers, see Section 34.2, "Registering Portlet Producers with a Custom WebCenter Application." For information about adding portlets to pages, see Section 34.3, "Adding Portlets to a Page."

3.9.4.1 OmniPortlet and Web Clipping

The Integrated WLS contains PortalTools, which provides access to the design time at runtime OmniPortlet and Web Clipping portlet. Design time at runtime means that users define portlet content after the portlet is placed on an application page and the page is run. This concept is explained more fully in Section 29.2.6, "Portlet Creation Style."

To access OmniPortlet and Web Clipping portlet producers:

1. Start the Integrated WLS (see Section 3.9.1).
2. From the Help menu, select WebCenter Preconfigured Server Readme.
3. In the Readme file, go to the heading Preconfigured Portlet Producers, and click the PortalTools Welcome Page link under PortalTools Portlet Producers.
This opens the PortalTools Welcome page.

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

**Note:** For information about registering a portlet producer, see Section 34.2, "Registering Portlet Producers with a Custom WebCenter Application."

Once you have registered a producer, its portlets become available on Oracle JDeveloper's Application Resources panel. In the Application Resources panel, under the Connections node, select a producer name to list its portlets, then drag a portlet onto a WebCenter application page. (For information about adding portlets to pages, see Section 34.3, "Adding Portlets to a Page.")

The PortalTools Welcome page contains producer URLs for three producers:

- **The Web Clipping producer** provides the Web Clipping portlet, which is a browser-based declarative tool that enables dynamic reuse of content from another web source. When the source changes, the content in the Web Clipping portlet also changes. With the Web Clipping portlet, you use a web browser to navigate to the web page that contains the desired content. Using Web Clipping Studio, which is accessed through the portlet, drill down through a visual rendering of the target page to choose the desired content. For detailed information about the Web Clipping portlet, see Chapter 36, "Creating Content-Based Portlets with Web Clipping."

- **The OmniPortlet producer** provides OmniPortlet, which is a declarative portlet-building tool that enables you to build portlets against a variety of data sources, including XML files, character-separated value files (for example, spreadsheets), web services, databases, web pages, and SAP data sources. OmniPortlet users can also choose a prebuilt layout for the data. Prebuilt layouts include tabular, news, bullet, form, chart, or HTML. For information about OmniPortlet, see Chapter 35, "Creating Portlets with OmniPortlet."

- **The Sample Portlet Producer** is useful for illustrating the potential of an OmniPortlet. The sample producer is for demonstration and should not be used to create real-use portlet instances.

### 3.9.4.2 The WSRP Sample Portlet Producers and Portlets

The Integrated WLS includes sample WSRP portlet producers and portlets you can use with your application. For example, the WSRP WSDL URLs on the Rich Text Portlet Producer page include the Rich Text portlet, which offers browser-based rich text editing at runtime. To learn more about WSRP and the sample portlets, see Section 29.1.2.2, "Rich Text Portlet" and Section 29.1.2.4, "Parameter Form and Parameter Display Portlets."

To access WSRP sample portlet producers:

1. Start Integrated WLS (see Section 3.9.1).
2. From the Help menu, select WebCenter Preconfigured Server Readme.
3. In the Readme file, go to the heading Preconfigured Portlet Producers, and click the link for the Rich Text and Parameter Form Portlet Producer or Sample Portlets under WSRP Portlet Producers.
Both links open different WSRP Producer Test Pages—one for different WSRP producer versions of the Rich Text portlet, the other for different WSRP producer versions of sample portlets.

4. Copy the Web Services Description Language (WSDL) URL for a WSRP producer—either WSRP v1 WSDL or WSRP v2 WSDL.

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

---

**Note:** For information about registering a portlet producer, see Section 34.2, “Registering Portlet Producers with a Custom WebCenter Application.”

---

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

### 3.9.4.3 The PDK-Java Sample Portlet Producer and Portlets

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

To access PDK-Java sample portlet producers:

1. Start Integrated WLS (see Section 3.9.1).
2. From the Help menu, select WebCenter Preconfigured Server Readme.
3. In the Readme file, go to the heading Preconfigured Portlet Producers, and copy the link location for the PDK-Java Sample Producer or PDK-Java Struts Sample Producer link under PDK-Java Portlet producers.

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

---

**Note:** For information about registering a portlet producer, see Section 34.2, "Registering Portlet Producers with a Custom WebCenter Application.”

---

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

### 3.10 Accessing Connection Wizards

To consume portlets and certain WebCenter services in your application pages, you must first set up connections to portlet producers and various data source repositories. For example, the Documents service requires a Content Repository connection, and the Links service requires a Database connection.

This section describes the different ways to access the wizards for creating new connections. The actual steps to create the connections are discussed in their respective chapters in this guide.
Depending on how you invoke a wizard to create a connection, the connection is created in one of the following locations:

- **Under Application Resources in the Application Navigator**
  
  Connections created here can be used in the current application only. This is the most common way to create a repository connection.

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

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

  Connections created here can be reused across WebCenter 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.

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.

You can also access a connection wizard from the New Gallery dialog. If you use this method, you must specify where to create the connection—under Application Resources or IDE Connections.

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.

**See Also:** For information about creating and consuming connections:

- Chapter 13, "Integrating Content"
- Chapter 34, "Consuming Portlets"
- Part III, "Integrating, Presenting, and Publishing Content"
3.11 Integrating Mobile Support

Application Development Framework Mobile (ADF Mobile) is a standards-based framework that enables the rapid development of enterprise mobile applications. You can implement mobile support for your WebCenter application just like you would for any other Oracle ADF application.

Because ADF Mobile is built upon the component model of Java Server Faces (JSF), you can quickly develop applications for PDA browsers. ADF Mobile's mobile-specific extensions to JSF enable you to develop mobile applications using the same methodologies for developing JSF applications for the desktop. For more information about ADF Mobile, see Oracle Fusion Middleware Mobile Browser Client Developer’s Guide for Oracle Application Development Framework.

You can implement mobile support for WebCenter services by creating a new ADF Mobile front end using the APIs provided with WebCenter services and included products, such as Oracle WebCenter Discussions Server and Oracle Universal Content Management.
Preparing Your Application for Oracle WebCenter Services

This chapter describes how to prepare your application to consume Oracle WebCenter services. It contains the following sections:

- Section 4.1, "Preparing Your WebCenter Application to Consume Services"
- Section 4.2, "Extending Your Application with Custom Components"
- Section 4.3, "Configuring General Settings for Your Services"

Note: WebCenter services enrich existing portals and Web sites with enterprise 2.0 capabilities, including social computing services, personal productivity services, online awareness and communications, content integration, and Web analytics. For conceptual information about WebCenter services, see Section 1.1.4, "Introduction to Oracle WebCenter Services."

For important basic information about WebCenter applications, such as how to create an application, how to create customizable pages, and how to implement security, see Chapter 3, "Preparing Your Development Environment."

4.1 Preparing Your WebCenter Application to Consume Services

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

- Section 4.1.1, "How to Prepare Your Application to Consume Services"
- Section 4.1.2, "Setting Up an External Application Connection"
- Section 4.1.3, "Automated Task Flow Grants"

4.1.1 How to Prepare Your Application to Consume Services

You can configure any application to include a WebCenter service. When you create an application in JDeveloper, you can choose to base the application on a template. Although not a requirement, the WebCenter Application template makes all the appropriate WebCenter connection wizards and tag libraries readily visible and available in the New Gallery and Component palette. When you consume a WebCenter services task flow or component, the necessary libraries are automatically added to the project.
Depending upon the service you plan to consume, your application must meet certain prerequisites. For example, if the service must know the identity of users, then your application must provide some level of security with user authentication.

This section includes the following sub-sections:

- **Section 4.1.1.1, "Implementing Security for Services"**
- **Section 4.1.1.2, "Setting Up SSL-Protected Connections for Services"**

### 4.1.1.1 Implementing Security for Services

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

For details on how to implement a basic security solution for your WebCenter custom application, see **Section 3.5.1, "How to Configure ADF Security."** For details on how to implement a complete security solution for your custom application, see **Chapter 37, "Securing Your WebCenter 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 see the default task flow grants, as shown in **Figure 4–1.** You can further configure these grants in this view.
4.1.1.2 Setting Up SSL-Protected Connections for Services

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

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

To set preferences for SSL-protected WebCenter services connections:

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

The Preferences dialog displays (see Figure 4–2).
2. On the Preferences dialog, click HTTPS and Truststore Settings. The HTTPS and Truststore Settings pane displays (see Figure 4–3).

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.

### 4.1.2 Setting Up an External Application Connection

When a WebCenter 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 WebCenter services permit the use of an external application to connect with the service and define authentication for it:

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

For more information about setting up the external application for a WebCenter service, see Chapter 14, "Integrating the Documents Service," Chapter 18, "Integrating the Instant Messaging and Presence Service," Chapter 19, "Integrating the Mail Service," and Chapter 15, "Integrating the RSS Service." For more information about external applications in general, see Section 37.2, "Working with External Applications."

### 4.1.3 Automated Task Flow Grants

Table 4–1 lists the grants that are given when a task flow from the corresponding service is consumed. Common grants are granted whenever any Oracle WebCenter service is consumed.

<table>
<thead>
<tr>
<th>Service</th>
<th>Grant</th>
<th>Actions</th>
<th>Name (Target)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Common</td>
<td>TaskFlowPermission</td>
<td>View</td>
<td>/oracle/webcenter/framework/service/controller/taskflows/resourceViewer.xml#resourceViewer</td>
</tr>
<tr>
<td>Announcement</td>
<td>TaskFlowPermission</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>/oracle/webcenter/collab/announcement/view/taskflows/mini-view-definition.xml#announcement-mini-view</td>
</tr>
<tr>
<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>/oracle/webcenter/collab/announcement/view/taskflows/config-view-definition.xml#announcement-config-view</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/announcement/view/taskflows/announcement-view-definition.xml#announcement-resource-view</td>
</tr>
<tr>
<td>RegionPermission</td>
<td>View</td>
<td></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>oracle.webcenter.collab.announcement.view.pageDefs.oracle_webcenter_collab_announcement_view_jsf_pages_AnnouncementLinkResourceViewPageDef</td>
</tr>
<tr>
<td>Service</td>
<td>Grant</td>
<td>Actions</td>
<td>Name (Target)</td>
</tr>
<tr>
<td>--------------------------</td>
<td>----------------------------</td>
<td>----------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Discussions</td>
<td>TaskflowPermission</td>
<td>View</td>
<td>/oracle/webcenter/collab/forum/view/taskflows/main-taskflow.xml#forum-main</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/forum/view/taskflows/miniview-taskflow.xml#forum-miniview</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/forum/view/taskflows/popularTopic-taskflow.xml#forum-popularTopic</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/forum/view/taskflows/recentTopic-taskflow.xml#forum-recentTopic</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/forum/view/taskflows/watchedTopic-taskflow.xml#forum-watchedTopic</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/forum/view/taskflows/watchedForum-taskflow.xml#forum-watchedForum</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/forum/view/taskflows/config-taskflow.xml#forum-config-view</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/forum/view/taskflows/link-existing-taskflow.xml#forum-link-existing</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/forum/view/taskflows/link-new-taskflow.xml#forum-link-new</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/forum/view/taskflows/message-taskflow.xml#forum-message</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/forum/view/taskflows/resource-view-taskflow.xml#forum-resource-view</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/collab/forum/view/taskflows/scope-config-taskflow.xml#forum-scope-config-view</td>
</tr>
</tbody>
</table>

RegionPermission View

<table>
<thead>
<tr>
<th>Service</th>
<th>Grant</th>
<th>Actions</th>
<th>Name (Target)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>View</td>
<td>oracle.webcenter.collab.forum.view.pageDefs.oracle_webcenter_collab_forum_view_jsf_pages_ThreadExistingViewPageDef</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>oracle.webcenter.collab.forum.view.pageDefs.oracle_webcenter_collab_forum_view_jsf_pages_ThreadExistingViewPageDef</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>oracle.webcenter.collab.forum.view.pageDefs.oracle_webcenter_collab_forum_view_jsf_pages_ThreadNewViewPageDef</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>oracle.webcenter.collab.forum.view.pageDefs.oracle_webcenter_collab_forum_view_jsf_pages_ThreadExistingViewPageDef</td>
</tr>
</tbody>
</table>

Documents TaskflowPermission View

<table>
<thead>
<tr>
<th>Service</th>
<th>Grant</th>
<th>Actions</th>
<th>Name (Target)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>View</td>
<td>/oracle/webcenter/doclib/view/jsf/taskflows/mainView.xml#doclib-document-library</td>
</tr>
<tr>
<td></td>
<td></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>/oracle/webcenter/doclib/view/jsf/taskflows/recentDocuments.xml#doclib-recent-documents</td>
</tr>
</tbody>
</table>

RegionPermission View

<table>
<thead>
<tr>
<th>Service</th>
<th>Grant</th>
<th>Actions</th>
<th>Name (Target)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>View</td>
<td>oracle.webcenter.doclib.view.pageDefs.oracle_webcenter_doclib_view_jsf_pages_mainViewPageDef</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>oracle.webcenter.doclib.view.pageDefs.oracle_webcenter_doclib_view_jsf_pages_linkToExistingDocumentPageDef</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>oracle.webcenter.doclib.view.pageDefs.oracle_webcenter_doclib_view_jsf_pages_linkToExistingDocumentPageDef</td>
</tr>
</tbody>
</table>

External Application TaskflowPermission View

<table>
<thead>
<tr>
<th>Service</th>
<th>Grant</th>
<th>Actions</th>
<th>Name (Target)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></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></td>
<td>/oracle/adfinternal/extapp/view/fragments/extapp-change-password-taskflow.xml#extapp-change-password-taskflow</td>
</tr>
</tbody>
</table>

Table 4–1 (Cont.) Automated Task Flow Grants
### Table 4–1 (Cont.) Automated Task Flow Grants

<table>
<thead>
<tr>
<th>Service</th>
<th>Grant</th>
<th>Actions</th>
<th>Name (Target)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instant Messaging and Presence</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Links</td>
<td>TaskflowPermission</td>
<td>View</td>
<td>/oracle/webcenter/relationship/view/jsf/resources/links-detail.xml#links-detail</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>/oracle/webcenter/relationship/view/jsf/resources/links-detail-popup.xml#links-detail-popup</td>
</tr>
<tr>
<td></td>
<td>RelationshipPermission</td>
<td>Manage</td>
<td></td>
</tr>
<tr>
<td>Mail</td>
<td>TaskflowPermission</td>
<td>View</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>/oracle/webcenter/collab/mail/view/jsf/regions/content-view-definition.xml#mail-content-view</td>
</tr>
<tr>
<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>/oracle/webcenter/collab/mail/view/jsf/regions/mini-view-definition.xml#mail-mini-view</td>
</tr>
<tr>
<td></td>
<td>RegionPermission</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></td>
<td>oracle.webcenter.collab.mail.view.pageDefs.oracle_webcenter_collab_mail_view_jsf_pages_ContentViewPageDef</td>
</tr>
<tr>
<td>Page Editor</td>
<td>TaskflowPermission</td>
<td>View</td>
<td>/oracle/webcenter/.*</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>/oracle/adfinternal/pageeditor/.*</td>
</tr>
<tr>
<td></td>
<td>RegionPermission</td>
<td>View</td>
<td>oracle.webcenter.*</td>
</tr>
<tr>
<td>Page</td>
<td>TaskFlowPermission</td>
<td>View</td>
<td>/oracle/webcenter/page/view/jsf/fragments/page-create-page.xml#page-create-page</td>
</tr>
<tr>
<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>/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>/oracle/webcenter/page/view/jsf/fragments/golink-prop-panel-definition.xml#golink-prop-panel-definition</td>
</tr>
<tr>
<td>People</td>
<td>TaskFlowPermission</td>
<td>View</td>
<td>/oracle/webcenter/people/view/jsf/regions/people-contacts.xml#people-contacts</td>
</tr>
<tr>
<td>Recent Activities</td>
<td>TaskFlowPermission</td>
<td>View</td>
<td>/oracle/webcenter/recentactivity/controller/taskflows/recent-activities.xml#recent-activities</td>
</tr>
<tr>
<td>RSS</td>
<td>TaskFlowPermission</td>
<td>View</td>
<td>/oracle/webcenter/rssviewer/view/jsf/fragments/RSSViewerTaskFlow.xml#RSSViewerTaskFlow</td>
</tr>
</tbody>
</table>
### Table 4–1 (Cont.) Automated Task Flow Grants

<table>
<thead>
<tr>
<th>Service</th>
<th>Grant</th>
<th>Actions</th>
<th>Name (Target)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search</td>
<td>TaskFlowPermission</td>
<td>View /oracle/webcenter/search/controller/taskflows/searchResults.xml#search-view</td>
<td>/oracle/webcenter/search/controller/taskflows/localToolbarSearch.xml#search-toolbar</td>
</tr>
<tr>
<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>/oracle/webcenter/search/controller/taskflows/allSavedSearches.xml#all-saved-searches</td>
</tr>
<tr>
<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>/oracle/webcenter/search/controller/taskflows/customize.xml#search-customize</td>
</tr>
<tr>
<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>/oracle/webcenter/tagging/controller/taskflows/tag-selection.xml#tag-selection</td>
</tr>
<tr>
<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>/oracle/webcenter/tagging/controller/taskflows/tag-center-task-flow.xml#tag-center</td>
</tr>
<tr>
<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>/oracle/webcenter/tagging/controller/taskflows/tag-center-related-users.xml#tag-center-related-users</td>
</tr>
<tr>
<td>RegionPermission</td>
<td>View</td>
<td>oracle.webcenter.tagging.view.pageDefs.oracle_webcenter_tagging_view_jsf_fragments_launch_dialogPageDef</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>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>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>oracle.webcenter.tagging.view.pageDefs.oracle_webcenter_tagging_view_jsf_fragments_tag_center_related_usersPageDef</td>
</tr>
<tr>
<td>Worklist</td>
<td>TaskFlowPermission</td>
<td>View /oracle/webcenter/worklist/view/jsf/taskFlowDefs/worklist.xml#worklist</td>
<td>/oracle/ADFinternal/extapp/view/fragments/*</td>
</tr>
</tbody>
</table>

In addition to the services in Table 4–1, grants are also given when the External Application - Change Password task flow is applied when creating an external application in Oracle JDeveloper. Specifically, the TaskflowPermission is given and placed in /oracle/ADFinternal/extapp/view/fragments/*.
4.2 Extending Your Application with Custom Components

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

- Section 4.2.1, "Introducing the Resource Action Handling Framework"
- Section 4.2.2, "Registering a Resource Viewer"

4.2.1 Introducing the Resource Action Handling Framework

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

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

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

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

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

- Announcements
- Discussions
- Documents (URL rewriter points to the GET handler)
- Events
- Group Spaces (URL rewriter that maps to /webcenter/spaces/Space1Name)
- Mail
- Notes
- Page
4.2.2 Registering a Resource Viewer

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

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

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

To register a custom resource viewer:

1. In the Application Navigator, right-click the Page Flows node for the ViewController project and select New from the context menu.
2. Under Categories, select JSF, then ADF Task Flow.
3. Click OK.
4. In the Create Task Flow dialog, change File Name to resource-viewer.xml. The Task Flow ID should automatically update to resource-viewer (Figure 4-4).
Extending Your Application with Custom Components

Preparing Your Application for Oracle WebCenter Services

Figure 4-4  Create Task Flow Dialog for Resource Viewer

5. Click OK.
   Ensure that the resource-viewer.xml file is created under ..\public_html\WEB-INF.

6. From the Component Palette, ensure that ADF Task Flow is selected and drag View onto the task flow to create a page fragment.

7. Rename the page fragment to resource-viewer, and click in an open area of the canvas to accept the change.

8. Define a JSFF file for your page fragment.

9. Double-click the page fragment.

10. Click OK in the Create New JSF Page Fragment dialog.


Example 4-1 provides a very simple resource viewer. You could create a much more complex viewer.

Example 4-1  Simple Resource Viewer

```xml
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version="2.0"
   xmlns:af="http://xmlns.oracle.com/adf/faces/rich">
   <af:outputText value="Rendering #{pageFlowScope.resourceId}"
      inlineStyle="font-size:5"/>
</jsp:root>
```
12. Add a `resourceId` input parameter to the task flow, and set the value to `#{pageFlowScope.resourceId}`.

13. After the task flow is defined, you must register it in the `service-definition.xml` file. In the Application Navigator, expand Application Resources, Descriptors and ADF META-INF.

14. Double-click `service-definition.xml` to open it in the editor.

15. In the file, you see a service definition that has been commented out. Uncomment it by removing the `<!--` and `-->` tags around it.

16. Set the `id` attribute to the service identifier.

   For example, the service identifier used when defining the Tagging Button could be `mycompany.myproduct.myapp.mycomponent`.

17. Set the `taskFlowId` to 
   `/WEB-INF/resource-viewer.xml#resource-viewer`.

   This identifier is the task flow path and file name concatenated with the task flow identifier declared in the Create Task Flow dialog. You can omit the remaining tags.

18. The `<resource-view>` declaration has a class specification for the resource authorizer. This provides various ways to specify the resource view and determine which tagged objects are visible (and searchable) to users.

   **Example 4–2 Specifying a Task Flow ID in a resource-view Declaration**
   ```xml
   <resource-view taskFlowId="/somepath/somefile.xml#someId"
                  authorizerClass="some.service.id.Authorizer1"/>
   ```

   The Search, Tags, and Discussions services use this type of declaration. Instead of having a view ID (JSPX page), the service supplies a task flow to render the resource:
   - The attribute `taskFlowId` signifies that the renderer is a task flow.
   - A task flow is used to render the resource specified by the resource ID.
   - The task flow honors a `resourceId` input parameter and uses that to render the resource.

   **Example 4–3 Specifying a URL Rewriter Class Name (1)**
   ```xml
   <resource-view urlRewriterClass="oracle.webcenter.framework.internal.resource.IdentityResourceUrlRewriter"
                   authorizerClass="some.service.id.Authorizer5"/>
   ```

   or

   **Example 4–4 Specifying a URL Rewriter Class Name (2)**
   ```xml
   <resource-view urlRewriterClass="oracle.webcenter.framework.internal.resource.MessageFormatResourceUrlRewriter"
                   authorizerClass="some.service.id.Authorizer4">
     <parameters>
       <!-- This starts from the context root which would be /webcenter -->
       <parameter name="message" value="/content/conn/UCM/path/{0}"/>
     </parameters>
   ```
or

Example 4-5 Specifying a URL Rewriter Class Name (3)

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

Services, such as Oracle SES and Documents, use a URL rewriter to treat the resource ID as a URL and rewrite the URL in an appropriate fashion. A URL rewriter changes the resource ID into a URL that can be launched with a goLink. Two ResourceUrlRewriters are provided:

- oracle.webcenter.framework.internal.resource.IdentityResourceUrlRewriter is for Oracle SES. It takes the resourceId value as a URL as is.
- oracle.webcenter.framework.internal.resource.MessageFormatResourceUrlRewriter is for Documents. It takes in a message parameter (in the service-definition). The message parameter contains a {0}, which is replaced with the value of the resourceId. The urlExternal parameter applies to all URL rewriters. This parameter's value determines whether the JSPX that consumes a resource action handler link should open a new window.

The default behavior is to launch a new window dialog with that URL.

Some resources, like pages and wikis, can have no resource viewer but may allow dubbing their resource IDs as page view IDs (Example 4-6).

Example 4-6 No Resource Viewer (Resource ID as Page View ID)

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

The resource authorizer interface uses Java interfaces and classes to return useful information about service resources. For example, the oracle.webcenter.framework.resource.ResourceAuthorizer interface contains a method that returns the status (available or not available) for a unique resource ID (Example 4-7).

Example 4-7 Method for Returning Status Against a Unique Resource ID

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

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

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

    public ResourceStatus getResourceStatus();

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

Example 4–8 Possible ResourceStatus Values

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

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

        /**
         * Should be used when the user does not have view access on the resource.
         */
        USER_DOES_NOT_HAVE_ACCESS;
    }

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

Example 4–9 Resource URL Rewriter Method

    public String rewriteUrl(String id);

19. Your service should look similar to the one shown in Example 4–10.
Example 4–10 Sample Service Definition for Resource View

```
<service-definitions>
  <service-definition id="mycompany.myproduct.myapp.mycomponent"
                   version="11.1.1.0.0">
    <resource-view taskFlowId="/somepath/somefile.xml#someId"
                    authorizerClass="mycompany.myproduct.myapp.mycomponent.ResourceAuthorizer"/>
    <name>My Custom Component</name>
    <description>My Custom Component’s Description</description>
    <icon>/mycompany/myproduct/myapp/mycomponent/component.png</icon>
  </service-definition>
  ...
</service-definitions>
```

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

### 4.3 Configuring General Settings for Your Services

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

- **Time Zone**: By default, displays the time defined by the server running the WebCenter application. This is based on `java.util.Timezone`; for example, GMT+0800. It also supports the format Region/Country; for example, Europe/Amsterdam.

- **Date/Time**: By default, configured to the style `java.text.DateFormat.SHORT`. This is based on `java.text.DateFormat`; for example, `java.text.DateFormat.MEDIUM`. Supported values are SHORT, MEDIUM, LONG and FULL.

- **Language**: By default, defined by the locale of the user’s browser. This is based on `java.util.Locale`; for example, fr-CA.

- **Accessibility Settings**: A value to use with the `accessibility-mode` setting in the `trinidad-config.xml` file. Valid values are default, inaccessible, or screenReader.

- **Application Skin**: A value to use with the `skin-family` setting in the `trinidad-config.xml` file. This is based on the Trinidad skin-family attribute. Valid values are defined by skins included with the application.

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

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

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

Figure 4–5 shows the Insert EL Expression dialog with General Settings.
Configuring General Settings for Your Services

*Figure 4–5  Expression Builder for the generalSettings JSF Managed Bean*

Table 4–2 describes the preference values for the General Settings service.

<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>preferredDatePattern</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>
</tbody>
</table>
Configuring General Settings for Your Services

Preparing Your Application for Oracle WebCenter Services  4-17

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 4–11 Code for Displaying Current Date and Time in User’s Locale**

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

Or, to display a specific date and time:

**Example 4–12 Code for Displaying Specified Date and Time**

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

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 4–13 Setting the Accessibility Mode in trinidad-config.xml**

```html
<accessibility-mode>#{generalSettings.preferredAccessibilityMode}</accessibility-mode>
```

4.3.1 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 4–2.

Any WebCenter service that displays a date and time uses the settings configured in the General Settings service. For information on using the General Settings Service API to build a preferences user interface, see the Javadoc for oracle.webcenter.generalsettings. Specifically, the oracle.webcenter.generalsettings.model package contains APIs to get and set preference values for each of the settings. Use these APIs to build a custom user interface to allow users to view and set their preferred values.

For more information, see *Oracle Fusion Middleware Java API Reference for Oracle WebCenter*.

<table>
<thead>
<tr>
<th>General Settings Preference Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<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>timeZone</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 4–11 Code for Displaying Current Date and Time in User’s Locale**

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

Or, to display a specific date and time:

**Example 4–12 Code for Displaying Specified Date and Time**

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

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 4–13 Setting the Accessibility Mode in trinidad-config.xml**

```html
<accessibility-mode>#{generalSettings.preferredAccessibilityMode}</accessibility-mode>
```

4.3.1 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 4–2.

Any WebCenter service that displays a date and time uses the settings configured in the General Settings service. For information on using the General Settings Service API to build a preferences user interface, see the Javadoc for oracle.webcenter.generalsettings. Specifically, the oracle.webcenter.generalsettings.model package contains APIs to get and set preference values for each of the settings. Use these APIs to build a custom user interface to allow users to view and set their preferred values.

For more information, see *Oracle Fusion Middleware Java API Reference for Oracle WebCenter*. 
Part II

Using, Extending, and Customizing Your Application with Oracle Composer

Part II contains the following chapters:

- Section 5, "Introduction to Oracle Composer"
- Section 6, "Enabling Runtime Creation and Management of Pages"
- Chapter 7, "Enabling Runtime Editing of Pages Using Oracle Composer"
- Chapter 8, "Extending Runtime Editing Capabilities Using Oracle Composer"
- Section 9, "Performing Advanced Oracle Composer Configurations"
- Section 10, "Performing Oracle Composer-Specific MDS Configurations"
- Section 11, "Modifying Default Security Behavior of Oracle Composer Components"
- Section 12, "Configuring the Resource Catalog for Oracle Composer"
This chapter provides an overview of Oracle Composer and describes the various tasks users can perform on the application page at runtime.

Typically, the tasks for creating, deploying, and using web applications are performed by different categories of users in different environments:

- Application developers create applications in *design time*, which traditionally represents an IDE-based environment for creating or editing applications. For Oracle WebCenter, Oracle JDeveloper provides the design-time environment for creating web applications.

- Application administrators deploy these applications to managed servers.

- End users access the deployed applications at *runtime*, which represents a browser-based environment for using web applications.

Consider a dashboard-like application that displays information pertaining to the logged-in user, such as a worklist, open tasks, mail, and so on. The page developer creates the dashboard page and populates it with the required components and task flows. When this application is installed at a customer site, people at different levels in the organization may want the page modified to suit their requirement. For example:

- Department administrators may want to add a "Department News" task flow to the dashboard.

- LOB administrators may want to add an "Expense Incurred by Self/Directs" task flow to the dashboard.

- The country site administrator may want to add a task flow pertaining to the respective country’s operations or rules.

In addition, content contributors, moderators, and end users may want to add, delete, or modify components, task flows and other page artifacts based on the needs of the application or the needs of the business. 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 these simple modifications back onto the developer, Oracle Composer, in conjunction with Metadata Services (MDS), provides a runtime editing tool that enables business users to edit application pages. Changes made to the page at runtime are saved as metadata, separate from the base application definitions. This eliminates the need to revise the application and redeploy it to the production environment.

The concept of *design time at runtime* describes an environment that provides users with the option to edit application pages at runtime. Oracle Composer provides
components for controlling the application's design time at runtime behavior. This chapter describes the editing capabilities provided by Oracle Composer and the components that enable these capabilities. It contains the following subsections:

- Section 5.1, "Personalization and Customization"
- Section 5.2, "View and Edit Modes of a Page"
- Section 5.3, "Personalizing Capabilities in Page View Mode"
- Section 5.4, "Editing Capabilities in Design View in Page Edit Mode"
- Section 5.5, "Editing Capabilities in Source View in Page Edit Mode"
- Section 5.6, "Oracle Composer Components"
- Section 5.7, "Security and Oracle Composer"

5.1 Personalization and Customization

Changes made to a page at runtime are typically available only to the respective users, not everyone viewing the page. However, 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 containing a task flow to display details about the expenses incurred by the members of an LOB. When this task flow is modified, the LOB administrator would want all members to see the changes. To address such a requirement, Oracle Composer enables you to edit pages in different modes so that changes are saved as personalizations (affect only the individual user's view of the page) or customizations (affect everyone's view of the page).

To save a user's changes as personalizations or 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 customizations to a common layer that is accessible to all users. In this case, you cannot allow for changes to be saved as personalizations.

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 personalizations, and changes made by logged-in users with customization-level privileges are saved as 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 10.1, "Introduction to MDS."

5.2 View and Edit Modes of a Page

When a user opens an application page in a browser, the page opens in View mode. Additionally, Oracle Composer provides access to another mode called Edit mode to
enable users with appropriate privileges to edit application pages. These modes can be described as follows:

- **View mode**—The default view of a page when opened in a browser. In this mode, users can perform tasks such as rearranging components, collapsing and expanding components, and changing the layout. Typically, changes made to a page in View mode are available for the user alone and are saved as personalizations.

- **Edit mode**—The mode typically available to authenticated users with access privileges to edit page content. In addition to the tasks that can be performed in View mode, Oracle 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 customizations and are available for all users accessing the page.

**Note:** To ensure clarity and consistency while discussing runtime editing, this document uses the terms *personalize* and *edit* while referring to the tasks performed in View mode and Edit mode respectively.

### 5.3 Personalizing Capabilities in Page View Mode

Oracle Composer enables users to personalize pages in View mode and edit pages in Edit mode. This section provides an overview of personalization 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 Oracle Composer design-time components to a page: Panel Customizable, Show Detail Frame, Custom Actions, and Layout Customizable. For information about these components, see Section 5.6, "Oracle Composer Components." Changes made to a page in View mode are available only to the user making these changes.

This section includes the following subsections:

- **Section 5.3.1, "Rearrange Components"**
- **Section 5.3.2, "Change the Layout"**
- **Section 5.3.3, "Expand and Collapse Components"**
- **Section 5.3.4, "Customize Portlets"**

**Note:** For a detailed description of the editing tasks that you can perform on a page, see "Personalizing Your Page View" and "Working with Page Content" in the *Oracle Fusion Middleware User’s Guide for Oracle WebCenter.*

#### 5.3.1 Rearrange Components

This capability allows users to organize page components so that the content most useful to them displays at the top of the page. Users can rearrange components in two ways:

- **Dragging and dropping**

  Users can rearrange components by dragging and dropping them within the Panel Customizable component or across Panel Customizable
components on the page. As it is difficult to identify a Panel Customizable component in View mode, users can simply drag the component over the spot they want to place it. A solid box indicates a receptive drop location.

■ Using the Actions menu

The Move actions on a Show Detail Frame or portlet component enable users to move these components within a parent Panel Customizable component. If a Panel Customizable has multiple child components, then a child Show Detail Frame or portlet can be moved to the left and right or above and below, relative to the position of other child components. Figure 5–1 shows a sample Show Detail Frame component with Move Up and Move Down actions. In this example, selecting Move Down moves the Latest News component immediately below the Press Release component.

**Figure 5–1 Actions Menu on a Show Detail Frame**

5.3.2 Change the Layout

This capability allows users to switch between a set of predefined layouts provided by Oracle 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 Oracle Composer library enables the display of a Change Layout icon on the page, as shown in Figure 5–2.
5.3.3 Expand and Collapse Components

By enabling expand and collapse components on a page, you can include more detailed information on a page without cluttering its visual appearance. Users can collapse and expand the components as desired.

A Collapse icon on the header of a Show Detail Frame or portlet enables users to collapse such a component and display only its header. Figure 5–4 shows the Collapse icons on three Show Detail Frame components.
Personalizing Capabilities in Page View Mode

5.3.4 Customize Portlets

If the application contains portlets, then users can make them display the information that they want to see, hide them, or remove them from the page. Figure 5–6 shows the actions users can perform on a Web Clipping portlet. The Customize and Personalize options are available on portlets by default.

Figure 5–6 Actions on a Web Clipping Portlet
5.4 Editing Capabilities in Design View in Page Edit Mode

Oracle Composer's Edit mode provides a wide range of page customization capabilities. In Edit mode users can add content, edit page and component properties, delete components, rearrange components, change the page layout, and so on.

You can provide the capabilities described in this section to your application by adding the following Oracle 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 5.6, "Oracle Composer Components."

The Change Mode Link or Change Mode Button switches the page to Edit mode. In Edit mode, users can add content to the page, edit component properties, edit page properties, and so on. The Page Customizable component on a page enables these editing capabilities.

Figure 5–7 shows a Change Mode Link on an application page. When users click the Change Mode Link or Change Mode Button, the page opens in Oracle Composer.

Figure 5–7  Edit Link While in View Mode

The default Oracle Composer toolbar displays the name of the page being edited, a status indicator, View menu, Page Properties button, Reset Page button, and a Close icon. The View menu (Figure 5–8) provides two viewing options—Design and Source. Users can perform various editing tasks in these views.

Figure 5–8  View Menu in Oracle Composer

Design view (shown in Figure 5–9) is the default page view that provides a WYSIWYG rendering of the page and its content, where Edit and Delete controls are directly selectable on each component. In Design view users can also perform such tasks as adding content, editing page and component properties, changing page layout, and deleting components.
This section describes the editing capabilities that Oracle Composer provides in the Design view in page Edit mode. It contains the following topics:

- Section 5.4.1, "Add Content"
- Section 5.4.2, "Add Box Components Adjacent to Existing Ones"
- Section 5.4.3, "Rearrange Page Content"
- Section 5.4.4, "Edit Component Properties and Parameters"
- Section 5.4.5, "Reset and Override Component Properties in the Current Layer"
- Section 5.4.6, "Edit Resource Strings"
- Section 5.4.7, "Show or Hide Components"
- Section 5.4.8, "Delete Components"
- Section 5.4.9, "Change the Layout"
- Section 5.4.10, "Edit Page Properties"
- Section 5.4.11, "Wire Components to Page Parameters"
- Section 5.4.12, "Reset Page"
- Section 5.4.13, "Reset Customizations on Detecting Errors"
- Section 5.4.15, "Manage 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 5.7, "Security and Oracle Composer."

Oracle Composer provides most of these capabilities in the Source view as well. The individual sections mention whether a capability is available in both modes or only in Design mode.
For a detailed description of the runtime editing tasks that users can perform on a page, see "Personalizing Your Page View" and "Working with Page Content" in the Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

5.4.1 Add Content

Users may want to add custom content to the page, such as a portlet displaying news, for example, or stock updates. Oracle Composer enables users to add content within any container in page Design view and Source view. The Oracle Composer Catalog, also known as Oracle Business Dictionary, contains resources that users can add to a page. These include documents, Oracle ADF components, portlets, and task flows. The Catalog can be invoked in the following ways:

- In Design view, by clicking the Add Content button on any container component on the page.
- In Source view, by selecting a container component and clicking the Add Content button on the Source view toolbar.

Notes:

- If you restrict customization on a container component, then users cannot add components inside it at runtime. See Section 11.1, "Applying Component-Level Restrictions by Defining Customization Policies" for more information.
- Items such as task flows do not appear in the Catalog by default. Such items must be configured in the catalog definition file. For more information, see Section 12.2.1.3, "Enabling Task Flows in the Resource Catalog."

Figure 5–10 shows the default Catalog available in a custom WebCenter application.

Figure 5–10  Oracle Composer Catalog

The Catalog contains folders and components. An Open link next to an item in the Catalog indicates that the item is a folder. Users can drill down into the folder by clicking this link. An Add link next to an item indicates that the item can be added to the page. An Add link can appear next to a component or a folder. The component is added as the first child in the container on which the Add Content link was clicked.
If you configure drop handlers for adding resources to the page, the Add link shows a context menu with different options for adding the component to the page (Figure 5–11). For more information, see Section 8.6, "Configuring Drop Handlers in the Oracle Composer Catalog."

Figure 5–11  Context Menu with Drop Handlers for Adding a Component

Depending on its configuration, the Catalog exposes all or a subset of the following components to users:

- **ADF Faces Components**
  
  The ADF Faces Components folder provides Box, Movable Box, and Image components that are analogous to the JDeveloper design time components Panel.
Customizable, Show Detail Frame, and Image Link respectively. In JDeveloper, these components are available in the Oracle 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, Text, and Web Page components that are analogous to the JDeveloper design time components Output Text, Go Link, Rich Text Editor, and Inline Frame respectively. In JDeveloper, these components are available in the ADF Faces tag library.

The **ADF Faces Components** folder in the Catalog dialog contains these components, as shown in Figure 5–12.

**Figure 5–12  ADF Faces Components You Can Add to the Page**

- **Task Flows**
Users can add custom task flows and out-of-the-box Oracle WebCenter Services task flows to the page only if you have configured them in the catalog definition file. See Section 12.2.1.3, "Enabling Task Flows in the Resource Catalog" for details.

A task flow added to a page is automatically enclosed in a Movable Box component. As a result, the task flow displays a header with options to move or delete the component.

---

**Note:** The Movable Box component surrounding a task flow is displayed only if the task flow is displayed on the page. That is, if you do not grant even View permission on a task flow, the Movable Box surrounding it is also not displayed on the page.

---

**Portlets**

The Portlets folder lists all the registered producers. Users can navigate to the required portlet and add it to the page. The Portlets folder exposes portlets from any Java-PDK or WSRP producer that was registered in Oracle JDeveloper. For information about registering portlet producers, see Chapter 34, "Consuming Portlets."

**Documents**

The Documents folder lists documents and folders from the Documents service that users can add to the page.

To display documents, ensure that you have performed the steps in Section 12.2.1.4, "Enabling Documents in the Resource Catalog."


### 5.4.2 Add Box Components Adjacent to Existing Ones

A toolbar on the Box component in Design view provides options to add another Box component before or after the component. This capability enables users to modify the layout of components on the page.

When a user clicks an Add icon on the Box component, Oracle 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.
Users can delete Box components that they added using the Add icons, in both Design and Source view.

\[\textbf{Note:}\] If the Panel Group Layout has only two Box components and you delete a Box component, 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

**5.4.2.1 Add Box Above**

Clicking the Add Box Above icon (Figure 5–13) adds a Box component above the selected Box component. The layout attribute on the Panel Group Layout is set to vertical.

*Figure 5–13  Add Box Above Icon on a Box Component*

Figure 5–14 shows how the Box component appears in Source view before adding another Box component adjacent to it.

*Figure 5–14  Box Component Before Adding Another One Adjacent to It*
Figure 5–15 shows the two Box components vertically aligned in the component navigator.

**Figure 5–15  Box Component Added Above**

- `af:panelStretchLayout
- `af:panelGroupLayout-scroll
- `ps:layoutCustomizable
- `af:panelGroupLayout-vertical
  - `Box
  - `Box
  - `Box

Users can click the **Add Box Above** icon repeatedly to add many contiguous Box components vertically within the same parent Panel Group Layout.

**5.4.2.2 Add Box Below**

Clicking the **Add Box Below** icon (Figure 5–16) adds a Box component below the selected Box component. The layout attribute on the Panel Group Layout is set to vertical.

**Figure 5–16  Add Box Below Icon on a Box Component**

![Add Box Below Icon on a Box Component](image)

Figure 5–17 shows the two Box components vertically aligned in the component navigator.

**Figure 5–17  Box Component Added Below**

- `af:panelStretchLayout
- `af:panelGroupLayout-scroll
- `ps:layoutCustomizable
- `af:panelGroupLayout-vertical
  - `Box
  - `Box
  - `Box

Users can click the **Add Box Below** icon repeatedly to add many contiguous Box components vertically within the same parent Panel Group Layout.
5.4.2.3 Add Box Left
Clicking the Add Box Left icon (Figure 5–18) 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 5–18 Add Box Left Icon on a Box Component

Figure 5–19 shows the two Box components horizontally aligned in the component navigator.

Users can click the Add Box Left icon repeatedly to add many contiguous Box components horizontally within the same parent Panel Group Layout.

5.4.2.4 Add Box Right
Clicking the Add Box Right icon (Figure 5–20) adds another Box component to the right of the selected Box component. The layout attribute on the Panel Group Layout is set to horizontal.

Figure 5–20 Add Box Right Icon on a Box Component

Figure 5–21 shows the two Box components horizontally aligned in the component navigator.
Figure 5–21  Box Component Added to the Right

Users can click the Add Box Right icon repeatedly to add many contiguous Box components horizontally within the same parent Panel Group Layout.

5.4.3 Rearrange Page Content

This capability allows users to organize page components based on where they are easiest to use; for example, users can move content most useful to them to the top of the page. In Design view, users can rearrange content on the page in the following ways:

- Drag and drop a component within the same Panel Customizable component or across Panel Customizable (or Box) components on the page.
- Use the Move actions on the Actions menu of a Show Detail Frame or portlet to move it within the parent Panel Customizable component. Depending on the number of child components in the Panel Customizable and how these components are oriented, a component can be moved to the left and right, or up and down.
- To resequence components within a container, in the Component Properties dialog for the container, click the Child Components tab and use the up and down arrows shown against each child component (Figure 5–22).

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.
5.4.4 Edit Component Properties and Parameters

The Component Properties dialog enables users to edit component properties and the parameters associated with components, such as portlets and task flows. The Component Properties dialog is accessible in both Design view and Source view.

Note: A component cannot be edited if:

- The id attribute was not set for the component at design time
- The component or any of its attributes have been restricted
  For information about restricting customization, see Chapter 11, "Modifying Default Security Behavior of Oracle Composer Components."
- It is an ADF Faces component and Source view has been disabled for the application
  For information about disabling Source view, see Section 9.8, "Disabling Source View for the Application."

In Design view, clicking the Edit icon on a component displays the Component Properties dialog, shown in Figure 5–23.
The Display Options tab enables users to edit visual properties, such as background and title.

The Parameters tab (Figure 5–24) enables users to edit parameters for components such as portlets and task flows.

**Note:** While editing task flows, users must define values for all required parameters.
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 5–25 shows the two different icons used to indicate that the property was updated.

Note: Visual indication of property customization is relevant only if you have configured multiple customization layers in your application. For information, see Section 10.3, "Adding Customization Layers to View and Edit Modes: Example."
The Component Properties dialog also enables users to reset visual properties in the current context. For more information, see Section 5.4.5, "Reset and Override Component Properties in the Current Layer."

### 5.4.5 Reset and Override Component Properties in the Current Layer

Users can reset and override 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.

---

**Note:** The reset and override capabilities in the Component Properties dialog are relevant only if you have configured multiple customization layers in your application. For information, see Section 10.3, "Adding Customization Layers to View and Edit Modes: Example."

---

An **Override** option on the context menu for a property (Figure 5–26) 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 customizations applied to visual properties, which are rendered on the Display Options tab.

### 5.4.6 Edit Resource Strings

Component properties edited in Oracle 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 Oracle 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, Oracle 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 Oracle 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) WebCenter applications, see Section 9.11, "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 5–27**. Users can select this option to edit resource strings for properties.
The Select Text Resource dialog (Figure 5–28) provides options to search existing resources, edit or delete resource keys, and add new resource key/value pairs.

Figure 5–28  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 5–29), 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 Value/Key**: Create key and value pairs in the application override bundle.

  **Figure 5–29 Icons for Creating and Editing Resource String Keys**

Users can edit only strings that were created in the same MDS layer. Other strings appear as read-only, as shown in Figure 5–30.

**Figure 5–30 Read-Only Resource Strings in the Select Text Resource Dialog**

- **Display Value field**: For specifying a resource string to translate for display in the UI or modifying an existing string.

- **Description field**: For specifying a translatable description of the resource or modifying an existing description.

- **Search field**: For searching existing resources. Users can search for strings that were created in the same layer, in a different MDS layer, or resource bundles defined and used in a JSPX file at design time.
Editing Capabilities in Design View in Page Edit Mode

- Search results table: For displaying search results, editing strings, and specifying values for new strings.

Users can edit strings that were created in the same layer. Other strings created at design time or in a different MDS layer can only be used for the property, but not edited.

A link or a status indicator toggle displays in the last column and enables users to promote a string to be the active, to-be-translated resource (Figure 5–31) or identifies a string as the active resource (Figure 5–32).

Figure 5–31  Use Link on a Resource String

![Select Text Resource: region (Text)](image)
Note: When searching for resource strings created at design time, Oracle 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="viewcontrollerBundle"
       value="#{compBundle['test.resource.ComposerBundle']}/>"
```

- Create button: For creating a new resource string, on clicking the Create button, the search results table provides a new row with input fields for creating a new resource. Users can double click inside the row and create a new string by specifying values for the Key, Display Value, and Description, as shown in Figure 5–33, 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 this string for use as the current property value.

- **Delete** button: For deleting a selected resource from the search results table.
- **Refresh**: For refreshing the search results table.

**Notes**

- Oracle 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.

### 5.4.7 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:

---

**Figure 5–33  Fields for Creating a New Resource String in the Search Results Table**

<table>
<thead>
<tr>
<th>Select Text Resource: region (Text)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
</tr>
<tr>
<td>Display Value</td>
</tr>
<tr>
<td>Description</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Create</th>
<th>Delete</th>
<th>Search</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
<td>Display Value</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>---------------</td>
<td>---------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---
Click the **Edit** icon on the component. In the Component Properties dialog, clear the Show Component check box (Figure 5–34).

**Figure 5–34  Show Component Option in the Component Properties Dialog**

![Component Properties Dialog](image)

Checking the Show Component check box and clicking Apply or OK in the dialog renders the component on the page again.

Click the **Edit** icon on the component’s container. On the Child Components tab in the Component Properties dialog, click the Hide Component link shown against the component name.
5.4.8 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 11.1.1, "How to Define Type-Level Customization Policies."

5.4.9 Change the Layout

In Design view, the Change Layout icon enables users to select a layout from a set of eight predefined layouts. The default page layout is threeColumn.

In Source view, users can select the Layout Customizable component and view its properties. The layout options are displayed in the Component Properties dialog. Users can select any predefined layout and click Apply or OK.

**Note:** Predefined layout options are available to users only if you have added a Layout Customizable component to the page at design time. For more information, see Section 5.6.5, "Layout Customizable."
5.4.10 Edit Page Properties

Users can access the Page Properties dialog from both Design view and Source view. In this dialog, users can create or edit a page’s parameters and display properties. For pages created at runtime in a secured application, a Security tab enables users with Grant privilege to edit page access privileges for other users.

Figure 5–36 shows the Security tab in the Page Properties dialog.

Figure 5–36 Security Tab on the Page Properties Dialog

On the Parameters tab (Figure 5–37), 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.
5.4.11 Wire Components to Page Parameters

Oracle Composer enables users to link portlets and task flows with page parameters so that these components can read the page parameters and change their behavior accordingly.

Users can wire portlet and task flow parameters to page parameters using the Component Properties dialog. For more information, see "Wiring Pages, Task Flows, Portlets, and UI Components Together" in Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

In addition to wiring components and page parameters, Oracle Composer also enables users to pass values for display options and parameters through page URLs. For more information, see "Passing Parameter Values Through the Page URL" in Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

5.4.12 Reset Page

The Reset Page button is available on the page in both Design view and Source view. This button invokes the Reset Page dialog that provides options to remove edits made to a page and reset it to a previously saved version or its original, out-of-the-box state.

---

Note: The Reset Page button is a default add-on rendered on the Oracle Composer toolbar. You can disable this button if you do not want users to reset the page. For information, see Section 8.2.5, "How to Selectively Display Add-Ons."
Roll Back to a Specific Label

If your application is configured to use a database store, a new version of the page is generated each time a user saves customizations. If a sandbox is configured for your application, by creating a label with the SAVE_LABEL_% convention you can ensure that Oracle Composer resets the page as follows:

- If there is only one label name following the SAVE_LABEL_% convention, then Oracle Composer resets the page to its state in that label.
- If there is no label name following the SAVE_LABEL_% convention, then Oracle Composer resets the page to its original, out-of-the-box state.
- If there are multiple labels following the SAVE_LABEL_% convention, then Oracle 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 10.2.1, "How to Enable Oracle 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 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 5–1 shows the sample code to create a label with a SAVE_LABEL_ prefix.

Example 5–1  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);
```
5.4.13 Reset Customizations on Detecting Errors

Oracle 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 5–38, provides the option to roll back only the last change or all changes that users made in the current context.

![Figure 5–38 Error Dialog Prompting Users to Discard Changes](image)

- If the change is not yet saved to MDS, an Error dialog, shown in Figure 5–39, simply informs users that an error occurred and the page will be reloaded.

![Figure 5–39 Error Dialog Informing Users About Page Reload](image)

Errors occurring while performing any of 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.
- Dragging and dropping Show Detail Frame components on the page.

5.4.14 Create Labels On Saving Customizations

A Save and Label button on the Oracle Composer toolbar, shown in Figure 5–40, enables users to save their customizations and create a label with those customizations. 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."
5.4.15 Manage Customizations

The Customization Manager is an Oracle Composer add-on panel that enables users to manage customization metadata for a page and for page fragments or task flows on the page. The Customization Manager dialog displays a list of all task flows, pages, and page fragments on the page, and provides details about the layers in which these objects are customized. Users can download customization metadata for a selected page, fragment, or task flow, 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 8.2.4, "How to Display the Customization Manager Add-On." If you configure your application to display the Customization Manager add-on, the Oracle Composer toolbar displays a Customization Manager button, as shown in Figure 5–41.

Clicking the Customization Manager button invokes the Customization Manager dialog, shown in Figure 5–42.
The Customization Manager displays Promote, Download, Upload, and Delete links against each task flow, page, or fragment in each layer. Users can specify a JSPX page name in the search field to view customization details about objects on that page and to manage those customizations.

Users can manage customizations in the current layer and other layers configured in the application. The Current Context column displays options to manage customizations in the current layer. This column is useful for all users who want to manage customizations they have made. The All Layers column displays options to manage customizations in all layers in the application. This column is useful for administrators who want to manage customizations made in all layers.
Notes:

- If you have configured a sandbox for your application, all Customization Manager operations are performed in the sandbox; for example, when a user uploads a customization document, it is stored in the sandbox until the user saves or discards changes made during that session.

- You can customize the Customization Manager to only display options of your choice. For example, you can show or hide the Download, Delete, or Upload links. Further, you can choose to show the various options for all layers or the current layer 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 8.8, "Enabling Parameter Support on the Customization Manager Task Flow."

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 or fragment, from the selected layer. This is useful if a user wants to edit customization metadata for a component.

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 All Customizations link at the bottom of the page enables users to download a ZIP file containing customization documents for all customized objects on the page.

Upload Customization Metadata
An Upload link enables users to upload a customization document for the selected component in a specific layer. Figure 5–43 shows the Upload Customization dialog that lets users select the file they want uploaded.
The component 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 object from a specific layer.

**Promote Customization Metadata**
While customizing a page, users can save 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 customizations on the selected object 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 Oracle Composer, see Section 5.4.14, "Create Labels On Saving Customizations."

The Promote link invokes the Promote dialog, which lists all labels created in that layer, as shown in **Figure 5–44**.

**Figure 5–44  Select a Label Dialog**

Users can select a label and promote that label to the sandbox for the selected object. This means that customizations for the selected object 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.
5.5 Editing Capabilities in Source View in Page Edit Mode

Source view (Figure 5–45) provides a WYSIWYG and a hierarchical rendering of page components in a component navigator. Add Content, Edit, Delete, Cut, and Paste controls are available on the Source view toolbar provide specific operations in Oracle Composer. In Source view, users can access and modify properties of components that are not otherwise selectable in Design view. For example, many ADF Faces components can be edited only in Source view. Users can also edit components within a task flow.

Figure 5–45  Source View of a Page

By default, the component navigator displays at the top of the page within Oracle Composer. Users can choose to display it at the bottom, left, or right using the Source Position option on the View menu (Figure 5–46). 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 between the component navigator and the page displays a bread crumb trail for the selected component (Figure 5–47). 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 5–47  Bread Crumbs for a Component Selected in Page Source View

Task flows that can be edited have Edit Task Flow links next to them. On clicking an Edit Task Flow link, the component navigator displays the hierarchy of components within the task flow’s page or fragment. Users can edit this page or fragment and click Close to zoom out of the task flow. For more information, see Section 5.5.2, “Edit Content Inside a Task Flow.” Similarly, declarative components display Open links that enable users to edit components inside the selected declarative component’s facets.

Similar to Design view, in Source view also users can perform such tasks as adding components, editing page and component properties, changing page layout, and deleting components. Users can select a component on the page or in the component navigator and click the buttons on the Source view toolbar to perform these tasks. To edit or delete a component, users can also select options on the context menu for the component. In addition to these tasks, users can perform the following tasks in page Source view:

- Section 5.5.1, "Rearrange Page Content Using Cut and Paste Options"
- Section 5.5.2, "Edit Content Inside a Task Flow"
- Section 5.5.3, "Edit Content Inside a Declarative Component"
- Section 5.5.4, "Show or Hide Components"

Notes:

- The Save button is provided on the Oracle Composer toolbar (both in Design view and Source view) only if the application is configured to use a sandbox. For more information about sandboxes, see Section 10.2, “Using Oracle 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 5.7, "Security and Oracle Composer."
5.5.1 Rearrange Page Content Using Cut and Paste Options

Users can rearrange components on a page using the Cut and Paste options on the Source view toolbar (Figure 5–48). A Paste menu enables users to paste a component into, before, or after any other component in the component navigator.

**Figure 5–48  Cut and Paste Options on the Source View Toolbar**

The Cut and Paste buttons are grayed out if the user selects a component that does not support a cut or paste operation. Cut and paste operations are not supported under the following conditions:

- The target component is not on the same page as the cut component.
- The target component is nested inside the cut component.
- The target component is located before or after a faceted component, that is, a component inside a facet.
- The parent of the target component is not customizable.
- The parent of the cut component is not customizable.

**Notes:** A component on which customization is restricted can be rearranged inside its parent container if the parent is customizable.

**Example**

If your customizable page contains an ADF Faces Table component, then in Oracle Composer's Source view, users can resequence or move columns within or across tables and column groups. Consider a sample application created for selling computer accessories. The application home page has two tables containing details about the software and hardware offerings. The application home page has two tables containing details about the software and hardware offerings. The Hardware Options table contains four columns. The Hard disk column in turn contains the Magnetic disks and Solid State columns inside a column group, as shown in Figure 5–49.
Users can switch to Source view and resequence the Magnetic disks and Solid State columns within the Hard disk column or move them out of the column group and paste them next to any other column (the Wireless column, for example). Figure 5–50 shows the table structure in the component navigator.

Users can select a column and click Cut, then select another column and click Paste Before or Paste After.

5.5.2 Edit Content Inside a Task Flow

Oracle ADF Task flows consist of one or more views and each view in a task flow is associated with a page or page fragment. When you add a task flow to your page, the content on the fragment associated with the current view is displayed on the page. Oracle Composer enables users to edit components on the page or fragment used for the current view of a task flow. Since changes are made to the task flow’s page or fragment, they are reflected in all places where the task flow’s page or fragment is used.

The Source view provides an option to open a task flow and display only components on its page or fragment in the component navigator. Users can edit the page or fragment and close the task flow to navigate back to the page containing the task flow.

Users with Customize permission on a task flow can edit its page or fragment. By default, users can edit content inside any task flow on the page. However, if the task
flow is included inside a Show Detail Frame component, you can disable runtime editing of the task flow by setting the selectChild attribute on the enclosing Show Detail Frame component to false. For information about this attribute, see Section B.1.5, "Show Detail Frame Component." You cannot turn off the editing capability for task flows if they are not included inside Show Detail Frame components.

Each task flow in the component navigator displays an Edit Task Flow link next to it, as shown in Figure 5–51.

Figure 5–51  Edit Task Flow Link on a Task Flow

Clicking the Edit Task Flow link results in the Source view zooming in to display the task flow’s page and its components, as shown in Figure 5–52.

Figure 5–52  Components Within a Task Flow

Users can select components on the task flow’s page and edit them in Oracle Composer just like editing any other page components. However, Oracle 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 Oracle Composer that alerts each user to the others. However, this warning is displayed only if a sandbox is configured in the application. For information about how changes are saved in such cases, see "What Happens During Concurrent Edits".
Clicking the Up arrow or Close link next to an open task flow (Figure 5–53) results in the component navigator displaying the page containing the task flow.

**Figure 5–53  Options to Zoom Out of a Task Flow**

![Figure 5–53](image)

5.5.2.1 Reset Task Flow

The Reset Task Flow button on the Source view toolbar (Figure 5–54) invokes the Reset Task Flow dialog that provides options to remove 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 5.4.12, “Reset Page.”

**Figure 5–54  Reset Task Flow Button**

![Figure 5–54](image)

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 customizations. If a sandbox is configured for your application, by creating a label with the SAVED_LABEL_% convention you can ensure that Oracle Composer resets the task flow as follows:

- If there is only one label name following the SAVED_LABEL_% convention, then Oracle Composer resets the task flow to its state in that label.
- If there is no label name following the SAVED_LABEL_% convention, then Oracle Composer resets the task flow to its original, out-of-the-box state.
- If there are multiple labels following the SAVED_LABEL_% convention, then Oracle 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 10.2.1, "How to Enable Oracle 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 5–2 shows the sample code to create a label with a `SAVE_LABEL_` prefix.

**Example 5–2 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);
```

### 5.5.3 Edit Content Inside a Declarative Component

If the application page contains declarative components, users can edit these components in Oracle Composer. Editing declarative components is similar to editing task flows. For more information, see Section 5.5.2, "Edit Content Inside a Task Flow."

Figure 5–55 and Figure 5–56 show the Edit Component and Close options available on declarative components.
5.5.4 Show or Hide Components

To hide a component, right-click the component in the component navigator or on the page and choose Hide Component on the context menu (Figure 5–57).
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.

### 5.6 Oracle Composer Components

To make any JSPX document (*.jspx) editable at runtime, you must add Oracle Composer components to your page in Oracle JDeveloper at application design time.

**Note:** Oracle Composer works only with JSPX pages and ADF Faces. You cannot add these components to JSP pages. For information about adding Oracle Composer components to your page, see Section 7.1, "Designing Editable Pages Using Oracle Composer Components."

The Oracle Composer tag library (Figure 5-58) available from the Component Palette provides components that you can add to make a page editable.
This section provides an overview of the Oracle Composer components that are used to enable page editing. It contains the following subsections:

- Section 5.6.1, "Page Customizable"
- Section 5.6.2, "Change Mode Link and Change Mode Button"
- Section 5.6.3, "Panel Customizable"
- Section 5.6.4, "Show Detail Frame"
- Section 5.6.5, "Layout Customizable"

For more information about these components and other Oracle Composer components such as Custom Action, see Appendix B, "Oracle Composer Component Properties and Files."

### 5.6.1 Page Customizable

The Page Customizable component defines the editable area of a page. Within this area, you can edit component properties, add content to the page, arrange content, and so on.

Adding a Page Customizable component enables the runtime inclusion of Oracle Composer on the page. Users can edit pages in Oracle Composer using page-related controls available across the top of the page, Add Content buttons on components, and Edit icons on each editable page component, as shown in Figure 5–59.
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 7.1.2, "How to Enable Runtime Customization Using a Page Customizable."

5.6.2 Change Mode Link and Change Mode Button

The Change Mode Link or Change Mode Button component provides an easy way to switch from View mode of the page to Edit mode. Figure 5–60 shows a Change Mode Link component on the page.

For more information about using Change Mode Link or Change Mode Button, see Section 7.1.3, "How to Enable Switching Between Page Modes Using a Change Mode Link or Change Mode Button."

5.6.3 Panel Customizable

A Panel Customizable defines an area of the page onto which users can add components at runtime. Users can move or minimize Show Detail Frame components and portlets that are added as child components of a Panel Customizable. Users can also drag and drop these components into another Panel Customizable component on the page.

In Edit mode, the Panel Customizable component is rendered as a box with dotted lines. In fact, a Panel Customizable is referred to as a Box in the runtime Catalog. An Add Content button appears on each Panel Customizable component on the page, as shown in Figure 5–61. You can use this button to open the Resource Catalog Viewer and add components within the Panel Customizable.
5.6.4 Show Detail Frame

A Show Detail Frame component renders a border or chrome around its child component along with a header that contains icons to enable users to perform some operations. The actions available on this menu enable users to move the component, along with its content, to new positions on the page (Figure 5–62). 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 component must be included inside a Panel Customizable component for it to be movable.

Figure 5–62 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 9.1, "Enabling Custom Actions on Show Detail Frame Components by Using Facets."

For information about adding this component to a page, see Section 7.1.6, "How to Enable Component Customization Using Show Detail Frame Components."

5.6.5 Layout Customizable

The Layout Customizable component is a container that enables end users to lay out its child components in several predefined ways (for example, two column, three column, and so on). You can design your page in such a way that all components on the page are enclosed in a Layout Customizable component. In such a case, the layout is applied to the entire page.

Access predefined layouts using the layout changer. By default, the layout changer displays as a small green icon both in page View mode and page Edit mode. By using
the Layout Customizable component attributes you can choose to display the layout changer as an icon, text, or icon and text. In addition, you can decide whether to show or hide the layout changer in View mode. Figure 5–63 shows a Layout Customizable component and the predefined layouts that display when users click the layout changer.

Figure 5–63  Layout Customizable Component

For more information, see Section 7.1.5, "How to Enable Layout Customization for a Page Using a Layout Customizable."

5.7 Security and Oracle 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 customizations on a component. Oracle Composer determines if a component can be customized by honouring these rules. For example, before users can customize components, the components’ sources of security restrictions are queried to determine if customization is allowed. If a component does not permit customization, Oracle Composer considers it restricted; that is, Oracle Composer treats the component as if it were explicitly secured. Table 5–1 describes Oracle Composer behavior to reflect customization restrictions.

Table 5–1  Customization Restrictions on Oracle Composer Components

<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>Oracle Composer allows editing of components without restrictions.</td>
</tr>
<tr>
<td>The page is restricted</td>
<td>Oracle Composer displays but none of the options, for example Page Properties button and Edit icon, are accessible.</td>
</tr>
<tr>
<td>Some operations are restricted</td>
<td>Oracle 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>
</tbody>
</table>
Depending on the privileges granted while implementing security, users can perform different personalization and editing tasks when they log in to the application.

Note: In a secured application, it is recommended that you check all users’ privileges and enable the Edit link or button on the page only for privileged users. Unauthenticated users who stumble into page Edit mode can change component properties.

You can enable the Edit link or button for selected users by specifying an EL value for the rendered attribute on the Change Mode Link or Change Mode Button component. For more information, see Section B.1.2, "Change Mode Link and Change Mode Button."

This section describes the default security behavior of Oracle Composer components. It contains the following subsections:

- Section 5.7.1, "Page Security"
- Section 5.7.2, "Task Flow Security"
- Section 5.7.3, "MDS Customization Restrictions"
- Section 5.7.4, "Component Action-Level Security"

For information about overriding default security behavior, see Chapter 11, "Modifying Default Security Behavior of Oracle Composer Components."

### 5.7.1 Page Security

The application's jazn-data.xml file is the repository for page-level security information. Oracle Composer references this file and enables editing capabilities based on a user’s privileges:

- A user with the Personalize privilege on a page can perform only the personalizations described in Section 5.3, "Personalizing 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. Oracle Composer and WebCenter Customizable Components do not differentiate between Edit and Customize privileges.

**Notes:** Oracle Composer and WebCenter Customizable Components support cascading of page privileges with Grant being a super set of all privileges. A user with Grant privilege on a page is considered to have Edit, Personalize, and View privileges. A user with Personalize privilege is considered to additionally have the View privilege.
Table 5–2 explains Oracle Composer behavior based on page-level privileges. Only those page privileges that are relevant to Oracle Composer and WebCenter Customizable Components privileges 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. The table also does not list the View privilege because users with the View privilege cannot personalize or edit a page.

### Table 5–2 Mapping of Page Privileges to Oracle Composer Behavior

<table>
<thead>
<tr>
<th>Privilege</th>
<th>Oracle Composer Behavior</th>
</tr>
</thead>
</table>
| Edit or Customize | - Users can switch to Edit mode, where Oracle Composer is invoked, and edit the page.  
- Users with either the Edit or Customize privilege can perform all runtime editing tasks.  
With the Edit or Customize privilege, users can:  
  - Add content  
  - Edit component properties  
  - Rearrange content  
  - Delete components in Oracle Composer  
  - Personalize or customize a portlet  
  - Move a portlet or task flow in View mode. This is persisted as a personalization.  
  - Expand and collapse a task flow or portlet in View mode. This is persisted as a personalization.  
  - Resize a column of a table in a task flow. This is persisted as a personalization.  
  - Reset the page to its original state  
  - Delete components in View mode  
If users without Edit or Customize permission try to edit a page, a message appears stating that they do not have permission to do so. |
| Personalize   | - In View mode users can:  
  - Rearrange content  
  - Personalize a portlet  
  - Move a portlet or task flow in View mode. This is persisted as a personalization.  
  - Expand and collapse a task flow or portlet in View mode. This is persisted as a personalization.  
  - Resize a column of a table in a task flow. This is persisted as a personalization.  
  - Delete components in View mode  
**Note:** Having Personalize permission does not enable users to perform portlet customizations.  
If users without Personalize, Edit, or Customize permission try to edit a page, a message appears stating that they do not have permission to do so. |

### 5.7.2 Task Flow Security

When you add a task flow to a customizable page, Oracle Composer provides options for editing the task flow and components on the task flow’s page. In a secured
application, Oracle Composer provides editing capabilities based on the privileges provisioned on the customizable page and the task flow.

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 5–64.

**Figure 5–64 Structure of a Page Containing Nested Task Flows**

<table>
<thead>
<tr>
<th>Page Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Document.jspx</code></td>
</tr>
<tr>
<td>Component1</td>
</tr>
<tr>
<td>Component2</td>
</tr>
<tr>
<td>Region3 [region3.jsff]</td>
</tr>
<tr>
<td>ComponentA</td>
</tr>
<tr>
<td>ComponentB</td>
</tr>
<tr>
<td>RegionC [regionC.jsff]</td>
</tr>
<tr>
<td>ComponentX</td>
</tr>
<tr>
<td>ComponentY</td>
</tr>
</tbody>
</table>

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 7.3.1.1, "Considerations for Adding Task Flows" and Section 11.6, "Applying Task Flow Restrictions."

For more information about task flows and their security behavior, see Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework.

5.7.3 MDS Customization Restrictions

WebCenter applications have a default MDS configuration that restricts 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 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 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 Oracle Composer, and restricted attributes are not displayed in the Component Properties dialog for the selected component. For information, see Section 11.1, "Applying Component-Level Restrictions by Defining Customization Policies."

5.7.4 Component Action-Level Security

Panel Customizable and Show Detail Frame components enable the placement of restrictions on individual supported actions. For example, one can specify a restriction on whether the current user is allowed to minimize the Show Detail Frame.

It is left to you to enforce restrictions on the actions on a component. You can specify restriction on component actions in adf-config.xml. If a restriction is specified and applicable to the current user, the Panel Customizable or Show Detail Frame does not render the action.

For information about applying action-level restrictions, see Section 11.5, "Applying Action-Level Restrictions on Panel Customizable and Show Detail Component Actions."
This chapter describes how to use the Page service to create and manage pages at runtime. It includes the following sections:

- Section 6.1, "Introduction to Page Creation and Management"
- Section 6.2, "Creating Pages and Task Flows"
- Section 6.3, "Defining Values for the Page - Create New Task Flow Parameters"
- Section 6.4, "How to Create Task Flow View Pages"
- Section 6.5, "Managing Pages"
- Section 6.6, "Introduction to Custom Styles and Templates"
- Section 6.7, "Customizing Page Service Views"
- Section 6.8, "Introduction to the Page Service APIs"
- Section 6.9, "Page Service Samples"

6.1 Introduction to Page Creation and Management

The Page service enables you to create new pages and task flows in your application at runtime. You can base your pages on default or custom styles and templates. You can create them with either the Page - Create New task flow or the Page service APIs.

After you create pages or task flows, users can view and manage them with either the Page service data control or the Page service APIs. The APIs provide a means of defining a work area within the application, called a scope. Scopes are useful for categorizing custom pages that are of interest to a specific team or community (similar to a Group Space in WebCenter Spaces).

Table 6–1 describes the developer tools included with the Page service.

<table>
<thead>
<tr>
<th>Tool</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Page - Create New task flow</td>
<td>A task flow for creating pages or task flows at runtime.</td>
</tr>
</tbody>
</table>
The Page service is integrated with many Oracle WebCenter Services, such as the Links, Search, and Tags services. You can track the most recent changes in pages with the Recent Activities service. Because it is preconfigured to work with the Recent Activities service, the Page service automatically produces the information that the Recent Activities service uses to display the most recent additions, changes, or deletions in pages.

For more information about the services at runtime, see Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

### 6.2 Creating Pages and Task Flows

You can create pages and task flows at runtime with either the Page - Create New task flow or the Page service APIs.

<table>
<thead>
<tr>
<th>Tool</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PageServiceDC data control</td>
<td>A data control for viewing or deleting information about listed pages and task flows at runtime. PageServiceDC is available in design time after you add the Page service libraries to your application. By default, the Page service libraries are added to the application when you use the WebCenter template or when you add the Page - Create New task flow.</td>
</tr>
<tr>
<td>Page service APIs</td>
<td>APIs for creating and managing pages and task flows at runtime.</td>
</tr>
</tbody>
</table>

The Page service is integrated with many Oracle WebCenter Services, such as the Links, Search, and Tags services. You can track the most recent changes in pages with the Recent Activities service. Because it is preconfigured to work with the Recent Activities service, the Page service automatically produces the information that the Recent Activities service uses to display the most recent additions, changes, or deletions in pages.

For more information about the services at runtime, see Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

**Note:** The Page - Create New task flow provides a means of creating pages and task flows and editing task flow parameters. To execute operations, such as copy, rename, create scope, and so on, you must use the Page service APIs along with the Page - Create New task flow. The Page service APIs provide the flexibility of customization within the Page service.

The Page - Create New task flow enables you to invoke the Create Page dialog at runtime. The Create Page dialog in turn enables users to create pages based on predefined styles, schemes, and templates.

Page styles, schemes, and templates provide both a default page structure that describes the areas where you can place content (that is, the page layout) and a background color and image that contribute to a page look and feel. You can select the page style, scheme, scheme background color, and template when you create a page. You can additionally enhance the usefulness and presentability of a page using page layout components. These include an in-place HTML text editor, images, layout boxes, hyperlinks, and so on.

Figure 6–1 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 scheme, scheme background color, and layout when you revise a page. You can also start with a blank page and create layout, look, and feel from the start.

At runtime, you can view a list of pages created through the Page - Create New task flow and create, copy, update or delete those pages.


6.2.1 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 6.2.1.1, "How to Add the Page - Create New Task Flow"
- Section 6.2.1.2, "Setting Security for the Page Service"
- Section 6.2.1.3, "How to Create Pages at Runtime"
- Section 6.2.1.4, "Structure of Pages Created at Runtime"
- Section 6.2.1.5, "How to Access Pages Created at Runtime"
6.2.1.1 How to Add the Page - Create New Task Flow

To add the Page - Create New task flow to your WebCenter application:

1. Follow the steps described in Section 3.3, "Creating WebCenter Application-Enabled Pages."

2. Open the customizable page on which you want to add the service.


4. Drag and drop the Page - Create New task flow onto your page.

5. When prompted, select Region as the way to create the task flow.

6. Click OK.

7. Save your page.

8. Run your application to see that Create Page is now on the page.

9. Click Create Page.

The Create Page dialog opens (see Figure 6–1).

10. Enter a page name, then select the scheme, the scheme background color, and the template.

11. Click Create.

A page is created based on the selected information. For more details on where the page is created, see Section 6.2.1.4, "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 6.3, "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 6.5, "Managing Pages."

---

6.2.1.2 Setting Security for the Page Service

The Page service does not require its own, specific configuration of ADF security. It follows the security set up for the 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 Chapter 3.5, "Implementing Security in Your Application."

2. After you add the Page - Create New task flow to an application page, double-click the jazn-data.xml file in Application Resources to open the ADF Security Policies Editor.

3. If necessary, click the Task Flow tab to bring it forward.

4. In the Task Flow column, select page-create-new task flow (Figure 6–2).
5. Under **Granted to Roles**, click the **Add** icon to add a user role on which to grant task flow access privileges.

6. After you have added user roles, select a role in the ADF Security Policies dialog and then, under **Actions**, select the privileges to apply to the selected role.

7. Click the **Web Pages** tab to apply security to the page on which you added the Page - Create New task flow (Figure 6–3).

Follow the same procedure outlined in steps 3 through 6: select a page, add roles, apply privileges.
6.2.1.3 How to Create Pages at Runtime

For detailed information, see the chapter "Introducing the Page Service and Oracle Composer" in Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

6.2.1.4 Structure of Pages Created at Runtime

The pages you create at runtime using the Page - Create New task flow or the Page service APIs are stored in the mds folder, located under your application root. Table 6–2 provides examples of the default formats used for page and task flow file names.

Table 6–2 Default File Name Formats

<table>
<thead>
<tr>
<th>File Type</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Page</td>
<td>Page&lt;N&gt;.jspx (value of N starts from 1)</td>
</tr>
<tr>
<td></td>
<td>For example: Page1.jspx, Page2.jspx, Page&lt;N&gt;.jspx</td>
</tr>
<tr>
<td>Page Definition</td>
<td>PagelPageDef.xml, Page2PageDef.xml, Page&lt;N&gt;PageDef.xml</td>
</tr>
</tbody>
</table>

Note: The PageServiceDC data control does not require any security setting for a secure or non-secure application.
When you use the Page service APIs to create pages, you can provide a custom value for the file name format. For example: myPage<N>.jspx (value of N starts from 1)

The following examples show the structure of a page (Page1.jspx) and its associated page definition (Page1PageDef.xml) created at runtime by the user user1.

**Example 6–1  Page Created Under a Default Scope**
mds\oracle\webcenter\page\scopedMD\s8bba98ff_4cbb_40b8_beee_296c916a23ed\user\user1\Page1.jspx

**Example 6–2  Page Definition Created Under a Default Scope**
mds\pageDefs\oracle\webcenter\page\scopedMD\s8bba98ff_4cbb_40b8_beee_296c916a23ed\user\user1\Page1PageDef.xml

**Example 6–3  Page Created Under a Custom Scope**
mds\oracle\webcenter\page\scopedMD\<scopeGUID>\Page1.jspx

**Example 6–4  Page Definition Created Under a Custom Scope**
mds\pageDefs\oracle\webcenter\page\scopedMD\<scopeGUID>\Page1PageDef.xml

In Example 6–4, the scopeGUID is a unique ID assigned to a specific scope by the Page service.

**Example 6–5  Task Flow View Page Created Under a Default Scope**
mds\oracle\webcenter\page\scopedMD\s8bba98ff_4cbb_40b8_beee_296c916a23ed\user\user1\Page1.jspx

**Example 6–6  Task Flow Definition Created Under a Default Scope**
mds\oracle\webcenter\page\scopedMD\s8bba98ff_4cbb_40b8_beee_296c916a23ed\user\user1\Page1.xml

**Example 6–7  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 6–8  Task Flow View Page Created Under a Custom Scope**
mds\oracle\webcenter\page\scopedMD\<scopeGUID>\Page1.jspx

**Example 6–9  Task Flow Definition Created Under a Custom Scope**
mds\oracle\webcenter\page\scopedMD\<scopeGUID>\Page1.xml

**Example 6–10  Task Flow Page Definition Created Under a Custom Scope**
mds\pageDefs\oracle\webcenter\page\scopedMD\<scopeGUID>\Page1PageDef.xml

### Table 6–2 (Cont.) Default File Name Formats

<table>
<thead>
<tr>
<th>File Type</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task Flow View pages</td>
<td>Page1.jsff, Page2.jsff, PageN.jsff</td>
</tr>
<tr>
<td>Task Flow View Page Definitions</td>
<td>Page1PageDef.xml, Page2PageDef.xml, Page&lt;N&gt;PageDef.xml</td>
</tr>
<tr>
<td>Task Flow Definitions</td>
<td>Page1.xml, Page2.xml, Page&lt;N&gt;.xml</td>
</tr>
</tbody>
</table>
In Example 6–10, the `scopeGUID` is a unique ID assigned to a specific scope by the Page service.

6.2.1.5 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
```

6.3 Defining Values for the Page - Create New Task Flow Parameters

This section describes how to access and define values for Page - Create New task flow parameters. It contains the following subsections:

- Section 6.3.1, "How to Access Page - Create New Task Flow Parameters"
- Section 6.3.2, "Setting Scope in a Page - Create New Task Flow"
- Section 6.3.3, "Setting an Outcome Parameter"
- Section 6.3.4, "Specifying Styles"
- Section 6.3.5, "Specify an ADF Template"
- Section 6.3.6, "Showing a Command Link"
- Section 6.3.7, "Displaying an Image"
- Section 6.3.8, "Customizing the Label"

6.3.1 How to Access Page - Create New Task Flow Parameters

To access Page - Create New task flow parameters:

1. Go to the Application Navigator and right-click the `.jspx` page with the Page - Create New task flow.
2. Select Go to Page Definition, and scroll to select the Create Page task flow.
3. Click the Edit (pencil) icon to edit any of the parameters.

   The Page Data Binding Definition in the design view is shown in Figure 6–4.
Defining Values for the Page - Create New Task Flow Parameters

Figure 6–4  Page Data Binding Definition

The Edit Task Flow Binding dialog opens, as shown in Figure 6–5.

Figure 6–5  Edit Task Flow Binding for the Page - Create New Task Flow

The following optional Page - Create New parameters are listed.

- oracle_webcenter_page_createpage_scopename
- oracle_webcenter_page_createpage_outcome
- oracle_webcenter_page_createpage_templatefile
- oracle_webcenter_page_createpage_adftemplate
- oracle_webcenter_page_createpage_uitype
- oracle_webcenter_page_createpage_icon
- oracle_webcenter_page_createpage_label
6.3.2 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 6–6.

In Figure 6–6, `PageViewBean` is a custom managed bean that is invoking the `getScopeName()` method to fetch the value of `scopeName`. It uses the Page service API. By doing this, the Page - Create New task flow creates all pages and task flows under the specified scope, rather than default scope.

The following example code highlights only the creation of scope. If a scope already exists, then users can get the scope name from the `Scope` object.

**Example 6–11 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) {
```
6.3.3 Setting an Outcome Parameter

The outcome parameter defines a Java method that is called after the page is created. In this method, you can add dynamic content to the page, refresh the user interface that calls up the dialog to confirm creation of a new page, or initiate browser navigation to the newly created page. To specify a Java method to invoke after page creation (that is, it is called after users click the Create button in the Create Page dialog), enter the following in the Edit Task Flow Binding dialog:

```
oracle_webcenter_page_createpage_outcome = ${'view.MyTestClass.go'}
```

You can define the method either with a String parameter or with no parameter. The String parameter value is the new page's name with full path. If the method is defined both ways, then the one with a parameter is invoked.

In the following example, the method `go(String newPagePath)` is invoked. This method redirects users to the newly created page with the Java class `MyTestClass.java`. The value of `newPagePath` is passed by the Page service.

```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);
        }
    }
}
```
Defining Values for the Page - Create New Task Flow Parameters

```java
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());
}

6.3.4 Specifying Styles

The Page service provides out-of-the-box styles that are packaged within the Page service libraries. You can use these styles or create your own custom styles.

This section describes how to apply out-of-the-box styles and custom styles. It contains the following subsections:

- Section 6.3.4.1, "Out-of-the-Box Styles"
- Section 6.3.4.2, "Custom Styles"

---

**Note:** For more information about styles, see Section 6.6, "Introduction to Custom Styles and Templates."

---

6.3.4.1 Out-of-the-Box Styles

The following out-of-the-box-styles are provided (see Figure 6–1):

- Blank
- Left Narrow
- Right Narrow
- Three Column
- Stretch
- Text Page
- Web Page

To associate out-of-the-box styles with the Page - Create New task flow, use the optional parameter `oracle_webcenter_page_createpage_templatefile`. Use the parameter value to specify the location of the file that contains the list of out-of-the-box styles.

To associate out-of-the-box styles with the Page - Create New task flow:

1. Create a folder named `mystyles` under your application's `public_html` folder.
2. Within `mystyles`, create a file named `default_pageservice_styles.xml`.
3. Add entries for all the out-of-the-box styles using the `templateDef` element.

The `templateDef` element takes the following attributes:

- **name**: This attribute specifies the path to the `.jspx` page that defines a page style. When you use the default styles, you must specify the path to the page
that defines the style you intend to use. The default page style page files are located within the Page service libraries. For example:

```xml
name="/oracle/webcenter/page/pstemplates/TemplateThreeColumn.jspx"
```

TemplateThreeColumn.jspx is the page within the Page service library that defines the Three Column Template style.

- **title**: The value you enter for `title` is used as the lable for the style in the Create Page dialog. You can modify the title of an out-of-the-box style to any name. For example: `title="Three Column Layout"` or `title="Sales Standard"`.

- **icon**: This attribute specifies an image icon for a style.

For out-of-the-box styles, if you do not specify an icon for a style, then the Page service automatically uses the default icons assigned to each out-of-the-box style.

**Example 6–12** shows the content of a sample `default_pageservice_styles.xml` file that lists the other out-of-the-box styles:

**Example 6–12  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>
```


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.

**6.3.4.2 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).
2. Add an entry for the custom style in an XML file (for example, custom_pageservice_style.xml).

For example:

```xml
<templatesDef xmlns="http://xmlns.oracle.com/webcenter/page">
  <templateDef name="NewsTemplate.jspx" title="News" icon="/images/news.png"/>
</templatesDef>
```

**Note:** For more information about the syntax to use for templateDef element and the style-related .xml file, see Section 6.3.4.1, "Out-of-the-Box Styles."

3. Save the custom_pageservice_style.xml under a folder in your application's public_html folder.

For example:

```xml
.../public_html/mystyles/custom_pageservice_style.xml
```

4. The following files must be placed in the same folder as custom_pageservice_style.xml:
   - The file that defines the custom style, for example, NewsTemplate.jspx.
   - The page definition file of the custom style file, for example, NewTemplatePageDef.xml.
   - The ADF Template (for example, defaultTemplate.jspx) on which the custom style file is based. This step is applicable only if you are using ADF templates.

   **Note:** If you specify the location of an ADF template using the oracle_webcenter_page_createpage_adftemplate as described in following section, then you do not need to put this file (defaultTemplate.jspx) under the mystyles folder. You can specify a different location for the file by following Section 6.3.5, "Specify an ADF Template."

5. Open the Edit Task Flow Binding dialog as described in Section 6.3.1, "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:

```java
oracle_webcenter_page_createpage_templatefile = ${'/mystyles/custom_pageservice_style.xml'}
```
6.3.5 Specify an ADF Template

When you want to use the same template for both design time and runtime pages, you can use an ADF template. All that is required is to point the Page - Create New task flow to the template location.

To associate an ADF template with the Page - Create New task flow:

1. Open the Edit Task Flow Binding dialog as described in Section 6.3.1, "How to Access Page - Create New Task Flow Parameters."

2. In the Value column next to the `oracle_webcenter_page_creatpage_adftemplate` parameter, enter the path to the ADF template.

   For example:

   `${'templates/portalTemplate.jspx'}`

   In Figure 6–7, an ADF template called `portalTemplate.jspx` is used.

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.

6.3.6 Showing a Command Link

By default, the Page - Create New task flow is rendered as a command `button`. You can instead specify that it is rendered as a command `link` (Figure 6–8).

Figure 6–8 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 6.3.1, "How to Access Page - Create New Task Flow Parameters."

2. Revise the oracle_webcenter_page_createpage_uitype parameter to the following value: ${'link'}.

For example:

```
oracle_webcenter_page_createpage_uitype = ${'link'}
```

3. Click OK.

### 6.3.7 Displaying an Image

You can associate an icon with the Page - Create New task flow button or link. For example, Figure 6–9 shows the task flow rendered as a command link with a page icon.

**Figure 6–9  Command Link with a Page Icon**

![Create Page](image.png)

Figure 6–10 shows the task flow rendered as a command button with a page icon.

**Figure 6–10  Command Button with a Page Icon**

![Create Page](image.png)

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 6.3.1, "How to Access Page - Create New Task Flow Parameters."

2. Revise the oracle_webcenter_page_createpage_icon parameter to the following value: {'/images/page.jpg'}.

For example:

```
oracle_webcenter_page_createpage_icon = {'/images/page.jpg'}
```

3. Click OK.

### 6.3.8 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 6–11 shows the task flow rendered as a command button with a custom label.

**Figure 6–11  Custom Label**

![Create New Page](image.png)

To change the command button or link label:
1. Open the Edit Task Flow Binding dialog as described in Section 6.3.1, "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:
   
   \[
   \text{oracle_webcenter_page_createpage_label} = \{\text{Create New Page}\}
   \]

3. Click OK.

### 6.4 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 6.3.1, "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:
   
   \[
   \text{oracle_webcenter_page_createpage_label} = \{\text{Create Task Flow}\}
   \]

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 6.3.4.1, "Out-of-the-Box Styles."
6. Under your application's public_html folder, create a folder (for example, mystyles) and add the custom_taskflow_styles.xml file to it.

7. Make sure that the following files are also under the mystyles folder:
   - mytaskflow_view.jsff (task flow view page)
   - mytaskflow_viewPageDef.xml (task flow view page definition)
   - mytaskflow_view.xml (task flow definition)

8. Open the Edit Task Flow Binding dialog again, and provide the path to the XML file you created in step 5 to the oracle_webcenter_page_createpage_templatefile parameter.
   For example:
   ```
   oracle_webcenter_page_createpage_templatefile=$('${mystyles/custom_taskflow_styles.xml}')
   ```
   In this example, mystyles is a folder under the public_html folder of your application.

9. Run the home.jspx page, and then click the Create Task Flow button.

10. Enter the title of the task flow, select the News style (see Section 6.3.4.2, "Custom Styles"), and then click Create.

   The new task flow view page is rendered.

   For information on where the task flows are created, see section Section 6.2.1.4, "Structure of Pages Created at Runtime."

6.5 Managing Pages

You can manage pages using either the Page service data control or the Page service APIs. This section contains the following subsections:

- Section 6.5.1, "Using the Page Service Data Control to Manage Pages"
- Section 6.5.2, "Using the Page Service APIs to Manage Pages and Task Flows"

6.5.1 Using the Page Service Data Control to Manage Pages

You can customize Page service views with the Page service data control. The data control enables you to view information about existing pages at runtime and delete any of the listed pages. The Page service data control, PageServiceDC, is included in the Page service libraries and is available at design time after you add the libraries to the application.

This section describes how to add a Page service data control to a project and how to use it to view, edit, and delete pages. It contains the following subsections:

- Section 6.5.1.1, "How to Add the Page Service Data Control"
- Section 6.5.1.2, "How to View, Edit, and Delete Pages at Runtime"

6.5.1.1 How to Add the Page Service Data Control

This section provides an example of adding the Page service data control at design time.

To add a Page service data control to your project:

1. Ensure that you can see the PageServiceDC in the Application Navigator.
The PageServiceDC appears in the Application Navigator after you have performed either of the following actions:

- Added the **Page - Create New** task flow to your project (see Section 6.2.1.1, "How to Add the Page - Create New Task Flow").

- Added the data control from My Catalogs by expanding **WebCenter Services Catalog - Data Controls**, right-clicking **Page Data Control**, and selecting **Add to Project**.

2. In the Data Controls panel of the Application Navigator, expand **PageServiceDC**, as shown in Figure 6–12.

![Figure 6–12 Page Service Data Control](image)

3. Under **getPageTree()**, drag **PageTreeNode** and drop it on the page.

4. From the **Create** menu, select **Table**, and then select **ADF Read-only Table**.

5. In the dialog, enter the **scopeName** parameter to view all the pages listed under the specified scope.

   If you do not enter any value, then the data control takes the default scope. In Figure 6–13, the scope value is fetched from a custom managed bean.
6. Add a new column to the table you created at step 4.

7. To add a column for deleting pages at runtime, drag the `deletePage(string)` method and drop it in the new column.

8. From the Create menu, select Methods, and then select ADF Link.

9. In the Parameters section of the Edit Action Binding dialog, shown in Figure 6–14, specify the following value for the pageName parameter:

   ```java
   #{bindings.PageTreeNode.treeModel.rowData.pagePath}
   ```
10. Click OK.

11. From the Data Controls panel, expand `PageServiceDC`, `getPageTree()`, `PageTreeNode`, and then Operations.

12. Drag Execute and drop it on the page as an ADF Button.

13. In the Source view, set the button's Text attribute to Refresh.

   This provides a refresh feature on the list of pages to enable users to refresh the list after they have added a page or deleted one from the list.

14. Optionally, to display pages as links that invoke page Edit mode, view the page source and replace the `outputText` element in the table's page path column with a `goLink` as follows:

   Replace
   ```
   <af:outputText value="#{row.pagePath}"/>
   ```

   with
   ```
   <af:goLink destination="/faces#{row.pagePath}" text="#{row.title}" targetFrame="_top"/>
   ```
15. Save your page, and run it to your browser.

16. The created data control table should look something like Figure 6–15.

**Figure 6–15 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.jspx</td>
<td>/oracle/webcenter/p</td>
<td>steve</td>
<td>19/7/2008</td>
<td>False</td>
<td>deletePage</td>
</tr>
<tr>
<td>FinancePage</td>
<td>Page2.jspx</td>
<td>/oracle/webcenter/p</td>
<td>steve</td>
<td>19/7/2008</td>
<td>False</td>
<td>deletePage</td>
</tr>
<tr>
<td>HiPage</td>
<td>Page3.jspx</td>
<td>/oracle/webcenter/p</td>
<td>steve</td>
<td>19/7/2008</td>
<td>False</td>
<td>deletePage</td>
</tr>
<tr>
<td>MarketingPage</td>
<td>Page4.jspx</td>
<td>/oracle/webcenter/p</td>
<td>steve</td>
<td>19/7/2008</td>
<td>False</td>
<td>deletePage</td>
</tr>
<tr>
<td>OperationsPage</td>
<td>Page5.jspx</td>
<td>/oracle/webcenter/p</td>
<td>steve</td>
<td>19/7/2008</td>
<td>False</td>
<td>deletePage</td>
</tr>
</tbody>
</table>

6.5.1.2 How to View, Edit, and Delete Pages at Runtime

To view a page, follow the link generated from step 14 in the previous section. If the application has been secured properly, then an edit link should render on the page at runtime.

To delete a page, click the `deletePage` button in the row of the relevant page. This button was generated from in step 7 in the previous section. Click the Refresh button for the table to remove the deleted page row from the table.

For detailed information about how to view, edit and delete pages at runtime, see the chapter "Creating, Editing, and Deleting Pages" in *Oracle Fusion Middleware User’s Guide for Oracle WebCenter*.

6.5.2 Using the Page Service APIs to Manage Pages and Task Flows

The Page service provides APIs to manage pages and task flows. These include APIs to delete pages and task flows, change the hidden status, change the page scheme (name, background color, background image), and copy pages.

For more information about Page service APIs, see Section 6.8, "Introduction to the Page Service APIs."

6.6 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 6–1). 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 6.6.1, "How to Create Templates for Pages Created at Runtime"
- Section 6.6.2, "Using ADF Templates"
- Section 6.6.3, "Creating Styles for the Create Page Dialog"

**Note:** For more information about the templates.xml file, see Section 6.3.4, "Specifying Styles."

### 6.6.1 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 6.6.1.1, "How to Create a Page or Page Fragment Style"
- Section 6.6.1.2, "How to Edit Create Page Dialog Styles"

#### 6.6.1.1 How to Create a Page or Page Fragment Style

To create a page or page fragment style:

1. In Oracle JDeveloper, create a JSF template with all necessary information, such as header, welcome message, login/logout links, and so on.
2. When creating pages, create a page (JSPX) based on that template.
   
   When creating task flows, create a page fragment (JSFF) based on that template.
   
   When creating a page based on a template, the Page service actually copies the template and creates a new page based on it.
3. Put all the necessary content in the JSPX or JSFF file.
   
   For example, if there is a facet where you would like to display content, you can drop a Page Customizable component from Oracle 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 6–13 shows the sample code of a page fragment style, pstemplateview.jsff.
Example 6–13  Sample Code of a New Page Style

<?xml version='1.0' encoding='utf-8'?>
<!-- Copyright (c) 2006, 2008, Oracle and/or its affiliates. All rights reserved. -->
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version="2.1"
    xmlns:pe="http://xmlns.oracle.com/adf/pageeditor"
    xmlns:cust="http://xmlns.oracle.com/adf/faces/customizable"
    xmlns:f="http://java.sun.com/jsf/core"
    xmlns:af="http://xmlns.oracle.com/adf/faces/rich">
    <jsp:directive.page contentType="text/html;charset=utf-8"/>
    <f:loadBundle basename="oracle.webcenter.page.view.resource.PGUIBundle"
        var="res"/>
    <f:view>
        <af:form usesUpload='true' id="f1">
            <af:pageTemplate viewId="/oracle/webcenter/page/pstemplates/defaultTemplate.jspx" id="T">
                <f:facet name="content">
                    <pe:pageCustomizable id="pcl1">
                        <af:panelStretchLayout id="psl2" styleClass="replace_with_scheme_name" inlineStyle="replace_with_inline_style">
                            <f:facet name="center">
                                <af:panelGroupLayout id="pgl1" layout="scroll">
                                    <pe:layoutCustomizable id="lc1"
                                        showLayoutChanger="#{pageServiceBean.isEditMode}"
                                        text="Change Layout"
                                        showIcon="false"
                                        type="threeColumnNarrow"
                                        shortDesc="Layout Changer">
                                        <cust:panelCustomizable id="mainC"/>
                                        <f:facet name="contentA">
                                            <cust:panelCustomizable id="cnta"/>
                                        </f:facet>
                                        <f:facet name="contentB">
                                            <cust:panelCustomizable id="cntb"/>
                                        </f:facet>
                                    </pe:layoutCustomizable>
                                </af:panelGroupLayout>
                            </f:facet>
                        </af:panelStretchLayout>
                    </pe:pageCustomizable>
                </f:facet>
            </af:pageTemplate>
        </af:form>
    </f:view>
</jsp:root>
Alternatively, the JSPX or JSFF template file that you are creating can be based on an ADF template.

Example 6–14 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 6–14 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" 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">
          <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="lcl1"
                    showLayoutChanger="#{pageServiceBean.isEditMode}"
                    text="#{uib_o_w_w_r_WebCenter.LABEL_CHANGE_LAYOUT}"
                    showIcon="false" type="oneColumn"
```
Example 6–15 shows the sample code of a page style, NewsTemplate.jspx, that is based on an ADF template, portalTemplate.jspx. The viewId attribute of the af:pageTemplate tag provides the name of the ADF template used.

For information about using ADF templates, see the Oracle Fusion Middleware Web User Interface Developer’s Guide for Oracle Application Development Framework.

Example 6–15  Sample Code of a Page Style on 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 contentType="text/html;charset=utf-8"/>
    <f:view>
        <af:document title="#{pageDocBean.title}">
            <af:form usesUpload="true">
                <af:pageTemplate viewId="/mytemplates/portalTemplate.jspx" value="#{bindings.pageTemplateBinding}">
                    <f:facet name="pageContent">
                        <pe:pageCustomizable id='pgc1'>
                            <af:panelStretchLayout id='psl1' styleClass="#{pageDocBean.CSSStyle}"
                                inlineStyle="#{pageDocBean:inlineStyle}">
                                <f:facet name="center">
                                    <af:panelGroupLayout id='pgl1' layout='scroll'>
                                        <pe:layoutCustomizable id='lc1'>
                                            showLayoutChanger="#{pageServiceBean.isEditMode}"
                                            text="#{res['TEMPLATE.CHANGE_LAYOUT']}"
                                            showIcon='false' type='oneColumn'
                                        </pe:layoutCustomizable>
                                    </af:panelGroupLayout>
                                </f:facet>
                            </af:panelStretchLayout>
                        </pe:pageCustomizable>
                    </f:facet>
                </af:pageTemplate>
            </af:form>
        </af:document>
    </f:view>
</jsp:root>
```
To display a different style, or use a different template for a style, you must define it, for example, in /mytemplates/templates.xml. Place the directory /mytemplates under the web content root. For more information about the templates.xml file, see Section 6.3.4, "Specifying Styles."

Example 6–16 shows a sample templates.xml file.

Example 6–16 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"/>
</templatesDef>
</f:view>
</f:document>
</jsp:root>
```
6.6.1.2 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:

   ```xml
   <templateDef name="pstemplateview.jsff" title="News" icon="/images/news.png" type="taskflow"/>
   ```

   where
   - name is the name of the page style that you want to use.
   - title is the label displayed for the style in the Create Page dialog.
   - type is the type of page style. It can take the value page or task flow. The default value page, creates a page based on the style's associated template. The value task flow creates a task flow view of a page based on the style's associated template.

3. If you want to add a new style to the Create Page dialog, add a templateDef entry, as shown in the following example:

   ```xml
   <templateDef name="pstemplaterview3.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.

6.6.2 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 6.3.4, "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 6-15 shows a page template that is based on the ADF template portalTemplate.jspx.

When you specify an ADF template name for use during page creation, the Page service searches for the af:pageTemplate tag in the page template and updates the viewId attribute with the value you provided.

This section describes how to specify which ADF template to use under a given circumstance. It contains the following subsections:

- Section 6.6.2.1, "How to Specify the ADF Template Name Using the Task Flow Parameter"
- Section 6.6.2.2, "How to Specify the ADF Template Name Using API Parameters"

### 6.6.2.1 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.

### 6.6.2.2 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 schemeBGColor)
   ```
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 schemeBGColor)
```

where ADFTemplateViewID is the ID of the ADF page template view activity.

### 6.6.3 Creating Styles for the Create Page Dialog

To use your own page styles, define them in `/mytemplates/templates.xml` file. If you want to maintain a separate XML file with a different set of page styles, you can create an XML file and reference that from the Page - Create New task flow. For example, to display two different options to two users with different privileges, then you must create two XML files and define different styles in each XML file. You can then ensure that the Create Page dialog displays the respective options to each user.

This section describes how to create and reference a new style for the runtime Create Page dialog. It contains the following subsections:

- Section 6.6.3.1, "How to Create a Style for the Create Page Dialog"
- Section 6.6.3.2, "How to Reference a New Style from the Page - Create New Task Flow"

#### 6.6.3.1 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 6–17 shows the code entered into sample XML file `templates2.xml` that references the `ptemplateview.jsff` fragment you created in the previous section.

**Example 6–17 Sample Code for the Create Page Dialog 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.

6.6.3.2 How to Reference a New Style from the Page - Create New Task Flow

To reference the new XML file from the Page - Create New task flow:

1. Open the JSPX page that contains the Page - Create New task flow.
2. Right-click the JSPX page, and select Go to Page Definition.
3. In the page definition file, under Executables, select the Create Page task flow, and click the Edit icon on the header.

   The Edit Task Flow Binding dialog opens with the list of input parameters supported by the task flow.

4. Provide the template file name for the oracle_webcenter_page_createpage_templatefile parameter.

   If you define a constant value, for example \${'/mytemplates/templates2.xml'}, then this template is used for the Create Page dialog always.

   However, by providing an EL value for the parameter, you can ensure that a different template is used for the dialog based on different criteria, such as user or scope.

   For example, assume you have two users, user1 and user2, and want to display a different page style to each user. In this case, you must have two templates for the Create page dialog. Let us say you have created a new template, templates2.xml, and updated the default template, templates.xml. You can then specify \${TemplateBean.template} for the oracle_webcenter_page_createpage_templatefile parameter.

   To use this value, you must first create the managed bean, TemplateBean.java, with method getTemplate(), which returns templates.xml for user1 and templates2.xml for user2. At runtime, user1 and user2 are each shown different options in the Create Page dialog, as shown in Figure 6–16 and Figure 6–17.
5. Click **OK**.

6. Click **Save All** to save your work.

### 6.7 Customizing Page Service Views

This section describes various ways to customize Page service views. It contains the following subsections:

- Section 6.7.1, "Rendering Pages with ADF Faces Components"
- Section 6.7.2, "Managing User Security on Pages and Task Flows"
6.7.1 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 6.7.1.1, "Rendering Pages as Tabs Using ADF Faces Components"
- Section 6.7.1.2, "Rendering Pages as Links Using ADF Faces Components"
- Section 6.7.1.3, "Rendering Pages as Image Links Using ADF Faces Components"

6.7.1.1 Rendering Pages as Tabs Using ADF Faces Components

When you design your application to render pages as tabs, you can use a range of ADF Faces components to accomplish this. This section provides information about the ADF Faces components that support rendering pages as tabs. It contains the following subsections:

- Section 6.7.1.1.1, "Rendering Pages as Tabs Using af:navigationPane"
- Section 6.7.1.1.2, "Rendering Pages as Tabs Using af:panelTabbed"

6.7.1.1.1 Rendering Pages as Tabs Using af:navigationPane

Example 6–18 illustrates how pages can be rendered as tabs using navigationPane components.

Here a method called pages from a custom bean (MyPageServiceBean) is used to get the list of pages. For more information about getting the list of pages with the Page service API, see the getPages() method in Page service API section.

Example 6–18  Rendering Pages as Tabs Using af:navigationPane

```af:navigationPane id="tabs" 
<af:forEach var="tab" items="#{MyPageServiceBean.pages}">
  <af:commandNavigationItem text="#{tab.title}" id="cni1"/>
</af:forEach>
</af:navigationPane>
```

The output of the code depicted in Figure 6–18 renders as a series of tabs (Figure 6–18).

Figure 6–18  Render Pages as Tabs

![Figure 6–18  Render Pages as Tabs](image)

6.7.1.1.2 Rendering Pages as Tabs Using af:panelTabbed

Example 6–19 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. For more information about getting list of pages using Page service API, see the getPages() method in Page service API section.

Example 6–19  Rendering Pages as Tabs Using af:panelTabbed

```af:panelTabbed id="pt1" 
<af:forEach var="tab" items="#{MyPageServiceBean.pages}"
  <af:showDetailItem text="#{tab.title}" id="sdi1"/>
</af:forEach>
</af:panelTabbed>
```
The output of the code in Example 6–19 renders as a series of tabs (Figure 6–19).

Figure 6–19  Render Pages as af:panelTabbed Component

6.7.1.2 Rendering Pages as Links Using ADF Faces Components

This section provides examples of using ADF Faces Components to render pages as different types of links. It includes the following subsections:

- Section 6.7.1.2.1, "Rendering Pages as Links Using af:commandLink"
- Section 6.7.1.2.2, "Rendering Pages as Links Using af:goLink"

6.7.1.2.1 Rendering Pages as Links Using af:commandLink  Example 6–20 illustrates how pages can be rendered as links using af:commandLink.

Example 6–20  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 6–20 renders as a list of links (Figure 6–20).

Figure 6–20  Render Pages as Links

6.7.1.2.2 Rendering Pages as Links Using af:goLink  Example 6–21 illustrates how pages can be rendered as links using af:goLink.

Example 6–21  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}'>

6-34  Oracle Fusion Middleware Developer's Guide for Oracle WebCenter
The output of the code depicted in Example 6–21 renders as a list of links (Figure 6–20).

6.7.1.3 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 6–22 illustrates how pages can be rendered as image links.

Example 6–22  Rendering Thumbnail Views of Pages

In Example 6–22, pageCount is a method from a custom bean (MyPageServiceBean) that gets the number of pages in the current scope. Users must write this method on their own using the other methods from Page service APIs.

In Example 6–22, <image_name> is the name of the image with which you associate your page.

The output of the code in Example 6–22 is rendered as a linked series of thumbnail views of a page (Figure 6–21).
6.7.2 Managing User Security on Pages and Task Flows

You can manage access to pages and task flows either through the Security Panel, which is a part of Composer, or through Page service APIs.

For more information about runtime security, see the "Setting Page Access" section in the *Oracle Fusion Middleware User’s Guide for Oracle WebCenter*.

For more information about setting page access with the Page service APIs, see *Oracle Fusion Middleware Java API Reference for Oracle WebCenter*.

6.8 Introduction to the Page Service APIs

The Page service provides the following APIs:

- **Create Page**
  - Create Page at a user-defined Scope
- **Delete Page**
  - Delete Page with Page Path API
- **Create Task Flow**
  - Create Task Flow at a user-defined Scope
- **Delete Task Flow**
- **Create Scope**
- **Delete Scope**
- **Copy Page**
- **Get Page List**
- Change Page Title
- Change Hidden Status
- Change Page Scheme (name, background color, background image)
- Set Page Access

For more information about the Page service APIs, see Oracle Fusion Middleware Java API Reference for Oracle WebCenter.

6.8.1 Using the Page Service APIs

This section provides information about Page service 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 6.8.1.1, "Location of the Page Service APIs"
- Section 6.8.1.2, "How to Set up Your Application to Use the Page Service APIs"
- Section 6.8.1.3, "Example: How to Create a Page"

6.8.1.1 Location of the Page Service APIs

The Page service APIs are located in the oracle.webcenter.page.model.PageService class.

Example 6–23 Using Page Service APIs to Create a Task Flow View Page

PageDef createTaskflow(String nameFormat, String title, String pageTemplate, String pageTemplatePath, String cssStyle, String schemeBGImage, String schemeBGColor)
throws DuplicateNameException, InvalidNameException, LockUnavailableException;

where

- nameFormat is the name format of the task flow; it can contain a subpath, such as users/UserA/taskflow{0}.jspx.
- title is the title of the task flow.
- pageTemplate is the template used for the task flow.
- pageTemplatePath is the path for the task flow template.
- cssStyle is the cascading style sheet file for this task flow.
- schemeBGImage is the background image for the custom scheme.
- schemeBGColor is the background color for the custom scheme.

It returns the PageDef of the newly created task flow and throws the following:

- InvalidNameException if the name supplied cannot be used for update.
- DuplicateNameException if the page with the same name already exists.
- LockUnavailableException if another page or task flow operation is in progress.

6.8.1.2 How to Set up Your Application to Use the Page Service APIs

To use the Page service APIs, the WebCenter extension must be present in your JDeveloper application. In JDeveloper, go to the Help - About menu, and click the Extensions tab. Look for the WebCenter Page Service, as shown in Figure 6–22.
After confirming that the Page service is included with the WebCenter extension, you must add the Page service libraries to the project.

To add Page service libraries to your project:

1. Right-click your ViewController project and select Project Properties and then Libraries and Classpath.
2. Click Add Library and select the WebCenter Page Service and WebCenter Page Service View libraries, as shown in Figure 6–23.

Note: For information about adding the WebCenter extension, see Chapter 3, "Preparing Your Development Environment."
3. Click OK.

If you do not want to use JDeveloper as the code editor, then you can set the CLASSPATH to the following jar files:

- Jdev_Home/jdeveloper/webcenter/jlib/pagem.jar
- Jdev_Home/jdeveloper/webcenter/jlib/pages.jar
- Jdev_Home/jdeveloper/webcenter/jlib/page-service-view.jar
- Jdev_Home/jdeveloper/webcenter/jlib/page-service-skin.jar

### 6.8.1.3 Example: How to Create a Page

This section provides an example of how to create a page using the Page service APIs. Your requirements may call for different methods. Use the way the `createPage` method is invoked in this sample code as a model for invoking the methods you require on the `PageService` object.

Typically, pages created using Page service APIs are based on templates. Before you start creating pages using the APIs, you must ensure that each template file, for example `MyPageTemplate1.jspx`, has a page definition file, in this case `MyPageTemplate1PageDef.xml`, in the same directory as the JSPX page.

Example 6–24 shows how to use the `createPage()` API to create a scope named `MyScope`, create an MDS session instance, instantiate the Page service class with this new scope, and then create a page within this scope. The new page is created based on the `MyPageTemplate1.jspx` template.
**Example 6–24  Sample Showing How to Create a Page**

...  

```java
public void createMyPage
{
    try
    {
        MDSSession mdsSess = (MDSSession)ADFContext.getCurrent().getMDSSessionAsObject();
        PageServiceConfig config = new PageServiceConfig(mdsSess, DEFAULT_SCOPE);
        mPageService = PageServiceFactory.createInstance(config);

        String pageNameFormat = "Page{0}.jspx";
        String pageTitle = "New Page";

        // The template used to create the new page is
        // "mytemplates/MyPageTemplate1.jspx"
        // path "mytemplates/" should be defined as a MDS namespace.
        // Also MyPageTemplate1.jspx should have a pageDef called
        "MyPageTemplate1PageDef.xml" and
        // it has the same path as the jspx file.

        PageDef newPage = mPageService.createPage(
            PERSONAL_USER_PAGE, pageNameFormat, pageTitle,
            "MyPageTemplate1.jspx",
            "/mytemplates/",
            null, null, null);
    }
    catch(Exception e)
    {
        // Handle exception
    }
}
...  
```

**Note:** Pages and task flows created using Page service APIs do not have permissions granted by default. You must explicitly define permissions on the page.

For more information about the Page service APIs, see *Oracle Fusion Middleware Java API Reference for Oracle WebCenter.*

### 6.9 Page Service Samples

The Custom Portal Demo sample illustrates some Page service functionality, including the following:

- How to add the Page - Create New task flow
- How to configure task flow parameters
- How to use an ADF template
- How to use custom styles
- How to manage pages
The Custom Portal Demo sample is posted on the Oracle WebCenter Suite 11g Demonstrations and Samples page on the Oracle Technology Network (OTN) at:

http://www.oracle.com/technology/products/webcenter
Enabling Runtime Editing of Pages Using Oracle Composer

Oracle Composer provides components for controlling the application's design time at runtime behavior. This chapter describes how to add Oracle Composer components to your application page, thereby enabling runtime editing in the page.

For more information about Oracle Composer, see Chapter 5, "Introduction to Oracle Composer."

This chapter contains the following sections:

- Section 7.1, "Designing Editable Pages Using Oracle Composer Components"
- Section 7.2, "Designing Editable Pages Using Oracle Composer Components: Example"
- Section 7.3, "Populating Pages with Content"
- Section 7.4, "Troubleshooting Oracle Composer Problems"

7.1 Designing Editable Pages Using Oracle Composer Components

The Oracle 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 Oracle Composer provides options for entering page edit mode and modifying the page according to your requirements.

You can enable customizations in WebCenter and non-WebCenter 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 Oracle Composer components and their attributes, see Section 5.6, "Oracle Composer Components."

This section explains how to add Oracle Composer components to a page at design time to make it editable at runtime. It contains the following subsections:

- Section 7.1.1, "How to Create a Customizable Page"
- Section 7.1.2, "How to Enable Runtime Customization Using a Page Customizable"
- Section 7.1.3, "How to Enable Switching Between Page Modes Using a Change Mode Link or Change Mode Button"
Section 7.1.4, "How to Define Editable Areas of a Page Using Panel Customizable Components"

Section 7.1.5, "How to Enable Layout Customization for a Page Using a Layout Customizable"

Section 7.1.6, "How to Enable Component Customization Using Show Detail Frame Components"

Section 7.1.7, "How to Create a Page Template for Creating Customizable Pages"

Section 7.1.8, "How to Enable Customization in a Populated Page"

Section 7.1.9, "What Happens When You Add Oracle Composer Components"

Section 7.1.10, "What Happens at Runtime"

Section 7.1.11, "What You May Need to Know When Designing Editable Pages"

**Before You Begin**

In a secured application, user privileges impact runtime customization capabilities. Therefore, to enable users to edit pages in a secured application, you must ensure that you grant Edit or Customize privileges to users or roles.

For the steps to implement a basic security model in your application, see Section 3.5.1, "How to Configure ADF Security." For the steps to grant privileges to users or roles, see Section 11.2.2, "How to Define Roles and Grant Privileges in the jazn-data.xml File."

### 7.1.1 How to Create a Customizable Page

The steps for creating a customizable JSPX page in your WebCenter application are available in Section 3.3, "Creating WebCenter Application-Enabled Pages."

When you create a new page in JDeveloper, it is listed in the Application Navigator under **Web Content** as shown in Figure 7–1. Additionally, the page is opened in the editor and becomes the active editor panel.

*Figure 7–1 New Customizable Page (MyPage.jspx) Shown in the Application Navigator*
7.1.2 How to Enable Runtime Customization Using a Page Customizable

Adding a Page Customizable component to the page ensures that Oracle 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 7.1.9, "What Happens When You Add Oracle Composer Components."

---

**Note:** For considerations you must make before adding a Page Customizable to the page, see Section 7.1.11, "What You May Need to Know When Designing Editable Pages."

---

To add a Page Customizable component to the page:

1. Open a customizable JSPX page.
2. In the Component Palette, select ADF Faces and drag a Panel Stretch Layout component onto the page.

---

**Notes:** The Panel Stretch Layout component stretches the child in the center facet to fill all of the available space. This holds true when users resize the browser.

---

3. In the Component Palette, select Oracle Composer.
4. Add a Page Customizable component to the center facet of the Panel Stretch Layout.

You must ensure that the Page Customizable is nested inside an af:document element. The Page Customizable is a rich client component that requires the rich client framework to function properly.

5. The required attributes for a Page Customizable component are populated with default values when you add the component to the page.

You can define or modify attribute values by referring to Table B–1 in Section B.1, "Oracle Composer Component Properties."

6. Save the page.

By default, a Panel Customizable component is added as a child component and the Page Editor Panel is added as a facet in the Page Customizable component, as shown in Figure 7–2.
Figure 7–2  Page Customizable Component

Example 7–1 shows the pe:pageCustomizable tag in the page source view.

Example 7–1  Page Customizable Component in the Page Source View

```
<pe:pageCustomizable id="pageCustomizable1"
  <cust:panelCustomizable id="panelcustomizable1"
    layout="scroll"/>
    <f:facet name="editor">
      <pe:pageEditor id="pep1"/>
    </f:facet>
  </pe:pageCustomizable>
```

7.1.3 How to Enable Switching Between Page Modes Using a Change Mode Link or Change Mode Button

To enable users to switch to Edit mode of a page easily, you must add a Change Mode Link or Change Mode Button component to the page.

Note: An alternative way to enable switching to Edit mode is by using the Change Mode API. For more information about this API, see Oracle Fusion Middleware Java API Reference for Oracle WebCenter.

For things you should consider before adding a Page Customizable to the page, see Section 7.1.11, "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 Oracle Composer.
2. In the Structure window, within the top facet of the Panel Stretch Layout that you added in the previous section, drag a **Change Mode Link** or **Change Mode Button** component.

You must ensure that the Change Mode Link or Change Mode Button is nested in an `af:document` element. The Change Mode Link or Change Mode Button component is a rich client component that requires the rich client framework to function properly.

---

**Notes:**

- If you have not used a Panel Stretch Layout on your page, then add the Change Mode Link or Change Mode Button component above the Page Customizable component in the Structure window. This ensures that the Change Mode Link or Change Mode Button is displayed properly at runtime.

- Use a Change Mode Link or Change Mode Button only when you have a Page Customizable on the page. You may have problems running a page that contains a Change Mode Link or Change Mode Button but no Page Customizable component.

---

3. The required attributes for a Change Mode Link or Change Mode Button component are set by default when you add the component to the page.

Optionally, you can set any other attributes by referring to Table B–2 in Section B.1, "Oracle Composer Component Properties."

---

**Note:** In a secured application, it is recommended that you check all users' privileges and enable the Edit link or button on the page only for privileged users. Unauthenticated users who stumble into page Edit mode can change component properties.

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 7–3.
7.1.4 How to Define Editable Areas of a Page Using Panel Customizable Components

The Panel Customizable component is required for page composition or content management tasks, such as adding, arranging, or removing portlets or regions. By default, one Panel Customizable component is automatically added as a direct child of the Page Customizable component. You can add more Panel Customizable components within this Panel Customizable component according to your requirements.

It is only within a Panel Customizable component that you can drag and drop components at runtime.

Note: For considerations you must make before adding a Page Customizable to the page, see Section 7.1.11, "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 Oracle Composer.

2. Drag a Panel Customizable component to the Structure window and drop it at any suitable location within the form.

You must ensure that the Panel Customizable is nested in an af:document 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, “Oracle Composer Component Properties.”

Notes: If you select stretch layout for the Panel Customizable, then the first child component is stretched to fill up available space in the Panel Customizable component. Any other child components are ignored, though they are not removed from the page.

7.1.5 How to Enable Layout Customization for a Page Using a Layout Customizable

Use the Layout Customizable component to enable the runtime definition or modification of the layout of components on a page or an area of a page.

To add a Layout Customizable component:

1. From the Component Palette, select Oracle Composer.

2. Drag a Layout Customizable component to the Structure window and drop it inside the Panel Customizable component.

The target Panel Customizable component must be a child of the Page Customizable component.

Ensure that the Layout Customizable is nested in an af:document element. The Layout Customizable is a rich client component and requires the rich client framework to function properly.

Note: You can delete the direct child Panel Customizable in the Page Customizable and add the Layout Customizable as a direct child of the Page Customizable. However, you must ensure that the Page Customizable has only one direct child component.
3. The required attributes for a Layout Customizable component are set by default when you add the component to the page. Optionally, you can set any other attributes by referring to Table B–3 in Section B.1, "Oracle 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 7–5. A child Panel Customizable component is added by default in the Layout Customizable component. Additionally, a Panel Customizable component is added within each facet of the Layout Customizable component. These Panel Customizable components enable you to add content inside the Layout Customizable component at runtime.

The Panel Customizable added as a direct child provides the main area—the central area in a layout at runtime. The Panel Customizable components added within the two default Layout Customizable facets provide the two content areas, A and B. When you select a predefined layout at runtime, these three areas are arranged to display content in the selected pattern. See Predefined Layout Types for more information about how the content is laid out for each layout type.

Note: In a oneColumn layout, A and B are rendered only if the Panel Customizable components contain child components.

Figure 7–5  Layout Customizable Component
7.1.6 How to Enable Component Customization Using Show Detail Frame Components

When you want to enable customizations such as 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 Oracle Composer.
2. Drag a Show Detail Frame component to the Structure window and drop it inside a Panel Customizable component.
   
   The Show Detail Frame should be nested in a Panel Customizable component on the page.
3. The required attributes for a Show Detail Frame component are set by default when you add the component to the page.

   Optionally, you can set any other attributes by referring to Table B–7 in Section B.1, “Oracle Composer Component Properties.”

   The cust:showDetailFrame tag is added inside the cust:panelCustomizable tag as show in Figure 7–6.
For a better understanding of this type of task, see Section 7.2, "Designing Editable Pages Using Oracle Composer Components: Example."

7.1.7 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 Oracle 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 ViewController project of the application in which you want to create the template, right click the ViewController project, and choose New.

2. In the New Gallery, expand Web Tier, select JSF and then JSF Page Template, and click OK.

3. In the Create JSF Page Template dialog, enter a file name for the template, select a layout, and declare a new facet.

4. Click OK.

5. In the template, add a Change Mode Link or Change Mode Button component above the Panel Stretch Layout component.

6. Add a Page Customizable component within the center facet of the Panel Stretch Layout component.

7. Delete the child Panel Customizable component from the Page Customizable and add a Panel Group Layout component in its place.

8. Set the layout attribute on the Panel Group Layout to scroll.
9. Add a Facet Ref component inside the Panel Group Layout and specify the same name that you used to declare the facet in step 3. The template displays the facet as shown in Figure 7–7.

![New Page Template File](image)

Figure 7–7 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.

7.1.8 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.

Note: For best results, move components using the Structure window and not by editing the source code.

7.1.9 What Happens When You Add Oracle 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>/mds/oracle/adf/rc/metadata` directory. To add components to the default Resource Catalog available to end users of your application, see Chapter 12, "Configuring the Resource Catalog for Oracle Composer."

- A Resource Catalog Viewer is configured for the application. At runtime, Oracle 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, which is available in the <Application_Root>/<Project_Name>/public_html/WEB-INF directory. For more information, see Section 9.10, "Configuring the Persistence Change Manager."

■ When you create an application, a minimal adf-config.xml file is also created. When you add a Page Customizable to an application page, the required configuration is added to the adf-config.xml file. For example, change persistence is configured in the adf-faces-config section in this file. For more information, see Section 9.10, "Configuring the Persistence Change Manager."

■ The DataBindings.cpx file in the <Application_Root>/<Project_Name>/public_html/WEB-INF/afmsrc/<Project_Name> directory, is updated to enable the presence of task flows on the page.

■ The page definition file is updated with the binding for the Oracle Composer task flow, which is available as part of the WebCenter Framework extension JAR file. Example 7–2 shows the code in the page definition file after a Page Customizable component is added to an application page.

Example 7–2  Page Definition File After Adding Page Customizable Component

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<pageDefinition xmlns="http://xmlns.oracle.com/adfm/uimodel" 
    version="11.1.1.54.55" id="MyPagePageDef"
    Package="view.pageDefs">
    <parameters/>
    <executables>
        <variableIterator id="variables"/>
        <taskFlow id="pageeditorpanel"
            taskFlowId="#{pageEditorBean.pageEditorPanel}"
            xmlns="http://xmlns.oracle.com/afdf/controller/binding"/>
    </executables>
    <bindings/>
</pageDefinition>
```

See Also: Section B.2, "Oracle Composer-Specific Files and Configurations" for information about files that are created or modified when you add Oracle Composer components.

7.1.10 What Happens at Runtime

At runtime, users can perform all the tasks described Section 5.3, "Personalizing Capabilities in Page View Mode" and Section 5.4, "Editing Capabilities in Design View in Page Edit Mode."

Each Oracle Composer component provides runtime capabilities that are described in Section 5.6, "Oracle Composer Components."

Note: Runtime customizations that you perform on a page are not carried over when you deploy the application to a target server.

7.1.11 What You May Need to Know When Designing Editable Pages

When adding Oracle Composer components to your customizable page, consider the following:
To enable runtime customization of components, add only one Page Customizable to a page.

**Note:** Do *not* add a second Page Customizable component to your page. Only the first Page Customizable component is picked up when you run the page.

Ensure that the Page Customizable component has only one direct child component.

When you add multiple direct child components at design time, only the first child component is rendered at runtime. The first child component is stretched to fit the page. All other direct child components are ignored and not rendered on the page.

Place all components you want to be customizable within the Page Customizable component.

To enable runtime editing, you *must* ensure that the ID attribute is defined on all components on the page. Runtime editing of components that have no ID value is not supported in Oracle Composer.

If your page includes components with no ID value, then you may encounter problems while editing the page in Oracle Composer.

To ensure that the Change Mode Link or Change Mode Button is displayed properly at runtime, in the Structure window, add it above the Page Customizable component.

To enable View mode personalization, place a Show Detail Frame component within a Panel Customizable component.

Use a Show Detail Frame to enclose a single child only. If you must enclose multiple components in a Show Detail Frame, then place a grouping component, such as a Panel Group Layout or Panel Customizable, within the Show Detail Frame component and then place the ADF Faces components or other content within this grouping component.

Portlets need not be placed within Show Detail Frame components. Portlets come equipped with a header and display options that are similar to Show Detail Frame components.

### 7.2 Designing Editable Pages Using Oracle Composer Components: Example

In this example, assume you want to create a page that is customizable at runtime. The page is named MyPage.jspx in a WebCenter application named MyWebCenterApp.

To create a customizable page:

1. Create a WebCenter application named MyWebCenterApp by performing the steps in Section 3.2, "Creating a WebCenter Application."
2. Create a JSPX page named MyPage.jspx by performing the steps in Section 3.3, "Creating WebCenter Application-Enabled Pages."
4. Add a Page Customizable to the center facet by following the steps in Section 7.1.2, "How to Enable Runtime Customization Using a Page Customizable."

5. Set the border color of the Page Customizable to blue to differentiate the editable area from other noneditable areas.

   Under the Style category in the Property Inspector for the Page Customizable, click the Box tab and set the Border Color attribute to Blue.

6. Set the border color of child Panel Customizable to red.

   Under the Style category in the Property Inspector for the Panel Customizable, click the Box tab and set the Border Color attribute to Red.

7. Add a Change Mode Link to the top facet of the Panel Stretch Layout by performing the steps in Section 7.1.3, "How to Enable Switching Between Page Modes Using a Change Mode Link or Change Mode Button."

8. Inside the Panel Customizable that is a direct child of the Page Customizable, add a Layout Customizable component by following the steps in Section 7.1.5, "How to Enable Layout Customization for a Page Using a Layout Customizable."

9. Add Show Detail Frame components inside each Panel Customizable on the page by performing the steps in Section 7.1.6, "How to Enable Component Customization Using Show Detail Frame Components," and set the text attribute on the three Show Detail Frame components to Welcome, Support, and About, in that order.

10. For the sake of this example, add a Rich Text Editor and an Image component inside two of the Show Detail Frame components that are 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 7–8.

---

**Note:** The Panel Stretch Layout component stretches the child in the center facet to fill all the available space in the browser. This is true even if you resize the browser. Therefore, by placing your components within this child component, you can ensure that the customizable portion of your page occupies the complete browser area.

The page opens in View mode. Click the **Edit** link on the page to enter Edit mode. The page opens in Oracle Composer. In Oracle Composer, you can perform all editing tasks described in Section 5.4, "Editing Capabilities in Design View in Page Edit Mode."

Figure 7–9 shows how the page appears in Edit mode at runtime.
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 3.5.1, "How to Configure ADF Security."

2. Create three users ahunold, sking, and ngreenbe.
   For the detailed procedure, see Section 3.5.2, "How to Create 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.

7.3 Populating Pages with Content

After you create an editable page with the required Oracle 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 Oracle 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.

7.3.1 What You May Need to Know When Adding Content to the Page

Consider the following when adding content to your editable page:
Add components inside Panel Customizable components that are nested in the Page Customizable component. This ensures that the components can be edited at runtime.

You can include any Oracle ADFFaces component or task flow as child component of a Show Detail Frame component. However, portlets contain headers similar to those provided by Show Detail Frame components and can be added to Panel Customizable components directly. There are no additional benefits to including portlets in Show Detail Frame components.

For components that you want to make selectable in Source view in Oracle Composer, ensure the following:

- The component is compliant with Rich Client framework and generates a client side component
- The clientComponent attribute is set to true.

If your page has the ADF Faces components Output Text and Output Formatted nested inside a Page Customizable component, then ensure that you set the clientComponent attribute value. If this attribute value is not set, then you may encounter errors while trying to move or rearrange components on the page at runtime.

To consume portlets in your editable page, you must first register the portlet producers with your application. See Chapter 34, "Consuming Portlets" for details.

### 7.3.1.1 Considerations for Adding Task Flows

Users can personalize, customize, and edit task flows that are added to a customizable page. In a secured application, users with Customize permission on the task flow can also edit components on the task flow's page.

Consider the following points when adding task flows to your customizable page:

- To provide a consistent user experience in terms of look and feel, layout design, and interaction, and to avoid problems, such as too much white space, too many scroll bars, or slow, jerky, and unpredictable responses when displaying large data sets, ensure that the task flows are created by following the guidelines in Appendix D, "Guidelines for Creating Task Flows to Be Used in Oracle Composer-Enabled Pages."

- In a secured application, to enable users to view task flow content in Oracle Composer, ensure that the task flow has a TaskFlowPermission grant in the application's jazn-data.xml file, with at least View action provisioned.

- 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.

- Ensure that you specify valid values for all required parameters on the task flow.

For more information, see Section 11.6, "Applying Task Flow Restrictions."

### 7.4 Troubleshooting Oracle Composer Problems

This section provides information to assist you in troubleshooting problems you may encounter while using Oracle Composer.
Configuring ADF Logging for Oracle Composer

While creating your applications in JDeveloper, you can use the JDeveloper debugging tools to easily find errors in your web pages or page definition files. You can also set up the Java logger to display Java diagnostic messages. For detailed information about configuring logging, see the chapter titled “Testing and Debugging ADF Components” in the Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework. To configure logging for Oracle 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 Oracle Composer.
**Solution**

This error occurs if your application page includes both ADF Data Visualization components and Oracle 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.adfdt.model.dvt.objects.DvtElementObjectFactory; import oracle.adfdt.model.managers.FactoryManager; ...
static { FactoryManager.getInstance().registerFactory(new DvtElementObjectFactory()); }
```
This chapter describes how to use Oracle Composer's declarative and programmable extensibility mechanism to customize Oracle Composer to suit your business needs. It contains the following sections:

- Section 8.1, "Overview of Extensibility Options"
- Section 8.2, "Creating Oracle Composer Add-Ons"
- Section 8.3, "Creating Custom Property Panels"
- Section 8.4, "Extending the Expression Builder"
- Section 8.5, "Configuring Event Handlers for Oracle Composer UI Events"
- Section 8.6, "Configuring Drop Handlers in the Oracle Composer Catalog"
- Section 8.7, "Defining Property Filters"
- Section 8.8, "Enabling Parameter Support on the Customization Manager Task Flow"
- Section 8.9, "Customizing the Oracle Composer Toolbar"
- Section 8.10, "Troubleshooting Problems with Oracle Composer Extensibility Features"

8.1 Overview of Extensibility Options

Oracle Composer provides a framework on which to build customizable application pages. In addition to its default capabilities, you can extend the Oracle 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 Oracle Composer extension file, pe_ext.xml. For information about these files, see Section 8.1.8, "Configuration Files."

This section describes the options available for extending Oracle Composer runtime capabilities declaratively. It contains the following subsections:

- Section 8.1.1, "Oracle Composer Add-Ons"
- Section 8.1.2, "Oracle Composer Custom Property Panels"
- Section 8.1.3, "Oracle Composer Events Handlers"
- Section 8.1.4, "Drop Handlers."
- Section 8.1.5, "Component Property Filters"
8.1.1 Oracle Composer Add-Ons

The Page Properties and Reset Page buttons on the Oracle Composer toolbar (Figure 8–1) are examples of add-ons. Click these buttons to display panels for editing page properties and resetting page customizations. Typically, add-ons are custom task flows that are rendered as buttons on the Oracle Composer toolbar in page Edit mode. You can create add-ons that appear along with the Page Properties and Reset Page add-ons. For example, you can create an add-on to display page revision history so that it displays a Revisions button on the Oracle Composer toolbar. Clicking this button would display the page's revision history. You can also replace the Page Properties and Reset Page add-ons with custom add-ons that you create.

Figure 8–1 Default Add-Ons on the Oracle Composer Toolbar

The process of configuring custom add-ons includes creating the task flows, packaging them into JAR files, and defining them in the Oracle Composer extension file. For more information, see Section 8.2, "Creating Oracle Composer Add-Ons."

8.1.2 Oracle 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 Oracle 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 Oracle Composer extension file. For more information, see Section 8.3, "Creating Custom Property Panels."

8.1.3 Oracle Composer Events Handlers

Oracle 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 Oracle Composer, an event handler ensures that a specific action is performed. An event handler is the code that is called back by Oracle Composer when a composer event is invoked. Each UI event in Oracle Composer is associated with an event handler. Sometimes it may be necessary to augment Oracle Composer's innate capabilities by performing a different action or multiple actions on invoking an event. For example, when a user clicks Save, in addition to the save operation that Oracle 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 8.5, "Configuring Event Handlers for Oracle Composer UI Events."
8.1.4 Drop Handlers

Oracle Composer 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. Oracle Composer provides drop handlers to handle the add operation in the Catalog. Drop handlers are Java classes registered with Oracle Composer and called when users click an Add link in the Oracle Composer 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. For more information, see Section 8.6, "Configuring Drop Handlers in the Oracle Composer Catalog."

8.1.5 Component Property Filters

The Component Properties dialog displays properties of a selected component. By default, Oracle 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 8.7, "Defining Property Filters."

8.1.6 Customization Manager

Customization Manager is a task flow that enables users to download, upload, reset, and delete customizations on objects like pages and task flows. You can configure the Customization Manager either in Oracle Composer or outside of it, on some administrative page. For information about configuring and using Customization Manager, see Section 8.8, "Enabling Parameter Support on the Customization Manager Task Flow."

8.1.7 Oracle Composer Toolbar Customization

The default Oracle Composer toolbar displays elements on two rows. The first row displays the page name and status indicator, and the second row displays the View menu, Page Properties button, Reset Page button, and Close button, as shown in Figure 8–2.

Figure 8–2 Oracle Composer Toolbar with Default Elements

You can customize the toolbar by adding, deleting, or rearranging elements. You can also override existing elements with custom elements. For example, you can remove the message showing the page name if you do not want users to see the name of the page they are editing.

For information about customizing the Oracle Composer toolbar, see Section 8.9, "Customizing the Oracle Composer Toolbar."

8.1.8 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:
Creating Oracle Composer Add-Ons

- Oracle Composer extension file (pe_ext.xml)

The Oracle Composer extension file, pe_ext.xml, enables you to extend the editing capabilities provided by Oracle Composer. Within this file you can add elements to register new Oracle 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 Oracle 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 Oracle 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 Oracle Composer-specific configurations you can make in adf-config.xml, see Section B.2.2, "adf-config.xml."

8.2 Creating Oracle Composer Add-Ons

Oracle Composer provides the following default add-ons for runtime editing:

- Page Properties

This dialog opens when users click the Page Properties button on the Oracle Composer toolbar. The Page Properties dialog displays the current page’s properties and enables users to modify property values.

- Reset Page

The Reset Page dialog opens when users click the Reset Page button on the Oracle Composer toolbar. The Reset Page dialog enables users to remove 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 5.4.12, "Reset Page."

In addition to these, you can register new add-ons with Oracle Composer. For example, you can create an add-on to display page revision history so that it displays a Revisions button on the Oracle Composer toolbar. Clicking this button would display the page’s revision history.

This section contains the following topics:

- Section 8.2.1, "How to Create and Register Add-Ons"
- Section 8.2.2, "What Happens at Runtime"
- Section 8.2.3, "How to Exclude Oracle Composer Default Add-Ons"
8.2.1 How to Create and Register Add-Ons

You can create and register custom task flows that can be invoked from buttons on the Oracle Composer toolbar. All registered add-ons have an associated button that displays on the toolbar.

This section steps through the procedure of creating an add-on and registering it with Oracle Composer. It includes an example that demonstrates how to create an add-on that displays information about the Fusion Order Demo (FOD) application. The example add-on renders an About FOD button on the Oracle Composer toolbar (Figure 8–3) that users can click to invoke a task flow that contains information about FOD.

Figure 8–3  About FOD Button in Oracle Composer Toolbar

This section contains the following subsections:

- Section 8.2.1.1, "Creating an Add-On Task Flow"
- Section 8.2.1.2, "Registering Add-Ons with Oracle Composer"
- Section 8.2.1.3, "Registering Add-Ons in adf-config.xml"

8.2.1.1 Creating an Add-On Task Flow

Oracle Composer add-ons are task flows you create using JSPX pages or page fragments.

To create an add-on:

1. In your WebCenter application project, create a JSFF file called custompanelview.jsff:
   a. From the File menu, select New.
   b. In the New Gallery dialog, expand Web Tier, select JSF, then JSF Page or JSF Fragment.
   c. Click OK.
2. Design the fragment by adding the code shown in Example 8–1.

Example 8–1  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:af='http://xmlns.oracle.com/adf/faces/rich'>
  <af:panelGroupLayout id="pnlgrp1" layout="vertical" halign="center">
    <af:spacer id="sp1" height="20" />
    <af:image id="customimage" source="/images/FusionOrderDemoLogo.jpg"
      shortDesc="Fusion Order Demo Logo"/>
    <af:spacer id="sp2" height="20" />
    <af:outputText id="output1" value="Fusion Order Demo (FOD) is a sample
```
Creating Oracle Composer Add-Ons

shopping cart application based on Oracle ADF Framework. It also uses the Webcenter Framework to enable Collaboration, Customization and Personalization 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.


5. Save the task flow definition file.

Optionally, if you create the task flow in one application but want to consume it in another application, you must first package the task flow in an ADF library and add the resulting JAR in the consuming application.

To package the task flow in an ADF library:

1. Create a deployment profile for the task flow:
   a. Right-click ViewController and choose New.
   b. In the New Gallery, expand General, select Deployment Profile, and then ADF Library JAR File, and click OK.
   c. In the Create Deployment Profile -- ADF Library JAR File dialog, enter a name for your deployment profile and click OK.
   d. In the ADF Library JAR Deployment Profile Properties dialog, click OK.
   e. In the Project Properties dialog, click OK.
2. In the Application Navigator, right-click the project folder, choose Deploy, deployment profile name, to, and then choose to ADF Library JAR file. This creates a deploy folder including the JAR file, in the project folder located at <Application_Root>\ViewController\deploy\.

You can add this JAR file to any application in which you want to consume the add-on.

8.2.1.2 Registering Add-Ons with Oracle Composer

After you create the task flows, you must register them with Oracle Composer so that they are displayed on the Oracle Composer toolbar along with the default options.
To register an add-on with Oracle Composer:

1. If it does not already exist, create the Oracle Composer extension file, `pe_ext.xml` in the META-INF directory under the project’s Web context root (for example, in the `APPLICATION_HOME\ViewController\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 8–2 shows the code of the extension file with a `<panel>` entry.

Example 8–2 Oracle Composer Extension File

```xml
<?xml version="1.0" encoding="US-ASCII" ?>
<pe-extension xmlns="http://xmlns.oracle.com/adf/pageeditor/extension">
  <addon-config>
    <panels>
      <panel name="oracle.fod.custom.panel" title="About POD"
        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."

8.2.1.3 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:

   ```xml
   xmlns:pe="http://xmlns.oracle.com/adf/pageeditor/config"
   ```

3. Add a `<page-editor-config>` entry with the namespace and include an `<addon-panels>` entry.

When you register a custom add-on, the default add-ons are not displayed in Oracle Composer by default. To display all the default add-ons, you must set the
show-default-addons attribute on the <addon-panels> tag to true. The default value for this attribute is false.

Within <addon-panels>, add <addon-panel> entries for the new panels, as shown in Example 8–3.

The name attribute must contain the name you used to register the panel in the Oracle Composer extension file.

**Example 8–3  New Add-On Referenced in adf-config.xml**

```xml
<pe:page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">
  <addon-panels show-default-addons="true">
    <addon-panel name="oracle.fod.custom.panel"/>
  </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 Oracle Composer.

For information about the Oracle Composer-specific configurations you can make in adf-config.xml, see Section B.2.2, "adf-config.xml."

### 8.2.2 What Happens at Runtime

Custom add-ons that you register with Oracle Composer are rendered on the Oracle Composer toolbar along with the default add-ons.

In the example, an About FOD button is rendered on the Oracle Composer toolbar (Figure 8–4).

**Figure 8–4  About FOD Button in Oracle Composer Toolbar**

![About FOD Button in Oracle Composer Toolbar](image)

Clicking this button displays the About FOD task flow (Figure 8–5).

*Note:* The add-on panel is automatically sized to fit the content inside the task flow.
8.2.3 How to Exclude Oracle Composer Default Add-Ons

You can choose to show or hide the Page Properties, Reset Page, and Customization Manager buttons on Oracle Composer toolbar. To hide any of these add-ons, you must set the show-default-addons attribute on <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 8–4 shows the code to hide the Page Properties add-on.

```
Example 8–4 The adf-config.xml File with the Reset Page Option Excluded
<addon-panels show-default-addons="true">
  <!-- Hide the Reset Page add-on -->
  <addon-panel name="oracle.adf.pageeditor.addonpanels.page-reset"
  rendered="false"/>
  . . .
</addon-panels>
```

The excluded add-on (Reset Page) is not displayed on the Oracle Composer toolbar. The other default add-ons are displayed on the toolbar in the default order.

8.2.4 How to Display the Customization Manager Add-On

The Customization Manager is an Oracle Composer add-on that enables users to manage customizations on task flows, pages, and page fragments on a given page. For more information, see Section 5.4.15, “Manage Customizations.”

The Customization Manager add-on is available in the Oracle Composer library, but is not rendered by default. You can enable it by configuring it in your application’s adf-config.xml file using an <addon-panel> element, as shown in the following example:

```
<addon-panels show-default-addons="true">
  <addon-panel name="oracle.adf.pageeditor.addonpanels.customization-manager"/>
  . . .
</addon-panels>
```
Creating Custom Property Panels

8.2.5 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 Oracle Composer. To selectively hide an add-on, you must set the `show-default-addons` attribute on `<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 8–5.

**Example 8–5  rendered Attribute Setting in the adf-config.xml File**

```xml
<addon-panels show-default-addons="true">
    <addon-panel name="oracle.adf.pageeditor.addonpanels.page-reset"
                 rendered="#{securityBean.userInPageResetRole}" />

    ...  

</addon-panels>
```

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`, Oracle Composer does not display the Reset Page button on the Oracle Composer toolbar. If the returned value is `true`, Oracle Composer displays the Reset Page button.

**Note:** If you hide the Reset Page button on the Oracle Composer toolbar, the Reset Task Flow button displayed while editing task flow content is also hidden.

The default value for `rendered` is `true`. That is, if you do not specify a `rendered` attribute for an add-on, the add-on is always displayed.

8.3 Creating Custom Property Panels

Oracle Composer displays the properties of a component in the Component Properties dialog when a user clicks the **Edit** icon on the component. The Component Properties dialog provides a series of tabs. Each tab displays a group of related attributes. The attributes have associated values that control a component's behavior and visual style properties. For example, the Style tab displays the component's style-related properties, such as width, height, and background color.

Similarly, when a user clicks the **Page Properties** button, a Page Properties dialog opens with its own series of tabs. These tabs contain display-related page properties, page parameters, and security settings.

You can create and register custom property panels to render along with the tabs displayed in the Component Properties or Page Properties dialog. In addition, you can
remove the default panels or replace them with custom property panels. For example, you can develop a friendlier property panel for an Image component by displaying a picker for its Source property. This would make it easier for users to select an image from the available options.

This section describes how to create custom property panels. It also describes how to exclude, override, and selectively render default property panels. It contains the following subsections:

- Section 8.3.1, "How to Create and Register Custom Property Panels"
- Section 8.3.2, "What Happens at Runtime"
- Section 8.3.3, "How to Exclude Default Property Panels"
- Section 8.3.4, "How to Override Default Property Panels"
- Section 8.3.5, "How to Selectively Render Property Panels"

### 8.3.1 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 Oracle Composer extension file. 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 8.3.1.1, "Creating a Custom Property Panel"
- Section 8.3.1.2, "Registering a Custom Property Panel for a Component"
- Section 8.3.1.3, "Registering a Custom Property Panel for a Task Flow"

#### 8.3.1.1 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. Oracle 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, Oracle 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 Oracle Composer extension file are similar to those for creating and declaring Oracle Composer add-ons. For detailed information, see Section 8.2.1.1, "Creating an Add-On Task Flow" and Section 8.2.1.2, “Registering Add-Ons with Oracle Composer.”

#### 8.3.1.2 Registering a Custom Property Panel for a Component

After declaring a custom property panel in the extension file, you must also register it in the extension file and associate it with a component, if required. Registering a custom property panel ensures that the panel displays automatically in the Oracle Composer Component Properties and Page Properties dialog.

Use the property-panels element to register the custom property panel in the pe_ext.xml file and to associate the custom panel with a component.

To register a property panel for a component:
1. Create an Oracle Composer extension file, \texttt{pe\_ext.xml}, if it does not already exist.

For information about creating the extension file, see Section 8.2.1.2, "Registering Add-Ons with Oracle Composer."

2. Add a \texttt{<property-panels>} element inside the \texttt{<addon-config>} section in the \texttt{pe\_ext.xml} file.

3. Add a \texttt{<property-panel>} declaration within the \texttt{<property-panels>} element.

You can have multiple \texttt{<property-panel>} entries.

4. Within the \texttt{<property-panel>} element, add a \texttt{<component>} element to specify the runtime class name of the component (optional) and a \texttt{<panel>} element to specify the name you used to declare the panel in the \texttt{<addon-config>} section of the file.

A custom property panel registered for a specific task flow appears only when its associated task flow is selected. Otherwise, default property panels appear.

\begin{itemize}
\item \textbf{Example 8–6} shows a custom property panel that is associated with a Command Button component by specifying the component's fully qualified class name. For information about Oracle ADF components and their runtime classes, see Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework.
\end{itemize}

\begin{itemize}
\item \textbf{Example 8–6 Code to Register a Property Panel for a Component}
\end{itemize}

\begin{verbatim}
<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>
\end{verbatim}

\begin{itemize}
\item \textbf{Note:} When registering a property panel, if you do not associate it with a component or task flow, then the registered panel is rendered for all pages, task flows, and components in the Component Properties and Page Properties dialogs.
\end{itemize}

To configure multiple property panels for a component, you can include multiple \texttt{panel} elements within a \texttt{property-panel} element. For more information about the \texttt{property-panels} element and its nested elements, see Section B.2.1.2, "\texttt{property-panels}.”

\begin{itemize}
\item \textbf{8.3.1.3 Registering a Custom Property Panel for a Task Flow}
\end{itemize}

You can define custom property panels for a task flow by registering it with the Oracle Composer extension file. \textbf{Example 8–7} 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

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 8.2.1.2, "Registering Add-Ons with Oracle 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 8–7 shows a custom property panel associated with a dashboard task flow.

   **Example 8–7 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>
   
   **Note:** When registering a property panel, if you do not associate it with a component or task flow, 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."

   **8.3.2 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.
8.3.3 How to Exclude Default Property Panels

You can use the rendered attribute to show or hide a default property panel in the Page Properties and Component Properties dialogs. Set the attribute value to true to show the default property panel; set it to false to hide a default property panel.

Example 8–8 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, "Oracle Composer Default Add-Ons and Property Panels."

Example 8–8 rendered Attribute for a Property Panel

```xml
<pe-extension xmlns="http://xmlns.oracle.com/adf/pageeditor/extension">
  <addon-config>
    <property-panels>
      <property-panel name="cmdbtn">
        <component>oracle.rich.CommandButton</component>
        <panel name="prop.panel.cmdbtn"/>
        <panel name="oracle.adf.pageeditor.pane.content-style-editor"
          rendered="false"/>
      </property-panel>
    </property-panels>
  </addon-config>
  . . .
</pe-extension>
```

8.3.4 How to Override Default Property Panels

You can override a default property panel with a custom panel by setting the rendered attribute on the default panel to false and registering your custom panel with the same title that was used for the default panel in the Component Properties or Page Properties dialog. That is, provide the title of the default panel as the value for the title attribute within the panel element. Your custom property panel is then rendered on the default tab in lieu of the tab’s default properties.

For a list of default property panels that you can override, see Section B.3, "Oracle Composer Default Add-Ons and Property Panels."

8.3.5 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:

```xml
<property-panel name='global-but-just' rendered='#{bean.showProperty}'>
```

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.ComposerTab.
8.4 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 8–7. 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.

![Expression Language Editor](image)

You can customize the expression builder to provide more options by making the necessary entries in the Oracle Composer extension file, `pe_ext.xml`. This section explains how to extend the expression builder to add custom capabilities. It contains the following subsections:

- Section 8.4.1, "How to Extend the Expression Builder"
- Section 8.4.2, "What Happens at Runtime"

8.4.1 How to Extend the Expression Builder

You can extend the expression builder by adding custom options to the expression value dropdown list. Custom options must be defined and included in the `pe_ext.xml` file. For each new option you want to add, you must define and include a `<selector>` element in the `pe_ext.xml` file.

To configure custom options in the Oracle Composer extension file:

1. Create a Java bean, for example `CustomELParameter.java`, with the logic to populate the expression dialog with your custom options. Example 8–9 shows the code of a sample Java bean used to display a new string, Test, as a custom value in the expression dialog.

```
package view;

import java.util.ArrayList;
import java.util.List;
import oracle.adf.view.page.editor.elbuilder.ELParameter;
import oracle.adf.view.page.editor.elbuilder.ELParameterValue;

public class CustomELParameter implements ELParameter {
```

Example 8–9  Sample Code to Add a Custom Value in the Expression Dialog
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[] {"Test", "tests"};
}

private String mName = "Test";
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 8.2.1.2, "Registering Add-Ons with Oracle 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:

```xml
<elbuilder-config>
    <!-- define selector -->
    <selector id="CustomELParameter">
        view.CustomELParameter
    </selector>

    <!-- include selector -->
    <selectors>
        <selector id="CustomELParameter"/>
    </selectors>
</elbuilder-config>
```

   You can have multiple `<selector>` entries for different custom options.

4. Save the `pe_ext.xml` file.

### 8.4.2 What Happens at Runtime

The selector in the runtime expression builder displays the custom value along with the default values, as shown in Figure 8–8.
8.5 Configuring Event Handlers for Oracle Composer UI Events

Event handlers are Java classes registered with Oracle Composer and called when a user performs an action on the page. For example, when a user clicks a Save button, Oracle Composer calls back into the application code to give the application a chance to respond to the Save action—or event. In addition to the event handlers that Oracle Composer provides by default, you might want to configure the application to perform additional workflows. This is useful in performing application-specific tasks on such events. Moreover, this is the recommended approach to, for example, save the changes on a custom property panel.

Oracle Composer provides a means of registering event handlers for different events. This section describes how. It contains the following subsections:

- Section 8.5.1, "How to Create and Register Handlers for Composer UI Events"
- Section 8.5.2, "What Happens When You Create and Register Event Handlers"

8.5.1 How to Create and Register Handlers for Composer UI Events

When you register an event handler with Oracle 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 Oracle Composer. It contains the following subsections:

- Section 8.5.1.1, "UI Events that Support Event Handler Registration"
- Section 8.5.1.2, "Creating a Save Event Handler: Example"
- Section 8.5.1.3, "Registering an Event Handler with Oracle Composer"

8.5.1.1 UI Events that Support Event Handler Registration

Table 8–1 lists the UI events for which Oracle Composer currently supports registering of handlers.
Configuring Event Handlers for Oracle Composer UI Events

8.5.1.2 Creating a Save Event Handler: Example

To register an event with Oracle 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 8–1.

A Save event handler is called when a user clicks the Save button on the Oracle Composer toolbar or the Apply or OK button in the Component Properties dialog 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 Oracle 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 ViewController 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.

   **Tip:** You can use the SaveEvent.isCommit method to
8.5.3 Registering an Event Handler with Oracle Composer

After creating and implementing an event handler, you must register it with Oracle 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 Oracle Composer extension file, `/META-INF/pe_ext.xml`. For more information about creating this file, see Section 8.2.1.2, "Registering Add-Ons with Oracle 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.

8.5.2 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. Oracle 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 Oracle Composer continues with the next event handler. A warning message is logged.

8.5.3 Performing Additional Oracle Composer Event Handler Configurations

For every UI event triggered on the page in Oracle Composer, the corresponding event handler calls back a method from a listener registered with the application. By performing some additional configurations, you can specify the sequence in which event handlers must be executed and configure a listener to terminate a process or notify Oracle Composer that the event has been handled.

Specify a Sequence Number for an Event Handler

By specifying the sequence for event handlers, you can decide on the order in which the event handlers, and therefore the listeners, are called. You can assign a sequence number to a listener or modify the default value by defining a `sequence` attribute against the registered event handler in the `pe_ext.xml` file.

To specify a sequence number for a listener:

1. In the application's `pe_ext.xml` file, locate the `event-handler` element for the handler that you want to sequence.

2. Add a `sequence` attribute as follows:

   ```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.

Oracle Composer built-in listeners and listeners with no sequence numbers are assigned a default sequence number of 100. If you want your event handler to be called before other event handlers, you must specify a value lesser than 100.

3. Save the file.

Stop Event Processing and Notify Oracle Composer
You can configure an event handler to terminate processing of the current event and all subsequent events by throwing an exception. For this, you must declare the method used while implementing the listener as throws AbortProcessingException. In addition, you can configure the handler to notify Oracle 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 Oracle Composer. To enable this, you must implement the DeletionListener interface as shown in Example 8–10.

Example 8–10  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 Oracle 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)
            }
        }
        ...
    }
}
```
8.6 Configuring Drop Handlers in the Oracle Composer Catalog

Drop handlers are Java classes registered with Oracle Composer and called when users click an Add link in the Oracle Composer Catalog. A drop handler declares the data flavors it can handle. Each resource in the Oracle Composer Catalog has a flavor. When a user clicks an Add link next to a resource, Oracle Composer queries all the registered drop handlers 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. A user selects a drop handler from the context menu and that drop handler adds the resource to the page. An example for a resource with multiple 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 Oracle 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 8.6.1, "How to Create and Register Drop Handlers"
- Section 8.6.2, "What Happens at Runtime"

### 8.6.1 How to Create and Register Drop Handlers

This section describes how to create a sample drop handler for adding XML content from the Oracle Composer Catalog and register the drop handler with Oracle Composer. It contains the following subsections:

- Section 8.6.1.1, "Creating a Drop Handler"
- Section 8.6.1.2, "Registering a Drop Handler with Oracle Composer"

#### 8.6.1.1 Creating a Drop Handler

To create a drop handler with Oracle 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 Oracle Composer Catalog and create a drop handler named Custom XML for adding XML content.

To create a drop handler:

1. In JDeveloper, select the ViewController 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 Oracle 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 **Oracle 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 Oracle Composer 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 Oracle Composer 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

8-24 Oracle Fusion Middleware Developer's Guide for Oracle WebCenter


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.

### 8.6.1.2 Registering a Drop Handler with Oracle Composer

After implementing the drop handler, you must register it with Oracle Composer. Registration is necessary for ensuring that the handler is called by Oracle Composer when a user clicks an Add link in the Catalog. Register drop handlers in the Oracle Composer extension file, `/META-INF/pe_ext.xml`. For more information about creating this file, see Section 8.2.1.2, "Registering Add-Ons with Oracle Composer."

To register a drop handler:

1. In the `pe_ext.xml` file, add the following entries:

   ```xml
   <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.

8.6.1.3 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`, from the `Application_Root/mds/oracle/adf/rc/metadata` directory.

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="/eda/text/qualifier.png"/>
       <attribute attributeId="Description" value="/eda/text/qualifier.png"/>
       <attribute attributeId="IconURI" value="/af/pe/images/elementtext_ qualifier.png"/>
     </attributes>
     <parameters>
       <parameter id="xml">
       </parameter>
     </parameters>
   </component>
   ``


   At runtime, the Oracle Composer Catalog shows the XML component, as shown in Figure 8–9. For more information, see Section 12.4, "Adding XML Content to the Resource Catalog."
8.6.2 What Happens at Runtime

When a user clicks the Add link against a component in the Oracle Composer 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 8–10 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.
8.7 Defining Property Filters

Some component properties are not displayed in Oracle Composer’s Component Properties dialog because they are filtered out by default. The default filters are defined in the `<filter-config>` section of the Composer’s extension file (/META-INF/pe_ext.xml).

Global filters filter attributes for all components. They are defined using the `<global-attribute-filter>` tag. Tag-level filters filter attributes for a specified component only. They are defined using the `<taglib-filter>` tag.

---

**Note:** In an extension file, you can have any number of `<taglib-filter>` tags under `<filter-config>`, but you can have only one `<global-attribute-filter>` tag to define all global attribute filters.

---

You can define additional filters to hide more properties in the Component Properties dialog. This section describes how. It contains the following subsections:

- Section 8.7.1, "How to Define Property Filters"
- Section 8.7.2, "What Happens at Runtime"
- Section 8.7.3, "How to Remove Property Filters"
8.7.1 How to Define Property Filters

Oracle 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. Oracle Composer merges the filtering information from all the extension files.

To define property filters in an extension file:

1. 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="accessKey" />
       <attribute name="attributeChangeListener" />
       <attribute name="autoSubmit" />
       <attribute name="binding" />
     </global-attribute-filter>
     <taglib-filter namespace="http://xmlns.oracle.com/adf/faces/rich">
       <tag name="commandButton">
         <attribute name="text" />
         <attribute name="icon" />
       </tag>
     </taglib-filter>
   </filter-config>
   
   Note: For information about creating the `pe_ext.xml` file, see Section 8.2.1.2, "Registering Add-Ons with Oracle Composer."

2. Save the file.

8.7.2 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.

8.7.3 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 Oracle Composer's built-in filters or by another extension file defined elsewhere the application.

Note: After you remove a property filter, it is rendered in the Component Properties dialog even if a filter is defined for that property in another extension file.

To remove a property filter:

1. Edit the Oracle 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:
3. Save the file.

8.8 Enabling Parameter Support on the Customization Manager Task Flow

By default, an administrator cannot pass parameters to the Customization Manager task flow in Oracle Composer. If you enable parameter passing on the task flow, administrators can customize Customization Manager by setting certain parameters. Table 8–2 describes the parameters supported by the Customization Manager task flow.

Table 8–2 Customization Manager Task Flow Parameters

<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>dynamicTaskflowMap</td>
<td>Map&lt;String,List&lt;TaskflowID&gt;&gt;</td>
<td>Used to return a list of IDs of all task flows on a given page or fragment. For example: Page1.jspx -&gt; DT1, DT2 Page2.jspx -&gt; DT3, DT4 where, Page_n.jspx is the page name and DT_n are the IDs of all dynamic taskflows on that particular page.</td>
</tr>
<tr>
<td>showBase</td>
<td>true/false</td>
<td>Used to show or hide the column for the base document. Default: false</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. Default: true</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>showLayers</td>
<td>Comma-separated list of layer names</td>
<td>Used to specify a list of layers that must be available to the user in the Current Context and All Layers columns.</td>
</tr>
</tbody>
</table>
This section describes the steps to enable parameter editing on the Customization Manager task flow. It contains the following subsections:

- Section 8.8.1, "How to Enable Parameter Support on the Customization Manager Task Flow"
- Section 8.8.2, "What Happens When You Enable Parameter Support"

### 8.8.1 How to Enable Parameter Support on the Customization Manager Task Flow

To enable administrators 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 `<page-editor-config>` entry with the namespace and include an `<addon-panels>` entry.

   Within `<addon-panels>`, add an `<addon-panel>` entry for the default Customization Manager panel and define the `parameters` attribute by setting it to `appUtilBean.customizationManagerParams`, as shown in Example 8–11.

   ```xml
   <pe:page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">
       <addon-panels>
           <addon-panel name="oracle.adf.pageeditor.addonpanels.customization-manager" parameters="#{AppUtilBean.customizationManagerParams}"/>
           ...  
       </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
   package oracle.apps.view;
   import java.util.HashMap;
   import java.util.Map;
   ```

### Table 8–2 (Cont.) Customization Manager Task Flow Parameters

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>showDownload</td>
<td>true/false</td>
<td>Used to show or hide the Download link.</td>
</tr>
<tr>
<td>Default: true</td>
<td></td>
<td></td>
</tr>
<tr>
<td>showUpload</td>
<td>true/false</td>
<td>Used to show or hide the Upload link.</td>
</tr>
<tr>
<td>Default: true</td>
<td></td>
<td></td>
</tr>
<tr>
<td>showDelete</td>
<td>true/false</td>
<td>Used to show or hide the Delete link.</td>
</tr>
<tr>
<td>Default: true</td>
<td></td>
<td></td>
</tr>
<tr>
<td>showPromote</td>
<td>true/false</td>
<td>Used to show or hide the Promote link.</td>
</tr>
<tr>
<td>Default: true</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This section describes the steps to enable parameter editing on the Customization Manager task flow. It contains the following subsections:

- Section 8.8.1, "How to Enable Parameter Support on the Customization Manager Task Flow"
- Section 8.8.2, "What Happens When You Enable Parameter Support"
public class AppUtilBean
{
  public Map<String, String> getCustomizationManagerParams()
  {
    Map<String, String> paramMap = new HashMap<String, String>();
    paramMap.put("showUpload", "false");
    return paramMap;
  }
}

In this example, as the showUpload parameter is set to false, administrators cannot control the display of the Upload link.

4. Save the adf-config.xml file.

8.8.2 What Happens When You Enable Parameter Support

The application administrator can now set Customization Manager task flow parameters. Customization Manager displays options to users based on the new parameter settings. For example, the Upload option is not displayed if the administrator sets the showUpload parameter to false.

8.9 Customizing the Oracle Composer Toolbar

You can customize the default Oracle Composer toolbar by adding, removing, or rearranging elements and overriding existing elements. To enable toolbar extensibility, the Oracle Composer toolbar elements have been grouped into sections. You can work with these sections to customize your toolbar. Each section has a specific name and contains one or more elements inside it, as shown in Figure 8–11.
Customizing the Oracle Composer Toolbar

Table 8–3 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 Source view.</td>
</tr>
<tr>
<td>addonpanels</td>
<td>Area containing Oracle Composer add-ons. The add-ons are all displayed in a sequence within this area. For more information, see Section 8.2, &quot;Creating Oracle 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 Oracle Composer toolbar in the following ways:
- Rearrange and hide built-in elements
- Add new sections with custom elements
- Modify default sections to display custom elements

This section describes the steps to customize the Oracle Composer toolbar. It contains the following sections:
- Section 8.9.1, "How to Rearrange or Hide Toolbar Elements"
- Section 8.9.2, "What Happens at Runtime"
- Section 8.9.3, "How to Add New Sections to the Toolbar"
- Section 8.9.4, "How to Override a Toolbar Section to Display Custom Content"

8.9.1 How to Rearrange or Hide Toolbar Elements

Use the toolbarLayout attribute on a Page Customizable component to control which toolbar sections are displayed and the order in which they appear.
To customize the toolbar for all editable pages in the application, you can create a template, add a Page Customizable component to the template, and specify the toolbarLayout attribute against the Page Customizable component. You can then base all the pages in the application on that template.

If you do not specify a value for toolbarLayout, this attribute is internally set to message stretch statusindicator newline menu addonpanels stretch help button, which is the default layout for the Oracle Composer toolbar sections.

To customize the toolbar on an application page:

1. Open your customizable JSPX page in JDeveloper and select the Page Customizable component in the Structure window.
2. In the Property Inspector, specify space-separated values for the toolbarLayout attribute.

   The section names in Table 8–3 are valid values for this attribute.

   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.

### 8.9.2 What Happens at Runtime

At runtime, the toolbar displays the sections you specified in the order you specified them (Figure 8–12).

#### Figure 8–12 Customized Oracle Composer Toolbar

![Customized Oracle Composer Toolbar](image)

### 8.9.3 How to Add New Sections to the Toolbar

You can add custom sections by creating facets and specifying the facet names in the toolbarLayout attribute. Populate the new facets with elements you want to display on the toolbar. The following example shows you how to create a section; specifically, how to add a Report a Bug button that opens a popup that enables users to report a bug.

To add a custom section:

1. Open the JPSX page for which you want to customize the toolbar.
2. Add a facet inside the Page Customizable and name it bugreport.
3. Add a Command Toolbar Button component inside the facet and set the text attribute to Report a Bug.
4. To add an image to the button, specify the path to the image using the icon attribute.
5. Add the Show Popup Behavior component to invoke a popup on clicking the Command Toolbar Button.

In Source view, the Page Customizable would appear as follows:

```xml
<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 Oracle Composer toolbar displays a Report a Bug button, as shown in Figure 8–13.

![Oracle Composer Toolbar with Custom Section](image)

Clicking the Report a Bug button displays the File a Bug dialog that enables users to report a bug.

8.9.4 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"/>
```
Figure 8–14 shows the Oracle Composer toolbar displayed for the page at runtime.

8.10 Troubleshooting Problems with Oracle Composer Extensibility Features

This section provides information to assist you in troubleshooting problems you may encounter while using the Oracle Composer extensibility features.

For information about configuring logging, see "Configuring ADF Logging for Oracle Composer".

**Problem**
You created an add-on, but it does not appear on the Oracle 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 `addon-panel` entry exists under the `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 8.2, “Creating Oracle 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 8.3, "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 8.7.1, "How to Define Property Filters" and Section 11.4, "Applying Attribute-Level Security."
Performing Advanced Oracle Composer Configurations

This chapter describes how to perform certain advanced Oracle Composer configurations to enhance the end user experience. It contains the following sections:

- Section 9.1, "Enabling Custom Actions on Show Detail Frame Components by Using Facets"
- Section 9.2, "Enabling Custom Actions on Show Detail Frame Components By Using Facets: Example"
- Section 9.3, "Enabling Custom Actions on a Show Detail Frame Enclosing a Task Flow"
- Section 9.4, "Enabling Custom Actions On a Show Detail Frame Enclosing a Task Flow: Example"
- Section 9.5, "Enabling Custom Actions on Task Flows that Display in the Component Navigator"
- Section 9.6, "Creating Event-Enabled Task Flows"
- Section 9.8, "Disabling Source View for the Application"
- Section 9.9, "Applying Styles to Components"
- Section 9.10, "Configuring the Persistence Change Manager"
- Section 9.11, "Configuring Runtime Resource String Editing"

9.1 Enabling Custom Actions on Show Detail Frame Components by Using Facets

You can use the Show Detail Frame facets to define and display custom actions on Show Detail Frame components. For example, if your Show Detail Frame contains a list of services provided in your application, you can add a custom action, Show Detailed Information, which opens up a task flow containing details about the various services.

For information about the facets supported by Show Detail Frame components, see Section B.1.5, "Show Detail Frame Component."

Oracle JDeveloper displays all facets available to the Show Detail Frame component in the Structure window, however, only those that contain UI components appear activated.
To add a Show Detail Frame facet, perform the following steps:
1. Right-click a Show Detail Frame component in the Structure window, and select Facets - Show Detail Frame.
2. Click the arrow to the right of this option.
3. From the list of supported facets, select the facet you want to add.

**Note:** A check mark next to a facet name means the facet is already inserted in the Show Detail Frame component. Selecting that facet name again would result in the facet getting deleted from the page.

The `f:facet` element for that facet is inserted in the page.

4. Add components in the facet according to your design requirements.

For an end-to-end example of creating and using Show Detail Frame facets, see Section 9.2, "Enabling Custom Actions on Show Detail Frame Components By Using Facets: Example."

### 9.2 Enabling Custom Actions on Show Detail Frame Components 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 9.2.1, "How to Create an ADF Task Flow"
- Section 9.2.2, "How to Include an Additional Actions Facet"
- Section 9.2.3, "How to Create a Redirection Page"
- Section 9.2.4, "How to Create Navigation Rules Between Pages"
- Section 9.2.5, "What Happens at Runtime"

#### 9.2.1 How to Create an ADF Task Flow

To create an ADF task flow:

1. From the File menu, select New.
2. In the New dialog, select JSF under Web Tier node, and select ADF Task Flow under Items.
3. Click OK.
4. In the Create Task Flow dialog, click OK to create a task flow definition file by accepting the default values.
5. From the ADF Task Flow tag library in the Component Palette, drop two view elements, view1 and view2, onto the task flow definition file.

6. Drop a Control Flow Case from view1 to view2.

7. Click the first view element and then click the second view element to draw the control flow line.
   Name this control flow case next.

8. Similarly, drop a Control Flow Case from view2 back to view1 and name it prev.

9. Create a backing bean called BackingBean.java to contain values for two variables view1 and view2.

   view1 and view2 are initialized with initialValue1 and initialValue2 respectively. Ensure that the code of the bean is as show in the following example:

   ```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.
9.2.2 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 9–1.

8. Save the page.

9.2.3 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.

9.2.4 How to Create Navigation Rules Between Pages

To define navigation rules between the pages:

1. In the Application Navigator, under the project’s WEB-INF folder, double-click the 
faces-config.xml file to open it.

2. Click the Overview tab to view the file in Overview mode.

3. Click the Add button in the Managed Bean section.

4. In the Create Managed Bean dialog, specify backingBean as the name.

5. For the Class Name field, click the Browse button adjacent to it and browse to the 
BackingBean Java class that you created earlier. Click OK.

6. Click OK.

7. From the Scope list, select session and click OK.

8. Select the Navigation Rules tab on the page.

9. In the From Views table, select Page1.jspx.

10. Under Navigation Cases, select Page2.jspx in the To View ID column, customize 
in the From Outcome column, and true in the Redirect column.

11. In the From Views table, select Page2.jspx.

12. Under Navigation Cases, select Page1.jspx in the To View ID column, back in the 
From Outcome column, and true in the Redirect column.

In Source view, these entries appear as follows:

<managed-bean>
    <managed-bean-name>backingBean</managed-bean-name>
    <managed-bean-class>view.BackingBean</managed-bean-class>
    <managed-bean-scope>session</managed-bean-scope>
</managed-bean>

<navigation-rule>
    <from-view-id>/Page1.jspx</from-view-id>
    <navigation-case>
        <from-outcome>customize</from-outcome>
        <to-view-id>/Page2.jspx</to-view-id>
        <redirect/>
    </navigation-case>
</navigation-rule>

<navigation-rule>
    <from-view-id>/Page2.jspx</from-view-id>
    <navigation-case>
        <from-outcome>back</from-outcome>
        <to-view-id>/Page1.jspx</to-view-id>
        <redirect/>
    </navigation-case>
</navigation-rule>
13. Save the file.

9.2.5 What Happens at Runtime

In JDeveloper, run Page1.jspx. The Actions menu on the showDetailFrame1 component displays the Customize action, as shown in Figure 9–2.

Figure 9–2 Customize Action on the Actions Menu

![Customize Action on the Actions Menu](image)

Clicking Customize takes you to Page2.jspx (Figure 9–4), where you can update the values for Label1 and Label2.

Figure 9–3 Page with Option to Edit Text

![Page with Option to Edit Text](image)

Clicking OK takes you back to Page1.jspx, which reflects the recent text changes, as shown in Figure 9–4.
9.3 Enabling Custom Actions on a Show Detail Frame Enclosing a Task Flow

You can customize a task flow by including it in a Show Detail Frame component on which you define custom actions. Further, you can define custom actions that, when invoked at runtime, trigger the desired navigational flow. For example, you can define a custom action on a Show Detail Frame that specifies that the target task flow fragment opens in a separate browser window rather than inside the Show Detail Frame.

You can specify custom actions on Show Detail Frame components by adding Custom Action components as children of a Show Detail Frame component on the page. Custom actions defined in this way would be available only on the Show Detail Frame instance, which has the custom action as its child. Alternatively, you can specify custom actions in the application's adf-config.xml file. Custom actions defined in this way are available on all Show Detail Frame instances in the application.

This section provides information about defining custom actions on a Show Detail Frame. It includes the following subsections:

- Section 9.3.1, "How to Define Custom Actions at the Instance Level"
- Section 9.3.2, "How to Define Custom Actions at the Global Level"
- Section 9.3.3, "How to Configure Custom Actions that Display Task Flow Views in a Separate Browser Window"

9.3.1 How to Define 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 Oracle 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 Oracle 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 header.

---

**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, "Oracle 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 9–1.

**Example 9–1  Custom Action Code**

```xml
<cust:showDetailFrame text='showDetailFrame 1' id='sdf1'>
  <af:region value='#{bindings.taskflowdefinition1.regionModel}'
    id='r1'/>
  <cust:customAction action='navigatefromview1' id='ca1'
    location='both' icon='/Logo1.JPG'
    text='View1 Action'
    shortDesc='Custom View1 Action'/>
  <cust:customAction action='navigatefromview2'
    location='both' id='ca2'
    icon='/Logo2.JPG' text='View2 Action'
    shortDesc='Custom View2 Action'/>
</cust:showDetailFrame>
```

---

**Notes:** You can add custom actions for all of the task flow's ADFc outcomes, but depending on the task flow view that is displayed, several or none of the custom actions are available at runtime.

If you define a custom action without a corresponding task flow action, then the custom action is not rendered on the Show Detail Frame header at runtime.

---

**See Also:** "Creating ADF Task Flows" in Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework

4. Save and run the page.

At runtime, when you select an action from the Show Detail Frame's Actions menu, the associated control flow rule is triggered and the target task flow fragment is rendered.

### 9.3.2 How to Define 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 for all Show Detail Frame components in an application, at runtime the header of any Show Detail Frame displays only
Enabling Custom Actions on a Show Detail Frame Enclosing a 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 9–2.

**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 9–2 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 11.5, "Applying Action-Level Restrictions on Panel Customizable and Show Detail Component Actions," you already have a cust:customizableComponentsSecurity section in the adf-config.xml file. You can define custom actions in that section itself instead of the cust:adf-config-child section shown in Example 9–2.
- 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 9.4, "Enabling Custom Actions On a Show Detail Frame Enclosing a Task Flow: Example."
Resolving Conflicts Between Global-Level and Instance-Level Custom Actions

Each custom action is uniquely identified by the value of its action attribute. If you have defined custom actions with the same action attribute value at the global and instance levels, then there may be conflicts in how these custom actions are invoked at runtime, depending on other attribute values. At such times, the inheritGlobalActions attribute of the Show Detail Frame defines the behavior of other custom action attributes (other than the action attribute) as follows:

---

**Note:** Regardless of the inheritGlobalActions setting (true or false) on the Show Detail Frame component:

- The rendered attribute is not inherited even if it is not specified at the instance level.
- The location attribute at the global or instance level should be set to the same value at both the global and instance levels.

---

- If inheritGlobalActions=true, or you have not specified a value for inheritGlobalActions (defaults to false), the behavior of custom action attributes is as follows:
  - If you defined a custom action attribute at the global and instance levels, then the attribute value specified at the instance level is used.
  - If you defined a custom action attribute only at the instance level, then that attribute value is used.
  - If you defined a custom action attribute only at the global level, then that value is ignored and the default value is used.

- If inheritGlobalActions=true, the behavior of custom action attributes is as follows:
  - If you defined a custom action attribute at the instance level, then its value is used regardless of whether the same attribute is specified at the global level.
  - If you defined a custom action attribute only at the global level, then that value is used.
  - If you have not defined a custom action attribute at the global or instance levels, then the attribute's default value is used.

After you have designed your application pages, you must deploy the application to the production environment. For more information, see Chapter 38, "Testing and Deploying Your WebCenter Application."

---

**Note:** Runtime customizations that you perform on the page in the development environment are not carried over when you deploy the application to a target server.

9.3.3 How to Configure Custom Actions that Display Task Flow Views in a Separate Browser Window

Custom actions typically display the target task flow views in place, inside the Show Detail Frame component. However, you can define a custom action to display a task flow view in a separate browser window.
To display a task flow view in a separate browser window, the control flow rule to that view must be prefixed with `dialog:` in the task flow definition file and in the `action` attribute of the custom action corresponding to that view. The following example shows an `action` attribute definition:

```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="300"
    windowWidth="200"
    windowModalityType="applicationModal"
    returnListener="#{pageFlowScope.recentPagesBean.refreshMainView}" visible="false"/>
```

Example 9–3 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 9–3  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"/>
    </customActions>
</customizableComponentsSecurity>
```
Example 9–4 shows how you can specify an instance-level custom action corresponding to the task flow outcome by adding a Custom Action component from the Oracle 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 9–4  Instance-Level Custom Action Defined in the JSPX Page**

```xml
<cust:showDetailFrame id="sdf_for_RecentPagesTF1"
    text="Recent Pages" stretchContent="false"
    showResizer="never">
    <af:region id="RecentPagesTF1"
        value="#{bindings.regionBinding1.regionModel}"/>
    <cust:customAction action="dialog:simpleEditPoup"
        text="My Simple Edit"
        shortDesc="Simple Edit"
        location="menu"
        rendered="#{!changeModeBean.inEditMode}"
        icon="/adf/pe/images/editproperties_ena.png"
        actionComponent="SDFCustomActionCmd_simpleEditPoup"/>
</cust:showDetailFrame>
```

**Note:** A custom action that is launched in a popup dialog is sent as a new request to server. If you are using an Oracle 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.

### 9.3.4 What Happens at Runtime

Custom actions corresponding to navigation outcomes of the current task flow view 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.

### 9.4 Enabling 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 7.1.1, "How to Create a Customizable Page."

2. Add a Custom Action component from the Oracle Composer tag library as a child of the Show Detail Frame component, and set the Action and Text attributes to ViewGadget and View Gadget respectively.

3. Add two more Custom Action components to the Show Detail Frame:
   - Set the Action and Text attributes for the first component to EditSettings and Edit Settings respectively.
   - Set the Action and Text attributes for the second component to AboutGadget and About Gadget respectively.

4. Save and run `MyPage.jspx`. The `view_gadget` page fragment is rendered in the Show Detail Frame component (named My Gadget) on the page. The Actions menu displays the About Gadget and Edit Settings options. Click About Gadget to navigate to the about_gadget fragment. Note that the Actions menu now displays the navigation rules for the other two fragments (Figure 9–5).

**Figure 9–5 Custom Actions on a Show Detail Frame Enclosing a Task Flow**
To see this in action, look at the sample application, CustomActions.jws, on the Oracle WebCenter Suite 11g Demonstrations and Samples page on the Oracle Technology Network (OTN):

http://webcenter.oracle.com

9.5 Enabling Custom Actions on Task Flows that Display in the Component Navigator

The component navigator in Oracle Composer Source view provides an option to zoom into a task flow and display the components on its page or fragment, as shown in Figure 9–6.

Figure 9–6   Edit Action on a Task Flow Instance

Users can zoom in, edit the page or fragment, and zoom out of the task flow to navigate back to the page containing the task flow. In addition to the Edit Task Flow and Close links displayed next to a task flow name, you can configure your application to also display custom actions next to a task flow name, as shown in Figure 9–7.
This section describes the procedure to enable custom actions on task flows in the component navigator. It contains the following sections:

- **Section 9.5.1, "How to Configure Custom Actions in the Component Navigator"**
- **Section 9.5.2, "What Happens at Runtime"**

### 9.5.1 How to Configure Custom Actions in the Component Navigator

To display custom actions against a task flow in the component navigator, you must create a Java bean that defines the custom action behavior and call this bean from the application page containing the task flow. This section describes the steps to do so in detail. It contains the following subsections:

- **Section 9.5.1.1, "Defining the Logic for the Custom Action"**
- **Section 9.5.1.2, "Creating a JSPX Page Containing the Custom Action"**
- **Section 9.5.1.3, "Calling the JSPX Page from the Application Page Containing the Task Flow"**

#### 9.5.1.1 Defining the Logic for the Custom Action

To begin with, you must decide on the custom action you want to provide to users and create a Java bean with the logic to be implemented when a user selects the custom action. This section describes the steps to implement simple logic to display a message to users on clicking the custom action. The sample bean also contains the code to display the Close/Edit Task Flow links along with the custom link.

To create a Java bean:

1. From the **File** menu, choose **New**.
2. In the New Gallery dialog, expand **General**, select **Java**, then **Java Class**, and click **OK**.
3. In the Create Java Class dialog, enter **BackingBean** in the **Name** field and click **OK**.

The **BackingBean.java** file displays in Source view.
4. Import the required libraries as follows:

```java
import javax.faces.application.FacesMessage;
import javax.faces.context.FacesContext;
import javax.faces.event.ActionEvent;
import oracle.adf.view.page.editor.sourceview.ComponentInfo;
```

5. Add the following code:

```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.

9.5.1.2 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 Oracle Composer.

To create the JSPX page:

1. In your WebCenter 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
<?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">
```
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="]"> dc_ot4"/>
   ```

The source of the `customLink.jspx` page is as follows:

```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">
  <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="]"> dc_ot2"/>
  </af:panelGroupLayout>
  <af:xmlContent>
    <component xmlns="http://xmlns.oracle.com/adf/faces/rich/component">
      <facet>
        <facet-name>zoom</facet-name>
      </facet>
    </component>
  </af:xmlContent>
</af:componentDef>"
5. Save the JSPX page.

9.5.1.3 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
<?xml version='1.0' encoding='UTF-8'?>
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version="2.1"
    xmlns:f="http://java.sun.com/jsf/core"
    xmlns:h="http://java.sun.com/jsf/html"
    xmlns:af="http://xmlns.oracle.com/adf/faces/rich"
    xmlns:pe="http://xmlns.oracle.com/adf/pageeditor"
    xmlns:cust="http://xmlns.oracle.com/adf/faces/customizable">
    <jsp:directive.page contentType="text/html;charset=UTF-8"/>
    <f:view>
        <af:document id="d1">
            <af:form id="f1">
                <af:panelStretchLayout topHeight="50px" id="psl1">
                    <f:facet name="top">
                        <pe:changeModeLink id="cml1"/>
                    </f:facet>
                    <f:facet name="center">
                        <!-- id="af_one_column_header_stretched" -->
                        <pe:pageCustomizable id="pageCustomizable1">
                            <cust:panelCustomizable id="panelCustomizable1" layout="scroll">
                                <f:facet name="center">
                                    <pe:pageCustomizable id="pageCustomizable1">
                                        <cust:panelCustomizable id="panelCustomizable1" layout="scroll">
                                            <f:facet name="center">
                                                <pe:pageCustomizable id="pageCustomizable1">
                                                    <cust:panelCustomizable id="panelCustomizable1" layout="scroll">
                                                        <f:facet name="editor">
                                                            <pe:pageEditorPanel id="pep1"/>
                                                        </f:facet>
                                                    </cust:panelCustomizable>
                                                </pe:pageCustomizable>
                                            </f:facet>
                                        </cust:panelCustomizable>
                                    </pe:pageCustomizable>
                                </f:facet>
                            </cust:panelCustomizable>
                        </pe:pageCustomizable>
                    </f:facet>
                </af:panelStretchLayout>
            </af:form>
        </af:document>
    </f:view>
</jsp:root>
```

The task flow, taskflowdefinition1, contains the following view.jsff fragment:

```xml
<?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 Oracle 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:
The `sourceViewNodeAction` attribute can take the name of a JSPX file or an EL value that evaluates to a JSPX file name.

### 9.5.2 What Happens at Runtime

When you run `MyPage.jspx` to the browser and open the page in Oracle Composer Source view, the component navigator displays a Test link and an Edit Task Flow link next to each task flow instance on the page. Figure 9–8 shows `MyPage.jspx` in the component navigator, with the Test link highlighted. Clicking the Test link displays a sample message to the user.

![Figure 9–8 Custom Action Displayed Against a Task Flow in the Component Navigator](image)

### 9.6 Creating Event-Enabled Task Flows

Oracle Composer provides a means of contextually wiring task flow events. You can wire a contextual event to an action handler to enable the passing of values from a producer component to a consumer component when the event is triggered on the producer.

For events to be available at runtime, event capability must be included in the task flow at design time. When you add event-enabled task flows to your customizable page, each task flow's Component Properties dialog includes an Events tab, where much of the wiring activity takes place. For information about including event capabilities, see "Creating Contextual Events" in Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework.

### 9.7 Configuring an Application Page to Display in Source View by Default

When a user invokes Oracle Composer by clicking the Edit button or link on a page, the page opens in Design view by default. If your business so requires, you can configure your application pages to display in Oracle Composer Source view by default.
9.7.1 How to Configure an Application Page to Display in Source View by Default

The Edit Mode view attribute on the Page Customizable component enables you to select either Design view or Source view as the default page view in Edit mode.

To open a page in Source view by default:

1. Open your JSPX page in JDeveloper.
2. Select the Page Customizable component in the Structure Navigator.
3. In the Property Inspector, click the Edit Mode view attribute dropdown list and select source.

**Note:** To open the page in Source view for selected users only, you can specify an EL value for the Edit Mode view attribute.

4. Save the page.

Setting the Source View Position and Size

By default, the component navigator in Source view is displayed at the top of the page, and its default height is 200 pixels. You can specify a different position or size for the component navigator using the sourceViewPosition and sourceViewSize attributes respectively on the Page Customizable component. For more information about these attributes, see Section B.1.1, "Page Customizable."

9.7.2 What Happens at Runtime

When a user clicks the Edit link or button on the page, Oracle Composer opens the page in Source view.

9.8 Disabling Source View for the Application

By default, Source view is enabled in Oracle Composer-enabled pages. You can disable Source 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 9.8.1, "How to Disable Source View"
- Section 9.8.2, "What Happens at Runtime"

**Note:** For information about the editing capabilities in Source view, see Section 5.5, "Editing Capabilities in Source View in Page Edit Mode."

9.8.1 How to Disable Source View

You can disable Source view by setting the enable-source-view entry to false in the application's adf-config.xml file.

**Note:** For information about the Oracle Composer-specific configurations you can make in adf-config.xml, see Section B.2.2, "adf-config.xml."
To disable Source view:

1. Open the application's adf-config.xml file, located in the ADF META-INF folder under Descriptors in the Application Resources panel.

2. Add the enable-source-view property and set its value to false, as shown in Example 9–5.

   **Example 9–5  Disabling Source View in adf-config.xml**

   ```xml
   <page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">
     <enable-source-view>false</enable-source-view>
   </page-editor-config>
   ``

3. Save the file.

### 9.8.2 What Happens at Runtime

At runtime, when you switch to page Edit mode, the page is rendered in Design view and the View menu is not displayed to the user, as shown in Figure 9–9.

**Figure 9–9  Design View of Page without View Menu**

---

### 9.9 Applying Styles to Components

You can change the look and feel of Oracle 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 WebCenter application. For more information about style selectors and skins, see Chapter 19 "Customizing the Appearance Using Styles and Skins" in *Oracle Fusion Middleware Web User Interface Developer’s Guide for Oracle Application Development Framework*.

- Use JDeveloper style properties to specify style information through the Property Inspector. For more information, see Understanding `contentStyle` and `inlineStyle` Properties.
You can adjust the look and feel of Page Customizable, Panel Customizable, Layout Customizable, and Show Detail Frame components at design time by changing the style-related properties `inlineStyle` and `styleClass`.

Show Detail Frame components have another associated style property, `contentStyle`, which defines the style for the content inside the component. The `styleClass` property enables you to select an extra style from the skin, whereas the `inlineStyle` and `contentStyle` properties override the comparable styles specified in the application skin for that particular instance of the component.

The `inlineStyle` property overrides `styleClass`. Additionally, properties set on a component instance affect only that instance of the component. Other component instances in the application are not affected.

**Note:** The background property is also useful in adjusting the look and feel of Show Detail Frame components. It is used to provide a dark, medium, or light color scheme for the component instance. Unlike `inlineStyle`, `contentStyle`, and `styleClass` properties, the background property works with skins. Depending on which value is specified for a component instance's background property, the skin applies the relevant style.

**Understanding contentStyle and inlineStyle Properties**

The style properties `inlineStyle` and `contentStyle` are alike in the types of attributes they support. They differ in their range of influence. While `inlineStyle` provides style information for the entire component, `contentStyle` provides style information only for component content. The `contentStyle` property is available to Show Detail Frame components but not to Panel Customizable components.

The `inlineStyle` property applies CSS to the root of the component, that is, the topmost DOM element. It does not override styles on child elements that are picking up color, font, and so on from a skin. For example, if a component header is skinned, then setting `inlineStyle` does not affect the component header. The `contentStyle` applies CSS to the DOM element that surrounds the content part of the component. In a Show Detail Frame, content refers to the area below the header.

On component content, the value specified for `contentStyle` takes precedence over the value specified for `inlineStyle`. Additionally, `contentStyle` on a component instance takes precedence over both `inlineStyle` and the `contentStyle` values of a parent component (such as a portlet nested in a Panel Customizable component).
9.10 Configuring the Persistence Change Manager

To persist user personalizations and customizations across sessions, your application must be configured to use the change persistence framework. When you add Oracle Composer components to a customizable page, Oracle Composer configures the application to use ComposerChangeManager so that changes made to a page at runtime are persisted appropriately.

The default change persistence behavior in release 11.1.1.1 applications is to save Edit mode changes to MDS. View mode changes are not persisted and are available only for that session. To persist View mode changes also to MDS so that they are available across sessions, you can configure your 11.1.1.1 applications to use ComposerChangeManager.

This section describes the default change manager configuration in new WebCenter applications containing Oracle Composer-enabled pages and explains how to configure existing release 11.1.1.1 applications to use ComposerChangeManager. It contains the following sections:

- Section 9.10.1, "Overview of the Default Change Manager Configuration"
- Section 9.10.2, "How to Configure ComposerChangeManager in Existing 11.1.1.1 Applications"

9.10.1 Overview of the Default Change Manager Configuration

The first time you add an Oracle Composer component to your application page, Oracle Composer does the following to enable change persistence:

- Adds the CHANGE_PERSISTENCE context parameter to the web.xml file, and sets the value to ComposerChangeManager. This context parameter registers the ChangeManager class to be used to handle persistence, as shown in the following example:

  `<context-param>`
Configuring the Persistence Change Manager

Sets the persistent-change-manager element in the adf-config.xml file to the MDSDocumentChangeManager. Oracle 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 uri="http://xmlns.oracle.com/adf/faces/customizable">
    <tag name="showDetailFrame">
      <persist-operations>all</persist-operations>
      <attribute name="expansionMode">true</attribute>
    </tag>
    <attribute name="contentStyle">true</attribute>
  </tag>
  <tag name="panelCustomizable">
    <persist-operations>all</persist-operations>
  </tag>
  ...
</taglib-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 11.4.1, "How to Define Change Persistence at the Component Level."

For a list of ADF Faces components and attributes that are implicitly persisted, see the chapter titled "Allowing User Customizations at Runtime" in the Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework.
ComposerChangeManager

A ChangeManager class is required for persisting customizations performed by end users. ComposerChangeManager handles change persistence both within a session and across sessions (to MDS). It delegates a users changes to the appropriate change manager as follows:

- View mode changes are routed to FilteredPersistenceChangeManager to ensure that implicit customizations, such as column resizing and header collapse, work according to the filter rules configured in the application's adf-config.xml file.
- Edit mode changes are routed to MDSDocumentChangeManager to ensure that both implicit and explicit customizations are always persisted and available across sessions.

**Notes:** When you add Oracle Composer components to your application page, the option to enable user customizations for the duration of the session (in the Project Properties dialog) is disabled. This is because changes in Oracle Composer-enabled application pages must be persisted across sessions, to MDS.

### 9.10.2 How to Configure ComposerChangeManager in Existing 11.1.1.1 Applications

Release 11.1.1.1 WebCenter applications containing Oracle Composer-enabled pages use MDSDocumentChangeManager for persisting Edit mode changes by default. Since Oracle Composer now provides ComposerChangeManager to persist user personalizations and customizations, you can configure your existing applications to use ComposerChangeManager so that user personalizations (in View mode) are also persisted to MDS.

This section contains the following sections:

- **Section 9.10.2.1, "Updating CHANGE_PERSISTENCE Context Parameter in the web.xml File"**
- **Section 9.10.2.2, "Adding Relevant Entries to the adf-config.xml File"**

#### 9.10.2.1 Updating CHANGE_PERSISTENCE Context Parameter in the web.xml File

You must update the CHANGE_PERSISTENCE context parameter to ComposerChangeManager.

To modify the CHANGE_PERSISTENCE parameter:

1. Open your application’s web.xml file from the Application_Root\Project_Name\public_html\WEB-INF directory.
2. Locate org.apache.myfaces.trinidad.CHANGE_PERSISTENCE and update it to the following:

   ```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>
   ```
9.10.2.2 Adding Relevant Entries to the adf-config.xml File

You must configure change persistence in the adf-faces-config section of your application's adf-config.xml file.

To configure change persistence in the adf-config.xml file:

1. Open the adf-config.xml file.

   The adf-config.xml file is located in the ADF META-INF folder under Descriptors in the Application Resources panel.

2. Add the following code under the adf-config section:

   ```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>
   <taglib-config>
       <taglib uri="http://xmlns.oracle.com/adf/faces/customizable">
           <tag name="showDetailFrame">
               <persist-operations>all</persist-operations>
               <attribute name="expansionMode">
                   <persist-changes>true</Persist-changes>
               </attribute>
               <attribute name="contentStyle">
                   <persist-changes>true</Persist-changes>
               </attribute>
           </tag>
           <tag name="panelCustomizable">
               <persist-operations>all</persist-operations>
           </tag>
       </taglib>
       <taglib uri="http://xmlns.oracle.com/adf/pageeditor">
           <tag name="layoutCustomizable">
               <persist-operations>all</persist-operations>
               <attribute name="type">
                   <Persist-changes>true</Persist-changes>
               </attribute>
           </tag>
       </taglib>
   </taglib-config>
   </adf-faces-config>
   
   This ensures that MDSDocumentChangeManager is used to persist View mode changes to component tags and attributes listed in the taglib-config section.

   By default, the following View mode changes are persisted:

   - **Panel Customizable component**: Rearranging of components within the container or across containers.
   - **Show Detail Frame component**: Expanding, collapsing, and resizing of components.

   You can enable change persistence for other tags and attributes by defining them in this section and setting the persist-changes attribute to true. For more information, see Section 11.4.1, "How to Define Change Persistence at the Component Level."

3. Save the adf-config.xml file.
Configuring Runtime Resource String Editing

9.11 Configuring Runtime Resource String Editing

Component properties edited in Oracle 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 Oracle 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, Oracle 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 Oracle 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.

Release 11.1.1.1 WebCenter applications containing Oracle Composer-enabled pages do not support editing of resource strings at runtime. You can enable resource string editing in your existing application by performing a few simple configurations. This section describes the steps to enable resource string editing in an existing release 11.1.1.1 application. It contains the following sections:

- Section 9.11.1, "Overview of the Resource String Editor"
- Section 9.11.2, "How to Enable the Resource String Editor in Your Application"
- Section 9.11.3, "How to Configure the Override Bundle"
- Section 9.11.4, "What Happens at Runtime"

9.11.1 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 5.4.6, "Edit Resource Strings."

Changes made to resource strings in Oracle 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 Oracle Composer libraries. This bundle becomes available to the application when you add Oracle Composer components to the page at design time. At runtime, Oracle 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, Oracle 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**
- Oracle 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.

### 9.11.2 How to Enable the Resource String Editor in Your Application

To enable resource string editing in your application, you must add a resource-string-editor element in the page-editor-config section of your application's adf-config.xml file, as follows:

```xml
<pe:page-editor-config>
  <resource-string-editor>
    <enabled>true</enabled>
  </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.
9.11.3 How to Configure the Override Bundle

If you enable resource string editing in your application, you must configure your application to use the override bundle, `oracle.adf.view.page.editor.resource.ComposerOverrideBundle.xlf`, which is available in the `pageeditor.jar` file. This override bundle is used for reading and writing to the application override bundle when users create or edit resource strings at runtime.

To configure the override bundle:

1. Open your application in JDeveloper.
2. Form the Application menu, select Application Properties.
3. Select Resource Bundles in the left panel of the Application Properties dialog (Figure 9–11).

![Figure 9–11 Application Properties Dialog](image)

4. Click the **Add** button.
5. In the File Name field in the Select Resource bundle dialog, enter `oracle.adf.view.page.editor.resource.ComposerOverrideBundle.xlf`, as shown in Figure 9–12, and click **Open**.

**Note:** If you have problems locating the override bundle, see the last item in Section 9.12, "Troubleshooting Problems with Advanced Oracle Composer Configurations."
6. In the Application Properties dialog, select the **Override** check box. This ensures that the file is available to users at runtime.

7. Click **OK**.

Repeat the steps in this section to expose any of the resource bundles in your application to users at runtime.

---

**Note:** To make a resource bundle accessible for runtime read and write operations, you must ensure that the Override option is selected for that bundle in the Application Properties dialog.

You can make properties bundles, list resource bundles, and XLIFF resource bundles accessible at runtime.

---

### 9.11.4 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 5.4.6, "Edit Resource Strings."

---

### 9.12 Troubleshooting Problems with Advanced Oracle Composer Configurations

This section provides information to assist you in troubleshooting problems you may encounter on performing advanced Oracle Composer configurations.
For information about configuring logging, see "Configuring ADF Logging for Oracle Composer".

**Problem**

You added a global- or instance-level Custom Action. However, it neither displays on the chrome or in the Action menu on the Show Detail Frame.

**Solution**

Ensure the following:

- The first child of the Show Detail Frame must be af:region.
- The view currently displayed on the task flow must have an outcome in its task flow definition file that matches the action of Custom Action.
- If the action has the prefix dialog:, the outcome in the task flow definition must also have the same prefix.

For information, see Section 9.3, "Enabling Custom Actions on a Show Detail Frame Enclosing a Task Flow."

**Problem**

Oracle Composer is not working on ADF application pages. Errors are reported in the logs when using Oracle Composer's Resource Catalog or Component Properties dialog. None of the customizations are saved. Objects are not getting added from the Resource Catalog.

**Solution**

Oracle 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 Oracle Composer, the value of CHANGE_PERSISTENCE is changed to oracle.adf.view.page.editor.change.ComposerChangeManager.

**Problem**

You are unable to configure resource string editing in your application as you cannot find the override bundle, ComposerOverrideBundle.xlf, while performing the steps in Section 9.11, "Configuring Runtime Resource String Editing."

**Solution**

The Oracle Composer override bundle becomes available to the application only when you add Oracle Composer components to the page. Therefore, ensure that you have created a JSPX page and added Oracle Composer components to the page before you configure resource string editing.

If you still cannot locate the override bundle while performing the steps in Section 9.11, "Configuring Runtime Resource String Editing," add an entry in the application's adf-config.xml file, as shown in the following example, and then perform the configuration:
Troubleshooting Problems with Advanced Oracle Composer Configurations

<adf-resourcebundle-config
xmlns="http://xmlns.oracle.com/adf/resourcebundle/config">
  <applicationBundleName>
    resourcebundles/xliffBundles/FirstRRAppOverrideBundle
  </applicationBundleName>
  <bundleList>
    <bundleId override="true">
      oracle.adf.view.page.editor.resource.ComposerOverrideBundle
    </bundleId>
  </bundleList>
</adf-resourcebundle-config>
This chapter describes how to create customization layers for saving changes made to the page at runtime. It also describes how to enable Oracle Composer sandbox creation to provide users the option to preview their changes before saving them to the back end. It contains the following sections:

- Section 10.1, "Introduction to MDS"
- Section 10.2, "Using Oracle Composer Sandbox"
- Section 10.3, "Adding Customization Layers to View and Edit Modes: Example"

10.1 Introduction to MDS

Most industries customize their enterprise applications to serve different audiences and domains. Problems can arise when an application is modified at the site level. For example, upgrading an application with site-level customizations may lead to data loss or data-merge errors. Consequently, a new version of the application cannot be deployed until all merge conflicts are reconciled.

In the metadata domain, MDS provides the customization feature to address such problems. The customization feature allows for the creation of nonintrusive customization layers that are applied on top of the base application definitions. Customization layers, or layered changes, are described in their own documents and are stored separately from the base application definition. At runtime, applicable customizations are loaded from the metadata store and layered over the base metadata definition to produce the desired effect. Product upgrades and patches affect only the base metadata definition, so customizations continue to function properly.

**Customization Layers**

MDS enables clients to specify multiple customization types. For example, you can add customizations to runtime modes, application or user roles, application states, or any client specified criteria. Each such customization type is called a *customization layer* and is depicted using a *CustomizationClass*. A *CustomizationClass* is the interface MDS uses to identify the customization layer to be overlaid on the base definition. For example, you can configure your application to save customizations based on the departments to which users belong. You can create a customization layer called *DepartmentCC* in which 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 personalizations in one layer and Edit mode changes as 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 10.3, "Adding Customization Layers to View and Edit Modes: Example."

**Oracle Composer Sandbox**

Typically, in a custom WebCenter application, runtime customizations are saved immediately in the JDEV_HOME/jdev/system_ directory/o.mds.dt/adr/s/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 Oracle Composer to create a sandbox if you are using a database repository to store customizations. For information about creating a sandbox, see Section 10.2, "Using Oracle Composer Sandbox."

**10.2 Using Oracle 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 Oracle 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 Oracle Composer.

Personalizations 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 Oracle Composer toolbar in both Design view and Source view of the page.

This section discusses the steps you can take to enable sandbox creation and describes runtime behavior of a sandbox-enabled application. It contains the following subsections:

- Section 10.2.1, "How to Enable Oracle Composer Sandbox Creation"
- Section 10.2.2, "What Happens at Runtime"
- Section 10.2.3, "How to Disable Sandbox for an Application"
- Section 10.2.4, "How to Destroy Stale Sandboxes"
- Section 10.2.5, "How to Enable Application Sandbox Creation"
- Section 10.2.6, "What Happens When You Enable Application Sandbox Creation"
10.2.1 How to Enable Oracle Composer Sandbox Creation

You can enable a sandbox only if your application uses a database store. Therefore, you must ensure that you have configured a database store before performing the steps in this section. For information about setting up a database store, see Oracle Fusion Middleware Configuring and Managing JDBC for Oracle WebLogic Server.

This section describes how to enable the creation of an Oracle Composer sandbox. It contains the following subsections:

- Section 10.2.1.1, "Updating Your Application’s adf-config.xml File"
- Section 10.2.1.2, "Updating Your Application’s web.xml File"
- Section 10.2.1.3, "Selecting a Database Store for Your Application"

10.2.1.1 Updating Your Application’s adf-config.xml File

You must update the adf-config.xml file to define and configure namespaces for all the metadata for which you want to support sandbox creation. This section describes how.

Note: For information about the Oracle 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 10–1.

Example 10–1 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 <sandbox-namespaces> element within the <page-editor-config> section and add individual <namespace> tags for all namespaces for which you want to enable sandbox creation, as show in Example 10–2.

Example 10–2 Configuring Namespaces for Sandbox Creation

```xml
<sandbox-namespaces>
  <namespace path="/pages"/>
  <namespace path="/pageDefs"/>
</sandbox-namespaces>
```

4. Save the adf-config.xml file.
10.2.1.2 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 so that all requests are routed through these filters first:
   
   ```
   <filter>
   <filter-name>WebCenterComposerFilter</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 10–3.

   **Example 10–3 Filter Mappings for Composer-Specific Filter**

   ```
   <filter-mapping>
   <filter-name>WebCenterComposerFilter</filter-name>
   <url-pattern>/*</url-pattern>
   </filter-mapping>
   <filter-mapping>
   <filter-name>WebCenterComposerFilter</filter-name>
   <url-pattern>/faces/*</url-pattern>
   <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.

---

Using Oracle Composer Sandbox

**10.2.1.3 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 38.3.3, "Deploying a Custom WebCenter Application to a Managed Server."

**10.2.2 What Happens at Runtime**

When a user switches to page Edit mode, the presence of a Save button on the Oracle 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 Oracle 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 10–1.

*Figure 10–1 Confirm Close Dialog in Oracle Composer*

![Confirm Close Dialog](image)

Users can select from the following options:

* Click Save to commit the sandbox and close Oracle Composer.
* Click Don't Save to destroy the sandbox and close Oracle Composer.
* Click **Cancel** to close the dialog and return to Oracle 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 10–2**.

**Figure 10–2 Page Concurrency Message**

If two or more users have zoomed into a task flow at the same time in the same customization layer and are editing it, then a concurrency message appears, as shown in **Figure 10–3**.

**Figure 10–3 Task Flow Concurrency Message**

---

**Note:** If a user zoomed into a task flow, made some customizations, and zoomed out of the task flow, then the task flow concurrency message continues to display to other users until that user saves the customizations.

---

Changes that are saved last overwrite prior changes. For example, if users A and B are editing a page or task flow simultaneously, then concurrency issues are handled as follows:
• If A saves the page or task flow first, then A’s changes are committed to MDS. Later, when B saves the page or task flow, A’s changes are overwritten with B’s changes.

• If A deletes a component while B is trying to personalize (say move) that same component in View mode, a WebCenter error page is displayed to B. B has to simply navigate back to the original page. The deleted component does not appear, and B can continue working on other components.

10.2.3 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 <page-editor-config> element, add the <disable-sandbox> attribute and set it to true.

3. Save adf-config.xml.

10.2.4 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.

• 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 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_renters page Edit mode.

To configure WebCenterComposerSessionListener:


2. Define a new listener, WebCenterComposerSessionListener, as shown in the following example:

   <listener>
     <description>Oracle Composer Http Session Listener</description>
     <display-name>Oracle Composer Http Session Listener</display-name>
     <listener-class>
       oracle.adf.view.page.editor.webapp.WebCenterComposerSessionListener
     </listener-class>
   </listener>

For information about the Oracle Composer-specific configurations you can make in web.xml, see Section B.2.3, "web.xml."

### 10.2.5 How to Enable Application Sandbox Creation

Consider an example where you have created an application that displays a banner at the top of the page, a sidebar on the left with options to manage pages, and the page at the center of the browser window. You may want to enable users to edit the page in Oracle Composer and edit the sidebar using some other mechanism such as updating an XML file. Enabling Oracle Composer sandbox in such a case only saves changes to page content but not the sidebar. If you enable the application sandbox, then all changes made at runtime are saved at one time, to one location. If you enable sandbox creation at the application level, then Oracle Composer does not create its own sandbox when page Edit mode is invoked. Instead, the application sandbox is used. That is, a user interface provided by the application is used for committing and destroying the sandbox. To enable the application to provide a sandbox for Oracle Composer, you must first perform a set of configurations.

Another example where it is useful to enable the application sandbox is if you want users to edit page content, but ensure that the page is saved only on approval from the manager. By using an application sandbox, changes made by a user are in the sandbox until the manager approves the changes. The page is saved on approval.

To enable an application sandbox for use in Oracle Composer:

1. **Implement a ComposerSessionOptionsFactory class to provide MDSSessionOptions for each request.**

   For information about performing this task, see Section 10.3.4, "How to Implement the ComposerSessionOptionsFactory Class."

2. **Register your implementation with Oracle Composer.**

   For information about performing this task, see Section 10.3.5, "How to Register the Implementation with Oracle Composer."

3. **Configure WebCenterComposerFilter.**

   For information about performing this task, Section 10.3.6, "How to Configure WebCenterComposerFilter."

### 10.2.6 What Happens When You Enable Application Sandbox Creation

At runtime, the application-defined user interface enables users to save their changes to the back end or close Oracle Composer without saving the changes. In this case, Oracle Composer has no control over when customizations are canceled or committed to the back end.

### 10.3 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
Adding Customization Layers to View and Edit Modes: Example

identify the customization layer to be overlaid on the base definition. See Section 10.3.3, "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 Oracle 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 personalization capabilities to all users and customization capabilities to selective users. To enable customizations in the Edit mode, the site tip layer is added. To enable personalization in the View mode, the user tip layer is added. By default, the user tip layer is applied on top of the site tip layer. The user tip layer stores all customizations made in the View mode in a specific location created for the user who made them. Such changes are visible only to that user. The site tip layer stores all customizations made in the Edit mode and are visible to all users.

To enable tip layers at runtime, Oracle 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 personalizations (View mode) or 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 10.3.1, "How to Add Oracle Composer to a JSF Page"
- Section 10.3.2, "How to Create a Custom SiteCC Tip Layer"
- Section 10.3.3, "How to Create a Custom UserCC Tip Layer"
- Section 10.3.4, "How to Implement the ComposerSessionOptionsFactory Class"
- Section 10.3.5, "How to Register the Implementation with Oracle Composer"
- Section 10.3.6, "How to Configure WebCenterComposerFilter"
- Section 10.3.7, "How to Redirect the Servlet to Enable Switch Between MDS Customization Layers"
- Section 10.3.8, "What Happens at Runtime"

### 10.3.1 How to Add Oracle 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 Oracle 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.
Adding Customization Layers to View and Edit Modes: Example

To add Oracle Composer to a JSF Page:

1. Open the JSPX page and select Oracle 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.

### 10.3.2 How to Create a Custom SiteCC Tip Layer

This section describes how to create a custom SiteCC tip layer, site, in which all site-level customizations performed in Edit mode are stored. In this sample application, site-level customizations are stored in the 
/mds/mdsys/cust/site/webcenter/pagename.jspx.xml directory.

To create the site tip layer:

1. From the File menu, choose New.
2. In the New Gallery dialog, expand General, select Java, then Java Class, and click OK.
   
   The Create Java Class dialog opens.
3. In the Name field, enter SiteCC.
4. In the Extends field, enter CustomizationClass.
   
   This imports oracle.mds.cust.CustomizationClass.
5. Click OK.
   
   The SiteCC.java file is rendered in the Source view.
6. To implement the `getCacheHint`, `getName`, and `getValue` methods, double-click the bulb on the left corner and select **Implement Methods**.

7. In the Implement Methods dialog, click **OK**.

8. In the Source view, press **Enter** after `public class SiteCC extends CustomizationClass`, and add the following string:

   ```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;
   ```

9. 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 };  
       }
   }
   ```

10. Save the `SiteCC.java` file.

### 10.3.3 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 personalizations 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 personalizations performed in View mode are saved in the `/mds/mdssys/cust/user/scott/pagename.jspx.xml` directory.

---

**Note:** For illustration purpose, this example describes a simple scenario where a UserCC tip layer is created for just one user, scott. In a real life application, you can configure the UserCC layer to return the name of the user who requested the page.
To create the user tip layer:

1. From the File menu, choose New.
2. In the New Gallery dialog, expand General, select Java, then Java Class, and click OK.
   The Create Java Class dialog opens.
3. In the Name field, enter UserCC.
4. In the Extends field, enter CustomizationClass.
   This imports the oracle.mds.cust.CustomizationClass.
5. Click OK.
   The UserCC.java file displays in the Source view.
6. To implement the getCacheHint, getName, and getValue methods, double-click the bulb on the left corner and select Implement Methods.
7. In the Implement Methods dialog, click OK.
8. In the Source view, press Enter after public class UserCC extends CustomizationClass, and add the following string:

   ```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:
import oracle.mds.core.MetadataObject;
import oracle.mds.core.RestrictedSession;
import oracle.mds.cust.CacheHint;

9. 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};
     }
   }
   ```

10. Save the UserCC.java file.
10.3.4 How to Implement the ComposerSessionOptionsFactory Class

In this section, the ComposerSessionOptionsFactory class provided by the ADF is implemented to supply MDS SessionOptions for each HTTP request.

To implement the ComposerSessionOptionsFactory class:

1. From the File menu, choose New.
2. In the New Gallery dialog, expand General, select Java, then Java Class, and click OK.

The Create Java Class dialog opens.

3. In the Name field, enter AppsSessionOptionsFactoryImpl.
4. Click OK.

The AppsSessionOptionsFactoryImpl.java file is rendered in the Source view.

5. Import the following libraries:
   import oracle.adf.view.page.editor.mds.ComposerSessionOptionsFactory;
   import oracle.adf.view.page.editor.mode.ModeContext;
   import oracle.mds.config.CustClassListMapping;
   import oracle.mds.config.CustConfig;
   import oracle.mds.core.SessionOptions;
   import oracle.mds.cust.CustClassList;
   import oracle.mds.cust.CustomizationClass;

6. Add the following code to implement the ComposerSessionOptionsFactory class and provide SessionOptions:

   public class AppsSessionOptionsFactoryImpl
   implements ComposerSessionOptionsFactory
   {

   public SessionOptions createSessionOptions(SessionOptions
      defaultSessionOptions, String mode)
   {
       CustomizationClass[] custLayer;
       CustConfig custConfig = null;

       if (ModeContext.EDIT_MODE.equals(mode))
       {
           //Mode is Edit, change to SiteCC
           custLayer = EDIT_LAYER;
       }
       else
       {
           //Mode is View, change to UserCC + SiteCC
           custLayer = VIEW_LAYER;
       }

       try
       {
           CustClassList custClassList = new CustClassList(custLayer);
           CustClassListMapping custClassListMapping =
               new CustClassListMapping("/", null, null, custClassList);
           custConfig = new CustConfig(new CustClassListMapping[]
               { custClassListMapping });
       }
       catch (Exception e)
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());
}

private static final CustomizationClass[] EDIT_LAYER =
    new CustomizationClass[]{ new SiteCC() };

private static final CustomizationClass[] VIEW_LAYER =
    new CustomizationClass[]{ new SiteCC(), new UserCC() };

10.3.5 How to Register the Implementation with Oracle Composer

For the site and user customization layers to function, you must register the
ComposerSessionOptionsFactory class with Oracle 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:
<page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">
    <session-options-factory>view.AppsSessionOptionsFactoryImpl</session-options-factory>
</page-editor-config>

10.3.6 How to Configure WebCenterComposerFilter

You must configure the WebCenterComposerFilter filter in the web.xml file
located in the ViewController\public_html\WEB-INF folder in your application
directory. This filter registers Oracle 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 WebCenter application and gets the SessionOptions
with the new customized layer. If you have not set the Sandbox or VersionContext in the SessionOptions, then Oracle Composer sets its own Sandbox and returns it to the ADF. For more information on Sandbox, see Section 10.2.1, "How to Enable Oracle Composer Sandbox Creation." The composerFilter and its filter mapping must be configured before ADFBindingFilter. For example, see the following web.xml file:

....

<!-- WebCenterComposerFilter goes here -->
<filter>
  <filter-name>composerFilter</filter-name>
</filter>

<filter-class>oracle.adf.view.page.editor.webapp.WebCenterComposerFilter</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>

....

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 Oracle Composer-specific configurations you can make in web.xml, see Section B.2.3, "web.xml."

10.3.7 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 10.3.7.1, "How to Create the AppNavigationUtils Class"
- Section 10.3.7.2, "How to Create AppCloseHandler"
- Section 10.3.7.3, "How to Register the AppCloseHandler"
- Section 10.3.7.4, "How to Create AppModeBean"

10.3.7.1 How to Create the AppNavigationUtils Class

To create the AppNavigationUtils class:

1. From the File menu, choose New.
2. In the New Gallery dialog, expand General, select Java, then Java Class, and click OK.

   The Create Java Class dialog opens.
3. In the Name field, enter AppNavigationUtils and click OK.

   The AppNavigationUtils.java file is rendered in the Source view.
4. Import the following libraries:

   ```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();
   }
   
   }
   ```

10.3.7.2 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:
   ```java
   import oracle.adf.view.page.editor.event.CloseEvent;
   import oracle.adf.view.page.editor.event.CloseListener;
   ```

5. Add the following code:
   ```java
   public class AppCloseHandler
   implements CloseListener
   {
       public void processClose(CloseEvent closeEvent)
       {
           AppNavigationUtils.redirectToSamePage();
       }
   }
   ```

10.3.7.3 How to Register the AppCloseHandler
After creating **AppCloseHandler**, you must register it in the Oracle Composer extension file, **pe_ext.xml**.
To register the event handler:

1. In the **pe_ext.xml** file, add the following entries:
   ```xml
   <event-handlers>
   <event-handler event="close">view.AppCloseHandler</event-handler>
   </event-handlers>
   ```

2. Save the file.
For more information about event handlers, see Section 8.5, "Configuring Event Handlers for Oracle Composer UI Events."

10.3.7.4 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:
   ```java
   import javax.faces.event.ActionEvent;
   import oracle.adf.view.page.editor.mode.ModeContext;
   ```

5. Add the following code:
public class AppModeBean
{
    public void edit(ActionEvent actionEvent)
    {
        ModeContext.getCurrent().setEditMode();
        AppNavigationUtils.redirectToSamePage();
    }
}

10.3.8 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 10–4.

Figure 10–4  Customized Page

Go to \mds\mdssys\cust\site\webcenter in your application directory, and open the pagename.jspx.xml file. This is where the site-level customizations 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 (personalizations) that you made to the page are stored.

10.4 Troubleshooting MDS-Related Problems with Oracle Composer

This section provides information to assist you in troubleshooting MDS-related problems you may encounter while using Oracle Composer.
For information about configuring logging, see "Configuring ADF Logging for Oracle 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 10.2, "Using Oracle Composer Sandbox."
Modifying Default Security Behavior of Oracle Composer Components

You can override the default security definitions at various levels in keeping with your business requirement. For information about the default security behavior of Oracle Composer components, see Section 5.7, "Security and Oracle Composer."

This chapter describes how to override the default security behavior of Oracle Composer components. It contains the following subsections:

- Section 11.1, "Applying Component-Level Restrictions by Defining Customization Policies"
- Section 11.2, "Applying a Component Instance-Level Customization Restriction By Using Security Roles: Example"
- Section 11.3, "Applying Tag-Level Security Using the customizationAllowed Attribute"
- Section 11.4, "Applying Attribute-Level Security"
- Section 11.5, "Applying Action-Level Restrictions on Panel Customizable and Show Detail Component Actions"
- Section 11.6, "Applying Task Flow Restrictions"

11.1 Applying Component-Level Restrictions by Defining Customization Policies

By default, the MDS restricts customization of page components, that is, users cannot customize components or their attributes at runtime. To enable 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 customization of components and their attributes. This section describes how to apply MDS type-level restrictions or instance-level restrictions on components and their attributes. It contains the following subsections:

- Section 11.1.1, "How to Define Type-Level Customization Policies"
- Section 11.1.2, "How to Define Instance-Level Customization Policies"
11.1.1 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 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 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 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 11.2.3, "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:

   Note: For information about creating customizable pages using page templates, see Section 7.1.7, "How to Create a Page Template for Creating Customizable Pages."
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
<?xml version="1.0" encoding="UTF-8" ?>
<mds-component-config version="11.1.1.000" xmlns="http://xmlns.oracle.com/mds/config">
  <type-config>
    <standalone-definitions>
      <classpath>File_Path/standalone.xml</classpath>
    </standalone-definitions>
  </type-config>
</mds-component-config>
```

3. Make the standalone files and associated XSDs available in a shared library JAR file; otherwise MDS cannot load them.

At runtime, MDS searches all mds-config instances in the META-INF directories and loads all the customization restrictions specified in standalone files.

### 11.1.2 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 11.1.1, "How to Define Type-Level Customization Policies"). To apply restrictions on a component’s attributes, perform the steps described in this section.

Instance-level policies for component attributes are defined in an RDF file. The RDF file is picked up by MDS, and the policies are implemented.

To define instance-level customization policies:

1. Create an RDF file with the same name as the JSPX file, but using the RDF extension, for example main.jspx.rdf.

Create this file in the application_home/project/public_/html/mdssys/mdx folder. Creating the RDF file in this folder structure ensures that the file is picked up by MDS and the policies defined here are implemented.
2. Include tags in the following format to apply restrictions on a component’s attribute:

```
tag_id(xmlns(tag_prefix=tag_namespace))/@tag_prefix:attribute_name
```

The following example shows how to apply a restriction on a Go Image Link component and the text attribute of a Command Button component:

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:dc="http://purl.org/dc/elements/1.1/">
    <rdf:Description rdf:about="/"
        <!-- Opens the customization on a tag, for example, an af:goImageLink whose ID is '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(xmlns(af=http://xmlns.oracle.com/adf/faces/rich))/@af: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.

### 11.2 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 Oracle 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 11.2.1, "How to Configure ADF Security"
- Section 11.2.2, "How to Define Roles and Grant Privileges in the jazn-data.xml File"
- Section 11.2.3, "How to Customize the SessionOptions Object to Include Customization Policy"
- Section 11.2.4, "How to Register the Implementation with Oracle Composer"
- Section 11.2.5, "How to Configure WebCenterComposerFilter"
- Section 11.2.6, "How to Apply an Instance-Level Customization Restriction"
- Section 11.2.7, "What Happens at Runtime"

11.2.1 How to Configure ADF Security

Before applying an instance-level customization restriction, you must secure your application using ADF security. For information, see Section 3.5.1, "How to Configure ADF Security."

11.2.2 How to Define Roles and Grant Privileges in the jazn-data.xml File

In a previous section, a customization restriction is applied to a Panel Customizable component so that, at runtime, only the user with admin role can access all the features of Panel Customizable, and therefore, can customize its attributes.

In this section, two roles, admin and customer, are created and page permissions are granted to both users. Users with admin role can customize the Panel Customizable (panelCustomizable1) component. However, because of the customization restriction you applied in an earlier section, users with customer role do not have the rights to customize panelCustomizable1, and these users cannot use all the features of panelCustomizable1.

To create roles in the jazn-data.xml file:

1. In the Application Resources panel, right-click the jazn-data.xml file and select Open.

2. Click the Users and Roles tab at the bottom of the editor to open the Users section of the file, shown in Figure 11–1.
3. Click the Application Roles tab.

4. In the Application Roles page, click the New Application Role icon in the Roles table, shown in Figure 11–2.

5. In the Name field, specify the user name, admin.
6. Optionally, in the Display Name and Description fields, specify a display name and description for the new role.

7. Repeat steps 3 through 6 to add another role, customer.

8. Click the Users tab.

9. In the Users page, create new users by clicking the New User icon and specifying credentials, and assign roles to these users by clicking the Assign Roles icon in the Assigned Roles table and selecting Assign Application Role, as shown in Figure 11–3.

10. Click the ADF Policies tab at the bottom of the page.

11. Select the Web Pages tab on the page.

12. From the list of page definitions, select the page on which you want to grant privileges, as shown in Figure 11–4.

13. In the Granted to Roles column, click the Add Application Role icon.
14. In the Select Roles dialog (Figure 11–5), select the admin and customer roles and click OK.

![Select Roles Dialog](image)

15. Select each role and grant all the permissions (View, Customize, Edit, Grant, Personalize), as shown in Figure 11–6.

![Permissions List in the jazn-data.xml File](image)

For more information about creating users and roles, see the section "Enabling ADF Security in a Fusion Web Application" in the Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework.

11.2.3 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**.
   
   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.share.ADFContext;
   import oracle.adf.share.security.SecurityContext;
   import oracle.adf.view.page.editor.mds.ComposerSessionOptionsFactory;
   import oracle.adf.view.page.editor.mode.ModeContext;
   import oracle.mds.config.CustClassListMapping;
   import oracle.mds.config.CustConfig;
   import oracle.mds.core.SessionOptions;
   import oracle.mds.cust.CustClassList;
   import oracle.mds.cust.CustomizationClass;
   import oracle.mds.cust.CustomizationPolicy;
   ```

6. Add the following code to implement the `ComposerSessionOptionsFactory` class and provide `SessionOptions`:

   ```java
   public class AppsSessionOptionsFactoryImpl
   implements ComposerSessionOptionsFactory
   {
       public AppsSessionOptionsFactoryImpl()
       {
       }
       public SessionOptions createSessionOptions(SessionOptions sessionOptions,
          String mode)
       {
           CustomizationClass[] custLayer;
           CustConfig custConfig = null;
           CustomizationPolicy cPol = null;

           if (ModeContext.EDIT_MODE.equals(mode))
           {
               // Mode is Edit, change to SiteCC
               custLayer = EDIT_LAYER;
           }
           else
           {
               // Mode is View, change to UserCC + SiteCC
               custLayer = VIEW_LAYER;
           }
           try
           {
               CustClassList custClassList = new CustClassList(custLayer);
               CustClassListMapping custClassListMapping =
                   new CustClassListMapping("/", null, null, custClassList);
               custConfig = new CustConfig(new CustClassListMapping[])
           }
   ```
### 11.2.4 How to Register the Implementation with Oracle Composer

Perform the steps described in Section 10.3.5, "How to Register the Implementation with Oracle Composer" to register the `ComposerSessionOptionsFactory` class implementation, where user roles were included in the MDS session's customization policy.

### 11.2.5 How to Configure WebCenterComposerFilter

To configure `WebCenterComposerFilter`, perform the steps described in Section 10.3.6, "How to Configure WebCenterComposerFilter."

### 11.2.6 How to Apply an Instance-Level Customization Restriction

In this section, you enable 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 Oracle 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>
   </pe:pageCustomizable>
   ``

   The design view should look like Figure 11–7.

   **Figure 11–7  Design View: Customization Allowed**

4. Add a Show Detail Frame component within the Panel Customizable and include another Panel Customizable component as a child of the Show Detail Frame component.

   **Note:** This step is for illustration purpose only as it is easier to explain runtime behavior by showing multiple components. For detailed information on how to add Oracle Composer components, see Section 3.3.2, "How to Create a Page in a Manually-Created WebCenter Application."

### 11.2.7 What Happens at Runtime

What happens at runtime is influenced by which user is logged on. To see the difference, log in as one user and then the other. For example:

In the Application Navigator, right-click the JSPX page and select Run.

Log in to the page as a user with admin role, and switch to Edit mode. The Add Content button and the Edit icon are displayed on the outer and inner Panel.
Customizable components. As a user with admin role, you can switch to Source view and access the properties of the outer Panel Customizable, as shown in Figure 11–8. That is, users with admin role can edit this component.

Figure 11–8  Edit Mode for admin

Log in to the page as a user with customer role, and switch to Edit mode. The Add Content button and the Edit icon are not available on the outer Panel Customizable. That is, this component is not editable for users with customer role. In Source view, the Add Content and Edit icons are grayed out for the component, as shown in Figure 11–9.
11.3 Applying Tag-Level Security Using the customizationAllowed Attribute

By default, a Page Customizable component enables 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 11.3.1, "How to Enable Customization on an Image Component"
- Section 11.3.2, "How to Restrict Customization on an Image Component"

For more information, see Extended Metadata and Annotation Properties in Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework.
**11.3.1 How to Enable Customization on an Image Component**

To enable customization on an Image component:

1. In the *MyPage.jspx* page that you created in Section 7.2, "Designing Editable Pages Using Oracle Composer Components: Example," add an ADF Faces Image component, for example, *brandingImage.gif*, inside the Panel Customizable called *panelCustomizable1*.

   Customization is enabled automatically on the Image component since it is nested inside a Page Customizable component.

2. Run *MyPage.jspx*.

3. Switch to Edit mode of the page.

4. From the **View** menu, select **Source** to switch to Source view of the page.

5. Select the image in the component hierarchy in the Source View panel and click the **Show the properties of Image** icon.

   In the Component Properties dialog, the properties are grayed out and cannot be edited.

**11.3.2 How to Restrict Customization on an Image Component**

You can restrict customization of an image on the page by setting the `customizationAllowed` attribute to `false` on the image component.

To restrict customization on the Image Component:

1. In Oracle JDeveloper, select the Image component that you added in the previous section, and in the Property Inspector set the value for `customizationAllowed` to `false`.

2. Run *MyPage.jspx*.

3. Switch to page Edit mode.

4. From the **View** menu, select **Source** to switch to the Source view of the page.

5. Select the image in the component hierarchy in the Source View panel, and click the **Show the properties of Image** icon.

   In the Component Properties dialog, the properties are grayed out and cannot be edited.

**11.4 Applying Attribute-Level Security**

You can choose whether specific component attributes can be customized at runtime. Additionally, you can specify whether the changes made to an attribute must be persisted to a persistence store, such as MDS. You can apply attribute-level security in the following ways:

- By defining property filters
- By persisting changes at the component level

For information about applying property filters, see Section 8.7.1, "How to Define Property Filters."

This section describes how to persist changes at the component level. It contains the following subsections:
11.4.1 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.

Example 11–1 shows a sample adf-config.xml file with persist-changes attributes defined for some attributes.

Example 11–1 Using the persist-changes Attribute in the adf-config.xml File

```
<adf-config xmlns="http://xmlns.oracle.com/adf/config">
  <adf-faces-config xmlns="http://xmlns.oracle.com/adf/faces/config">
    <persistent-change-manager>
      <persistent-change-manager-class>
        oracle.adf.view.rich.change.MDSDocumentChangeManager
      </persistent-change-manager-class>
    </persistent-change-manager>
    <taglib-config>
      <taglib uri="http://xmlns.oracle.com/adf/pageeditor">
        <tag name="imageLink">
          <attribute name="destination">
            <persist-changes>true</persist-changes>
          </attribute>
        </tag>
        <tag name="customAction">
          <attribute name="text">
            <persist-changes>true</persist-changes>
          </attribute>
          <attribute name="icon">
            <persist-changes>true</persist-changes>
          </attribute>
        </tag>
      </taglib>
    </taglib-config>
  </adf-faces-config>
</adf-config>
```

11.4.2 What Happens at Runtime

If you set persist-changes to true, then a change made to the component attribute is persisted in any instance of that component. If you set persist-changes to false, then even if you change the attribute value in the Component Properties dialog at runtime, the change is not saved.
11.5 Applying Action-Level Restrictions on Panel Customizable and Show Detail Component Actions

The ability of a user to perform actions on Panel Customizable and Show Detail Frame components is inherited from page security. Inheritance is based on the value of the application-wide switch, `enableSecurity`, in the `adf-config.xml` file. If you select the WebCenter 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 11.5.1, "How to Add an enableSecurity Section to adf-config.xml"
- Section 11.5.2, "Defining Security at the Actions Category Level"
- Section 11.5.3, "Defining Security at the Actions Level"

11.5.1 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 11–2, and set the `enableSecurity` element in that section to `true`.

```
<cust:customizableComponentsSecurity
xmlns="http://xmlns.oracle.com/adf/faces/customizable/config">
  <cust:enableSecurity value="true"/>
  <cust:actionsCategory>
    ..........................................
  </cust:actionsCategory>
</cust:customizableComponentsSecurity>
```

Restrictions on actions can be implemented at the following levels:

- **Page level**: You can define security for WebCenter 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 personalizations on that page. The `enableSecurity` element enables you to override the security inheritance behavior. It can take either of the following values:

  - **true**: If set to `true` (the default when not specified), then the ability of a user to modify a component is first determined from the page permissions and then adjusted according to the current set of actions defined for that type of permission. For example, if a user has customize permission, then the actions that constitute the customize category (move, customize, and so on) are available to the user, but they are overridden by the actions that are defined in the `adf-config.xml` file.
Applying Action-Level Restrictions on Panel Customizable and Show Detail Component Actions

- **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.

### 11.5.2 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 11–1 describes the different actionsCategory attributes and the actions they support by default.

For information about the Oracle Composer-specific configurations you can make in adf-config.xml, see Section B.2.2, "adf-config.xml."
Applying Action-Level Restrictions on Panel Customizable and Show Detail Component Actions

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.
- **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.

---

**Table 11-1  Actions Categories and Show Detail Frame Actions Mapping**

<table>
<thead>
<tr>
<th>actionsCategory</th>
<th>Actions Supported</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>personalizeActionsCategory</code></td>
<td>showMoveAction, showRemoveAction, showMinimizeAction, showResizer, allowAction</td>
</tr>
<tr>
<td><code>customizeActionsCategory</code></td>
<td>showEditAction, showAddContentAction, showSplitAction</td>
</tr>
</tbody>
</table>

---

**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 11.5.3, "Defining Security at the Actions Level."

---

**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 11-3 shows the `actionsCategory` entry that you can add to the customizable components actions security section in the `adf-config.xml` file.
Applying Action-Level Restrictions on Panel Customizable and Show Detail Component Actions

Example 11–3  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 11–4.

Example 11–4  EL Used in Customizable Components an actionsCategory 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 37–7 in Section 37.10, "Overriding Inherited Security on Portlets and Customizable Components."

11.5.3 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 enable or disable 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 Oracle 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 add content to the page and move or rearrange page content.

In Oracle 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 11–2 explains the ability to add content depending on the allowAction and showAddContentAction settings.
Table 11–2  Actions Settings for Adding Content to a Page

<table>
<thead>
<tr>
<th>allowAction Setting</th>
<th>showAddContentAction Setting</th>
<th>Add Content Button Enabled?</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 11–3 explains the ability to add content using the Add Box icons depending on the allowAction and showSplitAction settings.

Table 11–3  Actions Settings for Rearranging Content on a Page

<table>
<thead>
<tr>
<th>allowAction Setting</th>
<th>showSplitAction Setting</th>
<th>Add Box Icons Enabled?</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 11–4 explains the ability to move content on the page depending on the allowAction and showMoveAction settings.

Table 11–4  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 showEditAction and showRemoveAction are applicable only in Design view of a page. That is, if you set showEditAction and showRemoveAction on Show Detail Frame components to false, then users cannot edit or delete such components in Design view of the page. However, users can still select such components in Source view and edit or delete them.
Example 11–5 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 11–5  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>
```

### 11.6 Applying Task Flow Restrictions

Task flow permissions, stored in a secured application’s jazn-data.xml file, control the ability to zoom into and edit the task flow in Oracle Composer. By default, no permissions are provisioned on task flows. To enable users to edit a task flow at runtime, you must explicitly grant Edit, Customize, or Grant permission on the page.

**Before You Begin:**

Since editing task flow permissions has far reaching impact on the expected task flow editing experience, this capability is disabled by default. To enable task flow permission check, run the server with the following startup parameter:

```
-Doracle.composer.enableTaskflowPermissionCheck=true
```

To grant or revoke permissions on a task flow in the jazn-data.xml file:

1. From the Application menu, select Secure, and then ADF Policies.
2. In the jazn-data.xml file, select the task flow in the Overview editor and click the Add button in the Granted to Roles column, shown in Figure 11–10.
3. In the Select Roles dialog (Figure 11-11), select the application roles to which you want to grant permissions or create a new role by clicking the Add button.

4. Click OK.

5. Select one or more permissions in the Actions column.

11.7 Troubleshooting Problems with Oracle Composer Components Security

This section provides information to assist you in troubleshooting security-related problems you may encounter while using Oracle Composer components.

For information about configuring logging, see "Configuring ADF Logging for Oracle 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 11.6, "Applying Task Flow Restrictions."
This chapter describes the default Resource Catalog in your application and explains how to customize the runtime Catalog to display different content to different users.

This chapter contains the following sections:

- Section 12.1, "Overview of Resource Catalog"
- Section 12.2, "Modifying the Content of the Default Resource Catalog"
- Section 12.3, "Creating a Custom Resource Catalog"
- Section 12.4, "Adding XML Content to the Resource Catalog"
- Section 12.5, "Adding XML Content to the Resource Catalog: Examples"
- Section 12.6, "Filtering Items in the Resource Catalog"
- Section 12.7, "Configuring Multiple Resource Catalogs"

### 12.1 Overview of Resource Catalog

The Resource Catalog, also known as Oracle Business Dictionary, provides a consolidated view of the contents of one or more otherwise unrelated repositories in a unified search and browse user interface. Resources originate in their source repository and are then exposed to the developer through the Resource Catalog as shown in Figure 12–1.
When using JDeveloper, you see two types of Resource Catalogs—design time catalogs and runtime catalogs.

Design time catalogs are created in the Resource Palette and are like favorites lists for developers. You can define connections and then create catalogs to organize resources exposed by those connections. Those resources can then be re-used in any application you are developing.

See Also: Oracle JDeveloper online Help for more information about the Resource Palette and catalogs.

To locate this information in Oracle JDeveloper, from the Help menu, select Table of Contents. In Help Center, expand JDeveloper Basics and select Working with the Resource Palette.

Runtime catalogs are artifacts of an application, like JSPX pages, that are used only at runtime. These catalogs determine which components, task flows, and portlets are available for placing on a page in Edit mode. For information about adding Oracle Composer components to a page, see Section 7.1, "Designing Editable Pages Using Oracle Composer Components."

Default Resource Catalog Configuration

When you add a Page Customizable component to the page, the following configurations occur 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>\mds\oracle\adf\rc\metadata directory. Catalog definitions are XML files that specify the contents of the catalog.

- A default Resource Catalog is configured for the application. At runtime, Oracle Composer provides an option to add content to the page from this Resource Catalog.
The default Resource Catalog contains the **ADF Faces Components** folder, which contains components that a user can add to the page (Figure 12–2).

**Figure 12–2 Components in the Default Resource Catalog**

If you have registered a portlet producer with your application, then that producer's portlets are displayed in a Portlets folder in the Resource Catalog. If you have not registered any producer, the Portlets folder is hidden.

You can determine the components that are visible to users by modifying the default Resource Catalog or creating one or more of your own Resource Catalogs. Resource Catalogs can contain the following components:

- **Oracle ADF Faces components**: The ADF Faces Components folder provides Box, Movable Box, and Image components that are analogous to the JDeveloper design time components Panel Customizable, Show Detail Frame, and Image Link respectively. In JDeveloper, these components are available in the Oracle Composer tag library.

  The ADF Faces Components folder also provides Hyperlink, Text, and Web Page components that are analogous to the JDeveloper design time components Go Link, Rich Text Editor, and Web Page respectively. In JDeveloper, these components are available in the ADF Faces tag library.
- **Portlets**: You can add Java-PDK or WSRP portlets from any producer that was registered in Oracle JDeveloper. See Chapter 34, "Consuming Portlets" for detailed information about registering portlet producers and adding portlets to the page.

- **Task Flows**: If you have created task flows in Oracle JDeveloper, you can add those task flows from the JAR files onto your page at runtime.

### 12.2 Modifying the Content of the Default Resource Catalog

You can determine the contents of your Resource Catalog by modifying the default catalog definition file or creating a new catalog definition file and configuring Oracle Composer to point to it.

This section provides an example that illustrates how to include items of your choice in the default catalog definition file. It contains the following sections:

- Section 12.2.1, "How to Add Items to Your Resource Catalog"
- Section 12.2.2, "What Happens at Runtime"
- Section 12.2.3, "What You May Need to Know When Defining a Resource Catalog"
- Section 12.2.4, "How to Define Task Flow Parameters and Attributes of the Enclosing Show Detail Frames for Task Flows"

#### 12.2.1 How to Add Items to Your Resource Catalog

If you want to modify the contents of the default Resource Catalog, you must edit the default catalog definition file, `default-catalog.xml` and include the components of your choice. This section describes how to add more items to the default Resource Catalog. It contains the following subsections:

- Section 12.2.1.1, "Adding Connections for Resources"
- Section 12.2.1.2, "Defining the Catalog Contents"
- Section 12.2.1.3, "Enabling Task Flows in the Resource Catalog"
- Section 12.2.1.4, "Enabling Documents in the Resource Catalog"
- Section 12.2.1.5, "Enabling Display of Saved Searches in the Resource Catalog"

#### 12.2.1.1 Adding Connections for Resources

Many of the resources included in catalogs, such as Oracle WebCenter Services and portlets, are accessed through connections defined in your application’s `connections.xml` file.

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 3.10, "Accessing Connection Wizards."

**To create a connection from the Resource Palette:**

1. From the Resource Palette, click the **New** icon in the upper left corner.
2. Choose **New Connection** and then the type of connection you want to create.
   - The wizard for your connection type appears.
3. Step through the wizard filling in the necessary information about the connection.
   - When you click **Finish**, the connection should appear in the Resource Palette under **Connections**.
4. Right click your connection in the Resource Palette and choose **Add to Application** from the context menu.

Alternatively, you can drag and drop the connection from the Resource Palette to the Application Resources panel of the Application Navigator.

### 12.2.1.2 Defining the Catalog Contents

If you modify the default catalog definition file, your modifications appear in the catalog at runtime. The default catalog definition file, `default-catalog.xml`, is available in the `<Application_Root>/mds/oracle/adf/rc/metadata` directory.

You can modify the catalog content according to the rules of catalog definitions. Refer to the following for assistance in modifying the definition:

- Section C.2, "Default Catalog Definition"
- Section C.3, "XML Schema"
- Section C.4, "Catalog Definition Attributes"

The default catalog definition contains commented entries for displaying documents from the Documents service and out-of-the-box service task flows. If you want the Resource Catalog to display these components, you must enable relevant entries in the catalog definition file.

#### Example

This example describes the procedure for adding a **Discussions** folder to the Resource Catalog. The Discussions folder contains the Discussions service main view task flow as a resource.

To expose the new **Discussions** folder in the Resource Catalog, you must add the code shown in bold in Example 12–1 to the `default-catalog.xml` file.

**Example 12–1 Modified Sample default-catalog.xml File**

```xml
<?xml version='1.0' encoding='UTF-8'?>
<catalogDefinition xmlns="http://xmlns.oracle.com/adf/rcs/catalog" id='catalogDefinition' name='Default Resource Catalog' description='Default resource catalog definition containing sample entries'>
  <contents>

    <!-- **********************************************************************
    * Custom folder exposing ADF Faces components                           *
    * Comment out this element if you want to remove it from the catalog.   *
    ********************************************************************** -->
    <customFolder id='facesComponents' name='ADF Faces Components' factoryClass='oracle.adfinternal.view.page.editor.componentcatalog.adapter.ComponentObjectFactory' description='ADF Faces components you can add to application pages'/>

    <!-- **********************************************************************
    * Dynamically include the portlets custom folder into the catalog at    *
    * runtime. The custom folder is only included if the portlet runtime    *
    * jar files are included in the application classpath.                  *
    ********************************************************************** -->
    <customContent id='portletContent' description='ADF portlet components you can add to application pages'/>

</contents>
</catalogDefinition>
```

Modifying the Content of the Default Resource Catalog

```xml
<!-- **********************************************************************
* This entry is added to modify the default-catalog.xml                *
* In this entry  resource is published within a folder.             *
* -->

<folder name="Discussions" id="mydiscussions">
  <contents>
    <resource id="discussionsMainView"
      name="Discussions"
      description="Main view of the Discussions Service"
      repository="application.classpath"
      path="forum-view.jar/ADF_TaskFlow/oracle+webcenter+collab+forum+view+taskflows+main-task-flow.xml#forum-main"/>
  </contents>
</folder>

<!-- **********************************************************************
* Uncomment the following <customFolder> element if you have configured the Document Library service in your application and want to include documents in the catalog.                                               *
*********************************************************************** -->

<!-- **********************************************************************
* Uncomment the following <resource> element if you have configured the Document Library service in your application and want to include the Document Library main view in the catalog.                              *
* NOTE: application.classpath is an implicitly defined "repository" for accessing ADF Libraries that are included in your classpath  
*********************************************************************** -->

<!-- **********************************************************************
* To create a link to a task flow in an ADF Library:                      *
* 1) add a <resource> tag to your catalog definition & set the "repository" attribute to 'application.classpath'      *
* (see doclibMainView above)                                           *
* 3) set the 'repository' attribute to the name of your file system connection.                                    *
* 4) set the 'path' attribute for the task flow you are interested in.  *
*    The path is of the following form:                             *
*    path_to_jar/ADF_TaskFlow/task_flow_path                         *
*********************************************************************** -->
```
To obtain the task_flow_path, use the file system connection from (1) and navigate to the task flow in your ADF Library. Mouse-over your task flow so its tool-tip is displayed. The tooltip shows the fully qualified ID of the task flow. Take this value and replace all occurrences of "/" with "+". For example:

Tool-tip:
oracle/webcenter/collab/announcement/view/taskflows/main-view-definition.xml#announcement-main-view

Task_flow_path:
oracle+webcenter+collab+announcement+view+taskflows+main-view-definition.xml#announcement-main-view

After you add the specified code, the Discussions folder is displayed in the runtime Resource Catalog.

**To run and test the page:**
1. Run any editable JSPX page in your application.
2. Switch to page Edit mode.
3. Click an Add Content button on the page to view the Discussions folder in the Catalog dialog.

### 12.2.1.3 Enabling Task Flows in the Resource Catalog

If you want the Resource Catalog to display an out-of-the-box service task flow or custom task flow that you created in your application, you must add entries for the task flow in the `default-catalog.xml` file. Custom task flows must additionally be packaged in an ADF library, even if they are part of the same application.

This section describes how to package a custom task flow in an ADF library and how to enable the display of task flows in the Resource Catalog. It contains the following subsections:

- [Section 12.2.1.3.1, "How to Package a Custom Task Flow in an ADF Library"](#)
- [Section 12.2.1.3.2, "How to Enable the Display of Task Flows in the Resource Catalog"](#)

#### 12.2.1.3.1 How to Package a Custom Task Flow in an ADF Library

To package a custom task flow in an ADF library:

1. Create a deployment profile for the task flow:
   a. Right-click ViewController and choose New.
   b. In the New Gallery, expand General, select Deployment Profile, and then ADF Library JAR File, and click OK.
   c. In the Create Deployment Profile -- ADF Library JAR File dialog, enter a name for your deployment profile and click OK.
   d. In the ADF Library JAR Deployment Profile Properties dialog, click OK.
   e. In the Project Properties dialog, click OK.
2. In the Application Navigator, right-click the project folder, choose **Deploy**, *deployment profile name*, to, and then choose **to ADF Library JAR file**.

This creates a **deploy** folder including the JAR file, in the project folder located at `<Application_Root>\ViewController\deploy\`.

### 12.2.1.3.2 How to Enable the Display of Task Flows in the Resource Catalog

To enable the display of task flows in the Resource Catalog:

1. Open the `default-catalog.xml` file in the `<Application_Root>\mds\oracle\adf\rc\metadata` directory.

2. Add a `<resource>` element to your catalog definition similar to the following example:

   ```xml
   <resource id="doclibMainView"
     name="Document Library Task Flow"
     description="Main view of the Document Library Service"
     repository="application.classpath"
     path="doclib-service-view.jar/ADF_TaskFlow/oracle+webcenter+doclib+view+jsf+taskflows+mainView.xml#doclib-document-library"/>
   ```

   **Note:** Ensure that the task flow library is in the application class path.

3. Set the `path` attribute for the task flow you want to include in the catalog. The path must be specified in the following format:

   ```text
   jar_file_name/ADF_TaskFlow/task_flow_path
   ```

   where:

   - `jar_file_name` is the name of the task flow JAR file. This JAR is picked up from the application classpath.
   - `task_flow_path` is the path to the task flow within the JAR file.

   To obtain the task flow path for any task flow exposed in the Resource Palette, including out-of-the-box service task flows, navigate to the task flow in your ADF Library and right-click the task flow. From the context menu, select **Show Catalog Reference**. A message dialog displays the values of the `repository` and `path` attributes for that task flow.


The task flows that you have defined in `default-catalog.xml` are now available in the Resource Catalog.

**Note:** To enable users to view the task flow's content at runtime, ensure that the task flow you add to the Resource Catalog has a `TaskFlowPermission` grant in the application's `jazn-data.xml` file, with at least `View` action provisioned. Without this, the task flow appears blank when added to the page.
12.2.1.4 Enabling Documents in the Resource Catalog

The default catalog definition file has entries for displaying documents from the Documents service, but these are commented out. Consequently, Documents service task flows (which include Document Manager, Document List Viewer, Recent Documents, and Content Presenter) are not available at runtime when using the default version of the Resource Catalog (see Section C.2, "Default Catalog Definition").

If you have configured the Documents service in your application (see Section 13.2, "Configuring Content Repository Connections") and want to expose its task flows in Oracle Composer at runtime, you can uncomment the Documents service entries in the default catalog definition file.

To enable display of Documents service task flows:

1. Open the default-catalog.xml file in JDeveloper.
2. Uncomment the <customFolder> element with an id value doclibDocuments (for the Document Manager, Document List Viewer, Recent Documents task flows).
3. Uncomment the <resource> element with an id value doclibContentPresenter (for the Content Presenter task flow).
4. Save the file.

Documents service task flows now display in a Documents folder in the Oracle Composer Catalog.

12.2.1.5 Enabling Display of Saved Searches in the Resource Catalog

Consider an example where your custom WebCenter Application page includes the Search task flow. At runtime, users can perform searches using this task flow and also save the results. To enable users to add such saved searches to their application pages, you must update the catalog definition file to include a saved searches folder. The Oracle Composer Catalog then displays this folder, which lists all saved searches.

To display saved searches in the Resource Catalog so that users can add them to their application pages, you must add a customFolder entry in the catalog definition file, as shown in the following example:

```xml
<customFolder id="savedSearchesFolder"
  factoryClass="oracle.webcenter.search.view.rc.SearchServiceContextFactory">
  <attributes>
    <attribute value="Saved Searches" attributeId="Title" isKey="true"/>
    <attribute value="Saved searches you can add to application pages" attributeId="Description" isKey="true"/>
    <attribute value="SAVED_SEARCHES.KEYWORDS" attributeId="Subject" isKey="true"/>
    <attribute value="oracle.webcenter.search" attributeId="WEBCENTER_SERVICE_ID" isKey="false"/>
    <attribute value="/adf/webcenter/foldersavedsearches_qualifier.png"/>
  </attributes>
</customFolder>
```

Note: As the Search service searches content created inside WebCenter applications by WebCenter services, ensure that you have configured other services that own resources, such as announcements, discussions, documents, tags, wikis/blogs, and pages. For more information, see Chapter 24, "Integrating the Search Service."
The Oracle Composer Catalog displays a saved searches folder, as shown in Figure 12–3. Users can open the folder, select a saved search, and add it to their page.

Figure 12–3  Saved Searches Folder in the Catalog

12.2.2 What Happens at Runtime

After you enable Oracle Composer and define your catalog contents, you can run the page from Oracle JDeveloper. The Resource Catalog displays all elements that you defined in the catalog definition file.

To run and test the page:

1. In the Application Navigator, open your customizable MyPage.jspx and run it.
2. Switch to page Edit mode.
3. Click an Add Content button on the page to display the Catalog dialog.

In addition to the ADF Faces Components folder, the Resource Catalog displays all the elements you defined in the catalog definition file at design time.

For example, Figure 12–4 shows the Discussions folder that you enabled in Example 12–1.
You can look at the sample application, ModifiedResourceCatalog.jws, containing a similar modified catalog definition, on the Oracle WebCenter Suite 11g Demonstrations and Samples page on the Oracle Technology Network (OTN) at:

http://webcenter.oracle.com

12.2.3 What You May Need to Know When Defining a Resource Catalog

To correctly implement resource catalogs, you must know where the application looks for the definitions and the XML schema upon which the definitions are based. See Appendix B, "Oracle Composer Component Properties and Files" for further information.

12.2.4 How to Define Task Flow Parameters and Attributes of the Enclosing Show Detail Frames for Task Flows

When you add a task flow from the Resource Catalog to your application page, the task flow is automatically enclosed in a Show Detail Frame component. To enable runtime editing of attributes associated with the enclosing Show Detail Frame, you must first define those attributes in the catalog definition file where the task flow is registered.
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 define this attribute in the catalog definition file. When a user selects the task flow for editing, the Component Properties dialog displays this Show Detail Frame attribute along with other task flow attributes.

Similarly, to prepopulate task flow parameters so that users do not have to set parameter values at runtime, you can define those parameters in the catalog definition file using an EL expression. When a user selects the task flow for editing, the Component Properties dialog displays these parameters with prepopulated values.

To define task flow parameters and Show Detail Frame attributes:

1. Open the catalog definition file that contains the task flow for which you want to set the parameters and Show Detail Frame attributes.

2. Within the task flow's <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 prefixed with parameter.; the attributeId for a Show Detail Frame attribute must contain the attribute name prefixed with attr. as shown in the following example:

   ```xml
   <resource id="sampletaskflow" name="Sample Taskflow"
    path="sampletaskflow.jar/ADF_TaskFlow/WEB-INF+sample-task-flow-definition.xml#sample-task-flow-definition"
    repository="application.classpath">
   <attributes>
   <attribute value="dark" attributeId="attr.background"
    isKey="false"/>
   <attribute value="#{myBean.myParam1}" attributeId="parameter.myParam1"
    isKey="false"/>
   </attributes>
   </resource>
   </contents>
   </catalogDefinition>
   
   Note: Retain the default value of false for the isKey attribute. Set it to true only if the value is coming from a resource bundle.

3. Save the catalog definition file.

### 12.3 Creating a Custom Resource Catalog

You can define the contents of your application's Resource Catalog either by modifying the existing, default catalog definition file, default-catalog.xml, or by creating a new catalog definition file. Create a custom Resource Catalog when you do
not want to expose the ADF Faces components that are available in the default catalog, but instead want to include components or task flows that you specify.

This section describes how to create a custom Resource Catalog. It contains the following subsections:

- Section 12.3.1, "How to Create a Custom Resource Catalog"
- Section 12.3.2, "What Happens at Runtime"

12.3.1 How to Create a Custom Resource Catalog

To override default settings and display your custom Resource Catalog you must first create a new catalog definition file. Then you must specify the name of the new file in the application's adf-config.xml file.

To create a catalog definition:

1. Open your application in Oracle JDeveloper.
2. From File menu, choose New.
3. In the New Gallery dialog, expand General, select XML, then XML Document, and click OK.
4. In the Create XML File dialog, enter the file name, for example, users-catalog.xml, and the path of your catalog definition file. Catalog definition files must be located in the Resource Catalog root directory (<JDEV_HOME>/<jdev/mywork/<Application_Name>/mds/oracle/adf/rc/mds/oracle/adf/rc/metadata) or a subdirectory of the root. For example, the Resource Catalog root might be:

   C:\JDeveloper\mywork\RCSampleApp\mds/oracle/adf/rc\metadata

   You could create your catalog here or create a subdirectory for it. For example:

   C:\JDeveloper\mywork\RCSampleApp\mds/oracle/adf/rc\metadata\mycatalogs

5. Click OK.
6. You can now code your XML file according to the rules of catalog definitions.

   Refer to the following to help you create the definition:

   - Section C.2, "Default Catalog Definition"
   - Section C.3, "XML Schema"
   - Section C.4, "Catalog Definition Attributes"

   Tip: If you do not want to write your catalog definition from the start, you could copy the contents of default-catalog.xml into your new catalog definition as a starting point.

Example 12–2 shows a sample catalog definition file containing the ADF Faces Components folder with its layout components, and a Discussions folder with the Discussions service task flows.

Example 12–2 The users-catalog.xml File

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<catalogDefinition xmlns="http://xmlns.oracle.com/adf/rcs/catalog"
    id="catalogDefinition">
```
Creating a Custom Resource Catalog

name='Default Resource Catalog'
description='Default resource catalog definition containing sample entries'>
<customFolder id='facesComponents' name='ADF Faces Components'
description='ADF Faces components you can add to application pages'
factoryClass='oracle.adfinternal.view.page.editor.componentcatalog.adapter.ComponeObjectFactory' />
<filder id='forumsFolder'>
<attributes>
<attribute value='Discussions' attributeId='Title' isKey='false' />
<attribute value='Discussion Forum Taskflows' attributeId='Description' isKey='false' />
<attribute value='' attributeId='Subject' isKey='false' />
<attribute value='/adf/webcenter/folderdiscussions_qualifier.png' attributeId='IconURI' />
attributeId='IconURI' />
</attributes>
</folder>
</contents>
</catalogDefinition>

where:

<resource> elements are used to define resources that you can add to the page
<folder> elements are used to define folders in the catalog
7. Add an `<rcv-config>` element to register `users-catalog` in your `adf-config.xml` file as shown in the following example:

```xml
<rcv-config xmlns="http://xmlns.oracle.com/adf/rcs/viewer/adf-config">
  <default-catalog catalog-name="users-catalog"/>
</rcv:rcv-config>
```

The value for `catalog-name` specifies the name of the catalog definition file to use.

The name you specify is relative to the Resource Catalog root directory and should not include the file extension `.xml`. For example, if you saved your new catalog definition as `C:\JDeveloper\mywork\RCSampleApp\mds\oracle\adf\rc\metadata\mycatalogs\users-catalog.xml`, then the catalog name is `mycatalogs/users-catalog`. The name uses the `/` separator because it represents part of an MDS path.

These entries enable you to override the default Resource Catalog. You must add them specifically because they are not included in the `adf-config.xml` file by default.

---

**Note:** For information about the Oracle Composer-specific configurations you can make in `adf-config.xml`, see Section B.2.2, "adf-config.xml."

---

8. Save `adf-config.xml`.

### 12.3.2 What Happens at Runtime

When you run your application and edit a page, Oracle Composer now displays content based on your custom catalog.

**To run and test the page:**

1. Run any editable JSPX page in your application.
2. Switch to page Edit mode.
3. Click an Add Content button on the page.

The Catalog dialog displays your custom catalog.

Figure 12-5 shows the custom catalog, `users-catalog`, that was described in Example 12-2.
12.4 Adding XML Content to the Resource Catalog

You can add custom XML components to your Resource Catalog by including the XML code for such components in the catalog definition file. On adding such a component to the page from the Resource Catalog, the XML fragment is displayed on the page. You can add a variety of components, such as simple ADF Faces components, compound objects containing two or more components, and JSF Verbatim tags that let you add arbitrary HTML content inside them. An example for arbitrary HTML that you can add to your catalog is a YouTube video.

This section describes how to include custom components in the catalog definition file. It contains the following sections:

- Section 12.4.1, "How to Add XML Content to the Resource Catalog"
- Section 12.4.2, "What Happens at Runtime"
12.4.1 How to Add XML Content to the Resource Catalog

To add raw XML to your catalog definition file, you must include it in a component element within the contents section. Example 12–3 shows the code to add a Panel Customizable component to the Resource Catalog.

Example 12–3 Panel Customizable Component in the Catalog Definition File

```xml
<!-- panelCustomizable -->
<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_qualified.png"/>
  </attributes>
  <parameters>
    <parameter id="xml">
      <![CDATA[
      ]]>
    </parameter>
  </parameters>
</component>
```

To add raw XML content to the Resource Catalog:

1. Open the catalog definition file in the `<Application Root>`\`mds\oracle\adf\rc\metadata` directory.

2. Add a component element within the contents section, and set the `factoryClass` attribute to `oracle.adf.rc.component.XmlComponentFactory`.

   `factoryClass` is the name of a Java class that implements the `oracle.adf.rc.component.ComponentFactory` interface and generates a resource catalog item based on a set of parameters. Each `ComponentFactory` implementation defines the set of parameters it requires. For example, `XmlComponentFactory` requires a single parameter named `xml` and the value of the parameter is a block of XML representing a component or components that can be inserted into a page.

3. Define attributes on the `component` element.

   For more information, see Section C.4, "Catalog Definition Attributes."

4. Within the `component` element, add a `parameters` element with a nested `parameter` element.

5. Set the `id` on the `parameter` element to `xml`.

   This indicates that the component contains raw XML.

6. Add a `CDATA` tag within the `parameter` element.

7. Include the XML fragment for your component within the `CDATA` tag. The `CDATA` tag indicates that the component contains character data.

   Specify the namespace for the custom component you added, and set its `id` to `#`.

   This ensures that an ADF faces ID is generated for the component when it is dropped on the page at runtime. This ID is useful while performing other
configurations in your application. For example, while configuring a drop handler for the resource.

8. Save your catalog definition file.

12.4.2 What Happens at Runtime

The components you add to your catalog definition file are rendered in the runtime Resource Catalog. The code in Example 12–3 results in the Oracle Composer Catalog displaying the components shown in Figure 12–6.

Figure 12–6 Panel Customizable Component Titled Box in the Catalog

12.5 Adding XML Content to the Resource Catalog: Examples

This section provides examples of various custom components added to the catalog definition file.

12.5.1 Compound Object Containing Nested ADF Faces Components

Example 12–4 shows the code to add an ADF Faces component that has another ADF Faces component nested inside it.
Example 12–4  Nested ADF Faces Components in the Catalog Definition File

```xml
<component id='sdf' factoryClass="oracle.adf.rc.component.XmlComponentFactory">
  <attributes>
    <attribute attributeId="Title" value="MOVABLE_BOX" isKey="true"/>
    <attribute attributeId="Description" value="MOVABLE_BOX.DESCRIPTION" isKey="true"/>
    <attribute attributeId="Subject" value="MOVABLE_BOX.KEYWORDS" isKey="true"/>
    <attribute attributeId="IconURI" value="/adf/webcenter/moveframe_qualifier.png"/>
  </attributes>
  <parameters>
    <parameter id="xml">
      <![CDATA[
        <cust:showDetailFrame
          xmlns:cust="http://xmlns.oracle.com/adf/faces/customizable"
          id="#" text="#{componentExtensionBundle.MOVABLE_BOX}"
        >
          <cust:panelCustomizable id="#" layout="scroll"/>
        </cust:showDetailFrame>
      ]]>"
    </parameter>
  </parameters>
</component>
```

Figure 12–7 shows how the component appears in the Oracle Composer Catalog. When a user adds the Movable Box component to their page, a Box component (Panel Customizable component) is added automatically as a child component.
12.5.2 YouTube Video Nested in a Verbatim Component

Example 12-5 shows the code to add a flash object showing a YouTube video to the Resource Catalog. The <object> element was copied directly from the YouTube site, available from the embed field or the Copy embed html context menu on the video, and pasted inside a Verbatim tag.

Example 12-5 Raw HTML Nested in a Verbatim Component in the Catalog Definition File

```xml
<!-- XML content - SDF around verbatim <object> -->
<component id="yt1" factoryClass="oracle.adf.rc.component.XmlComponentFactory">
  <attributes>
    <attribute attributeId="Title" value="Shuttle Launch 1"/>
    <attribute attributeId="Description" value="YouTube Video (Verbatim)"/>
    <attribute attributeId="IconURI" value="/adf/webcenter/videofile_qualifier.png"/>
  </attributes>
  <parameters>
    <parameter id="xml">
      <![CDATA[
        <cust:showDetailFrame id="#" stretchContent="false" text="Shuttle Launch (Verbatim)"
        showResizer="never"
      ]]>
    </parameter>
  </parameters>
</component>
```
Figure 12–8 shows how the component appears in the Oracle Composer Catalog. When a user adds this component to their page, it displays as shown in Figure 12–9.

**Figure 12–8  YouTube Video in a Verbatim Component in the Catalog**
12.5.3 YouTube Video Nested in an Output Text Component

Example 12–6 shows the code to add a YouTube video within an ADF Faces Output Text component. The XML code available from the embed field on the YouTube site or Copy embed html context menu on the video was copied and pasted against the value parameter of the Output Text component. Consequently, users can select the Output Text component in Oracle Composer and edit the video object in Oracle Composer’s Component Properties dialog.

The XML content here is the same as that used in Section 12.5.2, "YouTube Video Nested in a Verbatim Component," but in this case the XML is escaped, since it is being used as the value of an attribute.

Example 12–6 Raw HTML Nested In an Output Text Component in the Catalog Definition File

<!-- XML content - SDF around unescaped outputText -->
<component id="yt2" factoryClass="oracle.adf.rc.component.XmlComponentFactory">
  <attributes>
    <attribute attributeId="Title" value="Shuttle Launch 2"/>
    <attribute attributeId="Description" value="YouTube Video (Output Text)"/>
    <attribute attributeId="IconURI" value="/adf/webcenter/videofile_qualifier.png"/>
  </attributes>
  <parameters>
    <parameter id="xml">
      <![CDATA[
        <cust:showDetailFrame id="#" stretchContent="false" text="Shuttle Launch (Output Text)"
        xmlns:cust="http://xmlns.oracle.com/adf/faces/customizable">
          <f:attribute xmlns:f="http://java.sun.com/jsf/core"
            name="sdf_selection_rule" value="sdf_for_edit_mode_only"/>
          <af:outputText id="#" escape="false" value="&lt;object width="560& quot;...
Figure 12–10 shows the video object in the Catalog.

**Figure 12–10  YouTube Video in an Output Text Component in the Catalog**

Figure 12–11 shows how the object is rendered on the page.
Figure 12–11  YouTube Video on an Application Page

Figure 12–11 shows the object in the Edit dialog, which is invoked when you click Expression Builder next to the Value property field for the Output Text component in the Component Properties dialog.

Figure 12–12  XML for the YouTube Video Object in the Edit dialog for the Output Text’s Value Property

Figure 12–12 shows the object in the Edit dialog, which is invoked when you click Expression Builder next to the Value property field for the Output Text component in the Component Properties dialog.
12.6 Filtering Items in the Resource Catalog

If you want to arrange catalog content so that it is essentially the same for all users, but displays different subsets of the content to different users, then you can configure your Resource Catalog to filter out selected items based on specific criteria.

For example, if two users A and B have different privileges on a page, you can configure your application to display the entire catalog to user A and only a subset of items in the catalog to user B.

This section describes how to filter items in the Resource Catalog. It contains the following subsections:

- Section 12.6.1, "How to Filter Items in the Resource Catalog"
- Section 12.6.2, "What Happens at Runtime"

12.6.1 How to Filter Items in the Resource Catalog

To display subsets of Resource Catalog items to different users, you must implement the `oracle.adf.rc.spi.plugin.catalog.CatalogDefinitionFilter` interface. Additionally, you must associate this filter with the Resource Catalog.

To implement the `CatalogDefinitionFilter`:

1. From the File menu, choose New.
2. In the New Gallery dialog, expand General, select Java, and then Java Class, and then click OK.
3. In the Create Java Class dialog, specify a name for the class, for example, CatalogFilter.
4. Under the Optional Attributes section, add the `oracle.adf.rc.spi.plugin.catalog.CatalogDefinitionFilter` interface.
5. Click OK.
6. Specify the logic for filtering items.

---

Note: When implementing the filter logic it is common to want information about the current user and current page. This information can be obtained from the FacesContext.

---

Example 12–7 shows a sample `CatalogDefinitionFilter` implementation. It is configured to hide the Discussions folder (forumsFolder) for user ngreenbe from the custom catalog, users-catalog.xml, described in Section 12.3, "Creating a Custom Resource Catalog." The entire catalog, with the ADF Faces Components and Discussions folders, is shown to all other users.

**Example 12–7  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;
```

Configuring the Resource Catalog for Oracle Composer   12-25
import javax.faces.context.ExternalContext;
import javax.faces.context.FacesContext;

public class CatalogFilter implements CatalogDefinitionFilter {

    public boolean includeInCatalog(CatalogElement element, Hashtable env) {
        try {
            FacesContext fctx = FacesContext.getCurrentInstance();
            ExternalContext ectx = fctx.getExternalContext();
            String user = ectx.getRemoteUser();
            String resourceId = element.getId();
            if ((user != null && user.equals("ngreenbe")) &&
                resourceId != null && resourceId.equals("forumsFolder")) {
                return false;
            }
        } catch (Exception e) {
            System.out.println(e);
        }
        return true;
    }
}

7. Edit the application's catalog definition file to add the definitionFilter entry inside the <catalogDefinition> element:

    <catalogDefinition xmlns="http://xmlns.oracle.com/adf/rcs/catalog"
        id="catalogDefinition"
        name="Default Resource Catalog"
        description="Default resource catalog definition containing sample entries"
        definitionFilter="java.class.name">
        
    The value for java_class_name is the name of the Java class you created, for example, webcenter.CatalogFilter.

8. Save the catalog definition file.

12.6.2 What Happens at Runtime

When your catalog definition is opened, each entry in the catalog is passed through the CatalogDefinitionFilter to determine which entries should be displayed and which should be excluded. Depending on your implementation in the Java class, different subsets of items are displayed to different users.

Figure 12–14 and Figure 12–13 show the catalogs that are rendered for ngreenbe and any other user, for example, sking, on implementing the CatalogDefinitionFilter interface using the logic in Example 12–7.
Figure 12–13 Resource Catalog Displayed to ngreenbe
The sample application, CustomResourceCatalog.jws, contains a similar implementation of CatalogDefinitionFilter. You can access it from the Oracle WebCenter Suite 11g Demonstrations and Samples page on the Oracle Technology Network (OTN) at:

http://webcenter.oracle.com

12.7 Configuring Multiple Resource Catalogs

To expose different items to different users based on specific criteria, such as the page being edited, the user, or the user role, you can associate multiple Resource Catalogs with your application. This section describes how. It contains the following subsections:

- Section 12.7.1, "How to Configure Multiple Resource Catalogs"
- Section 12.7.2, "What Happens at Runtime"

12.7.1 How to Configure Multiple Resource Catalogs

The main difference between using multiple catalogs and a single catalog is that, in addition to creating custom catalogs, you must implement the ResourceCatalogSelector API to select the appropriate catalog for the user and
the page being edited. You must include the catalog-selector entry in your application's adf-config.xml file to specify the name of your ResourceCatalogSelector class. If you do not specify a catalog-selector, the default catalog is used for all editable pages and all users see the same content. If you specify a catalog-selector, the default catalog is used only when the specified selector returns null.

To implement multiple resource catalogs:

1. Create your custom catalogs by performing the steps outlined in Section 12.3, "Creating a Custom Resource Catalog."

2. Implement the oracle.adf.rc.model.config.ResourceCatalogSelector interface in your own resource catalog selector class, for example, CatalogSelector.java.

Example 12–8 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 ngreenbe. If there is a problem with rendering either of these catalogs, then the default catalog is shown to users.

Example 12–8  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 & 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("ngreenbe") || user.equals("sking")) {
    // return users-catalog for users ngreenbe 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 <catalog-selector> tag as show in the
following example:

<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>

The value for catalog-name corresponds to the default catalog definition file
stored in the Resource Catalog root within MDS.

Note: For information about the Oracle Composer-specific
configurations you can make in adf-config.xml, see Section B.2.2,
"adf-config.xml."

12.7.2 What Happens at Runtime

Different catalogs are displayed to users based on the criteria you specified.

Based on the ResourceCatalogSelector implementation in Example 12–8,
Figure 12–15 and Figure 12–16 show the two catalogs that are shown to user sking or
ngreenbe and to administrator aahunold respectively.
Figure 12–15  Catalog Displayed to sking and ngreenbe
You can look at these catalog definitions in the StoreFrontModule in FOD. For information about FOD, see Chapter 2, "Introduction to the WebCenter Sample Application."

12.8 Troubleshooting Problems with the Resource Catalog

This section provides information to assist you in troubleshooting problems you may encounter while using the Resource Catalog.

For information about configuring logging, see "Configuring ADF Logging for Oracle 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.
Troubleshooting Problems with the Resource Catalog

Configuring the Resource Catalog for Oracle Composer

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:

```
path_to_jar/ADF_TaskFlow/task_flow_path
Ex. doclib-service-view.jar/ADF_TaskFlow/oracle+webcenter+doclib+view+jsf+taskflows+mainView.xml#doclib-document-library
```
Part III contains the following chapters:

- Chapter 13, "Integrating Content"
- Chapter 14, "Integrating the Documents Service"
- Chapter 15, "Integrating the RSS Service"
Oracle WebCenter enables content integration through:

- The Documents service, which enables end users to view and manage documents and other types of content in your organization's content repositories.
  
The Documents service includes the Content Presenter task flow, which enables end users to select content in a variety of ways and then display those items using available display templates.

- Content repository data controls, which enable read-only access to a content repository, and maintain tight control over the way the content displays in a custom WebCenter application.

This chapter discusses how to configure connections to content repositories to provide access to the content, and how to add content to pages in a custom WebCenter application. For more information about managing and including content in your WebCenter applications, see:

- Chapter 14, "Integrating the Documents Service" to integrate the Documents service in custom WebCenter applications at design time to provide end users with a user-friendly interface to manage, display, and search documents at runtime.

- "Managing Content Repositories" in Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter to configure and manage content repositories used by WebCenter applications.

- "Working with the Documents Service" in Oracle Fusion Middleware User’s Guide for Oracle WebCenter to work with the Documents service and task flows at runtime in WebCenter applications.

This chapter includes the following sections:

- Section 13.1, "Introduction to Content Integration"
- Section 13.2, "Configuring Content Repository Connections"
- Section 13.3, "Adding Content to a JSF Page"
- Section 13.4, "Creating Custom Templates for Displaying Content Using Content Presenter"
- Section 13.5, "Configuring Content Data Controls for JCR Adapters"
- Section 13.6, "Integrating Content Using Content Data Controls: Examples"
13.1 Introduction to Content Integration

Oracle WebCenter's content integration capabilities enable you to include in your WebCenter applications decentralized content located across multiple repositories through the Documents service task flows or JCR adapters using JCR APIs.

The Documents service includes the Content Presenter, Document Manager, Document List View, and Recent Documents task flows, to provide different ways of integrating content into a custom WebCenter application. For information about the Documents service and its task flows, see Chapter 14, "Integrating the Documents Service".

The Content Presenter task flow allows you to add content from an Oracle Content Server content repository to a page in your WebCenter application, rendering the content in a selected display template. A Content Presenter task flow can be added either:

- at design time, during development of a custom WebCenter application (see Section 13.3, "Adding Content to a JSF Page").
- at runtime, by integrating the Documents service in your custom WebCenter application (see Chapter 14, "Integrating the Documents Service"), allowing end users to add the task flow to editable pages (see "Displaying Content Using the Content Presenter Task Flow" in Oracle Fusion Middleware User’s Guide for Oracle WebCenter)

JCR APIs enable independent access to content, regardless of the underlying repository or the type of the content; for example, documents, relational content, and so on. The JCR 1.0 API, as defined in JSR 170, provides a set of basic capabilities for reading, writing, browsing, and searching content. The JCR data controls also enable you to connect to and read from other JCR 1.0 repositories. To add content available in Oracle Content Server, Oracle Portal, Oracle WebCenter Adapter for Microsoft SharePoint, and file systems to JSF pages, you first create connections to the repositories, then use the connections to create data controls based on the repositories.

This section provides overview information about the provisions that enable you to use these methods to integrate content into your custom WebCenter applications:

- Section 13.1.1, "Overview of Content Adapters"
- Section 13.1.2, "Overview of Content Presenter Display Templates and the Templates Registry."
- Section 13.1.3, "Overview of Content Data Controls"

13.1.1 Overview of Content Adapters

In your custom WebCenter application, the following adapters are integrated by default:

- Oracle Content Server: The Oracle Content Server adapter is used to integrate content from an automated information system managed by Oracle Content Server. This system enables sharing, managing, and distributing of business information through a Web site as a common access point.
- Oracle Portal: The Oracle Portal adapter is used to integrate content from a content repository located in the Oracle Portal schema.
- Oracle WebCenter Adapter for Microsoft SharePoint: This adapter is used to integrate content from the Microsoft SharePoint 2007 repository.
File System: The File System adapter is used to add content located on your operating system’s file system to JSF pages.

**Note:** The File System adapter is intended to be used in the development environment only.

The Oracle Content Server, Oracle Portal, and file system adapters are bundled with Oracle WebCenter Extension bundle.

### 13.1.2 Overview of Content Presenter Display Templates and the Templates Registry

A *Content Presenter display template* is a JSFF file (JSF page fragment) that defines how content repository items should render on a page. Several useful display templates are provided out-of-the-box to get you started (see Table 14–4 and Table 14–5). Additionally, you can develop custom display templates to handle custom content properties or to satisfy specific content rendering requirements within your organization. Template development supports the full set of Rich ADF components so you can quickly and easily create robust and attractive templates to display your content.

Content Presenter display templates can handle either one content item or multiple content items. Templates that handle multiple content items can optionally take advantage of single item templates. 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.

Each content item is associated with a specific *content type* defined in the Oracle Content Server repository. A content type defines the properties of the content item. As a custom display template developer, you will need to know the names of the properties defined for the associated content type so that you can define how to display the selected content item(s) on the page. One way to determine the properties for the existing content types defined in Oracle Content Server is to use the Content Presenter Configuration dialog in WebCenter Spaces.

The *Content Presenter templates registry* (`templates-registry.xml`) maintains the registry of available display templates for use with Content Presenter. The default registry contains the default display templates provided out-of-the-box. As you create custom display templates, you must add a section for each template that you develop to the templates registry file.

Custom Content Presenter display templates not only allow you to provide specialized rendering of content items, but they can also be used to standardize the display of content items that are different. For example, consider a scenario where you want to create a list of Press Releases, but there are two different kinds of Press Releases defined in your content repository, each one using different property names for the short title and long body. You can create two custom display templates with similar formatting, then create a custom multiple item Press Release listing template to choose the correct single item template for each of the two kinds of Press Releases based on their content type.

At runtime, Content Presenter checks the templates registry to identify the display templates that can render the currently selected content item(s), and presents these templates to the end user for selection in the Content Presenter Configuration dialog.

For steps and examples, see Section 13.4, "Creating Custom Templates for Displaying Content Using Content Presenter".
13.1.3 Overview of Content Data Controls

A content data control is a container for all the data objects, collections, methods, and operations used to create user interface (UI) components within your custom WebCenter 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 File System, Oracle Portal, and Oracle Content Server data controls. The custom attributes are requirement-specific and can be added while creating content data controls.

The following sections describe the methods, parameters, and default attributes that are common across File System, Oracle Portal, and Oracle Content Server data controls:

- Section 13.1.3.1, "The getItems Method"
- Section 13.1.3.2, "The search Method"
- Section 13.1.3.3, "The advancedSearch Method"
- Section 13.1.3.4, "The getURI Method"
- Section 13.1.3.5, "The getAttributes Method"

13.1.3.1 The getItems Method

The getItems method returns the files and folders stored starting at a particular location in the repository. This method enables you to publish content in forms, tables, and hierarchical trees. Using this method, you can also create navigation lists and buttons.

Table 13–1 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 getItems.</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 13–2.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<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>
13.1.3.2 The search Method

The search method enables you to create a simple search by name pattern or keyword.

Table 13–3 describes the parameters of the search method.

**Table 13–3 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>
<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>

The search method returns the attributes described in Table 13–4.

**Table 13–4 Return Attributes of the search Method**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<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>

13.1.3.3 The advancedSearch Method

The advancedSearch method enables you to perform an advanced search by creating a set of search criteria out of any available attribute.

Table 13–5 describes the parameters of the advancedSearch method.

**Table 13–5 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>
</tbody>
</table>
The advancedSearch method returns the attributes described in Table 13–6.

Table 13–6  Return Attributes of the advancedSearch Method

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<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>

13.1.3.4 The getURI Method

The getURI method returns the URI attribute, which is the direct access URL of the file or folder. Its path parameter describes the path to the object. You can use this method to create links to content and to inline content in your page. The getURI method returns the URI attribute.

13.1.3.5 The getAttributes Method

The getAttributes method returns the list of attributes and their values for a given file or folder. Its path parameter describes the path to the object.

Table 13–7 describes the attributes that the getAttributes method returns.

Table 13–7  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>

13.2 Configuring Content Repository Connections

Content repository connections are required to access the repository content to be published on JSF pages. This section describes how to configure content repository connections based on Oracle Content Server, Oracle Portal, and file system adapters.

Content Presenter can connect to and retrieve content available in Oracle Content Server content repositories. With JCR data controls, you can connect to Oracle Content Server, Oracle Portal, and file systems.

Connections can be created under Application Resources in the Application Navigator and under IDE Connections in the Resource Palette, as described in Section 3.10, "Accessing Connection Wizards".

This section includes the following subsections:
13.2.1 How to Create a Content Repository Connection Based on the Oracle Content Server Adapter

This section describes how to configure an Oracle Content Server-based content repository connection. To successfully perform the subsequent steps, you must use Oracle Content Server release 10.1.3.4.1 or 10.1.3.5 (required for Content Presenter). For information about how to configure Oracle Content Server for Oracle WebCenter, see "Oracle Content Server Prerequisites" in Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter.

To create an Oracle Content Server-based repository connection:

1. In Oracle JDeveloper, open your WebCenter application.
2. In the Application Resources panel, right-click Connections and choose New Connection, then Content Repository.
3. In the Create Content Repository Connection dialog, specify a connection name for the connection; for example, MyOCSConnection.
   
   Tip: You can choose between creating application-specific connections in the Application Resources panel and creating common connections in the Resource Palette (IDE Connections).

4. Select Oracle Content Server from the Repository Type box.
5. Select Set as primary connection for Documents service, if you intend to make it the default connection for the Documents service. Selecting this option writes a configuration to the adf-config.xml file that identifies this connection as the default connection for the Documents service. If a Document Library task flow is used without any connectionName input parameter, then this connection is used. For information about the Documents service, see Chapter 14, "Integrating the Documents Service".


6. Under Configuration Parameters, enter values for the parameters, as shown in Table 13–8 and Figure 13–1.

Table 13–8 Configuration Parameters for Oracle Content Server

<table>
<thead>
<tr>
<th>Configuration Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>RIDC Socket Type</td>
<td>Determines whether the client library connects on the Content Server listener port or the Web server filter. It accepts socket, socketssl, or web.</td>
</tr>
<tr>
<td></td>
<td>- socket: Uses an intradoc socket connection to connect to the Oracle Content Server. The client IP address must be added to the list of authorized addresses in the Oracle Content Server.</td>
</tr>
<tr>
<td></td>
<td>- socketssl: Uses an intradoc socket connection to connect to the Oracle 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.</td>
</tr>
<tr>
<td></td>
<td>- web: Uses an HTTP(S) connection to connect to the Oracle Content Server.</td>
</tr>
<tr>
<td></td>
<td>Table 13–9 includes more information on the configuration parameters required for each RIDC socket type.</td>
</tr>
<tr>
<td>Server Host Name</td>
<td>Host name of the system where the Oracle Content Server is running. For example: mycontentserver.mycompany.com.</td>
</tr>
<tr>
<td>Content Server Listener Port</td>
<td>Port of the server specified in the Server Host Name field. For example: 4444.</td>
</tr>
<tr>
<td>URL of the Web Server Plugin</td>
<td>If the RIDC socket type is web, then the URL must be in this format: http://host_name/web_root/plugin_root. For example: <a href="http://mycontentserver/cms/idcplg">http://mycontentserver/cms/idcplg</a></td>
</tr>
<tr>
<td>Admin Username</td>
<td>The user name with administrative rights for the Oracle Content Server instance. It defaults to sysadmin.</td>
</tr>
<tr>
<td>Admin Password</td>
<td>The password for the Oracle Content Server admin user. For example: password. This is needed for web connection 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>Cache Invalidation Interval</td>
<td>The polling interval used by the Oracle Content Server service provider interface (SPI) to check for cache invalidations. This interval is calculated in minutes and defaults to 0. The value 0 disables cache invalidation. The minimum value is 2 minutes.</td>
</tr>
</tbody>
</table>
Table 13–9 Oracle 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>
</tr>
</thead>
<tbody>
<tr>
<td>Server Host Name</td>
<td>Not applicable</td>
<td>Optional</td>
<td>Optional</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Defaults to local host.</td>
<td>Defaults to local host.</td>
</tr>
<tr>
<td>Content Server</td>
<td>Not applicable</td>
<td>Port specified for the incoming provider in the server.</td>
<td>Port specified for the ssl incoming provider in the server.</td>
</tr>
<tr>
<td>Listener Port</td>
<td></td>
<td>Defaults to 4444</td>
<td>Defaults to 4444</td>
</tr>
<tr>
<td>URL of the Web Server Plugin</td>
<td>Mandatory</td>
<td>Not applicable</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Admin Username</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 sysadmin</td>
</tr>
<tr>
<td>Admin Password</td>
<td>Mandatory</td>
<td>Not applicable</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Key Store and Private Key</td>
<td>Not applicable</td>
<td>Not applicable</td>
<td>Mandatory</td>
</tr>
<tr>
<td>Identity Propagation</td>
<td>Not supported at runtime</td>
<td>Supported</td>
<td>Supported</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>
</tr>
<tr>
<td>External Application</td>
<td>Mandatory</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>
</tr>
<tr>
<td>User Name and Password</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The password is not verified during the test connection operation.</td>
<td>The password is not verified during the test connection operation.</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 13.2.6.1, "What You Should Know About Using Identity Propagation and External Application Authentication Methods" for information. If you select the Identity Propagation authentication method, then you must configure secure socket layer (SSL) in Oracle Content Server. For information, see the section "Oracle Content Server - Security Considerations" in Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter.

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.

**Note:** If the Specify login credentials for the current JDeveloper session checkbox is selected, then the credentials you entered are used. If the checkbox is not selected, then the connection is tested using External Application credentials (if they exist), otherwise null credentials are used.

If you have selected External Application for authentication and also specified either public or shared credentials, then you can leave these fields blank, as the public or shared credentials can be used to login at design time. However, if you have selected External Application but have not specified public or shared credentials, then you must specify user name and password here.

10. Click OK.

11. In the Application Resources panel, expand the repository connection you just created. The Connection Credentials dialog displays.
12. Enter the **User Name** and **Password** for the Oracle Content Server connection and click **OK**. The connection displays under the Application Resources panel.

### 13.2.2 How to Create a Content Repository Connection Based on the Oracle Portal Adapter

This section describes how to create a content repository connection based on the Oracle Portal adapter. Before creating a repository connection, see Section 13.2.6, "What You May Need to Know When Creating a Repository Connection". You can use this connection to configure a content data control that will enable you to add content from the Oracle Portal repository to JSF pages.

To create an Oracle Portal repository connection:

1. In Oracle JDeveloper, open your WebCenter 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**.
4. Select **Oracle Portal** from the **Repository Type** box, as shown in Figure 13–2.

#### Figure 13–2 Content Repository Connection - Oracle Portal

![Create Content Repository Connection](image)

5. Select **Set as primary connection for Documents service**, if you intend to make it the default connection for the Documents service. Selecting this option writes a configuration to the `adf-config.xml` file that identifies this connection as the default connection for the Documents service. If a document library task flow is used without any `connectionName` input parameter, then this connection is used. For information about the Documents service, see Chapter 14, "Integrating the Documents Service".

---

**Note:** The Connection Credentials dialog displays if you do not save the credentials, that is, you do not select the **Specify login credentials for the current JDeveloper session** checkbox, or if you do not specify an External Application service that uses public or shared credentials.

**Tip:** You can choose between creating application-specific connections in the Application Resources panel and creating common connections in the Resource Palette (IDE Connections).
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 13–10 and Figure 13–3.

---

### Table 13–10 Parameters for Creating Oracle Portal-based Content Data Control

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Driver</strong></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>
<tr>
<td><strong>Host Name</strong></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><strong>JDBC Port</strong></td>
<td>Value to identify the TCP/IP port.</td>
</tr>
<tr>
<td><strong>SID</strong></td>
<td>Unique system identifier of an Oracle database instance.</td>
</tr>
<tr>
<td><strong>Service Name</strong></td>
<td>The service name for an Oracle database instance.</td>
</tr>
</tbody>
</table>
e. Click OK.

7. In the Create Content Repository Connection dialog, in the Login Timeout (ms) field, specify the time in milliseconds. This timeout determines how long the application must wait when trying to establish a session with the content repository. If the network connection or the content repository server to which you are trying to connect is slow, then consider overriding the default value specified in this field.


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.
10. In the Application Resources panel, expand the repository connection you just created. The Connection Credentials dialog displays.

**Note:** The Connection Credentials dialog displays if you do not save the credentials, that is, you do not select the **Specify login credentials for the current JDeveloper session** checkbox, or if you do not specify an External Application service that uses public or shared credentials.

11. Enter the user name and password for Oracle Portal connection and click **OK**. The connection displays under the Application Resources panel.

**See Also:** To create a data control using this connection, perform the procedure described in **Section 13.5.1, "How to Configure a Content Repository Data Control"**.

### 13.2.3 How to Create a Content Repository Connection Based on the File System Adapter

This section describes the procedure to create a content repository connection based on the file system adapter.

To create a File System repository connection:

1. In Oracle JDeveloper, open your WebCenter application.
2. In the Application Resources panel, right-click **Connections** and choose **New Connection** and then **Content Repository**.
3. In the Create Content Repository Connection dialog, enter a name for the connection; for example, **MyConnection**.

   **Tip:** You can choose between creating application-specific connections in the Application Resources panel and creating common connections in the Resource Palette (IDE Connections).

4. Select **File System** from the **Repository Type** box.
5. Select **Set as primary connection for Documents service**, if you intend to make it the default connection for the Documents service. Selecting this option writes a configuration to the **adf-config.xml** file that identifies this connection as the
default connection for the Documents service. If a document library task flow is used without any `connectionName` input parameter, then this connection is used. For information about the Documents service, see Chapter 14, "Integrating the Documents Service".

**Note:** To be able to set a connection as primary connection for the Documents service, you must select the **Application Resources** option in **Create Connection In**. This is because connections created as IDE connections cannot be used directly; they must be added to applications.

6. Under **Configuration Parameters**, 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 13.6, "Integrating Content Using Content Data Controls: Examples".
- When you work on a UNIX system, the File System adapter inherits the case-sensitive file name characteristic of UNIX systems. So, on UNIX systems, you must ensure that references to files follow the same case as that used in the original file names. For example, suppose the `Test.html` file was created on a Microsoft Windows system. When you reference this file on a Linux system, you must ensure that you use `Test.html`, and not `test.html` or `TEST.html`.

7. Leave the **Login Timeout (ms)** field blank.

8. Click **Test Connection** to check whether you have entered the connection details correctly. You should see a **Success!** status, as shown in Figure 13–4.
9. Click OK.

10. In the Application Resources panel, expand the repository connection you just created.

13.2.4 How to Create a Content Repository Connection Based on the Oracle WebCenter Adapter for Microsoft SharePoint

The Oracle WebCenter adapter for Microsoft 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 13.2.6.3, "What You Should Know About Oracle WebCenter Adapter for Microsoft SharePoint" before configuring and using Oracle WebCenter adapter for Microsoft SharePoint.

This section includes the following subsections:

- Section 13.2.4.1, “Creating a Content Repository Connection Based on the Oracle WebCenter Adapter for Microsoft SharePoint”
- Section 13.2.4.2, “Mapping Microsoft SharePoint Content and Services”

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

To create a repository connection based on Oracle WebCenter adapter for Microsoft SharePoint:

1. In Oracle JDeveloper, open your WebCenter application.
2. In the Application Resources panel, right-click **Connections** and choose **New Connection**, then **Content Repository**.

3. In the Create Content Repository Connection dialog, specify a connection name for the connection; for example, **MySPConnection**.

   **Tip:** You can choose between creating application-specific connections in the Application Resources panel and creating common connections in the Resource Palette (IDE Connections).

4. Select **JCR SharePoint Adapter** from the **Repository Type** box.

   **Note:** This option displays only if the extension for Oracle WebCenter adapter for Microsoft SharePoint is installed.

5. Select **Set as primary connection for Documents service**, if you intend to make it the default connection for the Documents service. Selecting this option writes a configuration to the **adf-config.xml** file that identifies this connection as the default connection for the Documents service. If a Document Library task flow is used without any **connectionName** input parameter, then this connection is used. For information about the Documents service, see Chapter 14, "Integrating the Documents Service".

   **Note:** To be able to set a connection as primary connection for the Documents service, you must select the **Application Resources** option in **Create Connection In**. This is because connections created as IDE connections cannot be used directly, they must be added to applications.

6. Under **Configuration Parameters**, enter the **SharePoint URL**, that is, the web address of the SharePoint site to which you want to connect. For example, **http://mysharepoint.mycompany.com**, as shown in **Figure 13–1**.
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 13.2.6.3, “What You Should Know About Oracle WebCenter Adapter for Microsoft 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 13.2.6.1, "What You Should Know 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 service that uses public or shared credentials.

13. Enter the **User Name** and **Password** for this connection and click **OK**. The connection displays under the Application Resources panel. This step is optional.

### 13.2.4.2 Mapping Microsoft SharePoint Content and Services

The Oracle WebCenter adapter for Microsoft 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 the path: `/sp:Site/sp:RootWeb/sp:Lists/Shared Documents/sp:Files`, as shown in **Figure 13–6**.
Chapter 13: Configuring Content Repository Connections

Figure 13–6  The Shared Documents Library Path for a SharePoint Services v3 Team Site

For the Moss 2007 site created during installation, the Documents Center Document Library appears at path:
/sp:Site/sp:RootWeb/sp:Webs/Docs/sp:Lists/Documents/sp:Files, as shown in Figure 13–7. This path reflects the structure in which the Documents Center is a subsite in the SharePoint main site.

Files are JCR nodes of primary type nt:file, and of mix-in type sp:File. Folders are JCR nodes of primary type nt:folder and of mix-in type sp:Folder. The sp: mix-ins are similar to nt:unstructured in their definitions.

Figure 13–7  The Documents Center Document Library Path for Moss 2007 Site

13.2.5 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 13–8. You can double-click folders and files to view them.
You can use repository connections to create JCR data controls that enable integration of the repository content with JSF pages. See Section 13.5, "Configuring Content Data Controls for JCR Adapters" and Section 13.6, "Integrating Content Using Content Data Controls: Examples" for information. These connections can also be consumed through the Documents service task flows. See Chapter 14, "Integrating the Documents Service" for information.

### 13.2.6 What You May Need to Know When Creating a Repository Connection

This section includes:

- Section 13.2.6.1, "What You Should Know About Using Identity Propagation and External Application Authentication Methods"

- Section 13.2.6.2, "What You Should Know About Oracle Portal"

- Section 13.2.6.3, "What You Should Know About Oracle WebCenter Adapter for Microsoft SharePoint"

#### 13.2.6.1 What You Should Know About Using Identity Propagation and External Application Authentication Methods

The Create Content Repository Connection dialog provides the following options for authentication methods:

- **Identity Propagation**: If you select this option, no credentials are passed to the repository. The repository connector instead uses the current user’s identity as determined from the Java security context. This must only be used when the application and the repository use the same identity store to authenticate users. To apply this authentication method, it is best to configure security for your application using the Configure ADF Security wizard since repositories may support only authenticated users or provide only limited access to the public or guest user. See Section 13.5.4, "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 37.2, "Working with External Applications".

The External Application service allows different types of credentials to be associated with a connection:
Configuring Content Repository Connections

- Public credentials are used whenever an application is not secured, or the user has not yet logged in.
- Shared credentials are used for any authenticated user.
- If shared credentials are not specified, then mapped credentials can be used to obtain a result similar to identity propagation when the application and content repository do not use the same user store. In that case the external application service provides runtime screens through which users can provide their credentials for accessing the content repository. Those credentials are securely stored and used for any future connection that this user tries to establish.

If you intend to configure the External Application service, then click New to launch the Register External Application wizard and do the following:

a. Specify name of the external application.

b. In the Login URL field, enter the URL of the external application; for example, http://content-server.mycompany.com/idc/

c. Then, under Authentication Details, select Basic from the list. Leave the fields under Login Details blank.

   This option is selected when click through login is not required. If click through login is enabled, then you must select another option based on the authentication method used in the repository.

d. Accept the default values and click Next.

e. Enter shared credentials that can be used at design time and runtime. Do not specify shared credentials, if users must specify their own credentials.

f. Enter public credentials that can be used at design time and runtime. Specify public credentials only if you intend public users to view documents or contents.

   Click Finish.

13.2.6.2 What You Should Know About Oracle Portal

The following should be considered while using the Oracle Portal adapter:

- To use Oracle Portal-based content data control capabilities, you can install Oracle Portal release 10.1.4.1, 10.1.4.2, or 11. If you use the Oracle Portal release 10.1.4.1, then you must update it with the required patches. Consult Oracle Application Server Release Notes for Microsoft Windows (for the required platform) for release 11.1.1.0.0 to know the exact patch number.

- Only file, image, imagemap, and text item types and custom types based on these item types are supported.

- The portal:container page type and its extensions are supported.
Content is always exposed in the default language of the page group. For example, if there are three page groups with different default languages, then the content displays only for those default languages and not for any translations that exist.

13.2.6.3 What You Should Know About Oracle WebCenter Adapter for Microsoft SharePoint

Consider the following while configuring and using Oracle WebCenter Adapter for Microsoft 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**: The following Microsoft SharePoint versions are supported:
  - Microsoft Office SharePoint Server (MOSS) 2007 SP2
  - Microsoft Windows SharePoint Services (WSS) version 3 SP2
- **Versioning Settings**: The only supported Microsoft SharePoint 2007 Document Library versioning settings are:
  - Require Check Out : No
  - Content Approval : No
  - Document Version History : No versioning

If any other Versioning Settings are configured, the adapter will not function correctly. For example, upload operations will fail if Require Check Out is set to yes, and new versions or documents will have restricted visibility, if document version history or content approval are enabled.

- **Session-Based Memory Cache**: For performance reasons, Oracle WebCenter adapter for Microsoft SharePoint maintains a session-based memory cache. A custom WebCenter 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 the native SharePoint web services, as the adapter does not rely on custom additional server-side components. Consequently:
  - Full text search only returns items that are indexed by the SharePoint server.
  - Set of searchable columns is limited.
  - A fixed set of node types is supported in searches.
  - Some searches require additional network round trip as the adapter does not receive the full information necessary to map results to JCR from the web services.

The adapter exposes content from a Microsoft SharePoint server in a hierarchical way which is similar to the way by which the content is exposed by the native web interface. The items of the hierarchy have a primary type of either nt:folder, nt:file, or a specialization of these. Additional SharePoint specific characteristics are in some instances added by means of mixins, that is,
sp:Folder, sp:File, and so on. The SharePoint specific name space, which is identified by the sp prefix in these examples, is used for items controlled by the SharePoint server. Items controlled by the user - webs, documents, list items and the like - are in the default name space. The adapter allows write access to SharePoint document libraries. That is, it allows folders and files to be created within document libraries in correspondence with creating folders or uploading files to the native SharePoint web interface. Examples which demonstrate this hierarchical mapping are described in the next point.

■ Site Structure: The structure mapping of a SharePoint site is as follows:

  - /sp:Site: The root element of the site.
  - /sp:Site/sp:RootWeb: The root web element.
  - /sp:Site/sp:RootWeb/sp:Lists: The path to the root of all lists in the site.
  - /sp:Site/sp:RootWeb/sp:Lists/Shared Documents/sp:Files: The path to the start of the file and folder hierarchy within a document library. In this example, the Shared Documents library is shown. All site document libraries are mapped under /sp:Site/sp:RootWeb/sp:Lists, each mapped by their own document library name.
  - /sp:Site/sp:RootWeb/sp:Lists/Shared Documents/sp:Files/myFiles: An example of a folder within the sites Shared Documents document library.
  - /sp:Site/sp:RootWeb/sp:Webs/mySubSite: A site's subsites are all mapped under /sp:Site/sp:RootWeb/sp:Webs. This example shows the path to a subsite named mySubSite. The subsite may itself contain Document Libraries.
  - The adapter does not support the write access to a SharePoint Document Library with settings that require one or more mandatory SharePoint columns.

■ Search and indexing:

  - JCR and native Microsoft SharePoint search: The Oracle WebCenter adapter for Microsoft 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 Microsoft SharePoint applies a client-side filtering on result sets to ensure that the correct constraint on URL is applied. The adapter's LIKE limit parameter controls this feature, and if this limit is not set, it defaults to applying a 64 character limit. The parameter can be set to 0 to disable client-side filtering. The parameter can also be set to some other positive value to apply a different character limit. However, this limit can only be increased if the SharePoint instance supports LIKE tests on URLs greater than 64 characters.
– **Creation and Modification timestamps**: When displaying the contents of a SharePoint Document Library, the adapter reports the Creation and Modification timestamps as the times that the document was created on or last updated in the library. In the case of some document types, for example Microsoft Office documents, the native SharePoint search evaluates against the document metadata stored within each document. This can be different from the times when the document was uploaded or updated in the Document Library, and therefore, it appears to give misleading search results.

– **Combining search keywords**: The adapter does not support combining search keywords in an OR expression.

– **Search by Last Modifier**: The adapter does not support search by Last Modifier.

– **Supported JCR search operators and properties**: The adapter supports the following:

  * Relational operators for comparison: =, >=, >, <=, LIKE, <>, IS NULL, IS NOT NULL. Predicates, descendants, and self operators (/) are supported on the last location step.

  * Properties for comparison: Size, Author, Modified, URL, Title, IsDocument, ContentClass, SiteName, Description, FileName, jcr:data, jcr:mimeType, jcr:created, jcr:lastModified.

### 13.2.7 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 13–9. The Edit Repository Connection dialog displays, as shown in Figure 13–10.
2. Change the appropriate parameters.
3. Test the connection and click OK.

13.2.8 How to Edit a WebCenter Application-Specific Content Repository Connection

Application-specific content repository connections exist under the Application Resources panel.

To edit a WebCenter 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 13–11. The Edit Content Repository Connection dialog displays, as shown in Figure 13–12.
2. Change the appropriate parameters. Depending on the repository type, see the earlier sections that describe how to create repository connections. For example, for Oracle Content Server, see Section 13.2.1, "How to Create a Content Repository Connection Based on the Oracle Content Server Adapter".

3. Test the connection and click OK.

13.2.9 How to Use an Existing Repository Connection for a New WebCenter Application

You can use an existing repository connection for any WebCenter 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 WebCenter 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 13–13.

   Tip: Alternatively, right-click the connection and choose Add to Application. The connection is displayed under the Application Resources panel.

You can now configure a data control for your new application from this repository connection, as described in Section 13.5.1, "How to Configure a Content Repository Data Control".
13.3 Adding Content to a JSF Page

You can display the folders and files from a connected content repository on a JSF page in a WebCenter application in several different ways, as shown in Table 13–11:

<table>
<thead>
<tr>
<th>Content Type</th>
<th>Content Display Selections</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content repository root folder</td>
<td>ADF Go Link, Documents service task flow (Document Manager, Document List View, or Recent Documents)</td>
</tr>
<tr>
<td>Subfolders</td>
<td>ADF Go Link, Documents service task flow (Document Manager, Document List View, Content Presenter)</td>
</tr>
<tr>
<td>Documents of various types (XML, PDF, JAVA, TXT, DOC, XLS, HTM)</td>
<td>ADF Go Link, ADF Inline Frame, Content Presenter task flow</td>
</tr>
<tr>
<td>Images (PNG, JPG, GIF)</td>
<td>ADF Go Link, ADF Image, ADF Inline Frame, Content Presenter task flow</td>
</tr>
</tbody>
</table>

For more information about adding content in one of the Documents service task flows (Document Manager, Document List View, Recent Documents, or Content Presenter), see Section 14.2.3, "Adding Documents Service Task Flows at Design Time."

When you add content in a Content Presenter task flow, you can select a template to display the content. The selected template may be one of the built-in templates provided with WebCenter, or a custom template that has been developed for your organization. To develop custom display templates, see Section 13.4, "Creating Custom Templates for Displaying Content Using Content Presenter."

13.4 Creating Custom Templates for Displaying Content Using Content Presenter

When you add content to a page using the Content Presenter task flow, you can select a template to display the content. For more information about display templates, refer to Section 13.1.2, "Overview of Content Presenter Display Templates and the Templates Registry".

If the out-of-the-box display templates (see Table 14–4 and Table 14–5) do not meet your needs, you can define custom templates for the content that you want to display in your custom WebCenter application, or for selection by end users at runtime.

This section contains the following information:

- Section 13.4.1, "How To Define Custom Display Templates"
- Section 13.4.2, "Performance Considerations for Custom Display Templates"
- Section 13.4.3, "What Happens at Runtime"

13.4.1 How To Define Custom Display Templates

Depending on your needs, the approach you take to defining custom 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.
After defining custom display templates, you must register them specifically for Content Presenter in the templates-registry.xml file.

This section includes about defining and registering custom display templates:

- Section 13.4.1.1, "EL Expressions for Content Item Nodes"
- Section 13.4.1.2, "Defining Single Content Item Display Templates"
- Section 13.4.1.3, "Defining Multiple Content Item Display Templates"
- Section 13.4.1.4, "Registering Templates in the Templates Registry"
- Section 13.4.1.5, "Importing Custom Display Templates for WebCenter Spaces Applications"
- Section 13.4.1.6, "Testing Custom Display Templates"

### 13.4.1.1 EL Expressions for Content Item Nodes

This section provides descriptions of the EL expressions that you can use in your custom display template definitions to retrieve and display specific information about a content item node. Use the EL expressions described in the following tables as you define your custom display templates using the tags described in Section 13.4.1.2, "Defining Single Content Item Display Templates" and Section 13.4.1.3, "Defining Multiple Content Item Display Templates":

- Table 13–12, "EL Expressions for a Content Item Node"
- Table 13–13, "EL Expressions for Content Item Node Properties"
- Table 13–14, "EL Expressions for Content Item Node Property Values"
- Table 13–15, "EL Expressions for Content Item Node or Property Icons"
- Table 13–16, "EL Expressions for Content Item Node URLs"

**Note:** As you build your EL expressions, JDeveloper auto-suggests valid completions as you type.

#### Table 13–12  EL Expressions for a Content Item Node

<table>
<thead>
<tr>
<th>EL Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>#{node.createdBy}</code></td>
<td>Returns the user name of the node’s creator.</td>
</tr>
<tr>
<td><code>#{node.createdDate}</code></td>
<td>Returns the node’s creation date.</td>
</tr>
<tr>
<td><code>#{node.hasParentNode}</code></td>
<td>Returns true if the current node has a valid parent node ID, or a node that has no parent.</td>
</tr>
</tbody>
</table>
Use the EL expression described in Table 13–13 and Table 13–14 to perform actions on content item node properties and property values. To determine the names of the properties defined for a given content type, refer to Step 4 of Section 13.4.1.2, "Defining Single Content Item Display Templates."

### Table 13–13 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>#{node.propertyMap['myProp'].isBinary}</td>
<td>Returns true if the current property is of type Property.BINARY.</td>
</tr>
</tbody>
</table>
Table 13–13  (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'].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>
</tbody>
</table>
Table 13–13 (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'].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>
<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. Example: #{node.propertyMap['myRefNode'].linkAsNode.primaryProperty.url.renderUrl}</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>
</tbody>
</table>

Table 13–14 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>
<tr>
<td>#{node.propertyMap['myProp'].value.longValue}</td>
<td>Returns the value of this property as java.lang.Long.</td>
</tr>
</tbody>
</table>
13.4.1.2 Defining Single Content Item Display Templates

Examples of single content items for which you may want to create custom 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 view of an article).
- Different versions of a similar item (such as Press Releases that may be formatted differently for different groups, using a different set of properties).

The definition of a single content item display template uses the JSP tags listed in Table 13–17.

<table>
<thead>
<tr>
<th>JSP Tag</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
</table>
| contentTemplateDef | Required. Defines a single content item template.                          | `<dt:contentTemplateDef var="node">
|               | Attributes:                                                                 | `<af:outputText value="#{node.name}" />
|               | var - Specifies the content node that will be rendered by this display      | `</dt:contentTemplateDef>`                                            |
|               | template. In the template definition code, you can use the EL expressions  |                                                                       |
|               | described in Section 13.4.1.1, "EL Expressions for Content Item Nodes" to |                                                                       |
|               | retrieve required information about the node.                               |                                                                       |
| renderProperty | Optional. Retrieves and renders the string value(s) of the specified node  | <!-- Handling of text-based primary properties (HTML, text, etc.). -->   |
|               | property inline.                                                            | `<dt:contentTemplateDef var="node">
|               | Attributes:                                                                 | `<cmf:renderProperty id="rp1" name="#{node.primaryProperty.name}" node="#{node}"/>
|               | id - Identifies this component. This value must be unique within the       |                                                                       |
|               | closest parent component that is a naming container.                       |                                                                       |
|               | name - Specifies the name of the property to get. If this is a system      |                                                                       |
|               | property (cm_createdBy, cm_createdDate, cm_modifiedDate, and cm_path) then |                                                                       |
|               | the value on the node will be used. If not specified, then the            |                                                                       |
|               | primaryProperty will be used if defined.                                   |                                                                       |
|               | node - Specifies the node to use. This value can either be a bound        |                                                                       |
|               | attribute to a managed bean, or a request attribute variable.              |                                                                       |
|               | blockSize - Specifies the size of the blocks in bytes, to read the bytes   |                                                                       |
|               | of a binary property. Default is 2048 bytes.                               |                                                                       |
|               | startIndex - Specifies the index (from 0) in the document's bytes at      |                                                                       |
|               | which to start printing. Defaults to 0.                                   |                                                                       |
|               | endIndex - Specifies the index (from 0) in the document's bytes at which   |                                                                       |
|               | to stop printing. Defaults to value of blockSize.                          |                                                                       |
|               | rendered - Specifies whether or not this component should be rendered      |                                                                       |
|               | (during Render Response Phase), or processed on any subsequent form submit.|                                                                       |
|               | The default value is true.                                                 |                                                                       |
To define a display template for a single content item:

1. In JDeveloper, create a JSFF file:
   a. From the File menu, choose New.
   b. In the New Gallery dialog, expand Web Tier, select JSF, then JSF Page Fragment.
   c. Click OK.
   d. In the Create New JSP Page Fragment dialog, enter a name for the display template file, and save the file in the appropriate directory:
      - If you are developing a display template for a WebCenter Spaces application, the JSFF file must be in the following directory under your project’s ViewController/Web Content directory:
        /oracle/webcenter/content/templates/custom
        (On the file system, this path is:
         yourprojectdir/ViewController/public_html/oracle/webcenter/content/templates/custom).
      - If you are developing a display template for a custom WebCenter application, or to be added to WebCenter Spaces through a shared library (as described in the white paper "Extending WebCenter Spaces" available on the Oracle Technology Network), save your JSFF file in any directory structure under your project’s ViewController/Web Content directory.

2. Select View, then Component Palette to open the Component Palette.

3. From the dropdown list at the top of the Component Palette (Figure 13–14):
Creating Custom Templates for Displaying Content Using Content Presenter

- Select **WebCenter Content Display Templates** for a list of display template tags.
- Select **WebCenter Content Management Faces** for the renderProperty tag.

Figure 13–14  **Content Display Template Tags in Component Palette**

4. In Source view, drag and drop the required display template tags from the Component Palette onto the page to define your template. Refer to Table 13–17 for information about each tag and required parameter values.

As described in Section 13.1.2, "Overview of Content Presenter Display Templates and the Templates Registry", a display template defines how a content repository item should render on a page. Each content item is associated with a specific content type defined in the Oracle Content Server repository. A content type defines the properties of the content item, and is identified in the templates registry file when you register the template (see Table 13–19).

As a custom display template developer, you will need to know the names of the properties defined for the associated content type so that you can define how to display the selected content item(s) on the page. One way to determine the properties for the existing content types defined in Oracle Content Server is to use the Content Presenter Configuration dialog in WebCenter Spaces. Refer to the section "Displaying Content Using the Content Presenter Task Flow" in Oracle Fusion Middleware User's Guide for Oracle WebCenter to open the Content Presenter Configuration dialog:

a. On the **Content** page of the Content Presenter Configuration dialog, select **Results of a Query** to a view a list of content types.

b. Select the **Content Type** for the content item that you wish to display.

c. From the **Add Fields** list, select **Filters** to add the Additional Query Filters section to the **Content** page, then click **Add** to list all the properties that are defined for the selected content type (Figure 13–15).
Creating Custom Templates for Displaying Content Using Content Presenter

Figure 13–15 Identifying Content Type Properties

Example 13–1 and Example 13–2 show sample single content item display template definitions.

Example 13–1 Sample display template definition for the structure of Press Releases generated by the Marketing Department (mkt-dept-press-release-list-item.jsff)

This template handles a Press Release that uses a property named dDocName to describe the heading of the Press Release document, and xDestinationUrl to describe the location of the document.

Note: When referencing the content type properties, you must include the leading letter d or x as shown to reference the internal Oracle Content Server property name. The prefix d identifies a system property, and the prefix x identifies a user property.
Creating Custom Templates for Displaying Content Using Content Presenter

Example 13–2 Sample display template definition for the structure of Press Releases generated by the Product Management department (prod-mgmt-press-release-list-item.jsff)

This template 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.

13.4.1.3 Defining Multiple Content Item Display Templates

Examples of multiple content items for which you may want to create custom 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.

The definition of a multiple content item display template uses the JSP tags listed in Table 13–18.
<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><code>&lt;dt:contentListTemplateDef var=&quot;nodes&quot;&gt; &lt;af:iterator value=&quot;#{nodes}&quot; var=&quot;node&quot;&gt; &lt;af:outputText value=&quot;#{node.name}&quot; /&gt; &lt;/af:iterator&gt; &lt;/dt:contentListTemplateDef&gt;</code></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. In the template definition code, you can use the EL expressions described in Section 13.4.1.1, &quot;EL Expressions for Content Item Nodes&quot; to retrieve required information about the node.</td>
<td></td>
</tr>
<tr>
<td>contentListTemplate</td>
<td>Optional. Nested under contentListTemplateDef.</td>
<td><code>&lt;dt:contentListTemplateDef var=&quot;nodes&quot;&gt; &lt;!-- Reuse the default bulleted list view, but indent it with a &lt;blockquote&gt; --&gt; &lt;f:verbatim&gt; &lt;blockquote&gt; &lt;/f:verbatim&gt; &lt;/dt:contentListTemplateDef&gt;</code></td>
</tr>
<tr>
<td></td>
<td>Calls another multiple item template.</td>
<td></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;f:verbatim&gt; &lt;blockquote&gt; &lt;/f:verbatim&gt;</td>
</tr>
<tr>
<td></td>
<td>category - Required. Specifies the target template category in templates-registry.xml.</td>
<td><code>&lt;dt:contentListTemplate nodes=&quot;#{nodes}&quot; category=&quot;oracle.webcenter.content. templates.default.category&quot; view=&quot;oracle.webcenter.content. templates.default.list.bulleted&quot;/&gt;</code></td>
</tr>
<tr>
<td></td>
<td>view - Required. Specifies the target view name in templates-registry.xml.</td>
<td><code>&lt;f:verbatim&gt; &lt;/blockquote&gt; &lt;/f:verbatim&gt;</code></td>
</tr>
<tr>
<td></td>
<td>id - Optional. Specifies the JSF component ID.</td>
<td><code>&lt;f:verbatim&gt; &lt;/dt:contentListTemplateDef&gt;</code></td>
</tr>
<tr>
<td></td>
<td>rendered - Optional. Specifies whether the component should be rendered.</td>
<td></td>
</tr>
</tbody>
</table>
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 13.4.1.2, "Defining Single Content 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 13–18 for information about each tag and required parameter values.

Example 13–3 shows a sample multiple content item display template definition.

Example 13–3 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"), which is registered in templates-registry.xml.

```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:iterator rows="0" var="node" varStatus="iterator" value="#{nodes}">
    <dt:contentTemplate node="#{node}" view="templates.pressrelease.listitem"
      nodesHint="#{nodes}"/>
  </af:iterator>
</dt:contentListTemplateDef>
```

---

**Table 13–18 (Cont.) Content Display Template Tags for Multiple Content Items**

<table>
<thead>
<tr>
<th>JSP Tag</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
</table>
| contentTemplate | Optional. Nested under contentListTemplateDef. Calls a single item template. | `<dt:contentListTemplateDef var="nodes">
  <af:iterator rows="0" var="node" varStatus="iterator" value="#{nodes}">
    <dt:contentTemplate node="#{node}" view="templates.pressrelease.listitem"
      nodesHint="#{nodes}"/>
  </af:iterator>
</dt:contentListTemplateDef> |
| nodesHint     | Optional. When the display template is called in an iterating component, allows the pre-inclusion of all possible templates. This attribute is usually required when contentTemplate is used inside a contentListTemplateDef tag. |                                                                         |
| view          | Optional. Specifies the target view name in templates-registry.xml.         |                                                                         |
| id            | Optional. Specifies the JSF component ID.                                     |                                                                         |
| rendered      | Optional. Specifies whether the component should be rendered.                |                                                                         |
13.4.1.4 Registering Templates in the Templates Registry

All display templates must be registered in the Content Presenter templates registry file, templates-registry.xml.

- For custom WebCenter application, the templates registry file is located in YourAppDir/mds/oracle/webcenter/content/templates/registry/.
- For WebCenter Spaces applications, you must export the templates registry file from MDS to your file system before you can edit it, and then import the modified registry back into MDS.

The templates-registry.xml file includes two display template sections:

- The <content-templates> section contains the display template definitions for single content items. See Table 13–19.
- The <content-list-templates> section contains the display template definitions for multiple content items. See Table 13–20.

| Table 13–19  Templates Registry Tags for Single Content Item Display Templates |
|---------------|--------------------------------------------------------------------------------|
| JSP Tag        | Description                                                                    |
| content-templates | Defines the section for display templates for single content items.             |
| content-repository | Nested under content-templates tag.                                              |
| content-type   | Nested under content-repository tag.                                            |
|                | Defines the single content item templates that will display on the Template page in the Content Presenter Configuration dialog, dependent on the selected content item(s). |

**Attributes:**

- name - Specifies the name of specific content repository where content resides, or * to browse all connected content repositories.

- defaultView - Identifies the view to display if user does not select any template.

- name - Specifies the name of the content type defined in Oracle Content Server with which this template can be used (the display template defines how to handle the properties of the content type). Use * to allow the template to be used for any content type. This value is shown in the Content Presenter Configuration dialog in the Content Type list when the selection of content items is based on Results of a Query. Refer to the section "Displaying Content Using the Content Presenter Task Flow" in Oracle Fusion Middleware User’s Guide for Oracle WebCenter for the steps to use Results of a Query to select content items to display in a Content Presenter task flow.
Creating Custom Templates for Displaying Content Using Content Presenter

**Table 13–19 (Cont.) Templates Registry Tags for Single Content Item Display Templates**

<table>
<thead>
<tr>
<th>JSP Tag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>view</td>
<td>Nested under content-type tag. Definitions one or more sets of names, descriptions, and JSFF files for single content item templates that display on the Template page in the Content Presenter Configuration dialog.</td>
</tr>
<tr>
<td></td>
<td><strong>Attributes:</strong></td>
</tr>
<tr>
<td></td>
<td>id - Specifies the identifier that is called from a JSFF template definition file to be used when the name is selected by an end user in the Content Presenter Configuration dialog. This value must be unique among all others in this templates registry file.</td>
</tr>
<tr>
<td>name</td>
<td>Nested under view tag. Specifies the text string for an item in the Template View list in the Content Presenter Configuration dialog.</td>
</tr>
<tr>
<td>description</td>
<td>Nested under view tag. Specifies the description in the Template View list in the Content Presenter Configuration dialog when the end user selects the name (and hint text when the cursor is positioned on the item).</td>
</tr>
<tr>
<td>template-def</td>
<td>Nested under view tag. Specifies the JSFF file that defines the template to display the content item, and processes the content according to the referenced view.</td>
</tr>
</tbody>
</table>

**Table 13–20 Templates Registry Tags for Multiple Content Item Display Templates**

<table>
<thead>
<tr>
<th>JSP Tag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>content-list-templates</td>
<td>Defines the section for display templates for multiple content items.</td>
</tr>
<tr>
<td>content-repository</td>
<td>Nested under content-list-templates tag. Definitions name/description pairs for the multiple content item templates that display on the Template page in the Content Presenter Configuration dialog.</td>
</tr>
<tr>
<td></td>
<td><strong>Attributes:</strong></td>
</tr>
<tr>
<td></td>
<td>name - Specifies the name of specific content repository where content resides, or * to browse all connected content repositories.</td>
</tr>
<tr>
<td>category</td>
<td>Nested under content-repository tag. Defines name/description pairs for the multiple content item templates that display on the Template page in the Content Presenter Configuration dialog.</td>
</tr>
<tr>
<td></td>
<td><strong>Attributes:</strong></td>
</tr>
<tr>
<td></td>
<td>defaultView - Identifies the view to display if the end user does not select any template.</td>
</tr>
<tr>
<td></td>
<td>id - Specifies a unique identifier to distinguish this category from all others in this templates registry file (for example, com.my.company.unique.category).</td>
</tr>
<tr>
<td>name</td>
<td>Nested under category tag. Specifies the text string for an item in the Template Category list in the Content Presenter Configuration dialog.</td>
</tr>
<tr>
<td>description</td>
<td>Nested under category tag. Specifies the description in the Template Category list in the Content Presenter Configuration dialog when the end user selects the name. This description also provides the hint text when the cursor is positioned on the item.</td>
</tr>
</tbody>
</table>
To register your custom display templates in templates-registry.xml:

1. Open templates-registry.xml in an editor.
   - If you are creating a display template for a WebCenter Spaces application, export the latest registry file using the exportMetadata command. For example:
     ```
     exportMetadata('webcenter','WLS_Spaces','/myapps/dir/mds','/oracle/**/templates-registry.xml')
     ```
   
   **Note:** For detailed syntax and examples, see "exportMetadata" in Oracle Fusion Middleware WebLogic Scripting Tool Command Reference.

   - For custom WebCenter applications, you can easily add the registry file to your project by adding a Content Presenter task flow onto a page of the project. This action forces the registry file to be added to your project, and you can then delete the page or the Content Presenter task flow.

2. (Recommended) Save a backup copy of the registry file prior to making edits to it.

3. Edit the registry file to add your content display templates:
   - In the <content-templates> section, register the display template(s) that are defined for a single content item. Refer to Table 13–19 for information about tags and attributes.
   - In the <content-list-templates> section, register the display template(s) for displaying multiple content items, which may call the single item definitions. Refer to Table 13–20 for information about tags and attributes.

---

<table>
<thead>
<tr>
<th>JSP Tag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>view</td>
<td>Nested under category tag. Defines one or more sets of names, descriptions, and JSFF files for multiple content item templates associated with the selected category that display on the Template page in the Content Presenter Configuration dialog. Attributes: id - Specifies the identifier that is called from a JSFF template definition file to be used when the name is selected by an end user in the Content Presenter Configuration dialog. Different &lt;content-type&gt; elements can have views with the same id value if the &lt;content-type&gt; elements have unique name attributes. When a view of a non-unique id is requested, the content type will be used to determine which view (therefore which JSFF file) will be used. See Example 13–5 to see how the same view id maps to two different content types and JSFF template definition files.</td>
</tr>
<tr>
<td>name</td>
<td>Nested under view tag. Specifies the text string for an item in the Template View list in the Content Presenter Configuration dialog.</td>
</tr>
<tr>
<td>description</td>
<td>Nested under view tag. Specifies the description beneath the Template View list when the end user selects the name (and hint text when the cursor is positioned on the item).</td>
</tr>
<tr>
<td>template-def</td>
<td>Nested under view tag. Specifies the JSFF file that defines the template to display the content item, and processes the content according to the referenced view.</td>
</tr>
</tbody>
</table>
Creating Custom Templates for Displaying Content Using Content Presenter

If your application integrates the Documents service (see Chapter 14, "Integrating the Documents Service"), the information you specify here is reflected in the Content Presenter Configuration dialog at runtime, as described in Table 13–19 and Table 13–20.

---

**Note:** Content Presenter defines its own resource bundle for the out-of-the-box display templates. For custom display templates, you can specify a resource bundle at the top of the registry file to use your own `bundleKey` attributes to achieve localization of your custom display template names and descriptions. You can also remove the `bundleKey` attribute from any custom display template registry entry to reduce the amount of logging.

---

Example 13–4 and Example 13–5 show a sample registry file definition. Example 13–5 shows a scenario where you want to create a list of Press Releases, but there are two different kinds of Press Releases defined in your content repository, each one using different property names for the short title and long body. You can create two custom display templates with similar formatting, then create a custom multiple item Press Release listing template to choose the correct single item template for each of the two kinds of Press Releases based on their content type.

**Example 13–4  High-level structure of the templates registry file**

```xml
<display-templates
xmlns="http://xmlns.oracle.com/webcenter/content/templates/registry">
  ...single content item definitions
</content-templates>
<content-list-templates>...
multiple content item definitions
</content-list-templates>
</display-templates>
```

**Example 13–5  Detailed structure of the templates registry file**

```xml
<display-templates xmlns="http://xmlns.oracle.com/webcenter/content/templates/registry">
  <!- register templates for single content items ->
  <content-templates>
    <content-repository name="mkt15">
      <content-type
        defaultView="mycorp.content.templates.pressrelease.listitem"
        name="IDC:MktgPressRelease">
        <view id="mycorp.content.templates.pressrelease.listitem">
          <name>
            Marketing Press Releases
          </name>
          <description>
            This is a view of Marketing Department Press Releases
          </description>
          <template-def>
            /oracle/webcenter/content/templates/custom/
            mkt-dept-press-release-list-item.jsff
          </template-def>
        </view>
      </content-type>
    </content-repository>
  </content-templates>
</display-templates>
```
<content-type
defaultView="mycorp.content.templates.pressrelease.listitem"
name="IDC:ProdMgmtPressRelease">
<view id="mycorp.content.templates.pressrelease.listitem">
  <name>
  Product Management Press Releases
  </name>
  <description>
  This is a view of Product Management Press Releases
  </description>
  <template-def>
  /oracle/webcenter/content/templates/custom/
  prod-mgmt-press-release-list-item.jsff
  </template-def>
</view>
</content-type>
</content-repository>

<!-
register templates for multiple content items ->
<content-list-templates>
  <category defaultView="mycorp.content.templates.pressrelease.listing"
id="mycorp.content.category">
    <name>
    MyCorp Templates
    </name>
    <description>
    MyCorp custom templates that display multiple content items.
    </description>
    <view id="mycorp.content.templates.pressrelease.listing">
      <name>
      Company Press Release List
      </name>
      <description>
      List view for MyCorp's Press Releases.
      </description>
      <template-def>
      /oracle/webcenter/content/templates/custom/
      press-release-list-view.jsff
      </template-def>
    </view>
    <view id="mycorp.content.templates.pressrelease.table">
      <name>
      Company Press Release Table
      </name>
      <description>
      Items display in a sortable table which includes the document name,
      date created, and date modified.
      </description>
      <template-def>
      /oracle/webcenter/content/templates/custom/
      press-release-tabular-view.jsff
      </template-def>
    </view>
  </category>
</content-list-templates>
13.4.1.5 Importing Custom Display Templates for WebCenter Spaces Applications

If you are creating display templates for a WebCenter Spaces application, import the JSFF files you have created and the revised templates registry file into MDS using the importMetadata command. For example:

```
importMetadata('webcenter', 'WLS_Spaces', '/myapps/dir/ViewController/public_html', '/oracle/webcenter/content/templates/custom/*.jsff')
importMetadata('webcenter', 'WLS_Spaces', '/path_to_app_dir/mds', '/oracle/**/templates-registry.xml')
```

Note: For detailed syntax and examples, see "importMetadata" in Oracle Fusion Middleware WebLogic Scripting Tool Command Reference.

This process assumes that the template development environment is running within the same environment as the running WebCenter Spaces environment. The importMetadata and exportMetadata tasks are designed to import and export to the local WebCenter Spaces file system, even if the commands are executed from a remote WebLogic Scripting Tool (WLST) client. If remote template development is desired, it is the responsibility of a WebCenter Spaces administrator to import and export templates and the registry to the WebCenter Spaces file system and manually transition the templates to and from template developers.

13.4.1.6 Testing Custom Display Templates

Prior to integrating with a running production application, test your new display templates and changes to the templates registry file in a local custom application and a staged WebCenter Spaces application. See Section 14.2.3, "Adding Documents Service Task Flows at Design Time" and Section 14.2.4.1, "Content Presenter Task Flow Parameters and Out-of-the-Box Display Templates".

13.4.2 Performance Considerations for Custom 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 custom 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 13–21 shows whether or not node properties are loaded immediately upon node retrieval, by retrieval mechanism.

### Table 13–21 Loading of Node Properties By Node Retrieval Mechanism

<table>
<thead>
<tr>
<th>OCS Global Profile Properties</th>
<th>Loaded When Node Is First Retrieved?</th>
<th>GET BY PARENT ID (“Contents Under a Folder”)</th>
<th>SEARCH (“Results of a Query”)</th>
<th>GET BY UUID (“Single Content Item” and “List of Items”)</th>
</tr>
</thead>
<tbody>
<tr>
<td>VaultFileSize</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dCheckoutUser</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dCreateDate</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dDocAccount</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dDocAuthor</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dDocName</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dDocTitle</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dDocType</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dFormat</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dID</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dInDate</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dIsCheckedOut</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dOutDate</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dReleaseDate</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dReleaseState</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dRevClassID</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dRevLabel</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dRevRank</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dRevisionID</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dSecurityGroup</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dStatus</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dWebExtension</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dWorkflowState</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>idcPrimaryFile</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>idcRenditions</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>xCollectionID</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>xComments</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>xForceFolderSecurity</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>xHidden</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>xInhibitUpdate</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>xReadOnly</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
</tbody>
</table>

#### 13.4.3 What Happens at Runtime

After you have defined and registered custom display templates, 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 14.2.3, "Adding Documents Service Task Flows at Design Time" and Section 14.2.4.1, "Content Presenter Task Flow Parameters and Out-of-the-Box Display Templates").

If you integrate the Documents Service in your application (see Chapter 14, "Integrating the Documents Service"), end users of your application can select content and your custom display templates in the Content Presenter Configuration dialog to include content on editable pages of the application, as described in "Displaying Content Using the Content Presenter Task Flow" in Oracle Fusion Middleware User’s Guide for Oracle WebCenter. To understand how Content Presenter identifies which display templates to expose in the Content Presenter Configuration dialog, see Section 13.4.3.1, "Identifying Display Templates for Selected Content Items."

13.4.3.1 Identifying Display Templates for Selected Content Items

At runtime, after a user selects content in the Content Presenter Configuration dialog, Content Presenter checks the templates registry 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.

For **single content item** templates, Content Presenter generates a list of valid templates using the following precedence order:

1. Check the templates registry 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 templates registry 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 templates registry for any reason, Content Presenter will automatically select a "closest match" template to display the content item(s) using the precedence order above.

---

As an illustration of the flexibility of Content Presenter, Example 13–5 shows a templates registry file with two template definitions with the same `view_id` (`mycorp.content.templates.pressrelease.listitem`). One view specifies a template definition file defined for Marketing Press Releases (`mkt-dept-press-release-list-item.jsff`), the other view specifies a
template definition file defined for Product Management Press Releases (prod-mgmt-press-release-list-item.jsff). To determine which template definition file to use, Content Presenter checks the name attribute of the parent <content-type> tag, mapping this value to the content type of the current content item. If it matches, Content Presenter displays that template for selection in the Content Presenter Configuration dialog; if it does not match, Content Presenter moves on to the second view with the same view id. Again, if the name attribute of that view’s parent <content-type> tag matches the content type of the current content item, Content Presenter displays that template for selection in the Content Presenter Configuration dialog. If no match is found, then the checks continue as defined by the precedence order to build a list of valid templates.

13.5 Configuring Content Data Controls for JCR Adapters

This section describes how to create a content data control based on a content repository connection:

- Section 13.5.1, "How to Configure a Content Repository Data Control"
- Section 13.5.2, "What Happens When You Configure a Content Repository Data Control"
- Section 13.5.3, "How to Edit a Content Repository Data Control"

13.5.1 How to Configure a Content Repository Data Control

The procedure to create content repository data controls is the same irrespective of the types of the connections that you use to create data controls. If you have not created a connection to your content repository, then see Section 13.2, "Configuring Content Repository Connections". In this section, a content data control is created using an Oracle Content Server-based connection.

To create a content data control using an existing content repository connection:

1. In the Application Navigator, select the Model 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 Model 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.
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 Oracle 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.

13.5.2 What Happens When You Configure a Content Repository Data Control

When the data control is successfully configured, it shows under the Data Controls panel. Expand it in the Data Controls panel to see the hierarchical list of methods, parameters, and operations for the new data control, as shown in Figure 13–16. For detailed information on using the Data Control panel, see Chapter titled "Using Oracle ADF Model in A Fusion Web Application" of Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework.
Table 13–22 describes the icons and object hierarchy of a data control.

Table 13–22 The Data Controls Panel Icons and Object Hierarchy

<table>
<thead>
<tr>
<th>Icon</th>
<th>Name</th>
<th>Description</th>
<th>Used to Create...</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Data Control Icon" /></td>
<td>Data Control</td>
<td>Represents a data control. You cannot use the data control itself to create UI components, but you can use any of the child objects listed under it. Depending on how your business services were defined, there may be multiple data controls.</td>
<td>Serves as a container for the other objects, and is not used to create anything</td>
</tr>
<tr>
<td><img src="image" alt="Method Icon" /></td>
<td>Method</td>
<td>Represents an operation in the data control or one of its exposed structures that may accept parameters, perform some business logic and optionally return single value, a structure or a collection of those.</td>
<td>Command components</td>
</tr>
<tr>
<td><img src="image" alt="Method Return Icon" /></td>
<td>Method Return</td>
<td>Represents an object that is returned by a custom method. The returned object can be a single value or a collection. A method return appears as a child under the method that returns it. The objects that appear as children under a method return can be attributes of the collection, other methods that perform actions related to the parent collection, and operations that can be performed on the parent collection.</td>
<td>For single values: text fields and selection lists. For collections: forms, tables, trees, and range navigation components. When a single-value method return is dropped, the method is not invoked automatically by the framework. A user either has to also create an invoke action as an executable, or drop the corresponding method as a button to invoke the method.</td>
</tr>
<tr>
<td><img src="image" alt="Attribute Icon" /></td>
<td>Attribute</td>
<td>Represents a discrete data element in an object (for example, an attribute in a row). Attributes appear as children under the collections or method returns to which they belong. Only the attributes that were included in the view object are shown under a collection. If a view object joins one or more entity objects, that view object’s collection will contain selected attributes from all of the underlying entity objects.</td>
<td>Label, text field, date and selection list components</td>
</tr>
</tbody>
</table>
The following files (Figure 13–17) are created under the Model project:

- **DataControls.dcx**: Oracle JDeveloper creates this file the first time a data control is created. This file lists all the Oracle ADF data controls created under the current project. This file is required to initialize the data control.

- **datacontrolname.xml**: This file includes attributes, accessors, and operations of a data control.

- **advancedSearch_return.xml**: This file includes the return type definition for the advancedSearch method.

- **getAttributes_return.xml**: This file includes the return type definition for the getAttributes method.

- **getItems_return.xml**: This file includes the return type definition for the getItems method.

- **getURI_return.xml**: This file includes the return type definition for the getURI method.

- **search_return.xml**: This file includes the return type definition for the search method.

- **Return.xml**: This file defines the standard operations on collections.
### 13.5.3 How to Edit a Content Repository Data Control

This section describes a generic procedure to edit content data controls that you configured as described in Section 13.5, "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 13–18. The Edit Content Repository Data Control dialog displays, as shown in Figure 13–19.

**Figure 13–18  Edit Data Controls**

**Figure 13–19  Edit Content Repository Data Control Dialog**

3. To select a different connection, use the **Connection Name** list.
Integrating Content Using Content Data Controls: Examples

4. To add a connection, click the **Create new content repository connection** icon to display the Create Content Repository Connection dialog. Depending on the type of connection that you intend to create, see relevant sections in Section 13.2, "Configuring Content Repository Connections" for information about its configuration parameters.

5. To edit an existing connection, click the **Edit selected content repository connection** icon to display the Edit Content Repository Connection dialog. Depending on the type of connection that you intend to modify, see relevant sections in Section 13.2, "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.

13.5.4 Securing a Content Repository Data Control

You can enable security for your content repository connections. For information, see Chapter 37, "Securing Your WebCenter Application".

13.6 Integrating Content Using Content Data Controls: Examples

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 the JSPX page. When a method is dropped, its ADF Faces tag is added to the source of the JSPX 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 JSPX page. The destination of the `af:goLink` is set to the EL expression `#{bindings.URI.inputValue}`. This EL expression ties the destination value of the `af:goLink` to the binding container's URI value.

In the page definition file, `methodIterator` is added to the `executables` element, and `methodAction` is added to the `bindings` element. The `methodIterator` and `methodAction` elements define which data control and method is to be used. The `NamedData` attribute of the `methodAction` defines what input parameter values should be provided to this method call. The path and type parameters are common across all data control methods. However, `NamedData` for search and `advancedSearch` includes additional parameters that are unique to those methods. For example, in the `advancedSearch` method, the `NamedData` can also be used to define the search predicates, whether the search is recursive, and so on.

The page definition includes the `AttrNames` element, which defines the attributes of the items available in the bindings container to ADF. These items are based on default attributes and custom attributes that were defined when creating the data control.

For detailed information on using the Data Control panel and working with page definition files, see the chapter "Using Oracle ADF Model in A Fusion Web Application" of Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework.

The examples in this section use the following data control methods:

- `getURI` to add textual and clickable image links to the repository folder.
- `getItems` to publish the repository content in an ADF table and tree.
- `search` and `advancedSearch` to display those items of the repository that match the search criteria.
This section includes the following subsections:

- Section 13.6.1, "How to Publish Content As Links"
- Section 13.6.2, "What Happens at Runtime"
- Section 13.6.3, "How to Publish Content in a Table"
- Section 13.6.4, "What Happens at Runtime"
- Section 13.6.5, "How to Publish Folder Content in a Tree"
- Section 13.6.6, "What Happens at Runtime"
- Section 13.6.7, "How to Add Search Capabilities to Content Repositories"
- Section 13.6.8, "What Happens at Runtime"
- Section 13.6.9, "What You May Need to Know When Using Search Capabilities"

13.6.1 How to Publish Content As Links

This section describes how to create hyperlinks to files stored in a file system and convert them into textual and image links. You use the ADF Go Link and the getURI method to create textual links and the Image of ADF Faces to create image links.

This section includes the following procedures:

- Section 13.6.1.1, "Publishing Content As a Textual Link"
- Section 13.6.1.2, "Creating a Clickable Image to Link to a Document"

Before you begin:

1. Configure a repository connection as described in Section 13.2, "Configuring Content Repository Connections".
2. Configure a data control as described in Section 13.5, "Configuring Content Data Controls for JCR Adapters".
3. Create a JSPX page as described in Section 3.3, "Creating WebCenter Application-Enabled Pages."

13.6.1.1 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 .jspx page 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 13–20.
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 13–21.

If this is the first time you have dropped a node onto the page, then the Edit Action Binding dialog displays.

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 13–22. 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 13–23.

9. Right-click your page and choose Run. The page appears in your browser window with the new link, as shown in Figure 13–24.

10. Click the link to check that the correct file is displayed.

In Section 13.6.1.2, "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:

**Note:** To grant edit, personalize, customize, and view permissions at the attribute level, see Section 13.5.4, "Securing a Content Repository Data Control".
1. In the Application Navigator, right-click the .jspx 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 13–25.

Figure 13–25 Page Data Binding Definition

2. Go to the source view of the page definition.

3. In the executables element, add another methodIterator and change the methodIterator id and value of Binds, as shown in Bold in the following example:

   <executables>
     <variableIterator id="variables"/>
     <methodIterator Binds="getURI.result" DataControl="MyDataControl"
                    RangeSize="25" BeanClass="model.MyDataControl.getURI_return"
                    id="getURIIterator"/>
     <methodIterator Binds="getURI1.result" DataControl="MyDataControl"
                    RangeSize="25" BeanClass="model.MyDataControl.getURI_return"
                    id="getURIIterator1"/>
   </executables>

4. In the Bindings element, add another methodAction and change the methodAction id, ReturnName, and NDValue, as shown in Bold in the following example:

   <methodAction id="getURI"
                RequiresUpdateModel=true Action="invokeMethod" MethodName="getURI"
                IsViewObjectMethod="false" DataControl="MyDataControl"
                InstanceName="MyDataControl"
                ReturnName="MyDataControl.methodResults.getURI_MyDataControl_getURI_result">
     <NamedData NDName="path" NDValue="/PlasmaNews.html" NDType="java.lang.String"/>
   </methodAction>
   <methodAction id="getURI1"
                RequiresUpdateModel=true Action="invokeMethod" MethodName="getURI"
                IsViewObjectMethod="false" DataControl="MyDataControl"
                InstanceName="MyDataControl"
                ReturnName="MyDataControl.methodResults.getURI1_MyDataControl_getURI1_result">
     <NamedData NDName="path" NDValue="/FusionOrderDemoLogo.jpg" NDType="java.lang.String"/>
   </methodAction>

5. In the Bindings element, add another attributeValues tag to specify the new Id, as shown in bold in the following example:

   <attributeValues
             IterBinding="getURIIterator"
             id="URI"
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 13–26.

![Oracle JDeveloper Context Menu for the getURI method](image)

**Figure 13–26** 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 13–27 shows the new link, FOD Logo, in addition to the Plasma News link that was added in the first part of Section 13.6.1.1, "Publishing Content As a Textual Link".
You can add as many links as required by following these steps.

13.6.1.2 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 .jspx 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 13.6.1.1, "Publishing Content As a Textual Link", in the Structure window, right-click af:goLink, choose Insert Inside af:goLink, and then ADF Faces, as shown in Figure 13–28. The Insert ADF Faces Item dialog displays.

Figure 13–28 Insert Inside af:goLink - ADF Faces
3. From the Select an ADF Faces item to create list, select Image, as shown in Figure 13–29.

**Figure 13–29  Insert ADF Faces Item**

4. Click OK. The Insert Image dialog displays.

5. Browse to the image file you intend to display as the image link; for example, plasma.jpg.

6. Click Finish.

7. Right-click your page and choose Run. Figure 13–30 shows a sample output.

**Figure 13–30  ADF Object Image Link in a Browser**

13.6.2 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 JSPX page. For information about the page lifecycle, see the chapter "Understanding the Fusion Page Lifecycle" of *Oracle*.  

Integrating Content Using Content Data Controls: Examples
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.

### 13.6.3 How to Publish Content in a Table

In this section, the `getItems` data control method is used to publish file and folder information in a table. This section describes the following procedures:

- Section 13.6.3.1, "Displaying Files and Folders in Read-Only Format"
- Section 13.6.3.2, "Displaying the Name Attribute As a Go Link"
- Section 13.6.3.3, "Configuring a Table to Show Only Files"

**Before you begin:**

1. Configure a repository connection as described in Section 13.2, "Configuring Content Repository Connections".
2. Configure a data control as described in Section 13.5, "Configuring Content Data Controls for JCR Adapters".
3. Create a JSPX page as described in Section 3.3, "Creating WebCenter Application-Enabled Pages."

#### 13.6.3.1 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 13–31.

*Figure 13–31  The Return node of the `getItems` method*
3. To create a table that lists every file and folder available through this data control, drop the Return node on to the page or under af:form in the Structure window. From the Create menu, choose Table and then ADF Read-only Table, as shown in Figure 13–32.

Figure 13–32 The Create Context Menu for getItems Method

4. In the Edit Table Columns dialog, as shown in Figure 13–33, 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 13–33 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 13–34. You must enter a leading slash (/). To
modify or delete this path later, click the arrow icon next to the node in design mode and choose **Go to Binding** from the context menu.

Leave the `type` parameter blank. This implies that the table displays both files and folders.

**Figure 13–34 Edit Action Binding**

![Edit Action Binding](image)

6. Click **OK**. You should now see a table on the JSPX page that looks like **Figure 13–35**.

**Figure 13–35 Read-Only Table for Publishing Folder Content**

<table>
<thead>
<tr>
<th>Name</th>
<th>Location</th>
<th>URL</th>
<th>Type</th>
<th>Last Modified Date</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>#(..name)</code></td>
<td><code>#(..path)</code></td>
<td><code>goLink1</code></td>
<td><code>#(..primaryType)</code></td>
<td><code>#(..lastModified)</code></td>
</tr>
<tr>
<td><code>#(..name)</code></td>
<td><code>#(..path)</code></td>
<td><code>goLink1</code></td>
<td><code>#(..primaryType)</code></td>
<td><code>#(..lastModified)</code></td>
</tr>
<tr>
<td><code>#(..name)</code></td>
<td><code>#(..path)</code></td>
<td><code>goLink1</code></td>
<td><code>#(..primaryType)</code></td>
<td><code>#(..lastModified)</code></td>
</tr>
</tbody>
</table>

**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 13–36** 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).

13.6.3.2 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 13.6.3.1, "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 13–37), 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

```xml
<af:column sortProperty="name" sortable="false"
    headerText="#{bindings.Return.hints.name.label}" id="c2">
    <af:outputText value="#{row.name}" id="ot1"/>
</af:column>
```

with an af:goLink as shown in bold:

```xml
af:column sortProperty="name" sortable="false"
    headerText="#{bindings.Return.hints.name.label}" id="c4">
```
3. In the Structure window, right-click af:column - URL and select Delete.

4. Right-click the page in the Application Navigator and choose Run. You should see a list of hyperlinked file and folder names like the one shown in Figure 13–38. This figure shows the name attribute as a link. Clicking a link in the URI column opens the respective file or folder and shows its contents.

**Figure 13–38  Folder Content Displayed as Hyperlinks**

<table>
<thead>
<tr>
<th>Name</th>
<th>Location</th>
<th>Type</th>
<th>Last Modified</th>
</tr>
</thead>
<tbody>
<tr>
<td>Archive</td>
<td>Archive</td>
<td>int-file</td>
<td>2/25/2009</td>
</tr>
<tr>
<td>graph.jpg</td>
<td>graph.jpg</td>
<td>int-file</td>
<td>2/25/2009</td>
</tr>
<tr>
<td>plasma.jpg</td>
<td>plasma.jpg</td>
<td>int-file</td>
<td>2/25/2009</td>
</tr>
<tr>
<td>Welcome.html</td>
<td>Welcome.html</td>
<td>int-file</td>
<td>2/25/2009</td>
</tr>
<tr>
<td>grey_box_icon.png</td>
<td>grey_box_icon.png</td>
<td>int-file</td>
<td>2/25/2009</td>
</tr>
<tr>
<td>warning.jpg</td>
<td>warning.jpg</td>
<td>int-file</td>
<td>2/25/2009</td>
</tr>
<tr>
<td>question_mark.png</td>
<td>question_mark.png</td>
<td>int-file</td>
<td>2/25/2009</td>
</tr>
<tr>
<td>PCOM/Arch_webCenter.doc</td>
<td>PCOM/Arch_webCenter.doc</td>
<td>int-file</td>
<td>2/25/2009</td>
</tr>
<tr>
<td>posimages.jpg</td>
<td>posimages.jpg</td>
<td>int-file</td>
<td>2/25/2009</td>
</tr>
<tr>
<td>FusionOrderDemoFace.jpg</td>
<td>FusionOrderDemoFace.jpg</td>
<td>int-file</td>
<td>2/25/2009</td>
</tr>
</tbody>
</table>
| Thumbnails            | Thumbnails     | int-file | 2/25/2009     

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 13–39.

**Figure 13–39  Folder Contents**

Current folder: /Archive

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 13–40, and click OK.
4. Run the page to view the output. Figure 13–41 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.

In the next section, the Name column will be configured to show only files.

### 13.6.3.3 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 `.jspx` page and choose Go to Page Definition.

   **Note:** The RangeSize binding setting, which is used to control the number of items displayed on a page, is set to 10 by default in the page definition file. You can change it, as required, in the Property Inspector.

2. In the Overview tab, double-click `getItems` under Bindings. The Edit Action Binding dialog displays.
3. The type options are `nt:file` and `nt:folder`. To specify the display of only files, enter `nt:file` under the Value column, as shown in Figure 13–42, and click OK.

**Figure 13–42 Display Files Only**

![Edit Action Binding](image)

4. Now run the page. Figure 13–43 shows only files in the single-column table because the column type is `nt:file`.

**Figure 13–43 Files Displayed in a Single-Column Table**

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>graph.jpg</td>
<td></td>
<td></td>
</tr>
<tr>
<td>nintendo_wii.jpg</td>
<td></td>
<td></td>
</tr>
<tr>
<td>iPodSpeakerNews.html</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PlasmaNews.html</td>
<td></td>
<td></td>
</tr>
<tr>
<td>package_icon.jpg</td>
<td></td>
<td></td>
</tr>
<tr>
<td>plasma.jpg</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WIINews.html</td>
<td></td>
<td></td>
</tr>
<tr>
<td>gray_box_icon.jpg</td>
<td></td>
<td></td>
</tr>
<tr>
<td>warning.jpg</td>
<td></td>
<td></td>
</tr>
<tr>
<td>question_mark.jpg</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FODforWebCenter.doc</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ipodspeaker.jpg</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FusionOrderDemoLogo.jpg</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Thumbs.db</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

13.6.4 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:

```xml
<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.

### 13.6.5 How to Publish Folder Content in a Tree

In this section, you will use the `getItems` method to publish content in a hierarchal tree format. This section describes the following procedures:

- Section 13.6.5.1, "Displaying Files and Folders in Read-Only Format"
- Section 13.6.5.2, "Displaying File Names As Hyperlinks"

**Before you begin:**

1. Configure a repository connection as described in Section 13.2, "Configuring Content Repository Connections".
2. Configure a data control as described in Section 13.5, "Configuring Content Data Controls for JCR Adapters".
3. Create a JSPX page as described in Section 3.3, "Creating WebCenter Application-Enabled Pages."

#### 13.6.5.1 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 13–44.
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 13–45.

If this is the first time you have dropped a node onto the page, the Edit Action Binding dialog displays.

4. To create a tree that displays everything under the base path, enter the slash (/) for the path parameter, as shown in Figure 13–46. 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 URI and primary type, 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 13–47, which enables the tree to find its child items.
8. Click **OK**. A tree displays in the JSPX page that looks like Figure 13–48.

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 13–49 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`).
Integrating Content Using Content Data Controls: Examples

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.

13.6.5.2 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 nodeStamp and delete af:outputText-#{node}.
2. Right-click nodeStamp and choose Insert inside nodeStamp and then ADF Faces.
3. In the Insert ADF Faces Item dialog, from the Select an ADF Faces item to create list, select Switcher, and click OK.

A Switcher is added under nodeStamp in the Structure window, as shown in Figure 13–50.

4. Double-click af:switcher to display the Property Inspector, if it is not displayed.
5. Under the **Common** tab, in the **FacetName** field, enter the expression 
#{node.primaryType}.

6. 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.

7. 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 13–51**.

**Figure 13–51 Switcher Component with Two Facets**


9. In the Insert ADF Faces Item dialog, select **Output Text** and click **OK**.

10. 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}.


12. In the Insert ADF Item dialog, select **Go Link** from the **Select the item to be created** box, and click **OK**.

13. In the Structure window, double-click **af:goLink - goLink** to display the Property Inspector, if it is not already displayed.

14. Under the **Common** tab, in the **Text** field, enter the 
#{node.name} expression.

15. In the **Destination** field, enter the expression 
#{node.URI}.

16. From the **TargetFrame** list, select **_blank**.

17. Run the page. **Figure 13–52** 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.
13.6.6 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.

13.6.7 How to Add Search Capabilities to Content Repositories

With the help of two examples, this section describes how to add simple and advanced search capabilities for the integrated content. The simple search enables users to search for the content based on name or content fragments in specific locations. The advanced search enables users to search by attribute values of the content.

This section contains the following:

- **Section 13.6.7.1, "Adding Simple Search Capabilities"
- **Section 13.6.7.2, "Adding Advanced Search Capabilities"

**Before you begin:**

1. Configure a repository connection as described in **Section 13.2, "Configuring Content Repository Connections"**.
2. Configure a data control as described in Section 13.5, "Configuring Content Data Controls for JCR Adapters".

3. Create a JSPX page as described in Section 3.3, "Creating WebCenter Application-Enabled Pages."

13.6.7.1 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 .jspx 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 .jspx 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 13–53

Figure 13–53 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 13–54 displays.

Figure 13–54 Table with Four Columns - search

7. Run this page and specify / for search_path and %jpg% for namePattern. All .jpg files stored in the root of your repository display, as shown in Figure 13–55.
13.6.7.2 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 .jspx 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 this example the page is called advancedSearch.jspx.

2. Double-click the .jspx page to open it.

3. In the Component Palette, select ADF Faces.

4. From the list of ADF Faces 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. Under view, double-click AdvancedSearch.java to open it.

8. Add the following import declarations. These declarations are required for the getPredicates method, which will be added in the next step.

   import java.util.ArrayList;
   import java.util.Calendar;
   import java.util.Date;
   import java.util.List;
   import oracle.vcr.datacontrol.search.Predicate;
   import oracle.vcr.datacontrol.search.Operator;

9. Add the following method to enable the advanced search based on last modified dates of the items stored in the repository:

   public List<Predicate> getPredicates() {
       ArrayList<Predicate> predicates = new ArrayList<Predicate>();
       if (id1.getValue() != null && !id1.getValue().equals('')) {
           Calendar cal = Calendar.getInstance();
           cal.setTime((Date)id1.getValue());
           predicates.add(new Predicate("jcr:content/jcr:lastModified"
10. From the Data Controls panel, drag the `advancedSearch` node onto the `.jspx` 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. Select the arrow next to the `Value` field for predicates 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 13–56, and click OK.

Figure 13–56 Variables Dialog - predicates


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 13–57.
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 `.jspx` 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 13–58**.

**Figure 13–57**  **Advanced Search - Design View**

**Figure 13–58**  **Advanced Search Results**
13.6.8 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 JSPX 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 JSPX 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 getPredicates 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')]

13.6.9 What You May Need to Know When Using Search Capabilities

Consider the following points while adding search capabilities:

- How certain operations work depends on the implementation of the adapter and the underlying repository. While read and query operations are similar, full text search works differently. Another example is, the file system and Oracle Portal adapters do not support search based on the primaryType attribute. The only supported way to search based on type is through the element (*, type) construct.

- If you use the Oracle Portal adapter, then the behavior of search functionality varies depending on whether Oracle Text is enabled or not. If Oracle Text is disabled, then the search is performed in the Oracle Portal content metadata. With Oracle Text turned on, all indexed content is searched, which includes the contents of files. This also applies to Oracle Content Server. That is, Oracle 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.
14

Integrating the Documents Service

Oracle WebCenter enables content integration through:

- The Documents service, which enables users to view and manage documents and other types of content in your organization's content repositories.

  The Documents service includes the Content Presenter task flow, which enables end users to select content in a variety of ways and then display those items using available display templates.

- Content repository data controls, which enable read-only access to a content repository, and maintain tight control over the way the content displays in a custom WebCenter application.

This chapter describes how to integrate the Documents service in custom WebCenter applications to provide end users with a user-friendly interface to manage, display, and search documents at runtime. For more information about managing and including content in your custom WebCenter applications, see:

- Chapter 13, "Integrating Content" to configure content repository connections that provide access to decentralized content, and learn how to create custom display templates to integrate and publish decentralized content in your WebCenter application using Content Presenter, as well as how to use Java Content Repository (JCR) controls to enable read-only access to a content repository.

- "Managing Content Repositories" in the Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter to configure and manage content repositories used by WebCenter applications.

- "Working with the Documents Service" in the Oracle Fusion Middleware User’s Guide for Oracle WebCenter to work with the Documents service and task flows at runtime in WebCenter applications.

This chapter includes the following sections:

- Section 14.1, "Introduction to the Documents Service"
- Section 14.2, "Basic Configuration for the Documents Service"
- Section 14.3, "Advanced Information for the Documents Service"

14.1 Introduction to the Documents Service

The Documents service provides features for accessing, adding, and managing files and folders; configuring file and folder properties; searching file and folder content in your organization’s content repositories; and displaying content from a content repository on a page in a predefined template using the Content Presenter task flow.
The Documents service allows end users of your application to manage folders and documents from your company's content repositories in two ways:

- Add any single folder or file in a content repository to a page. Individual files can be added to a page as links, images, or inline frames (depending on the file type).
- Add a Documents service task flow to a page:
  - Content Presenter task flow
  - Document Manager task flow
  - Document List View task flow
  - Recent Documents task flow

For detailed information about each Documents service task flow and what capabilities they provide to end users of your application, see "What You Should Know About the Documents Service Task Flows" in Oracle Fusion Middleware User's Guide for Oracle WebCenter.

---

**Note:** You can see the Documents service in action in the sample application, as described in Chapter 2, "Introduction to the WebCenter Sample Application."

---

To use the Documents service, you must have a 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 14.2.1, "Setting Up Connections."

### 14.2 Basic Configuration for the Documents Service

This section describes the steps to add the Documents service to your application:

- Section 14.2.1, "Setting Up Connections"
- Section 14.2.2, "Adding the Documents Service to the Oracle Composer Catalog"
- Section 14.2.3, "Adding Documents Service Task Flows at Design Time"
- Section 14.2.4, "Documents Service Task Flows and Task Flow Parameters"
- Section 14.2.5, "Setting Security for the Documents Service"

#### 14.2.1 Setting Up Connections

Before you can add the Documents service and task flows to a page, you must first set up the connection to your content repository. You can reuse connections you've created or create new ones. For information on creating a connection, refer to Section 13.2, "Configuring Content Repository Connections."

Once you set up a connection to a content repository, you can set the connection as active for the Documents service by selecting **Set as primary connection for Documents service.** You can only set one active connection at a time. If you create another connection and set the new connection as active, the first connection will no longer be active. You can always return to the Properties of the connection (in the Application Resources pane, under Connections, right-click the connection name and choose **Properties**) to toggle whether the connection is active.
When you use Oracle Content Server as your content repository, the security settings depend on the connection configuration. If the connection uses External Application Service Public Credentials, there is no need to set up application security.

If the connection uses Identity Propagation, follow the steps in Section 4.1.1.1, “Implementing Security for Services.”

For information on installing the back-end content repository, see Section 4.3.1 “Oracle Content Server Requirements” in the Oracle Fusion Middleware Installation Guide for Oracle WebCenter.

14.2.2 Adding the Documents Service to the Oracle Composer Catalog

Once you have set up content repository connections for using the Documents service in your application, you may also wish to enable the display of the Documents service in the Oracle Composer Catalog for use by end users of your application (as described in the chapter "Working with the Documents Service" in the Oracle Fusion Middleware User’s Guide for Oracle WebCenter). The default Oracle Composer Catalog definition file has the entries for the Documents service commented out. To expose the Documents service and its task flows in the Oracle Composer Catalog at runtime, you must uncomment these entries. Refer to Section 12.2.1.4, "Enabling Documents in the Resource Catalog”.

14.2.3 Adding Documents Service Task Flows at Design Time

In addition to adding the Documents service to your application to provide end users with the functionality described in Section 14.1, "Introduction to the Documents Service”, you can also add any of the Documents service task flows to your application during design time to provide these task flows built into the application for end users.

For detailed information about each Documents service task flow, see “What You Should Know About the Documents Service Task Flows” in Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

This section describes how to include Documents service task flows in your application in the following subsections:

- Section 14.2.3.1, “How to Add a Documents Service Task Flow at Design Time”
- Section 14.2.3.2, “How to Modify Documents Service Task Flow Parameters”
- Section 14.2.3.3, “How to Allow End Users to Customize Documents Service Task Flows at Runtime”
- Section 14.2.3.4, “How End Users Can Modify Content Presenter Task Flow Parameters at Runtime”

14.2.3.1 How to Add a Documents Service Task Flow at Design Time

To add a Documents service task flow (Content Presenter, Document Manager, Document List View, or Recent Documents) to your custom WebCenter application:
1. Follow the steps in Section 4.1.1, "How to Prepare Your Application to Consume Services" to implement security.

2. Add the Documents service to your application (see Section 14.2.1, "Setting Up Connections").

3. Create or open a JSF page in your application where you want the task flow to be added (see Section 3.3, "Creating WebCenter Application-Enabled Pages").

4. Choose a method for adding a Documents service task flow to the page, depending on the requirements of your application:
   - Section 14.2.3.1.1, "Adding Selected Folders and Files in a Documents Service Task Flow to a Page"
   - Section 14.2.3.1.2, "Adding Documents Service Task Flows to a Page Using the Resource Palette"
   - Section 14.2.3.1.3, "Adding Documents Service Task Flows to a Page Using Oracle Composer"

**Note:** By default, the task flow displays as full screen at runtime. You can use ADF layout components to modify this layout. To learn more about ADF layout components, refer to the Oracle Fusion Middleware Web User Interface Developer’s Guide for Oracle Application Development Framework.

### 14.2.3.1.1 Adding Selected Folders and Files in a Documents Service Task Flow to a Page

To add individual files or the content of folders from a content repository connection onto your page:

1. Before you add a task flow to your page, consider whether or not you want to allow end users of your application to edit the task flow:
   - To enable runtime editing of the page, see Chapter 7, "Enabling Runtime Editing of Pages Using Oracle Composer."
   - To allow end users of your application to customize a Content Presenter or Document List View task flow using the Edit action, add the task flow inside a Show Detail Frame Component (see Section 7.1.6, "How to Enable Component Customization Using Show Detail Frame Components").

2. In the Application Resources panel, navigate to Connections, then Content Repository.

3. Locate the name of your connection, then expand it if desired to navigate to a specific folder or file.

4. Drag the folder or file to the page, and release the mouse button to display a menu of available task flows, as shown in Figure 14–1 and Figure 14–2.

**Note:** To allow end users of your application to customize a Content Presenter or Document List View task flow using the Edit action, drag the folder or file inside a Show Detail Frame Component (see Section 7.1.6, "How to Enable Component Customization Using Show Detail Frame Components").
5. Choose the Documents service task flow that you wish to use. If the folder or file you are adding to your page does not support a particular task flow, the task flow does not display in this menu. Refer to Table 14–1 for information on Documents service task flow support by content type and by repository type.

Table 14–1 Documents Service Task Flow Support by Content Types and Content Repository

<table>
<thead>
<tr>
<th>Content Type</th>
<th>Oracle Content Server</th>
<th>Oracle Portal</th>
<th>File System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content repository root folder</td>
<td>Document Manager, Document List View, Recent Documents</td>
<td>Document Manager, Document List View</td>
<td>Document Manager, Document List View</td>
</tr>
<tr>
<td>Documents of various types</td>
<td>Content Presenter</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(XML, PDF, JAVA, TXT, DOC, XLS, HTM) and images (PNG, JPG, GIF)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

6. In the Edit Task Flow Binding dialog that displays, click **OK** to accept the default task flow parameters, or modify the parameters as desired. For more information, see Section 14.2.4, "Documents Service Task Flows and 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.

7. 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.

14.2.3.1.2 Adding Documents Service Task Flows to a Page Using the Resource Palette

To add a Documents service task flow onto your page using the Resource Palette, where you specify the initial parameters for the task flow (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):

1. Before you add a task flow to your page, consider whether or not you want to allow end users of your application to edit the task flow:

   - To enable runtime editing of the page, see Chapter 7, "Enabling Runtime Editing of Pages Using Oracle Composer."

   - To allow end users of your application to customize a Content Presenter or Document List View task flow using the Edit action, add the task flow inside a Show Detail Frame Component (see Section 7.1.6, "How to Enable Component Customization Using Show Detail Frame Components").

2. In the Resource Palette, expand the WebCenter Services Catalog. Under Task Flows, select the desired Documents service task flow and drag it onto your page (Figure 14–3).

Figure 14–3 Adding Documents Service Task Flow to JSF Page
3. From the menu that displays, choose Region to open the Edit Task Flow Binding dialog (Figure 14–4).

**Figure 14–4 Specifying Parameters for Content Presenter Task Flow**

4. 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 14.2.4, "Documents Service Task Flows and 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.

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.

After you add a Documents service task flow to a page, you can subsequently modify the parameters as desired (see Section 14.2.3.2, "How to Modify Documents Service Task Flow Parameters"). End users of your application, with appropriate privileges, can also customize the task flow (see Section 14.2.3.3, "How to Allow End Users to Customize Documents Service Task Flows at Runtime") or modify the task flow parameters (see Section 14.2.3.4, "How End Users Can Modify Content Presenter Task Flow Parameters at Runtime").
14.2.3.1.3 Adding Documents Service Task Flows to a Page Using Oracle Composer

To add a Documents service task flow onto your page by editing the page using Oracle Composer:

1. Make sure that the Documents service and its task flows are exposed in the Oracle Composer Catalog at runtime, as described in Section 12.2.1.4, "Enabling Documents in the Resource Catalog".

2. Make the page editable, as described in Chapter 7, "Enabling Runtime Editing of Pages Using Oracle Composer."

3. Save the page and run it.

4. Switch to page Edit mode.

5. Click Add Content in the target region of the page to open the Oracle Composer Catalog.

6. Next to Documents, click Open, then click Add next to the Documents service task flow that you want to add to your page (Content Presenter, Document List Viewer, Document Manager, or Recent Documents).

7. To customize a Content Presenter or Document List View task flow, add the Edit action to the task flow chrome and/or Actions menu, as described in Section 14.2.3.3, "How to Allow End Users to Customize Documents Service Task Flows at Runtime", beginning with Step 3.

14.2.3.2 How to Modify Documents Service Task Flow Parameters

To modify the parameters of a Documents Service task flow after you have created the region:

1. Go to the Bindings view of your page (click the Bindings tab next to the Source tab).

2. Under Executables, you’ll see the task flow you added. Select this task flow. Figure 14–5 shows an example of a Document List View task flow in the Executables section.

Figure 14–5 Document List View Task Flow in the Bindings View

3. Next to the Executables heading, click the pencil icon to display the Edit Task Flow Binding dialog.

4. In the Edit Task Flow Binding dialog, specify parameter settings for the task flow. For more information, see Section 14.2.4, "Documents Service Task Flows and Task Flow Parameters."
Basic Configuration for the Documents Service

14.2.3.3 How to Allow End Users to Customize Documents Service Task Flows at Runtime

When you add a Content Presenter or Document List View task flow to a page in your custom WebCenter application, you can optionally allow end users who have edit privileges on the page to customize the task flow through a user-friendly interface. To provide this capability, you add an Edit action to the task flow’s chrome (shown as a wrench icon) or to the Actions menu (Figure 14–7). 

Figure 14–7 Documents Service Task Flow Actions Menu in Edit Mode

To add the Edit action for use by end users of your application:

1. If you have not already done so, enable runtime editing of the page (see Chapter 7, "Enabling Runtime Editing of Pages Using Oracle Composer").
2. Make sure that the task flow is added to the page inside a Show Detail Frame Component (see Section 7.1.6, "How to Enable Component Customization Using Show Detail Frame Components").

3. Open the application’s adf-config.xml file, located in the ADF META-INF folder under Descriptors in the Application Resources panel.

4. If it is not already declared, add the cust namespace prefix, as shown in Example 14–1:

   **Example 14–1 Adding cust Namespace Prefix in the adf-config.xml File**

   ```xml
   <adf-config xmlns="http://xmlns.oracle.com/adf/config"
                xmlns:cust="http://xmlns.oracle.com/adf/faces/customizable/config"
                xmlns:adf="http://xmlns.oracle.com/adf/config/properties">
   
   5. Define the Edit action using the <customActions> element, as shown in Example 14–2:

   **Example 14–2 Adding Edit Action in the adf-config.xml File**

   ```xml
   <cust:customizableComponentsSecurity
       xmlns="http://xmlns.oracle.com/adf/faces/customizable/config">
   <customActions>
     <customAction action="advancedEdit" location="both"
                    rendered="#{composerContext.inEditMode}"
                    icon="/adf/webcenter/configureheader_en.png"
                    text="Edit"
                    shortDesc="Edit"/>
   </customActions>
   </cust:customizableComponentsSecurity>
   
   **Note:** The location attribute specifies whether to add a wrench icon, an Edit action on the Actions menu, or both:

   - location="chrome" adds a wrench icon on the task flow chrome.
   - location="menu" adds an Edit item on the Actions menu.
   - location="both" adds both a wrench icon and an Edit item.


When you implement this functionality, the end users of your application who have appropriate edit privileges have access to user-friendly interfaces to perform task flow customization:

- For a Content Presenter task flow, clicking the Edit action opens the Content Presenter Configuration dialog. For information about using this dialog, see "Displaying Content Using the Content Presenter Task Flow" in Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

14.2.3.4 How End Users Can Modify Content Presenter Task Flow Parameters at Runtime

At runtime, the parameters of a Content Presenter task flow on an editable page can be edited in two ways:

- If you add an Edit action to the Content Presenter task flow, as described in Section 14.2.3.3, "How to Allow End Users to Customize Documents Service Task Flows at Runtime", end users can click the Edit (wrench) icon and/or select Edit from the Actions menu to display the Content Presenter Configuration dialog, where they can specify content and a display template. For end user information about the Content Presenter Configuration dialog, see "Displaying Content Using the Content Presenter Task Flow" in Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

- In Oracle Composer, an end user of your application can click the properties Edit (pencil) icon for the task flow to display the Component Properties dialog. Parameters set in this way are generally intended for use at design time by developers creating a custom application, or for advanced users who want to bind a parameter to an EL expression. If an end user modifies a property value in the Component Properties dialog, the new value overrides the value specified in the Content Presenter Configuration dialog.

14.2.4 Documents Service Task Flows and Task Flow Parameters

Each Documents service task flow has its own set of parameters. You configure these values when you add the task flow to your page.

Table 14–2 specifies the task flow name that displays in the WebCenter Services Resource Palette, the content repository with which you can use the task flow, and parameters required to configure the task flow for your application.

### Table 14–2 Documents Service Task Flows

<table>
<thead>
<tr>
<th>Task Flow</th>
<th>Content Repository</th>
<th>Parameters (* = required)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content Presenter</td>
<td>Oracle Content Server</td>
<td>datasource, datatype, maxResults, taskFlowInstId, templateCategoryId, templateViewId</td>
<td>Shows selected documents in pre-defined templates. Section 14.2.4.1, &quot;Content Presenter Task Flow Parameters and Out-of-the-Box Display Templates&quot; describes the parameters.</td>
</tr>
</tbody>
</table>
### Table 14–2 (Cont.) Documents Service Task Flows

<table>
<thead>
<tr>
<th>Task Flow</th>
<th>Content Repository</th>
<th>Parameters (‘*’ = required)</th>
<th>Description</th>
</tr>
</thead>
</table>
| Document Manager  | ■ Oracle Content Server  
                ■ Oracle Portal (read and search only)  
                ■ Microsoft SharePoint  
                ■ File system (read and write only) | connectionName  
                startFolderPath | Provides comprehensive document management functionalities, such as copying, moving, pasting, and deleting files and folders. Section 14.2.4.2, “Document Manager Task Flow Parameters” describes the parameters. |
| Document List View | ■ Oracle Content Server  
                ■ Oracle Portal  
                ■ Microsoft SharePoint  
                ■ File system (read and write only) | connectionName  
                createdAfter  
                createdBefore  
                creator  
                lastModifiedAfter  
                lastModifiedBefore  
                lastModifier  
                showFolders  
                startFolderPath  
                taskFlowInstId | Shows a simple list view of folders and documents, allows users to navigate a folder hierarchy, and enables users to customize search queries. Section 14.2.4.3, “Document List View Task Flow Parameters” describes the parameters. |
| Recent Documents  | ■ Oracle Content Server | connectionName  
                groupSpace  
                lastModifier  
                lastModifiedAfter  
                lastModifiedBefore  
                maxDocuments  
                mostRecentFirst  
                taskFlowInstId | Shows the most recently created or modified documents by the current user. Section 14.2.4.4, “Recent Documents Task Flow Parameters” describes the parameters. |

### 14.2.4.1 Content Presenter Task Flow Parameters and Out-of-the-Box Display Templates

Table 14–3 describes the Content Presenter task flow parameters. Table 14–4 and Table 14–5 describe the out-of-the-box display templates for single and multiple content items, which can be specified as values for the templateViewId and templateCategoryId parameters.

If an end user of your application has permissions to edit the page on which you have included a Content Presenter task flow, and opens the Content Presenter Configuration dialog for that task flow, the values shown in the dialog for the content and display template are read-only, reflecting the parameters you specify in the Task Flow Bindings dialog for the Content Presenter task flow at design time. The end user cannot modify these task flow parameters in the Content Presenter Configuration dialog.

However, an end user with permissions to edit the page can open the Component Properties dialog (if the page is managed by Oracle Composer) for the Content Presenter task flow and modify Content Presenter properties on the Parameters tab.
The new values specified in the Component Properties dialog override the values specified at design time.

**Table 14–3 Content Presenter Task Flow Parameters**

<table>
<thead>
<tr>
<th>Task Flow Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Default Value</th>
</tr>
</thead>
</table>
| datasource          | string | Optional. The data source of the content. The value depends on the value of Data Source Type:  
  - When Data Source Type = dsTypeSingleNode, set Data Source to a single node identifier in the format: connection_name#dDocName:content_id  
    For example: myconnection.us.oracle.com#dDocName:STANL18_IDC-007619  
  - When Data Source Type = dsTypeFolderContents, set Data Source to a single node identifier in the format: connection_name#dCollectionID:collection_id  
    For example: myconnection.us.oracle.com#dCollectionID:45535  
  - When Data Source Type = dsTypeQueryExpression, set Data Source to a search expression.  
    For example: cm_isContent = true && dDocTitle like \'Big\'  
    See Note below.  
  - When Data Source Type = dsTypeMultiNode, set Data Source to a set of comma-delimited node identifiers in the format: connection_name#dDocName:content_id, connection_name#dDocName:content_id, ...  
    For example: myconn#dDocName:DOCUMENT_ID_12345,myconn#dDocName:DOCUMENT_ID_56789
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 Oracle Content Server.
  - collection_id = the dCollectionID found in the URL for the folder information page in Oracle Content Server.

**Note**: One way to determine the properties for the existing content types defined in Oracle Content Server is to use the Content Presenter Configuration dialog, as described in "Displaying Content Using the Content Presenter Task Flow" in Oracle Fusion Middleware User’s Guide for Oracle WebCenter. When the Content Source is Results of a Query, click the Add icon in the dialog to list all the properties that are defined for the selected content type.

| datasourceType     | string | Optional. The data source type of the content. Valid values are: None  
  - dsTypeSingleNode: A single content item.  
  - dsTypeFolderContents: The contents of a folder.  
  - dsTypeQueryExpression: The results of a query.  
  - dsTypeMultiNode: An ordered list of content items. |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>maxResults</td>
<td>long</td>
<td>Optional. The maximum number of results to display when datasourceType is dsTypeQueryExpression. This value can be null.</td>
<td>null</td>
</tr>
</tbody>
</table>
Basic Configuration for the Documents Service

The out-of-the-box display templates provided by Oracle WebCenter are listed in Table 14–4 and Table 14–5.

### Table 14–4  Out-of-the-Box Templates for Displaying Single Content Items

<table>
<thead>
<tr>
<th>Single Content Item Display Templates</th>
<th>View ID in templates-registry.xml</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>
<tr>
<td>Default List Item View</td>
<td>oracle.webcenter.content .templates.default.list.item</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>oracle.webcenter.content .templates.default.details</td>
<td>Displays any single content item, either directly in the browser (images, HTML) or as a link that downloads the associated file when clicked.</td>
</tr>
</tbody>
</table>
Basic Configuration for the Documents Service

Table 14–5  Out-of-the-Box Templates for Displaying Multiple Content Items

<table>
<thead>
<tr>
<th>Multiple Content Item Display Templates</th>
<th>View ID in templates-registry.xml</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accordion View</td>
<td>oracle.webcenter.content .templates.default.list. panel.accordion</td>
<td>Displays multiple content items in an accordion format, where each item can be expanded to display its details.</td>
</tr>
<tr>
<td>Bulleted View</td>
<td>oracle.webcenter.content .templates.default.list. bulleted</td>
<td>Displays multiple content items in a bulleted list format. Only content items will be displayed; folder items will be omitted.</td>
</tr>
<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 with Content Source set to Contents Under a Folder 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</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>
14.2.4.2 Document Manager Task Flow Parameters

Table 14–6 describes the Document Manager task flow parameters.

<table>
<thead>
<tr>
<th>Task Flow Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>connectionName</td>
<td>string</td>
<td>Optional. The name of the connection to your content repository. If no value is entered, the default connection is used. For example: ${'MyContent'}</td>
<td>The connection you selected as default in the Create Content Repository Connection dialog box. To select a default, check the Set as primary connection for Documents service checkbox.</td>
</tr>
<tr>
<td>startFolderPath</td>
<td>string</td>
<td>Optional. The name of the folder to use as the root folder for the task flow. For example: ${'/Manuals'}</td>
<td>The root folder of the content repository configured with the specified connection</td>
</tr>
</tbody>
</table>

14.2.4.3 Document List View Task Flow Parameters

Table 14–7 describes the Document List View task flow parameters.

<table>
<thead>
<tr>
<th>Task Flow Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>connectionName</td>
<td>string</td>
<td>Optional. Connection Name. The name of the connection to your content repository.</td>
<td>The connection you selected as default in the Create Content Repository Connection dialog box. To select a default, check the Set as primary connection for Documents Service checkbox.</td>
</tr>
<tr>
<td>createdAfter</td>
<td>string</td>
<td>Optional. Limit the display of task flow content to files and folders created after a specified date and time. If no value is entered, then no &quot;after&quot; filtering using the creation date is applied. The value uses the ISO 8601 format: $({'yyyy-mm-ddThh:mm:ss.zzzZ'})¹ For example: $({'2009-11-14T06:02.000+01:00'})</td>
<td>None</td>
</tr>
</tbody>
</table>

¹ The value uses the ISO 8601 format.
### Table 14–7 (Cont.) Document List View Task Flow Parameters

<table>
<thead>
<tr>
<th>Task Flow Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>createdBefore</td>
<td>string</td>
<td>Optional. Limit the display of task flow content to files and folders created before a specified date and time. If no value is entered, then no “before” filtering using the creation date is applied. The value uses the ISO 8601 format: ${'yyyy-mm-ddThh:mm:ss.zz zTZ'} For example: ${'2009-11-14T06:08:02.00+01:00'}</td>
<td>None</td>
</tr>
<tr>
<td>creator</td>
<td>string</td>
<td>Optional. Limit the display of task flow content to files and folders created by a particular user. Enter the user name as specified by the user's login credentials (for example, monty). If no value is entered, then documents created by any user are shown. For example: ${'monty'}</td>
<td>None</td>
</tr>
<tr>
<td>lastModifiedAfter</td>
<td>string</td>
<td>Optional. Limit the display of task flow content to files and folders modified after a specified date and time. The value uses the ISO 8601 format: ${'yyyy-mm-ddThh:mm:ss.zz zTZ'} For example: ${'2009-11-14T18:24:36.000Z'}</td>
<td>None</td>
</tr>
<tr>
<td>lastModifiedBefore</td>
<td>string</td>
<td>Optional. Limit the display of task flow content to files and folders modified before a specified date and time. If no value is entered, then no “before” filtering using the last modification date is applied. The value uses the ISO 8601 format: ${'yyyy-mm-ddThh:mm:ss.zz zTZ'} For example: ${'2009-11-14T18:24:36.000Z'}</td>
<td>None</td>
</tr>
</tbody>
</table>
Table 14–7  (Cont.) Document List View Task Flow Parameters

<table>
<thead>
<tr>
<th>Task Flow Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>lastModifier</td>
<td>string</td>
<td>Optional. Limit the display of task flow content to files and folders last modified by the specified user(s). Enter the user name(s) as specified by the user login credentials, separated by a space. If no value is entered, then all modified documents are shown. For example: ${'monty' 'monica'}</td>
<td>None</td>
</tr>
<tr>
<td>rootFolderPath</td>
<td>string</td>
<td>Optional. The name of the folder to use as the root folder for the task flow. For example: ■ ${'/PersonalSpaces/monty/Public'} ■ ${'/WebCenterB5/Proj_X/Specs'}</td>
<td>The root folder of the content repository configured with the specified connection</td>
</tr>
<tr>
<td>showFolders</td>
<td>boolean</td>
<td>Specifies whether to display documents and folders, or documents only: ■ ${true}: folders are shown along with files. ■ ${false} (default): folders are hidden and only files appear in the task flow.</td>
<td>${false}</td>
</tr>
<tr>
<td>taskFlowInstId</td>
<td>string</td>
<td>The unique identifier of the task flow, used internally to maintain the association of the task flow instance with its customization and personalization settings. The value is generated automatically when you add individual files or the content of folders to your page from the Application Resources Panel, displaying the content in a Document List View task flow. The value is blank when you add a Document List View task flow onto your page from the Resource Palette. Enter or edit this value to set the unique identifier for the task flow.</td>
<td>None</td>
</tr>
</tbody>
</table>

1 “TZ” is the time zone indicator. If the time being described is in UTC (Coordinated Universal Time), then the time zone indicator is “Z”. If the time is from any other time zone, then TZ describes the offset from UTC of the time zone. For example, if the time is in California in December (Pacific Standard Time, PST), then the TZ indicator would be "-08:00".

14.2.4.4 Recent Documents Task Flow Parameters

Table 14–8 describes the Recent Documents task flow parameters.
### Table 14–8 Recent Documents Task Flow Parameters

<table>
<thead>
<tr>
<th>Task Flow Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>connectionName</td>
<td>string</td>
<td>Optional. The name of the connection to your content repository. If no value is entered, the default connection is used.</td>
<td>The connection you selected as default in the Create Content Repository Connection dialog box. To select a default, check the Set as primary connection for Documents service check box.</td>
</tr>
<tr>
<td>groupSpace</td>
<td>string</td>
<td>Optional. The name of a group space for which to display the documents recently modified or created.</td>
<td>None</td>
</tr>
<tr>
<td>lastModifier</td>
<td>string</td>
<td>Optional. Limit the display of task flow content to files and folders last modified by any of the specified users. Enter the user name(s) as specified by the user login credentials, separated by a space. If no value is entered, then all modified documents are shown. For example: <code>${'monty monica'}</code> To display recent documents for only the currently authenticated user: <code>${securityContext.userName}</code></td>
<td>None</td>
</tr>
<tr>
<td>lastModifiedAfter</td>
<td>string</td>
<td>Optional. Limit the display of task flow content to files and folders modified after a specified date and time. If no value is entered, then documents modified in the last three months are shown. The value uses the ISO 8601 format: <code>${'yyyy-mm-ddThh:mm:ss.zzzTZ'}</code> For example: <code>${'2009-11-14T18:24:36.000Z'}</code></td>
<td>3 months prior to the current date</td>
</tr>
</tbody>
</table>
Basic Configuration for the Documents Service

14.2.5 Setting Security for the Documents Service

The Documents service task flows support the same security options that are supported by the content repository connections. This service can also use an external application with dedicated user accounts to access remote content repositories, such as an Oracle Content Server or Oracle Portal. For more information about using security with content repositories, see Section 13.1, "Introduction to Content Integration." For information about using external applications, see Section 37.2, "Working with External Applications."

If you are using a content repository that handles its own authentication, such as Oracle Portal or Oracle Content Server, you can associate that content repository with an external application definition to allow for credential provisioning. You can modify your connection to your content repository to use an external application without shared or public credentials. When you do this, the External Application - Change Password task flow is automatically integrated to allow your end users to provide their credentials.

To register an external application for an existing content repository connection:

1. In the Application Resources pane, right-click your connection name and choose Properties.

### Table 14–8 (Cont.) Recent Documents Task Flow Parameters

<table>
<thead>
<tr>
<th>Task Flow Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>lastModifiedBefore</td>
<td>string</td>
<td>Optional. Limit the display of task flow content to files and folders modified before a specified date and time. If no value is entered, then no &quot;before&quot; filtering using the last modification date is applied. The value uses the ISO 8601 format: ${'yyyy-mm-ddThh:mm:ss. zzzZ'} For example: ${'2009-11-14T18:24:36.000Z'}</td>
<td>None</td>
</tr>
<tr>
<td>maxDocuments</td>
<td>long</td>
<td>Optional. Limit the number of files displayed to the specified maximum. If no value is entered, then all matching files are shown. For example: ${10}. Note that there is no single quote surrounding the value.</td>
<td>${null}</td>
</tr>
<tr>
<td>mostRecentFirst</td>
<td>boolean</td>
<td>Optional. The default sort order of files in the task flow: ■ $(true) (default): the most recent documents list first. ■ $(false): the oldest documents list first. Note that there is no single quote surrounding the value.</td>
<td>${true}</td>
</tr>
</tbody>
</table>
Basic Configuration for the Documents Service

2. In the Edit Content Repository Connection dialog box, under Authentication, select External Application.

*Figure 14–8 Selecting External Application in the Content Repository Connection Dialog*

![Authentication settings dialog box with options for Identity Propagation, External Application, and Specify login credentials for the current JDeveloper session.]

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, then click Next.

*Figure 14–9 Naming the External Application*

![Application Name and Application Display Name fields set to MyOCS-extApp and MyOCS External App, respectively.]

5. On the General page, under Authentication Details, choose BASIC from the Authentication Method list, then click Next.

*Figure 14–10 Choosing the BASIC Authentication Method*

![Authentication Details dialog box with BASIC selected.]

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 14–11 Specifying the Public Credentials*

![Public Credentials fields with username 'jsmith' and password '**********'.]
9. Click Finish. You’ll notice that your connection now uses the external application for authentication.

**Figure 14–12 Using the New External Application**

<table>
<thead>
<tr>
<th>Authentication:</th>
<th>Identity Propagation</th>
</tr>
</thead>
<tbody>
<tr>
<td>External Application:</td>
<td>MyOCS-extApp</td>
</tr>
</tbody>
</table>

Specify login credentials for the current JDeveloper session:

- User Name: 
- Password: 

10. In the Edit Content Repository Connection dialog box, click OK. In the Application Resources panel, you’ll notice the external application now displays.

**Figure 14–13 External Application in the Application Resources Panel**

If you do not apply security, and the content repository requires a login to access the content, the user will not be able to authenticate, and thus will not see any content at runtime.

### 14.3 Advanced Information for the Documents Service

This section describes the following advanced uses of the Documents service:

- **Section 14.3.1, "Using the Documents Service with Other WebCenter Services"**
- **Section 14.3.2, "Setting Parameters to Upload Files to Content Repositories"**
- **Section 14.3.3, "Using Adapters with the Documents Service"**

#### 14.3.1 Using the Documents Service with Other WebCenter Services

You can use the Documents service with a variety of other WebCenter services. For example, you can add tags to documents in your content repository, search across your application and retrieve documents in the results, and track recent changes to the content repository.

**Note:** You can use the Tags service with the Document Manager task flow only.
You can see an example of how to use the Documents service with the Tags service in Section 23.3.1, "Optional Way to Show Tags on Pages."

To learn more about how you can use these services together, refer to Section 1.1.4, "Introduction to Oracle WebCenter Services."

---

**Note:** When you integrate the Documents service with the Search service, the Search service returns results from all content repository connections.

---

### 14.3.2 Setting Parameters to Upload Files to Content Repositories

The Document Manager task flow allows users to upload files into content repositories. Oracle WebCenter 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 14–9) or simply update the code in the Source view. After you’ve made your changes, save the `web.xml` file, then restart Oracle JDeveloper.

#### Table 14–9  Apache MyFaces Trinidad Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UPLOAD_MAX_MEMORY</td>
<td>The maximum amount of memory in kilobytes that a single file can use when uploaded.</td>
</tr>
<tr>
<td>UPLOAD_MAX_DISK_SPACE</td>
<td>The maximum amount of disk space in kilobytes that a single file can use when uploaded.</td>
</tr>
<tr>
<td>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, located at [http://myfaces.apache.org/trinidad/devguide/fileUpload.html](http://myfaces.apache.org/trinidad/devguide/fileUpload.html).
**Note:** An attempt to upload a file displays an error in JDeveloper applications that are based on one of the following templates:

- Generic Application
- Java Desktop Application
- Java Desktop Application (ADF)
- Java EE Web Application
- Portlet Producer Application

After you add a Document Manager task flow onto a JSF page in an application based on one of these templates, then attempt to use the task flow to upload a file into a content repository, the Upload Document page displays the message **Error: A value is required.**

This issue does not occur in applications created based on the following templates:

- WebCenter Application
- Fusion Web Application

To work around this issue and successfully upload a file using a Document Manager task flow, add the attribute `usesUpload="true"` to the `af:form` component on the JSP page. For example:

```jsp
<af:form usesUpload="true">
    ...
    <af:region value="#{bindings.profile1.regionModel}"/>
    ...
</af:form>
```

### 14.3.3 Using Adapters with the Documents Service

To learn more about using existing adapters for the content repository connections, see Chapter 13, "Integrating Content."
The RSS service enables your application users to add and view RSS 2.0 formatted feeds within your application. This chapter describes how to integrate the RSS service into a custom WebCenter application at design time.

This chapter includes the following sections:

- Section 15.1, "Introduction to the RSS Service"
- Section 15.2, "Basic Configuration for the RSS Service"

For more information about managing and including RSS feeds, see:

- "Managing the RSS Service" in the Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter
- "Working with the RSS Service" in the Oracle Fusion Middleware User’s Guide for Oracle WebCenter

### 15.1 Introduction to the RSS Service

Really Simple Syndication (RSS) provides a means of accessing the content of many different web sites from a single location—a news reader. Oracle WebCenter includes the RSS service that enables you to view news feeds from different web sites from within your WebCenter applications.

This section provides an overview of the features and requirements of the RSS service. It contains the following subsections:

- Section 15.1.1, "Understanding the RSS Service"
- Section 15.1.2, "Requirements for the RSS Service"
- Section 15.1.3, "What Happens at Runtime"

#### 15.1.1 Understanding the RSS Service

To integrate the RSS service into your custom WebCenter application, you add the RSS Viewer task flow and specify the URL for the required RSS feed, as shown in Figure 15–1. Your application users can then view the RSS feed at runtime.

The RSS service also enables you to display news feeds from external sources on your application pages. 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 37.2, "Working with External Applications."
15.1.2 Requirements for the RSS Service

The RSS service does not require any back-end server. You do not need to set up a connection to use this service. However, you can set up a proxy for the RSS service, if required.

15.1.3 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 change the URL of the RSS feed.

For information about the RSS service at runtime, see the "Working with the RSS Service" chapter in the Oracle Fusion Middleware User's Guide for Oracle WebCenter.
15.2 Basic Configuration for the RSS Service

This section describes how to set up the proxy server for the RSS service and add the RSS Viewer task flow to your custom WebCenter application.

This section contains the following subsections:

- Section 15.2.1, "Setting Up a Proxy Server for the RSS Service"
- Section 15.2.2, "Adding the RSS Service at Design Time"
- Section 15.2.3, "Setting Security for the RSS Viewer Service"

15.2.1 Setting Up a Proxy Server for the RSS Service

The RSS service does not require any connections. You can simply point to the URL of the RSS feed. However, if you want to point to an RSS feed that is external to your intranet and application, you may need to set up a proxy server for your application.

To set up a proxy server for the RSS service:

1. In Oracle JDeveloper, while your application is open and selected in the Application Navigator, choose the Tools menu option, then select 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 15–3).
4. Click OK.

Note: You can see the RSS service in action in the sample application Fusion Order Demo for WebCenter, as described in Chapter 2, "Introduction to the WebCenter Sample Application."
15.2.2 Adding the RSS Service at Design Time

This section describes the RSS Viewer task flow and how to add it to your application.

This section contains the following subsections:

■ Section 15.2.2.1, “About RSS Viewer Task Flow”
■ Section 15.2.2.2, “How to Add the RSS Viewer Task Flow to Your Application”

15.2.2.1 About RSS Viewer Task Flow

The RSS service has a single task flow called RSS Viewer, which you can add to your application to enable your users to access an RSS feed. You can add multiple instances of the task flow to your application and use the Edit Task Flow Binding dialog to point to multiple RSS feed locations.

15.2.2.2 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 4.1.1, "How to Prepare Your Application to Consume Services" to implement security, if necessary, and create a customizable page in your application.

2. Open the customizable page.

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 37.2, "Working with External Applications."
4. In the Resource Palette, open the **WebCenter Services Catalog**, then expand the **Task Flows** folder.

5. Click **RSS Viewer**, drag it to your page in the Design view, and choose **Region**.

6. In the Edit Task Flow Binding dialog, you can set the location of the RSS feed and enter the name (the application name, not the application display name) of the external application that the custom WebCenter application can use to authenticate RSS feed.

![Figure 15–4  Example of the Edit Task Flow Binding Dialog for the RSS Service](image)

**Table 15–1** describes the possible values for the RSS Viewer task flow binding parameters.

**Table 15–1  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 custom WebCenter application with the RSS feed. If the RSS feed does not require authentication, you do not need to set up and identify an external application for this service.</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 (Figure 15–5).

![Figure 15–5  RSS Viewer Task Flow in the Page Source](image)

15.2.3 Setting Security for the RSS Viewer Service

To use the RSS service with a public RSS feed, you do not need to set security. To use the RSS service with an RSS feed that requires authentication, you can set up an
external application for your custom WebCenter 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 on using external applications, see Section 37.2, "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 application, the View grant is automatically added to the authenticated-role.
Part IV contains the following appendixes:

- Chapter 16, "Integrating the Announcements Service"
- Chapter 17, "Integrating the Discussions Service"
- Chapter 18, "Integrating the Instant Messaging and Presence Service"
- Chapter 19, "Integrating the Mail Service"
- Chapter 20, "Integrating the People Connections Service"
- Chapter 21, "Integrating Oracle WebCenter Wiki and Blog Server"
Integrating the Announcements Service

This chapter explains how to integrate the Announcements service into a custom WebCenter application at design time. For more information about managing and including announcements, see:

- "Managing the Announcements and Discussions Services" in the Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter
- "Working with the Announcements Service" in the Oracle Fusion Middleware User’s Guide for Oracle WebCenter

This chapter includes the following sections:

- Section 16.1, "Introduction to Announcements"
- Section 16.2, "Basic Configuration for the Announcements Service"
- Section 16.3, "Advanced Information for the Announcements Service"

16.1 Introduction to Announcements

The Announcements service lets you create and expose announcements on your application pages. Access to announcements boosts community participation, problem resolution, and knowledge sharing.

You can use the Links service to link announcements to other services, such as Events or Discussions. For example, suppose your company is announcing a new product, you can link from the announcement directly to a discussion forum, where potential customers can ask other customers about the product, or link to an instant messenger to speak directly with a customer service representative to purchase the product.

Announcements can show either forum-level announcements or global announcements (that is, system announcements not specific to a forum).

This section contains the following subsections:

- Section 16.1.1, "Understanding the Announcements Service"
- Section 16.1.2, "Requirements for Announcements"
- Section 16.1.3, "What Happens at Runtime"

16.1.1 Understanding the Announcements Service

With the Announcements service, you can do the following:

- Create an announcement by clicking the Create icon, and then send it to a distribution list.
Edit an existing announcement by clicking the **Edit** icon next to the announcement.

Delete an existing announcement by clicking the **Delete** icon next to the announcement.

See the latest announcement data by clicking the **Refresh** icon on the Announcement view.

Personalize the Announcements view. You can select to display announcements sent today, this week, this month, or all announcements. A **Show** list provides options for displaying announcements created within the selected time range.

Manage announcements, such as publishing announcements on a later date and specifying auto-expire on certain dates.

The Announcements service is integrated with many WebCenter services, such as RSS, Search (to search announcement text), Instant Messaging and Presence (IMP), and Recent Activities. Use the Links service to link announcements to other services, such as Discussions.

*Figure 16–1* shows a sample announcement.

**Figure 16–1  Sample Announcement From the Announcement Task Flow**

---

### 16.1.2 Requirements for Announcements

The Announcements service requires a discussions server. You must install and configure the discussions server that comes with Oracle Fusion Middleware.

For information on installing the discussions server, see *Oracle Fusion Middleware Installation Guide for Oracle WebCenter*.

### 16.1.3 What Happens at Runtime

At runtime, user who have been granted the permission to do so can post announcements to all authenticated users. For example, an application administrator can use this service to announce the availability of a new feature or the need to shut down the application temporarily for maintenance (*Figure 16–2*).
Application administrators in custom WebCenter applications can access the Announcement Manager, which provides **Create**, **RSS**, and **Refresh** icons, and a **Show** dropdown list to control the display of announcements on the page. **Edit**, **Delete**, **Mail**, and **Links** icons are available for each announcement in the list.

Depending on the privileges users have on the page and whether the required WebCenter services are configured in the application, users may see only a subset of these options in the Announcement Manager. For example, the **Mail** icon displays only if the Mail service is configured in your application, and the **Delete** icon is displayed only to users with **grant** privilege. By default, this privilege is available to application administrators.

To view announcements in the Announcement Manager, ensure that the Announcements task flow has been added to your application. Then log in to your custom WebCenter application and click the **Open Announcement Manager** icon in the Announcements task flow (Figure 16–3).

**Notes:** The **Open Announcement Manager** icon does not display in the Announcements task flow if you do not have the required privileges.

When the Announcement Manager is opened from the sidebar view, it shows global announcements, not announcements for a particular forum in the discussions server.
16.2 Basic Configuration for the Announcements Service

This section describes required steps for adding this service to your application. It contains the following subsections:

- Section 16.2.1, "Setting Up a Connection for Announcements"
- Section 16.2.2, "Adding the Announcements Service at Design Time"
- Section 16.2.3, "Setting Security for Announcements"

16.2.1 Setting Up a Connection for Announcements

To take advantage of the Announcements service, you must first create a connection to the discussions server from your custom WebCenter application. To do so, ensure that you have the connection information for the discussions server.

16.2.1.1 Announcements Connections

The Announcements service requires a WebCenter Discussion connection to the discussions server. You can register additional WebCenter Discussion connections, but only one connection is active at a time.

When you create a WebCenter Discussion connection or set a connection as active, both the Announcements and Discussions services use this same connection. If you have an existing connection, then you can skip this section and see Section 16.2.2, "Adding the Announcements Service at Design Time."

If you do not have an existing connection, then you must create a new WebCenter Discussion connection.
16.2.1.2 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 WebCenter Discussion connection for the Discussions service, then that is used by default for the Announcements service. No extra configuration is required.

2. In the Resource Palette, click the New icon and select New Connection, WebCenter Discussion Connection.

3. On the Name page, select to create the connection in Application Resources. (A connection in Application Resources is available only for that application, while a connection in IDE Connections is available for all applications you create. If you plan to use the connection in other applications, then select IDE Connections so you do not need to re-create it.)

   **Note:** If you create a connection in IDE and not in the application, then the connection must be added to the application. For example, in the Resource Palette under IDE Connections, right-click the connection and select Add to Application.

4. For Connection Name, enter a meaningful name; for example, MyDiscussions.

5. Select the Set as default connection checkbox. You can have multiple connections, but only one can be active (default). If you have different discussions servers (for example, for each page in a space), then do not select the checkbox, but the service requires that one connection be marked as the active connection (Figure 16–5).

   **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:
   - forum.url: The URL for your discussions server instance. For example: http://discussions.example.com:8888/owc_discussions.
   - admin.user: The user name of your WebCenter Discussions Server administrator; for example, admin. This account is used by the Discussions and Announcement services to perform administrative operations on behalf of WebCenter users.
   - forum.connection.secure: Indicates that a secure communication is required between the custom WebCenter application and the discussions server. You must set this value to true.

   The default value is false. When set to true (secured mode), all Web Services calls from the custom WebCenter application are sent with a user name token and client certificate, and additional configuration about Web Services Security (WS-Security) is required on the next page. For more information, see the chapter "Configuring WS-Security for WebCenter Applications and Components" in the Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter and see the Oracle Fusion Middleware Security and Administrator’s Guide for Web Services.

8. Optionally, enter a value for the other parameter:
   - connection.time.out: The time (in seconds) the service should wait for the server to respond while making the connection. If the discussions server does not respond in the given time, then it cancels the connection and reports an error.
9. Click **Test Connection**, and if it is successful, then click **Next**.

10. On the Create Discussion Connection - Step 3 of 3 page, you must configure client-side WS-Security (Figure 16–7).

First obtain the client-side certificate and store it in a local directory. For more information, see the chapter "Configuring WS-Security for WebCenter Applications and Components" in the Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter.

Click the **Add Property** or **Add Secured Property** button to add each of the following WS-Security parameters:

- **keystore.location**: The certificate file (`webcenter.jks`) path in your local directory.
- **keystore.password**: The keystore password. Add this as a secured property.
- **keystore.type**: The keystore type associated with the certificate, for example, JKS.
- **encryption.key.alias**: The key alias to be used for encryption.
- **encryption.key.password**: The password for accessing the encryption key. Add this as a secured property.

**Note**: To encrypt property values, such as passwords, click the **Add Secured Property** button.

Ensure that the property values match those in the certificate.
11. Click Finish. Your connection should now appear as a node under Application Resources - Connections.

16.2.2 Adding the Announcements Service at Design Time

This section explains a basic incorporation of the Announcement service.

16.2.2.1 Announcements Task Flows

The announcement task flows let you add a main view or a sidebar view of announcements to your page.

<table>
<thead>
<tr>
<th>Task Flow</th>
<th>Required Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Announcements</td>
<td>parentId</td>
<td>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. If this parameter is not specified, then global announcements are shown.</td>
</tr>
</tbody>
</table>
16.2.2 How to Add Announcements to Your Application

The Announcements task flow provides a complete view of your announcements. To add the Announcements task flow to your custom WebCenter application:

1. Follow the steps described in Chapter 3, "Preparing Your Development Environment" to implement security and create a new customizable page in your application.

2. Open the page on which you want to add the service.


4. Drag Announcements from the Resource Palette and drop it onto the page inside of the \&lt;af:form \&lt;begin and \&lt;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, you can enter a value for the parentId parameter.
   - If global announcements are required, then do not specify a value for parentId.
   - If announcements from a specific forum are required (for example, when a forum is reserved for each group), then enter a value for parentId; for example, \${2}.

7. Click OK.

8. Save and run your page.

Figure 16–8 shows the Announcements view at runtime. The runtime view is based on the value that was provided for the parentId parameter.

<table>
<thead>
<tr>
<th>Task Flow</th>
<th>Required Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Announcements - Sidebar View</td>
<td>parentId</td>
<td>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. If this parameter is not specified, then global announcements in the discussions server are shown. 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 16.3.1, &quot;How to Add the Announcements - Sidebar View Task Flow.&quot;</td>
</tr>
</tbody>
</table>
16.2.3 Setting Security for Announcements

In a non-secured custom WebCenter application, identity propagation cannot happen. A user is a guest (or anonymous) user who can view only public categories and forums. In a non-secured 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, then WebCenter cannot fetch global announcements (that is, announcements not scoped to a forum): global announcements are not available for public users.

In an ADF-secured custom WebCenter 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 custom WebCenter application point to the same identity store. To simplify the login process for your custom WebCenter application, you should use the same login credentials for the application as you do for the discussions server. This way, your users can log in to the application one time and automatically connect to the discussions server.

**Note:** The Announcements services requires that the identity store be LDAP-based; that is, not file-based with jazn-data.xml.

For information about configuring ADF security, see Section 3.5, "Implementing Security in Your Application."

16.3 Advanced Information for the Announcements Service

This section describes optional features available with this service. It contains the following subsections:

- Section 16.3.1, "How to Add the Announcements - Sidebar View Task Flow"
- Section 16.3.2, "Customizing Announcements Views"
- Section 16.3.3, "Obtaining Group Space RSS News Feed URL for the Announcements Service"
16.3.1 How to Add the Announcements - Sidebar View Task Flow

The Announcements - Sidebar View is similar to the Announcements view in that it displays announcements and provides tools to create, edit, and delete announcements. It also provides controls for determining when an announcement is published and when it expires. After you create an announcement, you can select the Mail icon next to the announcement to mail the announcement to anyone you choose.

The Announcements - Sidebar View task flow lets you present announcements to end users where manage controls are not needed. Based on the content, it can render in a default view or a custom view.

To add the Announcements - Sidebar View to your custom WebCenter application, follow the same instructions that you did for the Announcements task flow in Section 16.2.2, "Adding the Announcements Service at Design Time," but drag and drop Announcements - Sidebar View onto the page.

![Figure 16–9  Edit Task Flow Binding Dialog for Announcements - Sidebar View](image)

Table 16–2 describes the Announcements - Sidebar View parameters. The look and feel of the Announcements - Sidebar View changes with the values provided for these parameters. For more information, see Section 16.3.2, "Customizing Announcements Views."

<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. For example, ${2}. If this parameter is not specified, then announcements default to global announcements.</td>
</tr>
</tbody>
</table>
Advanced Information for the Announcements Service

Table 16–2  (Cont.) Announcements - Sidebar View Task Flow Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>freeFlowView</td>
<td>A Boolean value representing whether to remove the announcement title (subject) and show the announcement body as is.</td>
</tr>
<tr>
<td></td>
<td>The default value is ${false}, which means that the announcement list displays the title and body in plain text and the truncateAt and expandedAnnouncements parameters control it.</td>
</tr>
<tr>
<td></td>
<td>Enter ${true} to remove the announcement title and ignore the values for truncateAt and expandedAnnouncements.</td>
</tr>
<tr>
<td>expandedAnnouncements</td>
<td>The number of announcements to display announcement details in plain text (that is, the body of the announcement). Users can click the title to display the full announcement content in rich text mode.</td>
</tr>
<tr>
<td></td>
<td>The value you enter for expandedAnnouncements is ignored if freeFlowView is set to true.</td>
</tr>
<tr>
<td>truncateAt</td>
<td>For announcements that display announcement body in plain text, this value specifies how many characters to display for each announcement.</td>
</tr>
<tr>
<td></td>
<td>Enter an Expression Language (EL) expression. For example, when the value is set to the EL expression ${50}, following their titles, announcements display no more than 50 characters. Users can click announcement titles to display the full announcement.</td>
</tr>
<tr>
<td></td>
<td>If no value is specified, then it displays 200 characters. If a non-valid positive integer value is specified, then it displays all characters in plain text.</td>
</tr>
<tr>
<td></td>
<td>This parameter takes effect with expandedAnnouncements. The value you enter for truncateAt is ignored if freeFlowView is set to true.</td>
</tr>
</tbody>
</table>

16.3.2 Customizing Announcements Views

You can change the look and feel of the Announcements - Sidebar View with the parameter values. This section shows two examples of this.

Figure 16–10 shows the Announcements - Sidebar View at runtime with the freeFlowView parameter set to ${true}.

Figure 16–10  Announcements - Sidebar View with freeFlowView Parameter

Figure 16–11 shows the Announcements - Sidebar View at runtime with the freeFlowView parameter set to false (or empty), the expandedAnnouncements parameter set to 2, and the truncateAt parameter set to 20.
16.3.3 Obtaining Group Space RSS News Feed URL for the Announcements Service

You can expose Oracle WebCenter Spaces functionality in a custom WebCenter application. Your custom application users can find out what is happening in a specific group space through RSS news feeds.

Configure RSS news feeds for the Announcements service to enable users to view group space announcements from within a custom application. To obtain the group space RSS news feed URL for the Announcements service, use either of the following WebCenter Spaces APIs:

- `getServiceRSSFeedURL`
- `getServiceRSSFeedURLbyGuid`

To obtain an RSS feed URL, you must identify the group space (by name or GUID) and specify the service required (by service ID). The service ID for the Announcements service is `GroupSpaceWSClient.ANNOUNCEMENT_SERVICE_ID`.

For information about how to use these APIs, see Section 27.2.5.3.8, "Retrieving RSS Feed URLs for Group Space Services."
This chapter explains how to integrate the Discussions service in a custom WebCenter application at design time. For more information about managing and including discussions, see:

- "Managing the Announcements and Discussions Services" in the Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter
- "Working with the Discussions Service" in the Oracle Fusion Middleware User's Guide for Oracle WebCenter

This chapter includes the following sections:

- Section 17.1, "Introduction to Discussions"
- Section 17.2, "Basic Configuration for the Discussions Service"
- Section 17.3, "Advanced Information for the Discussions Service"

17.1 Introduction to Discussions

The Discussions service lets you expose discussion forums on your application pages, so users can create forums, post questions, and search for answers. For example, customers can share product reviews, or a customer service department can answer questions online. Discussion forums additionally provide the means to preserve and revisit discussions.

The back-end server that provides the Discussion service manages content in a hierarchy. At the top of the hierarchy are categories, below that are forums, and then topics. Where categories are exposed in your WebCenter 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.

17.1.1 Understanding the Discussions Service

With the Discussions service, you can do the following (according to your permissions):

- Create a new discussion forum or a new topic.
- Navigate into a forum from a list of available forums.
- Edit an existing discussion forum, topic, or message by clicking the Edit icon.
- Delete a forum, topic, or message by clicking the Delete icon.
- Reply to an existing topic or message by clicking the Reply icon.
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 by clicking the **Watch** icon.

View the following in the sidebar:
- Watched topics
- Watched forums
- Most popular (frequently viewed) topics
- Most recent topics

---

**Note:** In a secured custom WebCenter 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 17.2.3, "Setting Security for Discussions."

---

The Discussions service is integrated with many WebCenter services, such as Instant Messaging and Presence, RSS, and Search (to search within forums) services. Use the Mail service to archive mails into discussions as threads. Use the Links service to link to a discussion from any WebCenter object, or, link to a WebCenter object. Use the Documents service to attach files to discussion threads, and use the Tags service to tag discussion threads help users to search for specific topics.

17.1.2 Requirements for Discussions

The Discussions service requires a discussions server. Install and configure the discussions server that comes with Oracle Fusion Middleware.

For information on installing the discussions server that comes with Oracle Fusion Middleware, see *Oracle Fusion Middleware Installation Guide for Oracle WebCenter*.

17.1.3 What Happens at Runtime

To create forums in custom WebCenter applications, you must be a moderator or administrator on the back-end discussion server. For more information, see the "Granting Administrator Role for Oracle WebCenter Discussions Server" in the *Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter*.

Figure 17–1 shows a discussion forum at runtime.
17.2 Basic Configuration for the Discussions Service

This section describes required steps for adding the Discussions service to your application. It contains the following subsections:

- **Section 17.2.1, "Setting up Connections for Discussions"
- **Section 17.2.2, "Adding the Discussions Service at Design Time"
- **Section 17.2.3, "Setting Security for Discussions"

17.2.1 Setting up Connections for Discussions

You must create a connection to the discussions server in your custom WebCenter application.

The Discussions service requires a WebCenter Discussion connection to the discussions server. You can register additional WebCenter Discussion connections, but only one connection is active at a time.

When you create a WebCenter Discussion connection or set a connection as active, both the Announcements and Discussions services use this same connection. If you have an existing connection, then you can skip this section and see **Section 17.2.2, "Adding the Discussions Service at Design Time."

If you do not have an existing connection, then you must create a new WebCenter Discussion connection.

17.2.1.1 Discussions Connections

The Discussions service requires a WebCenter Discussion connection to the discussions server.
17.2.1.2 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 WebCenter Discussion connection for the Announcements service, then that is used by default for the Discussions service. No extra configuration is required.

   However, to use different WebCenter Discussion connection for the Announcements service and the Discussions service, then a manual entry in adf-config.xml is required.

2. In the Resource Palette, click the **New** icon and select **New Connection**, **WebCenter Discussion Connection**.

3. On the **Name** page, select to create the connection in **Application Resources**. (A connection in Application Resources is available only for that application, while a connection in IDE Connections is available for all applications you create. If you plan to use the connection in other applications, then select IDE Connections so you do not need to re-create it.)

4. For **Connection Name**, enter a unique name; for example, **MyDiscussions**.

5. Select the **Set the default connection** checkbox. You can have multiple connections, but only one can be active (default). If you have different discussions servers (for example, for each page in a space), then do not select the checkbox, but the service requires that one connection be marked as the default connection (Figure 17–2).

   **Note:** After you create a connection as the default connection, you cannot edit it so that it is not the default. To use a different default connection, you must create a new connection and mark that as the default connection.
6. Click Next.

7. On the General page, enter values for the required parameters:
   - **forum.url**: The URL for your discussions server instance. For example: http://discuss-example.com:8888/owc_discussions.
   - **admin.user**: The user name of your WebCenter Discussions Server administrator; for example, admin. This account is used by the Discussions and Announcement services to perform administrative operations on behalf of WebCenter users.
   - **forum.connection.secure**: Indicates that a secure communication is required between the custom WebCenter application and the discussions server. You must set this value to true.
     The default value is false. When set to true (secured mode), all Web Services calls from the custom WebCenter application are sent with a user name token and client certificate, and additional configuration about Web Services Security (WS-Security) is required on the next page. For more information, see the chapter "Configuring WS-Security for WebCenter Applications and Components" in the Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter and see the Oracle Fusion Middleware Security and Administrator’s Guide for Web Services.

8. Optionally, enter a value for the other parameter:
   - **connection.time.out**: The time (in seconds) the service should wait for the server to respond while making the connection. If the discussions server does not respond in the given time, then it cancels the connection and reports an error.
9. Click **Test Connection**, and if it is successful, then click **Next**.

10. On the **Create Discussion Connection - Step 3 of 3** page, you must configure client-side WS-Security (Figure 17–4).

    First obtain the client-side certificate and store it in a local directory. For more information, see the chapter "Configuring WS-Security for WebCenter Applications and Components" in the *Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter*.

    Click the **Add Property** or **Add Secured Property** button to add each of the following WS-Security parameters:

    - **keystore.location**: The certificate file (*webcenter.jks*) path in your local directory

    **Note**: If the application is deployed to WebLogic Server, then this path should be accessible on the deployed server. Make sure that the path is available to the application on the deployed server.

    - **keystore.password**: The keystore password. Add this as a secured property.
    - **keystore.type**: The keystore type associated with the certificate. Valid values are: `jks` (Java Key Store) and `pks`.
    - **encryption.key.alias**: The key alias to be used for encryption.
    - **encryption.key.password**: The password for accessing the encryption key. Add this as a secured property.
Ensure that the property values match those in the certificate.

**Figure 17–4 Create Discussions Connection, Step 3**

11. Click Finish. Your connection should now appear as a node under Application Resources - Connections.

### 17.2.2 Adding the Discussions Service at Design Time

This section explains a basic incorporation of the Discussions service.

#### 17.2.2.1 Discussions Task Flows

The Discussions service provides several task flows (Table 17–1) to enable you to include the service in a form that best suits your needs.

<table>
<thead>
<tr>
<th>Task Flow</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discussion Forums</td>
<td>This task flow displays the Oracle WebCenter Discussions Manager View, which allows the user to see all the discussions and their respective replies.</td>
</tr>
<tr>
<td></td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td>All watched forums and topics are accessible from this task flow. The Watched Forums and Watched Topics task flows provide more focussed views of watched forums or watched topics.</td>
</tr>
<tr>
<td></td>
<td>The parameters alter the way the view appears. For more information, see Section 17.3.1, &quot;Customizing Discussions Views.&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.</td>
</tr>
<tr>
<td></td>
<td>For more information, see Section 17.3.2, &quot;Adding the Discussions - Popular Topics Task Flow.&quot;</td>
</tr>
</tbody>
</table>
### 17.2.2.2 How to Add Discussions to your Application

The **Discussion Forums** task flow provides a complete view of your discussions. To add the Discussion Forums task flow to your WebCenter application, follow these steps.

1. Follow the steps described in Section 4.1.1, "How to Prepare Your Application to Consume Services" to implement security and create a new customizable page in your application.

2. Open the page on which you want to add the Discussions service.


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. Table 17–2 describes the parameters.

<table>
<thead>
<tr>
<th>Task Flow</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<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 17.3.3, &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 17.3.4, &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 17.3.5, &quot;Adding the Discussions - Watched Topics Task Flow.&quot;</td>
</tr>
<tr>
<td>Discussions - Sidebar 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 17.3.6, &quot;Adding the Discussions - Sidebar View Task Flow.&quot;</td>
</tr>
</tbody>
</table>

**Note:** The parameters alter the way the view appears. For more information, see Section 17.3.1, "Customizing Discussions Views."
Table 17–2 Parameters for Discussion Forums Task Flow

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>categoryId</td>
<td>This optional parameter is an identifier for an existing category in your Oracle WebCenter Discussions instance 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 Oracle WebCenter Discussions administrator interface and then reference that category identifier here.</td>
</tr>
<tr>
<td>forumId</td>
<td>This parameter is an identifier for an existing forum that resides inside the given categoryId. It is optional. You should be certain about the specific forum for which you want to see data before entering this value.</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 Category ID; false, the default value, means you want the task flow to display the topics associated with the specified Forum ID. This parameter value works in combination with other parameters.</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>
</tr>
</tbody>
</table>

8. Click OK.

**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 and run your page.

**Note:** All instances of the Discussion Forums task flow in an application run against the same discussions server: it is unnecessary to add multiple Discussion Forums task flow instances. This is true for all service task flows that require connections to back-end servers, such as task flows from the Announcements or Mail services.

The Discussion Forums main view contains some features that require other services in your application.

- For the links in the main view to work, you must have configured the Links service. For more information, see Chapter 22, "Integrating the Links Service."
For users’ presence indicators to work, you must have configured the Oracle WebCenter Presence service. For more information, see Chapter 18, "Integrating the Instant Messaging and Presence Service."

Oracle recommends that the discussions server and WebCenter point to the same identity store. If the custom WebCenter application and the discussions page are configured to be secure, then upon logging in to the application, then the Discussions service respects those credentials in the discussions server.

17.2.3 Setting Security for Discussions

By default, authenticated users of custom WebCenter applications can view and participate in discussion forums. Any user who has logged in to a custom WebCenter application can view discussions; and users with sufficient permissions can create forums and topics. To create forums in custom WebCenter applications, you must be a moderator or administrator on the back-end discussion server that provides the WebCenter Discussions service.

In an unsecured custom WebCenter 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 custom WebCenter 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 a non-secured custom WebCenter 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 custom WebCenter 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 custom WebCenter application point to the same identity store. When you run the page, you are presented with a login page for user credentials. Enter the credentials with the required privileges in the discussions server.

Note: The Discussions services requires that the identity store be LDAP-based; that is, not file-based with jazn-data.xml.

For information about configuring ADF security, see Section 3.5, "Implementing Security in Your Application."

17.3 Advanced Information for the Discussions Service

This section describes optional features available with this service. It contains the following subsections:
17.3.1 Customizing Discussions Views

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 custom WebCenter application.
- **forumId:** This is similar to forumId and isCategoryView is set to false.
Figure 17–5 shows the Discussion Forums view with the value of isCategoryView left blank. All forums under the root category are listed.

Figure 17–5 Discussions Forum View with categoryId Set

With the categoryId parameter set, you can click any forum or any topic, as shown in Figure 17–6 and Figure 17–7.

Figure 17–6 Click any Forum in the Discussion Forums View

Figure 17–7 Click any Topic in the Discussion Forums View

Figure 17–8 shows the Discussion Forums view with the forumId parameter set.

Figure 17–8 Discussion Forums View with the forumId Parameter Set

With the forumId parameter given, you can click any topic, as shown in Figure 17–9.
17.3.2 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 custom WebCenter application, follow the same instructions that you did for the **Discussion Forums** task flow in Section 17.2.2.2, “How to Add Discussions to your Application,” but drag and drop **Discussions - Popular Topics** onto the page.

Table 17–3 describes the available parameters for this task flow.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>categoryId</td>
<td>This optional parameter is an identifier for an existing category in your Oracle WebCenter Discussions instance to which the view should be scoped.</td>
</tr>
<tr>
<td></td>
<td>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.)</td>
</tr>
<tr>
<td></td>
<td>For testing purposes, you may want to create a category through the Oracle WebCenter Discussions administrator interface and then reference that category identifier here.</td>
</tr>
<tr>
<td>forumId</td>
<td>This parameter is an identifier for an existing forum in your Oracle WebCenter Discussions instance for which popular topics should be fetched.</td>
</tr>
<tr>
<td></td>
<td>If both <code>categoryId</code> and <code>forumId</code> are given, then only <code>categoryId</code> is honored.</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>
</tr>
</tbody>
</table>

Figure 17–10 shows how the **Discussions - Popular Topics** task flow looks at runtime.
17.3.3 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 custom WebCenter application, follow the same instructions that you did for the Discussion Forums task flow in Section 17.2.2.2, "How to Add Discussions to your Application," but drag and drop Discussions - Recent Topics onto the page.

Table 17–4 describes the available parameters for this task flow.

Table 17–4 Parameters for Discussions - Recent Topics Task Flow

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>categoryId</td>
<td>This optional parameter is an identifier for an existing category in your Oracle WebCenter Discussions instance to which the view should be scoped.</td>
</tr>
<tr>
<td></td>
<td>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.)</td>
</tr>
<tr>
<td></td>
<td>For testing purposes, you may want to create a category through the Oracle WebCenter Discussions administrator interface and then reference that category identifier here.</td>
</tr>
<tr>
<td>forumId</td>
<td>This parameter is an identifier for an existing forum in your Oracle WebCenter Discussions instance for which popular topics should be fetched.</td>
</tr>
<tr>
<td></td>
<td>If both categoryId and forumId are given, then only categoryId is honored.</td>
</tr>
<tr>
<td>disableToolbar</td>
<td>This parameter indicates whether the toolbar, including the Refresh icon, should be displayed. It can be $true$, $false$, or undefined. If you leave this parameter undefined, then the default behavior is to show the toolbar.</td>
</tr>
</tbody>
</table>

Figure 17–11 shows how the Discussions - Recent Topics task flow looks at runtime. To view a topic post, simply click the relevant topic.

17.3.4 Adding the Discussions - Watched Forums Task Flow

The Discussions - Watched Forums task flow displays a view that allows users to see all of their watched forums under a given category ID.

You can watch discussion forums and topics to keep a close eye on the information most current and relevant to your efforts. The forums and topics you select to watch...
are personal, in that your selections appear on your view of watch lists. No other user is affected by the forums and topics you choose to watch.

When a user places a watch on a forum or a topic, whenever users add to that forum or topic, in addition to it appearing in the user’s watched forums or topics list, the user receives an mail notification.

All watched forums and topics are accessible from the Forums task flow. The Watched Forums and Watched Topics task flows provide more focussed views of watched forums or watched topics.

To add the Discussions - Watched Forums task flow to your custom WebCenter application, follow the same instructions that you did for the Discussion Forums task flow in Section 17.2.2.2, “How to Add Discussions to your Application,” but drag and drop Discussions - Watched Forums onto the page.

Table 17–5 describes the available parameters for this task flow.

Table 17–5 Parameters for Discussions - Watched Forums Task Flow

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>categoryId</td>
<td>This optional parameter is an identifier for an existing category in your Oracle WebCenter Discussions instance 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 Oracle WebCenter Discussions administrator interface and then reference that category identifier here.</td>
</tr>
<tr>
<td>disableToolbar</td>
<td>This parameter indicates whether the toolbar, including the Refresh icon, should be displayed. It can be ${true}, ${false}, or undefined. If you leave this parameter undefined, then the default behavior is to show the toolbar.</td>
</tr>
</tbody>
</table>

Figure 17–12 shows how the Discussions - Watched Forums task flow looks at runtime.

Figure 17–12 Discussions - Watched Forums View at Runtime

For information on how to watch a forum or topic at runtime, see "Working with the Discussions Service" in the Oracle Fusion Middleware User’s Guide for Oracle WebCenter.
17.3.5 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 custom WebCenter application, follow the same instructions that you did for the **Discussion Forums** task flow in Section 17.2.2.2, "How to Add Discussions to your Application," but drag and drop **Discussions - Watched Topics** onto the page.

Table 17–6 describes the available parameters for this task flow.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>categoryId</td>
<td>This optional parameter is an identifier for an existing category in your Oracle WebCenter Discussions instance to which the view should be scoped.</td>
</tr>
<tr>
<td></td>
<td>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.)</td>
</tr>
<tr>
<td></td>
<td>For testing purposes, you may want to create a category through the Oracle WebCenter Discussions administrator interface and then reference that category identifier here.</td>
</tr>
<tr>
<td>forumId</td>
<td>This parameter is an identifier for an existing forum in your Oracle WebCenter Discussions instance for which watched topics should be fetched.</td>
</tr>
<tr>
<td></td>
<td>If both categoryId and forumId are given, then only categoryId is honored.</td>
</tr>
<tr>
<td>disableToolbar</td>
<td>This parameter indicates whether the toolbar, including the Refresh icon, should be displayed. It can be ${true}, ${false}, or undefined. If you leave this parameter undefined, then the default behavior is to show the toolbar.</td>
</tr>
</tbody>
</table>

Figure 17–13 shows how the **Discussions - Watched Topics** task flow looks at runtime.

**Figure 17–13  Discussions - Watched Topics View at Runtime**

- Watched Topics
- Is there a load limit?
- What are the rules of the r...
- How do I create a forum topic?

For information on how to watch a forum or topic at runtime, see "Working with the Discussions Service" in the Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

17.3.6 Adding the Discussions - Sidebar View Task Flow

The **Discussions - Sidebar 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 - Sidebar View** task flow to your custom WebCenter application, follow the same instructions that you did for the **Discussion Forums** task flow in Section 17.2.2.2, “How to Add Discussions to your Application,” but drag and drop **Discussions - Sidebar View** onto the page.

**Figure 17–14** shows how the **Discussions - Sidebar View** task flow looks at runtime.

**Figure 17–14  Discussions - Sidebar View at Runtime**

---

### 17.3.7 Obtaining Group Space RSS News Feed URL for the Discussions Service

You can expose Oracle WebCenter Spaces functionality in a custom WebCenter application. Your custom application users can find out what is happening in a specific group space through RSS news feeds.

Configure RSS news feeds for the Discussions service to enable users to view group space discussions from within a custom application. To obtain the group space RSS news feed URL for the Discussions service, use either of the following WebCenter Spaces APIs:

- `getServiceRSSFeedURL`
- `getServiceRSSFeedURLbyGuid`

To obtain an RSS feed URL, you must identify the group space (by name or GUID) and specify the service required (by service ID). The service ID for the Discussions service is `GroupSpaceWSClient.DISCUSSION_SERVICE_ID`.

For information about how to use these APIs, see Section 27.2.5.3.8, "Retrieving RSS Feed URLs for Group Space Services.”
17.3.8 Using Custom Discussions APIs

The back-end Oracle WebCenter 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 product area).

17.3.9 Using the Discussions Service REST APIs

Oracle WebCenter provides REST APIs to support the Discussions service. Use the Discussions service REST APIs to post, read, update, and delete discussion forums, topics, and messages.

This section describes the REST APIs associated with the Discussions service. It contains the following subsections:

- Section 17.3.9.1, "Discussions Entry Point"
- Section 17.3.9.2, "Discussions Resource Type Taxonomy"
- Section 17.3.9.3, "Security Considerations"
- Section 17.3.9.4, "Discussions Resource Types"

For an introduction to the REST APIs, see Section 28, "Using Oracle WebCenter REST APIs."

17.3.9.1 Discussions Entry Point

Each REST service has a link element within the Resource Index that provides the entry point for that service. To find the entry point for the Discussions service, find the link element with a resourceType of:

```text
urn:oracle:webcenter:discussions:forums
```

The corresponding href or template element provides the URI entry point. The client sends HTTP requests to this entry point to work with the Discussions service.

For more information about the Resource Index, see Section 28.3.1, "The Resource Index."

For more information about resource types, see Section 28.3.2.1, "Resource Type."

17.3.9.2 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 28.3.2.1, "Resource Type."

The taxonomy for the Discussions service is:

```text
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).

17.3.9.3 Security Considerations
There are no specific security considerations for this service. For general security considerations, see Section 28.6, "Security Considerations for WebCenter REST APIs."

17.3.9.4 Discussions Resource Types
This section provides you with all the information you need to know about each resource type. It includes the following subsections:

- Section 17.3.9.4.1, "urn:oracle:webcenter:discussions:forums"
- Section 17.3.9.4.2, "urn:oracle:webcenter:discussions:forum"
- Section 17.3.9.4.3, "urn:oracle:webcenter:discussions:forum:topics"
- Section 17.3.9.4.4, "urn:oracle:webcenter:discussions:forum:topic"
- Section 17.3.9.4.5, "urn:oracle:webcenter:discussions:forum:topic:messages"
- Section 17.3.9.4.6, "urn:oracle:webcenter:discussions:forum:topic:message"

17.3.9.4.1 urn:oracle:webcenter:discussions:forums
Use this resource type to identify the URI to use to read (GET) and write (POST) discussion forums. The response from a GET operation includes each forum in this collection of forums, and each forum includes links used to operate on that forum. The response from a POST operation includes the forum that was created in this collection of forums and a link to operate on that forum.

Navigation Paths to forums
This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
    forums
```
```
resourceindex
    spaces
        spaces:resourceindex
            spaces:forums
```

Supported Methods for forums
The following methods are supported by this resource:

- **GET**
  - request - body: none, Parameters: startIndex, itemsPerPage (pagination)
  - response - body: forums
- **POST**
  - request - body: forum
  - response - body: forum

For more information, see Section 28.3.2.5, "Templates."
Resource Types Linked to From forums

Table 17–7 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:discussions:forums</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:discussions:forum</td>
</tr>
</tbody>
</table>

17.3.9.4.2  urn:oracle:webcenter:discussions:forum  Use this resource type to identify the URI to use to read (GET), update (PUT), and delete (DELETE) a specific discussion forum. The response from a GET operation includes the specific forum identified by the URI. The response from a PUT operation includes the modified version of the forum identified by the URI. The response from a DELETE operation is a 204.

Navigation Paths to forum

This section shows how the client can navigate through the hypermedia to access this resource:

resourceindex
   forums
      forum

resourceindex
   spaces
      spaces:resourceindex
         spaces:forums
            forum

resourceindex
   activities
      forum

Supported Methods for forum

The following methods are supported by this resource:

- **GET**
  - request - body: none
  - response - body: forum

- **PUT**
  - request - body: forum
  - response - body: forum

- **DELETE**
  - request - body: none
  - response - body: none

Writable Elements for forum

Table 17–8 lists the writable elements for this resource.
Read-only Elements for forum

Table 17–9 lists the read-only 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>id</td>
<td>Integer</td>
<td>Yes</td>
<td></td>
<td>ID of the forum</td>
</tr>
<tr>
<td>parentId</td>
<td>Integer</td>
<td>No</td>
<td></td>
<td>ID of the parent category</td>
</tr>
<tr>
<td>createdBy</td>
<td>String</td>
<td>No</td>
<td></td>
<td>ID of the user that created the forum</td>
</tr>
<tr>
<td>author</td>
<td>personReference</td>
<td>No</td>
<td></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>No</td>
<td></td>
<td>Date on which the forum was created</td>
</tr>
<tr>
<td>updatedBy</td>
<td>String</td>
<td>No</td>
<td></td>
<td>ID of the user that performed the last modification</td>
</tr>
<tr>
<td>modifiedBy</td>
<td>personReference</td>
<td>No</td>
<td></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>No</td>
<td></td>
<td>Date on which the forum was last modified</td>
</tr>
<tr>
<td>webUrl</td>
<td>String</td>
<td>No</td>
<td></td>
<td>URL for direct access to the Oracle WebCenter Discussions Server</td>
</tr>
<tr>
<td>topicCount</td>
<td>Integer</td>
<td>No</td>
<td></td>
<td>Number of topics</td>
</tr>
<tr>
<td>messageCount</td>
<td>Integer</td>
<td>No</td>
<td></td>
<td>Number of messages</td>
</tr>
<tr>
<td>locked</td>
<td>Boolean</td>
<td>No</td>
<td></td>
<td>True if this forum is locked</td>
</tr>
<tr>
<td>favorite</td>
<td>Boolean</td>
<td>No</td>
<td></td>
<td>True if this forum is marked as a favorite</td>
</tr>
</tbody>
</table>

Resource Types Linked to From forum

Table 17–10 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:discussions:forum</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:discussions:forum:topics</td>
</tr>
</tbody>
</table>

17.3.9.4.3 urn:oracle:webcenter:discussions:forum:topics Use this resource type to identify the URI to use to read (GET) and write (POST) discussion topics. The response from a GET operation includes each topic in this collection of topics, and each topic includes
links used to operate on that topic. The response from a POST operation includes the topic that was created in this collection of topics and a link to operate on that topic.

**Navigation Paths to topics**

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
  forums
    topics

resourceindex
  spaces
    spaces:resourceindex
      spaces:forums
        topics
```

**Supported Methods for topics**

The following methods are supported by this resource:

- **GET**
  - request - body: none, Parameters: startIndex, itemsPerPage (pagination)
  - response - body: topics

- **POST**
  - request - body: topic
  - response - body: topic

For more information, see Section 28.3.2.5, "Templates."

**Resource Types Linked to From topics**

Table 17–11 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:topics</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:discussions:forum:topic</td>
</tr>
</tbody>
</table>

17.3.9.4.4  **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 17–12 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 17–13 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>
</tbody>
</table>
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.

### Resource Types Linked to From topic

Table 17–14 lists the resource types that the client can link to from 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 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 last 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 Oracle WebCenter 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>

### 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
```

17.3.9.4.5  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.

### Table 17–14 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:topic:messages</td>
</tr>
</tbody>
</table>
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 28.3.2.5, "Templates."

Resource Types Linked to From messages
Table 17–15 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:discussions:forum:topic:messages</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:discussions:forum:topic:message</td>
</tr>
</tbody>
</table>

17.3.9.4.6  **urn:oracle:webcenter:discussions:forum:topic:message**  Use this resource type to identify the URI to use to read (GET), update (PUT), and delete (DELETE) a specific discussion topic message. The response from a GET operation includes the specific message identified by the URI. The response from a PUT operation includes the modified version of the message identified by the URI. The response from a DELETE operation is a 204.

Navigation Paths to message
This section shows how the client can navigate through the hypermedia to access this resource:

- resourceindex
- forums
- topics
- messages
- message

- resourceindex
- spaces
- spaces:resourceindex
- spaces:forums
- topics
- messages
- message

Supported Methods for message
The following methods are supported by this resource:

- **GET**
  - request - body: none
- **response - body:** message

- **PUT**
  - **request - body:** message
  - **response - body:** message

- **DELETE**
  - **request - body:** none
  - **response - body:** none

**Writable Elements for message**

Table 17–16 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 17–17 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 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 Oracle WebCenter 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 17–18 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:discussions:forum:topic:message</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:discussions:forum:topic:messages</td>
</tr>
</tbody>
</table>

17.3.10 Troubleshooting the Discussions Service

This section describes common problems and solutions for the Discussions service.

Problem
It can take several minutes to fetch data from recursive forums. For example, if you drop the Discussions task flow on the page with the categoryId and forumId parameters left blank and you set the showrecursive parameter to ${'true'}, then it could take several minutes to scroll through all the forums.

Solution
The option to show recursive forums is not recommended for performance reasons.

Problem
Users can log on to Oracle WebCenter but cannot log on to Oracle WebCenter Discussions.

Solution
Make sure that Oracle WebCenter Discussions is configured to use the same LDAP provider as the custom WebCenter application. Contact your administrator for details. For information about configuring LDAP, see the "Configuring the Identity Store" section in the Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter.

Problem
When you access a discussions forum from your custom WebCenter application, the following error message is displayed:

No default or active connection available for: Discussion Forum

Solution
Ensure the following:
- Ensure that you have created a WebCenter Discussion connection.
- Ensure the required WebCenter Discussion connection is marked as the default connection.

Problem
While performing certain operations on Discussions Manager task flows, you encounter error messages like "Forum not found" or "Topic not found."

Solution
Check the following:
- Ensure that the Oracle WebCenter 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 WebCenter application is deployed.
Integrating the Instant Messaging and Presence Service

This chapter explains how to integrate the Instant Messaging and Presence (IMP) service in a WebCenter application at design time.

For more information about managing and including instant messaging and presence, see:

- "Managing the Instant Messaging and Presence Service" in the Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter
- "Working with the Instant Messaging and Presence Service" in the Oracle Fusion Middleware User’s Guide for Oracle WebCenter

This chapter includes the following sections:

- Section 18.1, "Introduction to the IMP Service"
- Section 18.2, "Basic Configuration for the IMP Service"
- Section 18.3, "Advanced Information for the IMP Service"

18.1 Introduction to the IMP Service

The IMP service enables you to observe the presence status (online, offline, busy, or away) of other authenticated application users. Additionally, it provides instant access to interaction options, such as instant messages, emails, and phone calls.

WebCenter services that have user names with the same identity can integrate with the IMP service; for example, Discussions, Documents, or Mail.

This section provides an overview of IMP features and requirements. It contains the following subsections:

- Section 18.1.1, "Understanding the IMP Service"
- Section 18.1.2, "Requirements for IMP"

18.1.1 Understanding the IMP Service

Figure 18–1 shows the Presence icon indicating a user who is online.

Figure 18–1 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 18–2).

**Figure 18–2 Presence Icon Context Menu**

The context menu can include the following actions:

- **Send Mail** (This opens a compose window for the mail client set, either WebCenter Mail service or a local mail client.)
- **View Profile** (This opens the selected user’s profile page, with information such as email ID and contact numbers.)
- **Send Instant Message** (This opens the instant message client running on the computer, either Oracle Instant Messenger or Microsoft Communicator.)
- **Change Credentials** (This works as an alternative to using an external application. This option is available with Microsoft Office Live Communications Server (LCS) and Microsoft Office Communications Server (OCS).)
- **Start Phone Conference** (For presence servers that support the phone facility, this initiates a call between you and the selected user.)

You can enable your users to pull in existing instant messaging contacts (or buddies) and communicate with those contacts within the context of your application. Next to a contact name is an icon that indicates the presence state of each contact. Users can sort contacts can by name or online status. For more information, see Section 18.2.2.3, "Adding the Buddies Task Flow."

For detailed information about the IMP service at runtime, including screen shots and descriptions of the presence status options, see *Oracle Fusion Middleware User's Guide for Oracle WebCenter.*

### 18.1.2 Requirements for IMP

The IMP service requires a back-end presence server. WebCenter is certified with Microsoft Office Live Communications Server (LCS) 2005 and Microsoft Office Communications Server (OCS) 2007, and it can integrate with other presence servers. Oracle WebLogic Communications Services (OWLCS) 11g is available for download on Oracle Technology Network (OTN) at [http://www.oracle.com/technology/index.html](http://www.oracle.com/technology/index.html).

### 18.2 Basic Configuration for the IMP Service

This section describes the steps required for adding the IMP service to your application. It contains the following subsections:

- Section 18.2.1, "Setting up Connections for the IMP Service"
- Section 18.2.2, "Adding the IMP Service at Design Time"
- Section 18.2.3, "Setting Security for the IMP Service"
18.2.1 Setting up Connections for the IMP Service

After the presence server is properly installed and running, you must add a connection to it. This section describes how. It contains the following subsections:

- Section 18.2.1.1, "IMP Service Connections"
- Section 18.2.1.2, "How to Set Up Microsoft LCS Connections for the IMP Service"
- Section 18.2.1.3, "How to Set Up Microsoft OCS Connections for the IMP Service"
- Section 18.2.1.4, "How to Set Up OWLCS Connections for the IMP Service"

18.2.1.1 IMP Service Connections

The IMP service requires an Instant Messaging and Presence connection to the presence server (Microsoft LCS, Microsoft OCS, or OWLCS).

When a WebCenter 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. For Microsoft LCS and OCS connections, an external application is mandatory. For OWLCS connections, use identity propagation (instead of an external application). This setup enables you to configure additional security with Web Services Security (WS-Security).

The connection appends the domain provided with in the connection to the user name to create the Session Initiation Protocol (SIP) address. If the domain is oracle.com and presence is requested for user with name john, then the IM address resolved is john@oracle.com.

---

**Note:** While you can set up the connections to back-end servers at design time in Oracle JDeveloper, you can later add, delete, or modify connections in your deployed environment using Enterprise Manager Fusion Middleware Control. For more information, see Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter.

---

18.2.1.2 How to Set Up Microsoft LCS Connections for the IMP Service

To set up the connection to the LCS presence server:

1. In Oracle JDeveloper, open the application in which to consume the Instant Messaging and Presence service.
2. Ensure that security is enabled in your application according to the steps in Chapter 3.5, "Implementing Security in Your Application."
3. In the Application Navigator, under Application Resources, right-click Connections, then select Instant Messaging and Presence from the list.
4. 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.
5. On the Name page, for Connection Name, enter a unique name for your connection.
   No other connection should have the same name.
6. From the Connection Type list, select Microsoft Live Communication Server 2005.

7. Select the Set as default connection checkbox to use this as the default connection, as shown in Figure 18–9.

Figure 18–3  Create LCS Instant Messaging and Presence Connection, Step 1

8. Click Next

9. On the General page (Figure 18–4), for base.connection.url and domain, enter the location of your LCS instance.
For example:

- The `base.connection.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` could be `example.com`. The domain specified is used to construct each user's instant messaging ID. For example, if the domain is `oracle.com` and presence is requested for user with name `john` then the IM address resolved is `john@oracle.com`.

- The `connection.time.out` property is optional. It represents the time (in seconds) the service should wait for the server to respond while making the connection. If the presence server does not respond in the given time, then it aborts the connection and reports an error.

- For `LCSPoolName`, enter the name of the pool under which Microsoft Communications Server components are deployed. (See the Microsoft Live Communications Server documentation for more information.)

10. On the same page, select an **External Application** to leverage the authentication mechanism (user names and passwords) on the presence server.

*For LCS connections, an external application is mandatory.* The application maps the presence server user to the application user such that end users do not have to enter their user names and passwords each time they need information. For
detailed information about configuring an external application for the IMP service, see Section 18.2.3, "Setting Security for the IMP Service."

11. Click **Test Connection** to confirm that the connection is good.

12. Click **Next** to create the connection.

13. On the **Additional Properties** page (Figure 18–5), you can optionally add the `primary.domain` parameter.

**Figure 18–5  Create LCS Instant Messaging and Presence Connection, Step 3**

If the WebCenter user identity is qualified with a domain (for example, `john.deo@oracle.com`), and if the presence server domain is different (for example, `john.deo@example.com`) then specify the primary domain `oracle.com` here. If the user identity is qualified with a domain and the presence server uses the same `oracle.com` domain, then it is not necessary that you specify the `primary.domain`.

14. Click **Finish**.

You can see the new IM and presence connection under **Application Resources - Connections**.

**18.2.1.3 How to Set Up Microsoft OCS Connections for the IMP Service**

To set up the connection to the Microsoft OCS presence server:
1. In Oracle JDeveloper, open the application in which you plan to consume the Instant Messaging and Presence service.

2. Ensure that security is enabled in your application according to the steps in Chapter 3.5, "Implementing Security in Your Application."

3. In the Application Navigator, under Application Resources, right-click Connections, then select Instant Messaging and Presence from the list.

4. 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.

5. On the Name page, for Connection Name, enter a unique name for your connection.

No other connection should have the same name.

6. From the Connection Type list, select Microsoft Office Communications Server 2007.

7. Select the Set as default connection checkbox.

The service requires that one connection be marked as the default connection, as shown in Figure 18–6.
8. Click Next ()

9. On the General page (), enter the parameters for your Microsoft OCS instance.
### Figure 18–7  Create Microsoft OCS Instant Messaging and Presence Connection, Step 2

<table>
<thead>
<tr>
<th>General</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>General</td>
</tr>
<tr>
<td>Additional Properties</td>
</tr>
</tbody>
</table>

Enter the basic details of the connection (* = required):

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Property Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>base.connection.url</td>
<td></td>
</tr>
<tr>
<td>domain</td>
<td></td>
</tr>
<tr>
<td>connection.timeout</td>
<td></td>
</tr>
<tr>
<td>userDomain</td>
<td></td>
</tr>
<tr>
<td>ocsServer</td>
<td></td>
</tr>
</tbody>
</table>

External Application:  

Test Connection

For example:

- The `base.connection.url` is the location of your Microsoft OCS 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 Office Communications Server 2007 documentation for more information.)

- The `domain` could be `example.com`. The domain specified is used to construct each user's instant messaging ID. For example, if the domain is `oracle.com` and presence is requested for user with name `john` then the IM address resolved is `john@oracle.com`.

- The `connection.timeout` property is optional. It represents the time (in seconds) the service should wait for the server to respond while making the connection. If the presence server does not respond in the given time, then it aborts the connection and reports an error.

- The `userDomain` is the Active Directory domain on Microsoft OCS. This parameter is mandatory for Microsoft OCS connections.

- For `ocsServer`, enter the name of the pool under which Microsoft Communications Server components are deployed. This parameter is mandatory for Microsoft OCS connections. (See the Microsoft Office Communications Server 2007 documentation for more information.)

10. On the same page, select an **External Application** to leverage the authentication mechanism (user names and passwords) on the presence server.
For Microsoft OCS connections, an external application is mandatory. The application maps the presence server user to the application user such that end users do not have to enter their user names and passwords each time they need information. For detailed information about configuring an external application for the IMP service, see Section 18.2.3, "Setting Security for the IMP Service."

11. Click Test Connection to confirm that the connection is good.

12. Click Next to create the connection.

13. On the Additional Properties page (Figure 18–5), you can optionally add the primary.domain parameter.

Figure 18–8 Create Microsoft OCS Instant Messaging and Presence Connection, Step 3

If the WebCenter user identity is qualified with a domain (for example, john.deo@oracle.com), and if the presence server domain is different (for example, john.deo@example.com) then specify the primary domain oracle.com here. If the user identity is qualified with a domain and the presence server uses the same oracle.com domain, then it is not necessary that you specify the primary.domain.

14. Click Finish.

You can see the new IM and presence connection under Application Resources - Connections.
18.2.1.4 How to Set Up OWLCS Connections for the IMP Service

To set up the connection to the OWLCS presence server:

1. In Oracle JDeveloper, open the application in which you plan to consume the Instant Messaging and Presence service.

2. Ensure that security is enabled in your application according to the steps in Chapter 3.5, "Implementing Security in Your Application."

3. In the Application Navigator, under Application Resources, right-click Connections, then select Instant Messaging and Presence from the list.

4. 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.

5. On the Name page, for Connection Name, enter a unique name for your connection.

   No other connection should have the same name.

6. From the Connection Type list, select Oracle WebLogic Communication Server.

7. Select the Set as default connection checkbox.

   The service requires that one connection be marked as the default connection.

8. Click Next (Figure 18–9).
9. On the **General** page (Figure 18–10), for `base.connection.url` and `domain`, enter the location of your OWLCS instance.
Figure 18–10  Create OWLCS Instant Messaging and Presence Connection, Step 2

For example, the base.connection.url could be http://host:port, and the domain could be example.com. The domain specified is used to construct each user’s instant messaging ID. For example, if the domain is oracle.com and presence is requested for user with name john then the IM address resolved is john@oracle.com.

The following properties are optional:

- **connection.time.out**: The time (in seconds) the service should wait for the server to respond while making the connection. If the presence server does not respond in the given time, then it cancels the connection and reports an error.

- **policyURI**: URI to the security policy that is required for authentication on the OWLCS.

For **Authentication Method**:

- Select **Identity Propagation** to provide secure identity propagation with WS-Security.

---

**Note**: Identity propagation is the recommended authentication method for the IMP service. For more information about identity propagation, see Section 37.11, "Identity Propagation Mechanisms."
Select **External Application** to leverage the authentication mechanism (user names and passwords) on the presence server. 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 more information on configuring an external application for the IMP service, see Section 18.2.3, "Setting Security for the IMP Service."

10. Click **Test Connection** to confirm that the connection is good, then click **Next** to create the connection.

11. On the **Additional Properties** page, add any additional, optional parameters (Figure 18–11).

**Figure 18–11  Create OWLCS Instant Messaging and Presence Connection, Step 3**

Additional properties are necessary only for changing the default configuration of the Click2Dial call service.

**Note:** If values should be secured, then add them with the **Add Secured Property** button.

You can add the following parameters:

- **presence.url**: URL to the presence service. This must be supplied if the presence service is deployed on a separate server.
Basic Configuration for the IMP Service

- **contacts.url**: URL to the contact management service. This must be supplied if the contact management service is deployed on a separate server.

- **call.url**: URL for the third-party call server. If no value is supplied, then this uses the same value as base.connection.url.

- **call.method**: Supports values sip and pstn:
  - When set to sip, the IMP service forwards the user's SIP address to the third-party call service. The third-party call service must decide on the routing of the call.
  - When set to pstn, the user's phone number is based on the user's profile attribute (BUSINESS_PHONE). You can use the connection property call.number.attribute to change this default profile attribute (BUSINESS_PHONE) to any other attribute.

- **call.domain**: The domain name of the pstn gateway. If no domain name is supplied, then this uses the domain value specified when the connection was created. Supply a domain name only when call.method is set to pstn.

- **contact.number.attribute**: The attribute used to read users' phone numbers from the user profile. The default is BUSINESS_PHONE. Supply this attribute value only when call.method is set to pstn.

- **primary.domain**: If the WebCenter user identity is qualified with a domain (for example, john.doe@oracle.com), and if the presence server domain is different (for example, john.doe@example.com) then specify the primary domain oracle.com here. If the user identity is qualified with a domain and the presence server uses the same oracle.com domain, then it is not necessary that you specify the primary.domain.

12. Click Finish.

You can see the new IM and presence connection under Application Resources - Connections.

### 18.2.2 Adding the IMP Service at Design Time

This section explains a basic incorporation of the IMP service into your application. It contains the following subsections:

- **Section 18.2.2.1, "IMP Service Task Flows"**
- **Section 18.2.2.2, "How to Add the IMP Service to your Application"**
- **Section 18.2.2.3, "Adding the Buddies Task Flow"**

#### 18.2.2.1 IMP Service Task Flows

The IMP service includes the Buddies task flow.

#### 18.2.2.2 How to Add the IMP Service to your Application

To add the IMP service to your WebCenter application:

1. Follow the steps described in Chapter 3, "Preparing Your Development Environment" to implement security and create a new customizable page in your application.
2. Open the page on which you want to add the IMP service.
3. Ensure that you have configured your application to connect to the presence server. See Section 18.2.1, "Setting up Connections for the IMP Service."
4. In the Component Palette, drag and drop **Presence** to the page to add the Presence icon.

   Use the **Presence** icon anywhere you want to display a user, for example, as the author of a discussion topic, the sender/recipient of an email, 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 18.3.1, "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:** The Page service enables you to 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.

**18.2.2.3 Adding the Buddies Task Flow**

Optionally, you can add the Buddies task flow to an application page. For the Buddies task flow to work, you must have the IMP service configured in your application. Additionally, ensure that you have a **Presence Data** component tag at the end of the page to see the presence status of buddies.
18.2.3 Setting Security for the IMP Service

The IMP service requires user identity. You should make the WebCenter application secure using the ADF Security wizard and make pages secure in their page definition files. The user name you use to log on to the application is also used to log in to the presence server.

If you have an external application configured, then credentials can be 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 WebCenter application should point to the same identity store. The identity store must be LDAP-based; that is, not file-based with jazn-data.xml.

To access the presence server, the IMP service can use an external application with dedicated user accounts:

- Microsoft LCS and OCS support external application connections. With a secured application, users get buddies and presence status. Microsoft LCS and OCS provide an option for changing external credentials, which works as an alternative to using an external application. A logged-in user can click any Presence context menu and select Change Credentials.

With Microsoft LCS and OCS, if security is required, then they should be on a private trusted network.

- OWLCS supports both identity propagation and external application-based connections. With OWLCS, user creation and deletion is manual. Any time a new user is added to (or removed from) the application’s identity store, the same user must be created in (or removed from) the OWLCS user store.

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 WebCenter 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:
   - 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.

5. From the **Authentication Method** list, select **POST**. This submits login credentials within the body of a form. The external application for the IMP service requires this authentication method.

6. Click **Next**.

7. On the Additional Fields page:

   Click **Add Field**, and add an extra field with the name "Account." Make sure to select the **Display to User** checkbox, as shown in Figure 18–12.

---

**Note:** The external application for the IMP service requires this additional field. It must be displayed to users.
8. The External Application service allows different types of credentials to be associated with a connection:
   - When shared credentials are specified, every authenticated user uses the same credentials to access the external application; that is, the user name and password you can define here. A single presence session is created for all logged-in users. This is not accessible to public users.
   - With public credentials, without authenticating your WebCenter 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 37.2, “Working with External Applications.”

For information about configuring ADF security, see Section 3.5, “Implementing Security in Your Application.”

18.3 Advanced Information for the IMP Service

This section describes optional features available with the IMP service. It contains the following subsections:

- Section 18.3.1, "Customizing IMP Views"
- Section 18.3.2, "Troubleshooting the IMP Service"

18.3.1 Customizing IMP Views

Table 18–1 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 application’s JPSContext.</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>
</tbody>
</table>
18.3.2 Troubleshooting the IMP Service

This section describes common problems and solutions for the IMP service.

Problem
The Presence icon is not visible in your custom WebCenter application.

Solution
Ensure that an IMP connection exists in your application and has been set as active.

Problem
Buddies are not visible from within your custom WebCenter application. Further, the presence status of users is not available.

Solution
This problem may arise due to various reasons. Ensure the following:

- Ensure that the IMP connections are configured properly and the base URL and domain values are correct.
- Ensure that your back-end presence server (Microsoft LCS, Microsoft OCS, or OWLCS) is up and running. A quick way to verify this is to ensure that you can connect to the presence server using a supported SIP client (Microsoft Communicator or Oracle Communicator).
Advanced Information for the IMP Service

- Ensure that you have logged-on with valid user credentials and the user exists on the presence server. For Microsoft LCS and OCS, verify that you have provided correct credentials in the external application.
- Ensure that you have buddies on the presence server.
- If all these settings are working fine, then check with your administrator to ensure that Web Services for the presence server is installed properly and is running.

In addition to verifying these settings, ensure the following if the presence status of a user is not visible:

- Ensure that the user, whose presence you cannot view, has been added as your buddy in the application. You cannot view presence of a user who is not your buddy.
- Ensure that the user is online on the presence server.
- Ensure that the application page contains the **Presence Data** component. The **Presence Data** component must be added as the last tag on the page.

**Problem**

Changes in the presence status of users are not visible in your custom WebCenter application.

**Solution**

For each logged-in user's session, the IMP service fetches the presence information from the presence server and stores it in the presence cache. For any presence or buddy requests, the service returns the data from the cache until the cache expires. The default cache expiry period is 60 seconds.

To view the updated presence status, you can wait for the cache to expire and retrieve the latest presence status.

You can also change the cache expiry time by setting the `rtc.cache.time` service configuration property to the desired value (in seconds). Update `adf-config.xml` to include the highlighted entry, which shows the sample value as 30 seconds. Example 18–1 shows an example.

**Example 18–1 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 Start Phone Conference and Send Instant Message, are not available in the context menu of the IMP service in your custom WebCenter application.

**Solution**

Ensure that IMP service is configured in your custom WebCenter application. Also, ensure the following settings for the various context menu options:

- **Start Phone Conference** option unavailable: Ensure that you are using the OWLCS connection type. For Microsoft LCS and Microsoft OCS, this option is not supported.
- **View Profile** option unavailable: Ensure that your application is secured.
• **Send Instant Message** option unavailable: Ensure that the IMP service is configured in the application.

**Problem**
You are unable to send a message from your custom WebCenter 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. For Microsoft LCS and Microsoft OCS, the supported SIP client is Microsoft Communicator. For OWLCS, the supported SIP client is Oracle Communicator.

**Problem**
You are not able to start a phone conference. Clicking the **Start Phone Conference** option does not initiate any action.

**Solution**
Ensure the following:

• Ensure that you are starting the phone conference with a user other than the logged-in user. The caller and the user being called up should be different users, with different phone numbers.

• Ensure that you have provided correct telephone numbers, in the supported format, in the user profile.

• If you specified the `contact.number.attribute` attribute in the IMP connection, then ensure that the contact numbers are indeed provided under this attribute for users. Otherwise, ensure that contact numbers are provided under the `BUSINESS_PHONE` attribute.

**Problem**
After migrating a WebCenter application from 10.1.3.2 to 11g, the **Presence** icon in the Buddies task flow is not seen.

**Solution**
Change the skin in `trinidad-config.xml` to ADF rich skin:

```xml
<skin-family>blafplus-rich</skin-family>
```

For more information on the `trinidad-config.xml` file, see Section 3.2.2.4, "The WebCenter Application Template Default trinidad-config.xml File."
This chapter explains how to integrate the Mail service in a custom WebCenter application at design time.

For more information about managing and including mail, see:

- "Managing the Mail Service" in the Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter
- "Working with the Mail Service" in the Oracle Fusion Middleware User’s Guide for Oracle WebCenter

This chapter includes the following sections:

- Section 19.1, "Introduction to the Mail Service"
- Section 19.2, "Basic Configuration for the Mail Service"
- Section 19.3, "Advanced Information for the Mail Service"

**Note:** The Mail service supports only the Inbox. No other folders (or moving of messages) are supported.

### 19.1 Introduction to the Mail Service

This section provides an overview of Mail service features and requirements. It contains the following subsections:

- Section 19.1.1, "Understanding the Mail Service"
- Section 19.1.2, "Requirements for the Mail Service"

#### 19.1.1 Understanding the Mail Service

The Mail service enables users to access the inbox of a mail server that supports Internet Message Access Protocol4 (IMAP4) and Simple Mail Transfer Protocol (SMTP). Additionally, it enables users to compose a new mail message from within the application (with attachments) and delete, reply to, and forward messages.

The Mail service does not replace users' mail clients; it simply enables users to access and compose mail in a single, collaborative environment.

With the Mail service, you can do the following:

- Read a message by clicking the linked mail subject.
- View sender details (including date) by expanding the From field.
- Scroll down to see the other messages not in the view. You can navigate among the fetched messages as cached messages.
- Attach a file to a message by expanding the attachment section within the mail dialog and clicking Attach. Specify an attachment in the new dialog pop-up. Remove an attachment from the mail by clicking the Remove Attachment icon.
- Reply to a message by clicking the Reply or Reply All icon.
- Forward a message by clicking the Forward icon.
- Cancel an operation (for example, sending a mail) by clicking Cancel.

Figure 19–1 shows the Mail task flow at runtime. At the top of the view are three elements: a dropdown list that shows the number of mails to display (here, All), the Compose icon, and the Refresh icon. The Refresh icon provides a means of manually checking for new messages in the inbox.

![Mail Service 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 19–2).

![Message Filter](image)

Note: By default, All displays the 50 most recent messages. For information on how to increase this amount, see Section 19.3.2, “Configuring the Number of Mails Displayed.”
Click the **Compose** icon next to the dropdown list to compose a new message right from your application. Clicking this icon displays the Compose page, as shown in Figure 19–3.

Figure 19–3  Compose Mail

Use the **Search** icons to find mail addresses and contacts of users in the LDAP store that the WebCenter application uses. For any user not in the LDAP store, you must enter an explicit mail address.

For more information about the Mail service at runtime, see *Oracle Fusion Middleware User’s Guide for Oracle WebCenter*.

19.1.2 Requirements for the Mail Service

The Mail service requires a mail server that supports IMAP4 and SMTP protocols.

In the WebCenter Spaces application, the Microsoft Exchange mail server is required for automatic creation of distribution lists when group spaces are created. In a custom WebCenter application, this feature may not be desirable. To disable it, do not provide the LDAP (Active Directory) server details in the mail connection.

19.2 Basic Configuration for the Mail Service

This section describes the steps required for adding the Mail service to your application. It contains the following subsections:

- Section 19.2.1, "Setting up Connections for the Mail Service"
- Section 19.2.2, "Adding the Mail Service at Design Time"
- Section 19.2.3, "Setting Security for the Mail Service"
19.2.1 Setting up Connections for the Mail Service

Before you can use the Mail service, you must first set up the connection to your mail server. The Mail service supports any mail server based on IMAP4 and SMTP protocol.

---

**Note:** While you can set up the connections to back-end servers at design time in Oracle JDeveloper, you can later add, delete, or modify connections in your deployed environment using Enterprise Manager Fusion Middleware Control. For more information, *Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter*.

---

To create a connection to your mail server from the application:

1. In Oracle JDeveloper, open the application in which to consume the Mail service.
2. In your application, under Application Resources, right-click Connections, then select Mail from the list.
3. Select to create the Mail service connection in Application Resources.
   
   A connection in Application Resources is available only for that application; while a connection in IDE Connections is available for all applications you create. If you plan to use the connection in other applications, then select IDE Connections to avoid having to re-create it.
4. In the **Connection Name** field, enter a unique name for the connection.
5. When configuring a single mail account, select the **Set as default connection** checkbox to use this as the active connection (**Figure 19–4**).
When configuring multiple mail accounts, you are not necessarily required to select this as the default connection. Keep in mind, however, that the service requires that one connection be marked as the default connection.

**Note:** After you create a connection as the default connection, you cannot edit it so that it is not the default. To use a different default connection, you must create a new connection and mark that as the default connection.

6. On the **General** page, enter values for the following required parameters (Figure 19–5).
Basic Configuration for the Mail Service

19-6
Oracle Fusion Middleware Developer's Guide for Oracle WebCenter

Figure 19–5  Configure a New Mail Connection, Step 2

- mail.imap.host: The location of your IMAP server
- mail.imap.port: The IMAP server port number (default is -1)
- mail.smtp.host: The location of your SMTP server
- mail.smtp.port: The SMTP server port number (default is -1)

Optionally, enter values for the other parameters:
- mail.imap.secured: Indicates (true/false) a secure SSL connection (default is false)
- mail.smtp.secured: Indicates (true/false) a secure SSL connection (default is false)
- ldap.host and ldap.port: These, and all other LDAP values, are required only if Microsoft Exchange is the mail server. For the Mail service to create distribution lists reliably, the WebCenter Spaces application should use the same Active Directory server as Microsoft Exchange.

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 37.2, "Working with External Applications."

---

**Note:** External application credential provisioning is built into the mail connection. You do not need to drop the **External Application - Change Password** task flow on a page.

---

b. On the Name page:

For **Application Name**, enter a unique name to identify the application. This name must be unique not only within the WebCenter application, but also among other connections. Note that you cannot edit this field afterward.

For **Display Name**, enter a name for the application that end users see in the credential provisioning screens.

c. Click **Next**.

d. On the General page, you can optionally enter values if you want the external application for the Mail service to participate in Click Through Login.

For **Login URL**, enter the URL to which the HTML login page is submitted. View the HTML source of the application's login form to retrieve this URL.

For **User Name/ID Field Name**, enter the label that the application uses for the user name field, for example, User Name.

For **Password Field Name** field, enter the label that the application uses for the password field, for example Password.

From the **Authentication Method** list, select **POST**. This submits login credentials within the body of a form. The external application for the Mail service requires this authentication method.

e. Click **Next**.

f. On the Additional Fields page:

Click **Add Field**, and add an extra field with the name **Email Address**. Make sure to select the **Display to User** checkbox, as shown in Figure 19–6.
Figure 19–6  Email Address Additional Field

The External Application service allows different types of credentials to be associated with a connection.

When shared credentials are specified, every authenticated user uses the same credentials to access the external application; that is, the user name and password you define here.

With public credentials, without authenticating your WebCenter 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. You can optionally add LDAP parameters and values for the Active Directory server managing (WebCenter Spaces) group space distribution lists.

    For detailed parameter information, see Table 19–1.

    **Note:** For WebCenter Spaces and the Mail service to share an identity management system for setting up group space mailing lists, you must use Active Directory. For information about installing and configuring mail servers as the WebCenter administrator, see *Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter*.

    **Table 19–1   LDAP Directory Server Configuration Parameters**

    | Field                | Description                                                                 |
    |----------------------|-----------------------------------------------------------------------------|
    | LDAP Host            | Enter the host name of the computer where the LDAP directory server is running. |
    | LDAP Port            | Enter the port on which the LDAP directory server listens.                   |
    | LDAP Base DN         | Enter the base distinguished name for the LDAP schema. For example, CN=Users,DC=oracle,DC=com. |
    | LDAP Domain          | Enter the domain to be appended to distribution list names.                   |
    |                      | In WebCenter Spaces, for example, if the domain value is set to oracle.com, then the Finance Project group space maintains a distribution list named FinanceProject@oracle.com. |
    | LDAP Administrator User Name | Enter the user name of the LDAP directory server administrator. If LDAP is configured to run in secure mode, then add this property (set to true/false) to use LDAP while creating distribution lists. |
    | LDAP Administrator Password | The password is stored in a secured store.                                    |
    | LDAP Default User    | Enter a comma-delimited list of user names to whom you want to grant moderator capabilities. These users become members of every group space distribution list that is created. The users specified must exist in the base LDAP schema (specified in the LDAP Base DN field).
    | ldap.secured         | Indicate whether a secured connection (SSL) is required between the WebCenter application and the LDAP directory server. |

11. Click **Finish**.

    You can see the new mail connection under **Application Resources - Connections**.
19.2.2 Adding the Mail Service at Design Time

This section explains a basic incorporation of the Mail service. It contains the following subsections:

- Section 19.2.2.1, "Mail Service Task Flows"
- Section 19.2.2.2, "How to Add the Mail Service to your Application"

19.2.2.1 Mail Service Task Flows

The Mail service includes one task flow: Mail. This task flow displays a mail inbox.

19.2.2.2 How to Add the Mail Service to your Application

To add the Mail service to your application:

1. Follow the steps described in Chapter 3, "Preparing Your Development Environment" to create a customizable page in your application and implement security.

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 WebCenter application must be made secure using the ADF Security wizard, and the pages must be made secure on the page definition.

4. Open the page on which to add the service.


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.

   Figure 19–7 depicts the Mail task flow where the region parameter tabularView is set to true.

![Mail Task Flow where the Region Parameter Tabular Is Set to True](image-url)
Note: If you created a connection in IDE and not in the application, then the connection must be added to the application. For example, in the Resource Palette under IDE Connections, right-click the connection and select Add to Application.

9. Save your page and run it to the browser.

10. When you run the page with the Mail task flow for the first time, you are prompted to enter external application mail credentials from within the Mail task flow.

Enter the credentials, then click Submit.

Notes:

- All instances of the Mail task flow in an application run against the same mail server, and it serves no purpose to add multiple Mail task flow instances. This is true for all service task flows that require connections to back-end servers, such as task flows from the Discussions or Announcements services.

- After an application has been accessed with saved credentials, those credentials are persisted even after redeployment. To ignore saved credentials from previous deployments, edit appUID in adf-config.xml before redeploying the application. Edit the following value in bold to specify a modified value; for example, add 1 to the last value to update it as WCApp1-1235).

```xml
<adf:adf-properties-child
xmlns="http://xmlns.oracle.com/adf/config/properties">
  <adf-property name="adfAppUID" value="WCApp1-1234"/>
</adf:adf-properties-child>
```

19.2.3 Setting Security for the Mail Service

The Mail service works in a non-secured WebCenter 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.

With an ADF-secured application, the Mail service fetches mail for each logged-in user. The external application credentials for the user name used to log in to the application is fetched and used to log in to the mail server. The recommended approach is to have the mail server and the WebCenter application point to the same identity store.

Note: Even if the WebCenter application and the mail server share the same identity store, the Mail service does not support identity propagation. Single sign-on functionality is enabled through the external application mechanism.

For information about configuring ADF security, see Section 3.5, "Implementing Security in Your Application."
For information about using external applications, see Section 37.2, "Working with External Applications."

19.3 Advanced Information for the Mail Service

This section describes optional features available with the Mail service. It contains the following subsections:

- Section 19.3.1, "Invoking the Mail Compose Page"
- Section 19.3.2, "Configuring the Number of Mails Displayed"
- Section 19.3.3, "Troubleshooting the Mail Service"

19.3.1 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 19–1 Invoking the Mail Compose Page

/oracle/workplace/collab/mail/view/jsf/pages/ComposeView.jspx

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.

Example 19–2 shows parameters for the Mail Compose page (ComposeView.jspx):

Example 19–2 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="#{'Testing Mail......'}" to="#pageFlowScope['collab.mail.compose.subject']"/>
    <af:setActionListener from="#{'Testing 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:commandLink>

To hide the recipients fields (like To, Cc and Bcc), set the following action listener in ComposeView.jspx:

Example 19–3 Hiding Recipient Fields in the Mail Compose Page

<af:setActionListener from="#{'false'}" to="#pageFlowScope['collab.mail.message.showrecipients']"/>
19.3.2 Configuring the Number of Mails Displayed

By default, the Mail service displays the 50 most recent mail messages from your Inbox folder. Providing that your mail server supports the increase in memory cache that fetching additional mail requires, the administrator can change this to a higher number in the adf-config.xml file.

Add the mail.messages.fetch.size property as shown in Example 19–4:

**Example 19–4 Increasing the Number of Mails Displayed**

```xml
<adf-collaboration-config xmlns="http://xmlns.oracle.com/webcenter/collab/config">
  <service-config serviceId="oracle.webcenter.collab.mail">
    <property name="mail.messages.fetch.size" value="500"/>
  </service-config>
</adf-collaboration-config>
```

Alternatively, your Fusion Middleware administrator can increase this value with the WLST command `setMailServiceProperty`. For more information, see the *Oracle Fusion Middleware WebLogic Scripting Tool Command Reference*.

This change applies to all users; that is, following Example 19–4, all users see 500 recent mails in their Inbox in the WebCenter application. Increasing the number of messages shown correspondingly increases the cache size on the WebCenter application. Take care to set this to a reasonable size.

19.3.3 Troubleshooting the Mail Service

This section describes common problems and solutions for the Mail service.

**Problem**

Users are not able to retrieve their mail messages or send mail messages from within the custom WebCenter application.

**Solution**

Ensure the following:

- Ensure that a Mail service connection exists in your application.
- Ensure the required Mail service connection is marked as the default connection.
- Ensure the mail server configured in the connection is up and running. You can try connecting from any other mail client to ensure that the connection details are correct.

**Problem**

The Mail service within your custom WebCenter application requires users to log in, but users are not able to authenticate and do not see any content at runtime. When users access the Mail service, it throws the ExtApp Authorization Exception.

**Solution**

The Mail service works in a non-secured custom WebCenter 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 custom WebCenter application to run in secure mode, then you must configure ADF security for your application.

**Problem**
When users receive mail in WebCenter 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).
The People Connections service provides social networking tools for use in enhancing connection and communication within a project team and throughout an enterprise. It does this through a set of features that include:

- **Activity Stream** for viewing user activities generated through application or social networking actions.
- **Connections** for connecting to other application users to share information, comment on performance, exchange messages, and track activity.
- **Feedback** for giving *ad hoc* performance feedback to other users.
- **Message Board** for posting messages to other users.
- **Profile** for entering personal contact information and viewing the contact information of other users.

This chapter explains how to integrate the People Connections service with a custom WebCenter application and provides information about the REST APIs available for use in connecting to the People Connections service from outside the application framework.

This chapter includes the following sections:

- Section 20.1, "Introduction to the People Connections Service"
- Section 20.2, "Basic Configuration for the People Connections Service"
- Section 20.3, "Advanced Information for the People Connections Service"
- Section 20.4, "Troubleshooting the People Connections Service"

### 20.1 Introduction to the People Connections Service

The People Connections service provides social networking tools for creating, interacting with, and tracking the activities of one's enterprise connections. Its features enable users to manage their personal profiles, access the profiles of other users, provide ad hoc feedback, post messages, track activities, and connect with others.

In a production environment, an enterprise can leverage its back-end identity store as a means of providing People Connections with a population of potential connections. In a development environment, developers can add test-users to the `jazn-data.xml` file.
Introduction to the People Connections Service

This section provides an overview of the People Connections service. It contains the following subsections:

- **Section 20.1.1, "Understanding the People Connections Service"
- **Section 20.1.2, "Requirements for the People Connections Service"

### 20.1.1 Understanding the People Connections Service

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 section introduces the five People Connections categories. It contains the following subsections:

- **Section 20.1.1.1, "Activity Stream"
- **Section 20.1.1.2, "Connections"
- **Section 20.1.1.3, "Feedback"
- **Section 20.1.1.4, "Message Board"
- **Section 20.1.1.5, "Profile"

#### See Also:
For information about the People Connections service at runtime, see the chapter, "Working with the People Connections Service," in the Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

For information about People Connections task flows, see Section 20.2.2.1, "People Connections Service Task Flows."

### 20.1.1.1 Activity Stream

The Activity Stream feature tracks application activities. Table 20–1 lists the types of activities that may 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></td>
<td>■ Delete announcement</td>
</tr>
<tr>
<td>Discussions</td>
<td>■ Create forum</td>
</tr>
<tr>
<td></td>
<td>■ Delete forum</td>
</tr>
<tr>
<td></td>
<td>■ Create topic</td>
</tr>
<tr>
<td></td>
<td>■ Delete topic</td>
</tr>
<tr>
<td></td>
<td>■ Reply to topic</td>
</tr>
<tr>
<td></td>
<td>■ Delete reply</td>
</tr>
</tbody>
</table>

See Also: For information about connecting to a back-end (LDAP) identity store for the production version of your application, see the chapter, "Managing Security," in the Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter. For information about creating test users in jazn-data.xml, see Section 11.2.2, "How to Define Roles and Grant Privileges in the jazn-data.xml File."
Activity Stream compares somewhat to the Recent Activities service, which also tracks and reports on application activities (for more information, see Section 25, "Integrating the Recent Activities Service"). Both track the application activities of integrated services, though Activity Stream tracks a broader range of services. For example, Recent Activities tracks the Documents, Announcements, Discussions, and Page services. Activity Stream tracks these services as well as People Connections and Wikis and Blogs. Recent Activities tracks activities no matter who performs the action. Activity Stream tracks activities performed by a user's connections and includes information about who performed the activity. Recent Activities does not include names.

<table>
<thead>
<tr>
<th>Service</th>
<th>Tracked Activities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Documents</td>
<td>Create document&lt;br&gt; Edit document&lt;br&gt; Delete document&lt;br&gt; Add tag&lt;br&gt; Remove tag</td>
</tr>
<tr>
<td>Page</td>
<td>The following activities are tracked when they are performed within the service framework scope (for example, if a user creates pages under /myapp/mypages/, the page operations are not tracked):&lt;br&gt; Create page&lt;br&gt; Edit page&lt;br&gt; Delete page&lt;br&gt; Add tag&lt;br&gt; Remove tag</td>
</tr>
<tr>
<td>People Connections</td>
<td>People are connected&lt;br&gt; Message Board post&lt;br&gt; Feedback post&lt;br&gt; Photo updated&lt;br&gt; Profile updated&lt;br&gt; Personal status note updated</td>
</tr>
<tr>
<td>Wiki and Blog</td>
<td>Create wiki&lt;br&gt; Edit wiki&lt;br&gt; Add wiki comment&lt;br&gt; Delete wiki&lt;br&gt; Create blog entry&lt;br&gt; Edit blog entry&lt;br&gt; Add blog entry comment&lt;br&gt; Delete blog entry</td>
</tr>
</tbody>
</table>

Note: An Oracle WebCenter Wiki and Blog Server scheduled job (ActivityPublishJob) must be run in order for wiki and blog activities to be published to the Activity Stream. For more information, see the section, "About Administration Mode," in the Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter.
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.

**See Also:** For more information about Activity Stream, see Section 20.2.2.1.1, "Activity Stream Task Flows."

### 20.1.1.2 Connections

Connections provides users with a means of managing their own connections and viewing the connections of others. Use Connections to collect business friends and contacts into one or more smaller social groups. Use connections lists to manage the display of your connections.

**See Also:** For more information about Connections, see Section 20.2.2.1.2, "Connections Task Flows."

### 20.1.1.3 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.

**See Also:** For more information about Feedback, see Section 20.2.2.1.3, "Feedback Task Flows."

### 20.1.1.4 Message Board

Message Board provides users with a means of viewing, posting, and managing messages to themselves and to other users.

**See Also:** For more information about Message Board, see Section 20.2.2.1.4, "Message Board Task Flows."

### 20.1.1.5 Profile

Profile provides users with a variety of views into their own and other users’ personal profile information. Such information can include a user's email address, phone number, office location, department, manager, direct reports, and so on. Profile takes the bulk of its information from the back-end identity store that provides your WebCenter 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.

**See Also:** For more information about Profile, see Section 20.2.2.1.5, "Profile Task Flows."

### 20.1.2 Requirements for the People Connections Service

For a successful integration of People Connections with your custom WebCenter application, ensure that the following steps have been taken:

1. Create a WebCenter application (for more information, see Section 3.2, "Creating a WebCenter Application").
2. Set up the WebCenter schema, and create a database connection to it (for more information, see Chapter 4, "Preparing Your Application for Oracle WebCenter Services").
3. Create a JSF (.jspx) page (for more information, see Section 3.3, "Creating WebCenter Application-Enabled Pages").

4. Secure your application and create some test users (for more information, see Section 11.2.2, "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 20.2.2.2, "How to Add People Connections Service Task Flows to a Page").

6. Provide required values for task flow bindings (for more information, see Section 20.3.2.2, "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.

### 20.2 Basic Configuration for the People Connections Service

This section describes steps required for adding the People Connections service to your application. It includes the following subsections:

- Section 20.2.1, "How to Set Up a Database Connection for the People Connections Service"
- Section 20.2.2, "Adding the People Connections Service at Design Time"
- Section 20.2.3, "Setting Security for the People Connections Service"

#### 20.2.1 How to Set Up a Database Connection for the People Connections Service

The People Connections service requires a connection to a database that has the WebCenter schema installed. The script to create the WebCenter schema is included with the product. This section assumes you have installed the WebCenter schema and have created a JSF (.jspx) page in your WebCenter application.

**See Also:**
- For details about installing the database and the WebCenter schema, see the Oracle Fusion Middleware Installation Guide for Oracle WebCenter.
- For information about creating a WebCenter application, see Section 3.2, "Creating a WebCenter Application."
- For information about creating a JSF (.jspx) page, see Section 3.3, "Creating WebCenter Application-Enabled Pages."

To create the database connection:

1. In the Application Navigator, expand the Application Resources panel.

2. Right-click Connections, then select New Connection and then Database to open the Create Database Connection dialog (Figure 20–1).
3. Enter the following information for your database connection:

- **Connection Name**: WebCenter
- **Connection Type**: Oracle (JDBC)
- **Username**: username (a user with established access to the database)
- **Password**: password (the specified user’s password)
- **Host Name**: <host where you will install the WebCenter schema> (for example, localhost)
- **JDBC Port**: <port> (for example, 1521)
- **SID**: <system identifier for the database with the same JDBC port> (for example, ORCL)

You must enter the **Connection Name** exactly as WebCenter.
Note: There are cases when you may want to leverage an existing database connection for WebCenter services, and it may not be possible to change the database connection name to WebCenter. To enable WebCenter services to use a database connection with a name other than WebCenter, you must add the following `<data-source>` tag as a child of the `<wpsC:adf-service-config>` element in the `adf-config.xml` file. (`adf-service-config` is a child of `adf-config`, and `data-source` is a child of `adf-service-config` or a sibling of `extension-registry-config`.)

For example:

```xml
<adf-config
    xmlns="http://xmlns.oracle.com/adf/config"
    xmlns:mdsC="http://xmlns.oracle.com/adf/mds/config"
    xmlns:wpsC="http://xmlns.oracle.com/webcenter/framework/service"
    xmlns:jndiC="http://xmlns.oracle.com/adf/jndi/config">
  <wpsC:adf-service-config
      xmlns="http://xmlns.oracle.com/webcenter/framework/service">
    <data-source
        jndi-name="java:/comp/env/jdbc/NewDatabaseConnDS"/>
  </wpsC:adf-service-config>
</adf-config>
```

The syntax of the example name, `java:/comp/env/jdbc/NewDatabaseConnDS`, is derived from the example name `NewDatabaseConn` used to create the database connection in the Create Database Connection dialog.

4. Click OK.

Note: While you can set up the connections to back-end servers at design time in Oracle JDeveloper, you can later add, delete, or modify connections in your deployed environment using Enterprise Manager Fusion Middleware Control. For more information, see the Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter.

20.2.2 Adding the People Connections Service at Design Time

This section provides descriptions of People Connections task flows and steps you through the addition of the People Connections service to your application. It includes the following subsections:

- Section 20.2.2.1, "People Connections Service Task Flows"
- Section 20.2.2.2, "How to Add People Connections Service Task Flows to a Page"

20.2.2.1 People Connections Service Task Flows

This section describes and illustrates the task flows available through the People Connections service.

This section includes the following subsections:

- Section 20.2.2.1.1, "Activity Stream Task Flows"
Activity Stream—Provides users with the main view of the most recent activities of their connections (Figure 20–2).

Activity Stream—Quick View—Provides users with a truncated view of streamed activities, where only the most recent of their connections’ activities are shown. Users who want to see all activities can click a More link to navigate to the main view (Activity Stream) of the task flow (Figure 20–3).
20.2.2.1.2 Connections Task Flows  Three Connections task flows are available through the Resource Palette:

- Connections—Provides users with a means of viewing and managing their connections, creating connections lists, and sending and responding to invitations to connect (Figure 20–4).
Connections - Card—Provides a choice of three views:

- Card Style displays a connection’s photo, status message, and instant contact option (through the Presence service, for more information, see Chapter 18, “Integrating the Instant Messaging and Presence Service”).
- Iconic displays a connection's photo and user name, which links to a summary view of the user’s profile.
- List Style 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.

Select one of these views through task flow bindings (for more information, see Section 20.3.2, "Customizing People Connections Task Flows Using Task Flow Binding Parameters") (Figure 20–5).
Connections - Quick View—Provides users with a view of their connections' uploaded profile photos and a link to instant contact options (Figure 20–6).

Figure 20–6  Connections - Quick View Task Flow

Feedback Task Flows  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 20–7).
Feedback - Quick View—Provides users with a truncated view of the Feedback task flow, where only the most recent posts are shown. When there are more posts than can be displayed in the initial view, users can click a More link to navigate to the main view of the Feedback task flow. When user A accesses user B’s view of Feedback - Quick View, user A additionally sees an Add Feedback option (Figure 20–8).
20.2.2.1.4 **Message Board Task Flows** Two Message Board task flows are available through the Resource Palette:

- **Message Board**—Message Board provides users with the main view of Message Board messages and a means of adding, viewing, and managing Message Board messages. Management tasks include editing, deleting, and hiding messages (Figure 20–9).
Basic Configuration for the People Connections Service

Figure 20–9  Message Board Task Flow

Message Board - Quick View—Provides users with a truncated view of Message Board messages and a means of adding and viewing such messages. Users can click a More link to navigate to the Detailed View (Figure 20–10).

Figure 20–10  Message Board - Quick View Task Flow

20.2.2.1.5 Profile Task Flows  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 20–11).
Profile - Snapshot—Exposes any personal image a user may have uploaded and provides an Edit link for entering a personal status message (Figure 20–12).

Figure 20–12   Profile - Snapshot Task Flow

- Email: sip:monty@example.com
- Phone: not updated
- Time Zone: svcd f323eweq
- Tags Used: spaces reports social-networking web2.0
- Items Tagged: Events Discussions Welcome
Basic Configuration for the People Connections Service

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 20.2.2.2, “How to Add People Connections Service Task Flows to a Page,” and see Section 20.3.2.2, “People Connections Task Flow Binding Parameters.”

- Profile Gallery—Provides a summary view of each People Connections task flow. For example, it includes Activity Stream - Quick View, Connections - Quick View, and so on (Figure 20–13).

Figure 20–13 Profile Gallery Task Flow

20.2.2.2 How to Add People Connections Service Task Flows to a Page

The 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 application:

1. Prepare your application as described in Section 20.1.2, “Requirements for the People Connections Service.”
2. Open the page on which to add a People Connections task flow.
3. In the Resource Palette, open My Catalogs, then WebCenter 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 Section 20.3.2.2, "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:

```xml
<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 Section 20.3.2.2, "People Connections Task Flow Binding Parameters."

---

### 20.2.3 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.

- To secure the application, follow the steps documented in Section 4.1.1.1, "Implementing Security for Services," with the following exceptions:

- To create test users, follow the steps outlined in Section 11.2.2, "How to Define Roles and Grant Privileges in the jazn-data.xml File."

  To add additional user attributes, such as `business_email`, `title`, or `department`, to a test user:

  1. In `jazn-data.xml` double-click the `jazn-data.xml` node to open it in the IDE.

  2. Select the tab for **Source** at the bottom of the IDE 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@domain.com for the user monty, inside the <user> node for monty add:

<property name="business_email" value="monty@domain.com"/>

20.3 Advanced Information for the People Connections Service

This section describes advanced features available with the People Connections service. It includes the following subsections:

- Section 20.3.1, "Establishing Site-Level Settings for People Connections Features"
- Section 20.3.2, "Customizing People Connections Task Flows Using Task Flow Binding Parameters"
- Section 20.3.3, "Using the People Connections Service REST APIs"

20.3.1 Establishing Site-Level Settings for People Connections Features

Two scripts are available to configure and revise People Connections site-level settings:

- \texttt{RCUHOME/rcu/integration/webcenter/sql/oracle/settings-insert.sql}
- \texttt{RCUHOME/rcu/integration/webcenter/sql/oracle/settings-update.sql}

The variable \texttt{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. \texttt{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 contains the following subsections:

- Section 20.3.1.1, "Understanding the People Connections Site-Level Setting Scripts"
- Section 20.3.1.2, "Supported Site-Level Settings for People Connections Features"
- Section 20.3.1.3, "Example: Configuring Connections to Accept Invitations Automatically"

20.3.1.1 Understanding the People Connections Site-Level Setting Scripts

All site-level settings for People Connections features are stored in the WebCenter schema table \texttt{WC_PPL_COMMONSETTING}. 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 contains the following subsections:

- Section 20.3.1.1.1, "Preparing and Running settings-insert.sql"
- Section 20.3.1.2.1, "Preparing and Running settings-update.sql"

20.3.1.1.1 Preparing and Running settings-insert.sql

Before you can change People Connections site-level settings, you must run the \texttt{settings-insert.sql} script once. The script has \texttt{INSERT} statements for all supported settings. All \texttt{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 20.3.1.3, "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.

20.3.1.1.2 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 20.3.1.3, "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 20.3.1.2, "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 custom WebCenter application is running, restart it for the changes to take effect.

20.3.1.2 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 20.3.1.2.1, "Activity Stream Site-Level Settings"
- Section 20.3.1.2.2, "Connections Site-Level Settings"
- Section 20.3.1.2.3, "Feedback Site-Level Settings"
- Section 20.3.1.2.4, "Message Board Site-Level Settings"
- Section 20.3.1.2.5, "Profile Site-Level Settings"

20.3.1.2.1 Activity Stream Site-Level Settings

Table 20–2 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 20–3.

<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>For determining 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>For indicating 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>+, -</td>
</tr>
<tr>
<td>connectionsActivities.value</td>
<td>For specifying the users for which to show personal space activities in a user's Activity Stream. Users can override this setting on a task flow instance.</td>
<td>NONE, ALL</td>
</tr>
</tbody>
</table>

Table 20–2 Site-Level Settings for Activity Stream
**Table 20–2  (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>groupSpaceActivities.value</td>
<td>For specifying the group spaces for which to show activities in a user's Activity Stream</td>
<td>GSNONE, GSALL, GSSELECTED&lt;br&gt;GSNONE=No group space activities are shown&lt;br&gt;GSALL=Activities are shown for all of the group spaces of which the user is a member and on which the user has view access.&lt;br&gt;GSSELECTED=Activities are shown for a predefined list of group spaces of which the user is a member and on which the current user has view access. The predefined list is specified through the setting groupSpaceActivities.groupSpaceGuids.</td>
</tr>
<tr>
<td>groupSpaceActivities.groupSpaceGuids</td>
<td>For providing the GUIDs of the group spaces for which to show activities in a user's Activity Stream</td>
<td>LIST: followed by a comma-separated list of group space GUIDs without any spaces in-between. For example: LIST: sbba12345xyz1,sabcd54321abc&lt;br&gt;Up to 100 group space GUIDs can be specified in this fashion.</td>
</tr>
<tr>
<td>servicePublishedSettings[serviceId]</td>
<td>For indicating whether to show activities published by the WebCenter service serviceId in a user's Activity Stream</td>
<td>+, -&lt;br&gt;+=TRUE&lt;br&gt;-=FALSE&lt;br&gt;A serviceId is the WebCenter service ID of any service that publishes activities. For a list of service IDs, see Table 20–3. For example: servicePublishedSettings[oracle.webcenter.community] Users can override this setting on a task flow instance.</td>
</tr>
</tbody>
</table>

**Table 20–3  Service IDs**

<table>
<thead>
<tr>
<th>Service</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</td>
<td>oracle.webcenter.doclib</td>
</tr>
<tr>
<td>Events</td>
<td>oracle.webcenter.collab.calendar.community</td>
</tr>
<tr>
<td>Group Space Management</td>
<td>oracle.webcenter.community</td>
</tr>
<tr>
<td>Instant Messaging and Presence (IMP)</td>
<td>oracle.webcenter.collab.rtc</td>
</tr>
</tbody>
</table>
Table 20–3 (Cont.) Service IDs

<table>
<thead>
<tr>
<th>Service</th>
<th>ID</th>
</tr>
</thead>
<tbody>
<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>Notes</td>
<td>oracle.webcenter.note</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>Recent Activities</td>
<td>oracle.webcenter.recentactivity</td>
</tr>
<tr>
<td>RSS</td>
<td>oracle.webcenter.rss</td>
</tr>
<tr>
<td>Search</td>
<td>oracle.webcenter.search</td>
</tr>
<tr>
<td>Tags</td>
<td>oracle.webcenter.tagging</td>
</tr>
<tr>
<td>Wikis and Blogs</td>
<td>oracle.webcenter.wiki</td>
</tr>
<tr>
<td>Worklist</td>
<td>oracle.webcenter.worklist</td>
</tr>
</tbody>
</table>

20.3.1.2.2 Connections Site-Level Settings

Table 20–4 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 20–3.

Table 20–4 Site-Level Settings for Connections

<table>
<thead>
<tr>
<th>Setting Key</th>
<th>Description</th>
<th>Valid Site-Level Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>autoAcceptInvitations</td>
<td>For specifying that connection invitations are accepted automatically by default</td>
<td>+, -</td>
</tr>
<tr>
<td></td>
<td>Individual users can override this application-level setting in their own view, provided the application exposes the override control in a user preferences screen.</td>
<td>+=TRUE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-=FALSE</td>
</tr>
<tr>
<td>connectionsVisibility.accessControlSettings.accessControlLevel</td>
<td>For specifying who can view a user's Connections (that is, authenticated users), PUBLIC</td>
<td>SELF, CONNECtIONS, USERS</td>
</tr>
<tr>
<td>connectionsVisibility.personalizable</td>
<td>For indicating whether individual users can override in their own application views the application-level setting for who can view their Connections</td>
<td>+, -</td>
</tr>
<tr>
<td></td>
<td>This setting is honored only when the application exposes the override control in a user preferences screen.</td>
<td>+=TRUE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-=FALSE</td>
</tr>
</tbody>
</table>
**20.3.1.2.3 Feedback Site-Level Settings** Table 20–5 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 20–3.

<table>
<thead>
<tr>
<th>Table 20–5 Site-Level Settings for Feedback</th>
</tr>
</thead>
<tbody>
<tr>
<td>Setting Key</td>
</tr>
<tr>
<td>visibility.accessControlSettings.accessControlLevel</td>
</tr>
<tr>
<td>visibility.endUserConfigurable</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>addControl.accessControlSettings.accessControlLevel</td>
</tr>
<tr>
<td>addControl.endUserConfigurable</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>miniViewRowCount.value</td>
</tr>
<tr>
<td>miniViewRowCount.endUserConfigurable</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>postedUserActions.delete</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

**20.3.1.2.4 Message Board Site-Level Settings** Table 20–6 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 20–6.
### Table 20-6 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>visibility.accessControlSettings.accessControlLevel</td>
<td>For specifying who can view a user's Message Board</td>
<td>SELF, CONNECTIONS, USERS (that is, authenticated users), PUBLIC</td>
</tr>
<tr>
<td>visibility.endUserConfigurable</td>
<td>For indicating whether individual users can override in their own application views the application-level setting for who can view their Message Board</td>
<td>+, -</td>
</tr>
<tr>
<td></td>
<td>This setting is honored only when the application exposes the override control in a user preferences screen.</td>
<td>+=TRUE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-=FALSE</td>
</tr>
<tr>
<td>addControl.accessControlSettings.accessControlLevel</td>
<td>For specifying who can post to a user's Message Board</td>
<td>SELF, CONNECTIONS, USERS (that is, authenticated users), PUBLIC</td>
</tr>
<tr>
<td>addControl.endUserConfigurable</td>
<td>For indicating whether individual users can override in their own application views the application-level setting for who can post to their Message Board</td>
<td>+, -</td>
</tr>
<tr>
<td></td>
<td>This setting is honored only when the application exposes the override control in a user preferences screen.</td>
<td>+=TRUE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-=FALSE</td>
</tr>
<tr>
<td>miniViewRowCount.value</td>
<td>For specifying 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>For indicating 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</td>
<td>+, -</td>
</tr>
<tr>
<td></td>
<td>This setting is honored only when the application exposes the override control in a user preferences screen.</td>
<td>+=TRUE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-=FALSE</td>
</tr>
<tr>
<td>postedUserActions.edit</td>
<td>For indicating whether users are allowed to edit the messages they post on other users' Message Boards</td>
<td>+, -</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+=TRUE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-=FALSE</td>
</tr>
<tr>
<td>postedUserActions.delete</td>
<td>For indicating whether users are allowed to delete messages they post on other users' Message Boards</td>
<td>+, -</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+=TRUE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-=FALSE</td>
</tr>
</tbody>
</table>

#### 20.3.1.2.5 Profile Site-Level Settings

Table 20-7 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 20-3.
### Table 20–7  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>For indicating 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>For specifying 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>For indicating whether individual users can override in their own views the application-level setting for who can view their Profile Gallery</td>
<td>true, false</td>
</tr>
<tr>
<td>general-profile-settings.view.name.profileSectionName.access-control-level</td>
<td>For specifying 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 Note: CONNECTIONS is not a valid value for this setting.</td>
</tr>
</tbody>
</table>

For more information, see Table 20–8, "Profile Section Names".
### Table 20–7  (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><code>general-profile-settings.view.name.profileSectionName.allow-personalize-visibility</code></td>
<td>For indicating whether individual users can override in their own application views the application-level setting for who can view the Profile section identified by <code>profileSectionName</code> 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 20–8, &quot;Profile Section Names&quot;.</td>
<td>true, false</td>
</tr>
<tr>
<td><code>general-profile-settings.view.name.profileSectionName.allow-user-edit</code></td>
<td>For indicating whether users are allowed to edit the Profile section identified by <code>profileSectionName</code> on their own Profiles. For example: <code>general-profile-settings.view.name.personalInfo.allow-user-edit</code> For more information, see Table 20–8, &quot;Profile Section Names&quot;.</td>
<td>true, false</td>
</tr>
<tr>
<td><code>ootb-view-edit-settings.view.name.profileSectionName.profileFieldName.allow-edit</code></td>
<td>For indicating whether users are allowed to update the field identified by <code>profileFieldName</code> when they edit the Profile section identified by <code>profileSectionName</code> on their own profile. For example: <code>ootb-view-edit-settings.view.name.personalInfo.homePhone.allow-edit</code> For more information, see Table 20–8, &quot;Profile Section Names&quot; and Table 20–9, &quot;Profile Field Names&quot;.</td>
<td>true, false</td>
</tr>
</tbody>
</table>

### Table 20–8  Profile Section Names

<table>
<thead>
<tr>
<th>Section Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic</td>
<td>Profile summary&lt;br&gt;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>
<tr>
<td>personalInfo</td>
<td>Personal information</td>
</tr>
</tbody>
</table>

### Table 20–9  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>
</tbody>
</table>
20.3.1.3 Example: Configuring Connections to Accept Invitations Automatically

By default, when users send invitations to connect, recipients must explicitly accept them. Through site-level settings, the application administrator can configure the application to accept invitations automatically. This setting is identified by the key, `autoAcceptInvitations`, and the service ID, `oracle.webcenter.peopleconnections.connections`. To change the setting, the administrator performs the following steps:

1. Open the `settings-insert.sql` script and uncomment the following lines:

   ```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', '-')
   -- ;
   ```

---

### Table 20–9 (Cont.) Profile Field Names

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<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>
After modification, the lines appear as follows:

```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', '-')
;
```

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'
-- ;
```

After the 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'
;
```

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 custom WebCenter 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 20.3.1.2, "Supported Site-Level Settings for People Connections Features."

### 20.3.2 Customizing People Connections Task Flows Using Task Flow Binding Parameters

Each People Connections task flow has a set of required and optional task flow binding parameters. These provide a means of capturing information that is useful to the task flow’s successful function. For example, all People Connections task flows provide a binding parameter for capturing the ID of the current runtime user. This value (typically #{securityContext.userName}) enables the task flow to return People Connections data that is relevant to the current user.
In addition to providing required values for successful task flow rendering, you can use task flow binding parameters to customize the appearance and behavior of a task flow instance. For example, you can use parameter values to determine whether headers and footers are rendered, the number of rows and columns of information to show, whether to apply a filter to returned data, and the like.

You can provide task flow binding parameter values when you drag and drop a task flow onto an application page. Doing so opens the Task Flow Bindings dialog (for more information, see Section 20.2.2.2, "How to Add People Connections Service Task Flows to a Page"). You can also adjust task flow binding parameter values after you have placed a task flow on a page.

This section describes how to revise task flow parameter values at design time and provides a table that lists and describes the binding parameters associated with People Connections task flows. It includes the following sections:

- Section 20.3.2.1, "How to Revise People Connections Task Flow Binding Parameters"
- Section 20.3.2.2, "People Connections Task Flow Binding Parameters"

### 20.3.2.1 How to Revise People Connections Task Flow Binding Parameters

After you have added a task flow to a page, you may want to customize the instance by revising its binding parameter values. This section describes how to access the Edit Task Flow Binding dialog and change binding parameter values.

To access the Edit Task Flow Binding dialog:

1. Open the application page and set the view to bindings by clicking the **Bindings** tab at the bottom of the page.

2. Under **Executables**, double-click the task flow for which to revise task flow binding parameters (Figure 20–14) to open the Edit Task Flow Binding dialog (Figure 20–15).

   **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**.
3. Revise binding parameter values (for more information, see Section 20.3.2.2, "People Connections Task Flow Binding Parameters").

4. Click OK to save your changes and exit the dialog.

5. Save and run your page to see the results.

20.3.2.2 People Connections Task Flow Binding Parameters

Table 20–10 lists and describes task flow binding parameters applicable to the People Connections service.
## Table 20–10  People Connections Service Task Flow Binding Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>connectionListName</td>
<td>The name of a connections list  &lt;br&gt;This parameter is associated with the task flow Connections - Card.</td>
</tr>
<tr>
<td>display_maxConnections</td>
<td>The maximum number of connections to show in the task flow  &lt;br&gt;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.  &lt;br&gt;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.  &lt;br&gt;This parameter is associated with the following task flows:  &lt;br&gt;■ Connections - Card  &lt;br&gt;■ Connections - Quick View</td>
</tr>
<tr>
<td>display_numberOfColumns</td>
<td>The number of columns to show in the task flow  &lt;br&gt;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).  &lt;br&gt;This parameter is associated with the following task flows:  &lt;br&gt;■ Connections - Card  &lt;br&gt;■ Connections - Quick View</td>
</tr>
<tr>
<td>display_numberOfRows</td>
<td>The number of rows to show in the task flow  &lt;br&gt;For example, in a Connections - Card task flow that shows six connections and a value of 2 for display_numberOfColumns, a value of 2 for this parameter means connections are shown in two columns with two rows. That is, four connections are shown. A More link appears at the bottom of the task flow when there are more connections than can 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.  &lt;br&gt;See also display_maxConnections.  &lt;br&gt;This parameter is associated with the following task flows:  &lt;br&gt;■ Connections - Card  &lt;br&gt;■ Connections - Quick View</td>
</tr>
<tr>
<td>display_profileFormat</td>
<td>The default layout style for the task flow  &lt;br&gt;Enter one of the following:  &lt;br&gt;■ vcard—Displays each connection in a virtual business card, including the connection's user name and status message along with quick access buttons for interacting with one's connections.  &lt;br&gt;■ iconic—Displays the connection's personal profile photo and user name.  &lt;br&gt;■ list—Displays connections in a list that includes the personal profile photo, user name, information about recent profile updates, and quick access buttons for interacting with one's connections.  &lt;br&gt;This parameter is associated with the task flow Connections - Card.</td>
</tr>
</tbody>
</table>
### Table 20–10  (Cont.) People Connections Service Task Flow Binding Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>display_removalAllowed</td>
<td>A means of showing or hiding the <code>Remove</code> action on the task flow</td>
</tr>
<tr>
<td></td>
<td>Enter <code>true</code> to show the <code>Remove</code> action. Enter <code>false</code> to hide the <code>Remove</code> action. The default value (<code>true</code>) is applied if the input is invalid.</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the task flow <code>Connections - Card</code>.</td>
</tr>
<tr>
<td>display_sortedBy</td>
<td>The connections sort order</td>
</tr>
<tr>
<td></td>
<td>Enter <code>LAST_ACTIVITY_TIME</code> to sort connections in descending date/time order. Leave this field blank to sort connections alphabetically by name.</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the task flow <code>Connections - Card</code>.</td>
</tr>
<tr>
<td>displayCount</td>
<td>The number of items to show in the task flow</td>
</tr>
<tr>
<td></td>
<td>For example, enter <code>5</code> to specify that a maximum of five items can appear in the task flow. In quick views, a <code>More</code> link appears at the bottom of the task flow when there are more items than the specified number of items. Users click <code>More</code> to open the main view of the task flow where all items are accessible. In main views, <code>Previous</code> and <code>Next</code> links are shown. Users click these to page through entries.</td>
</tr>
<tr>
<td></td>
<td>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</td>
</tr>
<tr>
<td></td>
<td>Messages exceeding the specified value are truncated, and an ellipse (...) is appended to the end.</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the task flow <code>Feedback - Quick View</code>.</td>
</tr>
<tr>
<td>filterPattern</td>
<td>A filter against task flow content</td>
</tr>
<tr>
<td></td>
<td>For example, enter <code>pat</code> to show only those connections named <code>pat</code> (including <code>patrick</code> or <code>sripathy</code>).</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the task flow <code>Connections - Card</code>.</td>
</tr>
<tr>
<td>fromDate</td>
<td>The start date of a date range within which to show feedback messages</td>
</tr>
<tr>
<td></td>
<td>Enter dates in the format <code>YYYY-MM-DD</code>.</td>
</tr>
<tr>
<td></td>
<td>See also <code>toDate</code>.</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>hideActions</td>
<td>A means of hiding the actions normally associated with a Feedback or Message Board entry, such as Edit, Private, Hide, and Delete</td>
</tr>
<tr>
<td></td>
<td>■ Enter <code>#{true}</code> to hide actions associated with a Feedback or Message Board entry.</td>
</tr>
<tr>
<td></td>
<td>■ Enter <code>#{false}</code> to show such actions. When no value is entered, <code>false</code> is the default.</td>
</tr>
<tr>
<td></td>
<td>This parameter appears in the component properties for the following task flows:</td>
</tr>
<tr>
<td></td>
<td>■ Feedback – Quick View</td>
</tr>
<tr>
<td></td>
<td>■ Message Board – Quick View</td>
</tr>
</tbody>
</table>

Advanced Information for the People Connections Service
### Table 20–10  (Cont.) People Connections Service Task Flow Binding Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hideConfigure</td>
<td>A means of hiding the personalization option on the 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>
<tr>
<td>hideFooter</td>
<td>A means of showing or hiding the task flow footer</td>
</tr>
<tr>
<td></td>
<td>The task flow footer contains the <strong>More</strong> link that appears at the bottom of</td>
</tr>
<tr>
<td></td>
<td>a task flow when there are more items to show than can be accommodated in</td>
</tr>
<tr>
<td></td>
<td>the current view. Hiding the footer hides the <strong>More</strong> link. Hiding the</td>
</tr>
<tr>
<td></td>
<td>footer does not affect <strong>Previous</strong> and <strong>Next</strong> links that may also</td>
</tr>
<tr>
<td></td>
<td>appear at the bottom of a task flow.</td>
</tr>
<tr>
<td></td>
<td>Enter <code>true</code> to hide the footer. Enter <code>false</code> (default) to show the footer.</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the following task flows:</td>
</tr>
<tr>
<td></td>
<td>■ Activity Stream - Quick View</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>hideHeader</td>
<td>A means of showing or hiding the task flow header</td>
</tr>
<tr>
<td></td>
<td>Enter <code>true</code> to hide the task flow header. Enter <code>false</code> (default) to show</td>
</tr>
<tr>
<td></td>
<td>the task flow header.</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></td>
<td>■ Connections - Quick View</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>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 task flow Profile - Snapshot.</td>
</tr>
<tr>
<td>photoUploadAllowed</td>
<td>A means of allowing a photo upload from the profile screen</td>
</tr>
<tr>
<td></td>
<td>Enter <code>true</code> (default) to allow users to upload a photo to their profile.</td>
</tr>
<tr>
<td></td>
<td>Enter <code>false</code> to prevent users from uploading a photo.</td>
</tr>
<tr>
<td></td>
<td>This parameter is associated with the task flow Profile - Snapshot.</td>
</tr>
</tbody>
</table>
### Table 20–10 (Cont.) People Connections Service Task Flow Binding Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>resourceId</td>
<td>The current user ID</td>
</tr>
<tr>
<td></td>
<td>Enter #{securityContext.userName} to return the current user. This parameter is associated with the following task flows:</td>
</tr>
<tr>
<td></td>
<td>■ Activity Stream</td>
</tr>
<tr>
<td></td>
<td>■ Connections</td>
</tr>
<tr>
<td></td>
<td>■ Feedback</td>
</tr>
<tr>
<td></td>
<td>■ Message Board</td>
</tr>
<tr>
<td></td>
<td>■ Profile</td>
</tr>
<tr>
<td></td>
<td>■ Profile Gallery</td>
</tr>
<tr>
<td>showConnectionsActivities</td>
<td>A way to allow or omit connections’ activities in a user’s view of the Activity Stream</td>
</tr>
<tr>
<td></td>
<td>■ Enter #{true} to show the activities of the user’s connections in his or her Activity Stream. If a value is not specified, true is the default.</td>
</tr>
<tr>
<td></td>
<td>■ Enter #{false} to omit showing the current user's activities in his or her Activity Stream.</td>
</tr>
<tr>
<td></td>
<td>This parameter appears in the component properties for the Activity Stream task flow.</td>
</tr>
<tr>
<td>showSummarized</td>
<td>A means of summarizing notifications in an Activity Stream</td>
</tr>
<tr>
<td></td>
<td>■ Enter #{true} to provide a summary view of the actions performed on a given object.</td>
</tr>
<tr>
<td></td>
<td>For example, if there are two activities:</td>
</tr>
<tr>
<td></td>
<td>Monty created the event Event1.</td>
</tr>
<tr>
<td></td>
<td>Monty created the event Event2.</td>
</tr>
<tr>
<td></td>
<td>A value of true summarizes these into one notification in the Activity Stream: Monty created the events Event1 and Event2.</td>
</tr>
<tr>
<td></td>
<td>If more than three activities are summarized, a number is shown in lieu of a listing of individual activity names, for example: Monty created 4 events.</td>
</tr>
<tr>
<td></td>
<td>■ Enter #{false} to list each action on a given object individually. If a value is not specified, false is the default.</td>
</tr>
<tr>
<td></td>
<td>This parameter appears in the component properties for the Activity Stream task flow.</td>
</tr>
<tr>
<td>showUserActivities</td>
<td>A means of including or excluding the current user’s activities from the Activity Stream</td>
</tr>
<tr>
<td></td>
<td>■ Enter #{true} to include the activities of the current user. If no value is specified, true is the default.</td>
</tr>
<tr>
<td></td>
<td>■ Enter #{false} to exclude such activities</td>
</tr>
<tr>
<td></td>
<td>This parameter appears in the component properties for the Activity Stream task flow.</td>
</tr>
</tbody>
</table>
20.3.3 Using the People Connections Service REST APIs

Oracle WebCenter provides REST APIs to support the People Connections service. You can use the People Connections service REST APIs to perform the following actions:

- Browse other users’ application activity in an activity stream
- Browse a profile or connections list
- Update profile status messages
- Create and delete connection lists
- Add and remove members from connection lists
- Post, read, and delete messages
- Read and delete feedback

This section describes the REST APIs associated with the People Connections service. It contains the following subsections:

- Section 20.3.3.1, "Activity Stream REST APIs"
- Section 20.3.3.2, "Connections and Profile REST APIs"
- Section 20.3.3.3, "Feedback REST APIs"
- Section 20.3.3.4, "Message Board REST APIs"

See Also: For an introduction to the REST APIs, see Chapter 28, "Using Oracle WebCenter REST APIs."
20.3.3.1 Activity Stream REST APIs

Use the Activity Stream REST APIs to browse user application activities in an activity stream. This section provides information about the REST APIs to use to perform this action. It contains the following subsections:

- Section 20.3.3.1.1, "Activity Stream Entry Point"
- Section 20.3.3.1.2, "Activity Stream Resource Type Taxonomy"
- Section 20.3.3.1.3, "Activity Stream Security Considerations"
- Section 20.3.3.1.4, "Activity Stream Resource Types"

**Note:** When user activities are retrieved through REST, all of the user's activities are returned. This may not resemble the activities returned through a WebCenter application UI. This is because the options for configuring which activities to return in the WebCenter application UI have no effect on the activities that REST retrieves.

### 20.3.3.1.1 Activity Stream Entry Point

Each REST service has a link element within the Resource Index that provides the entry point for that service. For the People Connections service, each feature has its own link element. For example, to find the entry point for the Activity Stream feature of the People Connections service, find the link elements with a `resourceType` of:

```
urn:oracle:webcenter:activities:stream
```

The corresponding `href` or `template` element provides the URI entry point, which retrieves application activities for the current user from the Activity Stream. The client sends HTTP requests to this entry point to work with the Activity Stream feature of the People Connections service.

**See Also:** For more information about the Resource Index, see Section 28.3.1, "The Resource Index."

**See Also:** For more information about resource types, see Section 28.3.2.1, "Resource Type."

### 20.3.3.1.2 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 20.3.3.1.4, "Activity Stream Resource Types."

The resource type taxonomy for the Activity Stream feature of the People Connections service is:

```
urn:oracle:webcenter:activities:stream
  urn:oracle:webcenter:activities:activity
```

### 20.3.3.1.3 Activity Stream Security Considerations

You must be logged in to the REST service to access any of the People Connections REST APIs. After that, the underlying service handles permission checking and the like.

**See Also:** For general security considerations, see Section 28.6, "Security Considerations for WebCenter REST APIs."
20.3.3.1.4 **Activity Stream Resource Types**  This section provides you with all the information you need to know about each resource type. It includes the following subsections:

- "urn:oracle:webcenter:activities:stream"
- "urn:oracle:webcenter:activities:activity"

**urn:oracle:webcenter:activities:stream**

The *stream* response contains URIs for use in retrieving activities from the Activity Stream.

You can retrieve the activities from a user’s stream or activities from a user’s connections’ streams. To have even greater control over which activities to retrieve, use the activity stream query filter. With the query filter, you can:

- Specify the user to query
- Include the user’s connections’ activities in the results
- Include activities from group spaces and/or personal spaces in the results
- Restrict the results to activities from specific services

The options available to you depend on the path you take to get to the *stream* resource and the *rel* of the link that you use. For example, the activity stream query filter is available only from links with a *rel* attribute of `urn:oracle:webcenter:activities:stream`. If you access the activity stream query filter from the *person* resource, the *personGuid* parameter is prefilled.

**Table 20–11** shows the activities returned depending on the *rel* element of the link.

<table>
<thead>
<tr>
<th><em>rel</em></th>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>urn:oracle:webcenter:activities:stream:person</code></td>
<td>Activities from the user’s stream *(GUID/@self)*¹</td>
</tr>
<tr>
<td><code>urn:oracle:webcenter:activities:stream:connections</code></td>
<td>Activities from the user’s connections’ streams *(GUID/@connections)*¹</td>
</tr>
<tr>
<td><code>urn:oracle:webcenter:activities:stream</code></td>
<td>Activities determined by the activity stream query filter</td>
</tr>
<tr>
<td><code>urn:oracle:webcenter:activities:stream:space</code></td>
<td>Activities from the group space’s activity stream</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")
```
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:

* 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: comma separated list of service IDs. If null or empty, retrieves activities from all services.

* personal—Includes activities from the specified user’s personal space. Valid values: true or false. Default value: false.

* connections—Includes activities from the streams of the specified user’s connections. Valid values: true or false. Default value: false.

* groupSpaces—Includes activities from all group spaces of which the specified user is a member. Valid values: true or false. Default value: false.

For example, the following URI returns activities from the current user’s activity stream, for all services. It returns activities from the user’s personal space and from the streams of the user’s connections:

http://host:port/rest/api/activities?personal=true&connections=true&personGuid=@me&stoken=token

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&stoken=token
response—Body: 0 or more activities

Note: Because the stream resource includes activity items, the response may also return resource links from the activity type.

Resource Types Linked to From stream
Table 20–12 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>
</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.

Read-only Elements for activity
Table 20–13 lists the read-only elements for the activity resource.

See Also: For information about projection, see "Common Request Query Parameters".

<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>createdDate</td>
<td>Date (String)</td>
<td>Date the activity was created</td>
</tr>
</tbody>
</table>
Use the Connections and Profile REST APIs to browse a profile or a connections list, manage connections lists, add or remove connections, send invitations to connect, and update a profile status message.

### Table 20–13 (Cont.) Read-Only Elements for activity

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| activityType     | String      | Activity type
|                  |             | Unique within the service. For example the Discussions service can return the activityType `createForum`.                             |
| serviceId        | String      | Unique ID of the service that created the activity
|                  |             | **Note:** For a list of WebCenter service IDs, see Table 20–3, ”Service IDs”.                                                          |
| detail           | String      | Detail information for the activity
|                  |             | For example, this might return the contents of a message, the file name of a document, and the like.                                    |
|                  |             | Similar to detailURL. Both, either, or neither can be used.                                                                               |
| detailURL        | String      | Detail information for the activity
|                  |             | Similar to detail. Both, either, or neither can be used.                                                                                   |
| groupSpace       | groupSpaceReference | Information about the group space in which the activity was performed
|                  |             | **Note:** This reference is not present for activities that did not happen in a group space (for example, activities that happened in a personal space). Also, creation of a group space happens in a personal space, so that activity does not have this element either. |
| message          | String      | Localized string for this activity
|                  |             | This may contain replacement strings within curly braces ({}).                                                                            |
| permission       | PRIVATE     | Permission level of this activity                                                                                                         |
|                  | SHARED      |                                                                                                                                           |
|                  | PUBLIC      |                                                                                                                                           |
| scope            | String      | Scope of the activity
|                  |             | This might return a group space, a personal space, and the like.                                                                          |
|                  |             | For example, for a group space, it returns a string similar to the following: `s8bba98ff_4cbb_40b8_beee_296c916a23ed`                      |
| templateParams   | Param       | String replacement parameters for users and objects referenced in the message of this activity                                             |

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.
This section provides information about the REST APIs to use to perform these actions. It contains the following subsections:

- **Section 20.3.3.2.1, "Connections and Profile Entry Point"**
- **Section 20.3.3.2.2, "Connections and Profile Resource Type Taxonomy"**
- **Section 20.3.3.2.3, "Connections and Profile Security Considerations"**
- **Section 20.3.3.2.4, "Connections and Profile Resource Types"**

### 20.3.3.2.1 Connections and Profile Entry Point

Each REST service has a link element within the Resource Index that provides the entry point for that service. For the People Connections service, each feature has its own link element. For example, to find the entry points for Connections and Profile features of the People Connections service, find the link elements with a `resourceType` of:

- `urn:oracle:webcenter:people` (returns the current user profile)
- `urn:oracle:webcenter:people:person` (enables you to query for a user)
- `urn:oracle:webcenter:people:invitations` (returns invitations sent or received by the current user)

**Note:** The `people:person` and `people:invitations` resource types have a template but not an `href`.

The corresponding `href` or `template` element provides the URI entry point, which returns a list of people (`people`) or an individual user (`people:person`). The client sends HTTP requests to this entry point to work with the Connections and Profile features of the People Connections service.

**See Also:** For more information about the Resource Index, see Section 28.3.1, "The Resource Index."

For more information about resource types, see Section 28.3.2.1, "Resource Type."

### 20.3.3.2.2 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 20.3.3.2.4, "Connections and Profile Resource Types."

The resource type taxonomy for the Connections and Profile features in the People Connections service is:

- `urn:oracle:webcenter:people`
- `urn:oracle:webcenter:people:person`
- `urn:oracle:webcenter:people:person:list`
- `urn:oracle:webcenter:people:person:person:listNames`
- `urn:oracle:webcenter:people:person:person:listName`
- `urn:oracle:webcenter:people:person:person:list:member`
- `urn:oracle:webcenter:people:invitations`
- `urn:oracle:webcenter:people:invitation`

### 20.3.3.2.3 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 28.6, “Security Considerations for WebCenter REST APIs.”

20.3.3.2.4 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, itemsPerPage, projection

See Also: For information about REST API parameters, such as startIndex and itemsPerPage, see “Common Request Query Parameters”.

  - response—Body: 1 or more person

**Resource Types Linked to from people**
Table 20–14 lists the resource types that the client can link to from the people 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>
</tbody>
</table>
Advanced Information for the People Connections Service

urn:oracle:webcenter:people:person

The person response contains profile data and the URIs for use in retrieving a user profile.

Navigation Paths to person

This section shows how the client can navigate through the hypermedia to access the person resource:

- resourceIndex
  - people

- resourceIndex
  - people
  - person

- resourceIndex
  - person

- author
  - person

- resourceIndex
  - activities:stream
  - person

Supported Methods for person

The following method is supported by the person resource:

- GET

  - request—Parameters: q

    To retrieve a specified person, the format of q is:

    q=[loginid:equals:username]
    Or
    q=[guid:equals:guid]

    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 20–15 lists the read-only elements for the person resource.

Note: The elements present in a person response depend on how the user repository is configured and the elements it supports. Additionally, several of the pieces of data represented in Table 20–15, such as address, emails, photos, phoneNumbers, and organization, can have multiple instances.
See Also: For information about projection, see "Common Request Query Parameters".

## Table 20–15 Read-Only Elements for person

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Projection</th>
</tr>
</thead>
<tbody>
<tr>
<td>guid</td>
<td>String</td>
<td>Unique GUID of the person</td>
<td>Summary, Detailed</td>
</tr>
<tr>
<td>id</td>
<td>String</td>
<td>Unique login ID of the person (that is, the user name, for example, pat_coi)</td>
<td>Summary, Detailed</td>
</tr>
<tr>
<td>displayName</td>
<td>String</td>
<td>Display name of the person (the user's name, for example, Pat Coi). This may have the same value as id, depending on the repository configuration.</td>
<td>Summary, Detailed</td>
</tr>
<tr>
<td>birthday</td>
<td>Date (String)</td>
<td>Birth date of the person</td>
<td>Summary, Detailed</td>
</tr>
<tr>
<td>connected</td>
<td>Boolean (String)</td>
<td>Whether or not this person is connected to the current user</td>
<td>Detailed</td>
</tr>
<tr>
<td>language</td>
<td>String</td>
<td>Preferred language of the person</td>
<td>Summary, Detailed</td>
</tr>
<tr>
<td>timeZone</td>
<td>String</td>
<td>Time zone of the person</td>
<td>Summary, Detailed</td>
</tr>
<tr>
<td>name</td>
<td>name</td>
<td>Name information for the person</td>
<td>Summary, Detailed</td>
</tr>
<tr>
<td>address</td>
<td>address</td>
<td>Address information for the person</td>
<td>Summary, Detailed</td>
</tr>
<tr>
<td>emails</td>
<td>value</td>
<td>Emails for the person</td>
<td>Summary, Detailed</td>
</tr>
<tr>
<td>photos</td>
<td>value</td>
<td>Profile photos for the person</td>
<td>Summary, Detailed</td>
</tr>
<tr>
<td>phoneNumbers</td>
<td>value</td>
<td>Phone numbers for the person</td>
<td>Summary, Detailed</td>
</tr>
</tbody>
</table>
Resource Types Linked to from person

Table 20–16 lists the resource types that the client can link to from the person resource.

<table>
<thead>
<tr>
<th>Table 20–16</th>
<th>Related Resource Types for person</th>
</tr>
</thead>
<tbody>
<tr>
<td>rel</td>
<td>resourceType</td>
</tr>
<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
```
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 listName items
  
  **See Also:** For information about listName items, see “urn:oracle:webcenter:people:person:listName”.

- **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 20–17 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>
</table>


**urn:oracle:webcenter:people:person:listNames**

The listNames response contains URIs for use in retrieving the names of existing connections lists (GET) and to 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

  **Note**: @connections is a user’s default connections. It behaves like a connections list, but it is not included in the results of listNames.

- **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 20–18 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.

Navigation Paths to listName
This section shows how the client can navigate through the hypermedia to access the listName resource:

```text
resourceIndex
  person
    listNames
    listName
```

Writable Elements for listName
Table 20–19 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 20–20 lists the resource types that the client can link to from the listName resource.
The member response contains URIs for use in deleting a connection from a connections list or removing a connection from the default connections list.

**Navigation Paths to member**

This section shows how the client can navigate through the hypermedia to access the member resource:

```
resourceIndex
  person
    list
      member
```

**Supported Methods for member**

The following method is supported by the member resource:

- DELETE
  - request

**Writable Elements for member**

Table 20–21 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:list:member**

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 20–22 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 20–23 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.

```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:
To retrieve invitations sent from the current user, the format of \( q \) is:

\[
q=\text{[invitor:equals:@me]}
\]

- **Response—Body:** 1 or more invitations

<table>
<thead>
<tr>
<th>POST</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Request—Body:</strong> invitation</td>
</tr>
<tr>
<td><strong>Response—Body:</strong> invitation</td>
</tr>
</tbody>
</table>

**Note:** The methods supported depend on the path taken to get to the resource. For example, the POST method is supported only by the link from the person resource. Check the capabilities of the link to understand exactly which methods are supported.

## Writable Elements for invitations

Table 20–24 lists the writable elements for the **invitations** resource.

### Table 20–24 Writable Elements for invitations

<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 20–25 lists the resource types that the client can link to from the **invitations** resource:

### Table 20–25 Related Resource Types for invitations

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>urn:oracle:webcenter:people:invitation</td>
<td></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
Advanced Information for the People Connections Service

- Response—Body: invitation

- DELETE

- Request

Writable Elements for invitation
Table 20–26 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, IGNORED</td>
<td>The status of the invitation. 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 20–27 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>
<tr>
<td>sentDate</td>
<td>Date (String)(^1)</td>
<td>Date the invitation was sent</td>
</tr>
</tbody>
</table>

\(^1\) Data types, such as DATE and BOOLEAN, are stored in the API as STRING.

Resource Types Linked to from invitation
Table 20–28 lists the resource types that the client can link to from the invitation response.

<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>

20.3.3.3 Feedback REST APIs
Use the Feedback REST APIs to read and delete feedback. This section provides information about the REST APIs to use to perform these actions. It contains the following subsections:

- Section 20.3.3.3.1, "Feedback Entry Point"
- Section 20.3.3.3.2, "Feedback Resource Type Taxonomy"
- Section 20.3.3.3.3, "Feedback Security Considerations"
- Section 20.3.3.3.4, "Feedback Resource Types"

20.3.3.3.1 Feedback Entry Point
Each REST service has a link element within the Resource Index that provides the entry point for that service. For the People Connections service, each feature has its own link element. To find the entry points for
the Feedback feature of the People Connections service, find the link elements with a
resourceType of:

urn:oracle:webcenter:feedback

The corresponding href or template element provides the URI entry point, which
returns all received feedback for the current user. The client sends HTTP requests to
this entry point to work with the Feedback feature of the People Connections service.

See Also: For more information about the Resource Index, see
Section 28.3.1, "The Resource Index."
For more information about resource types, see Section 28.3.2.1,
"Resource Type."

20.3.3.3.2 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 20.3.3.4, "Feedback Resource Types."

the resource type taxonomy for the Feedback feature of the People Connections service
is:

urn:oracle:webcenter:feedback
    urn:oracle:webcenter:feedback:message

20.3.3.3 Feedback Security Considerations You must be logged in to the REST service to
access any of the People Connections REST APIs. After that, the underlying service
handles permission checking and the like.

See Also: For general security considerations, see Section 28.6,
"Security Considerations for WebCenter REST APIs."

20.3.3.4 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
Advanced Information for the People Connections Service

– 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".

Resource Types Linked to from feedback
Table 20–29 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</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></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

Note: Currently, the DELETE method for the urn:oracle:webcenter:feedback:message resource type is nonfunctional. The message is not deleted, even though the REST client receives an HTTP success status of 204.

Read-only Elements for message
Table 20–30 lists the read-only elements for the message resource.
Table 20–30 Read-only Elements for message

<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>
</tbody>
</table>

1 Data types, such as DATE and BOOLEAN, are stored in the API as STRING.

Resource Types Linked to from feedback

Table 20–29 lists the resource types that the client can link to from the feedback resource.

Table 20–31 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:feedback:message</td>
</tr>
</tbody>
</table>

20.3.3.4 Message Board REST APIs

Use the Message Board REST APIs to post, read, and delete messages to a user’s or a group space’s message board.

This section provides information about the REST APIs to use to perform these actions. It contains the following subsections:

- Section 20.3.3.4.1, "Message Board Entry Point"
- Section 20.3.3.4.2, "Message Board Resource Type Taxonomy"
- Section 20.3.3.4.3, "Message Board Security Considerations"
- Section 20.3.3.4.4, "Message Board Resource Types"

20.3.3.4.1 Message Board Entry Point

Each REST service has a link element within the Resource Index that provides the entry point for that service. For the People Connections service, each feature has its own link element. To find the entry points for the Message Board feature of the People Connections service, find the link elements with a resourceType of:

urn:oracle:webcenter:messageBoard

Note: As well as an entry point from the Resource Index, to navigate to an individual user’s message board, the Message Board feature also has an entry point from a group space response for the group space’s message board.

The corresponding href or template element provides the URI entry point, which returns the Message Board for the current user. The client sends HTTP requests to this entry point to work with the Message Board feature of the People Connections service.

See Also: For more information about the Resource Index, see Section 28.3.1, "The Resource Index."

For more information about resource types, see Section 28.3.2.1, "Resource Type."
20.3.3.4.2 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 20.3.3.4.4, "Message Board Resource Types."

The resource type taxonomy for the Message Board feature of the People Connections service is:

```
urn:oracle:webcenter:messageBoard
urn:oracle:webcenter:messageBoard:message
```

20.3.3.4.3 Message Board Security Considerations You must be logged in to the REST service to access any of the People Connections REST APIs. After that, the underlying service handles permission checking and the like.

**See Also:** For general security considerations, see Section 28.6, "Security Considerations for WebCenter REST APIs."

20.3.3.4.4 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) group space and individual user Message Board messages.

**Navigation Paths to messageBoard**
This section shows how the client can navigate through the hypermedia to access the messageBoard resource:

```
resourceIndex
  messageBoard

resourceIndex
  person
    messageBoard

resourceIndex
  spaces
    space
      messageBoard
```

**Supported Methods for messageBoard**
The following methods are supported by the messageBoard resource:

- GET
  - request—Parameters: `startIndex`, `itemsPerPage`
    **See Also:** For information about REST API parameters, such as `startIndex` and `itemsPerPage`, see "Common Request Query Parameters".
  - response—Body: message
**POST**
- request—Body: message

**See Also:** For information about message, see "urn:oracle:webcenter:messageBoard:message".

### Resource Types Linked to from messageBoard

Table 20–32 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 group space or individual user Message Board message.

### Navigation Paths to message

This section shows how the client can navigate through the hypermedia to access the message resource:

- resourceIndex
  - messageBoard
    - message
- resourceIndex
  - person
    - messageBoard
    - message
- resourceIndex
  - spaces
    - space
      - messageBoard
      - message

### Supported Methods for message

The following methods are supported by the message resource:

- **GET**
  - request
  - response—Body: message

**Note:** The REST GET command for reading (retrieving) messages retrieves all shown messages. Messages that are hidden through the application user interface are not retrieved. For information about hiding and showing messages, see "Working with Message Board Task Flows," in the Oracle Fusion Middleware User’s Guide for Oracle WebCenter.
Troubleshooting the People Connections Service

### Integrating the People Connections Service

20-57

- **PUT**
  - request—**Body**: message
  - response—**Body**: message
- **DELETE**
  - request

**Writable Elements for message**

Table 20–33 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 20–34 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><code>personReference</code></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>
</tbody>
</table>

1 Data types, such as `DATE` and `BOOLEAN`, are stored in the API as `STRING`.

**Resource Types Linked to from message**

Table 20–35 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><code>urn:oracle:webcenter:messageBoard:message</code></td>
</tr>
</tbody>
</table>

20.4 Troubleshooting the People Connections Service

This section provides information to assist you in troubleshooting problems you may encounter while using the People Connections service.

**People Connections task flows take up the full browser, even when empty**

This problem may arise if you are using Internet Explorer, version 8 (IE8). In IE8, go to **Tools, Compatibility View Settings**. In the Compatibility View Settings dialog, clear the following check boxes:

- Display intranet sites in Compatibility View
- Display all websites in Compatibility View
New Profile Image is Not Shown
Changes to a Profile image may not be rendered immediately due to a stale WebCenter Database cache. Clicking the Refresh icon under the stale image will cause the image to update.
This chapter explains how to integrate wiki and blog functionality into a custom WebCenter application.

This chapter includes the following sections:

- Section 21.1, "Introduction to the Wiki and Blog Services"
- Section 21.2, "Basic Configuration for the Wiki and Blog Services"
- Section 21.3, "Advanced Information for the Wiki and Blog Services"

21.1 Introduction to the Wiki and Blog Services

A wiki is a collection of useful content or information that users can browse and can update and remove, sometimes without the need for registration. This ease of interaction and the variety of operations makes wiki an effective tool for collaborative authoring, where multiple people create written content together using the wiki markup language.

Blogs provide a useful tool for discussing and/or evangelizing any type of idea, strategy, or point of view. Blogs may be projected out to a select group of people or to a wider audience. Typically, blogs invite readers to comment on the overall concepts.

You integrate wikis and blogs into your applications by using the Wiki and Blog services. This section contains the following subsections:

- Section 21.1.1, "Understanding the Wiki and Blog Services"
- Section 21.1.2, "Requirements for the Wiki and Blog Services"
- Section 21.1.3, "What Happens at Runtime"

21.1.1 Understanding the Wiki and Blog Services

You can expose the wiki and blog functionality into your existing portal, application, or web site. The Wiki service enables you to incorporate wikis, and the Blog service enables you to expose blogs on your application pages. There is no limitation on the number of wiki pages or blogs you expose in an application.

21.1.2 Requirements for the Wiki and Blog Services

The Wiki and Blog services rely on Oracle WebCenter Wiki and Blog Server. For information about installing Oracle WebCenter Wiki and Blog Server, see the Oracle Fusion Middleware Installation Guide for Oracle WebCenter. For information about
Basic Configuration for the Wiki and Blog Services

configuring the server, see the "Managing the Wiki and Blog Services" chapter in the Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter.

21.1.3 What Happens at Runtime

When you expose wikis and blogs into your custom WebCenter application, then at runtime you can add, edit, print, and delete wiki pages and blog entries. You can also add and remove blog comments and define the blog authors who can manage a blog. To work with wiki content at runtime, you use wiki HTML Editor controls or the wiki markup language.

The wiki and blog server functionality exposed at runtime in your custom WebCenter application depends on the value that you specify for the inline query string parameter while integrating a wiki or blog into your application. For example, if inline=0, the default user interface of the wiki and blog server is displayed. It includes a toolbar of useful links across the top of the page, a search feature, and additional navigation under the General heading. Figure 21–1 depicts a wiki page rendered with inline=0 in a custom WebCenter application.

For information about the Wiki and Blog services at runtime, see the "Working with Wikis and Blogs" chapter in the Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

Figure 21–1  Wiki Page Rendered in Inline=0 Mode

21.2 Basic Configuration for the Wiki and Blog Services

This section describes required steps for adding wikis and blogs to your application or portal. It contains the following subsections:

- Section 21.2.1, "Adding Wikis or Blogs to Your Application or Portal"
- Section 21.2.2, "Oracle WebCenter Wiki and Blog Server URL Endpoints and Query String Parameters"
- Section 21.2.3, "Oracle WebCenter Wiki and Blog Server Web Services Interface"
21.2.1 Adding Wikis or Blogs to Your Application or Portal

You can expose the wiki and blog functionality into your custom WebCenter applications by using any of the following integration methods:

- Through the Web Clipping portlet or any portlet capable of consuming a URL
- Through an IFRAME, which you include in a page in your custom WebCenter application
- Through the use of a custom-built user interface that you can create using the provided web services

Sample portlets that demonstrate some of these integration methods are located on the Oracle Technology Network (OTN). You can download the sample portlets from the Oracle WebCenter Suite 11g Demonstrations and Samples page on the OTN at the following URL:


From this link, you can learn more about the sample portlets, as well as how to download and install them.

---

**Note:** You can use Oracle Access Manager-based single sign-on, Oracle Single Sign-On (OSSO), and SAML-based security to secure Oracle WebCenter Wiki and Blog Server. For information, see the "Configuring a WebCenter Application to Use Single Sign-On" section in the *Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter*.

The users you set up for your custom WebCenter applications must match the users defined on the Oracle WebCenter Wiki and Blog Server. After a user is authenticated, if the user does not exist within the Oracle WebCenter Wiki and Blog Server, the user is created on the server and a default role is assigned to the user.

---

This section contains the following subsections:

- Section 21.2.1.1, "Adding Wikis or Blogs by Using a Portlet"
- Section 21.2.1.2, "Adding Wikis or Blogs by Using an IFRAME"
- Section 21.2.1.3, "Adding Wikis or Blogs by Using Web Services"

21.2.1.1 Adding Wikis or Blogs by Using a Portlet

You can bring a wiki into your application through a Web Clipping portlet or any portlet capable of consuming a URL. For examples of portlets capable of consuming a URL, see the Oracle WebCenter Suite 11g Demonstrations and Samples page on OTN here:


Oracle WebCenter Framework includes a preconfigured Web Clipping portlet producer. You can register the Web Clipping portlet producer with your custom WebCenter application and add a Web Clipping portlet. For information about adding a Web Clipping portlet, see Chapter 36, "Creating Content-Based Portlets with Web Clipping." Ensure that your application is secured and the user credentials defined in
your application match with the user credentials on Oracle WebCenter Wiki and Blog Server.

After adding a Web Clipping portlet, when you run your custom WebCenter application to your browser, you can use the Web Clipping Studio to consume the desired URLs into your application. For information about the URL formats to use for integrating wikis and blogs into your application, see Section 21.2.2, "Oracle WebCenter Wiki and Blog Server URL Endpoints and Query String Parameters."

Figure 21–2 shows a sample wiki portlet integrated into an application.

### Figure 21–2 Sample Wiki Portlet in an Application

![Sample Wiki Portlet in an Application](image)

#### 21.2.1.2 Adding Wikis or Blogs by Using an IFRAME

You can add wikis and blogs to a page in your application with an IFRAME. Simply reference the desired URL from within an IFRAME, as shown in Example 21–1 and Figure 21–3. For information about URL formats to use, see Section 21.2.2, "Oracle WebCenter Wiki and Blog Server URL Endpoints and Query String Parameters."

**Example 21–1 Referencing a Wiki URL from an IFRAME**

```html
<iframe src="http://server:port/owc_wiki/page/show.jz?inline=1&scope=domain" width="100%"></iframe>
```

Where:

* `server:port` refer to the server name and port number of the system where your Oracle WebCenter Wiki and Blog Server is deployed.

* `domain` refers to the name of the wiki domain that you want to integrate into your application.

Figure 21–3 shows an example of how you can use an IFRAME to integrate wiki. This example shows this being done using a portlet.
21.2.1.3 Adding Wikis or Blogs by Using Web Services

Oracle WebCenter Wiki and Blog Server provides web services that enable interaction between your application and the wiki server. You can add wikis and blogs to your applications by calling web services along with a custom-built user interface. This requires you to call the web services directly and create the user interface for interactions with the Oracle WebCenter Wiki and Blog Server. Alternatively, you can just redirect to the prebuilt wiki and blog pages.

For information about how to use web services, see Section 21.2.3, "Oracle WebCenter Wiki and Blog Server Web Services Interface."

21.2.2 Oracle WebCenter Wiki and Blog Server URL Endpoints and Query String Parameters

While integrating a wiki or blog into your application, when you reach the point where you enter the wiki or blog source in the URL-consuming portlet or in the IFRAME, use the formats provided in Table 21–1.

On one of your application pages if you want to add a hyperlink that references a specific wiki page or blog, then use the URL formats provided in Table 21–2.

You can use wiki and blog query string parameters to define context (within the application context or external to it) and look and feel (page background colors and fonts). Query string parameters are bits of information you add to a URL to refine the behavior of the URL target. Table 21–3 lists and describes query string parameters you can use in custom WebCenter application wiki and blog URLs.

For information about using these URLs and query string parameters in your custom WebCenter application, see Section 21.2.1, "Adding Wikis or Blogs to Your Application or Portal."
Table 21–1 URL Formats for Exposing Wikis and Blogs in Applications

<table>
<thead>
<tr>
<th>Type</th>
<th>URL Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>General Blog</td>
<td><a href="http://server:port/owc_wiki/blog/list.jz?inline=1&amp;name=domain">http://server:port/owc_wiki/blog/list.jz?inline=1&amp;name=domain</a></td>
</tr>
<tr>
<td>Personal Blog</td>
<td><a href="http://server:port/owc_wiki/blog/list.jz?inline=1&amp;name=user">http://server:port/owc_wiki/blog/list.jz?inline=1&amp;name=user</a></td>
</tr>
</tbody>
</table>

In Table 21–1, server and port refer to the server and port number where Oracle WebCenter Wiki and Blog Server is installed, and owc_wiki refers to your Oracle WebCenter Wiki and Blog Server deployed application.

Table 21–2 Formats for Hyperlinks to Wikis and Blogs

<table>
<thead>
<tr>
<th>Target</th>
<th>URL Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>General Blog</td>
<td><a href="http://server:port/owc_wiki/blog/list.jz?inline=1&amp;name=domain">http://server:port/owc_wiki/blog/list.jz?inline=1&amp;name=domain</a></td>
</tr>
<tr>
<td>Personal Blog</td>
<td><a href="http://server:port/owc_wiki/blog/list.jz?inline=1&amp;name=user">http://server:port/owc_wiki/blog/list.jz?inline=1&amp;name=user</a></td>
</tr>
</tbody>
</table>

Table 21–3 Query String Parameters Available for Use in Wiki and Blog URLs

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| inline    | ■ Value of 0 sets the view to normal, displaying the default user interface and features of Oracle WebCenter Wiki and Blog Server. This is the recommended mode for wiki administrators.  
■ Value of 1 strips away nonessential wiki and blog chrome. It renders left-side navigation that lists all wiki pages within the current domain. This is the recommended mode when integrating with a custom WebCenter application or portal.  
■ Value of 2 is similar to inline=1, except it turns off left-side navigation.  
For information about how a wiki page looks on Oracle WebCenter Wiki and Blog Server when rendered in different inline modes, see the “Working with Wikis and Blogs” chapter in the Oracle Fusion Middleware User’s Guide for Oracle WebCenter. |
| name      | Facilitates navigation to a specific blog, attributed either to a particular domain or user. |
| page      | Facilitates navigation to a specific page in a specified domain. This variable follows the syntax: page=domain:wikiPageName. In general, you should include both the domain and the page name to reduce the potential for any ambiguity. |
All of the URL formats specified in Table 21–1 and Table 21–2 can take query string parameters to focus the result provided on the target page. Parameters, such as inline and theme, are session-level variables. Once the URL passes session-level variables to the target, the variable values continue to apply, even if you leave and return to the original target page. Also, session-level variables are applied even when the parameter is not included in subsequent URLs.
Basic Configuration for the Wiki and Blog Services

21.2.3 Oracle WebCenter Wiki and Blog Server Web Services Interface

After Oracle WebCenter Wiki and Blog Server is installed, you can access the Oracle WebCenter Wiki and Blog Server Web Services end point with the following URL:

http://host:port/owc_wiki/services/WikiRemoteService

Where, the host and port refer to the host name and port number of the system where Oracle WebCenter Wiki and Blog Server is installed.

Oracle WebCenter Wiki and Blog Server Web Services provide access to the content in wiki pages and domains. Web Services also enable the creation, modification, and removal of wiki pages and domains. You can use Oracle JDeveloper (or other tools) to create a proxy for the Web Services from the WSDL definition, located here:


This section describes the Oracle WebCenter Wiki and Blog Server Web Services interface. It contains the following subsections:

- Section 21.2.3.1, "Definition of the Interface"
- Section 21.2.3.2, "Oracle WebCenter Wiki and Blog Server Web Services Security"
- Section 21.2.3.3, "Generating the Web Services Client Stub"
- Section 21.2.3.4, "Sample Code for Accessing the Web Services Interface"

21.2.3.1 Definition of the Interface

Some of the methods return information in JavaBeans. Table 21–4 lists the attributes of the DomainInfo, PageInfo, BlogInfo, BlogEntryInfo, and SearchResult beans. You can also use the getter methods of the described attributes, for example, long getCreated().

Table 21–4 Data Structures of Oracle WebCenter Wiki and Blog Server Web Services

<table>
<thead>
<tr>
<th>DomainInfo Bean</th>
<th>PageInfo Bean</th>
<th>BlogInfo Bean</th>
<th>BlogEntryInfo Bean</th>
<th>SearchResult Bean</th>
</tr>
</thead>
<tbody>
<tr>
<td>String domain;</td>
<td>String domain;</td>
<td>int entryCount;</td>
<td>String author;</td>
<td>String author;</td>
</tr>
<tr>
<td>String description;</td>
<td>String name;</td>
<td>long id;</td>
<td>String blogName;</td>
<td>String domain;</td>
</tr>
<tr>
<td>String author;</td>
<td>int revision;</td>
<td>String name;</td>
<td>long blogid;</td>
<td>String fragment;</td>
</tr>
<tr>
<td>long created;</td>
<td>int views;</td>
<td>String parent;</td>
<td>int commentCount;</td>
<td>String lastUpdate;</td>
</tr>
<tr>
<td>String startPage;</td>
<td>String author;</td>
<td>BlogType type;</td>
<td>String content;</td>
<td>String oid;</td>
</tr>
<tr>
<td>long created;</td>
<td>String editor;</td>
<td>String created;</td>
<td>String pageName;</td>
<td>String tags;</td>
</tr>
<tr>
<td>String editURL;</td>
<td>long modified;</td>
<td>String id;</td>
<td>String type;</td>
<td>String url;</td>
</tr>
<tr>
<td>String viewURL;</td>
<td>String viewURL;</td>
<td>String title;</td>
<td>String url;</td>
<td>String viewURL;</td>
</tr>
</tbody>
</table>

21-8 Oracle Fusion Middleware Developer's Guide for Oracle WebCenter
Basic Configuration for the Wiki and Blog Services

Web Services methods include methods for accessing and performing actions on domains, wiki pages, blogs, and blog entries, as described in the following tables.

**Table 21-5 Domain-Related Methods**

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DomainInfo[]</td>
<td>getAllDomainInfo(int maxResult, int offset, String key)</td>
<td>Returns a list of all domains.</td>
</tr>
<tr>
<td>DomainInfo</td>
<td>getDomainInfo(String domainName, String key)</td>
<td>Returns information about the specified domain.</td>
</tr>
<tr>
<td>void</td>
<td>createDomain(String domainName, String description, String startPage, String key)</td>
<td>Creates a domain with the specified attributes.</td>
</tr>
<tr>
<td>void</td>
<td>deleteDomain(String domainName, String key)</td>
<td>Deletes the specified domain.</td>
</tr>
<tr>
<td>PageInfo[]</td>
<td>getAllPageInfo(String domainName, int maxResult, int offset, String key)</td>
<td>Returns a list of all pages within a domain.</td>
</tr>
<tr>
<td>editDomainInfo</td>
<td>editDomainInfo(String domainName, String description, String startPage, String key)</td>
<td>Modifies the specified domain with the specified attributes.</td>
</tr>
</tbody>
</table>

**Table 21-6 Page-Related Methods**

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PageInfo</td>
<td>getPageInfo(String domainName, String pageName, String key)</td>
<td>Returns information about the specified wiki page.</td>
</tr>
<tr>
<td>void</td>
<td>createPage(String domainName, String pageName, PageEditMode mode, PageType type, String key)</td>
<td>Creates a wiki page with the specified attributes.</td>
</tr>
<tr>
<td>void</td>
<td>deletePage(String domainName, String pageName, String key)</td>
<td>Deletes the specified wiki page.</td>
</tr>
<tr>
<td>String</td>
<td>getPlainPage(String domainName, String pageName, String key)</td>
<td>Returns the page content in the wiki markup format.</td>
</tr>
<tr>
<td>String</td>
<td>getRenderedPage(String domainName, String pageName, String key)</td>
<td>Returns the content rendered to HTML.</td>
</tr>
</tbody>
</table>

String wcURL;
void savePage (String domainName, String pageName, String content, String key)

Updates the specified wiki page with the passed text content. The String content parameter takes text in the format that you specified in the type parameter while creating the wiki page.

PageInfo getAllPageInfoBetweenDates (String startDate, String endDate, int maxResults, int offset, String key)

Returns details about all pages created or updated within the specified date range. The default date format is MM/dd/yyyy HH:mm.

String getTotalPageCount(String key)

Returns the total number of wiki pages across all wiki domains.

String getTotalPageCountByDate(String startDate, String endDate, String key)

Returns the total number of pages created or updated within a given date range. The default date format is MM/dd/yyyy HH:mm.

1 The “restricted to members of the domain” option is displayed only if the wiki administrator has configured a list of domain members. The domain membership is otherwise open to all wiki users.

2 The “only admins are allowed to edit” option displays only if the currently authenticated user is a wiki administrator.

Table 21–7 Administrative Methods

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String key</td>
<td>login (String username, String passcode)</td>
<td>Logs in a user and obtains the key that is used on all subsequent web service methods. The web service login expires after 30 minutes.</td>
</tr>
<tr>
<td>null</td>
<td>logout (String key)</td>
<td>Logs out the current user as defined by the passed key.</td>
</tr>
<tr>
<td>null</td>
<td>addRoleToUser (String roleName, String user, String key)</td>
<td>Assigns a role to the specified user.</td>
</tr>
<tr>
<td>null</td>
<td>deleteRoleFromUser (String roleName, String user, String key)</td>
<td>Removes the specified role from a user.</td>
</tr>
<tr>
<td>null</td>
<td>createRole (String roleName, String key)</td>
<td>Creates a new role.</td>
</tr>
<tr>
<td>null</td>
<td>deleteRole (String roleName, String key)</td>
<td>Deletes a role.</td>
</tr>
<tr>
<td>null</td>
<td>addPermissionToRole (String roleName, String permID, String key)</td>
<td>Adds the specified permission to a role.</td>
</tr>
<tr>
<td>null</td>
<td>removePermissionFromRole (String roleName, String permID, String key)</td>
<td>Removes the specified permission from a role.</td>
</tr>
<tr>
<td>null</td>
<td>addUserAsDomainMember(String domainName, String user, String key)</td>
<td>Adds a user as a member of the specified domain.</td>
</tr>
</tbody>
</table>
### Table 21–7 (Cont.) Administrative Methods

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>null</td>
<td>deleteUserAsDomainMember(String domainName, String user, String key)</td>
<td>Removes a user from domain membership.</td>
</tr>
<tr>
<td>null</td>
<td>addRoleAsDomainMember(String domainName, String role, String key)</td>
<td>Adds a specified role as the member of a domain.</td>
</tr>
<tr>
<td>null</td>
<td>deleteRoleAsDomainMember(String domainName, String role, String key)</td>
<td>Removes a specified role from a domain membership.</td>
</tr>
</tbody>
</table>

### Table 21–8 Blog-Related Methods

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean</td>
<td>enablePersonalBlog(String description, boolean enable, String key)</td>
<td>Creates a personal blog for the user associated with the passed key (if it does not exist already). If the personal blog already exists, then this method does nothing.</td>
</tr>
<tr>
<td>String[]</td>
<td>getBlogs(BlogType type, String keyword, int maxResults, int offset, String key)</td>
<td>Returns a list of all blogs on the server (if BlogType is set to all), all personal blogs (if BlogType is set to user), or all blogs associated with domains (if BlogType is set to domain).</td>
</tr>
<tr>
<td>BlogInfo</td>
<td>getBlog(String blogName, String key)</td>
<td>Returns the following information about the requested blog: entry count (number of blog entries), ID of the blog, blog name, the parent (the user's name or the domain name), and the type (user or domain blog).</td>
</tr>
</tbody>
</table>

### Table 21–9 Blog Entry-Related Methods

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String[]</td>
<td>getAllBlogEntries(String blogName, String key)</td>
<td>Returns a list of blog entries within the specified blog.</td>
</tr>
<tr>
<td>null</td>
<td>createBlogEntry(String blogName, String title, String content, String key)</td>
<td>Creates a blog entry in the specified blog with the specified title and content. Only a blog author, a wiki administrator, or users identified as additional blog authors can execute this method. If the blog is a domain blog, then only a wiki administrator can perform this task. For information about additional blog authors, see the “Adding and Removing Additional Blog Authors” in the Oracle Fusion Middleware User’s Guide for Oracle WebCenter.</td>
</tr>
<tr>
<td>null</td>
<td>editBlogEntry(String blogEntryID, String title, String content, String key)</td>
<td>Modifies a specified blog entry with the specified title and content. Only a blog author, a wiki administrator, or users identified as additional blog authors can execute this method. If the blog is a domain blog, then only a wiki administrator can perform this task.</td>
</tr>
</tbody>
</table>
null deleteBlogEntry (String blogEntryID, String key) Deletes the specified blog entry. Only a blog author, a wiki administrator, or users identified as additional blog authors can execute this method. If the blog is a domain blog, then only a wiki administrator can perform this task.

String getBlogEntry (String blogEntryID, String key) Returns the content for the specified blog entry.

BlogEntryInfo[] getBlogEntryObject (String blogEntryID, String key) Returns a blog entry object that includes meta data about the blog entry as well as the actual content of the entry.

BlogEntryInfo[] getAllBlogEntryObjects (String blogName, String key) Returns all blog entry objects within the specified blog.

BlogEntryInfo[] getAllBlogEntryObjectsByBlogAndDate (String blogName, String startDate, String endDate, String key) Provides a list of blog entry objects (meta data and content) within the specified blog for entries created or modified between the provided dates.

BlogEntryInfo getAllBlogEntryObjectsByDate (String startDate, String endDate, String key) Returns all blog entry objects created or modified between the specified dates.

String[] getAllBlogEntriesBetweenDates (String startDate, String endDate, String key) Returns a list of blog entry IDs for entries created or modified in the specified date range.

String[] getAllBlogEntriesByBlogAndDate (String blogName, String startDate, String endDate, String key) Provides a list of blog entry IDs within the specified blog for entries created or modified between the provided dates.

String getTotalBlogEntryCount (String key) Returns the total number of blog entries.

String getTotalBlogEntryCountByDate (String startDate, String endDate, String key) Returns the total number of blog entries created or modified between the provided dates.

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>null</td>
<td>deleteBlogEntry (String blogEntryID, String key)</td>
<td>Deletes the specified blog entry. Only a blog author, a wiki administrator, or users identified as additional blog authors can execute this method. If the blog is a domain blog, then only a wiki administrator can perform this task.</td>
</tr>
<tr>
<td>String</td>
<td>getBlogEntry (String blogEntryID, String key)</td>
<td>Returns the content for the specified blog entry.</td>
</tr>
<tr>
<td>BlogEntryInfo[]</td>
<td>getBlogEntryObject (String blogEntryID, String key)</td>
<td>Returns a blog entry object that includes meta data about the blog entry as well as the actual content of the entry.</td>
</tr>
<tr>
<td>BlogEntryInfo[]</td>
<td>getAllBlogEntryObjects (String blogName, String key)</td>
<td>Returns all blog entry objects within the specified blog.</td>
</tr>
<tr>
<td>BlogEntryInfo[]</td>
<td>getAllBlogEntryObjectsByBlogAndDate (String blogName, String startDate, String endDate, String key)</td>
<td>Provides a list of blog entry objects (meta data and content) within the specified blog for entries created or modified between the provided dates.</td>
</tr>
<tr>
<td>BlogEntryInfo</td>
<td>getAllBlogEntryObjectsByDate (String startDate, String endDate, String key)</td>
<td>Returns all blog entry objects created or modified between the specified dates.</td>
</tr>
<tr>
<td>String[]</td>
<td>getAllBlogEntriesBetweenDates (String startDate, String endDate, String key)</td>
<td>Returns a list of blog entry IDs for entries created or modified in the specified date range.</td>
</tr>
<tr>
<td>String[]</td>
<td>getAllBlogEntriesByBlogAndDate (String blogName, String startDate, String endDate, String key)</td>
<td>Provides a list of blog entry IDs within the specified blog for entries created or modified between the provided dates.</td>
</tr>
<tr>
<td>String</td>
<td>getTotalBlogEntryCount (String key)</td>
<td>Returns the total number of blog entries.</td>
</tr>
<tr>
<td>String</td>
<td>getTotalBlogEntryCountByDate (String startDate, String endDate, String key)</td>
<td>Returns the total number of blog entries created or modified between the provided dates.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String[]</td>
<td>getAllBlogEntryComments (String blogEntryID, String key)</td>
<td>Returns all comments on a blog entry.</td>
</tr>
<tr>
<td>void</td>
<td>createBlogEntryComment (String blogEntryID, String text, String key)</td>
<td>Creates a comment on the blog entry.</td>
</tr>
</tbody>
</table>
The APIs `getAllDomainInfo()`, `getAllPageInfo()`, `search()`, and `getListOfBlogs()` support pagination. These APIs support the notion of "block fetch," where clients can specify the maximum number and block of results that should be returned.

The `getAllDomainInfo()`, `getAllPageInfo()`, and `getListOfBlogs()` APIs support two parameters, `maxResults` and `offset`, from which the desired (maximum) number of results are returned. However, the calling client must maintain the state of `offset` or the cursor. Clients should choose and provide a fixed value for `maxResults` in repeated calls to the API to get the correct result.

For example, setting `maxResults=10`, and `offset=1`, the first call returns the first 10 rows (if present). Subsequent calls increment the offset parameter (being maintained at the client side) with `maxResults` fixed at 10. The subsequent calls return rows from 11-20, 21-30, and so on until it returns all rows.

A value of `<=0` for `maxResults` returns the full result set in one call, without pagination. Also, `offset <=0` returns the full result set in one call (without pagination), starting from the first row.

The `search()` method supports the `maxResults` parameter, which you can use to restrict the number of rows returned.

### 21.2.3.2 Oracle WebCenter Wiki and Blog Server Web Services Security

All Oracle WebCenter Wiki and Blog Server Web Services methods are protected to prevent unauthorized access. Every method contains a String key parameter to ensure authorized access. This key is generated as a function of a user’s name and a preconfigured passcode. The passcode is an arbitrary string that an administrator creates in the credential store of the domain in which Oracle WebCenter Wiki and Blog Server has been deployed. As a developer, you need to obtain this shared passcode to use the Web Services interface to access Oracle WebCenter Wiki and Blog Server. For information about the passcode, see the "Generating the Passcode" section in the Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter.

To create the key, use the following method:

```java
String key = client.login(username, passcode);
```

Where:

- `username` refers to the name of the user on whose behalf Web Services is making the call (for example, the user logged into your custom WebCenter application who is accessing the wiki by means of a portlet).
- `passcode` refers to the shared password credential created by the administrator.

---

**Table 21–11 Search-Related Methods**

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SearchResult</td>
<td>SearchResult[] search(String searchText, SearchType type, int maxResults, String key)</td>
<td>Returns URLs to wikis and blog objects that contain the keywords specified in <code>searchText</code>. You can constrain the size of the results, if desired. Valid values for <code>SearchType</code> are all, wiki, or blog, which specifies the type of content to search.</td>
</tr>
</tbody>
</table>
21.2.3.3 Generating the Web Services Client Stub

To use the Oracle WebCenter Wiki and Blog Server Web Services client proxy, you must generate the Web Services client stub on Oracle WebLogic Server. After the client stubs have been created, you can use them in a custom WebCenter application.

To generate the Web Services client stub by using Oracle JDeveloper:

1. In Oracle JDeveloper, create a generic application and add the Web Services project technology.

2. From the File menu, select New.

3. In the New Gallery dialog, under Business Tier, select Web Services, and then in the panel on the right, select Web Service Proxy.

4. In the Create Web Service Proxy wizard, on the Create Web Service Proxy page, click Next.

5. On the Select Web Service Description page, in the WSDL Document URL field, specify the WSDL for Web Services. For Oracle WebCenter Wiki and Blog Server, enter the WSDL in the following format:

```
```

Where server and port refer to the server name and port number of the system where Oracle WebCenter Wiki and Blog server is installed.

6. Click Next.

7. On the Select Web Service Description page, in the Package Name field, specify the Java package and click Next.

8. On the Custom Mappings page, click Next.


10. Click Finish.

The service endpoints interfaces and methods are generated, as displayed in Figure 21–4.

**Figure 21–4 Generating Service Endpoints Interfaces and Methods**
11. Create a new Java class, specifying the package you selected in step 7.

12. In the Java class, add the code to access the Web Services interface. A sample code is described in Section 21.2.3.4, “Sample Code for Accessing the Web Services Interface.” If you use the sample code, you must specify your own data for details such the user name, passcode, and domain.

13. Right-click the page source and click Run.

21.2.3.4 Sample Code for Accessing the Web Services Interface

The following code is an example of a Java program that accesses the Web Services interface. This program lists all the page names in the Training domain. You must first create a web service proxy to use this code in a Java class.

```java
package wiki;
import wiki.WikiService;
import wiki.WikiServiceService_Impl;
import wikiservice.PageInfo;

public class ListPages {

Each Web Services method must authenticate the caller to the Web Services. Authentication consists of a user name and a preconfigured passcode. In this example, we code these values.

    private static final String username = "jsmith";
    private static final String passcode = "passcode";

We code the domain name in this example.

    private static final String domain = "Training";

    public static void main(String[] args) throws Exception {
        try {

Next, we create a client-side proxy to access the Web Services.

        WikiServiceService_Impl myImpl = new WikiServiceService_Impl();

We then set the end point of the proxy to the actual location where we deployed the Web Services.

        WikiService Service = myImpl.getWikiRemoteService();

Each Web Services method must pass a security key to authenticate the user and call the method. We can generate this key by calling the login web service using the user's name and the Web Services-configured passcode.

        String key = client.login(username, passcode);

Using the Web Services proxy, we fetch into an array the information about all the pages in the selected domain. If there is no such domain, then the program throws an exception. If the domain does not contain any pages, then the program returns an empty array.

        PageInfo[] pages = client.getAllPageInfo(domain, key);
        System.out.println("Pages in " + domain + " domain: ");
```
Throughout the pages array, print the name of each page using the getter method getName().

```java
for (int i = 0; i < pages.length; i++)
{
    System.out.println("  " + pages[i].getName());
}
}
```

If there is an exception, then the program captures the error and prints it.

```java
catch (Exception e)
{
    System.out.println("Exception: " + e);
}
}
```

For more examples of portlets capable of consuming a URL, see the Oracle WebCenter Suite page on the Oracle Technology Network (http://webcenter.oracle.com).

### 21.3 Advanced Information for the Wiki and Blog Services

**Adding a Search Analyzer**

Oracle WebCenter Wiki and Blog Server leverages the Lucene search engine for tokenizing wiki content (that is, to determine what constitutes a word). Lucene enables you to use custom search analyzers, which are language-specific extensions that you can use to add required rules, such as rules for separating words and ignoring the case. For example, Oracle WebCenter Wiki and Blog Server provides out-of-the-box support for multibyte characters. However, the search engine implementation in Oracle WebCenter Wiki and Blog Server does not support the ability to tokenize wiki content on anything other than whitespace. You can add your own search analyzer that provides the ability to search for multibyte characters.

To add a custom search analyzer:

1. Place the JAR file containing the .class files for your custom analyzer in the WEB-INF/lib folder of your deployed Oracle WebCenter Wiki and Blog Server.
2. Open the application_config.script file from the WEB-INF/classes folder of your deployed Oracle WebCenter Wiki and Blog Server.
3. Search for the custom_analyzer entry under wiki.
4. Update the custom_analyzer entry to include the full package and class name of your search analyzer in which you have added the required rules.
   
   For example, if you are using the Chinese Sandbox, then the entry should look like:

   ```
   custom_analyzer : org.apache.lucene.analysis.cn.ChineseAnalyzer
   ```

   **Note:** If the custom_analyzer entry does not exist, add the entry in application_config.script.

5. Save the application_config.script file.
Part V contains the following chapters:

- Chapter 22, "Integrating the Links Service"
- Chapter 23, "Integrating the Tags Service"
- Chapter 24, "Integrating the Search Service"
This chapter explains how to integrate the Links service in a custom WebCenter application at design time.

For more information about managing and including links, see:

- “Setting up the WebCenter Repository” in the Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter
- "Working with the Links Service” in the Oracle Fusion Middleware User’s Guide for Oracle WebCenter

This chapter includes the following sections:

- Section 22.1, "Introduction to the Links Service"
- Section 22.2, "Basic Configuration for the Links Service"

22.1 Introduction to the Links Service

This section provides overview information about the Links service features and requirements. It contains the following subsections:

- Section 22.1.1, "Understanding the Links Service"
- Section 22.1.2, "Requirements for the Links Service"
- Section 22.1.3, "What Happens at Runtime"

22.1.1 Understanding the Links Service

The Links service provides a way to view, access, and associate related information. For example, in a list of project assignments, you can link to the specifications relevant to each assignment. In a discussion thread about a problem with a particular task, you can link to a document that provides a detailed description of how to perform that task.

The Links service provides a means for the application developer to set up source objects (for example, the Discussions service) and target objects (for example, a document), thus enabling your users to create links between the two objects.

There are three actions associated with the Links service: create, delete, and manage. The manage action includes the create and delete actions.

The following custom JSF components are included in the service:

- **Links Detail Button**: This displays an icon and (optionally) a hyperlink that users click to open the Links Panel. To use the Links Detail Button, you must also include the Links - Dialog task flow as a region on the page.
Links Detail Menu Item: This adds a menu item that opens the Links Panel. You can embed this item in an ADF menu. To use the Links Detail Menu Item, you must also include the Links - Dialog task flow as a region on the page.

Links Status Icon: This has two versions:

The gray Links icon (Figure 22–1) indicates that no links are present in the Links dialog.

![Figure 22–1 The Links Icon (No Links Present)](image1)

The gold Links icon (Figure 22–2) indicates that links are present in the Links dialog.

![Figure 22–2 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 following WebCenter services:

- Mail
- People Connections
The Links service supports bidirectional links between objects. For example, when you create a link from a discussion topic to a document, a link from the document back to the discussion topic also is created. Similarly, when you delete the link from the discussion topic to a document, the link from the document back to the discussion topic is automatically deleted. Bidirectional linking is not available for URLs, notes, and specific list rows.

---

**Note:** The Lists and Notes services are available in WebCenter Spaces only.

---

### 22.1.2 Requirements for the Links Service

Links automatically recognize any WebCenter service in your application. After you have configured a service in your application, you can add links. However, links work only on secured pages. Links icons do not appear on unsecured pages. For more information, see Section 22.2.3, "Setting Security for the Links Service."

### 22.1.3 What Happens at Runtime

Linking provides an easy way for you to share information with your social network. Linking can help you realize a significant reduction of wasted time and effort normally spent looking for information.

The Links service exposes its features through a Links dialog (Figure 22–3), accessible wherever the Links icon (Figure 22–1) appears in your application.

**Figure 22–3  Links Dialog**

With the Links service, you can do the following:

- Link an object (such as a page) to an existing object (such as a discussion topic) by clicking the **Link** icon, selecting **Link to Existing**, choosing the resource **Discussions**. Optionally, you can choose a specific forum and click the topic title to choose a link.

- Link an object (such as a discussion topic) to a new object (such as a new note or URL) by clicking the **Link** icon for discussion topic, selecting **Link to New**, and choosing either **Note** or **URL**.

- Create multiple links from one object.

- Delete a link.
Basic Configuration for the Links Service

For more information about the Links service at runtime, see *Oracle Fusion Middleware User’s Guide for Oracle WebCenter*.

---

**Note:** You can see the Links service in action in the sample application, as described in Chapter 2, “Introduction to the WebCenter Sample Application.”

---

### 22.2 Basic Configuration for the Links Service

This section describes the steps required for adding the Links service to your application. It contains the following subsections:

- Section 22.2.1, "Setting up Connections for the Links Service"
- Section 22.2.2, "Adding the Links Service at Design Time"
- Section 22.2.3, "Setting Security for the Links Service"
- Section 22.2.4, "Troubleshooting the Links Service"

#### 22.2.1 Setting up Connections for the Links Service

The Links service requires a connection to the database where the WebCenter 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.

---

**Note:** For details about installing the database and the WebCenter schema, see *Oracle Fusion Middleware Installation Guide for Oracle WebCenter*.

---

To create the database connection:

1. In the Application Navigator, expand the **Application Resources** panel.
2. Right-click **Connections**, then click **New Database**.
3. Enter the following information for your database connection:
   - **Connection Name**: WebCenter
   - **Connection Type**: Oracle (JDBC)
   - **Username**: username
   - **Password**: password
   - **Host**: host where you install the WebCenter schema; for example, localhost
   - **JDBC Port**: port; for example, 1521
   - **SID**: system identifier for the database with the same JDBC port; for example, ORCL

You must enter the **Connection Name** exactly as "WebCenter". There are cases when you may want to leverage an existing database connection for WebCenter services, and it may not be possible to change the database connection name to "WebCenter".

To allow WebCenter services to use another database connection by a different name, you must add the following `<data-source>` tag as a child of the `<wpsC:adf-service-config>` element in the `adf-config.xml` file.
Basic Configuration for the Links Service

Integrating the Links Service

(adf-service-config is a child of adf-config, and data-source is a child of adf-service-config or sibling of extension-registry-config.)

For example:

Example 22–1 Adding a data-source Tag in adf-config.xml

<adf-config xmlns="http://xmlns.oracle.com/adf/config"
xmlns:mdsC="http://xmlns.oracle.com/adf/mds/config"
xmlns:wpsC="http://xmlns.oracle.com/webcenter/framework/service"
xmlns:jndiC="http://xmlns.oracle.com/adf/jndi/config">

  <wpsC:adf-service-config xmlns="http://xmlns.oracle.com/webcenter/framework/service">
  <data-source jndi-name="java:/comp/env/jdbc/NewDatabaseConnDS"/>
  </wpsC:adf-service-config>

</adf-config>

Note: The syntax for jndi-name is "java:/comp/env/jdbc/NewDatabaseConnDS". This is derived from the example name NewDatabaseConn used to create the database connection in the creation wizard.
4. Click OK.

---

**Note:** While you can set up the connections to back-end servers at design time in Oracle JDeveloper, you can later add, delete, or modify connections in your deployed environment using Enterprise Manager Fusion Middleware Control. For more information, see Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter.

---

### 22.2.2 Adding the Links Service at Design Time

This section explains a basic incorporation of the Links service. It contains the following subsections:

- Section 22.2.2.1, "Links Service Task Flows."
- Section 22.2.2.2, "How to Add the Links Service to your Application."
22.2.2.1 Links Service Task Flows
The Links service includes one task flow: Links Dialog.

22.2.2.2 How to Add the Links Service to your Application
To add the Links Dialog task flow to your WebCenter application:

1. Follow the steps described in Chapter 3, "Preparing Your Development Environment" to implement security and create a new customizable page in your application.
2. Ensure that you have set up the database connection to a database with the WebCenter schema installed (Section 22.2.1).
3. Open the page on which you want to add the Links service.
4. In the Component Palette, click WebCenter Links Service.
5. Drag and drop the Links Detail Button component onto your page inside the panelGroupLayout.
   The button is placed inside of a panelGroupLayout for the purposes of this example only. It is not required that you always place the button inside a panelGroupLayout.
6. In the Insert Links Detail Button dialog (Figure 22–5), enter a unique object description, ID, and name.

Figure 22–5 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.

The Properties in this dialog include:

- **ObjectDescription**: Links Detail Button
- **ObjectId**: linkButton
- **ObjectName**: linkButton
- **ServiceId**: OnDemand
Basic Configuration for the Links Service

---

**Note:** The Links service combines the `serviceId` and `objectId` to uniquely identify the object to which you bind the Links Detail Button.

---

8. Click **OK**.

The new button displays in your page source (Figure 22–6).

![The Link Detail Button in Your Page Source](image)

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.

22.2.3 Setting Security for the Links Service

Links work only on secured pages. Links icons do not appear on unsecured pages. For information about configuring ADF security, see Section 3.5, "Implementing Security in Your Application."

Table 22–1 shows the permissions granted when you add Links components to a page. Permissions, such as `RelationshipPermission` are automatically granted to authorized users when a service is consumed.

<table>
<thead>
<tr>
<th>Name</th>
<th>Class</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>* (This allows the logged-in user to create and delete links; otherwise, the user sees only available links.)</td>
<td><code>oracle.webcenter.relationship.model.security.RelationshipPermission</code></td>
<td>manage</td>
</tr>
</tbody>
</table>

22.2.4 Troubleshooting the Links Service

This section describes common problems and solutions for the Links service.
**Problem**
The Links icon does not appear.

**Solution**
The Links service requires a database connection to the WebCenter schema, where links information is stored. Make sure that you have created the connection to the database and made it available in the Application Resources panel of the Application Navigator. If the connection is available in the Resource Palette but not in Application Resources, then simply drag the connection from the Resource Palette to the Connections folder in Application Resources.

**Problem**
Existing links appear, but you are not able to create new links or delete existing links.

**Solution**
The RelationshipPermission permission is automatically granted to authorized users when a service is consumed. Verify that this permission has been granted. For information, see Section 4.1.3, "Automated Task Flow Grants."
This chapter explains how to integrate the Tags service in a custom WebCenter application at design time.

For more information about managing and including tags, see:

- “Setting up the WebCenter Repository” in the Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter
- "Working with the Tags Service" in the Oracle Fusion Middleware User’s Guide for Oracle WebCenter

This chapter includes the following sections:

- Section 23.1, "Introduction to the Tags Service"
- Section 23.2, "Basic Configuration for the Tags Service"
- Section 23.3, "Advanced Information for the Tags Service"

### 23.1 Introduction to the Tags Service

Tags enable users to apply their own meaningful terms to items, making those items more easily discoverable in search results and the Tag Center. The Tag Center is a page that displays the interactions between all the tags, tagged items, and their taggers in a WebCenter application.

Having multiple users tag objects contributes to the collective knowledge, which makes searching much more relevant. WebCenter 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 as well as 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 for others to find it as well.

The Tags service is available for pages, documents, and custom objects.

### 23.1.1 Understanding the Tags Service

You can apply one or more meaningful terms, called tags, to remind yourself and alert other users of the content they might expect to find at the tagged location. Anywhere you see the Tags icon (Figure 23–1), you can apply a tag.
Introduction to the Tags Service

Figure 23–1  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 23–2) displays all currently applied tags.

Figure 23–2  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.

23.1.2 Requirements for the Tags Service

The Tags service requires that you set up the WebCenter schema and create a database connection to the schema. Tag information is stored in the database. For details, see Section 23.2.1, "Setting up Connections for the Tags Service."

Note: Typically, when you add tags, you also add a search toolbar to enable searching for the objects with the tags. To add the search toolbar, follow the instructions in Section 24.2.2, "Adding the Search Service at Design Time."

There are no additional requirements to use the Tags service. However, to enable the ability to tag pages, you must manually add the Page service libraries to your project. Right-click your ViewController project and select Project Properties and then Libraries and Classpath. Click Add Library and select the WebCenter Page Service and WebCenter Page Service View libraries, as shown in Figure 23–3. Click OK.
Similarly, to enable the ability to tag documents, you must manually add the Document Library service libraries to your project, as shown in Figure 23–4.

**Figure 23–4 Documents Service Libraries**

23.1.3 What Happens at Runtime

At runtime, 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. By default, tagged items that are accessed through their tags are rendered in a popup window, but you can override this by specifying other types of open behavior.
Basic Configuration for the Tags Service

For more information about the service at runtime, see Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

Note: You can see the Tags service in action in the sample application, as described in Chapter 2, "Introduction to the WebCenter Sample Application."

23.2 Basic Configuration for the Tags Service

This section describes required steps for adding this service to your application.

23.2.1 Setting up Connections for the Tags Service

The Tags service requires a connection to the database with the WebCenter schema installed. The script to create the WebCenter schema is included with the product.

For details about installing the database and the WebCenter schema, see Oracle Fusion Middleware Installation Guide for Oracle WebCenter.

23.2.1.1 How to Set Up Connections for the Tags Service

To create the database connection:

1. In the Application Navigator, expand the Application Resources panel.
2. Right-click Connections, then click New Database.
3. Enter the following information for your database connection:
   - **Connection Name**: WebCenter
   - **Connection Type**: Oracle (JDBC)
   - **Username**: username
   - **Password**: password
   - **Host**: host where you install the WebCenter schema; for example, localhost
   - **JDBC Port**: port; for example, 1521
   - **SID**: system identifier for the database with the same JDBC; for example, ORCL

You must enter the **Connection Name** exactly as "WebCenter". There are cases when you may want to leverage an existing database connection for WebCenter services, and it may not be possible to change the database connection name to "WebCenter".

To allow WebCenter services to use another database connection by a different name, you must add the following `<data-source>` tag as a child of the `<wpsC:adf-service-config>` element in the `adf-config.xml` file. (adf-service-config is a child of adf-config, and data-source is a child of adf-service-config or sibling of extension-registry-config.)

For example:

```xml
<adf-config xmlns="http://xmlns.oracle.com/adf/config"
            xmlns:mdsC="http://xmlns.oracle.com/adf/mds/config"
            xmlns:wpsC="http://xmlns.oracle.com/webcenter/framework/service"
            xmlns:jndiC="http://xmlns.oracle.com/adf/jndi/config">
  <wpsC:adf-service-config
```

Note: You can see the Tags service in action in the sample application, as described in Chapter 2, "Introduction to the WebCenter Sample Application."
Basic Configuration for the Tags Service

Integrating the Tags Service

```xml
xmlns=http://xmlns.oracle.com/webcenter/framework/service>
  <data-source jndi-name="java:/comp/env/jdbc/NewDatabaseConnDS"/>
</wpsC:adf-service-config>
</adf-config>

Note: The syntax of the example name is "java:/comp/env/jdbc/NewDatabaseConnDS". This is derived from the example name NewDatabaseConn you would have used to create the database connection in the creation wizard.

Figure 23–5 Database Connection

4. Click OK.

Note: While you can set up the connections to back-end servers at design time in Oracle JDeveloper, you can later add, delete, or modify connections in your deployed environment using Enterprise Manager Fusion Middleware Control. For more information, see Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter.

23.2.2 Adding the Tags Service at Design Time

To enable basic tagging, you must add either the Tagging Button component or the Tagging Menu Item component and the Tagging - Dialog task flow. In addition, to tag custom objects, you must register a resource viewer to visualize the tagged component when it is discovered by users.

This section describes how to enable tagging by adding the Tagging Button and the Tagging - Dialog task flow.
23.2.2.1 Tags Service Components

The following JSF components are included in the service:

- **Tagging Button**: When you add a Tagging Button, a Tags link on the page enables you to invoke the Tag This Item (or Tag This Page) dialog at runtime.

- **Tagging Menu Item**: When you add a Tagging Menu Item, a menu option on the page enables you to invoke the Tag This Item (or Tag This Page) dialog at runtime.

23.2.2.2 Tags Service Task Flows

Table 23–1 lists the Tags service 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. It allows for a service ID input parameter that you can use to filter out tagged items that are not 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 input parameters are serviceId and resourceId. 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. If this task flow and the Tagging - Tagged Items task flow are wired together, then 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.) For more information, see Section 23.3.3, &quot;Using Tagged Cloud and Tagged Items Together.&quot;</td>
</tr>
<tr>
<td>Tagging - Tagged Items</td>
<td>This task flow displays items or pages 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. <strong>Note</strong>: This task flow should be wired to the Tagging - Tag Cloud task flow, so that tagged items appear in the Tag Center only when a tag is selected in the tag cloud. For more information, see Section 23.3.3, &quot;Using Tagged Cloud and Tagged Items Together.&quot;</td>
</tr>
</tbody>
</table>
23.2.2.3 How to Add the Tags Service to your Application

To add the Tags service to your WebCenter application:

1. Follow the steps described in Chapter 3, "Preparing Your Development Environment" to implement security and create a new customizable page in your application.

2. Ensure that you have set up a database connection to a database with the WebCenter 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 23–2 provides descriptions and example values for the fields in the dialog.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| **ResourceId** | The ID that uniquely identifies the object to which you are binding the Tagging Button. The value in this field need not be static. For example, you could use EL to insert a unique value for each row in a table. For example:  
  - To tag pages, enter #{facesContext.viewRoot.viewId}  
  - To tag custom components, enter customComponent123 |
| **ResourceName** | The name of the object to which you are binding the Tagging Button. The value in this field need not be static. For example, you could use EL to insert a unique value for each row in a table.  
  - To tag pages, enter #{facesContext.viewRoot.viewId}  
  - To tag custom components, enter Custom Component 1 |
| **ServiceId**  | An application-wide ID. (With custom components, you add this to service-definition.xml when adding the resource viewer.)  
  - To tag pages, enter oracle.webcenter.page  
  - To tag custom components, enter mycompany.myproduct.myapp.mycomponent |

6. Click OK.

7. From the Resource Palette, expand **WebCenter Services Catalog** and **Task Flows**.

8. Drag the **Tagging Dialog** from the Resource Palette and drop it on the page.

9. When prompted, select **Region** as the way to create the task flow.
10. For custom resources, follow the steps described in Section 4.2, "Extending Your Application with Custom Components" to register a resource viewer. For services exposing resources to be tagged (and therefore searched and viewed) WebCenter provides a Resource Action Handling framework. The Tags service uses this framework to allow acting on a search result.

In the Source view where you added the Tags service, you should see something similar to Example 23–1.

**Example 23–1 Tagging Button and Tagging - Dialog Task Flow Source**

```xml
<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 (Figure 23–7).

**Figure 23–7 Tag This Page Dialog**

![Tag This Page Dialog](image)

11. When the Tag This Page 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.

12. In the search toolbar, type the search criteria **home** and click the **Search** icon.

13. You should **home** under Tags. (Figure 23–8)

**Figure 23–8 Search Results with Tagged Items and Tags**

![Search Results with Tagged Items and Tags](image)

14. For custom resources, you also would see **Custom Component 1** under Tagged Items. Click **Custom Component 1** and a dialog appears with the resource viewer you just created (Figure 23–9).
23.2.3 Setting Security for the Tags Service

The Tags service does not require ADF security.

However, the Tagging - Personal View and Tagging - Dialog task flows do require authentication. Also, if the application is not secured, then pages are not returned in search results for tagged items.

When users are not authenticated, the user name associated with tags in the user's bookmarks are "anonymous." Multiple users using the same unprotected custom WebCenter applications risk overwriting each other's tags.

For information about configuring ADF security, see Section 3.5, "Implementing Security in Your Application."

---

**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 4.2, "Extending Your Application with Custom Components."

---

23.3 Advanced Information for the Tags Service

This section describes optional features available with this service.

23.3.1 Optional Way to Show Tags on Pages

While it is possible to add the Tagging Button and task flows on a page-by-page basis, typically you want to add the search toolbar and tags as part of a page template that you can apply whenever building a page that requires it. To add tags through a page template:

1. Open the project where you plan to create pages with the Tags service.
2. Right-click the project name and select New from the context menu.
3. Under Categories, select JSF and, under Items, select JSF Page Template.
4. For File Name, enter looksee1template.
5. Click OK.

6. Go to the Source view and modify the code to be similar to that in Example 23–2.

Example 23–2  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">
        <!-- Our toolbar -->
        <af:panelGroupLayout id="toolbar" layout="horizontal" halign="end"
            inlineStyle="background-color:cyan;width:100%">
        </af:panelGroupLayout>
        <af:panelStretchLayout startWidth="300"
            inlineStyle="height:600px; width:100%;">
            <!-- Our sidebar -->
            <af:panelSplitter id="sidebar" orientation="vertical">
                <f:facet name="first">
                </f:facet>
                <f:facet name="second">
                </f:facet>
            </af:panelSplitter>
        </af:panelStretchLayout>
    </af:panelGroupLayout>
    <f:facet name="start">
        <af:panelSplitter id="sidebar" orientations="vertical">
            <f:facet name="first">
            </f:facet>
            <f:facet name="second">
            </f:facet>
        </af:panelSplitter>
    </f:facet>
</af:pageTemplateDef>
```
7. On the Component Palette, select the Tagging Button and drop it on the page, inside the PanelGroupLayout. Make it the first child of the PanelGroupLayout.

8. Fill out the Insert Tagging Button dialog as shown in Figure 23–11.

Table 23–3 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.</td>
</tr>
<tr>
<td></td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td>In this case, enter oracle.webcenter.page.</td>
</tr>
<tr>
<td></td>
<td>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.

10. From the Resource Palette, expand the WebCenter Services Catalog and the Task Flows.

11. Drag Tagging Dialog from the Resource Palette and drop it on the page inside of the panelGroupLayout.

12. When prompted, select Region as the way to create the task flow.

13. Drag Search Toolbar and drop it after the Tagging Dialog you just added to the page template in the panelGroupLayout.

14. When prompted, select Region as the way to create the task flow.

15. Drag Document Library - Recent Documents and drop it in the second facet of the panelSplitter with identifier sidebar.
16. When prompted, select Region as the way to create the task flow.

17. In the Edit Task Flow Binding dialog, for connectionName, enter the name of the connection to your document library's content repository; for example, \${'MyFileSystem'}. For information on how to set up a connection to your content repository, see Section 13.2, "Configuring Content Repository Connections."

18. The source for your page template should now look similar to Example 23–3.

**Example 23–3  JSF Page Template with Tagging 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:panelGroupLayout>
    <af:panelStretchLayout startWidth="300"
        inlineStyle="height: 600px; width: 100%;">
        <f:facet name="start">
            <af:panelSplitter orientation="vertical">
                <f:facet name="first">
                    <af:region value="#{bindings.taskservicetaskflowdefinition1.regionModel}"
                        id="taskservicetaskflowdefinition1"
                        inlineStyle="position: absolute; left: 0px; width: 100%; height: 100%"/>
                </f:facet>
                <f:facet name="second">
                    <af:region value="#{bindings.doclibrecentdocuments1.regionModel}" id="doclibrecentdocuments1"/>
                </f:facet>
            </af:panelSplitter>
        </f:facet>
        <f:facet name="center">
            <af:facetRef facetName="content"/>
        </f:facet>
    </af:panelStretchLayout>
</af:panelGroupLayout>
</af:pageTemplateDef>
```

19. Create two new pages based upon the page template you just created. In the Create JSF Page dialog, select looksee1template1 from the Use Page Template list.
• Create a page called Search.jspx and add the Search task flow in the center facet.

• Create a page called Documents.jspx and add the Document Library - List View task flow as the center facet. In the Edit Task Flow Binding dialog, for connectionName, enter the name of the connection to your document library's content repository; for example, ${'MyFileSystem'}. For information on how to set up a connection to your content repository, see Section 13.2, "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 23–12).

Figure 23–12 Tag this Page Dialog for Search Page

4. Click Save.

5. Navigate to Documents.jspx (or run it separately).

6. Click the Tags link.

7. In the Tag this Page dialog, enter doclib documents page test customapp for Tags (Figure 23–13).

Figure 23–13 Tag this Page Dialog for Documents Page

8. Click Save.

9. In the search toolbar for the Documents.jspx page, enter search and click the Search icon.

10. Under Tags, click search.

23.3.2 Using the Resource Action Handling Framework to Tag Custom Objects

For WebCenter services exposing resources to be tagged (and therefore searched and viewed) WebCenter provides a Resource Action Handling framework. For example, rows in a table can be tagged.
To register a resource viewer, see Section 4.2, "Extending Your Application with Custom Components."

### 23.3.3 Using Tagged Cloud and Tagged Items Together

Using the events mechanism, you should wire together the **Tagging - Tag Cloud** and **Tagging - Tagged Items** task flows to show tagged items only for the selected tags in the tag cloud. This avoids tagged items displaying all items that have been tagged by all users, improving performance.

After adding the **Tagging - Tag Cloud** and **Tagging - Tagged Items** task flows to a page, there are two ways to wire them together.

---

**Note:** The *producer* for the event is the tag cloud, and the *consumer* of the event is the tagged items.

---

The first, and easiest, way is to cut and paste the following snippet to the page definition, and then replace the region name `relatedresources1` with the task flow name used in your page definition.

```xml
<eventMap xmlns="http://xmlns.oracle.com/adfm/contextualEvent">
  <event name="refreshAllRelated">
    <producer region="***">
      <consumer region="relatedresources1">
        handler="oracle_webcenter_tagging_view_jsf_fragments_tag_center_related_resourcesPageDef.getTagRelatedResources">
          <parameters>
            <parameter name="p0" value="${payLoad}"/>
          </parameters>
        </consumer>
      </producer>
    </event>
    <event name="setShowSystemResources">
      <producer region="***">
        <consumer region="relatedresources1">
          handler="oracle_webcenter_tagging_view_jsf_fragments_tag_center_related_resourcesPageDef setShowSystemResources"/>
        </consumer>
      </producer>
    </event>
    <event name="setShowPersonalResources">
      <producer region="***">
        <consumer region="relatedresources1">
          handler="oracle_webcenter_tagging_view_jsf_fragments_tag_center_related_resourcesPageDef setShowPersonalResources"/>
        </consumer>
      </producer>
    </event>
  </event>
</eventMap>
```

Figure 23–14 shows an example of this.
The second way to wire the together the **Tagging - Tag Cloud** and **Tagging - Tagged Items** task flows is to follow these steps. The following example shows steps for adding the `refreshAllRelated` event.

1. Right-click your page, then choose **Go to Page Definition** from the context menu.
2. On the Structure window, on lower left hand side, right-click **bindings**, and select **Insert after bindings** and then **eventMap**.
3. Right-click the **eventMap**, and select **Insert inside eventMap** and then **event**.

---

**Figure 23–14  Page Definition Replacement**

```xml
<executables>
  <taskFlow id="tagging-launch-dialog">
    xmlns="http://mule.oracle.com/ns/bindings"/>
  </taskFlow>
  <taskFlow id="tagging-launch-tagcloud">
    xmlns="http://mule.oracle.com/ns/bindings"/>
  </taskFlow>
  <taskFlow id="related-resource-tagging">
    taskFlowUri="/oracle/webcenter/tagging/controller/taskflows/related-resources.xml?tagging-related-resources"
    xmlns="http://mule.oracle.com/ns/bindings"/>
  </taskFlow>
</executables>

<bindings>
  <eventMap name="refreshAllRelated">
    <producer ref="relatedResources"/>
    <consumer ref="TagCloud.relatedResourcesPageDef.getTagRelatedResourcesPageDef">
      <parameters>
        <parameter name="0" value="/payLoad"/>
      </parameters>
    </consumer>
  </eventMap>
</bindings>
```
Figure 23–16  Page Definition - Event Map

4. In the dialog, enter the event name from the snippet above. For this example, enter `refreshAllRelated`.

5. Right-click `refreshAllRelated`, select Insert inside `refreshAllRelated` and then Producer.

6. When the dialog asks for the producer region, enter *.

7. Right-click the producer, select Insert inside product, and select Consumer.

8. When the dialog asks for the handler name, enter `oracle_webcenter_tagging_view_jsf_fragments_tag_center_related_resourcesPageDef.getTagRelatedResources`.

9. Because that dialog asks only for the consumer handler name and not the consumer region, you must add it manually. From the main window, select the consumer node. From the Property Inspector, you see a region field. Enter the related resources task flow name in your pageDef. By default, when you drag and drop the task flow to the page, Oracle JDeveloper names it `relatedresources1`. If you later change the name, then make sure to use that new name instead of `relatedresources1`. (Figure 23–17)
10. Right-click the consumer, Insert inside consumer, and then parameters.

11. Right-click parameters, Insert inside parameters, and then parameter.

12. When the dialog asks for the parameter name, enter anything; for example, p0. For the value, enter ${payLoad}.

Simply repeat the steps for the other events, substituting the corresponding event name and the consumer handler.

**Note:** The setShowSystemResources and setShowPersonalResources events do not take parameters, so step 9 to 11 are not needed.

Table 23–4 shows the information for the refreshAllRelated event.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Producer region</td>
<td>*</td>
</tr>
<tr>
<td>Consumer region</td>
<td>The related task flow name in your pageDef. By default, this is relatedresources1.</td>
</tr>
<tr>
<td>Consumer handler</td>
<td>oracle_webcenter_tagging_view_jsf_fragments_tag_center_related_resourcesPageDef.getTagRelatedResources</td>
</tr>
<tr>
<td>Parameter</td>
<td>This should be some meaningful name (for example, p0); value: ${payLoad}</td>
</tr>
</tbody>
</table>

You can then click a tag in the tag cloud to see associated tagged items.

These are the only two task flows that talk to each other using events, but you can design your own task flows that listen to WebCenter events.
This chapter explains how to integrate the Search service in a custom WebCenter application at design time.

For more information about managing and including search, see:

- "Managing the Search Service" in the *Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter*
- "Working with the Search Service" in the *Oracle Fusion Middleware User’s Guide for Oracle WebCenter*

This chapter includes the following sections:

- Section 24.1, "Introduction to Search"
- Section 24.2, "Basic Configuration for the Search Service"
- Section 24.3, "Advanced Information for the Search Service"

### 24.1 Introduction to Search

WebCenter search provides a unified, extensible framework that enables the discovery of information and people through an intuitive user interface. It honors application security settings by returning only the results a user is authorized to view. Because a search spans all accessible services—and can be extended to span your enterprise—users are not required to switch between applications to perform multiple searches. Additionally, users can save their searches for reuse. This includes searches with complex search refinement conditions.

Any WebCenter service that owns resources supports Search. This includes: announcements, discussions, documents, tags, wikis/blogs, and pages. You can use the adapter to Oracle Secure Enterprise Search (SES) instances to search resources outside of WebCenter.

If you have the Documents service in your application, then Search looks for terms in document names and in the documents themselves. The results include tags, so any matching tags on your documents and pages are also returned, with relevance based on the quality and frequency of tags.
24.1 Understanding the Search Service

WebCenter Search uses search adapters to find content created inside WebCenter applications by WebCenter services: It does not rely on Oracle Secure Enterprise Search (SES).

However, you can extend WebCenter searches to external content repositories by connecting the WebCenter application to an Oracle SES instance. Providing that the Oracle SES instance is set up to search external repositories, results from these search sources can appear alongside WebCenter application search results. Supported versions include Oracle SES 10.1.8.4 or later. You can register multiple Oracle SES connections but only one of them is active at a time.

WebCenter Spaces

In WebCenter Spaces, you can configure Oracle SES to search WebCenter services, instead of using the individual search adapters for each service. This provides a unified ranking system for all search results in WebCenter Spaces. For more information, see the section "Overriding Default Search Adapters in WebCenter Spaces" in the Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter.

24.1.2 What Happens at Runtime

With the Search service, you can create, run, refine, and save custom searches. By default, WebCenter Search looks for your search terms in every searchable service available to the application (a global search). You can perform a global search from any Search task flow, or from the Search Toolbar.

In a custom WebCenter application, search results are displayed in a separate window, where you can scroll through all the results. Optionally, in the Search Results window, click the More button. The results of saved searches appear in a new Search Results window.

The Search Preferences task flow lets users control which WebCenter services to search. Users also can control the order in which those services are listed in the results, and what information is included, such as the type, size, or owner of the object returned. (For information about adding the task flow, see Section 24.3.3, "Adding the Search Preferences Task Flow.")

To set search preferences in a custom WebCenter application:

1. Log in and locate the page that contains the Search Preferences task flow.
2. To specify the services to be searched, select one or more services on the **Available Services** list by checking their checkboxes, and then click the **Move** icon to move them to the **Selected Services** list.

   You can also specify the order in which the search results are displayed by clicking the move up and move down icons next to the **Selected Services** list.

3. To specify the type of information to display with the search results, click the **Columns** tab and move one or more items from the **Available Columns** list to the **Selected Columns** list. Choose from the following:
   - **Icon** displays an icon representing the item type.
   - **Title** displays linked titles.
   - **Person** to display the user name of the person who owns the item.
   - **Size** displays the size of the item, for example, file size.
   - **Date** displays the date the item was added to the application.
   - **Type** displays the type of the item, for example image.

   You can also specify the order in which this information is displayed by clicking the move up and move down icons next to the **Selected Columns** list.

4. Click **OK** to save your changes.

For more information about the Search service and task flows at runtime, see *Oracle Fusion Middleware User’s Guide for Oracle WebCenter*.

---

**Note:** You can see the Search service in action in the sample application, as described in Chapter 2, "Introduction to the WebCenter Sample Application."

---

### 24.2 Basic Configuration for the Search Service

This section provides the steps for adding the Search service to your application. It contains the following subsections:

- **Section 24.2.1, "Setting up Connections for the Search Service"**
- **Section 24.2.2, "Adding the Search Service at Design Time"**
- **Section 24.2.3, "Setting Security for the Search Service"**

#### 24.2.1 Setting up Connections for the Search Service

The Search service does not require a connection. However, to incorporate Oracle Secure Enterprise Search (SES) results in your result set, you must add a connection to Oracle SES. For more information, see **Section 24.3.4, "Including Oracle SES Results in WebCenter Search Results."**

#### 24.2.2 Adding the Search Service at Design Time

This section describes the Search service task flows and explains how to integrate the Search service into your application. It contains the following subsections:

- **Section 24.2.2.1, "Search Service Task Flows"**
- **Section 24.2.2.2, "How to Add the Search Service to Your Application"**
### 24.2.2.1 Search Service Task Flows

Table 24–1 lists the Search service task flows.

<table>
<thead>
<tr>
<th>Task Flow</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search</td>
<td>This task flow provides a rich search experience with several options for searching and features for refining and saving search results.</td>
</tr>
<tr>
<td>Search - Saved Searches</td>
<td>This task flow enables you to create a simple launch pad for running saved searches within the application.</td>
</tr>
<tr>
<td>Search Preferences</td>
<td>This task flow enables users to select which WebCenter services to search. It also provides users with a means of selecting and ordering the columns available in the search results of the Search task flow.</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>

### 24.2.2 How to Add the Search Service to Your Application

The Search Toolbar task flow offers the easiest way to add the Search service to your application. To add the Search Toolbar task flow to your WebCenter application:

1. Follow the steps described in Chapter 3, "Preparing Your Development Environment" to implement security and create a customizable page in your application.
2. Open the page on which to add the Search service.
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.

**Note:** To return results, the Search service requires that other WebCenter services are included in the application.

On the Source tab of the page where you added the toolbar, you can see the Search Toolbar task flow in the page source, similar to Example 24–1.

**Example 24–1** Search Toolbar Task Flow Source

```xml
<af:form id="f1">
  <af:region value="#{bindings.searchtoolbar1.regionModel}" id="r1"/>
</af:form>
```

In the Source view of the pagenamepageDef.xml file, you should see something similar to Example 24–2.
Example 24–2  pageDef.xml File with the Search Toolbar

```xml
<taskFlow id="searchtoolbar1"
  taskFlowId="/oracle/webcenter/search/controller/taskflows/
  localToolbarSearch.xml#search-toolbar"
  xmlns="http://xmlns.oracle.com/adf/controller/binding"/>
```

24.2.3 Setting Security for the Search Service

The query execution of individual services ensures that the user performing the search can view results. Search results are returned based on user privileges.

The Search service does not require ADF security. When unauthenticated, search queries provided by all services return only public content. However, only authenticated users of an ADF-secured WebCenter application can save searches. Unauthenticated users see the **Save** button grayed out (See Figure 24–2).

For information about configuring ADF security, see Section 3.5, "Implementing Security in Your Application."

24.3 Advanced Information for the Search Service

This section describes optional features available with this service. It contains the following subsections:

- Section 24.3.1, "Adding the Search Task Flow"
- Section 24.3.2, "Adding the Search - Saved Searches Task Flow"
- Section 24.3.3, "Adding the Search Preferences Task Flow"
- Section 24.3.4, "Including Oracle SES Results in WebCenter Search Results"
- Section 24.3.5, "Using Search Service APIs"
- Section 24.3.6, "Building Adapters for the Search Service"
- Section 24.3.7, "Troubleshooting the Search Service"

24.3.1 Adding the Search Task Flow

To add the **Search** task flow to your WebCenter application, follow the instructions provided for the **Search Toolbar** task flow in Section 24.2.2, "Adding the Search Service at Design Time," but instead, drag and drop the **Search** task flow onto the page.

In the Design view of the page, you should see something similar to Figure 24–1.
To run the page with the Search view:

1. In the Application Navigator, right-click the page and select Run.
   
   As long as the application has been ADF-secured, you should see a login page. It may take a few moments for the operation to complete.

2. If you configured ADF security, then enter the user name and password that you provided and click Submit.

3. You should see your page with the Search main view (Figure 24–2).

4. Enter some search criteria in the field and click the Search icon.
You should see the results displayed on the page. At this time, you get few, if any, results because the application is not populated with much besides the Search service itself.

### 24.3.2 Adding the Search - Saved Searches Task Flow

You can add the **Search - Saved Searches** task flow to provide a simple launch pad for running saved searches within the application. To add this task flow to your WebCenter application, follow the instructions provided for the **Search Toolbar** task flow in Section 24.2.2, "Adding the Search Service at Design Time," but instead, drag and drop the **Search - Saved Searches** task flow onto the page.

In the Design view of the page, you should see something similar to **Figure 24–3**.

![Figure 24–3 Search - Saved Searches Task Flow](image)

The task flow renders links that, when clicked, run a saved search.

### 24.3.3 Adding the Search Preferences Task Flow

By default, WebCenter searches all WebCenter services in your application. However, the **Search Preferences** task flow enables users to control that selection. Services that are not selected in preferences are not searched.

The **Search Preferences** task flow also provides a means of selecting and ordering the columns available in the results returned in the **Search** task flow.

To add the **Search Preferences** task flow to your WebCenter application, follow the instructions provided for the **Search Toolbar** task flow in Section 24.2.2, "Adding the Search Service at Design Time," but instead, drag and drop the **Search Preferences** task flow onto the page.

In the Design view of the page, you should see something similar to **Figure 24–4**.

![Figure 24–4 Search Preferences Task Flow](image)

For information about running the Search Preferences task flow at runtime, see Section 24.1.2, "What Happens at Runtime."
24.3.4 Including Oracle SES Results in WebCenter Search Results

If you have an Oracle Secure Enterprise Search (SES) instance for searching other repositories outside of Oracle WebCenter, then you can integrate it with your application. SES results appear in the same result set as WebCenter search results.

For secure searches, your WebCenter application and your Oracle SES instance must use the same identity management system.

This section describes how to include Oracle SES results in WebCenter searches and how to validate this configuration. It contains the following subsections:

- Section 24.3.4.1, "Including Oracle SES Results in WebCenter Searches"
- Section 24.3.4.2, "Validating the Oracle SES Connection"

24.3.4.1 Including Oracle SES Results in WebCenter Searches

To include Oracle SES results:

1. Add the search toolbar to a page, following Section 24.2.2, "Adding the Search Service at Design Time."

2. In the Application Navigator, expand Application Resources.

3. Right-click the Connections node, and select Oracle Secure Enterprise Search.

4. Enter values for the fields in the dialog.

   For example:

   a. In Oracle WebCenter, 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.

   b. For Connection Name, enter a meaningful name, for example, SES_instance.

   c. For SOAP URL, enter the URL for the Oracle SES Web Services endpoint, for example:

      http://host:port/search/query/OracleSearch

   d. Under Proxy Login, for App User Name, enter the trusted entity defined for this application and the Oracle SES instance; for example, wpadmin.

      You can find (or create) the trusted entity on the Global Settings - Federation Trusted Entities page of Oracle SES.

   e. For App Password, enter the password of the Federation Trusted Entities user.

   f. Select the box for Set as default connection for the Search Service.

---

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.
This ensures that your application uses that connection to access Oracle SES. The Search service uses only one Oracle SES connection.

**Note:** After you create a connection as the default connection, you cannot edit it so that it is *not* the default. To use a different default connection, you must create a new connection and mark that as the default connection.

**g.** Under **Testing**, for **User name**, enter the name of a user present in both the Oracle Identity Management server configured for your WebCenter application and the Oracle Identity Management server configured for Oracle SES.

You should see something similar to **Figure 24–5**.

**Figure 24–5 Create Oracle SES Connection Dialog**

![Create Oracle SES Connection Dialog](image)

5. Click **Test Connection**.

6. 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 24–6**.
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 24–3.

**Example 24–3  Oracle SES Connection in `adf-config.xml`**

```xml
<ses-properties>
  <connection>GenericSesConnection</connection>
  <data-group>oracle-sites</data-group>
</ses-properties>
```

The Oracle Secure Enterprise Search connection wizard does not provide a means of setting the `data-group` value in `adf-config.xml`. After the connection is established, you must set it manually. Check the `data-group` on your Oracle SES instance to set to the correct value.

You can also set the `data-group` value to be any a data source group created in your Oracle SES instance. Data 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 data source groups to make only a portion of your corporate data available to a WebCenter search. For more information on creating data groups, see the *Oracle SES Administrator’s Guide*.

---

**Note:** While you can set up the connections to back-end servers at design time in Oracle JDeveloper, you can later add, delete, or modify connections in your deployed environment using Enterprise Manager Fusion Middleware Control. For more information, see *Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter*.

---

### 24.3.4.2 Validating the Oracle SES Connection

To validate the inclusion of Oracle SES results in WebCenter 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**.
   
   It may take a few moments for the page to load. When it completes, you should see your search toolbar.

   *Figure 24–7* shows example results for the search term *webcenter*. 

---

---
For more information on Oracle SES, see the Oracle Secure Enterprise Search documentation on the Oracle Fusion Middleware documentation library (in the WebCenter product area).

24.3.5 Using Search Service APIs

Custom components can implement Search service APIs to expose their content to a WebCenter search. For more information, see Oracle Fusion Middleware Java API Reference for Oracle WebCenter.

24.3.6 Building Adapters for the Search Service

By default, the Search service searches any WebCenter service in your application. In some cases, you may want to add other sources to your search.

You can extend WebCenter search through search adapters. WebCenter framework automatically discovers these search adapters and consolidates them in formulating search results in your custom WebCenter application. Subsequently, when you expose your custom component to the Search service in your application, it is included in the federated search.

This section describes how to extend WebCenter search through search adapters. It contains the following subsections:

- Section 24.3.6.1, "How to Add a Search Source"
- Section 24.3.6.2, "How to Register a Custom Adapter"
- Section 24.3.6.3, "Search Adapter Attributes"
- Section 24.3.6.4, "What Happens at Runtime"

24.3.6.1 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 the Search service.
When performing a search, the search framework calls
`QueryManager.createRowQueryHandler` with a query object and a
`List<QName>` that describes the columns sought. In return, the framework receives a
`QueryHandler<Row>`.

A `QueryHandler` can be a `QueryFederator` or a `QueryExecutor`. A
`QueryFederator` is just a collection of other `QueryHandlers`. With the
`QueryFederator`, the framework can get a list of the child `QueryHandlers` and
keep recursing until all `QueryExecutors` are found.

The framework calls the `execute` method on the `QueryExecutor` to get a
`QueryResult<Row>`, which is an extension of `java.util.Iterator<Row>` that
iterates and retrieves rows of results.

One possible implementation would be to create these classes:

- `SampleQueryManager` ...
- `SampleRowQueryExecutor` ...
- `SampleRowQueryResult` ...
- `SampleRow` ...

To add a new search source with a similar query manager:

1. In Oracle JDeveloper, open the application in which to add the search source.
2. Confirm that the WebCenter Common library is included in your project:
   a. Right-click your `ViewController` project and select `Project Properties` and
      then `Libraries and Classpath`.
   b. If the `WebCenter Common` library is not included, as shown in Figure 24–8,
      then click `Add Library` and select it.
c. Click OK.

3. Create a class called SampleQueryManager.java (Example 24–4).

Example 24–4   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 24–5).
Example 24–5  SampleRowQueryExecutor.java

```
package mycompany.myproduct.myapp.mycomponent.query;
import java.text.DateFormat;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Iterator;
import java.util.List;
import java.util.Map;
import static oracle.webcenter.search.AttributeConstants.DCMI_EXTENT;
import static oracle.webcenter.search.AttributeConstants.DCMI_MODIFIED;
import static oracle.webcenter.search.AttributeConstants.DCMI_CREATOR;
import static oracle.webcenter.search.AttributeConstants.DCMI_TITLE;
import static oracle.webcenter.search.AttributeConstants.DCMI_URI;
import static oracle.webcenter.search.AttributeConstants.DCMI_IDENTIFIER;
import static oracle.webcenter.search.AttributeConstants.WPSAPI_ICON_PATH;
import oracle.webcenter.search.QName;
import oracle.webcenter.search.Query;
import oracle.webcenter.search.QueryExecutor;
import oracle.webcenter.search.QueryHandler;
import oracle.webcenter.search.QueryResult;
import oracle.webcenter.search.Row;
public class SampleRowQueryExecutor
    implements QueryExecutor<Row>
{
    private static final int PRESET_ESTIMATED_RESULT_COUNT = 10;
    private static final int PRESET_VALUES_BATCH_SIZE = 5;
    private static final int MAX_RESULT_LIMIT = 100;
    private int m_resultStart;
    private int m_resultLimit;
    private Query m_query;
    private List<QName> m_columns;
    private Map<String, String> m_properties;
    public SampleRowQueryExecutor(Query query, List<QName> columns)
    {
        m_query = query;
        m_columns = columns;
        m_properties = new HashMap<String, String>();
        // In actuality you may want to make sure this string is translateable
        m_properties.put(QueryHandler.TITLE, "Sample");
    }
    /**
     * Create a number of 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));
        }
    }
```
public QueryResult<Row> execute()
{
    List<Row> rows = new ArrayList<Row>();
    // We want 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
 * so that we 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 24–6).

Example 24–6  SampleRowQueryResult.java
package mycompany.myproduct.myapp.mycomponent.query;
import java.util.Iterator;
import oracle.webcenter.search.util.WrapperQueryResult;
import oracle.webcenter.search.Row;
public class SampleRowQueryResult extends WrapperQueryResult<Row>
{
    public SampleRowQueryResult(Iterator<Row> rows)
    {
        super(rows);
    }
}

6. Create a class called SampleRow.java (Example 24–7).
Example 24–7 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_CREATOR;
import static oracle.webcenter.search.AttributeConstants.DCMI_TITLE;
import static oracle.webcenter.search.AttributeConstants.DCMI_URI;
import static oracle.webcenter.search.AttributeConstants.WPSAPI_ICON_PATH;
public class SampleRow implements Row
{
    private Map<QName, Object> m_storage = new HashMap<QName, Object>();
    private DateFormat m_dateFormat = new SimpleDateFormat("yyyy-MM-dd");
    SampleRow(String id, String title,
        String creator, String lastModified,
        int size)
    {
        m_storage.put(DCMI_IDENTIFIER, id);
        // Only if you have an external URL to go to
        // m_storage.put(DCMI_URI, "http://my.external.url");
        m_storage.put(DCMI_TITLE, title);
        m_storage.put(DCMI_CREATOR, creator);
        // The below path points to a path accessible in the classpath to an icon
        m_storage.put(WPSAPI_ICON_PATH, "/adf/webcenter/search_qualifier.png");
        try
        {
            Date date = m_dateFormat.parse(lastModified);
            Calendar cal = Calendar.getInstance();
            cal.setTime(date);
            m_storage.put(DCMI_MODIFIED, cal);
        }
        catch (ParseException e)
        {
            e.printStackTrace();
        }
        m_storage.put(DCMI_EXTENT, new Long(size));
    }
    public Object getObject(QName columnName)
    {
        return m_storage.get(columnName);
    }
    public Iterator<QName> getColumns()
    {
        return m_storage.keySet().iterator();
    }
}
```
24.3.6.2 How to Register a Custom Adapter

To register your query manager with the Search service, you must edit the service-definition.xml file. The SampleQueryManager line exposes the custom adapter's query manager implementation to the Search service so that the custom adapter is included with any search user interface.

The entry should be similar to Example 24–8.

**Example 24–8 Sample Query Manager Entry in service-definition.xml**

```xml
<service-definition id="mycompany.myproduct.myapp.mycomponent"
   version="11.1.1.0.0">
  <resource-view taskFlowId="/somepath/somefile.xml#someId"
      authorizerClass="some.service.id.Authorizer1"/>
  <name>Custom Component 1</name>
  <description>Description for Custom Component 1</description>
  <search-definition xmlns="http://xmlns.oracle.com/webcenter/search"
     id="mycompany.myproduct.myapp.mycomponent.query" version="11.1.1.0.0">
    <query-manager-class>
      mycompany.myproduct.myapp.mycomponent.query.SampleQueryManager
    </query-manager-class>
  </search-definition>
</service-definition>
```

For more information about the resource-view tag (and Oracle WebCenter’s Resource Action Handling framework), see Section 4.2, “Extending Your Application with Custom Components.”

24.3.6.3 Search Adapter Attributes

This section describes the attributes available for searching a service. It contains the following subsections:

- Section 24.3.6.3.1, "Searchable Attributes"
- Section 24.3.6.3.2, "Keywords"
- Section 24.3.6.3.3, "Date Condition"
- Section 24.3.6.3.4, "Complex Date Condition"
- Section 24.3.6.3.5, "Person Condition"
- Section 24.3.6.3.6, "Complex Person Condition"
- Section 24.3.6.3.7, "Access"
- Section 24.3.6.3.8, "Selectable Attributes"

24.3.6.3.1 Searchable Attributes

The WebCenter 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 24.3.6.3.8, "Selectable Attributes.”

24.3.6.3.2 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. The WebCenter Search service does not tokenize before the call comes to the service, but here are some guidelines:
If two or more tokens are enclosed within quotes, then they should be interpreted as one word.

Unless the OR operator is explicitly set, it is assumed that the default conjunction among the words in the keywords field is AND.

### 24.3.6.3 Date Condition
For the Search service to support filtering of query results by last modified date, the service implementation of the WPSAPI must support the use of `oracle.webcenter.search.AttributePredicate`, accessible from the query object, that uses `oracle.webcenter.search.AttributeConstants.DCMI_MODIFIED`. The supported comparators must include the following:

- `oracle.webcenter.search.ComparatorConstants.GREATER_THAN`
- `oracle.webcenter.search.ComparatorConstants.GREATER_THAN_OR_EQUALS`

They may also include the following:

- `oracle.webcenter.search.ComparatorConstants.EQUALS`
- `oracle.webcenter.search.ComparatorConstants.NOT_EQUALS`
- `oracle.webcenter.search.ComparatorConstants.LESS_THAN`
- `oracle.webcenter.search.ComparatorConstants.LESS_THAN_OR_EQUALS`

The literals for comparison should be of type `java.util.Calendar`.

### 24.3.6.4 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.

### 24.3.6.5 Person Condition
For the Search service to support filtering of query results by person, the service implementation of the WPSAPI must support the use of `oracle.webcenter.search.AttributePredicate`, accessible from the query object, that makes use of the `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`.

### 24.3.6.6 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.
24.3.6.3.7 Access  The Search service (as a caller) expects that the query’s `getPredicate()` method likely returns an `oracle.webcenter.search.ComplexPredicate` that joins an `oracle.webcenter.search.TextPredicate` holding the keyword, and an `oracle.webcenter.search.ComplexPredicate` holding the date condition, and an `oracle.webcenter.search.AttributePredicate` holding the person condition.

24.3.6.3.8 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 service user interface can display the title column, if provided. It is strongly recommended that the title column comes with the QName `AttributeConstants.DCMI_TITLE`.

- **Resource ID** (Required)
  
  For resources to communicate a unique identifier that allows the Search service to render a link to navigate to the resource view (or any declared views) of the service, the column with the name `oracle.webcenter.search.AttributeConstants.DCMI_IDENTIFIER` must be returned as one of the columns accessible from a row within the `QueryResult<Row>`.

  The Search service extracts the `DCMI_IDENTIFIER` column and feeds it into the resource viewer as the value of the `resourceId` parameter when the link is clicked.

  If your resource view (or any declared views) requires multiple columns in the primary key, then try to bundle all of them into a composite string value for the `oracle.webcenter.search.AttributeConstants.DCMI_IDENTIFIER` column. The Search service treats that opaquely and passes it to your resource view, within which it may deconstruct the string into the various parts of the primary key.

- **Resource Type**
  
  For each resource returned, the Search service user interface can display the type column, if provided. The type column comes with the QName `AttributeConstants.DCMI_TYPE`. If the `DCMI_TYPE` column is not found, then the value of the `DCMI_FORMAT` column is used in its place. In your resource viewer, you can make use of the resource type to aid in the rendering of a resource.

- **Resource URL** (in place of Resource ID)
  
  In the absence of an `oracle.webcenter.search.AttributeConstants.DCMI_IDENTIFIER` column, the Search service constructs the link from the value of the column specified by `oracle.webcenter.search.AttributeConstants.DCMI_URI`. With absolute URLs, the `DCMI_URI` can launch a link from the search results to the URL.

  This allows for services, such as the Documents service and the Page service, to invoke the browser plug-in to render various results link targets, instead of invoking a resource view.

- **Last Modified Date**
For each resource returned, the Search service user interface can display the last modified date column, if provided. To mirror the searchable attribute of the last modified date, it is important that users can see the last modified date as a returned column in the search result.

For most users, the last modified date is more relevant than the creation date. To have better consistency between the searchable attributes and the selectable attributes (so that users know what date to search with after seeing some values in the columns), it is strongly recommended that service authors return `AttributeConstants.DCMI_MODIFIED` rather than `AttributeConstants.DCMI_CREATED`. The last modified date is of type `java.util.Calendar`.

**Creator**

For each resource returned, the Search service user interface can display the creator and last modified by column, if provided. The search user interface combines them in the same column as a list of contributors for the found resource. The QNames to use are `AttributeConstants.DCMI_CREATOR` and `AttributeConstants.WPSAPI_MODIFIED`, respectively. The creator is of type `java.lang.String`.

**Icon**

For each resource returned, the Search service user interface can display the icon column, if provided. The icon column comes with the QName `AttributeConstants.WPSAPI_ICON_PATH` and corresponds to a valid path to an icon file that is accessible in the class path. The icon path is of type `java.lang.String`.

**Size**

For each resource returned, the Search service user interface can display the size column, if provided. The size column comes with the QName `AttributeConstants.DCMI_EXTENT` and corresponds to the size of the resource found. The size is of type `java.lang.Number`.

### 24.3.6.4 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.

The Search service invokes a resource viewer using the Resource Action Handling framework. For information about registering a resource viewer to allow custom components to be found using Search, see Section 4.2, "Extending Your Application with Custom Components."

### 24.3.7 Troubleshooting the Search Service

This section describes common problems and solutions for the Search service.

**Problem**

There is a set timeout for querying each service. If a particular service does not return search results in the allotted time, then it times out. The Announcements and Discussions services are susceptible to timeouts. (In WebCenter Spaces, the Page service and Group Spaces could have timeouts from search execution.)
Solution

To improve performance at design time, you can increase the values (in milliseconds) of timeoutMs and prepare TimeoutMs in adf-config.xml. The following is the relevant fragment of adf-config.xml, found in the Application Resources panel in the ADF META-INF folder:

<execution-properties timeoutMs="3000" prepare TimeoutMs="1000"/>

At runtime, you can configure timeouts using WLST or Enterprise Manager.
Part VI contains the following chapters:

- Chapter 25, "Integrating the Recent Activities Service"
- Chapter 26, "Integrating the Worklist Service"
This chapter describes how to integrate the Recent Activities service into your custom WebCenter application. This service enables you to display the most recent changes to the services in your application.

This chapter includes the following sections:

- Section 25.1, "Introduction to the Recent Activities Service"
- Section 25.2, "Basic Configuration for the Recent Activities Service"
- Section 25.3, "Advanced Information for the Recent Activities Service"

For information about using the Recent Activities service, see Chapter "Working with the Recent Activities Service" in Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

### 25.1 Introduction to the Recent Activities Service

Oracle WebCenter Services are preconfigured to work with the Recent Activities service. Recent Activities enable users to view the most recent changes to the services in your application. For example, once you add the Documents service, it automatically produces the information that the Recent Activities service will use to display the most recent changes to the document library. This service is useful to your users who want a quick and easy way to view any additions or changes to a particular area of the application.

The Recent Activities service can track information from the following WebCenter Services:

- Documents added by the Documents service
- Announcements created by the Announcements service
- Discussion Forums created by the Discussions service
- Pages created or modified by the Page service

For more information about these services, see Chapter 16, "Integrating the Announcements Service," Chapter 17, "Integrating the Discussions Service," Chapter 14, "Integrating the Documents Service," and Chapter 6, "Enabling Runtime Creation and Management of Pages."

This section contains the following subsections:

- Section 25.1.1, "Understanding the Recent Activities Service"
- Section 25.1.2, "What Happens at Runtime"
25.1.1 Understanding the Recent Activities Service

The Recent Activities service provides a summary view of recent changes to a variety of services. You can specify the range of time to consider recent by selecting a time range from a list at the top of the task flow. Recorded activities include additions or revisions of pages, documents, discussion forums, and the like.

By default, the Recent Activities service works with the Documents, Announcements, Discussions, and Page services. You can narrow the services that the Recent Activities service tracks. For more information about this setting, see the next section, Section 25.3, "Advanced Information for the Recent Activities Service." You can also learn more about using Oracle WebCenter services in Section 1.1.4, "Introduction to Oracle WebCenter Services."

Note: You can see the Recent Activities service in action in the sample application, as described in Chapter 2, "Introduction to the WebCenter Sample Application."

The Recent Activities service compares somewhat to Activity Stream (for more information, see Section 20.1.1.1, "Activity Stream"). Both track the application activities of integrated services, though Activity Stream tracks a broader range of services. For example, Recent Activities tracks the Documents, Announcements, Discussions, and Page services. Activity Stream tracks these services as well as People Connections and Wikis and Blogs. Recent Activities tracks activities no matter who performs the action. Activity Stream tracks activities performed by a user's connections and includes information about who performed the activity. Recent Activities does not include names.

The basic difference between these two services can be summarized as follows: Recent Activities provides an overview of what is happening in an application. Activity Stream provides an overview of what is happening with a user's connections.

25.1.2 What Happens at Runtime

At runtime, the Recent Activities task flow fetches latest information about the services that are configured and running in your application. Figure 25–1 shows an example of the Recent Activities task flow at runtime, and displays the following activities that have occurred in the application since yesterday:

- one document has been uploaded or added using the Documents service
- five discussion threads have been modified or added using the Discussions service
- one announcement has been modified or added using the Announcements service

The time range choices (Last 30 days, Today, Since Yesterday, and so on) display based on the default values set for the Recent Activities Time Range parameter values at design time (see Table 25–1).

The Recent Activities service uses the search capabilities of WebCenter to query activities for specified periods. Search capabilities are implemented across the following WebCenter services: Documents, Announcements, Discussions, and Pages. These services, if configured in an application, respond to queries of the Recent Activities service and return items pertinent to the search criteria. Responses to queries (search results) are always based on the search predicate, DCMI_MODIFIED > [start time]. Here start time is one of the time periods that you configured in the Edit Task Flow Binding dialog at design time. You can select a select a time period in the Show dropdown list at runtime.
25.2 Basic Configuration for the Recent Activities Service

This section describes how to add the Recent Activities service to your custom WebCenter application.

This section contains the following subsections:

- Section 25.2.1, "Setting up Connections for the Recent Activities Service"
- Section 25.2.2, "Adding the Recent Activities Service at Design Time"
- Section 25.2.3, "Setting Security for the Recent Activities Service"

25.2.1 Setting up Connections for the Recent Activities Service

You do not need to set up specific connections for the Recent Activities service. You only need to ensure that the service(s) you are tracking with the Recent Activities service is configured. For more information about setting up connections for other services, see the individual services chapters in this guide.

25.2.2 Adding the Recent Activities Service at Design Time

To use the Recent Activities service, you must add its task flow to your application. Ensure that you have at least one of the services it tracks in your application. There is a single Recent Activities task flow available in the WebCenter Services Catalog, which you can add to your application to display information about services in your application to your users.

The Recent Activities service automatically picks up other services in your application and looks for recent activities within these services. Before running your application containing the Recent Activities service, however, you may want to ensure your application contains at least one of the services you wish to monitor.

To add the Recent Activities task flow to your application:

1. Follow the steps in Section 4.1.1, "How to Prepare Your Application to Consume Services" to implement security and create a customizable page in your custom WebCenter application.

2. Open the source of the customizable page.

3. Ensure that your application includes at least one of the services that the Recent Activities service tracks.
4. Create a new external application by right-clicking the ViewController project in your application, then choosing **New**.

5. In the New Gallery, under General, select **External Applications**, then select **External Application** from the Items list.

6. Click **OK**.

7. In the **Application Name** field, enter a name for your application, such as **MyExternalApplication**, then click **Next**.

8. On the **General** page, leave the **Login URL** field empty.

9. In the **User Name/ID Field Name** field, enter a user name. The user name should match the login credentials of the servers you are tracking. For example, if you are tracking the Documents service with Oracle Content Server, the login credentials must match those of the user on Oracle Content Server.

10. In the **Password Field Name** field, enter a password.

11. Under Authentication Details, choose **POST**, then click **Next**.

12. On the **Additional Fields** page, click the **Add Field** button.

13. For the Field Name, enter **Account**.

14. Select the **Display to User** checkbox.

15. Click **Finish**.

16. In the Resource Palette, open the **WebCenter Services Catalog**, then open the **Task Flows** folder.

---

**Note:** If you run an application that does not contain one of these services, the Recent Activities service displays an error saying *Warning: Unable to obtain recent activity data.*
17. Click Recent Activities and drag it onto your page after the `<af:form>` tag, and choose Region.

**Figure 25–3  Edit Task Flow Binding Dialog**

In the Edit Task Flow Binding dialog shown in Figure 25–4, you can set the values for the time range that displays at runtime. This time range controls what activities display, as you will see in the next section. If you leave these parameters empty, the Recent Activities service will use the default values of the service.

Table 25–1 describes the possible values for these parameters.

**Table 25–1 Recent Activities Time Range Parameter Values**

<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</td>
</tr>
<tr>
<td></td>
<td>Possible values: Today, Yesterday, or a Time in Minutes</td>
</tr>
<tr>
<td>timePeriodMedium</td>
<td>Default value: Since Yesterday</td>
</tr>
<tr>
<td></td>
<td>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.)</td>
</tr>
<tr>
<td></td>
<td>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.)</td>
</tr>
<tr>
<td></td>
<td>Possible values: Today, Yesterday, or a Time in Minutes</td>
</tr>
</tbody>
</table>
Figure 25–4  Example of the Edit Task Flow Binding Dialog for the Recent Activities Service

18. Click OK. The binding displays on your page, as shown in Figure 25–5.

Figure 25–5  The Recent Activities Task Flow on a Page in the Design View

19. Save your project and run your page to the browser.

Note: When you add the Recent Activities task flow to your page, you will see a task flow parameter called groupSpace. The default value of this parameter is null, which tells the task flow to search the entire application for recent activities.
25.2.3 Setting Security for the Recent Activities Service

The Recent Activities service uses the security applied to the underlying services being analyzed. So, the information returned by the services only contains content for which the current user has at least View privileges. For example, a user who does not have View privileges to view content in a document library will not see recently added documents in the Recent Activities view. Similarly, a public user will only see activities returned on services exposed to the public, and do not require authentication to view.

25.3 Advanced Information for the Recent Activities Service

This section explains that in addition to the task flow parameters, you can further refine the behavior of the Recent Activities service. It also describes how you can obtain RSS news feed for your Recent Activities service.

This section contains the following subsections:
- Section 25.3.1, "Refining the Behavior of the Recent Activities Service"
- Section 25.3.2, "Obtaining RSS News Feed URL for the Recent Activities Service"

25.3.1 Refining the Behavior of the Recent Activities Service

You can change a variety of the Recent Activities service parameters after you have added it to your application. When you add the task flow to your page, you also automatically add the following entry to the adf-config.xml file, located in the Application Resources panel, under Descriptors, in the ADF META-INF folder:

```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>
```

You can change the settings described in Table 25–2 in this entry.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>numServiceRows</td>
<td>Default value: 25, The maximum number of rows displayed for a single service. If there are more recent activity results than this, the most recent ones up to this number are returned. For example, if there are 32 recently modified documents and this setting is 25 then only the most recent 25 will show up in the results.</td>
</tr>
<tr>
<td>shortPeriod</td>
<td>Default value: Today, See Table 25–1. This setting is the same as the timePeriodShort task flow parameter, and is used when the task flow parameter is not specified.</td>
</tr>
<tr>
<td>mediumPeriod</td>
<td>Default value: Yesterday, See Table 25–1. This setting is the same as the timePeriodMedium task flow parameter and is used when the task flow parameter is not specified.</td>
</tr>
</tbody>
</table>
### 25.3.2 Obtaining RSS News Feed URL for the Recent Activities Service

You can expose Oracle WebCenter Spaces functionality in a custom WebCenter application. Your custom application users can find out what is happening in a specific group space through RSS news feeds.

Configure RSS news feeds for the Recent Activities service to enable users to view group space recent activities from within a custom application. To obtain the group space RSS news feed URL for the Recent Activities service, use either of the following WebCenter Spaces APIs:

- `getServiceRssFeedURL`
- `GetServiceRssFeedURLbyGUID`

To obtain an RSS feed URL, you must identify the group space (by name or GUID) and specify the service required (by service ID). The service ID for the Recent Activities service is `GroupSpaceWSClient.RECENT_ACTIVITY_SERVICE_ID`. For information about how to use these APIs, see Section 27.2.5.3.8, "Retrieving RSS Feed URLs for Group Space Services.”

---

**Table 25–2 (Cont.) Recent Activities Settings in the adf-config.xml File**

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>longPeriod</code></td>
<td>Default value: 10080 (Last 7 Days)</td>
</tr>
<tr>
<td></td>
<td>See Table 25–1. This setting is the same as the <code>timePeriodLong</code> task flow parameter and is used when the task flow parameter is not specified.</td>
</tr>
<tr>
<td><code>longestPeriod</code></td>
<td>Default value: 43200 (Last 30 Days)</td>
</tr>
<tr>
<td></td>
<td>See Table 25–1. This setting is the same as the <code>timePeriodLongest</code> task flow parameter and is used when the task flow parameter is not specified.</td>
</tr>
<tr>
<td><code>services-filter</code></td>
<td>Default value: null</td>
</tr>
<tr>
<td></td>
<td>Use this tag to filter out services you do not wish to track with this service. The following example omits any changes to the Page service from displaying in the Recent Activities service:</td>
</tr>
<tr>
<td></td>
<td><code>&lt;services-filter&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;exclude&gt;oracle.webcenter.page&lt;/exclude&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;/services-filter&gt;</code></td>
</tr>
<tr>
<td></td>
<td>By default, no services are omitted.</td>
</tr>
</tbody>
</table>

---

25-8 Oracle Fusion Middleware Developer's Guide for Oracle WebCenter
This chapter describes how to integrate the Worklist service into your application. Worklists enable users to view and take action on all tasks and notifications from a BPEL (Business Process Execution Language) server all in one place.

This chapter includes the following sections:

- Section 26.1, "Introduction to the Worklist Service"
- Section 26.2, "Basic Configuration for the Worklist Service"
- Section 26.3, "Advanced Information for the Worklist Service"

For information about managing and using the Worklist service, see:

- Chapter "Managing the Worklist Service" in the Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter
- Chapter "Working with the Worklist Service" in the Oracle Fusion Middleware User's Guide for Oracle WebCenter

### 26.1 Introduction to the Worklist Service

The Worklist service enables you to show Business Process Execution Language (BPEL) Worklist items assigned to the currently authenticated user. The BPEL Worklist items are open BPEL tasks from one or more BPEL Worklist Repositories to which your application is connected. The Worklist displays BPEL Worklist items that are a result of a task invoked as part of a BPEL Workflow process, or are a result of a message being sent to the Worklist channel on the Oracle User Messaging Service.

This section contains the following subsections:

- Section 26.1.1, "Understanding Worklists"
- Section 26.1.2, "Requirements for the Worklist Service"
- Section 26.1.3, "What Happens at Runtime"

### 26.1.1 Understanding Worklists

The Worklist service provides a personal, at-a-glance view of business processes that require your users' attention. These can include a request for document review or other types of business processes that come directly from your enterprise applications.

Worklist items come from a variety of sources. Some Worklist items are kicked off by events that are associated with an externally defined workflow. A workflow maps the route an item follows once an event kicks off. This type of workflow is defined in a Worklist BPEL server, such as Oracle BPM Worklist.
Messages, alerts, and notifications might also come from the User Messaging Service (UMS). The Worklist task flow includes a control for accessing messaging preferences on this server. Use these controls to specify the channels over which to receive Oracle User Messaging Service messages and to define messaging filters.

See Also: Oracle Fusion Middleware User’s Guide for Oracle WebCenter for more information about the service at runtime.

26.1.2 Requirements for the Worklist Service

To use the Worklist service, you must have a BPEL server installed. For information on installing BPEL for the Worklist service, see the “Back-End Requirements for the Worklist Service” in the Oracle Fusion Middleware Installation Guide for Oracle WebCenter.

For the Worklist to properly function, it must reside on a secured page, as the service uses the currently authenticated user to access the BPEL server. If the page or application that contains the Worklist service is not secured, you will encounter exception errors.

26.1.3 What Happens at Runtime

At runtime, the Worklist task flow fetches latest Worklist workflow items from BPEL workflows configured in the BPEL server. For more information, see the chapter “Working with the Worklist Service” in Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

To get a list of the items assigned to a user, the Worklist service uses the WebService access function of the public BPM client API. The Worklist service uses SAML authentication to authorize the logged-in user to fetch the list of latest items from the BPEL repository. The Worklist service fetches 25 most recent items for each Worklist connection, and determines the order of these items by using a predicate on the TaskQueryService API. For more information, see "Building a Custom Worklist Client" in Oracle Fusion Middleware Developer’s Guide for Oracle SOA Suite.

Figure 26–1 shows the Worklists service at runtime.
26.2 Basic Configuration for the Worklist Service

This section describes how to set up a connection for the Worklist service and add the task flow to your custom WebCenter application.

This section contains the following subsections:

- Section 26.2.1, "Setting up Connections for the Worklist Service"
- Section 26.2.2, "Adding the Worklist Service at Design Time"
- Section 26.2.3, "Setting Security for the Worklist Service"

26.2.1 Setting up Connections for the Worklist Service

To use the Worklist service, you must establish a connection to one or more BPEL servers. This section describes these connections and how to create them in a custom WebCenter application.
26.2.1.1 Worklist Service Connections
The Worklist service relies on a BPEL server. Once you create a connection using the Create Worklist Connection dialog, the connection is registered in your application's connections.xml and referenced in your application's adf-config.xml. You can see the connections to your BPEL Worklist Repositories in the Application Resources pane.

Note: While you can set up the connections to back-end servers at design time in Oracle JDeveloper, you can later add, delete, or modify connections in your deployed environment using Enterprise Manager Fusion Middleware Control. For more information, see the Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter.

26.2.1.2 How to Set Up Connections for the Worklist Service
The Worklist service provides a task flow that you can add to your application at design time or at runtime. Before you can run the task flow, you create a connection from your application to a BPEL server.

Note: The application you use must be prepared for service consumption. For information about doing so, see Chapter 4, "Preparing Your Application for Oracle WebCenter Services."

To set up the connection for the Worklist service:

1. In the Application Resources panel for your application, right-click Connections, then choose New Connection> Worklist.

2. In the Create Worklist Connection dialog, choose whether to create the connection in the Application Resources (only the current application can use this connection) or IDE Connections (other applications you create can use the same connection).

3. In the Connection Name field, enter a name for your connection. This name will be used as the display name of the Worklist server.

Note: At runtime, when users view their worklist items, they can choose to group the items by "Worklist Server." The Worklist Server name used is the name you enter in the Connection Name field.

4. In the URL field, enter the location of your BPEL/SOA server, such as http://bpel.example.com. This is the URL for the managed server running the SOA processes.

5. Select the SAML Token Policy URI to use for this server (Figure 26–2).

SAML (Security Assertion Markup Language) is an XML-based standard for passing security tokens defining authentication and authorization rights. An attesting entity (that already has trust relationship with the receiver) vouches for the verification of the subject by method called sender-vouches. Options available are:

- SAML Token Client Policy (oracle/wss10_saml_token_client_policy) - Select to verify your basic configuration without any additional security. This is the default setting.
SAML Token With Message Client Policy (oracle/wss10_saml_token_with_message_protection_client_policy) - Select to increase the security using SAML-based BPEL Web Services. If selected, you must configure keys stores both in your custom WebCenter application and in the BPEL application. For more information about this configuration, see the “Configuring Policies” in the Oracle Fusion Middleware Security and Administrator’s Guide for Web Services.

Figure 26–2  Create Worklist Connection Dialog

6. Click Test Connection to validate the URL.
7. Click OK.

The new connection displays in the Application Resources panel under Connections.

26.2.2 Adding the Worklist Service at Design Time

This section describes the Worklist task flow and how to add it to your application.

26.2.2.1 Worklist Service Task Flows

There is one task flow associated with the Worklist service, called "Worklist." You can find this task flow in the WebCenter Services Catalog in the Resource Palette.

26.2.2.2 How to Add the Worklist Service to your Application

You can add the Worklist service to your application before or after you have created your connection. However, you must set up your connection before you run the service.

To add the Worklist task flow to your application:

1. Follow the steps in Section 4.1.1, "How to Prepare Your Application to Consume Services" to implement security and create a customizable page in your application.
2. Open the page where you wish to add the Worklist task flow.

3. In the Resource Palette, in the My Catalogs pane, open the WebCenter 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 26–3.

5. Save your project, then run your page to a browser. The Worklist service displays in your browser.

26.2.3 Setting Security for the Worklist Service

The Worklist service displays the tasks for the currently authenticated user. So, for users to store and retrieve Worklist tasks on the BPEL server from your application, you must set up security on the application (as described in Section 4.1.1.1, "Implementing Security for Services"). The user names need to either exist in a shared user directory (LDAP), or set up similarly (same user name) on both the custom WebCenter application and the BPEL Server. For example, if the user rsmith needs to use the Worklist service in your custom WebCenter application to store and retrieve tasks from the BPEL server, you must ensure that the user rsmith exists on both the BPEL server and within your application.

To enable users to perform actions on assigned Worklist tasks, you can configure SSO. If you do not configure SSO, the users will be prompted to log into the Worklist Task Details page on the BPEL server. For more information, see the Oracle Fusion Middleware Security Guide.

Only authenticated users can view Worklist tasks on the BPEL server to which they have permissions to view. If the application end user does not supply appropriate credentials, or if the page containing the Worklist service is not secured, she will not see any Worklist items stored on the BPEL server and exception errors may occur.

26.3 Advanced Information for the Worklist Service

The Worklist service collates all the Worklist items for the authenticated user from all connections in your application defined for the Worklist service. You can use the create connection method described in Section 26.2.1.2, "How to Set Up Connections for the Worklist Service" to create additional connections, or right-click the connection name in the Application Resources list and choose Delete to remove a connection.
Part VII contains the following chapters:

- Chapter 27, "Integrating with Oracle WebCenter Spaces"
- Chapter 28, "Using Oracle WebCenter REST APIs"
This chapter describes how to expose Oracle WebCenter Spaces functionality in a custom WebCenter application, using WebCenter Spaces APIs. Through these APIs, you can create group spaces, manage group space membership, retrieve group space information, and more, from your custom WebCenter applications.

The chapter also describes how to expose information from other applications, such as custom WebCenter applications in WebCenter Spaces.

This chapter includes the following sections:

- Introduction to WebCenter Spaces
- Exposing WebCenter Spaces in Custom WebCenter Applications Using APIs
- Using the WebCenter Spaces REST APIs
- Exposing Enterprise Applications in WebCenter Spaces

Note: Custom-built task flows that implement group space APIs (described in this chapter) can be deployed on custom WebCenter applications, but you cannot deploy them to WebCenter Spaces.

Audience
This section is intended for developers who want to expose WebCenter Spaces functionality in their custom WebCenter applications, or who are building custom applications that will be integrated with WebCenter Spaces.

27.1 Introduction to WebCenter Spaces

WebCenter Spaces is a Web-based application, built using the Oracle WebCenter Framework, WebCenter services, and Oracle Composer, that offers the very latest technology for social networking, communication, collaboration, and personal productivity. Figure 27–1 shows the WebCenter Spaces home page.
Out of the box, WebCenter Spaces offers a configurable work environment that enables individuals and groups to work and collaborate more effectively. The application is a self-service solution for managing individual and group interactions. It also provides intuitive tools that allow business users to come together and share information by adding pages and resources such as documents, charts, reports, portlets, business applications, services, and other ADF resources or views.

WebCenter Spaces provides two work environments within a single application: personal spaces and group spaces. Personal spaces provide each user with a private work area for storing personal content, keeping notes, viewing and responding to business process assignments, maintaining a list of online buddies, and performing many other tasks relevant to his or her unique working day. Group spaces support discrete work areas organized around an area of interest or a common goal.

In WebCenter Spaces, group spaces support the formation and collaboration of project teams and communities of interest. Group spaces bring people together in a virtual environment for ongoing interaction and information sharing—in essence, forming a social network. Figure 27–2 shows an example of a group space.
Structurally, group spaces are comprised of pages, many of which are dedicated to a particular service. For example, a Lists page provides the means to create and publish multicolumn lists. A Documents page provides a central library for uploading, organizing, and managing group content. A Search page includes features for saving searches and managing search results. In addition to these and other default pages, a group space supports custom pages created by authorized users. Page creation is easy with a wide selection of predefined layouts. With little effort, users can provide pages neatly tailored to the unique needs of their team or community. For more information about group spaces, see the Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

WebCenter provides various services that group spaces can use to offer a wide variety of functionality in support of group objectives. For example, a Documents service provides features for uploading and managing content; a Discussions service provides features for creating, managing, and participating in discussion forums. Some other examples of these services include Events, Lists, Announcements, and Search. These services are what helps to make group spaces a productive, collaborative working environment. Services expose subsets of their features and functionality through task flows. A task flow is a reusable view that may expose all or a subset of the features available from a particular service. For example, a Recent Documents task flow provides a subset of the functionality offered through the Documents service by listing documents that have recently been opened, added, or affected in some way.

To help users get group spaces up and running quickly, they can be based on out-of-the-box templates. These templates provide the optimal structure for supporting the two main types of group space. Use the Group Project template for group spaces where each member potentially comes from a different department but all members contribute toward reaching a common goal. Use the Community of Interest template to help create a group space for a community of people joining together to collaborate, create content, share ideas, and so on. Additionally, WebCenter Spaces allows users to turn any group space into a template, so they can design group spaces to suit specific audiences and offer them up as templates for others to use. For more information about creating templates, see "Creating Your Own..."

WebCenter Spaces can consume portlets from any WSRP or Oracle PDK-Java portlet producer if the producer is registered first. Therefore, any portlets you create for custom WebCenter applications can be used in WebCenter Spaces too. For more information, see the "Managing Portlet Producers” chapter in the Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter.

27.2 Exposing WebCenter Spaces in Custom WebCenter Applications Using APIs

This section includes the following subsections:
- Introduction to WebCenter Spaces APIs
- Case Study 1: Purchasing Application Uses a Group Space to Evaluate Suppliers
- Case Study 2: Customer Support Center Application Uses a Group Space to Discuss Customer Escalations
- How to Set Up Your Custom WebCenter Application to Use the WebCenter Spaces APIs
- How to Provide Group Space Functionality in Custom WebCenter Applications
- How to Handle Exceptions Raised by WebCenter Spaces APIs
- Finding More Information on WebCenter Spaces APIs
- Troubleshooting Issues with WebCenter Spaces APIs

27.2.1 Introduction to WebCenter Spaces APIs

While using your custom WebCenter application, users may encounter situations where a group space would be useful to help them complete a particular task. In such cases, it would be much less disruptive to remain within the context of the current application, rather than having to switch to the WebCenter Spaces application. To this end, WebCenter Spaces provides access to a subset of its group space functionality through several APIs. Using these APIs, you can integrate powerful group space functionality into your custom WebCenter application.

You can use WebCenter Spaces APIs to:
- **Create and manage group spaces and templates.** You can create and delete group spaces, and add custom attributes. For more information, see Section 27.2.5.1, "Managing Group Spaces and Templates."
- **Manage group space membership.** You can add and remove group space members. For more information, see Section 27.2.5.2, "Managing Group Space Membership."
- **Retrieve group space and template information.** For example, you can retrieve the WebCenter Spaces URL or the URL of a specific group space. You can also retrieve group space and group space template metadata. For more information, see Section 27.2.5.3, "Retrieving Group Space and Template Information."

WebCenter Spaces APIs are contained within several classes. Table 27–1 lists the different classes and describes the purpose of the APIs within each class.
Group spaces have many different applications. From your perspective, as a custom WebCenter application developer, here are a couple of case studies describing scenarios where building and working with group spaces through the APIs might come in useful:

- In a purchasing application, a purchasing manager may create a group space to discuss the suitability of a new supplier. For more information, see Section 27.2.2, "Case Study 1: Purchasing Application Uses a Group Space to Evaluate Suppliers."

- In a Customer Support Center application, a support analyst may create a group space to collaborate with other users about a customer escalation. For more information, see Section 27.2.3, "Case Study 2: Customer Support Center Application Uses a Group Space to Discuss Customer Escalations."

### 27.2.2 Case Study 1: Purchasing Application Uses a Group Space to Evaluate Suppliers

Consider a purchasing application built using the Oracle WebCenter 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 group space would provide this collaborative environment. So the purchasing application includes a call to the `createGroupSpace` API, and this enables the purchasing manager to click a link (Create a New Group Space) that displays a Create Group Space dialog, directly from the purchasing application, as shown in Figure 27–3.

**Figure 27–3 Purchasing Application Includes a Group Space Creation Link**

[Image of a group space creation dialog]

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 group space, and also selects a template (Purchasing Projects) as shown in Figure 27–4. The Purchasing Projects template sets up the group space quickly so it is ready to use right away. For example, the template defines which services are required (events, a document library, and so on) and any custom pages that may be required. Because the APIs enable you to create your own Create Group Space dialog, you can apply your own look and feel and terminology.
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 group space from the Supplier ID so that the purchasing manager would not have to enter any details at all. The group space could then be created with just the click of a link.

When the purchasing manager clicks OK, WebCenter Spaces displays the new Supplier Evaluation group space in WebCenter Spaces, as shown in Figure 27–5.
Figure 27–5  Supplier Evaluation Group Space

In WebCenter Spaces, the purchasing manager becomes the moderator of the group space automatically. The moderator can add content to the group space, initiate discussions, invite members, and collaborate with interested parties. Once consensus is reached regarding the supplier, the purchasing manager can approve or reject the supplier concerned.

27.2.3 Case Study 2: Customer Support Center Application Uses a Group Space to Discuss Customer Escalations

Consider a Customer Support Center application built using the Oracle WebCenter 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 group space. The support analyst creates a group space from within the Customer Support Center application and adds the required members. These members are then notified and use the group space to start discussing the customer situation. The support analyst views their updates to the group space inside the support application, navigating to the group space whenever necessary to obtain more specific details. Based on the information she gets from the other members of the group space, she can diagnose the problem and offer the customer a solution very quickly.

27.2.4 How to Set Up Your Custom WebCenter Application to Use the WebCenter Spaces APIs

Before you can use the WebCenter Spaces APIs you must ensure that the WebCenter Spaces application is up and running and that your project is set up correctly in JDeveloper.

This section includes the following subsections:

- Verifying That WebCenter Spaces Is Up and Running
- Setting Up the Custom WebCenter Application to Use WebCenter Spaces APIs
- Securing the Connection Between the Application and WebCenter Spaces
- Setting Up the Group Space Client Context

27.2.4.1 Verifying That WebCenter Spaces Is Up and Running

If you want to use the WebCenter Spaces APIs, the WebCenter Spaces application must be up and running.

**To verify that WebCenter Spaces is up and running:**

1. Start WebCenter Spaces. The URL is:
   
   http://host:port/webcenter

   You do not need to log in.

2. Verify that the WebCenter Spaces Web service is up and running. The URL is:
   
   http://host:port/webcenter/SpacesWebService?WSDL

   You should see a page similar to the one shown in Figure 27–6. If you do not see this page, contact your Fusion Middleware administrator.
For more information about setting up WebCenter Spaces, see the "Getting WebCenter Spaces Up and Running" chapter in the Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter.

### 27.2.4.2 Setting Up the Custom WebCenter Application to Use WebCenter Spaces APIs

Before you can call the WebCenter Spaces APIs in your custom WebCenter application, you must ensure that your application contains the correct libraries and that there is a connection to the WebCenter Spaces Web service.

**To set up your application to use the WebCenter Spaces APIs:**

1. Start JDeveloper.
2. In the Application Navigator, expand the application for which you want to provide group space functionality.
   
   The application should be based on the WebCenter Application template.
3. Right-click the view-controller project and choose **Project Properties**.
4. Select **Libraries and Classpath**.
5. Check that the following libraries are available in your project, adding any as necessary:
Exposing WebCenter Spaces in Custom WebCenter Applications Using APIs

- ADF DVT Faces Runtime
- ADF Model Generic Runtime
- ADF Model Runtime
- JAX-RPC Client
- JAX-WS Client
- JAXB
- Oracle JEWT
- WebCenter Spaces Client
- \texttt{JDEV_HOME/jdeveloper/modules/oracle.adf.model_11.1.1/adfm.jar}
- \texttt{JDEV_HOME/jdeveloper/modules/oracle.xdk_11.1.1/xml.jar}

6. Click OK.

7. In the Application Resources panel of the Application Navigator, right-click \texttt{Connections} and choose \texttt{New Connection > URL}.

8. In the Name field, enter \texttt{SpacesWebServiceEndpoint}.

9. In the URL Endpoint field, enter the WebCenter Spaces Web service URL:

   \texttt{http://host:port/webcenter/SpacesWebService}

10. Click OK.

   The connection information is added to the \texttt{connections.xml} file in the application's \texttt{META-INF} directory, for example:

   \texttt{C:\JDeveloper\mywork\mySpacesApplication\adf\META-INF}

   If the \texttt{connections.xml} file does not exist, it is created.

11. Open the \texttt{connections.xml} file and verify that the code shown in Example 27–1 appears:

   \textbf{Example 27–1  WebCenter Spaces Web Service Endpoint URL Connection}

   \begin{verbatim}
   <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>
   \end{verbatim}

   If you need to set the WebCenter Spaces Web service endpoint at runtime, you can use the \texttt{setGroupSpaceWebServiceEndpoint} API (Example 27–2). You can write a wrapper API that takes the endpoint as a parameter and then calls \texttt{setGroupSpaceWebServiceEndpoint}, passing the parameter. An exception is raised if the endpoint is set in \texttt{connections.xml}. 

Integrating with Oracle WebCenter Spaces 27-11
Example 27–2  Setting the WebCenter Spaces Web Service Endpoint at Runtime

```java
client.setGroupSpacesWebServiceEndpoint("http://myserver.example.com/webcenter/SpacesWebService");
```

27.2.4.3 Securing the Connection Between the Application and WebCenter Spaces

Before using the WebCenter Spaces APIs in your custom WebCenter application, you must ensure that the communication between the application (the consumer) and WebCenter Spaces (the producer) is secure. This is so that the identity of the user invoking the APIs is propagated to WebCenter Spaces in a secure manner where the integrity and confidentiality of the communication is maintained.

To do this, the WebCenter Spaces APIs use a policy of SAML based token passing with message protection. Your administrator must create a Java keystore and update the credential store so that WebCenter Spaces can verify the authenticity of the security tokens received from your application. You must then register this keystore and update the credential store using JDeveloper.

For information about the steps that the administrator must perform, see “Securing WebCenter Spaces for Applications Consuming Spaces Client APIs with WS-Security” in the Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter.

Before You Begin

Obtain the following from your administrator:

- Consumer keystore to use to secure the connection. This is a .jks file (for example, consumer.jks).
- Consumer public alias key stored in the keystore (for example, consumer).
- Password of the consumer public alias key (for example, mypassword1).
- Producer public alias key stored in the consumer keystore (for example, producer). This is the alias specified by the administrator when importing the trusted certificate of the producer.
- Consumer keystore password (for example, mypassword2).

To secure the connection:

1. Copy the keystore to the file system where your custom WebCenter application is running, for example:

   ```
   DOMAIN_HOME/config/fmwconfig
   ```

2. Open the `jps-config.xml` file and configure the keystore created by your administrator (Example 27–3).

Example 27–3  Keystore Instance in jps.config.xml

```xml
<serviceInstance name="keystore" provider="keystore.provider" location="/consumer.jks"
    <description>Default JPS Keystore Service</description>
    <property name="keystore.type" value="JKS"/>
    <property name="keystore.csf.map" value="oracle.wsm.security"/>
    <property name="keystore.pass.csf.key" value="keystore-csf-key"/>
    <property name="keystore.sig.csf.key" value="enc-csf-key"/>
    <property name="keystore.enc.csf.key" value="enc-csf-key"/>
</serviceInstance>
```
You can use Fusion Middleware Control to specify the keystore location if you prefer. For more information see "Managing Security" in the Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter.

3. Take a backup of cwallet.sso.

4. Open a command line prompt.

5. Navigate to the directory where JDeveloper is installed, for example: C:\Jdev\jdeveloper\common\bin

6. Run the following command:

   - In Windows: wlst.cmd
   - In Linux: ./wlst.sh

7. Run:

   connect('admin_user', 'admin_password', 'host:port')

   Where:

   - admin_user is the user name for the administrator of the server on which the application is running (for example, weblogic)
   - admin_password is the password for the administrator
   - host:port is the server on which the application is running

8. Run the following commands:

   createCred(map="oracle.wsm.security", key="enc-csf-key", user="alias", password="key_password", desc="Enc Password")

   createCred(map="oracle.wsm.security", key="sign-csf-key", user="alias", password="key_password", desc="Enc Password")

   createCred(map="oracle.wsm.security", key="keystore-csf-key", user="keystore-csf-key", password="keystore_password", desc="Keystore password")

   Where:

   - alias is the consumer public alias key in the keystore
   - key_password is the password for the public key
   - keystore_password is the keystore password

You can obtain this information from your administrator, as detailed in the "Before You Begin" section.

If you need any additional help to run these commands, see the 'createCred' section in the Oracle Fusion Middleware WebLogic Scripting Tool Command Reference.

You can use Fusion Middleware Control to update the credential store if you prefer. For more information see the "Securing WebCenter Spaces for Applications Consuming Spaces Client APIs with WS-Security" section in the Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter.

9. Restart the server on which your application is deployed, and then your default server.
27.2.4.4 Setting Up the Group Space Client Context

Before calling any of the WebCenter Spaces APIs in your application code, a few setup steps are required.

To set up the group space client context

1. Domain-wide configuration settings are used when you call WebCenter Spaces APIs from an application deployed on WebLogic Server (WLS), unless you specifically choose to override them. To override or set security configuration parameters, use the APIs provided by the `GroupSpaceWSContext` class.

   In particular, you must set the SAML issuer name and the public key alias for WebCenter Spaces, which are required for data encryption. If the WebCenter Spaces 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 Spaces Web service endpoint, for example, `http://server.example.com:8912/webcenter/SpacesWebService`.
   - `samlIssuer` is the issuer URI of the SAML Authority issuing assertions for this SAML Asserting Party (for example, `http://www.example.com/webcenter`).
   - `producer` is the public key alias for WebCenter Spaces, for example, `wcspaces`.

   You can obtain this information from your administrator, as detailed in the "Before You Begin" section.

   For a full list of the APIs provided by the `GroupSpaceWSContext` class, see the `Oracle Fusion Middleware Java API Reference for Oracle WebCenter`.

2. Initialize the group space 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 group space APIs.

27.2.5 How to Provide Group Space Functionality in Custom WebCenter Applications

The WebCenter Spaces APIs enable you to perform commonly used group space operations within a custom WebCenter application. Most of these APIs are provided by the `GroupSpaceWSClient` class. The APIs can be grouped into three main categories:

- Managing Group Spaces and Templates
- Managing Group Space Membership
- Retrieving Group Space and Template Information

Table 27–2 lists the APIs in the GroupSpaceWSClient class.

<table>
<thead>
<tr>
<th>Group Space API</th>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>createGroupSpace</td>
<td>Managing Group Spaces and Templates</td>
<td>Creates a new group space based on a template. See Section 27.2.5.1.1, &quot;Creating a Group Space.&quot;</td>
</tr>
<tr>
<td>setCustomAttribute</td>
<td>Managing Group Spaces and Templates</td>
<td>Creates one or more custom attributes for a group space. See Section 27.2.5.1.2, &quot;Creating a Custom Attribute.&quot;</td>
</tr>
<tr>
<td>deleteGroupSpace</td>
<td>Managing Group Spaces and Templates</td>
<td>Permanently removes a group space from WebCenter Spaces. See Section 27.2.5.1.4, &quot;Deleting a Group Space.&quot;</td>
</tr>
<tr>
<td>createGroupSpaceTemplate</td>
<td>Managing Group Spaces and Templates</td>
<td>Creates a new group space template based on an existing group space. See Section 27.2.5.1.3, &quot;Creating a Group Space Template.&quot;</td>
</tr>
<tr>
<td>addMember</td>
<td>Managing Group Space Membership</td>
<td>Makes a user (or a group) a group space member with a specific role. See Section 27.2.5.2.1, &quot;Adding Members to a Group Space.&quot;</td>
</tr>
<tr>
<td>inviteMember</td>
<td>Managing Group Space Membership</td>
<td>Invites a list of users to become members of a group space. See Section 27.2.5.2.2, &quot;Inviting Users to Join a Group Space.&quot;</td>
</tr>
<tr>
<td>removeMember</td>
<td>Managing Group Space Membership</td>
<td>Revokes group space membership. See Section 27.2.5.2.3, &quot;Removing Members from a Group Space.&quot;</td>
</tr>
<tr>
<td>getRoles</td>
<td>Managing Group Space Membership</td>
<td>Retrieves all the roles for a given group space. See Section 27.2.5.2.4, &quot;Retrieving Role Information.&quot;</td>
</tr>
<tr>
<td>getGroupSpaces</td>
<td>Retrieving Group Space and Template Information</td>
<td>Retrieves a list of group spaces for a given query string. See Section 27.2.5.3.1, &quot;Retrieving a List of Group Spaces.&quot;</td>
</tr>
<tr>
<td>getPublicGroupSpaces</td>
<td>Retrieving Group Space and Template Information</td>
<td>Retrieves a list of public group spaces for a given query string. See Section 27.2.5.3.2, &quot;Retrieving a List of Public Group Spaces.&quot;</td>
</tr>
<tr>
<td>getGroupSpaceMetadata</td>
<td>Retrieving Group Space and Template Information</td>
<td>Retrieves information (metadata) about the group space. See Section 27.2.5.3.3, &quot;Retrieving Group Space Metadata.&quot;</td>
</tr>
<tr>
<td>getGroupSpaceMetadataByGuid</td>
<td>Retrieving Group Space and Template Information</td>
<td>Retrieves information (metadata) about a group space given the group space’s GUID. See Section 27.2.5.3.3, &quot;Retrieving Group Space Metadata.&quot;</td>
</tr>
</tbody>
</table>
Table 27–2 (Cont.) APIs for Performing Common Group Space Operations

<table>
<thead>
<tr>
<th>Group Space API</th>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>getGroupSpaceTemplates</td>
<td>Retrieving Group Space and</td>
<td>Retrieves a list of group space templates for a given query string. See</td>
</tr>
<tr>
<td></td>
<td>Template Information</td>
<td>Section 27.2.5.3.4, &quot;Retrieving a List of Group Space Templates.&quot;</td>
</tr>
<tr>
<td>getGroupSpaceTemplateMetadata</td>
<td>Retrieving Group Space and</td>
<td>Retrieves information (metadata) about the group space template. See</td>
</tr>
<tr>
<td></td>
<td>Template Information</td>
<td>Section 27.2.5.3.5, &quot;Retrieving Group Space Template Metadata.&quot;</td>
</tr>
<tr>
<td>getGroupSpaceTemplateMetadataByGuid</td>
<td>Retrieving Group Space and</td>
<td>Retrieves information (metadata) about the group space template given</td>
</tr>
<tr>
<td></td>
<td>Template Information</td>
<td>the template's GUID. See Section 27.2.5.3.5, &quot;Retrieving Group Space</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Template Metadata.&quot;</td>
</tr>
<tr>
<td>getWebCenterSpacesURL</td>
<td>Retrieving Group Space and</td>
<td>Retrieves the WebCenter Spaces URL. See Section 27.2.5.3.6, &quot;Retrieving</td>
</tr>
<tr>
<td></td>
<td>Template Information</td>
<td>the WebCenter Spaces URL.&quot;</td>
</tr>
<tr>
<td>getGroupSpaceURL</td>
<td>Retrieving Group Space and</td>
<td>Retrieves a group space URL. See Section 27.2.5.3.7, &quot;Retrieving a Group</td>
</tr>
<tr>
<td></td>
<td>Template Information</td>
<td>Space URL.&quot;</td>
</tr>
<tr>
<td>getServiceRSSFeedURL</td>
<td>Retrieving Group Space and</td>
<td>Retrieves RSS feed URLs for the specified group space service.</td>
</tr>
<tr>
<td></td>
<td>Template Information</td>
<td></td>
</tr>
<tr>
<td>getServiceRSSFeedURLbyGuid</td>
<td>Retrieving Group Space and</td>
<td>Retrieves RSS feed URL for a given group space by GUID and a given service.</td>
</tr>
<tr>
<td></td>
<td>Template Information</td>
<td></td>
</tr>
</tbody>
</table>

27.2.5.1 Managing Group Spaces and Templates

Use the following WebCenter Spaces APIs to manage your group spaces:

- createGroupSpace
- setCustomAttribute
- createGroupSpaceTemplate
- deleteGroupSpace

Before you begin

Before using any of these APIs, you must complete all the steps listed in Section 27.2.4, "How to Set Up Your Custom WebCenter Application to Use the WebCenter Spaces APIs."

This section includes the following subsections

- Creating a Group Space
- Creating a Custom Attribute
- Creating a Group Space Template
- Deleting a Group Space

27.2.5.1.1 Creating a Group Space  Use the createGroupSpace API to create a group space that is based on an existing group space template.

To use this API, specify:

- The internal name of the new group space. This name can contain spaces.
- A display name for the new group space.
- A description of the group space. Although this is not mandatory, we recommend that you provide a description to help users identify the group space’s purpose.
- The name of the group space template you want to use. This should be the internal name of the template, not the display name. For example, the internal names for the built-in templates are Basic, CommunityOfInterest, and ProjectSpace.

Optionally, you can provide a comma separated list of keywords to help users locate the group space in searches.

Example 27–4 creates a group space with the name *Databases* that is based on the *CommunityOfInterest* template. The example also specifies two search keywords (*databases* and *Oracle*).

```java
Example 27–4 Creating a Group Space

//create the group space
GroupSpaceWSMetadata gsMetadata =
client.createGroupSpace("Databases", "Databases" "A community for people interested in databases", "databases, oracle", "CommunityofInterest");

//print the group space GUID to provide confirmation of creation
System.out.println("GUID: " + gsMetadata.getGuid());
```

For an example of how to provide custom attributes for a new group space, see Section 27.2.5.1.2, "Creating a Custom Attribute."

### 27.2.5.1.2 Creating a Custom Attribute

Every group space comes with built-in attributes such as name, description, date created, icon, and so on. In addition, you can define custom attributes that store additional information (metadata) that is unique to the group space and its characteristics.

Use the `setCustomAttribute` API to specify a custom attribute for a group space. To use this API, specify the name of the group space and a name, description, type, and value for the custom attribute.

Example 27–5 creates the Databases group space and then creates a custom attribute for that group space. The attribute is named *Vendors*, with the description *List of vendors*. It takes string values of *Oracle* and *IBM*.

```java
Example 27–5 Creating a Custom Attribute

//create the group space
client.createGroupSpace("Databases", "Databases", "A community for people interested in databases", null, "CommunityofInterest");

//create the custom attribute
client.setCustomAttribute("Databases", "Vendors", "List of vendors", "java.lang.String", "Oracle, IBM");
```

### 27.2.5.1.3 Creating a Group Space Template

Use the `createGroupSpaceTemplate` API to create a group space template based on an existing group space. To use this API specify a name, display name, and description for the template, and the name of the group space to use to create the template.

Example 27–6 creates a group space template based on the *Databases* group space.

```java
Example 27–6 Creating a Group Space Template

GroupSpaceWSMetadata templMetadata =
```
client.createGroupSpaceTemplate("DatabasesTemplate", "Databases Template", "A template based on the Databases group space", "Databases");
//print the template GUID to provide confirmation of creation System.out.println('GUID: ' + templMetadata.getGuid());

27.2.5.1.4 Deleting a Group Space Use the deleteGroupSpace API to permanently delete a group space from WebCenter Spaces. To use this API, specify the name of the group space to delete. The API returns a boolean value to indicate whether the deletion was successful.

Example 27–7 deletes the group space with the name Databases. The example uses the boolean value returned by the API to print a message about the success or failure of the operation.

Example 27–7 Deleting a Group Space

//delete the group space
Boolean success = client.deleteGroupSpace("Databases");
//print a message depending on the result of the deletion
if(success)
{
    System.out.println('Operation Succeeded');
}
else
{
    System.out.println('Operation Failed');
}

27.2.5.2 Managing Group Space Membership
Use the following group space APIs to manage group space membership:
- addMember
- inviteMember
- removeMember
- getRoles

Before you begin
Before using any of these APIs, you must complete all the steps listed in Section 27.2.4, "How to Set Up Your Custom WebCenter Application to Use the WebCenter Spaces APIs."

This section includes the following subsections:
- Adding Members to a Group Space
- Inviting Users to Join a Group Space
- Removing Members from a Group Space
- Retrieving Role Information

27.2.5.2.1 Adding Members to a Group Space Use the addMember API to give users (and groups) group space membership and assign the new members to a group space role. To use this API, specify the name of the group space, and a list of users/groups. The list must contain objects of type GroupSpaceWSMembers that specify the user/group name and the role to give that user/group in the group space.
You must specify the name of a valid user or user group that exists in the WebCenter Spaces identity store. For the role, choose one of the default roles (Moderator, Participant, Viewer) or one of the custom group space roles (if any). To retrieve a list of the available roles for a group space, use the `getRoles` API. For more information, see Section 27.2.5.2.4, "Retrieving Role Information." For more information about roles, see the "What You Should Know About Group Space Roles and Permissions" section in the Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

Example 27–8 makes users (of type `GroupSpaceWSMembers`) named Pat and Vicki and everyone in the Sales and Marketing user group, members of the Databases group space. Pat and everyone in Sales and Marketing are granted the Viewer role. Vicki is assigned the Participant role.

---

**Note:** Use `setGroup(true)` to indicate when you are adding a group of users.

---

**Example 27–8  Adding Members to a Group Space**

```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 group space
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());
}
```

### 27.2.5.2.2 Inviting Users to Join a Group Space

Use the `inviteMember` API to invite users to become members of a group space. To use this API, specify the name of the group space and a list of users to invite. The list must contain objects of type `GroupSpaceWSMembers` that specify the user name and the role to give that user in the group space. An invitation to join the group space is sent to each user, and each user can then accept or reject that invitation.

You must specify a valid user name that exists in the WebCenter Spaces identity store. For the role, choose one of the default roles (Moderator, Participant, Viewer) or one of the custom group space roles (if any). To retrieve a list of the available roles for a group space, use the `getRoles` API. For more information, see Section 27.2.5.2.4, "Retrieving Role Information." For more information about roles, see the "What You Should Know About Group Space Roles and Permissions" section in the Oracle Fusion Middleware User’s Guide for Oracle WebCenter.
Example 27–9 invites users (of type GroupSpaceWSMembers) named Pat and Vicki to become members of the Databases group space with Viewer and Participant roles respectively.

**Example 27–9 Inviting Users to Join a Group Space**

```java
//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 group space
client.inviteMember("Databases", inviteMem);
```

27.2.5.2.3 Removing Members from a Group Space  Use the removeMember API to revoke group space membership. To use this API, specify the name of the group space, and a list of users. The list must contain objects of type GroupSpaceWSMembers that specify the user name. To obtain a list of current group space members, use the getGroupSpaceMetadata API. For more information, see Section 27.2.5.3.3, "Retrieving Group Space Metadata."

Example 27–10 removes users (of type GroupSpaceWSMembers) Pat and Vicki from the membership of the Databases group space.

**Example 27–10 Removing Members from a Group Space**

```java
//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 group space
client.removeMember("Databases", remMem);

//print a list of members to confirm the members were removed
GroupSpaceWSMetadata memMetaData = client.getGroupSpaceMetadata("Databases");
for (GroupSpaceWSMembers mems: memMetaData.getMembers())
{
    System.out.println(mems.getMember());
    System.out.println(mems.getRole());
}
```

27.2.5.2.4 Retrieving Role Information  Use the getRoles API to retrieve all the roles for a given group space. This is useful for determining which roles are available when adding or inviting members to a group space. Roles include the default roles (Moderator, Participant, Viewer) and also custom group space roles (if any). To use this API, specify the name of the group space.

Example 27–11 retrieves and displays role information for the Databases group space.
Example 27–11  Retrieving Role Information

```java
//retrieve the list of roles
List<String> roles = client.getRoles("Databases");
//print the list of roles
for (String role : roles)
{
    System.out.println(role);
}
```

27.2.5.3 Retrieving Group Space and Template Information

Use the following group space APIs to retrieve group space information:

- `getGroupSpaces`
- `getPublicGroupSpaces`
- `getGroupSpaceMetadata`
- `getGroupSpaceMetadataByGuid`
- `getGroupSpaceTemplates`
- `getGroupSpaceTemplateMetadata`
- `getGroupSpaceTemplateMetadataByGuid`
- `getWebCenterSpacesURL`
- `getGroupSpaceURL`

**Before you begin**

Before using any of these APIs, you must complete all the steps listed in Section 27.2.4, "How to Set Up Your Custom WebCenter Application to Use the WebCenter Spaces APIs."

This section includes the following subsections

- Retrieving a List of Group Spaces
- Retrieving a List of Public Group Spaces
- Retrieving Group Space Metadata
- Retrieving a List of Group Space Templates
- Retrieving Group Space Template Metadata
- Retrieving the WebCenter Spaces URL
- Retrieving a Group Space URL
- Retrieving RSS Feed URLs for Group Space Services

27.2.5.3.1 Retrieving a List of Group Spaces  
Use the `getGroupSpaces` API to obtain a list of group spaces that match a given query string. To use this API, specify a query string. A null value query string returns a list of all group spaces that are accessible to the current user.

The API returns group spaces that are accessible to the current user, up to a maximum of 500.

Example 27–12 returns a list of group spaces containing the string `Database`. 
Exposing WebCenter Spaces in Custom WebCenter Applications Using APIs

Example 27–12  Retrieving a List of Specific Group Spaces
List<String> allGroupSpaces = client.getGroupSpaces("Database");

Example 27–13 returns a list of all group spaces to which the current user has access. This is achieved by specifying a null query string.

Example 27–13  Retrieving a List of All Group Spaces
List<String> allGroupSpaces = client.getGroupSpaces(null)

27.2.5.3.2 Retrieving a List of Public Group Spaces  Use the getPublicGroupSpaces API to obtain a list of public group spaces that match a given query string. To use this API, specify a query string. A null value query string returns a list of all public group spaces.

The API returns group spaces that are accessible to all users, even those who are not logged in to WebCenter Spaces, up to a maximum of 500.

Example 27–14 returns a list of public group spaces containing the string Database.

Example 27–14  Retrieving a List of Specific Public Group Spaces
List<String> allPublicGroupSpaces = client.getPublicGroupSpaces("Database");

Example 27–15 returns a list of all public group spaces. This is achieved by specifying a null query string.

Example 27–15  Retrieving a List of All Public Group Spaces
List<String> allPublicGroupSpaces = client.getPublicGroupSpaces(null)

27.2.5.3.3 Retrieving Group Space Metadata  Use the getGroupSpaceMetadata or getGroupSpaceMetadataByGuid APIs to obtain information (metadata) about a particular group space. This includes information such as the description of the group space, the name of the user who created it, the date on which it was last updated, and so on.

To use the getGroupSpaceMetadata API, specify the name of the group space. To use the getGroupSpaceMetadataByGuid API, specify the GUID of the group space. Note that while the group space name may be changed during the existence of the group space, the GUID always remains the same. You can obtain the GUID of a group space as follows:

getGroupSpaceMetadata("spaceName").getGuid();

Both APIs return a bean object that contains more APIs that you can then use for retrieving the group space metadata. These APIs are provided by the GroupSpaceWSMetadata class. Table 27–3 lists the APIs returned by the bean object:

<table>
<thead>
<tr>
<th>Group Space API</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>getCrowedBy</td>
<td>Returns the name of the person who created the group space.</td>
</tr>
<tr>
<td>getCustomAttributes</td>
<td>Returns the custom attributes for a group space.</td>
</tr>
<tr>
<td>getDescription</td>
<td>Returns the description of the group space.</td>
</tr>
<tr>
<td>getDisplayName</td>
<td>Returns the display name of the group space.</td>
</tr>
</tbody>
</table>

Table 27–3  APIs for Retrieving Group Space Metadata
Example 27–16 retrieves the Description, Keywords, and Last Updated Date information for the Databases group space given the group space name.

```
//get the exact name of the group space
List<String> myGroupSpace = client.getGroupSpaces("Databases");
//check that the API returns a single result
if(myGroupSpace.size() == 1) {
    //retrieve the metadata
    GroupSpaceWSMetadata metadata = client.getGroupSpaceMetadata(myGroupSpace);
    //get group space description
    System.out.println("Description: " + metadata.getDescription());
    //get group space keywords
    System.out.println("Keywords: " + metadata.getKeywords());
    //get the date the group space was last updated
    System.out.println("Last Updated Date: "+ metadata.getLastUpdated().toString());
}
```

Example 27–17 retrieves the Display Name, Creator, and Last Updated Date information for the Databases group space given the group space GUID.

```
GroupSpaceWSMetadata metadata = client.getGroupSpaceMetadataByGuid("Guid");
//get group space display name
System.out.println("Display Name: " + metadata.getDisplayName());
//get the name of the user who created the group space
System.out.println("Created By: " + metadata.getCreatedBy());
//get the date the group space was last updated
System.out.println("Last Updated Date: "+ metadata.getLastUpdated().toString());
```

Example 27–18 retrieves the Name, Description, Type, and Value for every custom attribute associated with the Databases group space.

---

**Table 27–3 (Cont.) APIs for Retrieving Group Space Metadata**

<table>
<thead>
<tr>
<th>Group Space API</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>getGroupSpaceState</td>
<td>Returns the state of the group space: active, offline, deleted.</td>
</tr>
<tr>
<td>getGuid</td>
<td>Returns the GUID of the group space.</td>
</tr>
<tr>
<td>getIconURL</td>
<td>Returns the location of the group space icon.</td>
</tr>
<tr>
<td>getKeywords</td>
<td>Returns a comma separated list of keywords used to describe the group space.</td>
</tr>
<tr>
<td>getLastUpdated</td>
<td>Returns the date the group space was last updated.</td>
</tr>
<tr>
<td>getLogoURL</td>
<td>Returns the location of the group space logo.</td>
</tr>
<tr>
<td>getMailingList</td>
<td>Returns the mailing list for the group space.</td>
</tr>
<tr>
<td>getMembers</td>
<td>Returns the current list of group space members.</td>
</tr>
<tr>
<td>getName</td>
<td>Returns the internal name of the group space.</td>
</tr>
<tr>
<td>isDiscoverable</td>
<td>Returns if the group space is returned in searches made by users who are not members of the group space.</td>
</tr>
<tr>
<td>isPublic</td>
<td>Returns if the group space is available to users who are not logged in.</td>
</tr>
</tbody>
</table>
**Example 27–18 Retrieving Custom Attribute Metadata**

```java
//get the exact name of the group space
List<String> myGroupSpace = client.getGroupSpaces("Databases");
//check that the API returns a single result
if (myGroupSpace.size() == 1)
{
    //retrieve the metadata
    GroupSpaceWSMetadata metadata = client.getGroupSpaceMetadata(myGroupSpace);
    //get list of custom attributes
    List<CustomMetadata.Attribute> attributes = metadata.getCustomAttributes();
    //get name, description, type, and value for each custom attribute
    for (CustomMetadata.Attribute attribute : attributes)
    {
        System.out.println("Name:");
        System.out.println("Description:");
        System.out.println("Type:");
        System.out.println("value.toString():");
    }
}
```

Example 27–19 retrieves a list of members of the Databases group space.

**Example 27–19 Retrieving Membership Information**

```java
//get the exact name of the group space
List<String> myGroupSpace = client.getGroupSpaces("Databases");
//check that the API returns a single result
if (myGroupSpace.size() == 1)
{
    //retrieve the metadata
    GroupSpaceWSMetadata metadata = client.getGroupSpaceMetadata(myGroupSpace);
    //get the list of members
    for (String member : metadata.getMembers())
    {
        System.out.println("Member UID:");
    }
}
```

**27.2.5.3.4 Retrieving a List of Group Space Templates** Use the `getGroupSpaceTemplates` API to obtain a list of group space templates that match a given query string. To use this API, specify a query string. A null value query string returns a list of all templates that are accessible to the current user.

The API returns templates that are accessible to the current user, up to a maximum of 500.

Example 27–20 returns a list of group space templates containing the string Interest.

**Example 27–20 Retrieving a List of Specific Group Space Templates**

```java
List<String> allGroupSpaceTemplates = client.getGroupSpaceTemplates("Interest");
```

Example 27–21 returns a list of all group space templates to which the current user has access. This is achieved by specifying a null query string.

**Example 27–21 Retrieving a List of All Group Space Templates**

```java
List<String> allGroupSpaceTemplates = client.getGroupSpaceTemplates(null);
```
27.2.5.3.5 Retrieving Group Space Template Metadata

Use the `getGroupSpaceTemplateMetadata` and `getGroupSpaceTemplateMetadataByGuid` APIs to obtain information (metadata) about a particular group space template. This includes information such as the description of the template, the name of the user who created it, and so on.

To use the `getGroupSpaceTemplateMetadata` API, specify the name of the template. To use the `getGroupSpaceTemplateMetadataByGuid` API, specify the GUID of the template. Note that while the group space template name may be changed during the existence of the template, the GUID always remains the same.

Both APIs return a bean object that contains more APIs that you can then use for retrieving the group space template metadata. These APIs are provided by the `GroupSpaceWSMetadata` class. Table 27–4 lists the APIs returned by the bean object:

<table>
<thead>
<tr>
<th>Group Space API</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>getCreatedBy</code></td>
<td>Returns the name of the person who created the template.</td>
</tr>
<tr>
<td><code>getDescription</code></td>
<td>Returns the description of the template.</td>
</tr>
<tr>
<td><code>getDisplayname</code></td>
<td>Returns the display name of the template.</td>
</tr>
<tr>
<td><code>getGuid</code></td>
<td>Returns the GUID of the template.</td>
</tr>
<tr>
<td><code>getIconURL</code></td>
<td>Returns the location of the template icon.</td>
</tr>
<tr>
<td><code>getKeywords</code></td>
<td>Returns a comma separated list of keywords used to describe the template.</td>
</tr>
<tr>
<td><code>getLogoURL</code></td>
<td>Returns the location of the template logo.</td>
</tr>
<tr>
<td><code>getName</code></td>
<td>Returns the internal name of the template.</td>
</tr>
</tbody>
</table>

Table 27–4 APIs for Retrieving Group Space Template Metadata

Example 27–22 retrieves the GUID, Description, and Created By information for the CommunityofInterest group space template given the template name.

**Example 27–22 Retrieving Template Metadata Using the Template Name**

```java
GroupSpaceWSMetadata metadata = client.getGroupSpaceTemplateMetadata(myGroupSpaceTemplate);
//get the exact name of the group space template
List<String> myGroupSpaceTemplate = client.getGroupSpaceTemplates("CommunityofInterest");
//check that the API returns a single result
if(myGroupSpaceTemplate.size() == 1)
{
    //retrieve the metadata -- get GUID
    System.out.println("GUID: " + metadata.getGuid());
    //get template description
    System.out.println("Description: " + metadata.getDescription());
    //get name of user who created the template
    System.out.println("Keywords: " + metadata.getCreatedBy());
}
```

Example 27–23 retrieves the name of a group space template given the template GUID.

**Example 27–23 Retrieving Template Metadata Given the Template GUID**

```java
GroupSpaceWSMetadata templGuidMetadata = client.getGroupSpaceTemplateMetadataByGuid("Guid");
System.out.println("Template Name: " + templGuidMetadata.getName());
```
27.2.5.3.6 Retrieving the WebCenter Spaces URL Use the getWebCenterSpacesURL API to obtain the URL of WebCenter Spaces.

Example 27–24 retrieves the URL of the currently running instance of WebCenter Spaces.

Example 27–24 Retrieving the WebCenter Spaces URL

```java
String myWebCenterSpacesURL = client.getWebCenterSpacesURL();
```

27.2.5.3.7 Retrieving a Group Space URL Use the getGroupSpaceURL API to obtain the URL of a specific group space. This is useful for when you want to construct a hyperlink for a group space or you have a relative URL that you need to make into an absolute URL. To use this API, specify the name of the group space.

Example 27–25 retrieves the URL of the Databases group space.

Example 27–25 Retrieving a Group Space URL

```java
String myGroupSpaceURL = client.getGroupSpaceURL("Databases");
```

Example 27–26 prints a list of group spaces as hyperlinks.

Example 27–26 Printing a List of Group Spaces as Hyperlinks

```java
//get the list of group spaces
List<String> spaces = client.getGroupSpaces("*");
//print the list of group spaces as hyperlinks
for (String spaceName : spaces) {
    print("<a href =" + client.getGroupSpaceURL(spaceName) + ">
```}

To construct the URL of a particular group space page, retrieve the group space URL and then add the page information. For information about how to create pretty URLs for group space pages, see "Using Pretty URLs for WebCenter Spaces Pages" in the Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

27.2.5.3.8 Retrieving RSS Feed URLs for Group Space Services In WebCenter Spaces, group space members can find out what is happening in a group space through various RSS news feeds. The following group space RSS feeds are available:

- Announcements RSS - View group space announcements
- Discussions RSS - Track contributions to discussion forums
- Lists RSS - Watch for revisions to lists
- Recent Activity RSS - Monitor recent activities

You can retrieve the RSS feed URLs for these group space services, from a custom WebCenter application, using the following WebCenter Spaces APIs:

- getServiceRSSFeedURL
- getServiceRSSFeedURLbyGuid
To obtain an RSS feed URL, you must identify the group space (by name or GUID) and specify the service required (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 Group Space RSS News Feed URLs Using `getServiceRSSFeedURL`

Use the `getServiceRSSFeedURL` API to obtain service-related RSS news feed URLs for a particular group space by specifying the group space 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 group space
- `service_ID` is the ID of the service for which you want to retrieve the RSS news feed URL. Use one of the following: `ANNOUNCEMENT_SERVICE_ID`, `DISCUSSION_FORUM_SERVICE_ID`, `RECENT_ACTIVITY_SERVICE_ID`, or `LIST_SERVICE_ID`.

Therefore, depending on the service required, you can use the following:

```java
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 27–27** retrieves the recent activity RSS feed URL for a group space named `Finance_Project`:

```java
String recentActivityURL = client.getServiceRSSFeedURL("Finance_Project", GroupSpaceWSClient.RECENT_ACTIVITY_SERVICE_ID);
```

### Retrieving RSS News Feed URLs Using `GetServiceRSSFeedURLbyGuid`

Use the `GetServiceRSSFeedURLbyGuid` API to obtain service-related RSS news feed URLs for a particular group space by specifying the group space's GUID.

To retrieve the RSS news feed URL for a service using `GetServiceRSSFeedURLbyGuid`, use the following:

```java
String service_URL = client.GetServiceRSSFeedURLbyGuid("groupspace_GUID", service_ID);
```
Where:

- `service_URL` refers to the service parameter being retrieved.
- `groupspace_GUID` is a group space GUID. For information about obtaining a group space’s GUID, see Section 27.2.5.3.3, "Retrieving Group Space Metadata.”
- `service_ID` is the ID of the service for which you want to retrieve the RSS news feed URL. Use one of the following: `ANNOUNCEMENT_SERVICE_ID`, `DISCUSSION_FORUM_SERVICE_ID`, `RECENT_ACTIVITY_SERVICE_ID`, or `LIST_SERVICE_ID`.

Example 27–28 retrieves the recent activity RSS feed URL for a group space with the GUID: `s2201fa44_b441_4bdd_950e_47307f6f9800`:

```java
Example 27–28 Retrieving the RSS Feed URL for Recent Activity
String recentActivityURL = client.getServiceRSSFeedByUrlGuid("s2201fa44_b441_4bdd_950e_47307f6f9800", GroupSpaceWSClient.RECENT_ACTIVITY_SERVICE_ID);
```

### 27.2.6 How to Handle Exceptions Raised by WebCenter Spaces APIs

The `GroupSpaceWSException` class provides APIs for handling exceptions raised by the group space APIs.

#### Table 27–5 Group Space APIs in the `GroupSpaceWSException` Class

<table>
<thead>
<tr>
<th>API</th>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>getLocalizedMessage</code></td>
<td><code>GroupSpaceWSException</code></td>
<td>Composes the localized error message.</td>
</tr>
<tr>
<td><code>printStackTrace</code></td>
<td><code>GroupSpaceWSException</code></td>
<td>Prints the exception and its back trace to the standard error stream.</td>
</tr>
</tbody>
</table>

This section includes the following subsections:

- Providing Localized Error Messages
- Listing the Error Stack

#### 27.2.6.1 Providing Localized Error Messages

Sometimes you may find that the default error messages provided by the APIs are not specific enough for your particular application. In these cases you can provide your own error messages.

Use the `getLocalizedMessage` API to compose application-specific error messages.

Example 27–29 shows a servlet that includes code to create a group space. If any exceptions are raised during the creation process, a localized message is printed.

**Example 27–29 Printing a Localized Error Message**

```java
servlet1.java

doit()
{
  ...
  print("<h>Output</h>");
  try
    { 
```
GroupSpaceWSClient client = new GroupSpaceWSClient(context);
client.createGroupSpace("Databases", "Databases, "A community for people
interested in databases", "databases, oracle", "CommunityofInterest");
print("Successfully created group space");
}

27.2.6.2 Listing the Error Stack

For debugging purposes, it is often useful to see which errors ultimately led to the
failure of a particular operation to discover the underlying cause of the problem. Use
the printStackTrace API to list all the errors that caused a particular exception.

Example 27–30 prints the exception and all the errors leading up to it.

Example 27–30  Listing the Error Stack

servlet1.java
doGet()
{
... 
print("<b>Output</b>");
try
{
    GroupSpaceWSClient client = new GroupSpaceWSClient(context);
    client.createGroupSpace("Databases", "Databases, "A community for people
interested in databases", "databases, oracle", "CommunityofInterest");
    print("Successfully created group space");
}
catch(GroupSpaceWSException ex)
{
    ex.printStackTrace();
}

27.2.7 Finding More Information on WebCenter Spaces APIs

API Reference Documentation

For detailed syntax information for each API, see the Oracle Fusion Middleware Java API
Reference for Oracle WebCenter.

27.2.8 Troubleshooting Issues with WebCenter Spaces APIs

If you experience issues with WebCenter Spaces APIs, check the following:

1. Verify that the credential stores for both WebCenter Spaces and the client custom
   WebCenter application are configured correctly.

2. Ensure the group space client context is set up correctly in the custom WebCenter application by checking the alias passed in the context.setRecipientKeyAlias. The alias should be the public key alias of the producer (WebCenter Spaces), for example:

```java
GroupSpaceWSContext context = new GroupSpaceWSContext();
context.setEndPoint(endPointUrl);
context.setRecipientKeyAlias("webservice_client_api");
groupSpaceInternalWSClient = new GroupSpaceWSInternalClient(context);
```

In this example, the public key alias of the producer is `webservice_client_api`. See also, Setting Up the Group Space Client Context.

3. Check that keystores exist at both ends of the connection. For example:

- `webcenter.jks` (copied to WebCenter Spaces end)
- `clientapi.jks` (copied to custom WebCenter application end)

For example, the following commands generate `clientapi.jks` and `webcenter.jks`:

```
keytool -genkeypair -keyalg RSA -dname "cn=webcenter,dc=us,dc=oracle,dc=com" -alias webcenter -keypass mypassword
-keystore webcenter.jks -storepass mypassword -validity 360
keytool -exportcert -v -alias webcenter -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=us,dc=oracle,dc=com" -alias clientapi -keypass mypassword -keystore clientapi.jks -storepass mypassword -validity 360
keytool -exportcert -v -alias clientapi -keystore clientapi.jks -storepass mypassword -rfc -file clientapi.cer
keytool -importcert -alias clientapi -file clientapi.cer -keystore webcenter.jks -storepass mypassword
```

See also, Securing the Connection Between the Application and WebCenter Spaces.

### 27.3 Using the WebCenter Spaces REST APIs

Oracle WebCenter provides REST APIs to support various group space operations. You can use REST APIs to perform the following actions in WebCenter Spaces:

- List all group spaces in WebCenter Spaces, list public group spaces only, list joined group spaces, list discoverable group spaces.
- View details for a single group space.
- List members of a group space.
- List group space member role assignments.
- List custom attributes defined for a group space.
- View the details of a particular custom attribute.
- View and add group space lists.
- View and update list items (rows) for a group space list.
This section describes the WebCenter Spaces REST APIs. It contains the following subsections:

- Section 27.3.1, "WebCenter Spaces Entry Point"
- Section 27.3.2, "WebCenter Spaces Resource Type Taxonomy"
- Section 27.3.3, "WebCenter Spaces Security Considerations"
- Section 27.3.4, "WebCenter Spaces Resource Types"

For an introduction to the REST APIs, see Section 28, "Using Oracle WebCenter REST APIs."

### 27.3.1 WebCenter Spaces Entry Point

Each REST service has a link element within the Resource Index that provides the entry point for that service. To find the entry point for WebCenter Spaces REST APIs, find the link element with a `resourceType` of:

```plaintext
urn:oracle:webcenter:spaces
```

The corresponding `href` or `template` element provides the URI entry point which returns a list of group spaces accessible to the current user. The client sends HTTP requests to this entry point to work with WebCenter Spaces.

For more information about the Resource Index, see Section 28.3.1, "The Resource Index."

For more information about resource types, see Section 28.3.2.1, "Resource Type."

### 27.3.2 WebCenter Spaces Resource Type Taxonomy

When the client has identified the entry point, it can then navigate through the resource type taxonomy to perform the required operations. For more information about the individual resource types, see the appropriate section in Section 27.3.4, "WebCenter Spaces Resource Types."

The taxonomy for WebCenter Spaces is:

```plaintext
urn:oracle:webcenter:spaces
  urn:oracle:webcenter:space
  urn:oracle:webcenter:space:members
    urn:oracle:webcenter:space:member
  urn:oracle:webcenter:space:attributes
    urn:oracle:webcenter:space:attribute
  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
```

### 27.3.3 WebCenter Spaces Security Considerations

There are no specific security considerations for this service. For general security considerations, see Section 28.6, "Security Considerations for WebCenter REST APIs."
27.3.4 WebCenter Spaces Resource Types

The following sections provide all the information you need to know about each resource type:

- urn:oracle:webcenter:spaces
- urn:oracle:webcenter:space
- urn:oracle:webcenter:space:members
- urn:oracle:webcenter:space:member
- urn:oracle:webcenter:space:attributes
- urn:oracle:webcenter:space:attribute
- 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:list:row
- urn:oracle:webcenter:space:list:columns
- urn:oracle:webcenter:space:list:column

27.3.4.1 urn:oracle:webcenter:spaces

Use this resource type to identify the URI to use to view a list of group spaces (GET). The response from a GET operation includes each group space, and each group space includes links to operate on that group space.

Navigation Paths to spaces

This section shows how the client can navigate through the hypermedia to access this resource:

resourceindex
spaces

Supported Methods for spaces

The following methods are supported by this resource:

- GET
  - request - body: None
  - query parameters:
    * startIndex
    * itemsPerPage
    * projection - Allowed values are summary and details. The default is summary.
    * visibility - Determines which group spaces are included in the list: all group spaces, public group spaces only, joined group spaces, or discoverable group spaces.
      
     Allowed values are all, public, joined and discoverable. The default is joined.
Resource Types Linked to from spaces
Table 27–6 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>

27.3.4.2 urn:oracle:webcenter:space
Use this resource type to identify the URI to use to view details for a single group space (GET). The response from a GET operation includes the specific group space identified by the URI, including its creator, description, members, and custom attributes.

Note: The response from this resource also includes a link to the group space’s message board.

Navigation Paths to space
This section shows how the client can navigate through the hypermedia to access this resource:

resourceindex
  spaces
    space

Supported Methods for space
The following methods are supported by this resource:

- GET
  - request - body: None
  - query parameters:
    * projection - Allowed values are summary, and details. The default is details. Choose summary to exclude member and custom attribute details.
  - response - body: group space

For more information about query parameters, see Section 28.3.2.5, "Templates."

Read-only Elements for space
Table 27–7 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 group space.</td>
</tr>
<tr>
<td>creator</td>
<td>String</td>
<td>ID of the user who created the group space.</td>
</tr>
</tbody>
</table>
Resource Types Linked to from space

Table 27–8 lists the resource types that the client can link to from 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 information about the user who created the group space, including GUID, ID, display name, and a link to the profile icon (same user as creator).</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>Description of the group space.</td>
</tr>
<tr>
<td>displayName</td>
<td>String</td>
<td>Name of the group space as displayed to members.</td>
</tr>
<tr>
<td>iconUrl</td>
<td>String</td>
<td>Fully qualified URL for the group space icon image.</td>
</tr>
<tr>
<td>logoUrl</td>
<td>String</td>
<td>Relative URL for the group space logo image.</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>Internal group space name.</td>
</tr>
<tr>
<td>isOffline</td>
<td>Boolean</td>
<td>Indicates whether the group space is offline or online.</td>
</tr>
</tbody>
</table>

Table 27–8 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:activities:stream</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:messageBoard</td>
</tr>
</tbody>
</table>

27.3.4.3 urn:oracle:webcenter:space:members

Use this resource type to identify the URI to use to view a list of group space members (GET). The response from a GET operation includes each member of the group space, each member includes links to operate on that member.

Navigation Paths to members

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
  spaces
    space
```

Supported Methods for members

The following methods are supported by this resource:

- **GET**
  - **request - body**: None
  - **query parameters**: None
  - **response - body**: collection of members
Resource Types Linked to From members
Table 27–9 lists the resource types that the client can link to from this resource.

<table>
<thead>
<tr>
<th>rel</th>
<th>resourceType</th>
</tr>
</thead>
</table>
| self      | urn:oracle:webcenter:space:members
           | urn:oracle:webcenter:space:member |
| urn:oracle:webcenter:parent | urn:oracle:webcenter:space |

27.3.4.4 urn:oracle:webcenter:space:member
Use this resource type to identify the URI to use to view details for a group space member and their current role assignments (GET). The response from a GET operation includes the specific member identified by the URI, including a hyperlink to the member’s profile.

Navigation Paths to member
This section shows how the client can navigate through the hypermedia to access this resource:

resourceindex
space
  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 27–10 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 group space member.</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>User ID of the group space member.</td>
</tr>
<tr>
<td>role</td>
<td>String</td>
<td>Group space role assigned to the member.</td>
</tr>
<tr>
<td>displayName</td>
<td>String</td>
<td>Display name of the group space member.</td>
</tr>
</tbody>
</table>

Resource Types Linked to From member
Table 27–11 lists the resource types that the client can link to from this resource.
27.3.4.5 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 group space (GET). The response from a GET operation includes each custom attribute for the group space, and each attribute includes links to operate on that attribute.

**Navigation Paths to attributes**

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
  spaces
    space
```

**Supported Methods for attributes**

The following methods are supported by this resource:

- GET
  - request - body: None
  - query parameters: None
  - response - body: collection of attributes

**Resource Types Linked to From attributes**

Table 27–12 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:space:attributes</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:space:attribute</td>
</tr>
</tbody>
</table>

27.3.4.6 urn:oracle:webcenter:space:attribute

Use this resource type to identify the URI to use to view the name and value of single group space attribute (GET). The response from a GET operation includes the specific attribute identified by the URI.

**Navigation Paths to attribute**

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
  spaces
```

---

**Table 27–11 Related Resource Types for urn:oracle:webcenter:space:member**

<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>
</tbody>
</table>

---

**Table 27–12 Related Resource Types for urn:oracle:webcenter:space:attributes**

<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:parent</td>
<td>urn:oracle:webcenter:space</td>
</tr>
</tbody>
</table>
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 27–13 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 group space custom attribute.</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>Description of the group space 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 27–14 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:attribute</td>
</tr>
<tr>
<td>urn:oracle:webcenter:parent</td>
<td>urn:oracle:webcenter:space</td>
</tr>
</tbody>
</table>

27.3.4.7 urn:oracle:webcenter:space:resourceindex
Use this resource type to identify the URI to use to view the WebCenter Spaces resource index.

Navigation Paths to resourceindex
This section shows how the client can navigate through the hypermedia to access this resource:

resourceindex
  spaces
    space

Supported Methods for resourceindex
The following methods are supported by this resource:

- GET
  - request-body: None
query parameters: None
response - body: resource index

Resource Types Linked to From resourceindex

Table 27–15 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></td>
<td>urn:oracle:webcenter:space:lists</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:discussions:forum:</td>
</tr>
</tbody>
</table>

27.3.4.8 urn:oracle:webcenter:space:lists

Use this resource type to identify the URI to use to view (GET) and add (POST) group space lists. The response from a GET operation includes each list in the group space, and each list includes links used to operate on that list. The response from a POST operation includes the list that was created in this collection of lists and a link to operate on that list.

Navigation Paths to lists

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
  spaces
    resourceindex
    space
    lists
```

Supported Methods for lists

The following methods are supported by this resource:

- GET
  - request - body: None
  - query parameters:
    * startIndex
    * itemsPerPage
    * q

You can search on name, title, description, creator, created, modifier, and modified.

For string type elements (that is, name, title, description, creator, and modifier), you can use the following supported operands: equals, not.equals, and contains. For example:

```
q=name:contains:issues
q=creator:equals:monty
```

For date type elements (that is, created and modified), you can use the following supported operands: equals, not.equals, greater.than, greater.than.or.equals, less.than, less.than.or.equals. For example:
q=created:greater.than:10-SEP-2009

* projection - Allowed values are summary and details. The default is summary.

- response - body: collection of lists

POST
- request - body: list
- response - body: list

For more information about query parameters, see Section 28.3.2.5, "Templates."

Resource Types Linked to From lists
Table 27–16 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>

27.3.4.9 urn:oracle:webcenter:space:list
Use this resource type to identify the URI to use to view (GET), update (PUT), and delete (DELETE) a group space list. 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
Using the WebCenter Spaces REST APIs

- request - body: None
- response - body: None

Writable Elements for list
Table 27–17 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</td>
<td>Name of the group space list.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>characters</td>
<td></td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>No</td>
<td>0 or more</td>
<td>Description of the list.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>characters</td>
<td></td>
</tr>
<tr>
<td>columns</td>
<td>urn:oracle:webcenter:space:list:columns</td>
<td>Yes</td>
<td>1 or more</td>
<td>Columns that make up the list.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>characters</td>
<td></td>
</tr>
</tbody>
</table>

Read-only Elements for list
Table 27–18 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 group space list.</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the group space list.</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>Description of the list.</td>
</tr>
<tr>
<td>scopeguid</td>
<td>String</td>
<td>Global ID of the parent group space.</td>
</tr>
<tr>
<td>scopename</td>
<td>String</td>
<td>Name of the parent group space.</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>

Resource Types Linked to From list
Table 27–19 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</td>
</tr>
<tr>
<td></td>
<td>urn:oracle:webcenter:space:list:rows</td>
</tr>
</tbody>
</table>
27.3.4.10  urn:oracle:webcenter:space:list:rows

Use this resource type to identify the URI to use to view (GET) and create (POST) group space list rows (list items). The response from a GET operation includes each row in this list, each row includes links to operate on that row. The response from a POST operation includes the row that was created in this list and a link to operate on that row.

Navigation Paths to rows

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
  spaces
    resourceindex
      lists
        list
          rows
```

Supported Methods for rows

The following methods are supported by this resource:

- **GET**
  - **request - body**: None
  - **query parameters**:
    * `startIndex`
    * `itemsPerPage`
    * `q`
      You can search on `name`, `title`, `description`, `creator`, `created`, `modifier`, and `modified`.
      For string type elements (that is, `name`, `title`, `description`, `creator`, and `modifier`), you can use the following supported operands: `equals`, `not.equals`, `contains`, and `starts.with`. For example:
      ```
      q=name:contains:issues
      q=creator:equals:monty
      ```
      For date type elements (that is, `created` and `modified`), you can use the following supported operands: `equals`, `not.equals`, `greater.than`, `greater.than.or.equals`, `less.than`, `less.than.or.equals`. For example:
      ```
      q=created:greater.than:10-SEP-2009
      ```
  - **response - body**: collection of rows

- **POST**
  - **request - body**: row
– response - body: row

For more information about query parameters, see Section 28.3.2.5, “Templates.”

Resource Types Linked to From rows

Table 27–20 lists the resource types that the client can link to from this resource.

Table 27–20  Related Resource Types for urn:oracle:webcenter:space:list:rows

<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>

27.3.4.11 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 group space list row (list item). The response to a GET operation includes the specific row identified by the URI. The response to a PUT operation includes the modified version of the row identified by the URI. The response to a DELETE operation is a 204 status code.

Navigation Paths to row

This section shows how the client can navigate through the hypermedia to access this resource:

resourceindex
    spaces
        resourceindex
            lists
                list
                    rows
                        row

Supported Methods for row

The following methods are supported by this resource:

■ GET
   – request - body: None
   – query parameters: None
   – response - body: row

■ PUT
   – request - body: row
   – response - body: row

■ DELETE
   – request - body: None
   – response - body: None

For more information about query parameters, see Section 28.3.2.5, “Templates.”
**Writable Elements for row**

Table 27–21 lists the writable elements for this resource.

<table>
<thead>
<tr>
<th>Element Type</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 27–22 lists the read-only elements for this resource.

<table>
<thead>
<tr>
<th>Element Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>ID of the row (list item).</td>
</tr>
<tr>
<td>listId</td>
<td>ID of the group space list.</td>
</tr>
<tr>
<td>scope</td>
<td>Global ID of the parent group space.</td>
</tr>
<tr>
<td>creator</td>
<td>ID of the user that created the list item.</td>
</tr>
<tr>
<td>author</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 the list item was created.</td>
</tr>
<tr>
<td>modifier</td>
<td>ID of the user that last modified the list item.</td>
</tr>
<tr>
<td>modifiedByUser</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 the list item was last modified.</td>
</tr>
</tbody>
</table>

**Resource Types Linked to From row**

Table 27–23 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:row</td>
</tr>
</tbody>
</table>

**27.3.4.12 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 28.3.2.5, "Templates."

Resource Types Linked to From columns
Table 27–24 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: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>

27.3.4.13 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
```
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 27–25 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 column
Table 27–26 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 column (list item detail).</td>
</tr>
</tbody>
</table>

Resource Types Linked to From column
Table 27–19 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:column</td>
</tr>
</tbody>
</table>
27.4 Exposing Enterprise Applications in WebCenter Spaces

You can expose various enterprise applications inside WebCenter Spaces, including:

- Custom WebCenter applications—applications developed using the WebCenter 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 support for external applications, enable WebCenter Spaces to consume and present application data in a unified way. The following sections tell you how.

This section includes the following subsections:

- Exposing Custom WebCenter Applications in WebCenter Spaces
- Exposing Oracle Applications in WebCenter Spaces
- Exposing Non-Oracle Applications in WebCenter Spaces

27.4.1 Exposing Custom WebCenter Applications in WebCenter Spaces

You can expose custom WebCenter applications, built using the WebCenter Framework, as portlets and task flows in WebCenter Spaces:

- **Portlets**: Web components that are deployed inside a container and generate dynamic content. For more information, see Chapter 29, "Overview of Portlets."
- **Oracle JSF Portlet Bridge**: A technology that enables you to easily expose existing ADF applications and task flows as JSR 168 portlets. For more information, see Chapter 30, "Creating Portlets with the Oracle JSF Portlet Bridge."
- **Task flows**: ADF navigational components that define the transition between states and activities using call methods on managed beans, evaluating an EL expression, and so on. For more information see, "Getting Started with ADF Task Flows" in the Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework.

27.4.2 Exposing Oracle Applications in WebCenter Spaces

You can expose Oracle Applications Unlimited products, for example, E-Business Suite, Siebel, PeopleSoft, and JDEdwards, in WebCenter Spaces, 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 Spaces.
- **Prebuilt portlets**: Enterprise applications such as Oracle E-Business Suite ship portlets out of the box that you can register with WebCenter Spaces and consume in the same way as any other portlet.

27.4.3 Exposing Non-Oracle Applications in WebCenter Spaces

You can expose web applications developed using non-Oracle platforms in WebCenter Spaces, as follows:
WebCenter support for external applications: External applications enable credential stores to pass mapped user identities to web applications that require their own authentication. For more information, see "Registering External Applications" in the Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter and "Working with External Applications" in the Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

Oracle Composer iframe-level integration: The Oracle Composer Web Page page style and component display any web page content, including pages from applications built using a different web technology. For more information, see "Working with Page Layouts, Styles, and Schemes" in the Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

Raw HTML markup injection in Oracle Composer: The Oracle Composer HTML Markup layout component injects HTML or JavaScript snippets into WebCenter Spaces pages. You can use this component to display content from an external source. For more information see "Working with HTML Markup Layout Component Properties" in the Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

OmniPortlet: A portlet that publishes data from various data sources using a variety of layouts. For more information, see Chapter 35, "Creating Portlets with OmniPortlet.”
Using Oracle WebCenter REST APIs

Oracle WebCenter provides a set of REST APIs for retrieving and modifying server data. This chapter discusses the WebCenter REST APIs.

This chapter includes the following sections:

- Section 28.1, "Introduction to REST"
- Section 28.2, "Introduction to WebCenter REST APIs"
- Section 28.3, "Understanding the Link Model"
- Section 28.4, "Understanding Items Hypermedia"
- Section 28.5, "Navigating Hypermedia Using HTTP"
- Section 28.6, "Security Considerations for WebCenter REST APIs"
- Section 28.7, "Understanding Common Types"
- Section 28.8, "WebCenter REST API Examples"

This chapter includes some examples that demonstrate how to use WebCenter REST APIs. For more examples, see the Oracle WebCenter Demonstrations and Samples page on the Oracle Technology Network (OTN) at:


28.1 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.
28.2 Introduction to WebCenter REST APIs

In addition to enabling mobile access, WebCenter REST APIs allow browser-based tools using emerging Web 2.0 technologies to access and modify WebCenter and WebCenter Spaces data. In general, the WebCenter REST commands provide a more natural and easy-to-use alternative to a SOAP-style web services approach.

Oracle WebCenter REST APIs provide the following functionality:

- The Discussions service REST APIs enable a client to post, read, update, and delete discussion forums, topics, and messages. For more information, see Section 17.3.9, "Using the Discussions Service REST APIs."
- The People Connections service REST APIs enable a client to view profile data; manage connection lists, feedback, and messages; and view activities for users, lists, and group spaces. For more information, see Section 20.3.3, "Using the People Connections Service REST APIs."
- The WebCenter Spaces REST APIs enable a client to retrieve group space metadata and view, create, update, and delete group space lists and list items. For more information, see Section 27.3, "Using the WebCenter Spaces REST APIs."
- The Content Management REST service uses the CMIS (Content Management Interoperability Services) RESTful server binding to provide access to the CM VCR (Content Management Virtual Content Repository). For more information, see the Oracle Fusion Middleware Content Management REST Service Developer’s Guide.

28.3 Understanding the Link Model

Hypermedia is at the core of the two most successful web-based formats: HTML and ATOM. Hypertext (the H in HTML) is a proper subset of hypermedia. Hypertext and ATOM allow consumers to navigate to other hypermedia documents through links. Oracle’s link element is based on HTML and ATOM links.

Hypermedia drives RESTful application state (known as HATEOAS: Hypermedia As The Engine Of Application State).

<table>
<thead>
<tr>
<th>Note:</th>
<th>HATEOAS analogy to define application state:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>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).</td>
</tr>
<tr>
<td></td>
<td>HATEOAS dictates that this state—the application state—be captured wholly in hypermedia. 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.</td>
</tr>
</tbody>
</table>

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 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 28.3.1, "The Resource Index"
- Section 28.3.2, "The Anatomy of a Link"

### 28.3.1 The Resource Index

In WebCenter, the Resource Index is your starting point—the set of top-level URI entry points. It provides the way in to all the available WebCenter RESTful services. The Resource Index URI is the only URI that you need to know.

The WebCenter Resource Index URI is:

```http://host:port/rest/api/resourceIndex```

The first step in using the WebCenter 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 28–1 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.

**Example 28–1 Response to a GET on the Resource Index**

```json
{
  "resourceType": "urn:oracle:webcenter:resourceindex",
  "links": [
    {
      "template": "opaque-template-uri",
      "resourceType": "urn:oracle:webcenter:messageBoard",
      "href": "opaque-uri",
      "capabilities": "urn:oracle:webcenter:read"
    },
    {
      "resourceType": "urn:oracle:webCenter:cmis",
      "href": "opaque-uri",
      "capabilities": "urn:oracle:webcenter:read"
    },
    {
      "template": "opaque-template-uri",
      "resourceType": "urn:oracle:webcenter:discussions:forums",
      "href": "opaque-uri",
      "capabilities": "urn:oracle:webcenter:read"
    },
    {
      "resourceType": "urn:oracle:webcenter:resourceindex",
      "rel": "self",
      "href": "http://host:port/rest/api/resourceIndex",
      "capabilities": "urn:oracle:webcenter:read"
    },
    {
      "template": "opaque-template-uri",
      "resourceType": "urn:oracle:webcenter:activities:stream",
      "href": "opaque-uri",
      "capabilities": "urn:oracle:webcenter:read"
    }
  ]
}
```
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 28–2 shows a method that locates a URI given the Resource Index JSON data.

**Example 28–2 Locating the URI for a Particular Service in the Resource Index**

```javascript
/* 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 < jsonData.links.length; i++) {
        if (jsonData.links[i].resourceType == strResourceType) {
            return jsonData.links[i].href;
        }
    }
}
```

### 28.3.2 The Anatomy of a Link

Example 28–3 and Example 28–4 show the anatomy of a hypermedia link as XML and in JSON document fragments, respectively.
Example 28–3  Link in an XML Document Fragment

Example 28–4  Link in a JSON Document Fragment

Example 28–5  XML Naming Convention

Example 28–6  JSON Naming Convention

This section includes the following subsections that describe the different attributes of the hypermedia link:

- Section 28.3.2.1, "Resource Type"
- Section 28.3.2.2, "Relationship"
- Section 28.3.2.3, "Capabilities"
- Section 28.3.2.4, "Media Type"
- Section 28.3.2.5, "Templates"
28.3.2.1 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 28.5, "Navigating Hypermedia Using HTTP."

28.3.2.2 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

28.3.2.3 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:** 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.
28.3.2.4 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 XML. For more information about CMIS REST APIs, see the Oracle Fusion Middleware Content Management REST Service Developer’s Guide.

28.3.2.5 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 28–7 shows a URI template including several request query parameters.

**Example 28–7 URI Template**

```
http://host:port/.../lists?startIndex={startIndex}&itemsPerPage={itemsPerPage}&q={searchTerms}&projection={projection}
```

WebCenter REST APIs use a simple slot replacement syntax that follows many industry URI template schemes.

For example, using the template in Example 28–7, to see 10 list items on the first page, the client would provide a value of 1 for the `startIndex` parameter and a value of 10 for the `itemsPerPage` parameter, as shown in Example 28–8.

**Example 28–8 URI Template with Parameter Values**

```
http://host:port/.../lists?startIndex=1&itemsPerPage=10
```

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 28–9.

**Example 28–9 Encoding Special Characters in URI Templates**

```
http://host:port/.../lists?q=G%FCnter
```

**Note:** All unused parameters must be removed from a template before it can be used. Clients may not submit unprocessed templates to the service that produced it; doing so results in undefined behavior, generally returning a status code of 500.

Clients must process templates into valid URI form before submitting to the server. Clients must replace slots with appropriate values, taking care to properly URI encode any value replacing the slot token. If a client does not have a suitable value for one or more of the slots in the template, then it must replace the slot token with an empty string.
Common Request Query Parameters

Many resources support a common set of request query parameters. For example, when retrieving a collection of entities, it is common to change the shape of the results set by limiting the quantity or details of the results. The REST framework uses the following request parameters to scope results and provide security:

- **startIndex**—Specifies the index of the first matching result that should be included in the result set (0-n ... zero based). This is used for pagination.
- **itemsPerPage**—Specifies the maximum number of results to return in the response (1-n). This is used for pagination.
- **q**—Specifies implementation-specific searching. Searches may be specified using the following format (square brackets [ ] denote optional values):

  ```
  [[field1:operand]:value1;field2:operand:value2]]
  ```

  For example:

  ```
  &q=login:equals:monty
  &q=title:contains:issues
  &q=creator:equals:monty;description:contains:Urgent
  ```

While each resource uses the same format for the q parameter, the way search is implemented is different depending on the resource being searched. For more information about how each resource implements search, see the chapter for the specific service.

- **projection**—Reserved for implementation-specific projection of model representations, such as variable recursion depth, field or attribute filtering. Valid values are *summary* or *details*.

  For example, requesting a projection of *summary* for a collection of lists, returns only the title, description, and hypermedia links. Requesting a projection of *details* results in the server sending back a collection of lists that includes all the column metadata for each list. This may require additional processing time or database queries on the server.

  The following example request results in the response entity containing a deeper object graph.

  ```
  http://host:port/...lists&projection=details
  ```

For information about how these parameters are supported by specific resources, see the chapter for the appropriate service.

### 28.4 Understanding Items Hypermedia

A collection of *items* makes up the actual content of responses. This is at the same level as the *links* section described previously. Each item (including the top level tag in each response) has one common attribute (*resourceType*) in addition to resource-specific content and format. For details beyond the *resourceType*, see the chapter for the specific service.

### 28.5 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 28–1 describes the general pattern followed when constructing opaque resource URIs. The `resourceType` differentiates whether the HTTP method operates on a collection of resources or an individual resource.

<table>
<thead>
<tr>
<th>HTTP Method</th>
<th>Response for a Collection of Resources</th>
<th>Response for an Individual Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>Returns resource collection container (200 HTTP response code)</td>
<td>Returns resource (200 HTTP response code)</td>
</tr>
<tr>
<td>PUT</td>
<td>Cannot update a collection of resources (405 HTTP response code)</td>
<td>Updates and returns resource (200 HTTP response code)</td>
</tr>
<tr>
<td>POST</td>
<td>Creates and returns resource (201 HTTP response code)</td>
<td>Cannot create a resource within an individual resource (405 HTTP response code)</td>
</tr>
<tr>
<td>DELETE</td>
<td>Cannot delete a collection of resources (405 HTTP response code)</td>
<td>Deletes resource (204 HTTP response code)</td>
</tr>
</tbody>
</table>

HTTP Response Status Codes

Table 28–2 describes the potential response status codes.

<table>
<thead>
<tr>
<th>HTTP Response Status Code</th>
<th>Description</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>200 OK</td>
<td>Upon successful completion of a GET or PUT. This is accompanied by a resource representation as a response body.</td>
<td></td>
</tr>
<tr>
<td>201 Created</td>
<td>Upon successful completion of a POST. This has a location header to the newly-created resource.</td>
<td></td>
</tr>
<tr>
<td>204 No Content</td>
<td>Upon successful completion of a DELETE.</td>
<td></td>
</tr>
<tr>
<td>400 Bad Request</td>
<td>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>
<td></td>
</tr>
<tr>
<td>401 Unauthorized</td>
<td>Client may retry by submitting credentials. This may be accompanied with a fault response body to help diagnose the issue.</td>
<td></td>
</tr>
<tr>
<td>403 Forbidden</td>
<td>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>
<td></td>
</tr>
<tr>
<td>404 Not Found</td>
<td>Referencing a specific resource with an ID, but that resource does not exist.</td>
<td></td>
</tr>
<tr>
<td>405 Method Not Allowed</td>
<td>Method Not Allowed. This includes a list of valid methods for the requested resource.</td>
<td></td>
</tr>
<tr>
<td>406 Not Acceptable</td>
<td>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>
<td></td>
</tr>
<tr>
<td>409 Conflict</td>
<td>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>
<td></td>
</tr>
</tbody>
</table>
28.6 Security Considerations for WebCenter REST APIs

Many of the WebCenter REST URIs reference protected resources (similar to protected web pages) and require authentication for access. You can pass this authentication in with the request using basic authentication, or you can configure the client and the WebCenter REST service to use single sign-on. For more information about single sign-on, see the section "Configuring a WebCenter Application to Use Single Sign-On" in the Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter.

Basic authentication sends the user's password in plain text. If you use this type of authentication, you should consider securing the connection using SSL. For more information, see the section "Configuring WebCenter Applications and Components to Use SSL" in the Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter.

To provide additional security, every URI, for both href and template attributes, includes a stoken parameter. This is a session-scoped token and is generated by the server, using a custom URIService API. These tokens help prevent Cross-Site Request Forgery (CSRF) attacks.

For example:

```xml
<link
    template="opaque-template-uri/@me?startIndex={startIndex}&itemsPerPage={itemsPerPage}&stoken=generated-stoken"
    resourceType="urn:oracle:webcenter:messageBoard"
    href="opaque-uri/@me?stoken=generated-stoken"
    capabilities="urn:oracle:webcenter:read"/>
```

Note: The stoken is not used for authentication or identity propagation.

WebCenter REST APIs operate under the identity of the authenticated user. For example, the group space REST APIs only return information for, and allow changes to, group spaces to which the user has access.

Security Considerations for CMIS REST APIs

The CMIS REST APIs do not use the same authentication scheme as the other WebCenter REST APIs. Whereas other WebCenter 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.

28.7 Understanding Common Types

There are several types that are shared by multiple WebCenter REST APIs.

This section includes the following subsections:
- Section 28.7.1, "Common Types"
- Section 28.7.2, "Portable Contact Types"

28.7.1 Common Types

Common types provide a consistent way to reference objects used in the WebCenter REST APIs.

This section includes the following subsections:
- Section 28.7.1.1, "personReference"
- Section 28.7.1.2, "groupSpaceReference"

28.7.1.1 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.

Depending on where the personReference type is included, it also returns 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.

28.7.1.2 groupSpaceReference

This is a generic data type that represents a group space. It is used by several APIs, for example in the activity stream, to identify a group space in which a particular activity occurred. It is made up on the following elements:

- guid—the GUID of the group space
- name—The name of the group space
- displayName—The display name of the group space

The groupSpaceReference also includes an html link, rest link, and icon link.
28.7.2 Portable Contact Types

Portable Contact types provide users with a secure 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.

This section includes the following subsections:

- Section 28.7.2.1, "name Portable Contact Type"
- Section 28.7.2.2, "address Portable Contact Type"
- Section 28.7.2.3, "organization Portable Contact Type"
- Section 28.7.2.4, "value Portable Contact Type"

28.7.2.1 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.

28.7.2.2 address Portable Contact Type

This is a portable contact type that provides information about the user's address. It is made up of the following elements:

- formatted—The formatted version of the full address
- type—The type of the address, for example, Home, Work
- streetAddress—The street address
- poBox—The post office box number
- locality—The city or locality
- region—The state or region
- postalCode—The zip code or postal code
- country—The country

Some of the elements may not be present depending on the user repository configuration and data.

28.7.2.3 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
- 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.

### 28.7.2.4 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 valid values of thumbnail

### 28.8 WebCenter REST API Examples

This section includes some examples illustrating how to use the WebCenter REST APIs. It includes the following subsections:

- Section 28.8.1, "Navigating the Message Board Hypermedia"
- Section 28.8.2, "Displaying Activity Stream Data"
- Section 28.8.3, "Using the WebCenter People Connections Service REST APIs with Ext"
- Section 28.8.4, "Updating User Status"

### 28.8.1 Navigating the Message Board Hypermedia

This section includes examples to illustrate how to navigate the REST service hypermedia. The examples show how to read messages on a message board, post messages to another user's message board, and delete unwanted messages.

This section includes the following subsections:

- Section 28.8.1.1, "Accessing the Resource Index"
- Section 28.8.1.2, "Reading Messages"
- Section 28.8.1.3, "Creating a New Message"
- Section 28.8.1.4, "Updating a Message"
- Section 28.8.1.5, "Deleting a Message"
28.8.1.1 Accessing the Resource Index
The first step is always to access the Resource Index (Example 28–10).

**Example 28–10 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 28–11).

**Example 28–11 Response to Accessing the Resource Index**
200 OK
Accept: application/json;charset=UTF-8

```
{
  "resourceType": "urn:oracle:webcenter:resourceindex",
  "links": [
    {
      "resourceType": "urn:oracle:webcenter:resourceindex",
      "capabilities": "urn:oracle:webcenter:read",
      "rel": "self",
      "href": "http://host:port/rest/api/resourceIndex"
    },
    {
      "resourceType": "urn:oracle:webcenter:messageBoard",
      "capabilities": "urn:oracle:webcenter:read",
      "href": "opaque-messageBoard-URI"
    }
  ],
  ...repeating for other services...
}
```

You can examine this list to find the URI that you require to access your message board. You should look for the link with a resourceType of urn:oracle:webcenter:messageBoard. The href for this link is the one that you require to access your message board.

For other resources rel, type, and template also help find the correct link.

28.8.1.2 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 28–12).

To read messages on a message board, you must be logged in.

**Example 28–12 Retrieving Messages from Your Message Board (GET)**
GET /opaque-messageBoard-URI

The response provides information about all the messages on your message board (Example 28–13).

**Example 28–13 Response to Retrieving Messages from Your Message Board**
200 OK
Accept: application/json;charset=UTF-8

```
{
  "resourceType": "urn:oracle:webcenter:messageBoard",
  "links": [
```
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

### 28.8.1.3 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 28–14).

**Example 28–14 Creating a Message on Another User's Message Board (POST)**

```json
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 28–15).

**Example 28–15 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-8ecedb0400a3",
  "body": "sure; see you guys at 6."
}
```
28.8.1.4 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 28–16).

Example 28–16 Updating a Message (PUT)

```plaintext
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 28–17).

**Example 28–17 Updating a Message**

```
200 OK
Content-Type: application/json;charset=UTF-8

{
  "id": "36b8464f-afda-44b5-90ad-8ecedcb040a3",
  "body": "working late; see you guys at 7."
}
```

...deleted for brevity...

28.8.1.5 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 28–18).

**Example 28–18 Deleting a Message (DELETE)**

```
DELETE opaque:message:URI
```

The response is simply a status code of 204 (Example 28–19).

**Example 28–19 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.

28.8.2 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 Demonstrations and Samples page on the Oracle Technology Network (OTN) at:


In Example 28–20:

- The `getResourceURL` method shows you how to retrieve the URI entry point for the Activity Stream service by retrieving the JSON data for the main Resource
Index (/rest/api/resourceIndex) and locating the URI for the Activity Stream resource in that data.

- The `formatMessage` method processes the Activity Stream data into a displayable format. This involves locating an individual message and replacing any template parameters in the message with the name of the object or user that corresponds to that parameter. The parameters also include links to display the object or user. They may contain links to REST services for those objects or users, if available.

---

**Note:** To get the Resource Index and Activity Stream JSON data, make Ajax requests using Javascript (and possibly a client-side scripting library like Dojo) and pass the resulting data into the appropriate methods.

---

**Example 28–20  Displaying Activity Stream Data**

```javascript
/* 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.
}
```
WebCenter REST API Examples

*/ 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;
}

28.8.3 Using the WebCenter People Connections Service REST APIs with Ext

Ext is a popular JavaScript library for building tables and forms and linking them to
REST web services. The following example illustrates how to display a list of
connections from the People Connections service REST APIs using an Ext JsonStore
and GridPanel.

For more information about Ext, see:
http://www.extjs.com/

Note: You must run Ext on a web server. If that server is not your
WebCenter instance, then you will run into cross-site scripting
problems. If you cannot run the example on your WebCenter server,
and you want to use Ext, then consider a proxy, such as Apache or
WebCenter Ensemble.

This sample is available on the Oracle WebCenter Demonstrations and Samples page
on the Oracle Technology Network (OTN) at:

This section includes the following subsections:
- Section 28.8.3.1, “Creating the HTML Page”
- Section 28.8.3.2, “Creating the JavaScript File”
Section 28.8.3.3, "The Result"

### 28.8.3.1 Creating the HTML Page

Example 28–21 shows the simple HTML page. The HTML page must reference the required Ext files and the JavaScript file, `connections.js`. The body of the page contains a destination div. The GridPanel is rendered in this div.

**Example 28–21   The HTML Page**

```html
<head>
    <link rel="stylesheet" type="text/css" href="../../extjs/resources/css/ext-all.css" />
    <script type="text/javascript" src="../../extjs/adapter/ext/ext-base.js"></script>
    <script type="text/javascript" src="../../extjs/ext-all.js"></script>
    <script type="text/javascript" src="connections.js"></script>
</head>

<body>
    <div id="connections"></div>
</body>
```

### 28.8.3.2 Creating the JavaScript File

The JavaScript file, `connections.js`, enables the population of the list when the page loads. Example 28–22 shows that all of the JavaScript is a parameter to `Ext.onReady()`.

**Example 28–22   connections.js as a Parameter to Ext.onReady**

```javascript
Ext.onReady(function() {
    //our code goes here
});
```

The JavaScript does the following:

1. First, it queries the Resource Index (Example 28–23).

   Notice that the example asks for responses in JSON format.

**Example 28–23   Querying the Resource Index**

```javascript
//Connection Information
var resourceIndexURL = "http://webcenter_server/rest/api/resourceIndex";
var peopleServiceURN = "urn:oracle:webcenter:people";
var connectionsURN = "urn:oracle:webcenter:people:person:list";
var connectionsRel = "urn:oracle:webcenter:people:person:list:connections";

//create a request for the resourceIndex
Ext.Ajax.request(
    {url: resourceIndexURL, 
     method: 'GET', 
     success: function(response, opts) {
        //locate the people connections entry point
        var index = Ext.decode(response.responseText);
        for (x in index.links) {
            if (index.links[x].resourceType == peopleServiceURN) 
                getMe(index.links[x].href);
        }
    },
    failure: function(response, opts) {
```
2. Next, the JavaScript sends an authenticated request to the People Connections entry point, and gets the Connections list URL from the response (Example 28–24). By default, the REST API uses Basic authentication for secure APIs. In the example, credentials for passing to the Connections list URL are Base64 encoded. Ext does not have a Base64 utility object itself, but one has been developed for it at: http://extjs.com/forum/showthread.php?p=167166

Example 28–24 Getting the Connections List URL

```javascript
function getMe(requestURL) {
    // Encode a token to identify yourself with secure parts of the REST API.
    var unencodedToken = "username" + ":" + "password";
    var encodedToken = "Basic " + Ext.util.base64.encode(unencodedToken);

    // call the connections entry point
    Ext.Ajax.request({
        url: requestURL,
        method: 'GET',
        success: function(response, opts) {
            // locate the link for my connections
            if (index.links[x].resourceType == connectionsURN) {
                if (index.links[x].rel == connectionsRel)
                    getMyConnections(index.links[x].href, encodedToken);
            }},
            failure: function(response, opts) {
                alert('Could not communicate with the REST server. ');
            },
            headers: {
                'Accept': 'application/json',
                'Authorization' : encodedToken
            },
    });
}
```

3. The next step is to send an authenticated request to the Connections list URL (Example 28–25).

Example 28–25 Sending an Authenticated Request to the Connections List URL

```javascript
function getMyConnections(requestURL, encodedToken) {
    Ext.Ajax.request({
        url: requestURL,
        method: 'GET',
        success: function(response, opts) {
            displayData(response, opts);
        },
        failure: function(response, opts) {
            alert('Could not communicate with the REST server. ');
        },
        headers: {
            'Accept': 'application/json',
            'Authorization' : encodedToken
        },
    });
}
```
4. Then, the JavaScript adds the response to a JsonStore (Example 28–26).

Previously, this example named displayData() as the handler for a successful call to the Connections list URL. Now, it creates the function and adds the JsonStore code. In the JsonStore, the data model for the data is defined and mapped to results expected from the REST call.

**Example 28–26 Adding the Response to a JsonStore**

```javascript
function displayData(response, opts) {
// create a datamodel to display the JSON
var store = new Ext.data.JsonStore({
  // store configs
  autoDestroy: true,
  storeId: 'forumStore',
  // reader configs
  root: 'items',
  idProperty: 'id',
  fields: ['displayName', {name:'email', mapping:'emails.value'},
           {name:'phone', mapping:'phoneNumbers[0].value'}]
});
}
```

5. The JavaScript creates an Ext GridPanel to display the JsonStore by including the code in Example 28–27 in displayData().

The grid defines columns and a title. It also includes other display options, such as the size of the table.

**Example 28–27 Creating a GridPanel to Display the JsonStore**

```javascript
// create the Grid
var grid = new Ext.grid.GridPanel({
  store: store,
  columns: [
    {id: 'conn', header: 'Connection', width: 80, sortable: true, dataIndex: 'displayName'},
    {header: 'Email', width: 120, sortable: true, dataIndex: 'email'},
    {id: 'phone', header: 'Phone', width: 120, sortable: true, dataIndex: 'phone'}
  ],
  stripeRows: true,
  autoExpandColumn: 'conn',
  height: 200,
  width: 400,
  title: 'My Connections',
  viewConfig: {scrollOffset: 2}
});
```

6. Finally, the code renders the GridPanel by including the line in Example 28–28 in displayData().

**Example 28–28 Rendering the GridPanel**

```javascript
grid.render('connections');
```
28.8.3.3 The Result

Figure 28–1 shows the list of connections from the People Connections service REST APIs.

Figure 28–1 The Connections List

<table>
<thead>
<tr>
<th>Connection</th>
<th>Email</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>monty</td>
<td><a href="mailto:monty@example.com">monty@example.com</a></td>
<td>555-555-5551</td>
</tr>
<tr>
<td>pat</td>
<td><a href="mailto:pat@example.com">pat@example.com</a></td>
<td>555-555-5552</td>
</tr>
<tr>
<td>vicki</td>
<td><a href="mailto:vicki@example.com">vicki@example.com</a></td>
<td>555-555-5553</td>
</tr>
</tbody>
</table>

28.8.4 Updating User Status

The following example shows how to use REST APIs to update a user’s WebCenter profile status.

---

**Note:** You must host the sample on a web server (for example, Apache or Oracle HTTP Server) or an application server. To avoid cross site scripting errors, you should proxy URL access to the REST service. On Apache or OHS, the conf commands would look like (change `myspaceshost` and `port` accordingly):

```
ProxyPass /webcenter/ http://myspaceshost:port/webcenter/
ProxyPassReverse /webcenter/ http://myspaceshost:port/webcenter/
ProxyPass /rest/ http://myspaceshost:port/rest/
ProxyPassReverse /rest/ http://myspaceshost:port/rest/
ProxyPreserveHost on
```

This sample is available on the Oracle WebCenter Demonstrations and Samples page on the Oracle Technology Network (OTN) at:


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 28–29) includes an input field where users can enter the new status message (`statusMessage`). Clicking the **Update Status** button (or pressing Enter) calls the `updateStatus` method to make the initial call to the Resource Index.

---

Example 28–29 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>
</body>
```
<div>
<button id="button1" onclick="updateStatus();">Update Status</button>
</div>
<div id='statusResults'></div>

Figure 28–2 shows the HTML page.

**Figure 28–2  New Status Message HTML Page**

The code in Example 28–30 retrieves the Resource Index (resourceIndexURL variable set to /rest/api/resourceIndex). The Resource Index is returned as an AJAX request object (resourceIndexRequest).

**Example 28–30  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 28–31 traverses the links in the data returned by a REST call to find a specified string (URN) that identifies the required resource type.

**Example 28–31  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 28–32** uses getResourceURL to parse the Resource Index to find the user profile URL. The data is returned as an AJAX request object (profileRequest).

**Example 28–32  Retrieving the User Profile**

```javascript
function resourceIndexCallback(data) {
    //get my user profile
    profileRequest.get(
        getResourceURL(data, 'urn:oracle:webcenter:people'),
        profileCallback,
        clearUIBusy);
}
```
Example 28–33 takes the new status message provided by the user in the HTML page (statusMessage) and uses request.put to update the status object (retrieved by again calling getResourceURL).

**Example 28–33  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, '\"');
  // 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 28–34, renders the new status object.

**Example 28–34  Rendering the New Status Object**

```javascript
function renderStatusPutResults(data) {
  var name = profile.displayName;
  // get the URL for my profile HTML page in WebCenter Spaces
  var profileURL = getResourceURL(profile, 'urn:oracle:webcenter:spaces:profile');
  var html = 'Status for &lt;a href='' + profileURL + '' target=_blank''&gt;' + name + '&lt;/a&gt; is: ' + data.note;
  // clear the UI busy state and print the new status message
  clearUIBusy(html);
}
```

Example 28–35 shows the code for disabling and reenabling the UI.

**Example 28–35  Disabling the UI**

```javascript
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 28–36 shows the library that assists in AJAX calls. Most of this is not WebCenter specific and works for any XHR requests to a service that returns JSON.

The library includes a reusable XHR object. This object supports HTTP GET, POST, PUT, and DELETE. All functions are asynchronous and take a URL and two callback functions, one for success, and one for failure. Success calls are made with a JavaScript object containing the return data. Failure calls are made with an error string. POST and PUT also take a data argument which must be a JSON string.
Example 28–36  AJAX Library

```javascript
function AjaxRequest() {
  // constructor to create an object oriented AJAX request
  var xhr = null;

  // get XHR object. Should work for IE 6+, Safari, and Mozilla based browsers
  if (window.XMLHttpRequest) {
    xhr = new window.XMLHttpRequest;
  } else {
    try {
      xhr = new ActiveXObject('MSXML2.XMLHTTP.3.0');
    } catch (ex) {
      xhr = null;
    }
  }
  if (xhr == null) {
    alert("Your browser does not support AJAX");
  }

  this.get = function(url, callback, errorCallback) {
    xhr.open('GET', url, true);
    // the REST APIs return XML by default. Please return JSON
    xhr.setRequestHeader('Accept', 'application/json; charset=utf-8');
    sendRequest(null, callback, errorCallback);
  }

  this.post = function(url, data, callback, errorCallback) {
    xhr.open('POST', url, true);
    // set headers to send and receive JSON
    xhr.setRequestHeader('Accept', 'application/json; charset=utf-8');
    xhr.setRequestHeader('Content-Type', 'application/json; charset=utf-8');
    sendRequest(data, callback, errorCallback);
  }

  this.put = function(url, data, callback, errorCallback) {
    xhr.open('PUT', url, true);
    // set 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 {
    // we got an error. Format an error string
    data = 'Error: ' + xhr.status + ' ' + xhr.statusText + '<br />
           ' + xhr.responseText;
    errorCallback(data);
}
}
Part VIII contains the following chapters:

- Section 29, "Overview of Portlets"
- Section 30, "Creating Portlets with the Oracle JSF Portlet Bridge"
- Section 31, "Creating Portlets with the Portlet Wizard"
- Section 32, "Coding Portlets"
- Section 33, "Testing and Deploying Your Portlets"
- Section 34, "Consuming Portlets"
- Section 35, "Creating Portlets with OmniPortlet"
- Section 36, "Creating Content-Based Portlets with Web Clipping"
This chapter provides an overview of portlets, and describes, with the help of examples, the use of portlets. It explains portlet anatomy and the resources to create portlets. This chapter also explains the features, technologies, and tools to help you decide which portlet building technology best suits your needs.

This chapter includes the following sections:
- Section 29.1, "Introduction to Portlets"
- Section 29.2, "Portlet Technologies Matrix"

### 29.1 Introduction to Portlets

A portlet is a reusable Web component that can draw content from many different sources. Figure 29–1 illustrates the Products portlet, a portlet that shows the products available on a shopping web site.

For information about how to create this portlet, see "Building Portlets and Wiring Them Together in Your Application" in the Oracle Fusion Middleware Tutorial for Oracle WebCenter Developers.

![Figure 29–1 The Products Portlet](image)

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

In a custom WebCenter application, a portlet may or may not be rendered in an inline frame (IFRAME). Inline frames enable the placement of a document within a rectangular region that includes scroll bars and borders.

**Note:** For more information about portlets and inline frames, see Section 34.4.12, "What You May Need to Know About IFRAMEs."

Within this IFRAME, portlets can display many types of content, including HTML, formatted text, images, or elements of an HTML form.

This section includes the following subsections:

- Section 29.1.1, "Portlet Anatomy"
- Section 29.1.2, "Portlet Resources"

### 29.1.1 Portlet Anatomy

Portlet anatomy is the visual representation of the portlet on a page. Figure 29–2 illustrates a typical portlet anatomy of a page in a custom WebCenter application. Note that the same portlet displayed in a different application could look different.

**Figure 29–2  Portlet Anatomy**

What is rendered on the page is controlled not only by the portlet’s own logic, but by the attributes of the portlet tag that binds the portlet to the page. Values for these attributes are specified at the design time of the application that consumes the portlet, rather than through the portlet’s own logic.

**Note:** For information about attributes of the `adfp:portlet` tag, see Section 34.4, "Setting Attribute Values for the Portlet Tag."
Introduction to Portlets

For example, at application design time you can specify through portlet tag attributes that the runtime portlet should display a header and that its border should be of a specified thickness and color. In the header, you can include a portlet title and an Actions menu icon. The Actions menu icon is displayed on a portlet only when the portlet header is displayed. If you choose not to display a header, then the Actions menu is displayed on a FadeIn-FadeOut toolbar that renders on a mouse rollover.

These elements are sometimes referred to as portlet chrome. The appearance of portlet chrome can be controlled through a style sheet and through style-related attributes of the adfp:portlet tag. The values of style-related attributes take precedence over styles specified through a style sheet, or skin.

Note: For information about skins and style-related attributes, see Section 9.9, "Applying Styles to Components."

Through portlet tag attributes, you can include or omit display commands on the Actions menu. Actions menu items controlled through portlet tag attributes include Maximize and Restore. Maximize causes the maximized portlet to displace all other displayed portlets. These are displayed again when a user selects Restore.

Note: The Maximize attribute is meaningful for a portlet only when the portlet is placed inside a PanelCustomizable core customizable component.

Other Actions menu items controlled through portlet tag attributes include the display or omission of the mode settings that were specified when the portlet was developed. If the portlet was built without including additional modes, then these commands do not appear on the Actions menu even when you indicate that they should through portlet tag attributes. In other words, portlet tag attributes are sometimes on/off switches that enable or disable the portlet’s own built-in functionality.

Mode settings that appear on the Actions menu include such modes as About, Help, Personalize, and Customize.

Users select Personalize to alter their personal view of the portlet. The Personalize command is displayed on the Actions menu only to authenticated users (that is, users who are logged in). It does not display to Public or unauthenticated users. You must implement some form of application security for users to be able to personalize their portlet views.

Note: If you are a developer creating portlets, and you want to test the Personalize mode without creating a complete security model for your application, see Section 37.8, "Configuring Basic Authentication for Testing Portlet Personalization."

Customization enables application administrators to edit a portlet’s default settings at runtime. All users see the results of a customization.
29.1.2 Portlet Resources

Portlet resources include the many prebuilt portlets available out of the box from many sources, including Oracle Portal, Oracle E-Business Suite, and third-party sources. Portlet resources also include programmatic portlets built through the custom WebCenter application's JSR 168 (standards-based) and Oracle PDK-Java wizards, and through other portlet-building tools. Each of these tools offers different product features that are targeted toward different developer roles.

For specific information about each tool and its benefits, see Section 29.2, "Portlet Technologies Matrix."

This section includes the following subsections:

- Section 29.1.2.1, "JSF Portlets"
- Section 29.1.2.2, "Rich Text Portlet"
- Section 29.1.2.3, "Prebuilt Portlets"
- Section 29.1.2.4, "Parameter Form and Parameter Display Portlets"
- Section 29.1.2.5, "Web Clipping"
- Section 29.1.2.6, "OmniPortlet"
- Section 29.1.2.7, "Programmatic Portlets"
- Section 29.1.2.8, "Deciding Which Tool to Use"

29.1.2.1 JSF Portlets

What Are They?
JSF portlets are created using the Oracle JSF Portlet Bridge. The Oracle JSF Portlet Bridge enables application developers to expose their existing JSF applications and task flows as JSR 168 portlets. The Oracle JSF Portlet Bridge simplifies the integration of JSF applications with WSRP portlet consumers, such as Oracle Portal. You can create JSF portlets using the JSR 168 Java Portlet Wizard by invoking it from the New Gallery.

JSF portlets do not require separate source code from that of the JSF application. Since these portlets are created using the Oracle JSF Portlet Bridge, you need only maintain one source for both your application and your portlets. Similarly, when you deploy your JSF application, JSF portlets are also deployed with it. Therefore, using the bridge eliminates the requirement to store, maintain, and deploy your portlets separately from your application.

Who Is the Intended User?
Application developers with the knowledge of Faces and WSRP.
When Should They Be Used?

JSF portlets are best suited when application developers intend to display contents from a JSF application as a portlet without hosting the entire application, or without separately building a portlet for the same. When portletized, the consumption of the portlet is same as registering any WSRP producer using their provider URLs. ADF components that is task flows, can be exposed using the Oracle JSF Portlet Bridge as well. Figure 29–3 shows two portlets, one where users can enter a department number, and one that displays the employee information for the specified department. These portlets were created from ADF task flows using the Oracle JSF Portlet Bridge.

For information about how to create these portlets, see Section 30.3, "Creating JSF Portlets: Example."

Figure 29–3  JSF Portlets

29.1.2.2 Rich Text Portlet

What Is It?

The Rich Text portlet, based on the WSRP 2.0 standard, offers browser-based rich text editing at runtime. Select Customize from the portlet’s Actions menu to invoke a toolbar with all the rich-text editing tools you require to insert, update, and format display text. Click the editor’s Submit button to save your changes and hide the toolbar. (Selecting Refresh from the portlet’s Actions menu also hides the toolbar).
With the Rich Text portlet, when an authorized user selects the Customize menu item the portlet enters edit mode while the page that contains the portlet remains in view mode.

Portlet configuration settings—available at design time—include controls for the portlet's name, description, and display settings.

**Who Is the Intended User?**
Application developers can add the Rich Text portlet to a page and provide initial content. Unless the application developer prohibits it, end users with the appropriate privileges can update the content of the Rich Text Portlet at runtime.

**When Should It Be Used?**
The Rich Text portlet is a useful tool for runtime posting of enterprise announcements and news items. It provides straightforward, easy-to-use controls for entering and formatting display text.

### 29.1.2.3 Prebuilt Portlets

**What Are They?**
Prebuilt portlets are available through Oracle's partnerships with leading system integrators, software vendors, and content providers. You can access these portlets by using the keywords portal or portlet when searching the Oracle PartnerNetwork Solutions Catalog, available at [http://solutions.oracle.com](http://solutions.oracle.com). Examples of these include portlets for the following purposes:

- Generating point-to-point driving directions
- Accessing Information Technology (IT) information from a wide variety of sources
- Viewing summary information about news, stocks, and weather

**Who Is the Intended User?**
Fully developed, downloadable portlets are best suited for use by application developers who understand how to download, install, and register producers in Oracle WebCenter Framework. They are available for use by all levels of experience.

**When Should They Be Used?**
Use prebuilt portlets when your needs are satisfied by the functions the portlets offer and the level of personalization readily available is sufficient to complete the desired task.

Consider alternatives when you want to extend or personalize the portlet, for example, when you need a different user interface or when the functionality you require is not available out of the box.

### 29.1.2.4 Parameter Form and Parameter Display Portlets

**What Are They?**
The Parameter Form and Parameter Display portlets provide a quick and easy way to pass values between components. They are provided by the WSRP Tools producer.

The Parameter Form portlet has three output parameters that are set when values are submitted in the form inside the portlet. The parameters can then be used to drive the content of other portlets. You can customize the Parameter Form portlet to determine
how many of the three fields are displayed on the form, depending on how many parameters you require.

The Parameter Display portlet enables you to quickly test the wiring from the Parameter Form portlet. However, typically you use the values passed from the Parameter Form portlet to drive the content of some other portlet, for example to pass a zip code to a weather portlet, or a stock symbol to a stock ticker portlet.

Figure 29–4 Parameter Form and Display Portlets

For more information about linking portlets, see Section 34.7, "Contextually Linking WSRP 2.0 Portlets."

Who Is the Intended User?
The Parameter Form and Parameter Display portlets are best suited for use by application developers who want to provide contextual linking between portlets on a page. These portlets can be added to a page by any user with the appropriate privileges.

When Should They Be Used?
Use the Parameter Form and Parameter Display portlets when your needs are satisfied by the functions the portlets offer and the level of personalization readily available is sufficient to complete the desired task.

Consider alternatives when you want to extend or personalize the portlet, for example, when you need a different user interface or when the functionality you require is not available out of the box.

29.1.2.5 Web Clipping

What Is It?
Web Clipping is a browser-based declarative tool that enables the integration of any Web application with a custom WebCenter application. Web Clipping is designed to provide quick integration by leveraging the Web application's existing user interface. Web Clipping has been implemented as a PDK-Java producer.

To create a Web Clipping portlet, the custom WebCenter application developer uses a Web browser to navigate to a Web page that contains desired content. Through the
Web Clipping Studio, the application developer can drill down through a visual rendering of the target page to choose the desired content.

Web Clipping supports the following:

- **Navigation through various styles of login mechanisms**, including form- and JavaScript-based submission and HTTP Basic and Digest Authentication with cookie-based session management.

- **Fuzzy matching of clippings**, if a Web clipping gets reordered within the source page or if its character font, size, or style changes, then it is still identified correctly by the Web Clipping engine and delivered as the portlet content.

- **Reuse of a wide range of Web content**, including basic support of pages written with HTML 4.0.1, JavaScript, applets, and plug-in enabled content, retrieved through HTTP GET and POST (form submission).

- **Personalization**, allowing an application developer to expose input parameters that users can modify when they personalize the portlet. These parameters can be exposed as public parameters that an application developer can map as page parameters. This feature enables users to obtain personalized clippings.

- **Integrated authenticated Web content through Single Sign-On**, including integration with external applications, which enables you to leverage Oracle Single Sign-On and to clip content from authenticated external Web sites.

- **Inline rendering**, enabling you to set up Web Clipping portlets to display links within the context of the portlet. As a result, when a user clicks a link in the Web Clipping portlet, the results display within the same portlet. You can use this feature with internal and external Web sites.

- **Proxy Authentication**, including support for global proxy authentication and authentication for each user. You can specify the realm of the proxy server and whether all users automatically log in using a user name and password you provide, each user logs in using an individual user name and password, or all users log in using a specified user name and password.

- **Resource Tunneling** of images.

- **Open Transport API** for customizing authentication mechanisms to clipped sites.

**Who Is the Intended User?**

Web Clipping is best suited for use by application developers and component developers who want to leverage an existing Web page for rapid portlet development. This portlet can be added to a page by any user with the appropriate privileges.

**When Should It Be Used?**

Use Web Clipping when you want to repurpose live content and functionality from an existing Web page and expose it in your custom WebCenter application as a portlet. Consider alternatives to change the way information is presented in the clipped portlet. That is, you do not want to control the User Interface (UI) or application flow, and you are accessing Web-based applications. For a greater level of control, use OmniPortlet's Web page data source instead of Web Clipping. (For more information, see Section 29.1.2.6, "OmniPortlet.")

The following are some examples of when you can consider using the Web Clipping portlet:

- **Stock chart portlet**, you want to create a portlet that displays the stock market's daily performance chart from your financial advisor's Web site. You could clip this information from an external Web site, even if your company is using a proxy.
Web mail portlet. Your users want to access their confidential Web mail accounts through a portlet and to display their in-boxes in the portlet.

For more information about using Web Clipping, see Chapter 36, "Creating Content-Based Portlets with Web Clipping."

29.1.2.6 OmniPortlet

What Is It?
OmniPortlet is a declarative portlet-building tool that enables you to build portlets against a variety of data sources, including XML files, character-separated value files (CSV, for example, spreadsheets), Web Services, databases, and Web pages. OmniPortlet users can also choose a prebuilt layout for the data. Prebuilt layouts include tabular, news, bullet, form, chart, or HTML. HTML layout enables OmniPortlet users to write their own HTML and inject the data into the HTML.

Figure 29–5 shows a portlet created with OmniPortlet.

For information about how to create this portlet, see "Building Portlets and Wiring Them Together in Your Application" in the Oracle Fusion Middleware Tutorial for Oracle WebCenter Developers

Figure 29–5  Product Information Portlet

Like Web Clipping, OmniPortlet supports proxy authentication, including support for global proxy authentication and authentication for each user. You can specify whether all users automatically log in using a user name and password you provide, each user logs in using an individual user name and password, or all users log in using a specified user name and password.

Who Is the Intended User?
Business users with a minimum knowledge of the URLs to their targeted data may find OmniPortlet a valuable tool.

When Should It Be Used?
Use OmniPortlet when you want to build portlets rapidly against a variety of data sources with a variety of layouts. Consider alternatives when you want complete control of the design and functionality of the portlet.

The following are some examples of when you can consider using OmniPortlet:

- RSS news feed portlet: You want to create a portlet that displays live, scrolling news information to your users. The data comes from a Really Simple Syndication (RSS) news feed, such as Oracle Technology Network Headlines. You also want the portlet to contain hyperlinks to the news source.
- **Sales chart portlet**: You want to present up-to-date information about your company’s sales results. You also want to display data in the form of a pie chart, and your company stores its sales information in a remote relational database.

For more information about OmniPortlet, see Chapter 35, "Creating Portlets with OmniPortlet."

### 29.1.2.7 Programmatic Portlets

**What Are They?**

Programmatic portlets are portlets that you write yourself, in Java, using either the standard Java Portlet Specification (JPS) or PDK-Java. Oracle WebCenter Framework provides two declarative wizards for simplifying the creation of standards-based JSR 168 portlets and Oracle PDK-Java portlets. These wizards assist in the construction of the framework within which you create the portlet. Each wizard may include easy steps for the following:

- Configuring general portlet properties
- Specifying names and search terms
- Setting allowable content types and mapping display modes
- Specifying user-customizable preferences
- Adding security roles
- Enabling default caching
- Adding initialization parameters
- Adding navigation parameters

**Who Is the Intended User?**

Use of the wizards is easy, but the creation of portlet logic is best performed by experienced and knowledgeable Java developers who are comfortable with the Java Portlet Specification or PDK-Java and who understand the configuration of producers.

**When Should They Be Used?**

Use programmatic portlets when you have very specialized business rules or logic or when you require personalized authentication, granular processing of dynamic results, and complete user interface control. Additionally, use programmatic portlets when you want to satisfy any of the following conditions:

Consider using the programmatic approach when the out-of-the-box portlets do not address your needs.

The following list provides a couple of examples of when you can consider using programmatic portlets:

- **Photo Album portlet**: You want to create a portlet that facilitates uploading, storing, and viewing user photos.

- **Shopping Cart portlet**: You want to create a portlet that facilitates the viewing and purchasing of, for example, company-branded items, such as mouse pads, pens, flash drives, tee shirts, and the like.

For more information about using programmatic portlets, see Chapter 31, "Creating Portlets with the Portlet Wizard" and Chapter 32, "Coding Portlets."
29.1.2.8 Deciding Which Tool to Use

Figure 29–6 illustrates the spectrum of portlet resources described in the previous section. Notice how one end of the spectrum is geared toward a more declarative development environment (that is, develop-through-wizard) while the other end focuses more on hand-coding. You can choose your tool depending on which type of environment is most comfortable and suitable to your skill-base.

For more information about deciding which tool to use, see Section 29.2, "Portlet Technologies Matrix."

Figure 29–6 Portlet Resources from Declarative to Coded Development

29.2 Portlet Technologies Matrix

Table 29–1 summarizes the technologies and tools you can use with Oracle WebCenter Framework. The matrix describes the tools and technologies that are covered in more detail in this guide: Oracle JSF Portlet Bridge, OmniPortlet, Web Clipping portlet, and programmatic portlets, including standards-based (JSR 168) portlets and PDK-Java portlets.

Note: While these are the primary tools for building portlets, additional tools and technologies exist, such as other Oracle products, including Oracle Reports and Oracle BI Discoverer. These other tools are not covered in this guide.

Use Table 29–1 to quickly scan all the features and characteristics of the different portlet building technologies, then see the subsequent sections for more in-depth information.
### Table 29–1 Portlet Building Technologies Comparison Matrix

<table>
<thead>
<tr>
<th>Oracle JSF Portlet Bridge</th>
<th>Web Clipping</th>
<th>OmniPortlet</th>
<th>Programmatic Portlets Standards-Based/PDK-Java</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>General Suitability</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oracle JSF Portlet Bridge exposes JSF and ADF artifacts as JSR 168 portlets.</td>
<td>A simple wizard-based tool, accessible from a browser, that assists in retrieving and presenting Web content that originates from other Web sites in a custom WebCenter application.</td>
<td>Wizard-based tool, accessible from a browser, that assists in retrieving and presenting data from a wide variety of data sources.</td>
<td>PDK-Java offers Oracle-specific application programming interfaces (APIs) for building portlets for use in custom WebCenter applications and Oracle Portal. Standards-based portlets additionally work with portals of other vendors. Oracle WebCenter Framework supports both WSRP and JSR-168 standards.</td>
</tr>
<tr>
<td><strong>Expertise Required</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No expertise required.</td>
<td>No expertise required.</td>
<td>Basic understanding of one or more supported data sources and the concepts of portlet and page parameters.</td>
<td>Java, Servlet, JSP knowledge.</td>
</tr>
<tr>
<td><strong>Supported Data Sources</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(for details, see Section 29.2.2, &quot;Expertise Required&quot;)</td>
<td>No limitations.</td>
<td>CSV, XML, Web Service, SQL, Web site, JCA.</td>
<td>No limitations.</td>
</tr>
<tr>
<td><strong>Deployment Type</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WSRP producers using Oracle JDeveloper or manually.</td>
<td>PDK-Java producers</td>
<td>PDK-Java producers</td>
<td>PDK-Java uses PDK-Java producers. Standards-based portlets use WSRP producers.</td>
</tr>
<tr>
<td><strong>Caching Style</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Expiry-based caching, validation-based caching (auto invalidate when personalized).</td>
<td>Expiry-based caching, validation-based caching (auto invalidate when personalized).</td>
<td>Expiry-based caching, validation-based caching (auto invalidate when personalized).</td>
<td>Expiry-based, validation, and invalidation caching, Edge Side Includes. Note: JSR 168 does not support validation based caching. WSRP 1.0 does. If you use a pure WSRP portlet, then validation-based caching is also supported. If you host a JSR portlet on WSRP (as is done in Oracle WebCenter Framework) then validation-based caching is not supported.</td>
</tr>
<tr>
<td><strong>Development Tool</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Portlet Creation Style</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Develop a JSF application first, convert it into a JSF portlet later.</td>
<td>design time at runtime.</td>
<td>design time at runtime.</td>
<td>Develop first, add later.</td>
</tr>
<tr>
<td><strong>User Interface Flexibility</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 29–1 (Cont.) Portlet Building Technologies Comparison Matrix

<table>
<thead>
<tr>
<th>Ability to Capture Content from Web Sites</th>
<th>Oracle JSF Portlet Bridge</th>
<th>Web Clipping</th>
<th>OmniPortlet</th>
<th>Programmatic Portlets Standards-Based/PDK-Java</th>
</tr>
</thead>
<tbody>
<tr>
<td>No.</td>
<td>No, by its nature.</td>
<td>Yes, by using the Web Page Data Source.</td>
<td>Yes.</td>
<td>For PDK-Java, use the oracle.portal.provider.v2.* package. For standards-based portlets, use the java.net package.</td>
</tr>
<tr>
<td>Depends upon the application being portletized.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ability to Render Content Inline</th>
<th>Oracle JSF Portlet Bridge</th>
<th>Web Clipping</th>
<th>OmniPortlet</th>
<th>Programmatic Portlets Standards-Based/PDK-Java</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes.</td>
<td>Yes</td>
<td>Yes</td>
<td>No. However, inline rendering can be achieved through public portlet parameters.</td>
<td>Yes.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Charting Capability</th>
<th>Oracle JSF Portlet Bridge</th>
<th>Web Clipping</th>
<th>OmniPortlet</th>
<th>Programmatic Portlets Standards-Based/PDK-Java</th>
</tr>
</thead>
<tbody>
<tr>
<td>No.</td>
<td>No</td>
<td>Yes, 2D-3D charts.</td>
<td>Yes.</td>
<td>Yes using BI Beans.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Public Portlet Parameter Support</th>
<th>Oracle JSF Portlet Bridge</th>
<th>Web Clipping</th>
<th>OmniPortlet</th>
<th>Programmatic Portlets Standards-Based/PDK-Java</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes.</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes Note: In standards-based portlets, support is available for public parameters using WSRP 2.0's navigational parameter feature along with Oracle WebCenter Framework extensions to JSR 168.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Private Portlet Parameter Support</th>
<th>Oracle JSF Portlet Bridge</th>
<th>Web Clipping</th>
<th>OmniPortlet</th>
<th>Programmatic Portlets Standards-Based/PDK-Java</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes.</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ability to Hide and Show Portlets Based on User Privileges</th>
<th>Oracle JSF Portlet Bridge</th>
<th>Web Clipping</th>
<th>OmniPortlet</th>
<th>Programmatic Portlets Standards-Based/PDK-Java</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes. No, though it is possible to apply security managers that are not exposed through the user interface (UI).</td>
<td>No, though it is possible to apply security managers that are not exposed through the UI.</td>
<td>Yes.</td>
<td>Yes. For PDK-Java, by using the Security managers. For standards-based portlets, the Servlet security model is supported by using methods such as PortletRequest.isUserInRole() and PortletRequest.getUserPrincipal().</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Multilingual Support</th>
<th>Oracle JSF Portlet Bridge</th>
<th>Web Clipping</th>
<th>OmniPortlet</th>
<th>Programmatic Portlets Standards-Based/PDK-Java</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes.</td>
<td>N/A</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Pagination Support</th>
<th>Oracle JSF Portlet Bridge</th>
<th>Web Clipping</th>
<th>OmniPortlet</th>
<th>Programmatic Portlets Standards-Based/PDK-Java</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The rest of this chapter includes the following sections that provide more in-depth detail about the different portlet building technologies:

- Section 29.2.1, "General Suitability"
- Section 29.2.2, "Expertise Required"
- Section 29.2.3, "Deployment Type"
- Section 29.2.4, "Caching Style"
- Section 29.2.5, "Development Tool"
- Section 29.2.6, "Portlet Creation Style"
- Section 29.2.7, "User Interface Flexibility"
- Section 29.2.8, "Ability to Capture Content from Web Sites"
- Section 29.2.9, "Ability to Render Content Inline"
- Section 29.2.10, "Charting Capability"
- Section 29.2.11, "Public Portlet Parameter Support"
- Section 29.2.12, "Private Portlet Parameter Support"
- Section 29.2.13, "Ability to Hide and Show Portlets Based on User Privileges"
- Section 29.2.14, "Multilingual Support"
- Section 29.2.15, "Pagination Support"
- Section 29.2.16, "Authenticating to External Applications"

### 29.2.1 General Suitability

This section describes each portlet-building technology in terms of its usage characteristics (for example, wizard-based or programmatic).

#### 29.2.1.1 Oracle JSF Portlet Bridge

The Oracle JSF Portlet Bridge is a dialog-based tool that enables page developers to expose their existing JSF applications and task flows as JSR 168 portlets. Using the
Oracle JSF Portlet Bridge to create JSF portlets from JSF applications does not require a technical background.

29.2.1.2 Web Clipping
Web Clipping is a simple wizard-based tool that assists with retrieving and presenting Web content that originates from other Web sites in a custom WebCenter application. Web Clipping does not require a technical background.

Examples of portlets you can build using Web Clipping
The examples of portlets that you can build by using Web Clipping are as follows:
- Stock chart portlet
- Web mail portlet
- News portlet containing dynamic content from an existing Web site

29.2.1.3 OmniPortlet
OmniPortlet is an easy-to-use, wizard-based tool for presenting information from a wide variety of data sources in a variety of formats. OmniPortlet runs completely in the browser. Drop OmniPortlet on a page, click the Define link, and choose a data source and a presentation format. Select from a wide variety of data sources as follows:
- Spreadsheet
- SQL
- XML
- Web Service
- Web page
OmniPortlet does not require the use of an additional development tool or a strong technical background. Even so, it can be used for building reusable, high-performing portlets.

Examples of portlets you can build using OmniPortlet
The examples of portlets that you can build by using OmniPortlet are as follows:
- RSS news feed portlet
- Sales chart portlet

29.2.1.4 Programmatic Portlets
If the wizard-based portlet building tools do not satisfy your needs, then you can build your portlets programmatically using Java. The Java Community Process standardized the Java portlet APIs in 2003. Portlets built against the Java Specification Request (JSR) 168 standard are interoperable across different portal platforms. The Java Portlet Wizard, a tool available through the Oracle WebCenter, assists with building Java portlets.

---

**Note:** When building portlets in Java, you have full control over your portlet’s functionality. For example, you can control what it looks like and how it behaves.
Examples of portlets you can build using Java
The examples of portlets that you can build by using Java are as follows:
- Discussion forum portlet
- E-mail portlet

29.2.2 Expertise Required
While some portlet building tools do not require portlet development skills, others assume a strong technical background. This section describes each tool in terms of the level of knowledge required to use it effectively.

29.2.2.1 Oracle JSF Portlet Bridge
The Oracle JSF Portlet Bridge does not require a technical background. However, you must have an understanding of Faces and WSRP.

29.2.2.2 Web Clipping
Web Clipping does not require a technical background. However, to parameterize the Web page content that you clipped, you must have an understanding of public portlet parameters and page parameters.

29.2.2.3 OmniPortlet
OmniPortlet requires a basic knowledge of the data source you want to leverage in your portlet. Table 29–2 lists the types of data sources that can be used with OmniPortlet and describes the type of information required to work with each type.

<table>
<thead>
<tr>
<th>Data Source</th>
<th>Required Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spreadsheet</td>
<td>The URL that points to the spreadsheet containing the data to display in the portlet.</td>
</tr>
<tr>
<td>SQL</td>
<td>The connection information to the data source and the SQL query that retrieves the data from the database.</td>
</tr>
<tr>
<td>XML</td>
<td>The location of the XML source and optionally the address of the XSL filter and the XML schema.</td>
</tr>
<tr>
<td>Web service</td>
<td>The Web Services Description Language (WSDL) URL, the method of the Web service, and optionally the XSL filter URL and the XML schema URL.</td>
</tr>
<tr>
<td>Web page</td>
<td>The Web page data source uses the same environment as Web Clipping. No technical background is required.</td>
</tr>
<tr>
<td>J2EE Connector Architecture</td>
<td>Although not displayed on the OmniPortlet Wizard's Type page, a J2EE Connector Architecture (JCA) 1.0 adapter is also available. JCA provides a mechanism to store and retrieve enterprise data such as that held in ERF systems (Oracle Financials, SAP, PeopleSoft, and so on).</td>
</tr>
</tbody>
</table>

29.2.2.4 Programmatic Portlets
To build Java portlets, you must know at least a subset of Java EE. Knowing HTML, Java servlets, and XML is a must, and JSP experience is recommended. Additional Java knowledge is optional, depending on the task you want to perform. Using Java portlets you can access any data source supported by the Java language.
29.2.3 Deployment Type

Before a portlet can be consumed by an application, you must first deploy it, then register the producer you’ve deployed the portlet to. As shown in Figure 29–7, portlets can be deployed through the following two producer types:

- PDK-Java producers
- WSRP producers

Within WSRP, both version 1.0 and 2.0 are supported.

PDK-Java portlets are deployed to a Java EE application server, which is often remote and communicates with the consumer through Simple Object Access Protocol (SOAP) over HTTP. JSR 168 portlets are deployed to a WSRP producer, which is also remote and communicates with the consumer through WSRP (Web Services for Remote Portlets). JSF Portlets are deployed as JSR 168 portlets, that is, to a WSRP producer, either from Oracle JDeveloper or manually.

Figure 29–7 Portlet Producer Overview

29.2.3.1 PDK-Java Producers

PDK-Java producers use open standards, such as XML, SOAP, HTTP, or Java EE for deployment, definition, and communication with applications. Figure 29–8 shows how Oracle Portal incorporates portlets from a PDK-Java producer and the PDK-Java producer communicates with custom WebCenter application using SOAP over HTTP.
There are several benefits to developing portlets and exposing them through PDK-Java producers as follows:

- Deploy portlets remotely.
- Leverage existing Web application code to create portlets.
- Specify producers declaratively.
- Use standard Java technologies (for example, servlets and JSPs) to develop portlets.

To expose your portlets using a PDK-Java producer, you must first create a producer that manages your portlets and communicates with Oracle WebCenter Framework using SOAP. To learn how to expose your portlets using a PDK-Java producer, see Section 31.2, "Creating Java Portlets."

### 29.2.3.2 WSRP Producers

Oracle WebCenter supports Web Services for Remote Portlets (WSRP) versions 1.0 and 2.0. WSRP 2.0 support is for a preliminary (that is, pre-production) version of WSRP 2.0. This emerging standard provides support for inter-portlet communication and export or import of portlet customizations.

Use of WSRP 2.0 requires use of Oracle-specific extensions. For example, portlets produced by WSRP 2.0 producers must be deployed to Oracle's container to take advantage of the benefits this newer version provides. This is because standard portlet APIs (such as JSR 286) have not yet evolved to the level of the WSRP 2.0 communication protocol.
The Producer Registration wizard is the entry point for registering both WSRP 1.0 and 2.0 producers. The wizard automatically recognizes whether WSRP 1.0 or 2.0 is in play.

Architecturally, WSRP producers are very similar to PDK-Java producers. WSRP is a communication protocol between custom WebCenter application servers and portlet containers. WSRP is a standard that enables the plug-and-play of visual, user-facing Web services with intermediary Web applications.

Being a standard, WSRP enables interoperability between a standards-enabled container based on a particular language (such as JSR 168, .NET, Perl) and any WSRP portal. So, a portlet (regardless of language) deployed to a WSRP-enabled container can be rendered on any application that supports this standard.

---

**Note:** For more information about the WSRP architecture, see "The Relationship Between WSRP and JSR 168" in Chapter 31, "Creating Portlets with the Portlet Wizard."

---

To make standard portlets (such as JSR 168, .NET, Perl) available to a custom WebCenter application, you must package them in a portlet application and deploy them to a WSRP container. To learn more about WSRP, see the WSRP and JSR 168 Standards page on the Oracle Technology Network:


To learn how to package your portlets in a WSRP container, see Chapter 33, "Testing and Deploying Your Portlets." You can also test your WSRP producers online using the Oracle Portal Verification Service:

http://portalstandards.oracle.com/portal/page/portal/OracleHostedWSRPPortal/Welcome

---

29.2.3.3 Producer Architecture

**Figure 29–9** illustrates the basic architecture of portlet producers.
When users display a page in their Web browsers, the flow of the request works as follows:

1. The user requests a page from the Web browser by entering a URL in the browser’s address field.
2. The browser transmits the request to the application over HTTP.
3. The application contacts the portlet producers that provide the portlets that display on the requested page.
4. The producers make the necessary calls to their portlets so that the portlets generate the portlet content in the form of HTML or XML code.
5. The producers return the portlet content back to the application using their relevant protocols:
   - JSR 168 portlets and JSF portlets are initialized by WSRP producers, which communicate using the WSRP 1.0 or 2.0 protocol.
   - PDK-Java portlets are initialized by PDK-Java producers, which communicate using SOAP over HTTP.

Note: For more information about the portlet and producer architecture, visit the Portlet Development page on Portal Center (http://portalcenter.oracle.com).
JSF portlets, Web Clipping, OmniPortlet, and Java portlets communicate with Oracle WebCenter Framework through either WSRP or PDK-Java producers. You must register these producers with Oracle WebCenter Framework before you can use the portlets they produce in your custom WebCenter application.

The latest versions of Web Clipping and OmniPortlet are available through Application Development Runtime Service (ADRS). For more information, see Section 3.9, "Using Integrated WLS."

### 29.2.4 Caching Style

Portlet caching is key to rapid response to user requests. Portlets implement validation-based and expires-based caching using Java Object Cache. Invalidation-based caching continues to be implemented by a Web Cache that fronts the PDK-Java producer running the Web Clipping portlet and OmniPortlet.

Caching rules can be specified at a portlet's container level, encoded in the portlet's own logic, or, for JSR 168 portlets, established through the portlet wizard. Provided it is specified, container-level caching takes over when caching is not part of the portlet code.

At the application level, Oracle WebCenter Framework supports use of a Java cache for the establishment of application-level caching rules.

When not using caching, you may find accessing various data sources with Web Clipping and OmniPortlet to be time consuming. When you enable caching at the application level, you instruct the Java cache to maintain a copy of the portlet content. When data that was previously cached is requested, no time is lost in contacting the data source and regenerating its content. Instead, the previously cached portlet content is returned.

A portlet's content weighs heavily in determining the type of caching the portlet should use. For example:

- **Expiry-based caching:** Consider using expiry-based caching when the portlet content is static or when it is not critical that the most up-to-date content be displayed. When using expiry-based caching, you must specify the caching period.

- **Validation-based caching:** Consider using validation-based caching for portlets with dynamic content that changes frequently or unpredictably. The portlet associates its content with a caching key and returns the key value along with the content. When the portlet content is requested, the portlet decides, based on the caching key, if the current content is valid. If the portlet content is valid, then it returns a response indicating that the cached content can be used (that is, the content is valid) or generates the new portlet content and returns it along with a new caching key for that content.

#### 29.2.4.1 Oracle JSF Portlet Bridge

Oracle JSF Portlet Bridge supports expiry- and validation-based caching. JSF portlets can be cached in full, or you can use Edge Side Includes (ESI) to cache fragments of portlets.

#### 29.2.4.2 Web Clipping and OmniPortlet

In addition to invalidation-based caching, expiry-based caching can be specified for the Web Clipping portlet and OmniPortlet. Additionally, these portlets are refreshed automatically when they are personalized.
29.2.4.3 Programmatic Portlets
Java portlets support expiry- and validation-based caching. These portlets can be cached in full, or you can use Edge Side Includes (ESI) to cache fragments of portlets.

29.2.5 Development Tool
This section describes the development tools you can use to build different types of portlets.

29.2.5.1 Oracle JSF Portlet Bridge
Oracle JSF Portlet Bridge/JSF Portletization dialog is used to create a JSF portlet based upon a page or a task flow.

29.2.5.2 Web Clipping and OmniPortlet
OmniPortlet and Web Clipping use a browser-based wizard as the development tool.

29.2.5.3 Programmatic Portlets
Although you can use any Java development environment to build Java portlets, it is highly recommended that you use Oracle JDeveloper, a professional IDE. While you can consider other IDEs, Oracle JDeveloper includes the Java Portlet Wizard, to minimize your Java portlet development efforts.

The Java Portlet Wizard generates a starting skeleton and file structure for both JSR 168 and PDK-Java portlets. You need only add your own business logic to the skeleton. Oracle JDeveloper can also package and deploy your applications to your Java EE container. Also, Oracle JDeveloper helps you test your portlet producer. Oracle recommends that you use the Integrated WLS, provided through Oracle WebCenter Framework, as your development Java portlet runtime environment. For more information, see Chapter 3, "Preparing Your Development Environment."

29.2.6 Portlet Creation Style
Oracle WebCenter Framework supports the following types of portlet creation (Figure 29–10):

- Develop first, add later
- Design time at runtime

Develop first, add later portlet creation is usually the task of the portlet developer; design time at runtime portlet creation is the application developer’s responsibility.
29.2.6.1 Oracle JSF Portlet Bridge
Develop a JSF application first and then portletize the application using the JSF Portletization dialog at design time.

29.2.6.2 OmniPortlet and Web Clipping
OmniPortlet and Web Clipping both offer a "design time at runtime" portlet creation style. Register the portlet producers with the application that consumes the portlets, add the portlets to an application page, run the application, and then define the portlets in-place on the page.

29.2.6.3 Programmatic Portlets
Typically programmatic portlets offer a "develop first, add later" portlet creation style. Two wizards are available through Oracle WebCenter Framework to assist with the creation of Oracle PDK-Java and JSR 168 portlets. The wizards generate the basic files required for portlet creation. The developer hand-codes the portlet logic. The development sequence for programmatic portlets is to create the portlet, deploy it to a
producer, register the producer with the application that consumes the portlet, and then add the portlet to an application page.

---

**Note:** With extensive coding, you can create "design time at runtime" Java portlets. For example, Web Clipping and OmniPortlet are both Java portlets.

---

### 29.2.7 User Interface Flexibility

This section describes the portlet building tools in terms of the control you have over the user interface.

#### 29.2.7.1 Oracle JSF Portlet Bridge

Oracle JSF Portlet Bridge ensures that the portletized JSF application functions as a JSF portlet.

#### 29.2.7.2 Web Clipping

Because of its nature, Web Clipping always displays the remote Web site content, therefore UI flexibility is not a requirement for this portlet.

#### 29.2.7.3 OmniPortlet

OmniPortlet enables you to use different pre-built layouts, such as scrolling news, tabular, and chart. You can also use the built-in HTML layout to personalize the look and feel of your portlet using HTML and JavaScript.

---

**Note:** When using JavaScript in portlets, developers must ensure that the JavaScript identifiers are qualified. That is, identifiers must be unique for each portlet instance and must not clash with the JavaScript on the page.

---

#### 29.2.7.4 Programmatic Portlets

In Java portlets, you have full control over your portlet's user interface. Your portlet is free to generate any HTML content that conforms to the rendering rules for pages.

### 29.2.8 Ability to Capture Content from Web Sites

This section describes the portlet building tools in terms of their ability to include content from other sources.

#### 29.2.8.1 Oracle JSF Portlet Bridge

Oracle JSF Portlet Bridge’s capability to capture contents from Web sites depends upon the JSF application being portletized.

#### 29.2.8.2 Web Clipping

For portlets that display content from a remote Web site as it is presented at the source location, the best tool to use is Web Clipping. Web Clipping can tolerate the changes of the source HTML page to some extent. If a clipped table moves from one place to another in the source page, then the Web Clipping engine can find the table again using the internal "fuzzy match" algorithm. Portlets built with Web Clipping can also
maintain sessions to the remote Web sites. Web Clipping also supports user personalization of HTML form values.

29.2.8.3 OmniPortlet
For portlets using the data but not the layout from a remote Web site, the best choice is OmniPortlet. Use OmniPortlet to retrieve the data, process the data (format, filter, and so on), and present it in a portlet in a tabular, chart, or news format. OmniPortlet is a powerful tool that extracts data from Web pages by using its Web Page data source.

29.2.8.4 Programmatic Portlets
Java portlets can take advantage of low-level Java networking APIs to retrieve and process content from remote Web sites. To avoid unnecessary development efforts, before choosing Java always ensure that Web Clipping or OmniPortlet are not viable options.

29.2.9 Ability to Render Content Inline
Active elements in portlets, such as links or form buttons, enable users to navigate to remote URLs. In a News portlet, for example, a user can click a hyperlink to navigate to a news site with detailed information about news of interest. For example, a user clicks a news-summary link in a News portlet, leaves the application page, and lands on the news site.

You may have a requirement to keep your users within the context of the application page by rendering the requested content within the same portlet container. For example, a user clicks a news-summary link in a News portlet, and the portlet refreshes with the detailed news article.

This maintenance of context is what rendering content inline means.

29.2.9.1 Oracle JSF Portlet Bridge
Oracle JSF Portlet Bridge converts a JSF application with multiple pages into JSF portlets. The contents of these pages are rendered inline.

29.2.9.2 Web Clipping
The Web Clipping portlet supports URL rewriting for achieving inline content rendering. It can process the links originating from the source Web site and rewrite them to achieve the desired functionality.

The following options are available:

- Select not to rewrite the URLs within the portlet, in which case clicking the links takes users out of the custom WebCenter application to the Web site that provides the clipping. Whenever the link brings the user to a place that requires authentication, the user must enter login information before the link target is displayed.

- If the Web Clipping provider is registered with an external application and the clipping requires authentication, then you can instruct Web Clipping to rewrite all URLs within the portlet to point to the Login Server. In this case, navigation causes the user to leave the custom WebCenter application, while also using the Login Server to log the browser into the External Application.

- Select to rewrite all URLs within the portlet (inline rendering) to point back to the page so that all browsing within the Web Clipping portlet remains within the custom WebCenter application. If the Web Clipping provider is registered with an
External Application, then this causes the Web Clipping provider to log itself in to the External Application. In this case, the navigation within the custom WebCenter application through the Web Clipping provider is authenticated in the External Application.

### 29.2.9.3 OmniPortlet
Rendering content inline is not supported, but you can achieve inline rendering using public portlet parameters.

### 29.2.9.4 Programmatic Portlets
As you have full control over the links and buttons in Java portlets, you can easily implement the inline rendering functionality. To achieve inline rendering, you must append the private portlet parameters to the page URL.

If you use Struts in your portlet, then the PDK-Struts integration framework renders your content always in the same portlet container. Oracle recommends, however, that you use ADF Faces navigation for your new custom WebCenter application portlets.

If your portlet consists of multiple JSPs (for example, several steps in a survey or wizard), then your portlet can make use of a special parameter to specify at runtime the JSP to use to render the content.

### 29.2.10 Charting Capability
This section describes the portlet building tools in terms of their charting capability.

#### 29.2.10.1 Oracle JSF Portlet Bridge
Oracle JSF Portlet Bridge does not have charting capability. However, it supports portletization of a JSF application that contains ADF charts.

#### 29.2.10.2 Web Clipping
Web Clipping clips pre-existing content. So, while it does not create charts, it can retrieve and present HTML content that contains charts.

#### 29.2.10.3 OmniPortlet
OmniPortlet supports bar, line, and pie chart types. Charts are dynamically generated images, which can include hyperlinks.

#### 29.2.10.4 Programmatic Portlets
You can create sophisticated charts programmatically in Java portlets using Oracle’s Business Intelligence (BI) Beans.

---

**Note:** Oracle Reports and Oracle Discoverer portlets use BI Beans to create professional graphs.

### 29.2.11 Public Portlet Parameter Support
Typically, a portlet’s state is opaque (private); however, in Oracle WebCenter Framework portlets can describe public inputs (parameters) so values can be coordinated by the consuming application with other constituents of that application.

Inputs can include, public portlet parameters and private portlet parameters. These can be described as follows:
■ **Public portlet parameters:** Use public portlet parameters to pass values to a portlet. Public parameters can assist with rendering portlet content that is specific to a particular page or user. Portlet parameters are created by the component developer and then exposed to the application developer through the user interface. After adding a portlet to a page, application developers can assign values to public portlet parameters to make the information displayed in the portlet specific to the page.

Assigned values can be specific (such as a constant), a system variable (for example, the user name), or a page parameter. At runtime, the portlet receives the values from the sources specified.

■ **Private portlet parameters:** Use private portlet parameters to implement internal navigation in a portlet. Parameters are passed to the portlet every time the page is requested. Private portlet parameters can be passed exclusively from the portlet instance to the same portlet instance. Private portlet parameters do not require a full page refresh.

---

**Note:** A Refresh action is available for inclusion on the portlet’s Actions menu (`isNormalModeAvailable`). It refreshes the portlet without triggering a full page refresh. For details about setting this action programmatically, see Section 34.4, "Setting Attribute Values for the Portlet Tag."

---

Portlets supporting public portlet parameters enable application developers to tailor data input for each portlet instance. The component developer can focus on the portlet logic, while the application developer can address the interaction between the application page and its portlets.

All portlet building technologies discussed in this chapter (OmniPortlet, Web Clipping, and programmatic portlets) support public portlet parameters. OmniPortlet and Web Clipping provide complete support through their wizard interface. You can add public portlet parameter support to your programmatic portlets programmatically or with the Java Portlet wizard.

### 29.2.12 Private Portlet Parameter Support

This section describes the portlet building tools in terms of their support for private parameters.

#### 29.2.12.1 Oracle JSF Portlet Bridge

In your JSF portlets, you can implement internal navigation using private portlet parameters.

#### 29.2.12.2 OmniPortlet and Web Clipping

With the OmniPortlet and Web Clipping portlets, component developers do not have access to private portlet parameters.

#### 29.2.12.3 Programmatic Portlets

In your Java portlets, you can implement internal navigation using private portlet parameters.
29.2.13 Ability to Hide and Show Portlets Based on User Privileges

This section describes the portlet building tools in terms of their support for authorization functionality.

29.2.13.1 Oracle JSF Portlet Bridge
Oracle JSF Portlet Bridge supports the standard servlet mechanisms.

29.2.13.2 Web Clipping and OmniPortlet
Dynamically hide and show portlets built with Web Clipping and OmniPortlet by using security managers. Although Web Clipping and OmniPortlet do not expose security managers through the user interface, they make them available for editing through their XML provider definition files.

29.2.13.3 Programmatic Portlets
PDK-Java provides security managers for Java portlets. For example:

- **Group security manager**: The group security manager controls access to portlets based on group membership. For example, it shows the portlet to users who are members of a specified group, and hides the portlet from non-members.

- **Authentication level security manager**: The authentication level security manager controls access to the portlets based on authentication level. For example, it shows the portlet to authenticated users, and hides it from public users.

JSR 168 portlets support the standard servlet mechanisms.

29.2.14 Multilingual Support

This section describes the portlet building tools in terms of their support for other languages.

Web Clipping, OmniPortlet, JSF portlets, and Java portlets display textual information in the language selected by the end user.

29.2.15 Pagination Support

Support for pagination is useful when a portlet must display a relatively large set of records.

29.2.15.1 Oracle JSF Portlet Bridge
Oracle JSF Portlet Bridge supports pagination, if the application portletized has pagination implemented in it.

29.2.15.2 Web Clipping
Web Clipping does not support pagination.

29.2.15.3 OmniPortlet
OmniPortlet does not support pagination.

29.2.15.4 Programmatic Portlets
With Java, portlet pagination can be implemented programmatically.
29.2.16 Authenticating to External Applications

This section describes the portlet building tools in terms of authentication for external applications.

29.2.16.1 Oracle JSF Portlet Bridge

Since any portletized application can function as a WSRP portlet, the external application support is feasible through custom user attributes.

29.2.16.2 Web Clipping

Web Clipping’s integration with the external application framework provides a fully automated mechanism to store passwords to external Web sites. All you must do is provide an External Application ID when registering the Web Clipping producer.

29.2.16.3 OmniPortlet

OmniPortlet enables you to store connection information when the data source is password protected. The credentials to access the data source can either be shared across all users or saved individually for each user. OmniPortlet can storing database credentials and HTTP basic authentication user name-password pairs. Credentials are stored in a secured metadata services repository.

29.2.16.4 Programmatic Portlets

Java portlets support programmatic integration with the external application framework and any LDAP server, such as Oracle Internet Directory.
This chapter explains how to use the Oracle JSF Portlet Bridge to expose an application as a portlet.

This chapter includes the following sections:

■ Section 30.1, "Introduction to the Oracle JSF Portlet Bridge"
■ Section 30.2, "Creating a Portlet from a JSF Application"
■ Section 30.3, "Creating JSF Portlets: Example"

30.1 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 168 portlets.

---

**Note:** Unless otherwise noted, the term JSF applications encompasses Oracle ADF applications as well.

---

The Oracle JSF Portlet Bridge:

■ Simplifies portlet development by enabling you to provide portlet functionality using JSF rather than relying on the JSR 168 portlet APIs.
■ Simplifies exposing your JSF application to JSR 168 portlet consumers, such as Oracle Portal and Oracle WebCenter Spaces.
■ Eliminates the requirement to store, maintain, and deploy your portlets separately from your application by enabling the application to run simultaneously as a regular web application and as a portlet from the same installation.
■ Enables you to create portlets at a more granular level by exposing task flows as portlets. Because portletized task flows are WSRP portlets, this also enables you to use distributed task flows.

---

**Note:** The Oracle JSF Portlet Bridge is based on and conforms to JSR 301. JSR 301 is the standards effort to define the functionality for the Portlet 1.0 Bridge for JavaServer Faces. Oracle is the specification lead for this standard. More information is available at:

30.2 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 168 portlet. As part of this configuration, you indicate the initial JSF view (or task flow view) to invoke when the portlet is rendered. From that point on the Oracle JSF Portlet Bridge works with the JSF application to navigate through the additional views that are reachable from this initial view. So in the typical situation when you are exposing the entire JSF application as the portlet, you configure the Oracle JSF Portlet Bridge to render the application’s initial view in the portlet and the rest of the navigation works naturally within that same portlet.

This section includes the following subsections:

- Section 30.2.1, "How to Create a JSF Portlet Based on a Page"
- Section 30.2.2, "How to Create a JSF Portlet Based on a Task Flow"
- Section 30.2.3, "How to Test a JSF Portlet"
- Section 30.2.4, "What Happens at Runtime"
- Section 30.2.5, "What You May Need to Know When Creating a JSF Portlet"

30.2.1 How to Create a JSF Portlet Based on a Page

The simplest way to create a portlet from a JSF application is to generate a portlet based upon a page.

To create a JSF portlet from an existing application page:

1. In the JDeveloper Application Navigator, open the application that contains the JSF page to portletize.

2. Right-click the page to portletize and choose Create Portlet Entry.

3. In the Create Portlet Entry dialog (Figure 30–1), in the Portlet Name field, enter a name for the portlet.
Figure 30–1 The Create Portlet Entry Dialog for a Page

4. In the **Display Name** field, enter a descriptive name for your portlet.

5. In the **Portlet Title** field, enter a descriptive title for your portlet.

   The portlet title is displayed in the Resource Palette or Application Resources panel, so make the title something to help users decide whether the portlet is useful to them. The portlet title is also displayed on the portlet header when the portlet appears on a page.

6. In the **Short Title** field, enter a shorter title for your portlet. This short title is displayed on the portlet header when the portlet appears on a page on a mobile device.

7. In the **Description** field, enter a description for your portlet.

8. Select **Create navigation parameters for events** to create navigation parameters for any events exposed by the page.

   Navigation parameters enable a portlet to communicate with the page on which it resides and with other portlets on that page. If you select this option, navigation parameters are added to the `oracle-portlet.xml` file.

9. Click **OK**.

   The files `portlet.xml` and `oracle-portlet.xml` are created.

   The `portlet.xml` file contains the portlet entry (Example 30–1) and is opened ready for viewing or editing.

**Example 30–1 Generated Portlet Entry for a Page**

```xml
<?xml version='1.0' encoding='UTF-8' standalone='yes'?>
<portlet-app version='1.0' xsi:schemaLocation="http://java.sun.com/xml/ns/portlet/portlet-app_1_0.xsd http://java.sun.com/xml/ns/portlet/portlet-app_1_0.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/portlet/portlet-app_1_0.xsd">
  <portlet id="adf_jsf__MyJSFPage_jsp">
    <description>myApp_myPage_jspx</description>
    <portlet-name>myApp_myPage_jspx</portlet-name>
  </portlet>
</portlet-app>
```
The page you selected earlier is used as the entry point for the portlet View mode. This is indicated in the portlet.xml file by the javax.portlet.faces.defaultViewId.view initialization parameter. You can manually edit the portlet.xml file to define the pages for other default portlet modes:

- **Edit mode**: javax.portlet.faces.defaultViewId.edit
- **Help mode**: javax.portlet.faces.defaultViewId.help

**Note:** The value for the defaultViewId is relative to the application context root and must always start with a /. For example, in Example 30–1, 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>.

The oracle-portlet.xml file is an Oracle extension of portlet.xml to support WSRP 2.0 features such as navigation parameters used for inter-portlet communication. If you selected Create navigation parameters for events,
navigation parameters are added to this file to contain the payload for any events exposed by the page.

### 30.2.2 How to Create a JSF Portlet Based on a Task Flow

An advantage of using Oracle ADF is task flows that provide a modular approach for defining control flow in an application. Instead of representing an application as a single large JSF page flow, you can break it up into a collection of reusable task flows. In each task flow, you identify application activities, the work units that must be performed in order for the application to complete. An activity represents a piece of work that can be performed when running the task flow.

Task flows can be unbounded or bounded:

- An unbounded task flow is a set of activities, control flow rules, and managed beans interacting to allow a user to complete a task. An unbounded task flow consists of all activities and control flows in an application that are not included within any bounded task flow.

- A bounded task flow is a specialized form of task flow, having a single entry point and one or more exit points. It contains its own set of private control flow rules, activities, and managed beans. An Oracle ADF bounded task flow allows reuse, parameters, transaction management, and reentry. It can have zero to many exit points.

A typical application is a combination of an unbounded and one or more bounded task flows. The application can then call bounded task flows from activities within the unbounded task flow. For more detailed information about bounded and unbounded task flows, see the Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework.

This section includes the following subsections:

- Section 30.2.2.1, "Creating a Portlet From a Task Flow Using the Create Portlet Entry Dialog"
- Section 30.2.2.2, "Creating a Portlet From a Task Flow Using the Manage Portlet Entries of Task Flows Dialog"

#### 30.2.2.1 Creating a Portlet From a Task Flow Using the Create Portlet Entry Dialog

Use the Create Portlet Entry dialog to create a portlet from a single task flow.

**To make a portlet from a task flow using the Create Portlet Entry dialog:**

1. In the JDeveloper Application Navigator, open the JSF application that contains the task flow from which you want to make a portlet.

2. Right-click the task flow toportletize and choose **Create Portlet Entry**.

3. In the Create Portlet Entry dialog (Figure 30–2), in the **Portlet Name** field, enter a name for the portlet.
4. If the task flow is unbounded, the **Entry Point View** dropdown list displays all the view activities of the task flow. From this dropdown list, choose the view activity to use as the entry point for the portlet. By default, the first view activity in the task flow is chosen.

If the task flow is bounded, you do not see the **Entry Point View** dropdown list.

5. In the **Display Name** field, enter a descriptive name for your portlet.

6. In the **Portlet Title** field, enter a descriptive title for your portlet.

   The portlet title is displayed in the Resource Palette or Application Resources panel, so make the title something to help users decide whether the portlet is useful to them. The portlet title is also displayed on the portlet header when the portlet appears on a page.

7. In the **Short Title** field, enter a shorter title for your portlet. This short title is displayed on the portlet header when the portlet appears on a page on a mobile device.

8. In the **Description** field, enter a description for your portlet.

9. Select **Create navigation parameters for events** to create navigation parameters for any events exposed by the task flow.

   Navigation parameters enable a portlet to communicate with the page on which it resides and with other portlets on that page. If you select this option, navigation parameters are added to the *oracle-portlet.xml* file.

10. Click **OK**.

    When your portlet has been created, you should receive a message that says:

    *New portlet has been successfully created*

    In addition, the files *portlet.xml* and *oracle-portlet.xml* are created. The *portlet.xml* file contains the portlet entry (Example 30–2) and is opened ready for viewing or editing.
Example 30–2  Generated Portlet Entry for a Task Flow

```xml
<portlet id="adf_taskflow_task-flow-definition">
<description>task-flow-definition</description>
<portlet-name>task-flow-definition</portlet-name>
<display-name>task-flow-definition</display-name>
<portlet-class>
oracle.portlet.bridge.adf.application.ADFBridgePortlet
</portlet-class>
<init-param>
<name>javax.portlet.faces.defaultViewId.view</name>
</init-param>
</portlet>
</portlet>
```

**Note:** The `portlet.xml` file can include multiple portlets and can have a combination of pages and task flows exposed as portlets.

The `oracle-portlet.xml` file is an Oracle extension of `portlet.xml` to support WSRP 2.0 features such as navigation parameters used for inter-portlet communication. If you selected **Create navigation parameters for events**, navigation parameters are added to this file to contain the payload for any events exposed by the task flow. If your task flow includes input parameters, corresponding portlet navigation parameters are created in `oracle-portlet.xml` to be used for inter-portlet communication.

30.2.2.2 Creating a Portlet From a Task Flow Using the Manage Portlet Entries of Task Flows Dialog

If your project includes a lot of task flows, you may find it easier to select the task flows to create as portlets from a list. You can do this using the Manage Portlet Entries of Task Flows dialog. This dialog also lets you create portlets from multiple task flows at the same time, rather than having to create them individually.

To create a portlet from a task flow using the Manage Portlet Entries of Task Flows dialog

1. From the main menu, choose **File** and then **New**.
2. In the New Gallery, expand **Web Tier**, select **Portlets** and then **Manage Portlet Entries of Task Flows**, and click **OK**.
3. In the Manage Portlet Entries of Task Flows dialog, use the shuttle buttons to select which task flows you want to create as portlets.
4. Click **OK** to create portlets for the selected task flows.

5. If the task flow exposes any events, you must manually add navigation parameters to the `oracle-portlet.xml` file to contain the payload for those events.

### 30.2.3 How to Test a JSF Portlet

When you have created your JSF portlet you can test it using the Integrated WebLogic Server that comes packaged with JDeveloper.

**To test a JSF portlet:**

1. From the main menu, choose **Run** and then **Start Server Instance**.
   
   It may take a few moments for the Integrated WLS to start. When the instance has started, you should see a message similar to the following in your Log panel:

   ```
   DefaultServer started
   
   For more information, see Section 3.9, “Using Integrated WLS.”
   ```

2. You are now ready to run or deploy your portlet application to the Integrated WLS. Because your web application and portlet application are one and the same (the portlet application is your existing web application with additional portlet artifacts), you can run or deploy your web application as you normally would or you can run or deploy your portlet following the instructions in Section 33.2.1, “How to Test JSR 168 Portlets on Integrated WebLogic Server.”

   Note the distinction between Run and Deploy in the note in that section. Deploy provides a more persistent testing scenario.

3. When deployed, you can view the Producer Test Page by going to:

   ```
   http://host:port/context-root/info
   ```

   Post deployment you must verify that the application works correctly as a web application before it is consumed as a portlet application. For example, verify that the page you portletized earlier works:

   ```
   ```
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 34.2.1, "How to Register a WSRP Portlet Producer."

---

**Note:** Because the Oracle JSF Portlet Bridge uses WSRP v2 features, you must 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 34.3, "Adding Portlets to a Page."

### 30.2.4 What Happens at Runtime

After having created and tested the JSF portlet you can deploy the application to your production environment.

---

**Note:** Ensure that you deploy your application to a Java EE container with the Oracle Portlet Container installed. The Integrated WLS has the container installed, which is why it is recommended for testing.

---

After successful deployment, you can access both the application and the portlet producer test page. For example, the application URL would be similar to:

```
http://host:port/appcontextroot/faces/pagename.jspx
```

And the portlet producer test URL would be similar to:

```
http://host:port/appcontextroot/info
```

For more information, see Section 34.2.1, "How to Register a WSRP Portlet Producer."

### 30.2.5 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 168 portlet markup fragment rules. If you have chosen to use Oracle ADF, the markup in your page comes mainly from the Oracle ADF Faces components, most of which naturally render markup in a style that is compatible with JSF portlets.

For those components that might cause problems in a portlet environment, the application developer must take special care. Some components generate markup that conflicts with the portlet environment and hence restricts their use. Other components may allow program control (inputs) that enable developers to introduce values that conflict with the portlet environment. In this latter case, you as the developer must be aware of the potential to publish the page as a portlet and therefore properly encode a value.

This section includes the following subsections:

- Section 30.2.5.1, "General Guidelines"
- Section 30.2.5.2, "Portlet Guidelines"
- Section 30.2.5.3, "Security Guidelines"
30.2.5.1 General Guidelines

The guidelines that follow lay out the issues of which you should be aware as you code Oracle ADF pages that you may later choose to publish as portlets.

- Prior to creating a portlet from your JSF application, the application, including all of its pages and task flows, must run properly after you deploy it to Integrated WLS. If it cannot run as a regular web application, it cannot run as a portlet producer either.

- When writing an application that is supposed to run in both servlet and portlet environments, avoid casting to `HttpServletRequest` objects (or you get a `ClassCastException` when running as a portlet). Instead, use the abstraction provided by the `FacesExternalContext` object. For example, instead of:

  ```java
  HttpServletRequest request = 
  FacesContext().getCurrentInstance().getExternalContext().getRequest();
  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;
      }
  }
  ```

- When consuming a JSF Portlet, the consumer application automatically renders the portlet content within an inline frame (IFRAME). The implication is that any inline popup is then confined within the IFRAME. You should take this into consideration when specifying the size of the portlet.

- If your application is secured, ensure that you have secured identity propagation as described in Section 37.12, "Securing Identity Propagation Through WSRP Producers with WS-Security."

- Deep links are not directly supported. A by-product of the JSR 168 portlet container implementation on WSRP is that session cookie management is proxied by the consuming application rather than the client. Portlets that deep link to their full service application usually rely on shared session state to allow the transition from the current portlet context. As most applications rely on maintaining session context with a cookie, the current architecture prevents such state sharing. A deep link from the client directly to the producer server to invoke the application does
not establish a session cookie between the consumer and the producer, hence a second session is established. Applications that are required to share such state must implement their own schemes for transferring the data between the two contexts. A common implementation is to write this state to a reachable location and pass a reference to this state in the deep link.

■ Java EE login is not supported. Java EE applications can be configured with different authentication techniques, such as Basic and Digest. As portlets are fragments incorporated into a consumer’s page, it is generally expected that direct authentication occurs between the client and the consumer, not the client and the portlet producer. As a result, these authentication techniques are not supported in JSR 168. In the JSR 168 case, Java EE authentication occurs through WS-Security mechanisms that allow the Web service consumer to be authenticated and verified by the producer, and propagate the user identity for user authentication and authorization. Published Oracle ADF artifacts must not contain login links that trigger Java EE authentication.

■ For applications that take a long time to render, consider increasing the time out period when you register a producer that was created by the Oracle JSF Portlet Bridge. Note however that the time out period specified during producer registration is limited by the maximum time out period (maximumTimeout element) specified in adf-config-xml.

30.2.5.2 Portlet Guidelines

The portlet guidelines are as follows:

■ For resources and links, you must specify the location relative to the web-app-context-root. Otherwise, your images and other resources cannot be found by the portlet. Do not use relative (../) path notation. Portlets in Oracle WebCenter 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 168 container. The JSR 168 specification reflects this by mandating that all resource URLs either be absolute or context path relative.

■ Do not redirect or forward a request within your JSP. JSR 168 only supports requestDispatcher.include(). The use of httpServletResponse.sendRedirect() or requestDispatcher.forward() results in exceptions and errors. To work properly in a portlet environment, you must implement JSF navigation rules in faces-config.xml or Oracle ADF task flow control flow rules.

■ To minimize overall memory consumption in the application when running as a portlet, you should only store the minimal amount of data in the request scope.

■ If you portletize an application that contains downloadable resources, the portlet container may rewrite the file name in such a way that it may be too long for the browser to open or save. In such a case, you can save the resource to a different name and open the file using the appropriate program directly.

30.2.5.3 Security Guidelines

The security guidelines are as follows:

■ When you use the Create Portlet Entry or Manage Portlet Entries of Task Flows dialog to portletize a task flow in an Oracle ADF secured application (the security scheme being Authentication and Authorization), you must manually grant permission to the following wrapper task flow in the jazn-data.xml file of the producer application:
If you do not grant the permissions, the portlet does not render.

For example, if you have an application with two roles: authenticated-role, with view permission; and testrole, with customize, edit, grant, personalize, and view permissions, you must add the following entries inside the `<permissions>` tag of each role:

- For authenticated-role:
  ```
  <permission>
      <class>oracle.adf.controller.security.TaskFlowPermission</class>
      <name>/WEB-INF/adfp-portlet-bridge-container.xml#adfp-portlet-bridge-container</name>
      <actions>view</actions>
  </permission>
  ```

- For testrole:
  ```
  <permission>
      <class>oracle.adf.controller.security.TaskFlowPermission</class>
      <name>/WEB-INF/adfp-portlet-bridge-container.xml#adfp-portlet-bridge-container</name>
      <actions>customize,edit,grant,personalize,view</actions>
  </permission>
  ```

If you are portletizing a task flow or page that has role based authorization (that is, the task flow or page has been granted certain roles), WS-Security is required to propagate the user correctly. If you set up WS-Security, the JSF portlet does not render the content due to lack of permissions.


When registering the producer with the consuming application, you must ensure that you set the appropriate security properties. For more information, see Section 34.2.1, "How to Register a WSRP Portlet Producer."

### 30.2.5.4 JSF Guidelines

The JSF guidelines are as follows:

- When using the `h:commandLink` JSF standard HTML component ensure that you set the following `context-param` in `web.xml` so that the JSF Reference Implementation does not generate any external JavaScript resource. This is to work around an issue in the JSF Reference Implementation where the reference to the JavaScript resource is not properly encoded.

  ```
  <context-param>
      <param-name>com.sun.faces.externalizeJavaScript</param-name>
      <param-value>false</param-value>
  </context-param>
  ```

### 30.2.5.5 Oracle ADF Guidelines

The Oracle ADF guidelines are as follows:
If you are portletizing an Oracle ADF task flow, ensure that you are able to consume that task flow by dropping it onto a page as a region and running the page. This ensures that the task flow runs correctly before portletizing it.

ADF Faces Dialog Framework is not supported. If your application includes any buttons or icons that launch secondary browser windows, then the contents of the new windows are not displayed properly when the application is run as a portlet. As such, you should avoid using these components if you plan to portletize your application. Examples of components that launch secondary windows are:
- `<tr:inputDate>`
- `<tr:inputColor>`
- `<tr:popup>`
- the `useWindow` attribute of `<af:commandButton>`

Portletization of pages that contain Oracle Composer components is not supported.

The `<af.fileDownloadActionListener>` component is not supported.

Oracle ADF components/code that handle `prepareModel` must be idempotent. Any code running during the `prepareModel` phase must be rerunnable without effect to the underlying data/business logic. When running in the portlet environment, `prepareModel` is called twice during the complete JSF life cycle as opposed to being called once when running within a regular web application.

The reason for this difference is that the Oracle JSF Portlet Bridge runs JSF in two requests not one. It is implemented as if every JSF request redirected before the render phase. The consequence of this approach is that the JSF `restoreView` phase is called both when the request is first submitted and then again when request to render is received.

Do not access or reference request parameters from model code except in page parameters. Besides being a cleaner MVC2 implementation, following this guideline avoids problems when such artifacts are published as portlets. Where regular JSF artifacts run their entire 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}"
/>

<invokeAction id="doExecuteWithParams"
    Binds="ExecuteWithParams"
    Refresh="renderModel"
    RefreshCondition="${param.id != null}"
/>
```
- Do not reference page parameters in the `prepareModel` phase. This issue relates to the same problem just described for request parameters. Generally, page parameters depend on request parameters and are evaluated during the `restoreView` phase. As this phase is called a second time during portlet rendering and request parameters are not available, such use fail. Instead, move any dependency on a page parameter value into the model before JSF transitions from its execute phases to its render phase.

- If you are portletizing an application that contains only Trinidad components (`<tr:` tags only), then you must manually include the following libraries in your project:
  - `jdeveloper\modules\oracle.adf.view_11.1.1\adf-richclient-api-11.jar`
  - `jdeveloper\modules\oracle.adf.view_11.1.1\adf-richclient-impl-11.jar`

  This is so that the URLs to icons in the style sheet generated for the producer are encoded correctly.

- Because a portlet is a naming container, special consideration should be taken when using ADF Faces client-side APIs to find components. An example component ID:
  - When run as a regular web application: `id="demoTemplate:popup"`
  - When run as a portlet 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:

    ```javascript
    function showPopup(event) {
        event.cancel();
        // var popup = 
        // AdfPage.PAGE.findComponentByAbsoluteId("demoTemplate:popup");
        var source = event.getSource();
        var popup = source.findComponent("popup");
        popup.show({align: "after_end", alignId: "button");
    }
    ```
  - Use a relative path for `clientId` instead of an absolute path (starting with a `:`). For example, use:

    ```af:showPopupBehavior popupId="demoTemplate:iteratorpop" triggerType="mouseHover"/>
    ```

    Instead of:

    ```af:showPopupBehavior popupId=":demoTemplate:iteratorpop" triggerType="mouseHover"/>
    ```

    - For more information, see "What You May Need to Know About Naming Containers" in the *Oracle Fusion Middleware Web User Interface Developer’s Guide for Oracle Application Development Framework*. 

---

**Creating a Portlet from a JSF Application**

30-14  Oracle Fusion Middleware Developer’s Guide for Oracle WebCenter
30.3 Creating JSF Portlets: Example

This section includes the following subsections that step you through an example of creating JSF portlets:

- Section 30.3.1, "How to Create the Example Application"
- Section 30.3.2, "How to Create Portlets from the Task Flows in the Example Application"
- Section 30.3.3, "How to Consume the JSF Portlets from the Example Application"

30.3.1 How to Create the Example Application

The example application consists of a master task flow for entering a department number, and a detail task flow that displays the employees of the department provided in the master task flow. As you work through the example, remember to periodically save your progress.

To create the example application:
1. From the JDeveloper main menu, choose File and then New.
2. In the New Gallery, expand General, select Applications and then Fusion Web Application (ADF), and click OK.
3. On the Name your application page of the Create Fusion Web Application (ADF) wizard, in the Application Name field, enter PortletBridgeApplication and then click Finish.
4. In the Application Navigator, right-click the ViewController project and choose New.
5. In the New Gallery, expand General, select Java and then Java Class, and click OK.
6. In the Create Java Class dialog, in the Name field, enter Department.
7. In the Package field, enter hr and click OK.
8. Replace the code for Department.java with that in Example 30–3.

Example 30–3 Department.java

```java
package hr;
public class Department {
    private int deptno = 0;
    public Department() {
        super();
    }
    public void setDeptno(int deptno) {
        this.deptno = deptno;
    }
    public int getDeptno() {
        return deptno;
    }
}
```

9. Repeat steps 4 through 7 to create another Java class with the name DepartmentBean.
10. Replace the code for DepartmentBean.java with that in Example 30–4.
Example 30–4  DepartmentBean.java

```java
package hr;
import java.util.Map;
import oracle.adf.share.ADFContext;
public class DepartmentBean {
    private static String DEPARTMENT_KEY = "HR_DEPARTMENT";
    private static String DEPARTMENT_DEFAULT_VALUE = "10";
    public DepartmentBean() {
        Department dept = new Department();
        dept.setDeptno(10);
        Map sessionScope = ADFContext.getCurrent().getSessionScope();
        if (sessionScope.get(DEPARTMENT_KEY) == null)
            sessionScope.put(DEPARTMENT_KEY, dept);
    }
    public int getDeptno() {
        Map sessionScope = ADFContext.getCurrent().getSessionScope();
        Department dept = (Department)sessionScope.get(DEPARTMENT_KEY);
        return dept.getDeptno();
    }
    public void setDeptno(int deptno) {
        Map sessionScope = ADFContext.getCurrent().getSessionScope();
        Department dept = (Department)sessionScope.get(DEPARTMENT_KEY);
        dept.setDeptno(deptno);
    }
    public String selectDepartmentString() {
        Map sessionScope = ADFContext.getCurrent().getSessionScope();
        Department dept = (Department)sessionScope.get(DEPARTMENT_KEY);
        String deptno = (dept == null) ? DEPARTMENT_DEFAULT_VALUE :
            String.valueOf(dept.getDeptno());
        System.out.println("DepartmentBean.selectDepartmentString:Department number is " + deptno);
        return deptno;
    }
}
```

11. Repeat steps 4 through 7 again to create another Java class with the name Employee.

12. Replace the code for Employee.java with that in Example 30–5.

Example 30–5  Employee.java

```java
package hr;
public class Employee {
    private String firstName;
    private String lastName;
    private String title;
    private int deptno;
    public Employee(String firstName, String lastName, String title, int deptno) {
        setFirstName(firstName);
        setLastName(lastName);
        setTitle(title);
        setDeptno(deptno);
    }
    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }
    public String getFirstName() {
        return firstName;
    }
```
13. Repeat steps 4 through 7 again to create another Java class with the name EmployeesBean.

14. Replace the code for EmployeesBean.java with that in Example 30–6.

**Example 30–6  EmployeesBean.java**

```java
package hr;
import java.util.ArrayList;
import oracle.adf.view.rich.context.AdfFacesContext;
public class EmployeesBean {
    private static final String DEPARTMENT_NUMBER_KEY = "DEPTNO";
    private static final int DEPARTMENT_NUMBER_NULL_VALUE = -1;
    private static Employee[] employees = 
        { new Employee("Neil", "Russell", "Clerk", 10),
          new Employee("Terrence", "Bennett", "Manager", 10),
          new Employee("Blair", "Palmer", "VP", 10),
          new Employee("Cory", "O'Reilly", "Reporter", 20),
          new Employee("Tony", "McConnell", "Editor", 20),
          new Employee("Jennifer", "Kuffner", "VP", 30) 
        };
    public EmployeesBean() {
    }
    public void setDepartmentNumber(String deptno) {
        selectDepartment(deptno);
    }
    public int findDepartmentValue(String defaultValue) {
        AdfFacesContext afContext = AdfFacesContext.getCurrentInstance();
        String deptno =
            (defaultValue == null ?
             (String)afContext.getPageFlowScope().get(DEPARTMENT_NUMBER_KEY) :
             defaultValue);
        return (deptno == null ? DEPARTMENT_NUMBER_NULL_VALUE :
            Integer.valueOf(deptno));
    }
    public void selectDepartment(String deptno) {
        AdfFacesContext afContext = AdfFacesContext.getCurrentInstance();
        afContext.getPageFlowScope().put(DEPARTMENT_NUMBER_KEY, deptno);
    }
    public Employee[] getEmployees(String deptno) {
    }
```
int filterDeptno = findDepartmentValue(deptno);
Employee[] filteredEmployees = null;
if (filterDeptno == DEPARTMENT_NUMBER_NULL_VALUE) {
    filteredEmployees = employees;
} else {
    ArrayList empsInDept = new ArrayList();
    for (int i = 0; i < employees.length; i++) {
        if (employees[i].getDeptno() == filterDeptno) {
            empsInDept.add(employees[i]);
        }
    }

    filteredEmployees = new Employee[empsInDept.size()];
    empsInDept.toArray(filteredEmployees);
}
return filteredEmployees;

15. In the Application Navigator, right-click DepartmentBean.java and choose Create Data Control.

16. Right-click EmployeesBean.java and choose Create Data Control.
You should be able to see the two new data controls listed in the Data Controls panel.

17. Right-click the ViewController project and choose New.

18. In the New Gallery, expand Web Tier, select JSF and then ADF Task Flow, and click OK.

19. In the Create Task Flow dialog, in the File Name field, enter department.

20. Ensure that Create as Bounded Task Flow and Create with Page Fragments are selected and click OK.

---

Note: A task flow can raise ADFm events. When a task flow is turned into a portlet, all events that can be raised are exposed through navigational parameters. You can edit the associated oracle-portlet.xml file to remove any events that you do not want to be exposed.

21. In the Component Palette, select View from the list of activities, drag it onto department.xml and name it dept.

22. Double-click the view activity and click OK in the Create New JSF Page Fragment dialog to create the corresponding dept.jsff page.

23. In the Data Controls panel of the Application Navigator, expand the DepartmentBean data control.

24. Select the deptno element and drag it onto dept.jsff, choosing Text and then ADF Input Field w/Label from the context menu.

25. Select the selectDepartmentString() method and drag it onto dept.jsff, choosing ADF Button from the context menu.
26. Right-click `dept.jsff` and choose *Go to Page Definition*.

   If you view the source code, you can see `methodAction` created for the `selectDepartmentString()` method.

27. In the Structure window of the Application Navigator, right-click `selectDepartmentString` and choose *Insert inside selectDepartmentString* and then *events*.

28. Right-click the `events` tag and choose *Insert inside events* and then *event*.

29. In the Insert event dialog, in the `name` field, enter `DepartmentSelectedEvent` and then click *OK*.

   This event is raised whenever this `methodAction` runs. This can happen through a UI action, such as clicking the button associated with the `methodAction`, or through an `invokeAction` call to call this `methodAction` directly.

   You can create several event entries for a `methodAction` and all are raised when the `methodAction` fires.

   The payload for the event is the return value for the underlying method. If you examine at the `selectDepartmentString()` method in `DepartmentBean.java` (Example 30–4), you can see that it returns a String representation of the department number.

30. In the Application Navigator, right-click the `ViewController` project and choose *New*.

31. In the New Gallery, expand `Web Tier`, select `JSF` and then `ADF Task Flow`, and click *OK*.

32. In the Create Task Flow dialog, in the `File Name` field, enter `employees`.

33. Ensure that *Create as Bounded Task Flow* and *Create with Page Fragments* are selected and click *OK*.

   **Note:** A detail task flow can contain event consumers to handle ADFm events. However, when a task flow is exposed as a portlet, the only context that can be passed to the task flow is through portlet parameters which are mapped to task flow parameters.

34. In the Component Palette, select `View` from the list of activities, drag it onto `employees.xml` and name it `emp`.

35. Double-click the view activity and click *OK* in the Create New JSF Page Fragment dialog to create the corresponding `emp.jsff` page.

36. In the Data Controls panel of the Application Navigator, expand the `EmployeesBean` data control.

37. Select the `getEmployees(String)` method and drag it onto `emp.jsff`, choosing *Method* and then *ADF Button* from the context menu.
38. In the Edit Action Binding dialog, click OK.

39. Right-click emp.jsff and choose Go to Page Definition.

If you view the source code, you can see the event consumer methodAction binding.

When a methodAction binding receives an event, the underlying method is called with the parameters for the event passed to the parameters for the method.

**Note:** The methodAction binding can be both an event producer and an event consumer. For a methodAction to be an event producer, it must have an event tag defined in the metadata, like the one you created earlier.

All methodAction bindings are event consumers and require no additional metadata.

40. In the source code of the emp.jsff page, delete the button that you added earlier.

```
<af:commandButton actionListener="#{bindings.getEmployees.execute}"
   text="getEmployees"
   disabled="#{!bindings.getEmployees.enabled}" id="cb1"/>
```

Deleting the button from the source view means the methodAction is not deleted from the page definition.

41. In the Data Controls panel of the Application Navigator, expand the getEmployees(String) method.

42. Select Employee and drag it onto emp.jsff, choosing Table and then ADF Read-only Table from the context menu.

43. In the Edit Table Columns dialog, click OK.

44. In employees.xml, click the Source tab.

45. Create a managed-bean entry for the EmployeesBean class.

The code should now look like that in Example 30–7.

**Example 30–7 EmployeesBean Managed Bean**

```xml
<?xml version="1.0" encoding="windows-1252"?><adf-config xmlns="http://xmlns.oracle.com/adf/controller" version="1.2">
   <task-flow-definition id="employees">
      <default-activity>employees</default-activity>
      <managed-bean>
         <managed-bean-name>contextProvider</managed-bean-name>
         <managed-bean-class>hr.EmployeesBean</managed-bean-class>
         <managed-bean-scope>pageFlow</managed-bean-scope>
      </managed-bean>
      <view id="emp">
         <page>/emp.jsff</page>
      </view>
      <use-page-fragments/>
   </task-flow-definition>
</adf-config>
```
46. Create an input-parameter-definition entry and map the value of the parameter to the appropriate EmployeesBean method to store this value.

This is needed so that the task flow can consume WSRP navigational parameters when it is exposed as a portlet. When you turn this task flow into a portlet, the task flow parameter is converted into a portlet parameter.

The code should now look like that in Example 30–8.

**Example 30–8  Input Parameter Definition**

```xml
<?xml version='1.0' encoding='windows-1252' ?>
<adf-config xmlns="http://xmlns.oracle.com/adf/controller" version="1.2">
  <task-flow-definition id="employees">
    <default-activity>employees</default-activity>
    <input-parameter-definition>
      <description>Main context parameter</description>
      <display-name>Department Number</display-name>
      <name>deptno</name>
      <value>${pageFlowScope.contextProvider.departmentNumber}</value>
      <class>java.lang.String</class>
    </input-parameter-definition>
    <managed-bean>
      <managed-bean-name>contextProvider</managed-bean-name>
      <managed-bean-class>hr.EmployeesBean</managed-bean-class>
      <managed-bean-scope>pageFlow</managed-bean-scope>
    </managed-bean>
    <view id="emp">
      <page>/emp.jsff</page>
    </view>
    <use-page-fragments/>
  </task-flow-definition>
</adf-config>
```

47. Before you can turn your task flows into portlets, you must first ensure that they work within the original application. To quickly test this, you can create a page on which you can place the page fragments.

In the Application Navigator, right-click the ViewController project and choose New.

48. In the New Gallery, expand Web Tier, select JSF and then JSF page, and click OK.

49. In the Create JSF Page dialog, in the File Name field, enter testPage.jspx.

50. Select Create as XML Document (*.jspx) and click OK.

51. In the Application Navigator, expand the Page Flows node.

52. Drag the department task flow onto the page and choose Region from the context menu.

53. Drag the employees task flow onto the page and choose Region from the context menu.

54. In the Edit Task Flow Binding dialog, in the Value field for the deptno input parameter, enter ${'10'} and click OK.

55. In the Application Navigator, right-click testPage.jspx and choose Go to Page Definition.

57. In the Event Map Editor dialog, click the Add a New Event Entry icon.

58. In the Add a New EventMap Entry dialog, in the Producer field, drill down to select `deptPageDef.selectDepartmentString`.

59. In the Event Name field, make sure `DepartmentSelectedEvent` is selected.

60. In the Consumer field, drill down to select `empPageDef.getEmployees`.

61. In the Consumer Params section, click the Add Consumer Parameter icon.

62. In the Param Name field, enter `department`.

63. In the Param Value field, enter `${payLoad}`.

---

**Note:** The parameter name is not actually used. You can call your parameter anything. ADFm eventing just uses the position to pass these parameter values to the input parameters of the underlying methodAction method.

`payLoad` is a keyword. If you invoked the expression builder when in this field, you can also see `payLoad` as an option to choose. In this example, the payload is a scalar value (the department number as a string). If it was an object, you can further dereference it or pass the object as a whole. You can pass any valid EL for the value.

64. Click OK to close the Add a New EventMap dialog.

65. Click OK to close the Event Map Editor dialog.

66. In the Application Navigator, right-click `testPage.jspx` and choose Run.

You should be able to enter a value of 10, 20, 30 in the `deptno` field of the master department task flow and see the employees task flow update accordingly when you click `selectDepartmentString` (Figure 30–4).

---

**Figure 30–4  Example Application Running as a Web Application**

<table>
<thead>
<tr>
<th>deptno</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>selectDepartmentString</code></td>
<td></td>
</tr>
<tr>
<td>FirstName</td>
<td>lastName</td>
</tr>
<tr>
<td>Cory</td>
<td>O'Reilly</td>
</tr>
<tr>
<td>Tony</td>
<td>McConnell</td>
</tr>
</tbody>
</table>

---

### 30.3.2 How to Create Portlets from the Task Flows in the Example Application

After creating the example application and verifying that it runs correctly, you can create portlets from the task flows so that they can be consumed in other applications.
To create portlets from task flows:
1. In the Application Navigator, under the Page Flows node, right-click the department task flow and choose Create Portlet Entry.

2. In the Create Portlet Entry dialog, you can accept the default values for the name, title, and description fields, but ensure that you select the Create navigation parameters for events checkbox before you click OK.

   Selecting this checkbox creates a navigation parameter for the event raised by the task flow. When the event is raised by the task flow, the payload for the event is passed to the navigation parameter.

3. In the generated portlet.xml file, you can see the portlet-class, which shows that this is an ADFBridgePortlet. Otherwise this file contains the standard WSRP information and nothing about the ADFm event that is in the task flow.

4. In the Application Navigator, right-click oracle-portlet.xml and choose Open.

   In this file you can see the navigation-parameter created for the ADFm event. It is exposed through the bridge as _adf_event_DepartmentSelectedEvent. When this event is raised by the task flow, the value of this parameter contains the payload for the event. The portlet binding code automatically raises this ADFm event on the consumer side which it is received.

5. Right-click the employees task flow and choose Create Portlet Entry.

6. You can accept the default values for the name, title, and description fields, and click OK.

   You have now made both of your task flows available as portlets from within this application.

7. Right-click testPage.jspx and choose Run.

   Running the application has an implicit deployment. When you deploy an application containing portlet.xml, WebCenter automatically generates the appropriate WSRP entry point for the application. So, after you run the page, you can access the application through the WSRP entry point and through HTTP.

8. To access the WSRP Producer Test Page for your portlets, when your application displays in your browser, add /info after the context root for your application (typically, this means replacing the end of the URL, starting with /faces/..., with /info).

9. To check that the WSDL is accessible, click WSRP v2 WSDL in the WSRP Producer Test Page.

   Copy the WSDL URL of this page so that you can use it to register the producer later.

30.3.3 How to Consume the JSF Portlets from the Example Application

Now that you have created the portlets, you can use them in another application.

To consume JSF portlets:
1. From the main menu, choose File and then New.

2. In the New Gallery, expand General, select Applications and then WebCenter Application, and click OK.
3. On the Name your application page of the Create WebCenter Application wizard, in the **Application Name** field, enter `PortletBridgeConsumerApplication` and then click **Finish**.

4. Right-click the **ViewController** project and click **New**.

5. In the New Gallery, expand **Web Tier**, select **JSF** and then **JSF Page**, and click **OK**.

6. In the Create JSF Page dialog, in the **File Name** field, enter `PortletsRaisingEvents.jspx`.

7. Select **Create as XML Document (*.jspx)** and click **OK**.

   This is the page on which you will place the portlets. Before placing portlets on a page, you must register the portlet producer with the application.

8. In the Application Resources panel of the Application Navigator, right-click **Connections**, choose **New Connection** and then choose **WSRP Producer**.

9. On the Specify Producer Name page of the Register WSRP Portlet Producer wizard, in the **Producer Registration Name** field, enter `PortletBridgeTestProducer` and click **Next**.

10. On the Specify Connection Details page, in the **WSDL URL** field, enter the URL you copied in the previous section then click **Next**.

11. On the Specify Additional Registration Details page, click **Finish**.

12. In the Application Resources panel, expand the **Connections** node and then **WSRP Producer**.

   You can see the new WSRP producer. Expand this to see the portlets that you created from the department and employees task flows.

13. Select the **department** portlet and drag it onto the `PortletsRaisingEvents` page.

14. Select the **employees** portlet and drag it onto the page, just below the department portlet.

15. Right-click `PortletsRaisingEvents.jspx` and choose **Go to Page Definition**.

   In the source code you can see an entry for the department portlet that includes the event that you defined for the task flow. This event is raised whenever the underlying task flow raises the event.

   You can also see an entry for the employees portlet that includes the parameter that you defined for the task flow. This is used to pass data from anywhere on the page back through the portlet to the task flow.

   The page definition also includes a variable iterator that defines a page variable that was created because a portlet with navigational parameters was placed on the page. Portlets or components on the page that do not create ADFm events can pass context to portlets using this page variable.

16. Add a default value for the page variable as shown in **Example 30–9**.

   **Example 30–9 Default Value for Deptno Page Variable**

   ```xml
   <variableIterator id='variables'>
      <variable Name='PortletBridgeApplicationemployees1_1_deptno'
      Type='java.lang.Object
      DefaultValue='${'20'}'/>}
</variableIterator>
```
17. In the Structure panel, right-click PortletsRaisingEventsPageDef and choose Edit Event Map.

18. In the Event Map Editor dialog, click the Add a New Event Entry icon.

19. In the Add New EventMap Entry dialog, in the Producer field, drill down to select PortletBridgeApplicationdepartment1_1.

20. In the Event Name field, make sure DepartmentSelectedEvent is selected.

21. In the Consumer field, drill down to select PortletBridgeApplicationemployees1_1.

22. In the Consumer Params section, click the Add Consumer Parameter icon.

23. In the Param Name field, enter deptNo.

24. In the Param Value field, enter ${payLoad}.

25. Click OK to close the Add New EventMap Entry dialog.

26. Click OK to close the Event Map Editor dialog.

You should now be able to see a new event map entry in the source code of the page definition.

27. In the PortletsRaisingEvents page, select the employees portlet.

28. In the Property Inspector, expand the Common tab if necessary, and in the PartialTriggers field, enter portlet1 (the ID of the department portlet).

This makes sure that the employees portlet refreshes to reflect any changes made in the department portlet (portlet1).

29. In the Application Navigator, right-click PortletsRaisingEvents.jspx and choose Run.

30. In the deptNo field of the department portlet, enter 20 and click selectDepartmentString.

The employees portlet refreshes to show the employee details for department 20 (Figure 30–5).

---

Note: No payload or a null payload is propagated across the wire from the event producer portlet to the event consumer as an empty string. If the consumer must differentiate between an empty string and null payload, you can encode the null value in the payload in the event producer. The event consumer must also look for this custom encoding to detect the null payload.
### Figure 30–5  Portletized Task Flows in an Application

#### PortletBridgeApplication department

<table>
<thead>
<tr>
<th>deptno</th>
<th>20</th>
</tr>
</thead>
</table>

**selectDepartmentString**

#### PortletBridgeApplication employees

<table>
<thead>
<tr>
<th>firstName</th>
<th>lastName</th>
<th>title</th>
<th>deptno</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cory</td>
<td>O’Reilly</td>
<td>Reporter</td>
<td>20</td>
</tr>
<tr>
<td>Tony</td>
<td>McConnell</td>
<td>Editor</td>
<td>20</td>
</tr>
</tbody>
</table>
Creating Portlets with the Portlet Wizard

This chapter explains how to create Java portlets based on the Java Portlet Specification (JSR 168) or the Oracle Portal Developer Kit-Java (PDK-Java) in a WebCenter portlet application, using the Create JSR 168 Java Portlet and Create Oracle PDK-Java Portlet wizards.

This chapter includes the following sections:
- Section 31.1, "Introduction to Java Portlets"
- Section 31.2, "Creating Java Portlets"

31.1 Introduction to Java Portlets

This section includes the following sections:
- Section 31.1.1, "Introduction to Standards-Based Java Portlets"
- Section 31.1.2, "Introduction to PDK-Java Portlets"

31.1.1 Introduction to Standards-Based Java Portlets

Organizations engaged in custom WebCenter application projects have found application integration to be a major issue. Until now, users developed portlets using proprietary APIs for a single portal platform and often faced a shortage of available portlets from a particular portal vendor. All this changed with the introduction of the following standards:
- Web Services for Remote Portlets (WSRP)
- Java Specification Request (JSR) 168

These two standards enable the development of portlets that interoperate with different portal products, and therefore widen the availability of portlets within an organization. This wider availability can, in turn, dramatically increase an organization's productivity when building custom WebCenter applications.

WSRP is a web services standard that enables the plug-and-play of visual, user-facing web services with portals or other intermediary web applications. Being a standard, WSRP enables interoperability between a standards-enabled container and any WSRP portal. WSRP defines the following:
- Web Services Definition Language (WSDL) interface for the invocation of WSRP services
- Markup fragment rules for markup emitted by WSRP services
- The methods to publish, find, and bind WSRP services and metadata
JSR 168 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 168 defines container services which provide the following:

- A portlet API for coding portlet functionality
- The URL-rewriting mechanism for creating user interaction within a portlet container
- The security and personalization of portlets

Oracle actively participates in the WSRP committee and is also a member of the expert group for JSR 168.

**Note:** HTML forms can be submitted using either the `get` or `post` method, but the WSRP standard only requires the consumer (custom WebCenter application) to use the `post` method. Support of the `get` method is optional according to the standard. Since application consumers are not required to support the `get` method, Oracle recommends that you use the `post` method when developing your portlets.

The Relationship Between WSRP and JSR 168

WSRP is a communication protocol between a portlet client, for example, a WebCenter application, and a portlet container running on a remote server. JSR 168 describes the Java Portlet API for running portlet applications. Combining these standards enables developers to integrate their applications from any internal or external source as portlets with WSRP portals. Building pages becomes as simple as selecting portlets from the JDeveloper Component Palette.

Figure 31–1 shows the architecture of the WSRP specification.

**Note:** Figure 31–1 illustrates the use of JSR 168 portlets with WSRP.
Oracle WebCenter Framework can support communication between the custom WebCenter application and both the new Java Portlet APIs and our existing APIs (PDK-Java).

Figure 31–2 shows the architecture of the WSRP support. Notice that the JSR 168-compliant portlet container uses the WSRP protocol for communication and the PDK-Java portlet container uses Oracle’s proprietary SOAP protocol for communication.

Figure 31–2 Oracle WebCenter Portlet Architecture

For a description of how JSR 168 security concepts are exposed through WSRP, see Section 37.12, "Securing Identity Propagation Through WSRP Producers with WS-Security."

31.1.2 Introduction to PDK-Java Portlets

PDK-Java gives you a framework to simplify the development of Java portlets by providing commonly required utilities and enabling you to leverage existing development skills and application components such as JSPs, servlets, and static HTML pages. PDK-Java also enables you to create portlets without having to deal directly with the complexity of communications between Oracle WebCenter 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 Oracle WebCenter Framework for communication with web producers. It translates the information passed between Oracle WebCenter 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 Oracle WebCenter 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 custom WebCenter 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 Oracle WebCenter 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 custom WebCenter application. If the standard behavior of the Producer Runtime does not meet your requirements, then you can easily extend or override specific behaviors. Some of the primary classes are as follows:

- DefaultProviderDefinition
  (oracle.portal.provider.v2.DefaultProviderDefinition)
- DefaultProviderInstance
  (oracle.portal.provider.v2.DefaultProviderInstance)
- DefaultPortletDefinition
  (oracle.portal.provider.v2.DefaultPortletDefinition)
- DefaultPortletInstance
  (oracle.portal.provider.v2.DefaultPortletInstance)
- PortletRenderer
  (oracle.portal.provider.v2.render.PortletRenderer)
- PortletPersonalizationManager
  (oracle.portal.provider.v2.personalize.PortletPersonalizationManager)
- PortletSecurityManager
  (oracle.portal.provider.v1.http.DefaultSecurityManager)

The Producer Utilities provide methods for simplifying the rendering of portlets. The utilities include methods for constructing valid links (hrefs), rendering the portlet's container (including the header), rendering HTML forms that work within a page, and supporting portlet caching.

You can find the JavaDoc reference for the PDK-Java on Oracle Technology Network (OTN) at:

http://www.oracle.com/technology/products/webcenter/portlet_download.html
31.2 Creating Java Portlets

Before you begin you should make sure:

- You are familiar with portlet terminology such as portlet modes. For more information, see Chapter 29, "Overview of Portlets" and Section 31.2.5, "What You May Need to Know When Creating Java Portlets."
- You are familiar with Oracle JDeveloper and know how to build and deploy Java components using it.
- The application in which you are going to create your JSR 168 and PDK-Java portlets is either a Portlet Producer Application or any application scoped for portlet creation (any application except for WebCenter Application).

**Note:** The figures in this section were taken with the default Look and Feel and Theme settings in JDeveloper (Oracle and Fusion Blue). If you have changed these settings, then what you see on your screen may vary slightly, but the content and functionality remains the same. To change the Look and Feel and Theme settings, select Preferences from the Tools menu, and then select Environment.

This section includes the following subsections:

- Section 31.2.1, "How to Create a JSR 168 Java Portlet"
- Section 31.2.2, "What Happens When You Create a JSR 168 Java Portlet Using the JDeveloper Wizard"
- Section 31.2.3, "How to Create a PDK-Java Portlet"
- Section 31.2.4, "What Happens When You Create a PDK-Java Portlet"
- Section 31.2.5, "What You May Need to Know When Creating Java Portlets"

31.2.1 How to Create a JSR 168 Java Portlet

Using the Create JSR 168 Java Portlet wizard in JDeveloper you can expose your portlet over WSRP 2.0 quickly and easily. This wizard supports both WSRP 1.0 and WSRP 2.0. If you choose to create a WSRP 2.0 portlet, then an additional page appears in the wizard for the WSRP 2.0 enhancements.

In the Create JSR 168 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 168 Java portlet using the JDeveloper wizard:

1. In the JDeveloper Application Navigator, open the application under which you want to create your portlet.

   The application must be scoped for portlet creation. An easy way to achieve this is to use a Portlet Producer Application. WebCenter Applications are not scoped for portlet creation.

2. Right-click the project under which you want to create your portlet, and choose New.

3. In the New Gallery, expand Web Tier, select Portlets and then Standards-based Java Portlet (JSR 168), and click OK.
4. On the General Portlet Information page of the Create JSR 168 Java Portlet wizard (Figure 31–3), replace the default name provided in the Name field with one that better describes the purpose of the portlet.

Figure 31–3 The General Portlet Information Page

5. In the Class field, enter a name for the class for the portlet. You can accept the default name provided or supply your own. If you supply your own name, it must be a valid Java name.

6. From the Package dropdown list, select the package in which to create the class.

   Click the Browse button to find packages within the project, if required. If you do not select a specific package, the wizard uses the default package of the project.

7. From the Default Language dropdown list, select the default language that your portlet supports. The wizard uses English by default.

8. Select Enable users to edit portlet content if you want your portlet to support Edit mode. In the wizard, this option is selected by default.

   If you select this option, you can specify the details for the Edit mode later on in the wizard.

9. Select Enable inter-portlet communication using Oracle WSRP V2 extensions to create a portlet that supports Oracle WSRP 2.0 extensions and then click Next.

   Selecting this option creates the oracle-portlet.xml file that is used for WSRP 2.0 features, such as navigation parameters. The WSRP 2.0 standard extends WSRP 1.0 by including support for inter-portlet communication (through navigation parameters) and export and import of portlet customizations.
10. On the Additional Portlet Information page (Figure 31–4), in the **Portlet Title** field, enter a descriptive title for your portlet. The portlet title is displayed in the Resource Palette or Application Resources panel, so make the title something to help users decide whether the portlet is useful to them. The portlet title is also displayed on the portlet header when the portlet appears on a page.

Figure 31–4  The Additional Portlet Information Page

11. The display name, short title, description, and keyword attributes are not implemented in custom WebCenter applications. You do not need to enter any values for these fields unless your portlet is likely to be consumed by other applications. For example, Oracle Portal uses the portlet display name in the Portlet Repository.

12. At this point in the wizard, you can click **Finish** to create the portlet immediately, using the default values for all remaining settings.

To provide additional details for your portlet, click **Next** and follow the remaining steps.

13. On the Content Types and Portlet Modes page (Figure 31–5), to add a content type, select an existing content type (for example, `text/html`) and then click **Add**. In the Content Types dialog move the desired content types to the **Selected** list and then click **OK**.

By default, your portlet displays its content using the `text/html` content type.
14. To add another portlet mode to your portlet, select an existing mode (for example, view) under the appropriate content type and then click Add. In the Portlet Modes dialog, move the desired modes to the Selected list and then click OK.

By default, your portlet includes View mode. If you selected Enable users to edit portlet content on the General Portlet Properties page, then your portlet also includes Edit mode. For more information about portlet modes, see Section 31.2.5.1, “Guidelines for Portlet Modes.”

15. Select each of the portlet modes and specify the implementation method for rendering content.

The wizard generates a portlet dispatcher Java class and any other required files for each portlet mode. The exact nature of this class and any other generated files depends on the chosen implementation method:

- 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.

  This applies to all portlet modes. If any portlet mode uses this selection, the wizard generates a portlet application which uses the Oracle JSF Portlet Bridge.

- Select Map to Path to map the portlet mode to a web resource in the portlet 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.

16. Click **Next**.

If you selected **Enable users to edit portlet content** on the General Portlet Information page earlier in the wizard, then the Customization Preferences page appears where you can create additional customization preferences. Go to step 17.

If you did not select this option, the Security Roles page appears. Go to step 22.

17. On the Customization Preferences page (**Figure 31–6**), click **Add** to add a new customization preference to the portlet.

Customization preferences enable users of the portlet to specify values for the portlet at runtime. By default, the wizard includes a preference to enable users to customize the portlet title.

**Figure 31–6  The Customization Preferences Page**

18. In the Add New Preference dialog, in the **Name** field, enter a name for the new preference.

The name must be unique in the portlet. Use only letters, numbers, and the underscore character.

19. In the **Default Values** field, enter one or more default values for the new preference. Separate multiple values with commas.

20. To make the preferences translatable, select the **Translate** checkbox and then click OK.
JDeveloper generates a resource bundle class for translatable preferences, with strings for which you can obtain translations. At runtime, the portlet references the resource bundle entries.

**Note:** The Name is always translated, but there is not always a requirement to translate the Default Values. For example, if the value is an integer, then no translation is needed.

21. Repeat the preceding steps to add more preferences. When you are done click Next.

22. On the Security Roles page (Figure 31–7), to add an existing security role to your portlet, select the security role and move it to the Selected list.

Security roles enable you to set tiered levels of access to the portlet. For example, a View user can view the portlet but cannot edit it; a Customize user can customize portlet settings; a Manage user can perform all available functions associated with the portlet.

The Available list displays the security roles defined for the application in which you are creating the portlet. Moving a security role to the Selected list creates a reference of the security role in the application’s portlet deployment file (portlet.xml) that refers to the security role in the application’s web deployment file (web.xml).

**Figure 31–7 The Security Roles Page**

23. To define a new security role, click New Security Role.

24. In the Create New Security Role dialog, specify a name and description for the role and then click OK to add the role to the list of available roles.

The name should be unique within the application. The description should provide details about the access privileges and restrictions this role has on the portlet.
You can also manually create security roles in the `web.xml` file.

25. Click Next.

26. On the Caching Options page (Figure 31–8), to enable caching for your portlet, select **Cache Portlet**.

Selecting this option indicates that portlet caching is managed by the portlet container. The portlet itself may choose to cache content for any given response. The settings on this page apply only when the portlet itself does not specify a caching condition for a response.

If you do not want any default caching for this portlet, choose **Do Not Cache By Default**. In this case, the wizard actually sets a cache duration of 0 seconds. As stated earlier, this cache setting only comes into play when the portlet itself does not specify a caching condition for a response.

If you choose no caching here and you later decide to implement default caching for the portlet, then you can change the cache duration value in the `portlet.xml` file, which is generated by the wizard, to a number greater than zero.

For more information, see Section 29.2.4, "Caching Style."

![Figure 31–8 The Caching Options Page](image)

27. If you chose to cache the portlet, in the **Default Expiry Conditions** section, select:

   - **Cache Content Expires After** to expire the cached portlet content after a certain amount of time. Specify the time limit in the adjacent field.
   - **Cache Content Never Expires** to never expire the cached portlet content. You may want to select this option if the portlet contains static content that is unlikely to change.

28. Click Next.

29. On the Initialization Parameters page (Figure 31–9), click **New** to add an initialization parameter to the portlet.
This adds a new row to the table of parameters. Double-click each field in the row to provide a name, default value and description for the parameter.

Repeat this step to add more initialization parameters. When you are done, click Next.

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.

**Figure 31–9  The Initialization Parameters Page**

30. On the Portlet Navigation Parameters page (Figure 31–10), click Add to add a navigation parameter to the portlet.

This adds a new row to the table of parameters. Double-click each field in the row to provide a name, label and hint for the parameter.

Repeat this step to add more navigation parameters. When you are done, click Finish.

Navigation 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 32.1.2, "How to Implement Navigational Parameters (WSRP 2.0)" and Section 34.7, "Contextually Linking WSRP 2.0 Portlets."
31.2.2 What Happens When You Create a JSR 168 Java Portlet Using the JDeveloper Wizard

When you use the JDeveloper wizard to create a JSR 168 Java portlet, JDeveloper generates a default implementation of the portlet. Specifically, the following files are created:

- Two Java classes:
  - `portletName.java` is invoked by the portlet container and contains all the methods required by the portlet standards.
  - `portletnameBundle.java` contains all the translation strings for the portlet.
- `oracle-portlet.xml` is generated if you selected the Oracle WSRP V2 extensions option.
- `portlet.xml` is the portlet deployment file for the application.
- `web.xml` is the web deployment file for the application.
- Files for each portlet mode you selected for the portlet:
  - If you selected Generate JSP for the portlet mode, a JSP page is created for the mode, for example, `view.jsp`.
  - If you selected Generate ADF-Faces JSPX, a JSPX page is created for the mode, for example, `view.jspx`. You can add Faces components to this page.
  - If you selected Map to Path, no additional files are created as the code for the portlet mode resides in an existing resource. Code is added to the portlet's Java class to route requests to the specified target.
  - If you selected Custom Code, no additional files are created, but the code for the portlet mode resides in the portlet's Java class.

You can see all these files in the Application Navigator, as shown in Figure 31–11.
The next step is to extend the sample code with your own business logic to implement the desired functionality and features for your portlet. For more information, see the JSR 168 at:


For a specific example, see "Step 3: Create the Business Logic for the Standards-Based Portlet" in the Oracle Fusion Middleware Tutorial for Oracle WebCenter Developers.

### 31.2.3 How to Create a PDK-Java Portlet

Using the Create Oracle PDK-Java Portlet wizard in JDeveloper you can quickly and easily create PDK-Java portlets. You can choose which portlet modes you want to implement and the implementation method (JSP, HTTP servlet, Java class, or HTML) to use for each mode. The wizard then creates a simple implementation for each of the selected modes.

**To create a PDK-Java portlet using the JDeveloper wizard:**

1. In the JDeveloper Application Navigator, open the application under which you want to create your portlet.

   The application must be scoped for portlet creation. An easy way to achieve this is to use a Portlet Producer Application. WebCenter Applications are not scoped for portlet creation.

2. Right-click the project under which you want to create your portlet, and choose New.

   **Note:** To create the portlet in an existing producer, right-click the producer's `provider.xml` file and choose Add Portlet. This takes you directly to the General Portlet Information page of the Create Oracle PDK-Java Portlet wizard (step 8).

3. In the New Gallery, expand Web Tier, select Portlets and then Oracle PDK-Java Portlet, and click OK.
4. On the Provider Details page of the Create Oracle PDK-Java Portlet wizard (Figure 31–12), enter a name for the new producer to contain your portlet. This name must be unique within the project.

In the PDK-Java, the term provider is used instead of producer. A provider is the same thing as a producer.

Figure 31–12 The Provider Details Page

5. Select Generate Deployment Properties File.

This automatically generates two .properties files:

- `serviceID.properties` defines properties for a producer with that service ID. The service ID has the same value as the producer name.
- `_default.properties` is a default properties file. A producer application may have multiple producers, each with its own service ID. On registration, if no service ID is defined, then the default properties file is used.


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.

---

**Note:** Selecting Oracle PDK-Java Portlet opens the wizard for creating PDK-Java portlets. Selecting Standards-based Java Portlet opens the wizard for creating JSR 168-compliant portlets.
8. On the General Portlet Information page (Figure 31–13), 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 custom WebCenter applications so you do not need to enter a value in this field unless your portlet is likely to be consumed by other applications, such as Oracle Portal.

**Figure 31–13 The General Portlet Information Page**

9. In the **Timeout Interval (Seconds)** field, enter the number of seconds to allow for rendering the portlet.

10. In the **Timeout Message** field, enter a message to display if the rendering of the portlet exceeds the timeout interval specified and then click **Next**.

11. On the View Modes page (Figure 31–14), under Show Page, from the **Implementation Style** dropdown list, select the implementation style to use for the portlet’s Shared Screen mode.

   - Select **JSP** to implement the portlet’s Shared Screen mode as a JavaServer Page. In the **File Name** field, enter the name of the file to be generated by the wizard.

   - Select **HTTP Servlet** to implement the portlet’s Shared Screen mode as an HTTP servlet. In the **Package Name** field, enter the name of the package that contains the HTTP servlet. In the **Class Name** field, enter the Java class to be referenced with the portlet’s Shared Screen mode.

   - Select **HTML File** to implement the portlet’s Shared Screen mode as an HTML file. In the **File Name** field, enter the name of the file to be generated by the wizard. Note that, when you choose **HTML File**, the following code is added inside the `<renderer>` element of your `provider.xml` file:

     ```xml
     <showPage class="oracle.portal.provider.v2.render.http.ResourceRenderer">
     <resourcePath>provider_id/portlet_name/file.html</resourcePath>
     ```
charSet indicates the character set that the producer must use to encode the HTML page. The default character set is determined by JDeveloper preferences. If you require a different character set, you must update this element of provider.xml accordingly.

- Select Java Class to implement the portlet's Shared Screen mode as a Java class. In the Package Name field, enter the name of the package that contains the Java class. In the Class Name field, enter the name of the Java class.

For more information about Shared Screen mode, see Section 31.2.5.1.1, "Shared Screen Mode (View Mode for JSR 168)."

Figure 31–14  The View Modes Page

12. To implement Full Screen mode for your portlet, select Show Details Page and then select an implementation style as described for Full Screen mode in step 11.

For more information about Full Screen mode, see Section 31.2.5.1.5, "Full Screen Mode (PDK-Java)."

13. At this point in the wizard, you can click Finish to create the portlet immediately, using the default values for all remaining settings.

To provide additional details for your portlet, click Next and follow the remaining steps.

14. On the Customize Modes page (Figure 31–15), 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 31.2.5.1.2, "Edit Mode (JSR 168 and PDK-Java)."
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 31.2.5.1.3, "Edit Defaults Mode (JSR 168 and PDK-Java)."

16. Click Next.

17. On the Additional Modes page (**Figure 31–16**), 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 31.2.5.1.6, "Help Mode (JSR 168 and PDK-Java)."
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 31.2.5.1.7, "About Mode (JSR 168 and PDK-Java)."

19. Click Next.

20. On the Public Portlet Parameters page (Figure 31–17), 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 32.2.2, "How to Implement Public Parameters."
21. Oracle PDK-Java events are not supported in custom WebCenter applications, so on the Public Portlets Events page, click **Finish**.

For more information about events, see the *Oracle Fusion Middleware Developer’s Guide for Oracle Portal*.

### 31.2.4 What Happens When You Create a PDK-Java Portlet

When you use the JDeveloper wizard to create a PDK-Java portlet, JDeveloper generates a default implementation of the portlet. Specifically, the following files are created:

- Files for each portlet mode you selected, for example `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 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 31–18](image).
The next step is to extend the sample code with your own business logic to implement the desired functionality and features for your portlet. For more information, see the PDK-Java JavaDoc.

### 31.2.5 What You May Need to Know When Creating Java Portlets

When you write your portlets in Java for either JSR 168 or PDK-Java, you should follow the best practices described in this section.

This section includes the following subsections:

- **Section 31.2.5.1, "Guidelines for Portlet Modes"
- **Section 31.2.5.2, "Guidelines for Navigation within a Portlet"
- **Section 31.2.5.3, "Guidelines for JavaScript"
- **Section 31.2.5.4, "Guidelines for PDK-Java Portlets"

#### 31.2.5.1 Guidelines for Portlet Modes

Portlet mode exhibits the runtime portlet functionality seen by users. You can provide an extended set of modes in addition to the standard ones provided with Oracle WebCenter and Oracle Portal. Different portal products support extended portlet modes as follows:

- Oracle WebCenter and Portal have defined an extended set of modes that these two products support. Different modes are available for both PDK-Java and JSR 168 portlets. PDK-Java offers some modes not offered by JSR 168. If you are coding portlets to JSR 168, then you can declare custom portlet modes in the `portlet.xml` file that map to the extra modes offered by PDK-Java, or to accommodate any other functionality you may want to provide. For example, the JSR 168 Java Portlet wizard for JSR 168 portlets includes a custom mode called `print`, which you can use to provide a printer friendly version of the portlet.
Creating Java Portlets

Defining custom modes is especially useful if the portlet must interoperate with portal implementations from other vendors. You can offer any of these modes to users. In fact, it is recommended that some modes like Edit Defaults are offered.

- Third party portal products may have their own set of extended modes (JSR 168 custom modes) that producers can offer. In custom WebCenter applications and Oracle Portal, chrome UI for portlets only shows the custom modes that are defined in Oracle WebCenter Framework. Arbitrary custom modes that a third party or custom portlet producer offers are ignored and therefore not supported.

This section includes the following subsections that discuss each of the standard portlet modes:

- Section 31.2.5.1.1, "Shared Screen Mode (View Mode for JSR 168)"
- Section 31.2.5.1.2, "Edit Mode (JSR 168 and PDK-Java)"
- Section 31.2.5.1.3, "Edit Defaults Mode (JSR 168 and PDK-Java)"
- Section 31.2.5.1.4, "Preview Mode (JSR 168 and PDK-Java)"
- Section 31.2.5.1.5, "Full Screen Mode (PDK-Java)"
- Section 31.2.5.1.6, "Help Mode (JSR 168 and PDK-Java)"
- Section 31.2.5.1.7, "About Mode (JSR 168 and PDK-Java)"

31.2.5.1.1 Shared Screen Mode (View Mode for JSR 168) A portlet uses Shared Screen mode (known as View mode in JSR 168) to appear on a page with other portlets. This is the mode most people think about when they envision a portlet. A JSR 168 portlet must have a view mode, the rest 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 of an HTML table cell. The portlet should display only content that can be rendered within a table cell. Furthermore, the actual size of the table cell may vary depending on user settings, the browser width, and the amount and style of content in the portlet.

HTML Guidelines for Rendering Portlets

Plain HTML is the most basic way to render portlets and provides a great deal of flexibility to portlet developers. You can use almost any standard HTML paradigm, such as links, forms, images, tables, if it can display within an HTML table cell. Improperly written HTML may appear inconsistently across different browsers and, in the worst case, could cause parts of your page not to appear at all. Ensure that you adhere to the following rules:

- **Use standard HTML:** The official HTML specification is available from the W3C (more information available at: [http://www.w3.org/MarkUp/](http://www.w3.org/MarkUp/)).

- **Avoid unterminated and extraneous tags:** The behavior of pages with improperly terminated tags is unpredictable because it depends on what the browser chooses to do. Tools like weblint and HTML Tidy can help detect and fix hanging and unnecessary tags.

- **Consider restrictions imposed by container objects:** If your portlet is contained inside of an HTML element, such as a table cell, then you must ensure that your portlet can be rendered within that container. For example, if you place a portlet in a table cell, then you could not use frames in the portlet because they do not appear when inserted in a table.
- **Keep portlet content concise**: Do not try to take full screen content and expose it through a small portlet. You may end up with portlet content too small or cramped for smaller monitors. Full screen content is best viewed in Full Screen mode of PDK-Java.

- **Do not create fixed-width HTML tables in portlets**: You have no way to tell how wide a column your portlet has on a user’s page. If your portlet requires more room than given, then it might overlap with another portlet in certain browsers.

- **Avoid long, unbroken lines of text**: The result is similar to what happens with wide fixed-width tables. Your portlet might overlap other portlets in certain browsers.

- **Check behavior when resizing the page**: Test your portlet’s behavior when the browser window is resized to ensure that it works in different browser window sizes.

- **Check behavior when the default browser font changes**: People may choose whatever font size they want and they can change it at any time. Your portlet should handle these situations gracefully.

The HTML you use also affects the perceived performance of your site. Users judge performance based on how long it takes for them to see the page they requested, and browsers require time to interpret and display HTML. Given that, you should consider the following:

- **Avoid lengthy, complex HTML**: Portlets share a page with other portlets. Thus, portlet generation times can significantly affect the overall performance of the page. If portlets must render complex HTML or wait for external resources, such as third-party applications, then it can greatly slow the rendering of the page.

### Cascading Style Sheet Guidelines for Rendering Portlets

The fonts and colors of every portlet on a page should match the style settings chosen by the user. To accomplish this goal, these style selections are embedded automatically using a Cascading Style Sheet (CSS) on each page. The portlets access these settings for their fonts and colors, either directly or using the Application Programming Interface (API).

While different browsers have implemented varying levels of the full CSS specification, Oracle WebCenter Framework uses a very basic subset of this specification to allow for consistent fonts and colors. CSS implementation levels should not affect the consistency of your pages across browsers. Follow these guidelines for using CSS:

- **Use CSS instead of hard coding**: Hard coding fonts and colors is extremely dangerous. If you hard code fonts and colors, then your portlet may look out of place when the user changes the page style settings. Since you have no way of knowing the user’s font and color preference choices, you might also choose to hard code a font color that turns out to be the same as the user’s chosen background color, in which case your portlet appears to be invisible to that user.

- **Use the CSS APIs to format your text**: The style sheet definition is available at the top of pages, but you should not call it directly. Instead, use the APIs provided to format your text appropriately. This method ensures that your portlets work even if the style sheet changes in the future.

- **Avoid using CSS for absolute positioning**: Since users can personalize their pages, you cannot guarantee that your portlet can appear in a particular spot.
Follow Accessibility Standards: You should ensure that you code your style sheets according to existing accessibility standards (more information available at http://www.w3.org/TR/WCAG10-CSS-TECHS/).

31.2.5.1.2 Edit Mode (JSR 168 and PDK-Java) A portlet uses Edit mode to enable users to personalize the behavior of the portlet. Edit mode provides a list of settings that the user can change. These settings may include the title, type of content, formatting, amount of information, defaults for form elements, and anything that affects the appearance or content of the portlet.

Users typically access a portlet’s Edit mode by choosing Personalize from the portlet’s dropdown list of options. When users choose Personalize, a new page appears in the same browser window. The portlet typically creates a web page representing a dialog to choose the portlet’s settings. After applying the settings, users automatically return to the original page.

Guidelines for Edit Mode Operations
The following guidelines should govern what you expose to users in Edit mode:

- **Enable users to personalize the title of the portlet:** The same portlet may be added to the same page several times. Enabling the user to personalize the title helps alleviate confusion.
- **If using caching, invalidate the content:** If personalizations cause a change in portlet display or content, then you must ensure that the portlet content is regenerated and not returned from the cache. Otherwise, the user may see incorrect content.
- **Do not use Edit mode as an administrative tool:** Edit mode is meant to give users a way of changing the behavior of their portlets. If you want to change producer settings or do other administrative tasks, then you should create secured portlets specifically for those tasks.

Guidelines for Buttons in Edit Mode
For consistency and user convenience, Edit mode should implement the following buttons in the following order:

- **OK** saves the user personalizations and returns the portlet to view mode.
- **Apply** saves the user personalizations and reloads the current page.
- **Cancel** returns the portlet to view mode without saving changes.

Guidelines for Rendering Personalization Values
When you show the forms used to change personalization settings, you should default the values such that the user does not have to constantly reenter settings. When rendering the personalization values, use the following sequence to provide consistent behavior:

1. **User preference:** Query and display this user’s personalizations, if available.
2. **Instance defaults:** If no user personalizations are found, then query and display system defaults for the portlet instance. These are set in Edit Defaults mode and apply only to this portlet instance.
3. **Portlet defaults:** If no system default personalizations are found, then display general portlet defaults, which may be blank. General portlet defaults are sometimes hard coded into the portlet but should be overridden if either of the two previous conditions apply.
This logic enables the personalizations to be presented in a predictable way, consistent with the other portlets in the custom WebCenter application.

31.2.5.1.3 Edit Defaults Mode (JSR 168 and PDK-Java) A portlet uses the Edit Defaults mode to enable administrators to customize the default behavior of a particular portlet instance. Edit Defaults mode provides a list of settings that the application developer can change. These settings may include the title, type of content, formatting, amount of information, defaults for form elements, and anything that affects the appearance or content of the portlet.

These default personalization settings can change the appearance and content of that individual portlet for all users. Because Edit Defaults mode defines the system-level defaults for what a portlet displays and how it displays it, this mode should not be used as an administrative tool or for managing other portlets.

Administrators access Edit Defaults mode, when editing a page, by choosing Customize from the portlet's dropdown list.

When users click the Customize icon, the portlet displays in the same browser window. The portlet typically creates a web page representing a dialog to personalize the portlet instance settings. After applying the settings, users are automatically returned to the original page.

Guideline for Edit Defaults Mode Options
The following guideline should govern what you expose to page designers in Edit Defaults mode:

- Do not use Edit Defaults mode as an administrative tool. Edit Defaults mode gives users a way of changing the behavior of their portlets. If you want to change producer settings or do other administrative tasks, then you should create secured portlets specifically for those tasks.

Guidelines for Buttons in Edit Defaults Mode
For consistency and user convenience, Edit Defaults mode should implement the following buttons in the following order:

- OK saves the user personalizations and returns the portlet to view mode.
- Apply saves the user personalizations and reloads the current page.
- Cancel returns the portlet to view mode without saving changes.

Guidelines for Rendering Personalization Values
When you show the forms used to change personalization settings, you should default the values so that the application developer does not have to constantly reenter settings. When rendering personalization values, use the following sequence to provide consistent behavior:

1. Instance preferences: Query and display system defaults for the portlet instance.
2. Portlet defaults: If no system default personalizations are found, then display general portlet defaults, which may be blank. General portlet defaults are sometimes hard coded into the portlet but should be overridden by system defaults.

This logic enables the personalizations to be presented in a predictable way, consistent with the other portlets in the custom WebCenter application.
31.2.5.1.4 Preview Mode (JSR 168 and PDK-Java) A portlet uses Preview mode to show the user how the portlet looks before adding it to a page. Preview mode visually represents what the portlet can do. Not all portlet consumers call this mode. For example, Oracle Portal uses this mode but Oracle WebCenter Framework does not.

---

**Note:** Oracle Portal calls this mode when the user clicks the Preview icon on the Add Portlet page. A window then displays the preview of the chosen portlet. The user has the option to add that portlet to the page. This mode has no particular application in custom WebCenter applications, but used in Oracle Portal's Portlet Repository.

Guidelines for Preview Mode
The following guidelines should govern what you expose to users in Preview mode:

- **Provide an idea of what the portlet does.** Preview mode should generate enough content for the user to get an idea of the actual content and functionality of the portlet.

- **Keep your portlet previews small.** The amount of data produced in this mode should not exceed a few lines of HTML or a screen shot. Preview mode appears in a small area, and exceeding the window's size looks unprofessional and forces users to scroll.

- **Do not use live hyperlinks.** Links may not work as expected when rendered in Preview mode. Hyperlinks can be simulated using the underline font.

- **Do not use active form buttons.** Forms may not work as you expect them to when rendered in Preview mode. If you decide to render form elements, then do not link them to anything.

31.2.5.1.5 Full Screen Mode (PDK-Java) Portlets use Full Screen mode to provide a larger version of the portlet for displaying additional details. Full Screen mode lets a portlet have the entire window to itself. Not all portlet consumers call this mode. For example, Oracle Portal uses this mode but Oracle WebCenter Framework does not. In Oracle Portal, users access a portlet’s Full Screen mode by clicking the title of the portlet.

For example, if a portlet displays expense information, then it could show a summary of the top ten spenders in Shared Screen mode and the spending totals for everyone in Full Screen mode. Portlets can also provide a shortcut to web applications. If a portlet provided an interface to submitting receipts for expenses in Shared Screen mode, then it could link to the entire expense application from Full Screen mode.

Technically, JSR 168 portlets do not have Full Screen mode. However, you can implement the equivalent of Full Screen mode for a JSR 168 portlet with View mode (Shared Screen mode) and a maximized state for the window.

31.2.5.1.6 Help Mode (JSR 168 and PDK-Java) 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 content, and its capabilities with this mode.

Users access a portlet’s Help mode by choosing the Help action in the portlet.

**Guideline for Help Mode**
The following guideline should govern what you expose to users in Help mode:
Create how to use the portlet. Users may not know all the features your portlet provides just from its interface. Describe the features and how to get the most out of them.

31.2.5.1.7 About Mode (JSR 168 and PDK-Java) Users must be able to see what version of the portlet is currently running, its publication and copyright information, and how to contact the author. Portlets that require registration may link to web-based applications or contact information from this mode, as well.

Users access a portlet’s About mode by choosing About from the dropdown list in the portlet’s chrome. A new page appears in the same browser window. The portlet can either generate the content for this new page or take the user to an existing page or application.

Guideline for About Mode
The following guideline should govern what you expose to users in About mode:

- Display relevant copyright, version, and author information. Users want to know what portlet they are using and where they can get more information. The about page may become important when supporting your portlets.

31.2.5.2 Guidelines for Navigation within a Portlet
In some ways, navigation between different sections or pages of a single portlet is identical to navigation between standard web pages. Users can submit forms and click links. In typical, simple web pages, both of these actions involve sending a message directly to the server responsible for rendering the new content, which is then returned to the client. In portlets, which comprise only part of a page, the form submission or link rendered within the portlet does not directly target the portlet. It passes information to the portlet through the custom WebCenter application. If a link or form within a portlet does not refer back to the application, then following that link takes the user away from the application, which is not typically the desired behavior.

The component developer does not need to know the detailed mechanics of how the parameters of a form or link get passed around between the user, application, and portlet. However, they must understand that they cannot write links in a portlet the same way they do for typical, simple web pages.

Types of Links for Portlets
A portlet may render links of four classes, as follows:

- Intraportlet links require the portlet to be aware of the address of the custom WebCenter application because they actually refer to it in some way.
- Application links, like intraportlet links, must be aware of the address of the custom WebCenter application for the same reason.
- External links make no reference to the custom WebCenter application and work in portlets as they would do in a normal web page.
- Internal/Resource links, like external links, also make no reference to the custom WebCenter application.

Figure 31–19 contains a summary of these link types. The arrows indicate how the links reference the resources to which they logically refer.
This section includes the following subsections:

- Section 31.2.5.2.1, "Intraportlet Links"
- Section 31.2.5.2.2, "Application Links"
- Section 31.2.5.2.3, "External Links"
- Section 31.2.5.2.4, "Internal/Resource Links"

31.2.5.2.1 Intraportlet Links  Intraportlet links go to different sections or pages within a given portlet. Strictly speaking, they refer to the page containing the portlet, but they contain parameters that cause the portlet to render a different section or page within that page when it is requested by the user.

As a direct consequence, a portlet cannot expect to render links to different sections or pages of itself using relative links or absolute links based on its own server context. Intraportlet links are useful for intraportlet navigation, either as links or form submission targets.

31.2.5.2.2 Application Links  Application links refer to significant pages within the custom WebCenter application, such as the user's home page.

31.2.5.2.3 External Links  External links refer neither to the portlet (through a page) nor to any part of the custom WebCenter application. If selected, these links take the user away from the application, for example, www.oracle.com.

31.2.5.2.4 Internal/Resource Links  Internal/Resource links refer to internal (to the portlet) resources. Sometimes they are exclusively used internally during portlet rendering, for example as a server side include. On other occasions, they may be used
externally to reference portlet resources like images. In this latter case, you can use the PDK-Java `constructResourceURL` method in the `UrlUtils` class to retrieve images from behind a firewall using resource proxy. Note that in order for resource proxying to work, you must first set the JNDI variable, `oracle/portal/provider/sample/resourceUrlKey`, for the producer. For more information about setting JNDI variables, see Section 32.2.4.2, "Setting JNDI Variable Values."

For example, `lottery.jsp` of the lottery sample, which is available with PDK-Java, contains resource proxy requests for images.

```jsp
<%@ page contentType="text/html;charset=UTF-8" %>
<%@ page session="false" import="oracle.portal.provider.v2.render.*" %>
<%@ page import="oracle.portal.provider.v2.render.http.HttpPortletRendererUtil" %>
<%@ page import="oracle.portal.provider.v2.url.UrlUtils" %>
<%@ page import="oracle.portal.sample.v2.devguide.lottery.*" %>
<% LottoPicker picker = new LottoPicker();
picker.setIdentity(request.getRemoteAddr()); %>
<% PortletRenderRequest portletRequest = (PortletRenderRequest) request.getAttribute("oracle.portal.PortletRenderRequest"); %>
<% String name = portletRequest.getUser().getName(); %>
<P class="PortletHeading1" ALIGN="CENTER">Hi <%= name %>, Your Specially Picked</P>
<P ALIGN="CENTER"><IMG SRC="<%= UrlUtils.constructResourceURL(portletRequest, HttpPortletRendererUtil.absoluteLink(request, "images/winningnumbers.gif")) %>
 WIDTH="450" HEIGHT="69" ALIGN="BOTTOM" BORDER="0"></P>
<% int [] picks = picker.getPicks();
for (int i = 0; i < picks.length; i++) {
<% }
<IMG SRC="<%= UrlUtils.constructResourceURL(portletRequest, HttpPortletRendererUtil.absoluteLink(request, "images/ball" + picks[i])) %>
.gif" WIDTH="68" HEIGHT="76" ALIGN="BOTTOM" BORDER="0">"
</TD>
<% }
```

For session-based producers, any cookies returned from the original `initSession` call to the producer are sent with the request back to the producer to maintain the right session context.

### 31.2.5.3 Guidelines for JavaScript

JavaScript can often be useful within a portlet, but bear in mind the following guidelines within your portlets:

- You should never use JavaScript to redirect the page in which the portlet is rendered. If you want to direct users elsewhere, then you should do so in your portlet action handling code or open a new window in the browser.
- Ensure that identifiers in your JavaScript are qualified. By qualifying your identifiers, you ensure that they are unique and do not clash with any JavaScript on the page.
31.2.5.4 Guidelines for PDK-Java Portlets

In Oracle WebCenter Framework, PDK-Java portlets work somewhat differently than they did in Oracle Portal. As a result, you must be aware of the following new design considerations when you build PDK-Java portlets in Oracle WebCenter Framework:

- Your portlet must not contain any code that relies upon the URL format or parameters in the request that were not explicitly added by your portlet.

- You should never assume that your portlet is the only one on a page, regardless of the portlet mode. For example, even if you are in Edit mode, you should not assume that yours 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.
This chapter explains how you can enhance the Java portlets you created with the Oracle JDeveloper Create Java Portlet wizards.

This chapter includes the following sections:

- Section 32.1, "Enhancing JSR 168 Java Portlets"
- Section 32.2, "Enhancing PDK-Java Portlets"
- Section 32.3, "Testing Portlet Personalization"
- Section 32.4, "Creating a Struts Portlet"

Before You Begin

Before you begin looking through this chapter ensure that:

- You are familiar with portlet terminology such as portlet modes. For more information, see Chapter 29, "Overview of Portlets" and Section 31.2.5, "What You May Need to Know When Creating Java Portlets."
- You have access to a WebLogic Server (WLS) managed server that is configured for use as a portlet container, such as the WLS_Portal server.
- You are familiar with JDeveloper and know how to build and deploy Java components using it. You can download JDeveloper from OTN: http://www.oracle.com/technology/products/jdev/index.html

32.1 Enhancing JSR 168 Java Portlets

When you have built your initial portlet in the Create JSR 168 Java Portlet wizard as described in Section 31.2.1, "How to Create a JSR 168 Java Portlet," the next step is to enhance it. Because JSR 168 portlets adhere to the Java standards, you can find substantial information about enhancing them from many different sources, such as third-party books and web pages.

This section includes the following subsections:

- Section 32.1.1, "How to Add Personalization"
- Section 32.1.2, "How to Implement Navigational Parameters (WSRP 2.0)"
- Section 32.1.3, "How to Implement Export/Import of Customizations (WSRP 2.0)"
- Section 32.1.4, "How to Implement Rewritten URLs for Resource Proxy"
- Section 32.1.5, "How to Implement Stateless Resource Proxying"
- Section 32.1.6, "How to Disable Java Object Cache for Preference Store Access"
32.1.1 How to Add Personalization

As a quick example of adding personalization, you can enhance the portlet you created in Section 31.2.1, "How to Create a JSR 168 Java Portlet" with some code that enables a user in Edit or Edit Defaults mode to paste HTML into a field for the portlet to render. You can then easily redeploy the portlet and then test it.

Before You Begin

The steps that follow assume that you have:

- Built a portlet using the Create JSR 168 Java Portlet wizard in JDeveloper. For more information, see Section 31.2.1, "How to Create a JSR 168 Java Portlet."
- Added a preference called portletContent in the Customization Preferences page of the wizard.
- Registered the producer with a custom WebCenter application and added the portlet to a page. For more information, see Section 34.2.1, "How to Register a WSRP Portlet Producer" and Section 34.3.1, "How to Add a Portlet to a Page."
- Enabled Oracle ADF security for the application. For more information, see Section 37.3, "Setting Up Security for Your Application."

The user must be logged in to the application as an authenticated user to access the personalization feature.

To implement simple personalization:

1. In the JDeveloper Application Navigator, open the application that contains the portlet.
2. Expand the project that contains the portlet.
3. Right-click view.jsp and choose Open.
4. In the visual editor, click the Source tab and add the code indicated in bold in Example 32–1 to display the portletContent customization preference:

   Example 32–1 view.jsp Sample Code

   `<%@ page contentType="text/html"%>
   `<%@ taglib uri="http://java.sun.com/portlet" prefix="portlet"%>
   `<%@ page content-type="text/html"%>
   `<%@ taglib uri="http://java.sun.com/portlet" prefix="portlet"%>
   `<portlet:defineObjects/>
   `<%
   String[] str = {"Portlet Content"};
   PortletPreferences prefs = renderRequest.getPreferences();
   str = prefs.getValues("portletContent", str);
   for (int i=0; i<str.length; i++)
   {%
   `%><%=i<str.length-1)?str[i]++, str[i]<%>
   
5. In the Application Navigator, right-click edit.jsp and choose Open.

   Notice that the JSP consists of a form field, a form input field, and two form button fields.
6. In the visual editor, click the Source tab and add the code indicated in bold in Example 32–2 to implement a form field for users to enter a value for the portletContent customization preference:

Example 32–2  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>
<tr>    
<td width="20%">
    <p class="portlet-form-field" align="right">
        <%=  res.getString(PortletenhanceBundle.PORTLETTITLE) %>
    </p>
</td>
<td width="80%">
    <input class="portlet-form-input-field"
        type="TEXT"
        name="<%= Portletenhance.PORTLETTITLE_KEY %>
        value="<%= prefs.getValue(Portletenhance.PORTLETTITLE_KEY,
        res.getString("javax.portlet.title")) %>
        size="20">
    </td>
</tr>
</table>
<% String[] str = {"Portlet Content"};
    str = prefs.getValues("portletContent", str);%
<tr><td width="20%">
    <p class="portlet-form-field" align="right">
        Content
    </p>
</td>
<td width="80%">
    <textarea rows="10" cols="60" class="portlet-form-input-field"
        name="portletContent">
```
for (int i=0; i<str.length; i++)
{%><%= (i<str.length-1) ? str[i]+", " : str[i] %><%}%>
</textarea>
</td></tr>
<tr>
<td colspan="2" align="center">
<input class="portlet-form-button" type="submit"
name="<%=PortletEnhance.OK_ACTION%>"
value="<%=res.getString(PortletEnhanceBundle.OK_LABEL)%>">
<input class="portlet-form-button" type="submit"
name="<%=PortletEnhance.APPLY_ACTION%>"
value="<%=res.getString(PortletEnhanceBundle.APPLY_LABEL)%>">
</td>
</tr>
</table>
</form>

7. In the Application Navigator, right-click portletName.java and choose Open.

8. In the visual editor, click the Source tab and add the following two lines of code (indicated in bold) to the processAction method:

```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));
}
prefs.store();
```

9. Redeploy the portlet.

Notice that JDeveloper automatically saves and compiles the code before deploying the portlet. For a reminder of how to perform this step, see Chapter 33, "Testing and Deploying Your Portlets."

10. Reload the page that contains the portlet and you can see that the portlet now displays the text Portlet Content, which was one of the changes you made.

11. Click the Personalize link to see the form field that you added. Enter some text in this field and close the dialog. You can see the new text displayed in the portlet.

### 32.1.2 How to Implement Navigational Parameters (WSRP 2.0)

While JSR 168 and WSRP 1.0 do not address public portlet parameters, WSRP 2.0 introduces navigational parameters to enable inter-portlet communication. With the release of the new portlet Application Programming Interface (API) standard, JSR 286, the need for vendor-specific API extensions will no longer be necessary. Until then, you are required to use the Oracle-specific portlet container and API extensions.

Using the Create JSR 168 Java Portlet wizard, you can easily create a portlet with navigational parameters. When you register the producer and drop the portlet on a page, the portlet's parameters are automatically linked to page variables.
To add public parameters to JSR 168 portlets using WSRP 2.0 navigational parameters:

1. In the JDeveloper Application Navigator, open the application under which you want to create the portlet.
2. Right-click the project under which you want to create the portlet and choose New.
3. In the New Gallery, expand Web Tier, select Portlets and then Standards-based Java Portlet (JSR 168), and click OK.
4. On the General Portlet Information page of the Create JSR 168 Java Portlet wizard, select Enable inter-portlet communication using Oracle WSRP V2 extensions as shown in Figure 32–1.

**Figure 32–1 Creating a Portlet with Navigational Parameters**

5. Proceed through the wizard until you reach the Portlet Navigation Parameters page. For basic information about going through the wizard, see Section 31.2.1, "How to Create a JSR 168 Java Portlet."
6. On the Portlet Navigation Parameters page (Figure 32–2), click Add.
   This adds a new row to the table of parameters.
7. Replace the default values in the **Name**, **Label**, and **Hint** fields with something more meaningful for your parameter.

8. Add more parameters as required and then click **Finish**.

9. In the Applications Navigator, right-click **oracle-portlet.xml** and choose **Open**. You should see entries for the parameters that you added when you created the portlet, as shown in **Example 32–3**.

**Example 32–3  oracle-portlet.xml Sample, Navigational Parameters**

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<portlet-app-extension
 xmlns="http://xmlns.oracle.com/portlet/oracle-portlet-app">
 <portlet-extension>
   <portlet-name>portlet1</portlet-name>
   <navigation-parameters>
     <name>Parameter_01</name>
     <type>xsi:string</type>
     <label xml:lang="en">Parameter 1</label>
     <hint xml:lang="en">First parameter.</hint>
   </navigation-parameters>
   <navigation-parameters>
     <name>Parameter_02</name>
     <type>xsi:string</type>
     <label xml:lang="en">Parameter 2</label>
     <hint xml:lang="en">Second parameter.</hint>
   </navigation-parameters>
 <portlet-id>1164232649525</portlet-id>
 </portlet-extension>
</portlet-app-extension>
```

10. Add code to the portlet that uses the parameters.
When the parameters are included in the `oracle-portlet.xml` file, you can access and set them in the same way as any normal render parameter, and the values are automatically exposed by the consumer.

For an example of a portlet used to set the values of navigation parameters, see the Parameter Form portlet, provided with the sample WSRP producer.

11. You can use the following APIs to read parameters passed to the portlet.

```java
String param1 = request.getParameter("Parameter_01");
String param2 = request.getParameter("Parameter_02");
```

For an example of a portlet used to read and display the values of navigation parameters, see the Parameter Display portlet, provided with the sample WSRP producer.

12. Application developers can add these portlets to a page and link them using the navigational parameters. For more information, see Section 34.7, "Contextually Linking WSRP 2.0 Portlets."

### 32.1.3 How to Implement Export/Import of Customizations (WSRP 2.0)

Another new feature that arrives with WSRP 2.0 is the ability to keep customizations with portlets when moving them from one deployment to another. Customizations are portlet preferences that are set in edit defaults mode. For example, suppose that you create a portlet and then customize its title within your development environment. If you have enabled export and import for that portlet and its producer, then the customized title is transported along with the portlet when you deploy it in production environment. If you do not enable export and import, then all customizations are lost when you transport the portlet from one deployment environment to another.

**To implement export or import for a portlet and its producer:**

1. In the JDeveloper Application Navigator, open the application that contains the portlet.
2. Expand the project that contains the portlet.
3. Right-click the `oracle-portlet.xml` file for the portlet's producer and choose Open.
4. Ensure that the `allow-export` and `allow-import` tags are included for each portlet and the producer as shown in Example 32–4. Add them if necessary.

**Example 32–4  oracle-portlet.xml Sample, Export/Import**

```xml
<portlet-app-extension xsi:schemaLocation="./oracle-portlet-app.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="./oracle-portlet-app.xsd">
<portlet-extension>
<portlet-name>portlet1</portlet-name>
<navigation-parameters>
<name>Parameter_01</name>
<type>xsi:string</type>
<label xml:lang="en">First Parameter</label>
<hint xml:lang="en">hint0</hint>
</navigation-parameters>
<navigation-parameters>
<name>Parameter_02</name>
<type>xsi:string</type>
```
32.1.4 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, 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 32–5 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 32–5 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`.

### 32.1.5 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 32–6 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 32–6  Stateless Resource Proxying**

```java
request.setAttribute("oracle.portlet.server.useStatelessProxying", Boolean.TRUE);
String url = response.encodeURL(pathToResource);
request.removeAttribute("oracle.portlet.server.useStatelessProxying");
```

You can also use the following constant:

`oracle.portlet.server.containerimpl.PortletRequestImpl.USE_STATELESS_PROXYING`

This is shown in Example 32–7.

**Example 32–7  Stateless Resource Proxying Using Constant**

```java
request.setAttribute(PortletRequestImpl.USE_STATELESS_PROXYING, Boolean.TRUE);
String url = response.encodeURL(pathToResource);
request.removeAttribute(PortletRequestImpl.USE_STATELESS_PROXYING);
```

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`.

### 32.1.6 How to Disable Java Object Cache for Preference Store Access

In some cases, you may prefer to avoid the use of the Java Object Cache by your WSRP preference store. You can configure the use of caching by the WSRP preference store with the following JNDI variable:

`oracle/portal/wsrp/server/disableJavaObjectCache`

By default, this variable is set to `false`. You can set the variable yourself in the `web.xml` file for your portlet application as follows:

```xml
<env-entry>
  <env-entry-name>oracle/portal/wsrp/server/disableJavaObjectCache</env-entry-name>
  <env-entry-type>java.lang.Boolean</env-entry-type>
  <env-entry-value>true</env-entry-value>
</env-entry>
```
For more information about setting JNDI variables, see Section 32.2.4.2, "Setting JNDI Variable Values."

### 32.1.7 How to Implement Security for JSR 168 Portlets

You can secure JSR 168 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 168 portlet through its WSRP producer, see Section 37.12, "Securing Identity Propagation Through WSRP Producers with WS-Security."

### 32.2 Enhancing PDK-Java Portlets

When you have built your initial portlet in the Create Oracle PDK-Java Portlet wizard as described in Section 31.2.3, "How to Create a PDK-Java Portlet," the next step is to enhance it. You can find the JavaDoc reference for the PDK-Java on OTN at:

http://www.oracle.com/technology/products/webcenter/portlet_download.html

This section includes the following subsections that describe some enhancements that you might want to perform:

- Section 32.2.1, "How to Add Portlet Modes."
- Section 32.2.2, "How to Implement Public Parameters."
- Section 32.2.4, "How to Use JNDI Variables."
- Section 32.2.5, "How to Access Session Information."
- Section 32.2.6, "How to Enhance Portlet Performance with Caching."

The source code for many of the examples referenced in this section is available as part of the Portlet Developer’s Kit (PDK).

When you unzip PDK-Java, find the examples in:

../pdk/jpdk/v2/src/oracle/portal/sample/v2/devguide

### 32.2.1 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 31.2.3, "How to Create a PDK-Java Portlet." For each portlet mode that you select in the wizard, a basic skeleton is created. If you want to add a portlet mode after creating the portlet, you can do that manually by updating `provider.xml` and HTML or JSPs in JDeveloper.

The principles of implementing portlet modes using `RenderManager` are the same for all modes.

For more detailed information about the PDK runtime classes used in this section, see the JavaDoc on OTN:

http://www.oracle.com/technology/products/webcenter/portlet_download.html

For more information about the syntax of `provider.xml`, see the provider JavaDoc, also available on OTN.

**Before You Begin**

The steps that follow assume that you have:
Built a portlet using the Create Oracle PDK-Java Portlet wizard
Successfully registered the portlet’s producer
Added the portlet to a page

To add a portlet mode:
1. In the JDeveloper Application Navigator, open the application that contains the portlet.
2. Expand the project that contains the portlet.
3. Expand the Web Content node and then thehtdocs node and then the node for the producer.
4. Right-click the node for the portlet and choose New.
   The node for the portlet is located under Web Content > htdocs > provider.
5. In the New Gallery, expand Web Tier, select HTML or JSP, and click OK.
   You must create an HTML file or a JSP for each mode to add to your portlet. For example, to implement Help mode, create an HTML file to provide the help content.
6. In the resulting dialog, enter a file name for the HTML file or JSP and click OK.
7. In the visual editor, edit the content of the page to implement the desired functionality.
   For example, for Help mode, you could add the following HTML:
   `<p>This is the Help mode of your portlet!</p>`
8. In the Application Navigator, right-click the provider.xml file for the provider that owns the portlet and choose Open.
   The provider.xml file is located under Web Content > WEB-INF > providers > provider.
9. Change the value of the appropriate tag for the mode you are adding to true.
   For example, if you are adding Help mode, change the hasHelp tag as follows:
   `<hasHelp>true</hasHelp>`
   This indicates to the PDK Framework that a link or icon to that mode should be rendered.
10. Add the code to point to the HTML page or JSP that you created earlier for the mode.
    For example, for the Help page, add the following code:
    `<helpPage>/htdocs/myprovider/myportlet/myHelpPage.html</helpPage>`
11. Save your changes.
12. Redeploy your portlet. For more information, see Chapter 33, "Testing and Deploying Your Portlets."
    When you redeploy, JDeveloper automatically saves and compiles the code before deploying the portlet.
13. Copy the HTML file or JSP you created and the updated provider.xml file to the WLS instance where you plan to deploy the portlet.
This step is not necessary if you have redeployed the producer application to a server instance.

14. Refresh the producer.

15. Refresh the page containing your portlet.

You should now be able to access the new mode. For example, if you added Help mode, you should be able to click the Help link.

### 32.2.2 How to Implement Public Parameters

PDK-Java and Oracle WebCenter Framework provide public and private portlet parameters to enable portlet developers to easily write reusable, complex portlets. The Create Oracle PDK-Java Portlet wizard in JDeveloper creates portlets that are set up to use parameters. This feature enables you to focus solely on adding business logic to your portlets and does not require any changes to `provider.xml`.

For an overview of parameters, see Section 29.2.11, "Public Portlet Parameter Support" and Section 29.2.12, "Private Portlet Parameter Support."

#### Before You Begin

The steps that follow assume that you have:

- Followed through and understood Section 31.2.3, "How to Create a PDK-Java Portlet."
- Built a portlet using the wizard and successfully added it to a page.

---

**Note:** Each portlet is limited to 4K of data. The lengths of parameter and event names, display names, and descriptions all contribute toward this 4K limit. Hence, you should not use too many parameters and events for each portlet, or give them lengthy names and descriptions.

---

Using the Create Oracle PDK-Java Portlet wizard, you can easily create a portlet with public parameters. When you register the producer and drop the portlet on a page, the portlet's parameters are automatically linked to page variables.

**To create a portlet with public parameters to your portlet:**

1. In the JDeveloper Application Navigator, open the application that contains the portlet.

2. Right-click the project under which you want to create your portlet, and choose New.

   **Note:** To create the portlet in an existing producer, right-click the producer's `provider.xml` file and choose Add Portlet. This takes you directly to the General Portlet Information page of the Create Oracle PDK-Java Portlet wizard.

3. In the New Gallery, expand Web Tier, select Portlets and then Oracle PDK-Java Portlet, and click OK.

4. Proceed through the Create Oracle PDK-Java Portlet wizard until you reach the Public Portlet Parameters page.
See Section 31.2.3, "How to Create a PDK-Java Portlet" for basic information about going through the wizard.

5. On the Public Portlet Parameters page (Figure 32–3), click Add.

This adds a new row to the table of parameters.

Figure 32–3 Public Portlet Parameters Page of Portlet Wizard

6. Replace the default values in the Name, Display Name, and Description fields with something more meaningful for your parameter.

7. Add more parameters as required and then click Finish.

8. In the Application Navigator, right-click the provider.xml file for the provider that owns the portlet and choose Open.

The provider.xml file is located under Web Content > WEB-INF > providers > provider.

You should see entries for the parameters that you added on the Public Portlet Parameters page in the wizard, for example, as shown in Example 32–8.

Example 32–8 provider.xml Sample, Public Parameters

```xml
<?xml version = '1.0' encoding = 'UTF-8'?>
<providerDefinition version="3.1">
  <provider class="oracle.portal.provider.v2.DefaultProviderDefinition">
    <session>false</session>
    <passAllUrlParams>false</passAllUrlParams>
    <preferenceStore class="oracle.portal.provider.v2.preference.FilePreferenceStore">
      <name>prefStore1</name>
      <useHashing>true</useHashing>
    </preferenceStore>
    <portlet class="oracle.portal.provider.v2.DefaultPortletDefinition">
      <id>1</id>
      <name>MyPortlet</name>
      <title>My Portlet</title>
    </portlet>
  </provider>
</providerDefinition>
```
<description>My Portlet Description</description>
<timeout>40</timeout>
<showEditToPublic>false</showEditToPublic>
<hasAbout>false</hasAbout>
<showEdit>true</showEdit>
<hasHelp>false</hasHelp>
<showEditDefault>false</showEditDefault>
<showDetails>false</showDetails>
<inputParameter class="oracle.portal.provider.v2.DefaultParameterDefinition">
  <name>Parameter_01</name>
  <displayName>Parameter_01</displayName>
  <description>My first parameter</description>
</inputParameter>
<inputParameter class="oracle.portal.provider.v2.DefaultParameterDefinition">
  <name>Parameter_02</name>
  <displayName>Parameter_02</displayName>
  <description>My second parameter</description>
</inputParameter>
<inputParameter class="oracle.portal.provider.v2.DefaultParameterDefinition">
  <name>Parameter_03</name>
  <displayName>Parameter_03</displayName>
</inputParameter>
<renderer class="oracle.portal.provider.v2.render.RenderManager">
  <renderContainer>true</renderContainer>
  <renderCustomize>true</renderCustomize>
  <autoRedirect>true</autoRedirect>
  <contentType>text/html</contentType>
  <showPage>/htdocs/myportlet/MyPortletShowPage.jsp</showPage>
  <editPage>/htdocs/myportlet/MyPortletEditPage.jsp</editPage>
</renderer>
<personalizationManager class="oracle.portal.provider.v2.personalize.PrefStorePersonalizationManager">
  <dataClass>
    oracle.portal.provider.v2.personalize.NameValuePersonalizationObject
  </dataClass>
</personalizationManager>
</portlet>
</provider>

9. In the Application Navigator, right-click the portletnameShowPage.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 32–9.

Example 32–9  ShowPage.jsp Sample

```jsp
<%@page contentType="text/html; charset=windows-1252"
import="oracle.portal.provider.v2.render.PortletRenderRequest"
import="oracle.portal.provider.v2.ParameterDefinition"
%
PortletRenderRequest pReq = (PortletRenderRequest)
  request.getAttribute(HttpCommonConstants.PORLET_RENDER_REQUEST);
%>
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 custom WebCenter application. For more information, see Section 34.2.3, "How to Register an Oracle PDK-Java Portlet Producer."

13. Add the two portlets to a page in the application. For more information, see Section 34.3, "Adding Portlets to a Page."

The page definition should look similar to Example 32–10. Notice the variables at the page level and the parameters at the portlet level (indicated in bold).

**Example 32–10  Page Definition File Sample**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<pageDefinition xmlns="http://xmlns.oracle.com/adfm/uimodel" version='10.1.3.38.90' id='untitled1PageDef' Package='view.pageDefs'>
<executables>
  <variableIterator id="variables">
    <variable Name="portlet1_Parameter_01" Type="java.lang.Object"/>
    <variable Name="portlet1_Parameter_02" Type="java.lang.Object"/>
    <variable Name="portlet1_Parameter_03" Type="java.lang.Object"/>
  </variableIterator>
  <portlet id="portlet1" portletInstance="/oracle/adf/portlet/PdkPortletProducer1_1153936627784/applicationPortlets/Portlet1_abfc5a10_010c_1000_8003_82235f50d831" class="oracle.adf.model.portlet.binding.PortletBinding" xmlns="http://xmlns.oracle.com/portlet/bindings">
    <parameters>
      <parameter name="Parameter_01" pageVariable="portlet1_Parameter_01"/>
      <parameter name="Parameter_02" pageVariable="portlet1_Parameter_02"/>
      <parameter name="Parameter_03" pageVariable="portlet1_Parameter_03"/>
    </parameters>
  </portlet>
</executables>
</pageDefinition>
```

15. Run the application.

16. Enter values in the first portlet and the same values should be displayed in the second portlet.

### 32.2.3 How to Implement Private Parameters

In some cases, you might need a parameter that is known only to the portlet instance. These parameters are known as private parameters because they have no connection to the page and are known only to the portlet. Private parameters often come in handy when you are building navigation for your portlet. For example, if you have a portlet made up of multiple pages, then you can use these private parameters to jump to another resource of the portlet.

This section includes the following subsections:

- Section 32.2.3.1, "About Private Parameters"
- Section 32.2.3.2, "About Portlet URL Types"
- Section 32.2.3.3, "Building Links with the Portlet URL Types"
- Section 32.2.3.4, "Building Forms with the Portlet URL Types"
- Section 32.2.3.5, "Implementing Navigation within a Portlet"
- Section 32.2.3.6, "Restricting Navigation to Resources"

#### 32.2.3.1 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 custom WebCenter application, the client does not communicate directly with portlets. Instead, Oracle WebCenter Framework mediates between the client and the portlet. Moreover, because most pages have multiple portlets, Oracle WebCenter 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:

\[
\text{HttpPortletRendererUtil.portletParameter(HttpServletRequest request, String param)}
\]

HttpPortletRendererUtil is in the package oracle.portal.provider.v2.render.http.

For example:

\[
\text{qualParamQ = HttpPortletRendererUtil.portletParameter(r, "q");}
\]

To fetch the value of a portlet parameter from the incoming request, you can use the following API:

\[
\text{PortletRenderRequest.getQualifiedParameter(String name)}
\]

PortletRenderRequest is in the package oracle.portal.provider.v2.render.

For example:

\[
\text{valueQ = r.getQualifiedParameter("q");}
\]

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 32.2.3.3, "Building Links with the Portlet URL Types" and Section 32.2.3.4, "Building Forms with the Portlet URL Types."
32.2.3.2 About Portlet URL Types
When a portlet renders itself, Oracle WebCenter 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).

32.2.3.3 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:

```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 32.2.3.4, "Building Forms with the Portlet URL Types."

```jsp
<% String paramNameQ = "q";
    String qualParamNameQ =
        HttpServletRequestPortletUtils.portletParameter(paramNameQ);
    PortletRenderRequest pRequest = (PortletRenderRequest)
        request.getAttribute(HttpCommonConstants.PORTLET_RENDER_REQUEST);
    String paramValueQ = pRequest.getQualifiedParameter(paramNameQ);
%>
<!-- Output the HTML content -->
<center>
    Words similar to <%= paramValueQ %>
    <br>
    Click the link to search for words related to that word.
    <br>
    <ul>
    <% String[] relatedWords = Thesaurus.getRelatedWords(paramValueQ);
        NameValue[] linkParams = new NameValue[1];
        for (int i=0; i<=relatedWords.length; i++)
        {
            linkParams[0] = new NameValue(
                qualParamNameQ, relatedWords[i]);
        }
    %> 
    </ul>
    </center>
</s>```

### 32.2.3.4 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 32.2.3.3, "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)`.

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 32.2.3.3, "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) %>'>
```

---

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.

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 32.2.3.3, "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) %>'>
```
32.2.3.5 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 32.2.3.3, "Building Links with the Portlet URL Types" and Section 32.2.3.4, "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 32.2.5, "How to Access Session Information."

The following portlet code comes from the multipage 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>
```
Enhancing PDK-Java Portlets

You can modify the thesaurus example to operate with the use of this parameter. Specifically, you can use the form submission portlet to be the input for the thesaurus (the first page of the portlet), then navigate the user to the results page, which contains links to drill further into the thesaurus. The following examples illustrate these changes.

**Note:** The value of `pageParameterName` is the name of a portlet parameter, `next_page`, that the PDK-Java framework intercepts and interprets as an override to the value of the `showPage` parameter. If the PDK-Java framework encounters the qualified version of the parameter when the multipage portlet is requested, then it renders the resource identified by `next_page` rather than `first.jsp`. PDK-Java does not render the parameter within the portlet, that responsibility falls to the portlet.

You can modify the thesaurus example to operate with the use of this parameter. Specifically, you can use the form submission portlet to be the input for the thesaurus (the first page of the portlet), then navigate the user to the results page, which contains links to drill further into the thesaurus. The following examples illustrate these changes.

**Note:** The example that follows is most useful for relatively simple cases, such as this thesaurus example. If your requirements are more complex (for example, you want to build a wizard experience), then you should consider using an MVC framework such as Struts. For information about how to build portlets from struts applications, see Section 32.4, "Creating a Struts Portlet."

**ThesaurusForm.jsp:**

```jsp
<%@ Page language="java" pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/strutforms" prefix="form" %>

<form action="#{Actions.searchThesaurusAction}" method="post">
  <input type="hidden" name="showPage" value="first.jsp" />
  <input type="text" name="query" size="20" />
  <input type="submit" value="Submit" />
</form>
```

**Note:** The example that follows is most useful for relatively simple cases, such as this thesaurus example. If your requirements are more complex (for example, you want to build a wizard experience), then you should consider using an MVC framework such as Struts. For information about how to build portlets from struts applications, see Section 32.4, "Creating a Struts Portlet."
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 -->
<%@ page language="en" contentType="text/html" pageEncoding="UTF-8" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Thesaurus</title>
</head>
<body>
<%  
Thesaurus t = new Thesaurus();
String[] relatedWords = t.getRelatedWords(paramValueQ);
NameValue[] linkParams = new NameValue[2];
linkParams[0] = new NameValue(qualParamNameNextPage, "htdocs/path/ThesaurusLink.jsp");
for (int i=0; i<relatedWords.length; i++)
{
    linkParams[1] = new NameValue(qualParamNameQ, relatedWords[i]);
%>
<li>
<b> <%= relatedWords[i] %> </b>
<%= UrlUtils.constructHTMLLink(pRequest, UrlUtils.PAGE_LINK, 
"(words related to " + relatedWords[i] + ")",  
"", 
linkParams,  
true, 
true)%>
</li>
<%  }
%
</ul>
</body>
</html>
```

<a href="<%=XMLUtil.escapeXMLAttribute(pRequest.getRenderContext().getPageURL())%>">
Reset Portlet
</a>
</center>
```
32.2.3.6 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 32–11 allows resources under /htdocs/multiportlet to be shown. All other resources are restricted.

**Example 32–11  Whitelist Excerpt from the provider.xml File**

```
<portlet class="oracle.portal.provider.v2.DefaultPortletDefinition">
  <id>1</id>
  <name>Multipage</name>
  <title>A MultiPage Portlet</title>
  ...
  <renderer class="oracle.portal.provider.v2.render.RenderManager">
    <contentType>text/html</contentType>
    <showPage>/htdocs/multiportlet/controller.jsp</showPage>
    <pageParameterName>show_page</pageParameterName>
    <allowedPath>/htdocs/multiportlet/*</allowedPath>
  </renderer>
</portlet>
```

The pattern matching rules for this feature are similar to URL pattern matching in web.xml files. The rules are as follows:

- To match the defined patterns, the resource path must exactly match unless wildcards are used.
- The first wildcard is for path matching and consists of a string beginning with / and ending with /*. Any resource whose path starts with this string is matched. For an <allowedPath> value of /htdocs/sub1/*, valid values of the private parameter include /htdocs/sub1/file.jsp and /htdocs/sub1/sub2/file2.jsp.
- The second wildcard is for file type matching and consists of a string starting with *. and ending with a file extension. Valid values for the page parameter end with that file extension. For an <allowedPath> value of *.jsp, valid values of the private parameter include /htdocs/sub1/file.jsp and /htdocs/sub1/sub2/file2.jsp.
32.2.4 How to Use JNDI Variables

When writing Java portlets, you may set deployment-specific properties through the JNDI service such that their values may be retrieved from your producer code. In this way, you can specify any property in a producer deployment and then easily access it anywhere in your producer code.

You can use JNDI variables to change producer property values after the producer has been deployed. The environment entry must be declared in web.xml. It can then be updated on deployment using a deployment plan.

PDK-Java provides utilities to enable the retrieval of both producer and non-producer JNDI variables within a Java EE container.

This section includes the following subsections:
- Section 32.2.4.1, "Declaring JNDI Variables"
- Section 32.2.4.2, "Setting JNDI Variable Values"
- Section 32.2.4.3, "Retrieving JNDI Variables"

32.2.4.1 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>
```

The env-entry-name element contains the name by which you want to identify the variable. env-entry-type contains the fully qualified Java type of the variable. env-entry-value contains the variable's default value.

This section includes the following subsections:
- Section 32.2.4.1.1, "Variable Types"
- Section 32.2.4.1.2, "Variable Naming Conventions"
- Section 32.2.4.1.3, "Examples"

32.2.4.1.1 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.

32.2.4.1.2 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 32.2.4.3, "Retrieving JNDI Variables."

### 32.2.4.1.3 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`

### 32.2.4.2 Setting JNDI Variable Values

In your producer deployment, you may want to set a new value for some or all of your JNDI variables. You can perform this task by setting the values manually within a WLS deployment plan. Deployment plans can be created for the producer deployment through the WLS console.

**To set variable values manually within a deployment plan:**

1. Go to the producer deployment with the WLS console and create a new deployment plan if one does not exist.

2. Edit the deployment plan XML file. For each deployment property you want to set, add the following variable definition directly under the `<deployment-plan>` tag:

   `<variable-definition>`
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"]</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.

### 32.2.4.3 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 32.2.4.1.2, "Variable Naming Conventions" and are of the form:

- `oracle/portal/provider_service_name/variable_name`
- `oracle/portal/variable_name`

To use these APIs, you need only provide the `provider_service_name` and the `variable_name`. The utilities construct the full JNDI variable name, based on the information you provide, and look up the variable using code similar to that shown earlier and return the value of the variable.

The `EnvLookup` class (oracle.portal.utils.EnvLookup) provides two `lookup()` methods. One retrieves producer variables and the other retrieves non-producer variables. Both methods return a `java.lang.Object`, which can be cast to the Java type you are expecting.
The following code example illustrates the retrieval of a producer variable:

```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 32–1 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>
<tr>
<td>oracle/portal/provider/provider_name/definition</td>
<td>Location of producer's definition file.</td>
</tr>
<tr>
<td>oracle/portal/provider/global/log/logLevel</td>
<td>Log setting (0 through 8). 0 being no logging and 8 the most possible logging.</td>
</tr>
<tr>
<td>oracle/portal/provider/provider_name/maxTimeDifference</td>
<td>Producer's HMAC time difference.</td>
</tr>
<tr>
<td>oracle/portal/provider/&lt;service_name&gt;/resourceUrlKey</td>
<td>Authentication key for resource proxying through the Parallel Page Engine. See Oracle Fusion Middleware Administrator’s Guide for Oracle Portal for more information.</td>
</tr>
<tr>
<td>oracle/portal/provider/provider_name/rootDirectory</td>
<td>Location for producer personalizations. No default value.</td>
</tr>
<tr>
<td>oracle/portal/provider/provider_name/sharedKey</td>
<td>HMAC shared key. No default value.</td>
</tr>
<tr>
<td>oracle/portal/provider/provider_name/showTestPage</td>
<td>(non-producer) A Boolean flag that determines if a producer's test page is accessible. Defaults to true.</td>
</tr>
<tr>
<td>oracle/portal/provider/global/transportEnabled</td>
<td>A Boolean flag that determines whether Edit Defaults personalizations may be exported and imported.</td>
</tr>
</tbody>
</table>
32.2.5 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.
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 on OTN:

http://www.oracle.com/technology/products/webcenter/portlet_download.html

In the example in this section, session storage is used to count the number of times your portlet has rendered in Shared Screen mode.

**Before You Begin**

The steps that follow assume that you have:

- Followed through and understood Section 31.2.3, "How to Create a PDK-Java Portlet."
- Built a portlet using the wizard and successfully added it to a page.

**To implement session storage:**

- Import ProviderSession, PortletRendererUtil, and HttpPortletRendererUtil.
- Retrieve the producer session.
- Read and write the session by accessing it from within your Java portlet.
- Set the session to true in provider.xml.
- Register the producer for session storage and set the Login Frequency.

The steps that follow describe how to add a session count to your portlet that displays how many times the portlet has been rendered for the current session.

1. After using the wizard to create a portlet, you can edit the JSP for the Show page in Oracle JDeveloper. You must import the following classes:

```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.

```java
<% PortletRenderRequest pReq = (PortletRenderRequest) request.getAttribute(HttpCommonConstants.PORTLET_RENDER_REQUEST); ProviderSession pSession = pReq.getSession(); if (pSession != null) {
    String key = PortletRendererUtil.portletParameter(pReq, "count");
    Integer i = (Integer) pSession.getAttribute(key);
    if (i == null) {
        i = new Integer(0);
    }
    i = new Integer(i.intValue()+1);
    pSession.setAttribute(key, i);
}%>
<p>Render count in this session: <%=i%></p>
<%
} else {
%
<p>The session has become invalid</p>
<br>
Please log out and log in again.
<%
}
%
```

3. By default, the wizard does not set session to true in provider.xml. You must update this flag in order for the producer to receive session information from the portal. You should only set this tag to true if you are using session information in your producer or portlets. By setting this flag to true, extra load is added to the producer calls.

```xml
<provider class="oracle.portal.provider.v2.DefaultProviderDefinition">
  <session>true</session>
</provider>
```

For more information about the syntax of provider.xml, see the provider JavaDoc on OTN:

http://www.oracle.com/technology/products/webcenter/portlet_download.html

4. Register your producer with session support. For a reminder on how to register your portlet, see Section 33.4, "Registering and Viewing Your Portlet."

5. If you have not added your Java portlet to a page, then do so now. Ensure that you perform the following tasks:
Enhancing PDK-Java Portlets

- Refresh the producer to accept the new changes.
- Re-login in case your session is no longer valid.

32.2.6 How to Enhance Portlet Performance with Caching

Once you have completed the basic functionality of your portlet, you may want to turn your attention to portlet performance.

Caching is a common technique for enhancing the performance of web sites that include a great deal of dynamic content. The overhead involved in retrieving data and generating the output for dynamic content can be significantly reduced by proxying requests through a local agent backed by a large, low-latency data store known as a cache. The cache agent responds to a request in one of two ways, as follows:

- If a valid version of the requested content exists in the cache, then the agent simply returns the existing cached copy, thus skipping the costly process of content retrieval and generation. This condition is called a cache hit.
- If a valid version of the requested content does not exist in the cache, then the agent forwards the request to its destination and awaits the return of the content. The agent returns the content to the requester and stores a local copy in its cache for reuse if a subsequent request for the same content arises. This condition is called a cache miss.

Producers generate dynamic content (that is, portlets) and they reside remotely from the custom WebCenter application instance on which they are deployed. As such, caching might improve their performance. The architecture lends itself well to caching. You can cache the portlets rendered by your producer and reuse the cached copies to handle subsequent requests, minimizing the overhead your producer imposes on page assembly.

The producer can use any one of three different caching methods, depending upon which one is best suited to the application. The methods differ chiefly in how they determine whether content is still valid. Following are the three caching methods:

- **Expiry-based Caching**: When a producer receives a render request, it stamps its response with an expiry time. The rendered response remains in the cache and fills all subsequent requests for the same content until its expiry time passes. This caching scheme is perhaps the simplest and most performant because the test for cache validity requires very little overhead and does not involve any network round trips. Expiry-based caching suits applications where the content has a well-defined life span. For content with a less certain life span, however, expiry-based caching is less effective. For more information, see Section 32.2.6.1, "Activating Caching" and Section 32.2.6.2, "Adding Expiry-Based Caching."

- **Validation-based Caching**: When a producer receives a render request, it stamps its response with a version identifier (or E Tag). The response goes into the cache, but, before the PPE can reuse the cached response, it must determine whether the cached version is still valid. It sends the producer a render request that includes the version identifier of the cached content. The producer determines whether the version identifier remains valid. If the version identifier is still valid, then the producer immediately sends a lightweight response to the PPE without any content, which indicates the cached version can be used. Otherwise, the producer generates new content with a new version identifier, which replaces the previously cached version. In this form of caching, the PPE must always confirm with the producer whether the content is up to date. The validity of the cached copy is determined by some logic in the producer. The advantage of this approach is that the producer controls the use of the cached content rather than relying on a...
fixed period. For more information, see Section 32.2.6.1, "Activating Caching" and Section 32.2.6.3, "Adding Validation-Based Caching."

This section includes the following subsections:

- Section 32.2.6.1, "Activating Caching"
- Section 32.2.6.2, "Adding Expiry-Based Caching"
- Section 32.2.6.3, "Adding Validation-Based Caching"

**Before You Begin**

The steps that follow assume that you have:

- Followed through and understood Section 31.2.3, "How to Create a PDK-Java Portlet."
- Built a portlet using the wizard and successfully added it to a page.

### 32.2.6.1 Activating Caching

To use the caching features in your producers, you must first activate the middle tier cache. This cache is known as the PL/SQL Cache because it is the same cache used by mod_plsql, the Oracle HTTP Server plug-in that calls database procedures, and hence database producers, over HTTP.

Usually, your administrator is responsible for the configuration details of caching.

### 32.2.6.2 Adding Expiry-Based Caching

Expiry-based caching is a simple caching scheme to implement, and can be activated declaratively in your XML producer definition. You can set an expiry time for the output of any ManagedRenderer you use by setting its `pageExpires` property to the number of minutes you want the output to be cached for. For example, suppose you want portlet output to be cached for one minute.

**To add expiry-based caching:**

1. After you have used the Create Oracle PDK-Java Portlet wizard to build a portlet as described in Section 31.2.3, "How to Create a PDK-Java Portlet," edit the `provider.xml` file and set the `pageExpires` property tag of `showPage` to 1. This sets an expiry entry of one minute for the portlet.

   By default the wizard generates a standard and compressed tag for `showPage`.
   Expand the tag to include a subtag of `pageExpires`:

   ```xml
   <showPage class="oracle.portal.provider.v2.render.http.ResourceRenderer">
     <resourcePath>/htdocs/mycacheportlet/MyCachePortletShowPage.jsp
     <pageExpires>1</pageExpires>
   </showPage>
   ```

   For more information about the syntax of `provider.xml`, see the provider JavaDoc on OTN:


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.

32.2.6.3 Adding Validation-Based Caching

Adding validation-based caching requires slightly more effort, but gives you explicit 
control over exactly which requests to your producer are cache hits. As an example, 
you may want to update the cache only when data within the portlet has changed. To 
implement this algorithm, you must override the prepareResponse method. The 
signature of the BaseManagedRenderer.prepareResponse method is:

    public boolean prepareResponse(PortletRenderRequest pr) 
      throws PortletException, 
        PortletNotFoundException

In your version of prepareResponse(), do the following:

- Retrieve the cached version identifier set by the PPE in the render request by 
calling the HttpPortletRendererUtil.getCachedVersion() method:

    public static java.lang.String getCachedVersion 
      (PortletRenderRequest request)

- If the portlet finds the previously cached version valid, then the appropriate 
  header must be set by calling the 
  HttpPortletRendererUtil.useCachedVersion() method. It also instructs 
  the RenderManager that it is not necessary to call renderBody() to render the 
  portlet body.

    public static void useCachedVersion(PortletRenderRequest request)

Otherwise, use HttpPortletRendererUtil.setCachedVersion() to 
generate a new version of the portlet, which is cached. It also indicates to the PPE 
that the renderBody() method has to be called to regenerate the portlet content.

    public static void setCachedVersion(PortletRenderRequest request, 
      java.lang.String version, 
      int level) 
      throws java.lang.IllegalArgumentException

For validation-based caching, you need not update provider.xml. You can view the 
portlet by refreshing the page or adding the portlet to a page and updating the
content. If the content has changed, then the portlet shows the new content. If the content has not changed, then a cached version of the portlet is displayed.

### 32.3 Testing Portlet Personalization

If you have implemented personalization for your portlet, then the Personalize link only appears on the portlet for authenticated users. Hence, to test the personalization of a portlet (the Personalize link), you must have some form of security implemented for the application consuming the portlet. For testing purposes, you may prefer to just configure the most basic authentication possible. For more information, see Section 37.8, "Configuring Basic Authentication for Testing Portlet Personalization."

### 32.4 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>
...```

For more information about the syntax of provider.xml, see the provider JavaDoc on OTN:

http://www.oracle.com/technology/products/webcenter/portlet_download.html

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 32–4 and Figure 32–5 show how you submit a blog and save a blog entry.

**Figure 32–4  Submitting a Blog**

![](image)

`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"/>
```

**Figure 32–5  Saving a Blog Entry**

![](image)
Creating a Struts Portlet

This section includes the following subsections:

- Section 32.4.1, "How to Create a New Flow and View to Host the Portlet Actions"
- Section 32.4.2, "How to Create the New JSPs"
- Section 32.4.3, "How to Create a Portlet"
- Section 32.4.4, "How to Extend the Portlet to Add Business Logic"
- Section 32.4.5, "How to Register the Producer"

### 32.4.1 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>
```

```xml
<action path="/portal/saveNewBlog" name="blogForm" scope="request"
    type="view.SaveNewBlogAction" input="/view/enterNewBlog.jsp" >
    <forward name="success" path="/view/portal/newBlogConfirmation.jsp"/>
</action>
```

As you can see, only the path attributes are modified. The FormBean Action responsible for the application business logic remains unchanged.

### 32.4.2 How to Create the New JSPs

As specified in the previous step, the actions forward the request to new JSPs, which are responsible for rendering the portlet content. Your new portlet view JSPs can share the HTML with the standalone view, but ensure the portlet meets the following criteria:

- Uses styles that enforce a consistent look and feel with the rest of the page.
- Contains HTML code that is allowed in HTML table cells (that is, no `<html>`, `<body>`, and `<frame>` tags).
- Uses the PDK-Java version of the Struts HTML tag library. This renders URLs in the correct format to work in a portlet context.

The PDK Struts HTML tag library contains versions of the Struts HTML tags that can be used in portlet markup.

---

**Note:** You can register the Oracle PDK with Oracle JDeveloper so that you can drop the tags from the Oracle JDeveloper Components Palette. For more information, see the Registering a Custom Tag Library in JDeveloper section in the Oracle JDeveloper online Help.

---

### 32.4.3 How to Create a Portlet

You can create your Struts portlet either manually or by using the Create Oracle PDK-Java Portlet wizard. Although the wizard does not explicitly offer Struts support, you can use the wizard to build your Struts portlet.
Creating a Struts Portlet

To create a portlet:

1. In JDeveloper, open the Oracle PDK-Java Portlet wizard.
   For more information, see Section 31.2.3, "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 synchrionizer token mechanism is used or <actionInSession> is set to true), then enable session handling:
     <session>true</session>
   At the portlet level, perform the following:
   - If you want users to always return to the same portlet state as when they left the container page, then you can configure the struts renderer to save the struts action in the strutscontext:
     <actionInSession>true</actionInSession>
   If you prefer that users always start from the beginning of the portlet when they return from outside the container page, then you should not save the struts action:
     <actionInSession>false</actionInSession>
   Note that this setting is the default behavior.
   - Specify the first action to raise when the portlet is called. Use the following code:
     <showPage class="oracle.portal.provider.v2.render.http.StrutsRenderer">  
     <defaultAction>/portal/prepareNewBlog.do</defaultAction>
   </showPage>
   For more information about the syntax of provider.xml, see the provider JavaDoc on OTN:
   http://www.oracle.com/technology/products/webcenter/portlet_download.html

32.4.4 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.
32.4.5 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 33.4, "Registering and Viewing Your Portlet."
This chapter explains how to test and deploy JSR 168 and Oracle PDK-Java portlets. In a development environment, you can test your portlets on Integrated WebLogic Server (WLS), which comes packaged with Oracle JDeveloper, and also deploy portlet applications to an Oracle WebLogic Managed Server instance residing outside JDeveloper.

This chapter includes the following sections:

- Section 33.1, "Introduction to Portlet Deployment Testing in a Development Environment"
- Section 33.2, "Testing a Portlet Application on Integrated WebLogic Server"
- Section 33.3, "Deploying a Portlet Application to an Oracle WebLogic Managed Server Instance"
- Section 33.4, "Registering and Viewing Your Portlet"

For information about testing and deploying other custom WebCenter applications, see Chapter 38, "Testing and Deploying Your WebCenter Application".

For information about creating portlets, see Chapter 30, "Creating Portlets with the Oracle JSF Portlet Bridge" and Chapter 31, "Creating Portlets with the Portlet Wizard".

33.1 Introduction to Portlet Deployment Testing in a Development Environment

Before you deploy your portlet application, you are advised to test it in a development environment. Integrated WLS in JDeveloper enables you to test your portlets in a single step, without creating a deployment profile. To learn the benefits of testing applications on Integrated WLS, see Integrated WebLogic Server. An additional benefit is that portlet customizations that you perform at design time are maintained in your application workspace and become an integral part of the application source definition. These changes are packaged with the EAR file when the application is deployed to a WebLogic Managed Server instance (managed servers). This eliminates the requirement to export and import portlet customizations.

If, while testing your application with Integrated WebLogic Server, you make changes at runtime, these customizations are also retained. If you do not want to keep these runtime customizations when you deploy your application to a managed server, choose the menu option Build > Clean Runtime MDS Customizations.

To test your portlet application on an Oracle WebLogic Managed Server instance, or to deploy for production, you must configure a connection to the managed server, create deployment plans, and then deploy to the server instance as described in this chapter.
For information about deploying portlet applications using Oracle Enterprise Manager Fusion Middleware Control, WLS Administration Control, or WLST commands, see the chapter, "Managing Portlet Producers" in Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter.

33.2 Testing a Portlet Application on Integrated WebLogic Server

The connection to Integrated WLS (Default Server) connection is preconfigured and shows as IntegratedWLSConnection under the Application Server node in the IDE Connections panel of the Resource Palette, as shown in Figure 33–1. You can run multiple applications simultaneously and watch the progress of each application in the Run Manager panel. The Run Manager panel also lets you stop Default Server instances.

Figure 33–1 IntegratedWLSConnection in IDE Connections

For more information about Integrated WLS Server, see Integrated WebLogic Server.

This section includes the following subsections:

■ Section 33.2.1, "How to Test JSR 168 Portlets on Integrated WebLogic Server"

■ Section 33.2.2, "What Happens When You Test JSR 168 Portlets on Integrated WebLogic Server"

■ Section 33.2.3, "How to Test PDK-Java Portlet Applications on Integrated WebLogic Server"

■ Section 33.2.4, "What Happens When You Test PDK-Java Portlet Applications on Integrated WebLogic Server"
33.2.1 How to Test JSR 168 Portlets on Integrated WebLogic Server

When you run your portlet application on Integrated WLS, an instance of Default Server starts.

To test a JSR 168 portlet application:

1. Right-click the portlet.xml page in the project folder and select Run. Running portlet.xml triggers packaging and deployment of your portlet application on an instance of Integrated WLS named after the application.

   **Note:** When the Integrated WLS instance stops, the application is undeployed, and therefore, becomes unavailable.

For a more persistent testing scenario, you can deploy your portlet application to Integrated WLS by right-clicking the application and then selecting Deploy. This deploys your application to the Default Server instance so that it is always available when the Default Server is running. Thus, the application remains accessible for multiple consumer applications. If you choose this method, then you must first create deployment profiles, as described in Section 33.3.1, "How to Create Deployment Profiles". If you deploy your application to Integrated WLS, then the Deployment Configuration dialog displays to let you configure and customize deployment settings. The file system MDS repository pre-created by JDeveloper displays in the Repository Name field.

2. In the Select deployment type dialog, select Yes and then click OK.

   See the DefaultServer Log window to monitor the deployment progress. The DefaultServer - 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 33–2 displays in a browser window.
33.2.2 What Happens When You Test JSR 168 Portlets on Integrated WebLogic Server

When you run JSR 168 portlets on Integrated WLS, the following happens:

- wsdls and other configuration files are added to the WEB-INF directory to configure the portlets as a Web service.

- The web.xml file is updated with listener and server classes, filters, parameters, and other configurations that are required to run the JSR 168 portlet 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 168 portlets are added to the weblogic.xml file, for example, oracle.portlet-producer.wsrp.

  These configurations vary depending upon the portlet requirements.

33.2.3 How to Test PDK-Java Portlet Applications on Integrated WebLogic Server

When you run your portlet application on Integrated WLS, an instance of Default Server starts.
To test a portlet application, right-click a JSP page, for example, `index.jsp` in the project folder and select Run. Running `index.jsp` triggers packaging and deployment of your portlet application on an instance of Integrated WLS named after the application.

**Note:** When the Integrated WLS instance stops, the application is undeployed, and therefore, becomes unavailable.

For a more persistent testing scenario, you can deploy your portlet application to Integrated WLS by right-clicking the application and then selecting Deploy. This deploys your application to the Default Server instance so that it is always available when the default server is running. Thus, the application remains accessible for multiple consumer applications. If you choose this method, then you must first create deployment profiles, as described in Section 33.3.1, "How to Create Deployment Profiles." If you deploy your application to Integrated WLS, then the Deployment Configuration dialog displays to let you configure and customize deployment settings. The file system MDS repository pre-created by JDeveloper displays in the Repository Name field.

See the DefaultServer Log window to monitor the deployment progress. The DefaultServer - Log shows the URL of the application test page, as shown in Figure 33–3. The format of the URL is `http://host:port/application_name-Portlets-context-root/index.jsp`.

**Figure 33–3 DefaultServer - Log**

The PDK-Java Application Test page displays in a browser window, as shown in Figure 33–4.
Click the link underneath **Service Name**. Your browser should open with a page similar to the one shown in **Figure 33–5**. The URL of this page is the one required to register the producer with another application.

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 theportlet’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 33–5**.
33.2.4 What Happens When You Test PDK-Java Portlet 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 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`.

33.3 Deploying a Portlet Application to an Oracle WebLogic Managed Server Instance

To test your portlet application on an Oracle WebLogic Managed Server instance, or to deploy it for production, you must first create an application WAR deployment profile, a deployment descriptor, and a connection to the Oracle WebLogic Managed Server instance. Before you deploy or test your portlet application to a managed server, ensure that the managed server is created using the correct template and it contains all the required shared libraries as described in the section "Creating and Provisioning a WebLogic Managed Server Instance" in Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter.

---

**Note:** Before you deploy the application, you must remove the test-all application role if you enabled the automatic grants feature in the Configure ADF Security wizard. Because the test-all role makes all ADF resources public, its presence increases the risk that your application may leave some resources unprotected. You must therefore remove the role before you migrate the application-level policy store.

For more information, see the section "How to Remove the test-all Role from the Application Policy Store" in Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework.

---

For information about deploying portlet applications using Oracle Enterprise Manager Fusion Middleware Control, Oracle WebLogic Administration Console, or WLST commands, see the chapter, "Managing Portlet Producers" in Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter.

This section includes the following subsections:

- Section 33.3.1, "How to Create Deployment Profiles"
- Section 33.3.2, "How to Create and Provision a WebLogic Managed Server Instance"
- Section 33.3.3, "How to Create and Register the Metadata Service Repository"
- Section 33.3.4, "How to Create a WebLogic Managed Server Connection"
33.3.1 How to Create Deployment Profiles

The project-level deployment profile is packaged as a Web Application Archive (WAR) file. Deployment descriptors are server configuration files that define the configuration of an application for deployment and are deployed with the portlet application as needed. The deployment descriptors that a project requires depend on the technologies the project uses and on the type of the target application server.

**Note:** You can deploy PDK-Java portlets only as EAR files. Therefore, while creating deployment profiles, ensure that the WAR file is included in the application's EAR file. For information about how to create an EAR file, see Section 38.3.1.2, "Creating Deployment Profiles" in Chapter 38, "Testing and Deploying Your WebCenter Application".

This section includes the following subsections:

- Section 33.3.1.1, "Creating a WAR Deployment Profile"
- Section 33.3.1.2, "Creating a Deployment Descriptor"

33.3.1.1 Creating a WAR Deployment Profile

To create a WAR deployment profile:
1. In the Application Navigator, expand Web Content and WEB-INF.
2. Right-click web.xml and choose Create WAR Deployment Profile.
   
   Alternatively, from the main menu, choose File and then New. In the New Gallery, expand General, select Deployment Profiles and then WAR File, and click OK.
3. In the Create Deployment Profile -- WAR File dialog, enter a name for your deployment profile and click OK.
4. In the Edit WAR Deployment Profile Properties dialog, select the Specify Java EE Web Context Root option and enter a context root. You can also enter the name of the deployment profile. Then click OK.
5. In the Project Properties dialog, under Deployment Profiles, select the WAR File you just created and click OK.

33.3.1.2 Creating a Deployment Descriptor

To create a deployment descriptor:
1. From the main menu, choose File and then New.
2. In the New Gallery, expand General, select Deployment Descriptors and then Java EE Deployment Descriptor, and click OK.
3. In the Create Java EE Deployment Descriptor dialog, on steps 1 to 4, accept the default values and click Finish.
33.3.2 How to Create and Provision a WebLogic Managed Server Instance
To deploy a portlet producer application to a WebLogic Managed Server instance, you must first create a server instance and provision it with a required set of shared libraries. For more information, see the section "Creating and Provisioning a WebLogic Managed Server Instance" in Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter.

33.3.3 How to Create and Register the Metadata Service Repository
After creating the WebLogic Managed Server instance, you must create and register a Metadata Service Repository (MDS) schema for the application on the WebLogic Domain's Administration Server instance. For more information, see the section "Creating and Registering the Metadata Service (MDS) Repository" in Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter.

33.3.4 How to Create a WebLogic Managed Server Connection
In JDeveloper, you can deploy your portlet applications to Oracle WebLogic Managed Server instances that reside outside JDeveloper. You must first create a connection to the server instance.

Before You Begin:
Before you create a connection to the managed server, ensure that the server instance is up and running.

To create a WebLogic Managed Server connection:
1. From the main menu, select File and then New.
2. In the New Gallery, expand General, select Connections and then Application Server Connection, and click OK.
3. In the Create Application Server Connection wizard, on Step 1, enter a name for the new connection, for example, PortletServer. Then click Next.
4. On Step 2, specify the user name and password for authentication and click Next.
5. On Step 3, enter the host name of the WebLogic Server instance, for example, webcenter.portletserver.example.com and the port number, for example, 7001.
6. In the WLS Domain, specify the name of the domain in which the WebLogic Server instance is created, for example, wc_domain. Click Next.
7. On Step 4, click Test Connection. If the test is successful, the connection is set up.
8. Click Finish.

33.3.5 How to Deploy a Portlet Application to an Oracle WebLogic Managed Server Instance
After you have created the deployment profiles and a connection to the managed server for portlet deployment, you can deploy your portlet application to this server instance.

To deploy a portlet application:
1. In the Application Navigator, open the application to be deployed.
2. Right-click the project folder, choose **Deploy**, **deployment profile name**, to, and then choose the **connection name**.

---

**Note:** If your portlet application contains JSR 168 portlets, then the Select deployment type dialog displays. Click **OK** to add the configuration required to expose this application as a WSRP service.

---

3. In the Select Deployment Target dialog, select the managed server name, for example, **WLS_Portlet**, and click **OK**.

4. In the Deployment Configuration dialog, under the **MDS** tab, in the **Repository Name** dropdown list, select the metadata repository to be used by the application you are deploying.

5. In the **Partition Name** field, specify the application's partition name. Each application is recommended to have its own partition. The application name can be used as the partition name.

6. Under the **Connections** tab, modify the connections packaged with the portlet producer application being deployed.

---

**Note:** Consider the following about producer connections:

- In the **Connections** tab, you cannot change secure properties of the connections. You can modify these connections post-deployment using Fusion Middleware Control. For information, see the chapter "Managing Services" in **Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter**.

- To edit a WSRP connection, you must also edit the associated Web service connection, which follows the naming convention `connectionname-wsconn`, for example, `myWSRPproducer-wsconn`.

- To edit PDK-Java producers, you must also edit the underlying URL connection, which follows the naming convention `connectionname-urlconn`, for example, `myPDKproducer-urlconn`.

- WSRP 1.0 standard does not support export and import capabilities. Therefore, customizations made to WSRP 1.0 producers are not migrated during deployment.

---

7. Click **Deploy**. The Deployment - Log displays the deployment status.

The message **Deployment started** displays in the Deployment - Log window. If the application is successfully deployed to the targeted server instance, the message **Deployment finished** displays in the log.

### 33.3.6 What Happens When You Deploy a Portlet Application to an Oracle WebLogic-Managed Portlet Server

If you are deploying JSR 168 portlets, the configuration settings described in Section 33.2.2, "What Happens When You Test JSR 168 Portlets on Integrated WebLogic Server" are added to the EAR file.
If you are deploying PDK-Java portlets, the configuration settings described in Section 33.2.4, "What Happens When You Test PDK-Java Portlet Applications on Integrated WebLogic Server" are added to the application EAR file at design time.

If you are deploying your portlet application in a cluster environment, you will receive the following warning:

WARNING: oracle.webcenter.lifecycle.exception.LifecycleLockedException: A lock exists that prevents the export set import.

This is expected behavior. When you deploy your application, the producer metadata exported into the EAR file (as a MAR file) needs to be imported into an MDS schema for use in the production environment. Importing the metadata occurs automatically during deployment. This import only needs to happen with one node in the cluster, so a lock is created to prevent other nodes in the cluster trying to perform the same operation.

### 33.4 Registering and Viewing Your Portlet

After you have created and deployed the producer and its portlets, you should register the producer with an application and add one or more portlets to a page to check that it is working correctly. Registering a producer gives applications the information they require to locate and communicate with that producer. After you register a producer, it is exposed as a connection, and the producer and its portlets become available in the Application Resources panel under the Connections node, or in the Resource Palette.

To register producers for your JSR 168 portlets, follow the instructions provided in Section 34.2.1, "How to Register a WSRP Portlet Producer".

To register producers for your PDK-Java portlets, follow the instructions provided in Section 34.2.3, "How to Register an Oracle PDK-Java Portlet Producer".

To add your portlets to a page, follow the instructions provided in Section 34.3, "Adding Portlets to a Page".
This chapter describes how to add portlets to the pages of your custom WebCenter application and the options that accompany this process.

This chapter includes the following sections:

- Section 34.1, "Introduction to Consuming Portlets"
- Section 34.2, "Registering Portlet Producers with a Custom WebCenter Application"
- Section 34.3, "Adding Portlets to a Page"
- Section 34.4, "Setting Attribute Values for the Portlet Tag"
- Section 34.5, "Copying Portlets"
- Section 34.6, "Deleting Portlets from Application Pages"
- Section 34.7, "Contextually Linking WSRP 2.0 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 custom WebCenter application pages. Therefore, you should familiarize yourself with the information covered in the Oracle Fusion Middleware Web User Interface Developer’s Guide for Oracle Application Development Framework before reading this chapter.

For information about creating portlets, which can then be consumed by custom WebCenter application pages, see the following chapters:

- Chapter 29, "Overview of Portlets"
- Chapter 30, "Creating Portlets with the Oracle JSF Portlet Bridge"
- Chapter 31, "Creating Portlets with the Portlet Wizard"
- Chapter 35, "Creating Portlets with OmniPortlet"
- Chapter 36, "Creating Content-Based Portlets with Web Clipping"

### 34.1 Introduction to Consuming Portlets

Oracle WebCenter Framework enables you to consume a portlet by registering its producer either with an application or with the Resource Palette from where you can add it to any application. After you register the producer, its portlets appear under the registered producer's name under the Connections node in the Application Resources panel or in the Resource Palette.

Your application can consume portlets that you build and portlets that you receive from a third party, such as a packaged-application vendor.
There are many options associated with portlet consumption. For example, you can choose to place portlets straight onto a page or nest them in an Oracle Composer component, you can adjust many attributes of the portlet tag, and you can wire portlets to each other.

### 34.2 Registering Portlet Producers with a Custom WebCenter Application

Before you can add a portlet to a custom WebCenter application page, you must register the portlet's producer with the application. You can register portlet producers in two ways:

- Register the portlet producer with a specific application. Use this option if you are not likely to want to register the producer with other applications.
- Register the portlet producer using the Resource Palette. Use this option to use the producer's portlets in multiple applications.

A portlet that is available in the Resource Palette can be added to any of your custom WebCenter 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 you have not done so. You can drag and drop a whole producer connection from the Resource Palette into the Application Resources panel of the Application Navigator. This registers the producer with the application. Alternatively, you can right-click a producer in the Resource Palette and choose **Add to Application** from the context menu to register the producer with the currently open application.

JDeveloper provides wizards for registering both WSRP producers and Oracle PDK-Java producers.

---

**Note:** If your application is source controlled, you must manually create elements in the source control system for any new files created during producer registration. Any files that are already source controlled are checked out automatically by the producer registration process.

---

For more information about producers, see Section 29.2.3, "Deployment Type." For information about obtaining prebuilt portlets through Oracle, see Section 3.9, "Using Integrated WLS." For information about using JDeveloper's portlet creation wizards, see Chapter 31, "Creating Portlets with the Portlet Wizard." For more information about portlets, see Chapter 29, "Overview of Portlets."

This section includes the following subsections:

- Section 34.2.1, "How to Register a WSRP Portlet Producer"
- Section 34.2.2, "How to Map a Producer's Declared User Categories to an Application's Defined Java EE Security Roles"
- Section 34.2.3, "How to Register an Oracle PDK-Java Portlet Producer"
- Section 34.2.4, "How to Edit Portlet Producer Registration Settings"
- Section 34.2.5, "How to Test a Portlet Producer Connection"
- Section 34.2.6, "How to Refresh a Portlet Producer"
- Section 34.2.7, "How to Delete a Portlet Producer"
34.2.1 How to Register a WSRP Portlet Producer

When you register a WSRP portlet producer, you provide basic information that describes the producer’s operational parameters. This information is used by the portlet-consuming application to communicate with the producer and with the portlets through the producer.

Oracle WebCenter 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 168 portlets. To take advantage of more advanced features of WSRP 2.0, use the Oracle-specific oracle-portlet.xml metadata extensions.

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 the PeopleSoft portlet producer, you must set the require-iframe element to true in oracle-portlet.xml, for example:

```xml
<portlet-extension>
  <portlet-name>myPortlet</portlet-name>
  <portlet-id>18</portlet-id>
  <require-iframe>true</require-iframe>
  <minimum-wsrp-version>2</minimum-wsrp-version>
</portlet-extension>
```

**Note:** If you are registering a producer provided by Oracle WebLogic Portal, make sure you register the WSRP 2.0 WSDL URL.

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 Portlet Bridge exposes JSF applications and task flows as JSR 168 portlets. For more information, see Chapter 30, “Creating Portlets with the Oracle JSF Portlet Bridge.”

**Note:** In the Register WSRP Portlet Producer wizard, if you click Cancel after you have clicked Finish, the registration is not canceled.

**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.

For other methods of invoking the wizard, such as from the Resource Palette, see Section 3.10, "Accessing Connection Wizards."
2. In the Specify Producer Name page (Figure 34–1) of the Register WSRP Portlet Producer wizard, the Create Connection in option is set depending on how you accessed the wizard. The default selection is Application Resources if you invoked the wizard from an application, and Resource Palette if you invoked the wizard from the Resource Palette. You can change this option at this point.

![Figure 34–1 The Specify Producer Name Page (WSRP Producer)](image)

3. From the Target Project dropdown list, select the project to be configured for the WSRP producer connection.

   This should be the same project as the one in which you intend to consume the portlets.

   You can change this option only if you invoked the wizard from the Application Navigator.

4. In the Producer Registration Name field, enter a name for the producer registration that is unique among all connections and then click Next.

5. In the Specify Connection Details page (Figure 34–2), in the WSDL URL field, enter the producer’s URL.

   The syntax varies according to your WSRP implementation, for example, the sample WSRP producer uses the following syntax:
   
   - `http://host:port/context-root/portlets?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.
Registering Portlet Producers with a Custom WebCenter Application

- `portletssl[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/wsre2?WSDL

  You can access the producer test page through the URL:
  
  http://host:port/context-root/info

**Figure 34–2 The Specify Connection Details Page (WSRP Producer)**

6. If the application and the producer are separated by a firewall, an HTTP proxy is needed for communication between the application and the producer. If this is the case for your application, select **Use Proxy for Contacting Producer** and specify the URL and port number of the proxy.

   **Note:** The proxy fields in this step default to the proxy preferences set in JDeveloper Preferences (from the main menu, choose **Tools > Preferences**, and then select **Web Browser and Proxy**).

7. Click **Next**.

   The connection to the producer is tested. If there are any problems, an error message displays. You must resolve any problems before you can continue.

8. In the Specify Additional Registration Details page (Figure 34–3), in the **Default Timeout Interval (Seconds)** field, enter the number of seconds to wait for the producer to respond during design time operations.

   Some producers define additional registration properties. In such cases, the properties are displayed in a table on this page of the wizard. You can enter values for these additional properties in the table. These properties are producer-specific and are used only at registration time. That is, they collect information that
consumer applications send to producers at registration time; the producers store this information against the consumers and use it subsequently.

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 34.2.2, “How to Map a Producer's Declared User Categories to an Application's Defined Java EE Security Roles.” Click No to decline this opportunity and complete the registration process.

10. In the Configure Security Attributes page (Figure 34–4), from the Token Profile dropdown list, select the type of token profile to use for authentication with the WSRP producer:

   - None—No token; no WS-Security header is attached to the SOAP message.
     If you select this option, you do not want to complete the rest of the wizard. Click Finish.

   - SAML Token with Message Integrity—SAML (Security Assertion Markup Language) Token Profile-based identity propagation with certificate based message integrity. SAML is an XML-based approach 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.

   - SAML Token with Message Protection—SAML (Security Assertion Markup Language) Token based identity propagation with certificate based message integrity and confidentiality.
- **User Name Token without Password**—A Web service consumer can supply a user name token to identify the requester by user name only to authenticate that identity to the Web service producer.

- **User Name Token with Password**—A Web service consumer can supply a user name token to identify the requester by user name and password to authenticate that identity to the Web service producer.

This token profile is used when you want to consume WSRP producers that have a different identity store. For this token profile case, ensure that you define an external application pertaining to the producer and associate this external application with the producer being registered.

**Figure 34–4 The Configure Security Attributes Page (WSRP Producer)**

11. Select the configuration type:

- **Default**—If you choose default, then all the default keystore attributes, that is location, password, keystore type, signature key and alias, encryption key and alias are picked up from the JPS (Java Platform Security) configuration. The value for recipient alias is used from the policy being used. The WebLogic Server where the application is deployed must be configured for WS-Security. For more information, see “Securing a WSRP Producer with WS-Security” in Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter.

- **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 custom WebCenter application.

When unauthenticated, the identity *anonymous* is associated with the application user. The value *anonymous* may be inappropriate for the remote producer, so you may need to specify an alternative identity here. Keep in mind though, that in this case, the custom WebCenter application has not authenticated the user so the default user you specify should be a low privileged user in the remote producer.
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. This is done by setting the <strict-authentication> flag in oracle-portlet.xml to true. For more information about this flag, see Section A.3.2, "oracle-portlet.xml."

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 37.2.3, "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 34.2.2, "How to Map a Producer’s Declared User Categories to an Application’s Defined Java EE Security Roles." Click No to decline this opportunity and complete the registration process.

16. In the Specify Key Store page, in the Store Path field, provide the full path to the keystore that contains the certificate and the private key that is used for signing some parts (security token and SOAP message body) of the SOAP message.

If you are not sure of the full path, click Browse to navigate to and select the file. The selected file should be a keystore created with the Java keytool.
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 34.2.2, "How to Map a Producer's Declared User Categories to an Application's Defined Java EE Security Roles." Click **No** to decline this opportunity and complete the registration process.

### 34.2.2 How to Map a Producer's Declared User Categories to an Application's Defined Java EE Security Roles

The user categories the producer declares come from the portlets it contains. For example, if the producer contains one or more JSR 168 portlets created with the Standards-based Java Portlet (JSR 168) 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 custom WebCenter application's `web.xml` file properties.

For more information about security roles in JSR 168 portlets, see Section 37.12, "Securing Identity Propagation Through WSRP Producers with WS-Security."

This procedure continues forward from Section 34.2.1, "How to Register a WSRP Portlet Producer."

**To map producer-declared user categories with application-defined Java EE security roles:**

1. After clicking **Finish** in the Register WSRP Portlet Producer wizard, click **Yes** in the resulting dialog.

2. In the User Categories dialog (Figure 34–6), for each **User Category**, click the corresponding field in the **J2EE Security Role** column.

The User Categories dialog is also accessible when you edit producer registration settings. For more information see Section 34.2.4, "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.

### 34.2.3 How to Register an Oracle PDK-Java Portlet Producer

When you register a PDK-Java portlet producer, you provide basic information that describes the producer's operational parameters. This information is used by the portlet-consuming application to communicate with the producer and with the portlets through the producer.

When registration is successful, the newly registered producer is displayed in JDeveloper either in the Application Resources panel of the Application Navigator, or in the Resource Palette, depending on where you created the connection. You can then select portlets from the producer for placement on your application (.jspx) page.

---

**Note:** In the Register Oracle PDK-Java Portlet Producer wizard, if you click **Cancel** after you have clicked **Finish**, the registration is not canceled.

---

**To register a PDK-Java portlet producer:**

1. In the Application Resources panel of the Application Navigator, right-click **Connections**, choose **New Connection** and then choose **Oracle PDK-Java Producer**.

   For other methods of invoking the wizard, such as from the Resource Palette, see **Section 3.10, "Accessing Connection Wizards."

2. In the Specify Producer Name page (Figure 34–7) of the Register Oracle PDK-Java Portlet Producer wizard, the **Create Connection in** option is set depending on how you accessed the wizard. The default selection is **Application Resources** if
you invoked the wizard from an application, and **Resource Palette** if you invoked the wizard from the Resource Palette. You can change this option at this point.

**Figure 34–7  The Specify Producer Name Page (PDK-Java Producer)**

3. From the **Target Project** dropdown list, select the project to be configured for the PDK-Java producer connection.

   This should be the same project as the one in which you intend to consume the portlets.

   You can change this option only if you invoked the wizard from the Application Navigator.

4. In the **Producer Registration Name** field, enter a name for the producer registration that is unique among all connections and then click **Next**.

5. In the Specify Connection Details page (Figure 34–8), in the **URL Endpoint** field, enter the producer's URL using the following syntax:

   [http://host:port/context-root/providers](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](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 37.2.3, "Managing External Applications."

   This option is available only if you invoked the wizard from the Application Navigator, as external applications are scoped to individual applications.

9. Select **Enable Producer Sessions** to enable sessions between the producer and the consuming application.
When sessions are enabled, the server maintains session-specific information, such as user name. Message authentication uses sessions, so if the shared key is set, then this option should also be selected.

For sessionless communication between the producer and the server, deselect this option.

10. At this point in the wizard, you can click Finish to complete the registration, using the default values for all remaining steps.

To provide additional details, click Next and follow the remaining steps.

11. In the Specify Additional Details page (Figure 34–9), 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 for Subscriber ID is also passed every time a portlet call is made. If the producer does not see the expected value for Subscriber ID, then it might reject the registration call.

**Note:** Editing the producer registration to change the Subscriber ID after the initial registration has no effect. To specify a different Subscriber ID, you must reregister the producer.

13. In the Shared Key field, enter a shared key to use for producers that are set up to handle encryption and then click Finish.

The shared key is used by the encryption algorithm to generate a message signature for message authentication. Producer registration fails if the producer is
Registering Portlet Producers with a Custom WebCenter Application

34.2.4 How to Edit Portlet Producer Registration Settings

Both the WSRP and PDK-Java portlet producer registration wizards enable you to access and revise many of the values you entered when you registered the producer.

To edit portlet producer registration settings:
1. Navigate to the producer in the Application Resources panel of the Application Navigator or in the Resource Palette.
2. Right-click the producer to edit, and choose Properties.
3. Edit the producer registration properties as required, clicking Next to step through the pages of the wizard.
   You can also go directly to a specific step by clicking the links in the navigation panel on the left side of the wizard.
   - For information about WSRP Producer settings, see Section 34.2.1, "How to Register a WSRP Portlet Producer."
   - For information about Oracle PDK-Java Producer settings, see Section 34.2.3, "How to Register an Oracle PDK-Java Portlet Producer."

   You cannot edit the Producer Registration Name.
4. When you have changed all the necessary settings, click Finish.
5. Editing some properties, such as the WSDL URL or URL Endpoint, requires a producer refresh. When you make such a change, a dialog displays allowing you to refresh the producer immediately, save the changes but do not refresh the producer, or return to the edit producer registration wizard. Click Yes, No, or Cancel as appropriate.
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 34.2.5, "How to Test a Portlet Producer Connection."

34.2.5 How to Test a Portlet Producer Connection

The connection testing feature provides a means of testing the validity of a portlet producer connection.

To test a portlet producer connection:
1. Navigate to the producer in the Application Resources panel of the Application Navigator or in the Resource Palette.
2. Right-click the producer to test, and choose Test Producer Connection.
3. A progress bar appears while the test is underway. A success or failure dialog displays when the test is complete. Click **OK** to close this dialog.

If the failure dialog displays, consider editing the producer registration details and retesting the producer connection. Additionally, especially if the failure dialog takes a long time to display, ensure that the producer is available. For example, if the producer is provided through the Integrated WLS, ensure that the Integrated WLS is running, and then retest the connection.

### 34.2.6 How to Refresh a Portlet Producer

When you refresh a portlet producer, the portlets from that producer are also refreshed. Newly added portlets and any updates to existing portlets become available to any applications that are consuming portlets from this producer.

**Tip:** When a portlet is removed from a producer, be sure to manually delete the portlet from all application pages on which it has been placed. For more information, see Section 34.6, "Deleting Portlets from Application Pages."

---

**To refresh a portlet producer:**

1. Navigate to the producer in the Application Resources panel of the Application Navigator or in the Resource Palette.
2. Right-click the producer to refresh, and choose **Refresh Producer Registration.**
3. In the Portlet Producer Refresh dialog, click **Yes.**

### 34.2.7 How to Delete a Portlet Producer

If you no longer want to use a particular producer with your application, you can delete the producer. For information about deleting portlets and relevant page variables, see Section 34.6, "Deleting Portlets from Application Pages."

**Note:** if you delete a producer, any pages that consume portlets from that producer display an error message that the portlet is unavailable. The portlets continue to be unavailable even if you re-register the producer using the same name.

---

**To delete a portlet producer:**

1. Navigate to the producer in the Application Resources panel of the Application Navigator or in the Resource Palette.
2. Right-click the producer to delete, and choose **Delete.**
3. In the Delete Confirmation dialog, click **Yes.**

### 34.3 Adding Portlets to a Page

Placing a portlet on a custom WebCenter application page is a simple matter of dragging the portlet from the Application Resources panel or Resource Palette and dropping it on the page.

This section includes the following subsections:

- Section 34.3.1, "How to Add a Portlet to a Page"
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 custom WebCenter application. For more information, see Section 3.2, "Creating a WebCenter Application."
2. Creating an application page. For more information, see Section 3.3, "Creating WebCenter Application-Enabled Pages."
3. Registering the portlet's producer with the application. For more information, see Section 34.2, "Registering Portlet Producers with a Custom WebCenter Application."
4. Some of the portlets you plan to consume may come from applications that handle their own authentication. In such cases, you must register the application as an external application and identify it to the portlet producer that provides it. For more information, see Chapter 37, "Securing Your WebCenter 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 37.9, "Registering Custom Certificates with the Keystore."

34.3.1 How to Add a Portlet to a Page

You can add a portlet to a page by dragging and dropping it from the Application Resources panel of the Application Navigator or from the Resource Palette.

To add a portlet to a page:

1. In the Application Navigator, open the application that contains the page (\.jspx file) to which you want to add the portlet.
2. Expand the project that contains the page.
3. Locate the page, right-click it, and then choose Open.
   
   **Tip:** You can also double-click the page to open it.

4. Under the Connections node in the Application Resources panel of the Application Navigator, or in the Resource Palette:
   
   - If the producer is a WSRP producer, expand the WSRP Producer node.
   - If the producer is a PDK-Java producer, expand the Oracle PDK-Java Producer node.

5. Expand the node for the portlet producer that contains the portlet to add to the page.

Under the selected producer, all portlets contained by that producer are listed.
6. Drag the portlet from the producer node directly onto the page.

You should drag the portlet onto a form on the page. If you do not, a dialog displays prompting you to create a form to contain the portlet. Select:

- **ADF Faces - Form** if the page contains rich client components. This adds an `af:form` component to the page.
- **JSF HTML - Form** if the page contains HTML components (for example, if it is an upgraded 10.1.3.2 page). This adds an `h:form` or `tr:form` component to the page, depending on the surrounding document tag.

Do not select **HTML - Form** as this is not valid for portlets.

### 34.3.2 What Happens When You Add a Portlet to a Page

When you add a portlet to a page, a portlet tag (`adfp:portlet` or `adfph:portlet`) is added to the page source. This is the tag that represents the portlet component. This tag includes attributes that you can edit using the Property Inspector, or in the page source, to further control the behavior and appearance of the portlet. For information about these attributes, see Section 34.4, "Setting Attribute Values for the Portlet Tag."

The type of portlet tag used is determined by the following:

- If the project is configured for rich client components alone, the `adfp:portlet` tag is used.
- If the project is configured for Trinidad components alone, the `adfph:portlet` tag is used.
- If the project is configured for both rich client and Trinidad components, a dialog displays where you can choose which portlet tag to use.
- If the project is not configured for rich client or Trinidad components, the `adfp:portlet` tag is used and the project is automatically configured for rich client components.

This is so that the look and feel of the portlet matches that of other components on the page. For example, if you created your page as described in Section 3.3, "Creating WebCenter Application-Enabled Pages," the page is a rich client page. In this case, the portlet is added using the `adfp:portlet` tag.

#### Example 34–1  A Rich Client Page Containing a Portlet

```xml
<?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"
                    partialTriggers="com1"/>
            </af:form>
        </af:document>
    </f:view>
</jsp:root>
```
If you are working with an upgraded 10.1.3.2 application or an application that contains Trinidad components, the application uses HTML components, rather than rich client components. In this case, when you drag a portlet onto a page, the adfph:portlet tag is used.

**Example 34–2  A Trinidad Page Containing a Portlet**

```xml
<?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 Oracle 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 34–3:

**Example 34–3  Hierarchical Placement of the Portlet Tag**

```xml
af:document
    af:form
        cust:panelCustomizable
            adfp:portlet
```

**Note:** We recommend that you do not mix ADF Faces rich client components with HTML or Trinidad components on the same page. Doing so may produce unexpected results at runtime. Therefore, do not place a rich client portlet inside an Oracle Composer HTML component or an HTML portlet inside an Oracle Composer rich client component.

For information about Composer tags, see Chapter 7, "Enabling Runtime Editing of Pages Using Oracle Composer."

**Note:** When you drop an instance of OmniPortlet onto your page, open the Property Inspector and ensure that the AllModesSharedScreen, under the Display Mode category, is set to false, the default value. Setting this property to true may prevent you from editing certain sections of your OmniPortlet in the OmniPortlet wizard.
34.3.3 What Happens at Runtime

Once you place a portlet on a page, right-click the page and choose Run. This displays the page and runs the portlet in your default browser using JDeveloper's Integrated WLS. Different portlets may require additional runtime configuration. Notably, the content of an OmniPortlet or Web Clipping portlet instance is defined at runtime. For more information about OmniPortlet, see Chapter 35, "Creating Portlets with OmniPortlet." For more information about the Web Clipping portlet, see Chapter 36, "Creating Content-Based Portlets with Web Clipping." For more information about portlets generally, see Chapter 29, "Overview of Portlets."

When running a portlet that has an Edit mode (in a custom WebCenter application, this renders as a Personalize command on the portlet’s Actions menu), the Personalize option displays in the portlet’s Actions menu 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 37.8, "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.adf.ftinternal.view.faces.portlet.PortletDefinitionDTFactory"/>
```

This entry is required to enable consumption of portlets at runtime.

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 37.2, "Working with External Applications."

34.4 Setting Attribute Values for the Portlet Tag

In the source code view of a page, each portlet is represented by an adfp:portlet tag (or adfph:portlet tag), which includes a set of required and optional attributes. Required attributes, value and portletType, are provided automatically by the framework, and must not be altered. Optional attribute values are relevant when support for the attribute is built into the portlet. For example, you can set isAboutModeAvailable to true, but if no About mode has been defined for the portlet, then the attribute setting does not affect the portlet.

Portlets also support a set of style-related attributes, which are discussed more fully in Section 9.9, "Applying Styles to Components."
The portlet tag uses many attributes, which you can set at design time either through the JDeveloper Property Inspector or in the source code as attributes of the tag.

This section includes the following subsections:

- Section 34.4.1, "How to Set Attribute Values for the Portlet Tag Using the Property Inspector"
- Section 34.4.2, "How to Set Attribute Values for the Portlet Tag in Source Code"
- Section 34.4.3, "Common Attributes of the Portlet Tag"
- Section 34.4.4, "Appearance Attributes of the Portlet Tag"
- Section 34.4.5, "Behavior Attributes of the Portlet Tag"
- Section 34.4.6, "Portlet Modes Attributes of the Portlet Tag"
- Section 34.4.7, "Style Attributes of the Portlet Tag"
- Section 34.4.8, "Binding Attributes of the Portlet Tag"
- Section 34.4.9, "Customization Attributes of the Portlet Tag"
- Section 34.4.10, "Other Attributes of the Portlet Tag"
- Section 34.4.11, "What You May Need to Know About Maximize, Minimize, Restore, and Move"
- Section 34.4.12, "What You May Need to Know About IFRAMEs"

### 34.4.1 How to Set Attribute Values for the Portlet Tag Using the Property Inspector

The Property Inspector provides a quick and easy way to set attribute values for the portlet tag without having to edit the source code yourself.

**To set attribute values for the portlet tag using the Property Inspector:**

1. In the Application Navigator, open the application that contains the page on which the portlet appears.
2. Expand the project that contains the page.
3. Locate the page, right-click it, and then choose Open.
   
   **Tip:** You can also double-click the page to open it.
4. In the design view, select the portlet whose attributes you want to set.
5. In the Property Inspector, click the appropriate tab and set the desired attribute.
   
   Repeat this step as often as required.
6. Save your changes.
7. To test the changes you have made, right-click the page and choose Run.

### 34.4.2 How to Set Attribute Values for the Portlet Tag in Source Code

If you prefer working in source code, you can set attribute values for the portlet tag directly there.

**To set attribute values for the portlet tag in source code:**

1. In the Application Navigator, open the application that contains the page on which the portlet appears.
2. Expand the project that contains the page.

3. Locate the page, right-click it, and then choose **Open**.

   **Tip:** You can also double-click the page to open it.

4. In the design view, select the portlet whose attributes you want to set.

5. Click the **Source** tab. The portlet that you selected is highlighted in the source code.

6. Make your changes directly to the source code. **Example 34–4** shows an edited portlet tag.

**Example 34–4**  **An Edited Portlet Tag**

```xml
<adfp:portlet value="#{bindings.portlet1}"
    portletType="/oracle/adf/portlet/WsrpPortletProducer1/
    applicationPortlets/E0default_b452f828_010a_1000_8002_82235f57ea8"/
    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**.

### 34.4.3 Common Attributes of the Portlet Tag

**Table 34–1** describes the common attributes of the portlet tag.
### Table 34–1 Common Attributes of the Portlet Tag

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
</table>
| id        | Text string. For example: id="newsBrief" | The unique identifier of the portlet. This attribute is populated with a unique value by default when you add the portlet to a page. The value must follow a subset of the syntax allowed in HTML:  
  - Must not be a zero-length String.  
  - First character must be an ASCII letter (A-Z a-z) or an underscore (_).  
  - Subsequent characters must be ASCII letter or digits (a-Z a-z 0-9), underscores (_), or dashes (-). |
| title     | Text string. For example: title="Announcements" | The portlet title, which is displayed in the portlet header. The value specified here takes precedence over any title specified elsewhere (for example, in the portlet markup). If no value is specified here, the portlet extracts its title from the portlet markup (response). If no value is specified either here or in the portlet markup, the portlet extracts its title from the portlet definition. Note: Supplying a value to the title attribute at design time means that any change made to the title at runtime in Edit or Edit Defaults mode is ignored. |
| width     | Number expressed in pixels or as a percentage of available area:  
  - For pixels, enter npx, for example: width = 300px  
  - For percentage, enter n%, for example: width = 50% | The width of the area to allow for portlet display. If the actual portlet width is larger than the width value entered here, a scrollbar appears, provided displayScrollBar is set to auto or true. If displayScrollBar is set to false, and the actual portlet width exceeds the value expressed for the width attribute, the width attribute value is considered and the portlet content is truncated. |
Table 34–1  (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:</td>
<td>The height of the area to allow for portlet display.</td>
</tr>
<tr>
<td></td>
<td>height = 300px</td>
<td>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:</td>
<td>A URI specifying the location of an image to use as an icon, displayed to the left of the portlet title in the portlet header. You can use the icon to indicate the portlet's purpose, to reinforce branding, as a content indicator, or for some other reason.</td>
</tr>
<tr>
<td></td>
<td>icon=&quot;coffee.png&quot;</td>
<td>In the Property Inspector, click the Property Menu icon next to the field and then choose Edit to locate and select the required image. The value must be an absolute URI or a URI that is resolvable relative to the current page or the application context root. The URI provided in the preceding example is stored at the application context root, therefore a full path is not required.</td>
</tr>
<tr>
<td>partialTriggers</td>
<td>One or more component IDs. For example:</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>
<tr>
<td></td>
<td>partialTriggers=&quot;_id1 _id2 componentID5&quot;</td>
<td>Separate component IDs with spaces.</td>
</tr>
</tbody>
</table>

34.4.4 Appearance Attributes of the Portlet Tag

Table 34–2 describes the appearance attributes of the portlet tag.
### Table 34–2  Appearance Attributes of the Portlet Tag

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
</table>
| expansionMode         | minimized, normal | The default state of the portlet:  
  - minimized—The portlet's default display mode is collapsed (minimized).  
  - normal—The portlet's default display mode is neither collapsed nor expanded to the width of the page. |
| allModesSharedScreen  | auto, false, true | Whether a change in portlet mode renders the new mode on a new page, rather than the page on which the portlet resides:  
  - auto—All portlet modes are displayed inline if the remote portlet is configured, through its oracle-portlet.xml, to require an inline frame (IFRAME). This ensures that Oracle Bridge portlets are displayed inline.  
  - false—All portlet modes, except View (JSR 168) or Show (PDK-Java), are rendered each on their own page. This is useful for portlets such as OmniPortlet and the Web Clipping portlet, which require that modes other than Show mode display on pages other than the page on which the portlet resides.  
  - true—All portlet modes are displayed inline. One mode is swapped out for another on the same page. In other words, this attribute enables all portlet modes to display without leaving the context of a given page. |
| renderPortletInIFrame | auto, false, true | Whether the portlet is rendered in an IFRAME:  
  - auto—The portlet is rendered in an IFRAME if:  
    - parsing of the content fails  
    - a form in the content contains an input of type file  
    - the remote portlet is configured, through its oracle-portlet.xml, to require an IFRAME. This ensures that Oracle JSF Portlet Bridge portlets are displayed in an IFRAME.  
    For more information, see Section 34.4.12, "What You May Need to Know About IFRAMES."  
  - false—The portlet is rendered inline. HTML markup from a portlet that is not rendered in an IFRAME may interfere with other components on the Oracle ADF page.  
  - true—The portlet is rendered in an IFRAME. |
### Table 34–2 (Cont.) Appearance Attributes of the Portlet Tag

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<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 height specified.</td>
</tr>
<tr>
<td></td>
<td>true</td>
<td>- false—Never render a scroll bar. If the portlet content does not fit the height and width specified, the portlet renders in its actual size.</td>
</tr>
<tr>
<td></td>
<td>Default:</td>
<td>- true—Always render a scroll bar.</td>
</tr>
<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 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:</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>Default:</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></td>
<td>Default:</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>background</td>
<td>dark</td>
<td>The style selector to apply to the skin used by the portlet:</td>
</tr>
<tr>
<td></td>
<td>light</td>
<td>- dark—Apply the dark style selector to the skin.</td>
</tr>
<tr>
<td></td>
<td>medium</td>
<td>- light—Apply the light style selector to the skin.</td>
</tr>
<tr>
<td></td>
<td>Default:</td>
<td>- medium—Apply the medium style selector to the skin.</td>
</tr>
<tr>
<td></td>
<td></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>Text string. For example: shortDesc=&quot;Portlet for entering display text in place.&quot;</td>
<td>A short description of the portlet.</td>
</tr>
</tbody>
</table>
### Table 34–2 (Cont.) Appearance Attributes of the Portlet Tag

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<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>
<tr>
<td></td>
<td>Default: always</td>
<td></td>
</tr>
<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></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></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></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></td>
</tr>
</tbody>
</table>

*Note:* This attribute is available only for the adfp:portlet tag and not for the adfph:portlet tag.
### 34.4.5 Behavior Attributes of the Portlet Tag

Table 34–3 describes the behavior attributes of the portlet tag.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>partialTriggers</td>
<td>One or more component IDs. For</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>
<tr>
<td></td>
<td>example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>partialTriggers=&quot;_id1 _id2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>componentID5&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Separate component IDs with</td>
<td></td>
</tr>
<tr>
<td></td>
<td>spaces.</td>
<td></td>
</tr>
<tr>
<td>submitUrlParamters</td>
<td>false</td>
<td>Whether parameters in portlet links that point to the page on which the portlet is placed are made available to the page:</td>
</tr>
<tr>
<td></td>
<td>true</td>
<td>■ false—Parameters are not made available to the page. Rather, they are available only inside the portlet initiating the request.</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
<td>■ true—Parameters available on the container page.</td>
</tr>
</tbody>
</table>

### 34.4.6 Portlet Modes Attributes of the Portlet Tag

Portlet Modes attributes control the rendering of mode-switching UI actions, such as entering edit mode. The ability to render a portlet in a particular mode depends on the modes supported by the portlet and the user authorization. For example, if the isCustomizeModeAvailable attribute is set to true, but the action is not supported in the portlet, then the attribute setting does not affect the portlet.

Portlet Modes attributes, described in Table 34–4, are value binding expressions that evaluate to true or false:

- true means the portlet is allowed to render in the named mode.
- false means the portlet is not allowed to render in the named mode.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>isAboutModeAvailable</td>
<td>false</td>
<td>Whether to render an About command on the portlet’s Actions menu.</td>
</tr>
<tr>
<td></td>
<td>true</td>
<td>Users choose About to invoke the portlet’s About mode.</td>
</tr>
<tr>
<td></td>
<td>Default: true</td>
<td></td>
</tr>
<tr>
<td>isConfigModeAvailable</td>
<td>false</td>
<td>Whether to render a Configure command on a JSR 168 portlet’s Actions menu.</td>
</tr>
<tr>
<td></td>
<td>true</td>
<td>Users choose Configure to open the portlet’s Configuration settings.</td>
</tr>
<tr>
<td></td>
<td>Default: true</td>
<td></td>
</tr>
<tr>
<td>isCustomizeModeAvailable</td>
<td>false</td>
<td>Whether to render a Customize command on the portlet’s Actions menu.</td>
</tr>
<tr>
<td></td>
<td>true</td>
<td>Site administrators choose Customize to edit a portlet’s default personalization data.</td>
</tr>
<tr>
<td></td>
<td>Default: true</td>
<td></td>
</tr>
</tbody>
</table>
Table 34–4  (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>isDetailModeAvailable</td>
<td>false</td>
<td>Whether to render a Details command on a PDK-Java portlet's Actions menu.</td>
</tr>
<tr>
<td></td>
<td>true</td>
<td>Users choose Details to open the portlet in Full Screen mode.</td>
</tr>
<tr>
<td></td>
<td>Default: true</td>
<td></td>
</tr>
<tr>
<td>isHelpModeAvailable</td>
<td>false</td>
<td>Whether to render a Help command on the portlet's Actions menu.</td>
</tr>
<tr>
<td></td>
<td>true</td>
<td>Users choose Help 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 Print command on a JSR 168 portlet's Actions menu.</td>
</tr>
<tr>
<td></td>
<td>true</td>
<td>Users choose Print to displays 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 Refresh command on the portlet's Actions menu.</td>
</tr>
<tr>
<td></td>
<td>true</td>
<td>Users choose Refresh 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 Personalize command on the portlet's Actions menu.</td>
</tr>
</tbody>
</table>
|                            | true    | Users choose Personalize to alter their personal view of the portlet. This mode is equivalent to the Edit mode selection in the Standards-based Java Portlet (JSR168) Wizard. The Personalize command displays on the Actions menu only to authenticated users (that is, users who are logged in). It does not display to public or unauthenticated users. You must implement some form of application security for users to be able to personalize their portlet views. If you are a developer creating portlets, and you want to test the Personalize mode without creating a complete security model for your application, then see Section 37.8, "Configuring Basic Authentication for Testing Portlet Personalization."
|                            | Default: true |                                                                                                                                          |
| isPreviewModeAvailable     | false   | Whether to enable previewing of portlet content.                                                                                             |
|                            | true    | This mode has no particular application in custom WebCenter 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. |
|                            | Default: false |                                                                                                                                          |

34.4.7 Style Attributes of the Portlet Tag

Table 34–5 describes the style attributes of the portlet tag.
### 34.4.8 Binding Attributes of the Portlet Tag

Table 34–6 describes the 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 example: binding=&quot;#{frameActionsBean.Binding}&quot;</td>
<td>A binding reference to store the component instance. The binding reference binds an instance of the portlet to a managed bean property. Managed beans are any JavaBeans used by the application that are registered in the JSF faces-config.xml file.</td>
</tr>
</tbody>
</table>

### 34.4.9 Customization Attributes of the Portlet Tag

Table 34–7 describes the 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, true</td>
<td>Whether design time customizations of the portlet tag are allowed on this portlet.</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 customizations, but restrict who can actually perform them.</td>
</tr>
</tbody>
</table>

### 34.4.10 Other Attributes of the Portlet Tag

Table 34–8 describes the other attributes of the portlet tag.
34.4.11 What You May Need to Know About Maximize, Minimize, Restore, and Move

To accommodate the needs of the development environment, the behavior of the actions Minimize, Maximize, Restore, and Move for Show Detail Frame and portlet components differs between design time and runtime. At design time, these actions persist in a given WLS session, but do not persist over sessions (session means the time between starting and stopping WLS). At runtime, these actions persist both during a given WLS session and across sessions.

This difference has been introduced to enable an automatic reset of an application page at design time.

If persisting across sessions is not required at runtime, then a simple modification to the application’s web.xml file can turn it off. Go to the following parameter setting in the application’s web.xml file (Example 34–5):

Example 34–5 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 34–6):

Example 34–6 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, Maximize, Restore, and Move actions display only to users with Customize privileges. They do

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>iframeDtd</td>
<td>loose</td>
<td>Which DTD, if any, is specified in the doctype declaration that is created when portlet content is rendered inside a IFRAME:</td>
</tr>
<tr>
<td></td>
<td>none</td>
<td>■ none—No DTD is specified. This relaxes the restrictions on the HTML content being technically conformant HTML. Browsers usually handle such HTML acceptably, however, because some CSS style sheet from the ADF Faces page consuming the portlet is also imported into the IFRAME document, for that style sheet to work correctly, it may be necessary to declare the content conformant to the loose or strict DTDs.</td>
</tr>
<tr>
<td></td>
<td>strict</td>
<td>■ loose—The DTD <a href="http://www.w3.org/TR/html/loose.dtd">http://www.w3.org/TR/html/loose.dtd</a> is used.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ strict—The DTD <a href="http://www.w3.org/TR/html/strict.dtd">http://www.w3.org/TR/html/strict.dtd</a> is used.</td>
</tr>
<tr>
<td>iframeDtd</td>
<td></td>
<td>Default: loose</td>
</tr>
</tbody>
</table>

Table 34–8 Other Attributes of the Portlet Tag
not display to users with Personalize privileges. Customize users can test the effect of these actions by following these steps at design time:

1. Run the application page using JDeveloper’s Integrated WLS.
2. Log in as the administrator.
3. Maximize a portlet. Move portlets around. Make whatever changes you want using the relevant actions commands.
4. Log out, then log in as a user and check the effects of your actions.

### 34.4.12 What You May Need to Know About IFRAMEs

Placing a portlet inline on a page provides a better user experience as compared to placing it in an inline frame, or IFRAME. However, at times, it may be required to include the portlet markup inside an IFRAME. Setting the `renderPortletInIFrame` attribute to `auto` causes Oracle WebCenter Framework to render the portlet in an IFRAME if:

- The portlet is a JSF Portlet.
  
  When you add a JSF Portlet to your page, `renderPortletInIFrame` is set to `true` by default as ADF pages within portlets are too complex to render inline due to Javascript issues.
- The portlet contains a file upload element.
- The parser throws an exception as it is not able to parse the markup.
- You have specified that the portlet must be rendered in an IFRAME.

To ensure that the portlet is always included in an IFRAME, open the `oracle-portlet.xml` file and in the `<portlet>` section, set the `<prefer-iframe>` element to `true`.

---

**Note:** If you render a portlet within an IFRAME, then manipulating `window.location` may give unexpected results. If your portlet uses `window.location`, then you should ensure that your JavaScript is robust enough to handle the case where the portlet renders itself inside of an IFRAME.

---

Using IFRAMEs also has accessibility implications. For more information, see Section 1.3, "Accessibility Features."

### 34.5 Copying Portlets

When you copy portlets, the portlets and their copies must reside within the same application. For example, you can copy a portlet from one page in an application to another page in the same project in that application, or from one place on a page to another place on the same page. You can also copy a portlet from one project to another project in the same application, if the target project is configured for consuming portlets. The copies are references to the same portlet instance, so customizations or personalizations made to any instance of the portlet (original or copy) affect all the other instances.

Copying a portlet is more than a matter of copying and pasting the portlet view tag. It involves copying portlet-related entries from the application page's source. It may also involve copying portlet-related entries from the page definition file and removing
duplicate portlet binding information or creating a new method in the copied portlet’s binding bean.

When a portlet is copied, the target page must be an Oracle ADF Faces page. Any preexisting code on the target page must reflect that. This is quite easy to accomplish. When JDeveloper creates a new JSF page, it contains pure JSF tags. The first time you drop an Oracle ADF Faces component onto the page, tags are automatically updated to be Oracle ADF Faces tags. For example, an <html> tag becomes <afh:html>, <head> and <title="title"> tags become <afh:head title="title">, and so on. Therefore, a simple way to ensure the conversion of the target page to an Oracle ADF Faces page is to place any Oracle ADF Faces component on the target page. This performs any required code conversion for you automatically.

This section includes the following subsections:
- Section 34.5.1, “How to Copy a Portlet and Place it on the Same Page”
- Section 34.5.2, “How to Copy a Portlet from One Application Page to Another”

34.5.1 How to Copy a Portlet and Place it on the Same Page

Because all of the page’s resources are available to both portlet instances when you copy a portlet to the same page, there is no requirement to copy portlet-related information from the page’s Page Definition file. It is just a matter of copying and pasting the portlet’s view tag, and assigning a unique identifier to the copy.

To copy and place a portlet on the same page:
1. In the Application Navigator, open the application that contains the page on which the portlet to copy appears.
2. Expand the project that contains the page.
3. Locate the page, right-click it, and then choose Open.
   Tip: You can also double-click the page to open it.
4. In the design view, select the portlet to copy.
5. Click the Source tab. The portlet that you selected is highlighted in the source code.
6. Copy the portlet tag (Example 34–7).

Example 34–7 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: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_82235fcd4e2b" binding="#{backing_portlet_page.portlet1}" id="portlet1" isCustomModesAvailable="true"/>
            </h:form>
        </afh:body>
    </afh:html>
</f:view>
```
7. Paste the copied code fragment into the desired location of the page source.

8. Provide a unique value for the copy's `id` attribute (Example 34–8).

**Example 34–8 Changing the Portlet ID**

```xml
<afp:portlet value="#{bindings.portlet1}" portletType="/oracle/adf/portlet/pdksampleproducer_1153245807295/applicationPortlets/Portlet2_82d49b79_010c_1000_8006_82235ff4e2b" 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 `afp:portlet` tag has a binding attribute, for example:

   binding="#{backing_untitled2.portlet1}"

Then either remove this binding, or create a new method in the binding bean by opening the managed bean class for this managed bean and defining the new method.

For example, if portlet1 is copied, the pasted copy becomes portlet2 in the managed bean class, as shown in Example 34–9.

**Example 34–9 Creating a New Method for a Managed Bean in the Managed Bean Class**

```java
private PortletBase portlet2;
public void setPortlet2(PortletBase portlet2) {
    this.portlet2 = portlet2;
}

public PortletBase getPortlet2() {
    return portlet2;
}
```

10. From the main menu, choose File > Save All.

### 34.5.2 How to Copy a Portlet from One Application Page to Another

When you copy a portlet from one page to another in an application, portlet-related code must also be copied from the source page’s Page Definition file. This section describes the steps related to both copying from one application page to another and from one application project to another.

**To copy a portlet from one application 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 34–10).

   **Example 34–10  Source Page Code Fragment to Be Copied When Copying a Portlet**

   `<f:view>
   <afh:html binding="#{backing_portlet_page.html1}" id="html1">
     <afh:head title="portlet_page" binding="#{backing_portlet_page.head1}" id="head1">
       <meta http-equiv="Content-Type" content="text/html;charset=windows-1252"/>
     </afh:head>
     <afh:body binding="#{backing_portlet_page.body1}" id="body1">
       <h:form binding="#{backing_portlet_page.form1}" id="form1">
         <adfp:portlet value="#{bindings.portlet1}" portletType="/oracle/adf/portlet/pdksampleproducer_1153245807295/applicationPortlets/Portlet2_82d49b79_010c_1000_8006_82235ffc4e2b" binding="#{backing_portlet_page.portlet1}" id="portlet1" isCustomModesAvailable="true"/>
       </h:form>
     </afh:body>
   </afh:html>
   </f:view>

7. Go to the application page to which to copy the portlet (the target page).

   Portlets can reside only on Oracle ADF Faces pages. If the target page does not contain Oracle ADF Faces components, then ensure that the container objects—that is, any tags the portlet tag is nested in—use Oracle ADF tags.

   If you are copying the portlet to a page in a different project, the target project must be configured for consuming portlets. To configure the project, you must register a portlet producer with the project. For information, see Section 34.2, "Registering Portlet Producers with a Custom WebCenter 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 34–11).

   **Example 34–11  Code Fragment to Be Copied From a Page Definition File**

   `<portlet id='portlet1' portletInstance="/oracle/adf/portlet/pdksampleproducer_1153245/applicationPortlets/Portlet2_82d49_010c_1000_8006_82235" class='oracle.adf.model.portlet.binding.PortletBinding'`
Deleting Portlets from Application Pages

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.

34.6 Deleting Portlets from Application Pages

When you delete a portlet from an application page, if the portlet had parameters, then you should also delete page variables associated with those parameters from the application page's Page Definition file.

To delete a portlet and its related page variables from a page:
1. In the Application Navigator, open the application that contains the page on which the portlet to delete appears.
2. Expand the project that contains the page.
3. Locate the page, right-click it, and then choose Open.
   Tip: You can also double-click the page to open it.
4. In the design view, right-click the portlet to delete and choose Delete.
   This deletes the portlet from the page and the portlet binding from the Page Definition file.
5. If the portlet included variables, in the Application Navigator, right-click the page and choose Go to Page Definition.
6. Click the Source tab and locate the page variables associated with the deleted portlet, and delete them from the page definition file.
   For example, if you deleted portlet1, you would also delete the highlighted variables in Example 34–12:

**Example 34–12 Deleting Portlet-Related Page Variables from a Page Definition File**

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<pageDefinition xmlns="http://xmlns.oracle.com/adfm/uimodel"
    version="10.1.3.38.97" id="untitled1PageDef"
    Package="project1.pageDefs"/>
```
Contextually Linking WSRP 2.0 Portlets

One way to make your custom WebCenter 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. Table 34–9 summarizes the types of components you may tie with this type of contextual behavior.

### Table 34–9 Components You Can Synchronize

<table>
<thead>
<tr>
<th>Component Initiating Parameter</th>
<th>Can Initiate Partial Page Refresh</th>
<th>Component Reading Parameter Value</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle ADF Faces components</td>
<td>Yes</td>
<td>Oracle ADF Faces components</td>
<td>For more information about linking Faces components, see <em>Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework.</em></td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td>WSRP portlets</td>
<td>You can use Faces components, such as options or lists, to change the displayed content of portlets.</td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td>PDK-Java portlets</td>
<td>You can use Faces components, such as options or lists, to change the displayed content of portlets.</td>
</tr>
<tr>
<td>WSRP 2.0 portlets&lt;sup&gt;1&lt;/sup&gt;</td>
<td>Yes</td>
<td>Oracle ADF Faces components</td>
<td>Standards-based portlets can pass parameters to Faces components.</td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td>WSRP 2.0 portlets</td>
<td>You can pass parameters between standards-based portlets.</td>
</tr>
</tbody>
</table>

7. From the main menu, choose, select **File > Save All**.

34.7 Contextually Linking WSRP 2.0 Portlets

### Table 34–9 Components You Can Synchronize

<table>
<thead>
<tr>
<th>Component Initiating Parameter</th>
<th>Can Initiate Partial Page Refresh</th>
<th>Component Reading Parameter Value</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle ADF Faces components</td>
<td>Yes</td>
<td>Oracle ADF Faces components</td>
<td>For more information about linking Faces components, see <em>Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework.</em></td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td>WSRP portlets</td>
<td>You can use Faces components, such as options or lists, to change the displayed content of portlets.</td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td>PDK-Java portlets</td>
<td>You can use Faces components, such as options or lists, to change the displayed content of portlets.</td>
</tr>
<tr>
<td>WSRP 2.0 portlets&lt;sup&gt;1&lt;/sup&gt;</td>
<td>Yes</td>
<td>Oracle ADF Faces components</td>
<td>Standards-based portlets can pass parameters to Faces components.</td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td>WSRP 2.0 portlets</td>
<td>You can pass parameters between standards-based portlets.</td>
</tr>
</tbody>
</table>
WSRP 2.0 introduced navigational parameters, which can be used with page variables to link portlets on the page. When your goal is to link two components where at least one is a WSRP 2.0 portlet, you can use navigational parameters alone.

With navigational parameters, you publish to page variables, and individual components can subscribe to those page variables and reflect the changes made to those values. When you want to link more than two components, such as a task flow to two or more WSRP 2.0 portlets, you can use ADFm events instead of navigational parameters. The ADFm model enables one component to accept a value, process it, and then raise an event that another component can then process. This can continue until all events in the page are raised and consumed or discarded.

Note: An example of how to use ADFm events and navigation parameters with JSF portlets is provided in Section 30.3, "Creating JSF Portlets: Example."

This section includes the following subsections:

- Section 34.7.1, "How to Link Portlets With Navigational Parameters"
- Section 34.7.2, "What Happens at Runtime"
- Section 34.7.3, "How To Associate Navigational Parameters with ADFm Events"
- Section 34.7.4, "What Happens at Runtime"
- Section 34.7.5, "How To Cascade Events Across Multiple Components"
Contextually Linking WSRP 2.0 Portlets

■ Section 34.7.6, "What Happens at Runtime"

Before You Begin
The following examples show how to link WSRP portlets in various ways. This is illustrated using the Parameter Form Portlet and Parameter Display Portlet provided by the sample WSRP producer. The Parameter Form Portlet has three output parameters that get set when values are submitted in the form inside the portlet. The Parameter Display Portlet has three input parameters, the values of which are displayed in the portlet when they are set.

Before you begin working through the examples, you must create an application and register the sample WSRP producer.

1. Start JDeveloper.
2. From the main menu, choose File > New.
3. In the New Gallery, expand General, select Applications and then WebCenter Application, and click OK.
4. In the Name your application page of the Create WebCenter Application wizard, in the Application Name field, enter ContextualWiringApplication and then click Finish.
5. From the main menu, choose Run > Start Server Instance.
6. From the main menu, choose Help > WebCenter Preconfigured Server Readme.
7. Under Preconfigured Portlet Producers, click the Sample Portlets link under WSRP Portlet Producers.
8. In your browser, note the URL of the WSRP V2 WSDL.
9. In the Application Resources panel of the Application Navigator, right-click Connections and choose New Connection and then WSRP Producer.
10. In the Specify Producer Name page of the Register WSRP Portlet Producer wizard, in the Producer Registration Name field, enter MyWsrpSampleProducer and then click Next.
11. In the Specify Connection Details page, in the WSDL URL field, enter the URL of your sample WSRP producer (as noted in step 8), and then click Next.
   For example:
   http://127.0.0.1:7101/portletapp/portlets/wsrp2?WSDL
12. In the Specify Additional Registration Details page, click Finish.

If you prefer to work through these examples using your own portlets, you must ensure that you have created portlets that support parameters. For information about how to create WSRP portlets, see Section 31.2.1, "How to Create a JSR 168 Java Portlet.” For more specific information about how to implement parameters in your WSRP portlets, see Section 32.1.2, "How to Implement Navigational Parameters (WSRP 2.0).”

34.7.1 How to Link Portlets With Navigational Parameters

The simplest way to link WSRP 2.0 portlets, when you simply want to link two portlets, is to use navigational parameters.
To link your WSRP 2.0 portlets using navigational parameters:

1. In the Application Navigator, right-click the ViewController project of the ContextualWiringApplication and choose New.
2. In the New Gallery, expand Web Tier, select JSF and then JSF page, and click OK.
3. In the Create JSF Page dialog, in the File Name field, enter PortletsWithNavParam.jspx.

4. Select Create as XML Document (*.jspx) and click OK.
5. In the Component Palette, select ADF Faces from the dropdown list.

6. From the Layout panel, drag the Panel Splitter component onto the page you just created.
7. In the Property Inspector, for the Panel Splitter component, click or expand the Style tab and then click the Box tab.
8. In the Width field, enter 600 and select px - Pixel from the corresponding dropdown list.
9. Click or expand the Appearance tab, and in the SplitterPosition field, enter 300.

10. If the Integrated WLS is not running, from the main menu, choose Run > Start Server Instance.

11. In the Application Resources panel of the Application Navigator, under the Connections node, expand WSRP Producer and then expand MyWSRPSampleProducer.

   You should have registered the sample WSRP producer. If you have registered the producer, see the "Before You Begin" section.

12. Drag the Parameter Form Portlet onto the first facet of the PortletsWithNavParam page.

13. Drag the Parameter Display Portlet onto the second facet of the page.

   Adding the portlets to the page creates six page variables: three page variables for the three portlet parameters of each portlet. To link the portlets, you must make the two portlets use the same page variables for their parameters. In this example, the Parameter Display Portlet sets three page variables to the values entered. The Parameter Display Portlet then uses the values from these same page variables to determine the values of its parameters.

14. In the Projects panel of the Application Navigator, right-click PortletsWithNavParam.jspx and choose Go to Page Definition.

15. In the Structure panel of the Application Navigator, expand the executables node, then ParameterFormPortlet1_1, then parameters.

16. Select each parameter in turn and, from the Property Inspector, make a note of the page variable assigned to it. This should be the same as shown in Table 34–10:

<table>
<thead>
<tr>
<th>Portlet Parameter</th>
<th>pageVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>ora_wsrp_navigparam_Parameter1</td>
<td>ParameterFormPortlet1_1_ora_wsrp_navigparam_Parameter1</td>
</tr>
<tr>
<td>ora_wsrp_navigparam_Parameter2</td>
<td>ParameterFormPortlet1_1_ora_wsrp_navigparam_Parameter2</td>
</tr>
<tr>
<td>ora_wsrp_navigparam_Parameter3</td>
<td>ParameterFormPortlet1_1_ora_wsrp_navigparam_Parameter3</td>
</tr>
</tbody>
</table>
17. Now to make the portlet parameters for the Parameter Display Portlet point to the same page variable as the Parameter Form Portlet so that the two portlets communicate correctly.

In the Structure panel, expand **ParameterDisplayPortlet1_1**, then **parameters**.

18. Select each parameter in turn and, in the Property Inspector, set the **pageVariable** property to the same page variables as used by the Parameter Form Portlet, instead of those created for the Parameter Display Portlet. Use **Table 34–11** for guidance:

**Table 34–11  Parameter Settings for the Parameter Display Portlet**

<table>
<thead>
<tr>
<th>Portlet Parameter</th>
<th>pageVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>ora_wsrp_navigparam_Parameter1</td>
<td>ParameterFormPortlet1_1_ora_wsrp_navigparam_Parameter1</td>
</tr>
<tr>
<td>ora_wsrp_navigparam_Parameter2</td>
<td>ParameterFormPortlet1_1_ora_wsrp_navigparam_Parameter2</td>
</tr>
<tr>
<td>ora_wsrp_navigparam_Parameter3</td>
<td>ParameterFormPortlet1_1_ora_wsrp_navigparam_Parameter3</td>
</tr>
</tbody>
</table>

19. Next, you must ensure that the Parameter Display Portlet refreshes itself whenever the Parameter Form Portlet is updated to display the correct values.

In the PortletsWithNavParam page, select the Parameter Form Portlet.

20. In the Property Inspector, click or expand the **Common** tab and make a note of the value in the **Id** field (this should be *portlet1*).

21. In the PortletsWithNavParam page, select the Parameter Display Portlet.

22. In the Property Inspector, click or expand the **Common** tab and in the **PartialTriggers** field, enter the ID of the Parameter Form Portlet that you noted earlier (**portlet1**).

23. From the main menu, choose **File > Save All**.

24. In the Projects panel of the Application Navigator, right-click **PortletsWithNavParam.jspx** and choose **Run**.

### 34.7.2 What Happens at Runtime

When you run the page, you should see one portlet that enables you to change the value of the parameters (the Parameter Form Portlet) and another that displays the value of the parameters (the Parameter Display Portlet). When you change the value of a parameter in the Parameter Form Portlet and click **OK**, the corresponding parameter in the Parameter Display Portlet updates automatically to display the new value (see **Figure 34–10**).
34.7.3 How To Associate Navigational Parameters with ADFm Events

ADFm events provide a powerful way of linking many components, including WSRP 2.0 portlets. When a portlet is placed on a page, an event is raised by the portlet if the portlet has parameters. By subscribing to that event, many components can be updated when the parameters change.

The following procedure provides a simple example that shows how to update the parameters of a single portlet using ADFm events. For an example that shows how to update the parameters of several portlets, see Section 34.7.5, "How To Cascade Events Across Multiple Components."

To associate parameters with an ADFm event:
1. In the Application Navigator, right-click the ViewController project of the ContextualWiringApplication and choose New.
   You created this application in Section 34.7.1, "How to Link Portlets With Navigational Parameters."
2. Create a JSPX page using the instructions provided in steps 1 through 13 in Section 34.7.1, "How to Link Portlets With Navigational Parameters," setting the File Name of the page to PortletsWithAdfmEvents.jspx.
3. In the Projects panel of the Application Navigator, right-click PortletsWithAdfmEvents.jspx and choose Go to Page Definition.
5. In the Event Map Editor dialog, click the Add a New Event Entry icon.
6. In the Add New EventMap Entry dialog, from the Producer dropdown list, select ParameterFormPortlet2_1.
   This identifies the Parameter Form Portlet as the component that triggers the event.
7. The Event Name field is automatically populated with the value ParameterFormPortlet2_1_Event.
8. From the Consumer dropdown list, select ParameterDisplayPortlet2_1.
   This identifies the Parameter Display Portlet as a component that receives values when the event is triggered.
9. In the Consumer Params table, add the parameters listed in Table 34–12 by clicking the Add Consumer Parameter icon and specifying the name and value for each parameter.

This maps the portlet parameters in the consumer (the Parameter Display Portlet) to the payload of the event triggered by the producer (the Parameter Form Portlet).

<table>
<thead>
<tr>
<th>Table 34–12 Parameter Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Param Name (Consumer)</strong></td>
</tr>
<tr>
<td>ora_wsrp_navigparam_Parameter1</td>
</tr>
<tr>
<td>ora_wsrp_navigparam_Parameter2</td>
</tr>
<tr>
<td>ora_wsrp_navigparam_Parameter3</td>
</tr>
</tbody>
</table>

10. Click OK to close the Add New EventMap Entry dialog, and then OK again to close the Event Map Editor dialog.

11. Next, you must ensure that the Parameter Display Portlet refreshes itself whenever the Parameter Form Portlet is updated to display the correct values.

In the PortletsWithAdfmEvents page, select the Parameter Form Portlet.

12. In the Property Inspector, click or expand the Common tab and make a note of the value in the Id field (this should be portlet1).

13. In the PortletsWithAdfmEvents page, select the Parameter Display Portlet.

14. In the Property Inspector, click or expand the Common tab and in the PartialTriggers field, enter the ID of the Parameter Form Portlet that you noted earlier (portlet1).

15. From the main menu, choose File > Save All.

16. In the Application Navigator, right-click PortletsWithAdfmEvents.jspx and choose Run.

34.7.4 What Happens at Runtime

When you run the page, you should see one portlet that enables you to change the value of the parameters (the Parameter Form Portlet) and another that displays the value of the parameters (the Parameter Display Portlet). When you change the value of a parameter in the Parameter Form Portlet and click OK, the corresponding parameter in the Parameter Display Portlet updates automatically to display the new value (see Figure 34–10).
When you run the page, you should see one portlet that enables you to change the value of the parameter and another that displays the value of the parameter. When you change the value of the parameter in one portlet and click OK, the other portlet should update automatically to display the new value.

### 34.7.5 How To Cascade Events Across Multiple Components

The main advantage of associating your navigational parameters with ADFm events is the capability to cascade the events across many components. In this way, you can link many more components than you could with just navigational parameters alone.

**To cascade events across multiple components**

1. In the Application Navigator, right-click the ViewController project of the ContextualWiringApplication and choose New.
   
   You created this application in Section 34.7.1, "How to Link Portlets With Navigational Parameters."

2. Create a JSPX page using the instructions provided in steps 1 through 8 in Section 34.7.1, "How to Link Portlets With Navigational Parameters," setting the File Name of the page to CascadingAdfmEvents.jspx and the SplitterPosition to 200.

3. In the Layout panel of the Component Palette, drag a second Panel Splitter component onto the second facet of the page.

Your page should now look something like Figure 34–12:
4. If the Integrated WLS is not running, from the main menu, choose Run > Start Server Instance.

5. In the Application Resources panel of the Application Navigator, under the Connections node, expand WSRP Producer and then expand MyWSRPProducer. You should have registered the sample WSRP producer. If you have registered the producer, see the "Before You Begin" section.

6. Drag the Parameter Form Portlet onto the leftmost first facet of the PortletsWithNavParam page.

7. Drag the Parameter Display Portlet onto the other first facet of the page.

8. Drag another Parameter Display Portlet onto the second facet of the page.

9. In the Projects panel of the Application Navigator, right-click CascadingAdfmEvents.jspx and choose Go to Page Definition.

10. In the Structure panel of the Application Navigator, right-click CascadingAdfmEventsPageDef and choose Edit Event Map.

11. In the Event Map Editor dialog, click the Add a New Event Entry icon.

12. In the Add New EventMap Entry dialog, from the Producer dropdown list, select ParameterFormPortlet3_1. This identifies the Parameter Form Portlet as the component that triggers the event.

13. The Event Name field is automatically populated with the value ParameterFormPortlet3_1_Event.

14. From the Consumer dropdown list, select ParameterDisplayPortlet3_1.
This identifies the first Parameter Display Portlet as a component that receives values when the event is triggered.

15. In the Consumer Params table, add the parameters listed in Table 34–13 by clicking the Add Consumer Parameter icon and specifying the name and value for each parameter.

This maps the portlet parameters in the consumer (the Parameter Display Portlet) to the payload of the event triggered by the producer (the Parameter Form Portlet).

### Table 34–13 Parameter Values

<table>
<thead>
<tr>
<th>Param Name (Consumer)</th>
<th>Param Value (Producer)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ora_wsrp_navigparam_Parameter1</td>
<td>${payLoad.ora_wsrp_navigparam_Parameter1}</td>
</tr>
<tr>
<td>ora_wsrp_navigparam_Parameter2</td>
<td>${payLoad.ora_wsrp_navigparam_Parameter2}</td>
</tr>
<tr>
<td>ora_wsrp_navigparam_Parameter3</td>
<td>${payLoad.ora_wsrp_navigparam_Parameter3}</td>
</tr>
</tbody>
</table>

16. Click **OK** to close the Add New EventMap Entry dialog.

17. In the Event Map Editor dialog, click the Add a New Event Entry icon.

18. In the Add New EventMap Entry dialog, from the **Producer** dropdown list, select **ParameterDisplayPortlet3_1**.

   This identifies the first Parameter Display Portlet as the component that triggers the event.

19. The **Event Name** field is automatically populated with the value **ParameterDisplayPortlet3_1_Event**.

20. From the **Consumer** dropdown list, select **ParameterDisplayPortlet4_1**.

   This identifies the second Parameter Display Portlet as a component that receives values when the event is triggered.

21. In the Consumer Params table, add the parameters listed in Table 34–14 by clicking the Add Consumer Parameter icon and specifying the name and value for each parameter.

   This maps the portlet parameters in the consumer (the Parameter Display Portlet) to the payload of the event triggered by the producer (the Parameter Form Portlet).

### Table 34–14 Parameter Values

<table>
<thead>
<tr>
<th>Param Name (Consumer)</th>
<th>Param Value (Producer)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ora_wsrp_navigparam_Parameter1</td>
<td>${payLoad.ora_wsrp_navigparam_Parameter1}</td>
</tr>
<tr>
<td>ora_wsrp_navigparam_Parameter2</td>
<td>${payLoad.ora_wsrp_navigparam_Parameter2}</td>
</tr>
<tr>
<td>ora_wsrp_navigparam_Parameter3</td>
<td>${payLoad.ora_wsrp_navigparam_Parameter3}</td>
</tr>
</tbody>
</table>

22. Click **OK** to close the Add New EventMap Entry dialog, and then **OK** again to close the Event Map Editor dialog.

23. Next, you must ensure that the portlets refreshes themselves to display the correct values.

   In the CascadingAdfmEvents page, select the Parameter Form Portlet.
24. In the Property Inspector, click or expand the Common tab and make a note of the value in the Id field (this should be portlet1).

25. In the CascadingAdfmEvents page, select the first Parameter Display Portlet.

26. In the Property Inspector, click or expand the Common tab and in the PartialTriggers field, enter the ID of the Parameter Form Portlet that you noted earlier (portlet1).

27. Make a note of the value in the Id field (this should be portlet2).

28. In the CascadingAdfmEvents page, select the second Parameter Display Portlet.

29. In the Property Inspector, click or expand the Common tab and in the PartialTriggers field, enter the ID of the first Parameter Display Portlet that you noted earlier (portlet2).

30. From the main menu, choose File > Save All.

31. In the Application Navigator, right-click CascadingAdfmEvents.jspx and choose Run.

34.7.6 What Happens at Runtime

When you run the page, you should see one portlet that enables you to change the value of the parameters (the Parameter Form Portlet) and two that display the values of the parameters (the two Parameter Display Portlets). When you change the value of a parameter in the Parameter Form Portlet and click OK, an event is raised and the values from the parameters are sent as the payload to the first Parameter Display Portlet, which updates automatically to display the new value (see Figure 34–13). This then triggers an event in the first Parameter Display Portlet, which in turn provides a payload for the second Parameter Display Portlet.

In our example, the first Parameter Display Portlet does not perform any processing on the parameters, so the values in the second Parameter Display Portlet are the same. However, the advantage of using ADFm events over the simpler navigational parameters method is that the first Parameter Display Portlet could perform some processing on the parameter values and then pass the updated values to the second Parameter Display Portlet.
**Figure 34–13  Cascading ADFm Events**

<table>
<thead>
<tr>
<th>Parameter Form Portlet</th>
<th>Parameter Display Portlet</th>
<th>Parameter Display Portlet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter 1: Cascading</td>
<td>Parameter 2: ADFm</td>
<td>Parameter 3: Events</td>
</tr>
<tr>
<td>Parameter 2: ADFm</td>
<td>Parameter 2: ADFm</td>
<td>Parameter 3: Events</td>
</tr>
<tr>
<td>Parameter 3: Events</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

OK
Creating Portlets with OmniPortlet

OmniPortlet is a data publishing portlet that you add to your application at design time, and customize at runtime. It provides a runtime, wizard-based experience to allow page designers to publish data from a variety of data sources, including SQL, XML, web services, spreadsheets, and web pages to several different layouts, such as customizable charts and tables.

This chapter covers the information you may find useful if you are planning to use OmniPortlet in your Oracle JDeveloper environment. For information on how to use the OmniPortlet wizard, refer to Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

This chapter includes the following sections:

- Section 35.1, "Introduction to OmniPortlet"
- Section 35.2, "Adding OmniPortlet to Your Application"
- Section 35.3, "Customizing OmniPortlet"
- Section 35.4, "Troubleshooting OmniPortlet Problems"

For information about registering and configuring OmniPortlet, see Section 34.2, "Registering Portlet Producers with a Custom WebCenter Application" and Section F.2, "OmniPortlet Configuration Tips."

---

Note: You can find more information about developing different types of portlets and information about producers and other portlet technologies in Chapter 29, "Overview of Portlets."

35.1 Introduction to OmniPortlet

OmniPortlet is a subcomponent of Oracle WebCenter Framework that enables developers to easily publish data from various data sources using a variety of layouts without writing any code. You can base an OmniPortlet on almost any kind of data source, such as a web service, a SQL database, spreadsheet (character-separated values), XML, and even application data from an existing web page. Figure 35–1 shows an example of a portlet created using OmniPortlet.
OmniPortlet enables the custom WebCenter 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.

### 35.2 Adding OmniPortlet to Your Application

As described in Chapter 34, “Consuming Portlets,” you can add an OmniPortlet to a page created through Oracle JDeveloper. OmniPortlet is included in the Integrated WebLogic Server (Integrated WLS) that is installed with Oracle JDeveloper. After you start the Integrated WLS, you can register the OmniPortlet producer by using the Register Oracle PDK-Java Producer wizard. When this producer is registered, the portlets become available on the Oracle JDeveloper Resource Palette or in the Application Resources panel. For example, if you register the OmniPortlet producer as "OmniProducer," OmniPortlet displays in your IDE Connections list in the Resource Palette, as shown in Figure 35–2.
35.3 Customizing OmniPortlet

After you add an OmniPortlet to your application at design time, you can customize the content, layout, and other options, by running your application to a browser. This section provides a high-level introduction to the runtime customization experience. For more detailed information on using and customizing this portlet, refer to Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

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.
You can use several different data sources with OmniPortlet:

- Spreadsheet
- SQL
- XML
- Web Service
- Web Page

For more information on using these data sources with OmniPortlet, refer to the Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

### 35.4 Troubleshooting OmniPortlet Problems

This section provides information to help you troubleshoot problems you may encounter while using OmniPortlet.

**Note:** As the OmniPortlet producer exists and executes in a tier different from the WebCenter 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.

---

**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```

---

As the OmniPortlet producer exists and executes in a tier different from the WebCenter application and does not have access to the session information, you must expose XML files as PUBLIC in order for OmniPortlet to access them.
Web Clipping is a publishing portlet that enables you to integrate any web application with your WebCenter application. Web Clipping is designed to give you quick integration by leveraging the existing user interface of a web application. With Web Clipping, you can collect web content into portlets in a single centralized web page. You can use Web Clipping to consolidate content from web sites scattered throughout a large organization.

This chapter includes the information you are required to know to use Web Clipping in your Oracle JDeveloper environment. For information about how to use Web Clipping at runtime, refer to the "Working with the Web Clipping Portlet" chapter in the Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

This chapter includes the following sections:

- Section 36.1, "Introduction to Web Clipping"
- Section 36.2, "Adding Web Clipping to Your Application"
- Section 36.3, "Integrating Authenticated Web Content Using Single Sign-On"
- Section 36.4, "Advanced Features of Web Clipping"
- Section 36.5, "Current Limitations of Web Clipping"
- Section 36.6, "Troubleshooting Web Clipping"

For information about additional Web Clipping portlet producer configuration such as repository and proxy settings and producer security, see Appendix F, "Additional Portlet Configuration."

You can find information about developing different types of portlets and information about producers and other portlet technologies in Chapter 29, "Overview of Portlets."

### 36.1 Introduction to Web Clipping

With Web Clipping you can clip an entire web page or a portion of it and reuse it as a portlet. You can clip basic and HTML-form-based sites. Use Web Clipping when you want to copy content from an existing web page and expose it in your WebCenter application as a portlet.

Web Clipping supports the following features:

- Navigation through various styles of login mechanisms
Web Clipping supports various login mechanisms including form- and JavaScript-based submission and HTTP Basic and Digest Authentication with cookie-based session management.

- **Fuzzy matching of clippings**
  Fuzzy matching enables the Web Clipping engine to correctly identify a web clipping and deliver it as portlet content even if the web clipping is reordered within the source page or if its character font, size, or style changes.

- **Personalization**
  Personalization enables you to expose input parameters that end users can modify when they personalize the portlet. Parameters can be exposed as public parameters that you can map as page parameters. This feature enables end users to obtain personalized clippings.

- **Integrated authenticated web content through single sign-on**
  You can use external applications and leverage Oracle Single Sign-On to clip content from authenticated external web sites.

- **Inline rendering**
  Inline rendering enables you to set up Web Clipping portlets to display links within the context of the portlet. When a user clicks a link in the Web Clipping portlet, the results display within the same portlet. You can use this feature with internal and external web sites.

- **Proxy authentication**
  Web Clipping supports proxy authentication, including global proxy authentication and authentication for each user. You can use this feature if proxy servers require authentication. You can specify proxy server authentication details, including type (Basic or Digest) and realm in the `provider.xml` file. In addition, you can specify a scheme for entering user credentials:
  - All users automatically log in using a user name and password that you provide.
  - All users are required to log in using a user name and password that they provide.
  - All public users (not authenticated into the WebCenter application) automatically log in using a user name and password that you provide, while valid users (authenticated into the WebCenter application) log in using a user name and password that they provide.

  For more information, see Section F.3.2, "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 F.3.3.1, "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 36.4.1, "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 36.5, "Current Limitations of Web Clipping."

**Persistent storage of Web Clipping definitions**

Web Clipping definitions are stored persistently in a repository. By default, in JDeveloper, the Web Clipping producer is configured to use the file-based Oracle Metadata Service (MDS) as a repository. If you prefer, you can use a database schema for your Web Clipping repository. For information about configuring a Web Clipping repository, see Section F.3.1, "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.

### 36.2 Adding Web Clipping to Your Application

You can add a Web Clipping portlet to a JSP document created through JDeveloper. For more information, see Chapter 34, "Consuming Portlets." The Web Clipping portlet producer is included in:

- Integrated WebLogic Server (WLS), the default server installed with JDeveloper
- The WLS_Portal managed server in the default domain in a full Oracle Fusion Middleware installation

To add a Web Clipping portlet to your application:

1. In JDeveloper, start Integrated WLS. For information, see Section 3.9, "Using Integrated WLS."

2. In JDeveloper, register the Web Clipping producer. For more information, see Section 34.2.3, "How to Register an Oracle PDK-Java Portlet Producer."

During this step, you must provide the endpoint URL, for example:

http://localhost:7101/portalTools/webclipping/providers/webClipping

You can choose to register the Web Clipping portlet producer from the WLS_Portal managed server or Integrated WLS.
When the producer is registered, portlets become available either in the Application Resources panel or the Resource Palette.

3. From the Application Resources panel or the Resource Palette, drag the Web Clipping portlet onto your .jspx page. For more information, see Section 34.3, "Adding Portlets to a Page."

If you are using a PanelCustomizable or ShowDetailFrame component, then drag the portlet on top of that component instead of af:form. In the Structure pane, the Web Clipping portlet must display under the PanelCustomizable or ShowDetailFrame component. In the Page Editor, the portlet must display inside the PanelCustomizable or ShowDetailFrame component.

---

**Note:** When you add an instance of Web Clipping to your page in JDeveloper, open the Property Inspector for the portlet and ensure that the AllModesSharedScreen and RenderPortletInIFrame properties are set as follows:

- **AllModesSharedScreen** is set to `false` to display the Customize and Personalize in full page size. If you do not set this property to `false`, then when you edit the Web Clipping portlet at runtime, the text displayed on the page may be distorted.

- **RenderPortletInIFrame** is set to `true` to display the Web Clipping in its own inline frame (IFRAME) in the View mode.

---

4. Right-click the .jspx page and choose Run.

The Web Clipping portlet displays in your default browser.

5. Customize the Web Clipping portlet to define its contents. For information, see the "Working with the Web Clipping Portlet" chapter in the Oracle Fusion Middleware User's Guide for Oracle WebCenter.

---

**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 F.3.3.1, "Adding Certificates for Trusted Sites."

---

### 36.3 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.

To integrate an external application that requires authentication:

1. Open your custom WebCenter application in JDeveloper.

2. Register the external application, specifying the authentication information. For more information, see Section 37.2.3.1.1, "Registering an External Application in Oracle JDeveloper."

3. Register the Web Clipping producer. For information, see Section 34.2.3, "How to Register an Oracle PDK-Java Portlet Producer."
While registering the producer by using the Register Oracle PDK-Java Portlet Producer wizard, on the Specify Connection Details page, select the **Associate Producer with External Application** checkbox, and from the list of values, select the external application that you just registered. The **Enable Producer Sessions** checkbox gets selected automatically.

Figure 36–1 shows the external application named MyOracleSupport associated with a Web Clipping producer.

**Figure 36–1  Associating a Web Clipping Producer with an External Application**

4. Add the Web Clipping portlet from the producer that you just registered to a `.jspx` page.

5. Right-click the `.jspx` page and choose **Run**.

6. If you did not specify shared or public credentials for the external application, then the portlet contains an **Update login information** link. Click this link.

7. Enter your credentials, and then click **OK** to log in to the external web site.

Figure 36–2 shows the login screen of an external application named MyOracleSupport.
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 36–3).

You can now select any section of a web page that you intend to display in your Web Clipping portlet. For information about how to customize a Web Clipping portlet, see the "Working with the Web Clipping Portlet" chapter in the *Oracle Fusion Middleware User’s Guide for Oracle WebCenter*.

After you clip the required page of the external application, the web clipping, even though it is from a page requiring authentication, is available in your portlet.

### 36.4 Advanced Features of Web Clipping

Web Clipping supports certain advanced features. You can configure custom authentication methods by using the Web Clipping Transport API and rewrite image links to use a resource proxy.

This section includes the following subsections:

- **Section 36.4.1, "Using the Web Clipping Transport API"**
- **Section 36.4.2, "Rewriting Image Links to Use a Resource Proxy"**
36.4.1 Using the Web Clipping Transport API

To support custom authentication methods, you can use the Web Clipping Transport API. To extend the Web Clipping transport layer to support custom authentication methods, you must perform several implementation and deployment procedures.

This section includes the following subsections:
- Section 36.4.1.1, "Implementing the Web Clipping Transport API"
- Section 36.4.1.2, "Deploying the Web Clipping Transport API"

36.4.1.1 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:
   - HttpClientStudioTransportLiaison, which handles HTTP transport in Web Clipping Studio mode
   - HttpClientProviderTransportLiaison, which handles HTTP transport in Web Clipping Producer show mode
   To support more authentication methods, you must override the addRequestHeaders methods for both the Studio and Provider HttpClientTransportLiaison implementations to add your own authentication-specific headers. For information, see Oracle WebLogic Server Web Clipping Transport API Reference.

2. Compile the new subclasses and package them into a JAR file.
   For example, to compile the new subclasses, use the following command:
   javac -classpath path_to_wcejar -d classes/
   Where, path_to_wcejar refers to the path to the wce.jar file.
   To create the JAR file, for example, run the following command from the classes directory:
   jar cvf ../mytransport.jar
   Where, mytransport.jar refers to the JAR file users want to create.

36.4.1.2 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
Current Limitations of Web Clipping

HttpURLConnection and HttpClientTransportLiaison:

- 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.

36.4.2 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>

36.5 Current Limitations of Web Clipping

When you use Web Clipping, you should be aware of the following limitations.

- If the site that you intend to clip uses a large amount of JavaScript to manipulate cookies or uses the document.write JavaScript method to modify the HTML document being written, then you may not be able to clip content from the site.

- When you integrate with partner applications (by using mod_osso), you cannot clip directly through those partner applications in an authenticated manner. However, you can use partner applications through the external application framework.

- You cannot use the Web Clipping portlet to clip Oracle Portal pages and ADF pages. As a workaround, reregister the same producer in the destination portal and edit the portal manually.

- You cannot use the Web Clipping portlet to clip a web page that contains multiple frames, that is, a frameset.

- Note the following about Web Clipping and the use of a CSS:
  - If a web page contains multiple portlets that use a CSS, then they should not conflict if the CSS uses distinct style names, such as OraRef, to specify a style within an HTML tag, rather than using an HTML tag name, such as <A>, as the name of the style.
  - If one portlet uses a CSS, and that CSS overwrites the behavior of HTML tags by using the name of the tag, such as <A>, as the name of the style, and a second portlet on the same page does not use a CSS, the second portlet is affected by the style instructions of the CSS of the first portlet.
  - If two portlets on the same page use a different CSS and each CSS overwrites the behavior of HTML tags by using the name of an HTML tag, such as <A>,
as the name of the style, then the style that is displayed depends on the browser.

- Web Clipping checks for globalization support settings in the following way:

  1. Web Clipping checks the `Content-Type` in the HTTP header for the `charset` attribute. If this is present, then it assumes that this is the character encoding of the HTML page.

  2. If the `charset` attribute is not present, then Web Clipping checks the HTML `META` tag on the page to determine the character encoding.

  3. If the HTML `META` tag is not found, then Web Clipping uses the `charset` in the previous browsed page. If this is the first page, then it defaults to the ISO-8859-1 character encoding.

  4. If the value of the `charset` for `Content-Type` or `META` tag is not supported (for example, if the `charset` was specified as `NONE`), then Web Clipping uses the default character set, ISO-8859-1, not the `charset` in the previously browsed page.

- To use the Web Clipping portlet, you must use Netscape 7.0 or later, Microsoft Internet Explorer 5.5 or later for Windows 2000, or Microsoft Internet Explorer 6.0 or later for Windows XP. If you use browser versions older than these, then you may encounter JavaScript errors.

### 36.6 Troubleshooting Web Clipping

This section provides information to help you troubleshoot problems you may encounter while using Web Clipping.

**Encountered "x" at line n, column n. Was expecting one of : "x", "y" ...**

Parser error message written to the log file.

**Problem**

The web content displayed in the Web Clipping portlet contains invalid HTML or JavaScript.

**Solution**

This is a site-specific issue, not a Web Clipping error. Contact the site’s administrator for assistance.
Part IX

Completing Your WebCenter Application

Part IX contains the following appendixes:

- Section 37, "Securing Your WebCenter Application"
- Section 38, "Testing and Deploying Your WebCenter Application"
- Section 39, "Working Productively in Teams"
Securing Your WebCenter Application

This chapter describes how to use Oracle ADF Security in your WebCenter application to handle authentication and authorization.

This chapter includes the following sections:

- Section 37.1, "Introduction to WebCenter Application Security"
- Section 37.2, "Working with External Applications"
- Section 37.3, "Setting Up Security for Your Application"
- Section 37.4, "Creating a Login Portlet"
- Section 37.5, "Creating Login Pages and a Login Component"
- Section 37.6, "Adding Portlets to a Login Page"
- Section 37.7, "Creating a Public Welcome Page for Your Application"
- Section 37.8, "Configuring Basic Authentication for Testing Portlet Personalization"
- Section 37.9, "Registering Custom Certificates with the Keystore"
- Section 37.10, "Overriding Inherited Security on Portlets and Customizable Components"
- Section 37.11, "Identity Propagation Mechanisms"
- Section 37.12, "Securing Identity Propagation Through WSRP Producers with WS-Security"
- Section 37.13, "Implementing PDK-Java Portlet Security"

37.1 Introduction to WebCenter Application Security

WebCenter applications are dynamic and often involve input from users in the form of customization and personalization, and require a flexible security model. Consequently, the WebCenter security model is based on the ADF security model rather than the more traditional J2EE security model. ADF Security is a framework built upon the Java Authentication and Authorization Service (JAAS) standard. See Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework for a complete introduction to ADF Security.

37.2 Working with External Applications

Oracle WebCenter Framework defines an external application as any application that implements its own authentication process. That is, an application that does not take
part in the WebCenter Framework application’s single sign-on process. In some cases, the identity management solution may be the same, but the authentication process can be different.

When WebCenter Framework Service interacts with an application that handles its own authentication, you can associate that service with an external application to allow for credential provisioning. Therefore, the use of an external application definition provides a means of accessing content from these independently authenticated applications.

To replicate a single sign-on experience from the end user’s perspective, the external application service captures the username and password, and any other credentials for the external application, and supplies it to the WebCenter service requiring it. The WebCenter service then uses this and logs in on behalf of the end user. This username and password combination is securely stored in a credential store configured for the WebLogic domain where the application is deployed.

The user provides login credentials when prompted, and these credentials are mapped to the WebCenter application user and stored in the credential store configured for the domain. The credential store subsequently supplies that information during authentication to the external application. Unless the external application’s credentials change, the user supplies the credentials only once as the mapped information is read from the credential store for future requests.

---

**Note:** When logging in to an external application, if you clear the **Remember My Login Information** checkbox, then the credentials provisioned for that user session are lost if there is a failover in a high availability (HA) environment. You are prompted to specify the credentials again if you try to access the external application content in the same user session.

---

The external applications that are to be used by the custom WebCenter application can be specified before deployment through a wizard in Oracle JDeveloper, or post-deployment through the application server management interfaces (WebLogic Scripting Tool and Oracle Enterprise Manager). See *Oracle Fusion Middleware Administrator’s Guide* for more information.

This section contains the following:

- **Section 37.2.1, "Using External Applications"**
- **Section 37.2.2, "Supplying User Credentials"**
- **Section 37.2.3, "Managing External Applications"**

### 37.2.1 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 37.2.1.1, "Secured Service Connections"**
- **Section 37.2.1.2, "Automated Single Sign-On"**
37.2.1.1 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 37–1.

**Note:** The following WebCenter Services must be configured with external applications for credential mapping support to be available:

- Instant Messaging and Presence (IMP)
- Mail
- Documents
- RSS

For more information about the identity propagation mechanisms used by WebCenter Services, see Section 37.11, "Identity Propagation Mechanisms".

---

**Figure 37–1 Configure a New Mail Connection Page**

![Configure a New Mail Connection Page](image)

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 37–2.
At run time, the producer uses the information associated with the external application to authenticate the user to the application, and consequently consume its portlets. The producer code is responsible for actually performing the authentication interaction with the external application. The external application support provided with WebCenter Framework simply provides the information needed for authentication to the portlet producer. The use of external applications is supported for both Oracle PDK-Java as well as WSRP producers.

For example, a producer provides a stock portfolio portlet from a portlet-producing application that has its own authentication mechanism. In this case the developer:

- Defines the external application. This can be done through the Oracle JDeveloper wizard or through Oracle Enterprise Manager.
- Associates the external application with the portlet producer.

### 37.2.1.2 Automated Single Sign-On

With automated single sign-on, the user directly links to the application and is automatically authenticated to the secured web application, as their credentials are retrieved from the credential store. This provides the end user with a seamless single sign-on experience.

**Note:** Automated login is not supported for:

- External applications using BASIC authentication.
- External applications configured for SSO.
- External applications with a customized login form (built using ADF Faces) that does not implement the J2EE security container login method `j_security_check` for authentication.
- External sites that do not support UTF8 encoding.
Rather than using a URL directly to the web application, links to the application are proxied through the external application’s automated login servlet (adfextapplogin). If the user is not authenticated to the external application and they have not previously stored their credentials in the credential store, they will be challenged for their password through the credential provisioning page discussed below. If, however, the user has previously defined credentials, they will be returned from the credential store and the user will automatically be logged on to the application.

The proxy URL references the external application in question and redirects to the URL that is specified in the external application definition.

/adfextapplogin?extappid=<extappid>

For example, if you had a WebCenter application in which you defined an external application that represented the myoracle.com Web site (external application identifier is "myoracle"), the proxy URL would look like this:

/adfextapplogin?extappid=myoracle

The link’s target attribute should also be set appropriately. For example, if you use <a href=>, then set the target attribute appropriately in addition to specifying the target in the href. The target attribute specified for the servlet will determine how the Cancel button functions as described below.

/adfextapplogin?extappid=<extappid> [target= _self | _blank]

If you specify target=_blank the link opens in new window. If you specify target=_self the link opens in current window. If the target parameter is not specified the link opens in current window.

This parameter also affects how the Cancel button on the credential provisioning page works. If _blank is specified, the new window is closed when Cancel is clicked; if _self is specified (or the target parameter is not used), the user is returned to the calling page.

Note: Automated login for external sites is not supported for sites that do not support UTF8 encoding.

37.2.2 Supplying User Credentials

How you allow an end user to define their credentials for an external application depends on the use of the external application. For most services, the credential provisioning screen is incorporated into the task flow that the service exposes and for these no further configuration steps are required. You can, however, add the External Application - Change Password task flow component to applications using these services thereby allowing the end user to preemptively set the appropriate user name and password for each of the external applications that is registered with your custom WebCenter 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 37.2.3, "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 18.2.2, "Adding the IMP Service at Design Time".

At run time, the credential provisioning screen displays login data fields composed of the data fields specified through external application registration. Users fill in the data fields with their login information for the specified external application and that login information is passed to the external application or service. Entering the credentials in the provisioning screen also results in the credentials being persisted in the credential store configured for the WebLogic domain.

By default, the login information the user entered is preserved in a credential store, which handles logins for future sessions. The user does not have to enter login information again (unless the user's credentials change). However, the end user can choose to use the information for the current session by deselecting the Remember my Login Information checkbox on the credential provisioning page.

### 37.2.3 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 37.2.3.1, "Working with External Applications in Oracle JDeveloper"
- Section 37.2.3.2, "Working with External Applications in Enterprise Manager"
- Section 37.2.3.3, "Working with External Applications Using WLST"

#### 37.2.3.1 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 37.2.3.1.1, "Registering an External Application in Oracle JDeveloper"
- Section 37.2.3.1.2, "Editing External Application Registration Details in Oracle JDeveloper"
- Section 37.2.3.1.3, "Deleting External Application Registration Details in Oracle JDeveloper"

#### 37.2.3.1.1 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 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 WebCenter applications. Select **Application Resources** to make the external application available only in the WebCenter application in which it is registered, or select **Resource Palette** to
make the external application available from the Resource Palette to any new WebCenter 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 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 37.2.3.1.2, "Editing External Application Registration Details in Oracle JDeveloper". If you leave the field blank the display name is set to the value of the Name field by default.

7. Click Next.

8. On the General page, in the Login URL field enter the URL to which the HTML login page is submitted.
   View the HTML source of the application’s login form to retrieve this URL.

   **Note:** The fields on the General pane are only required if the external application being defined participates in click-through login as described in Section 37.2.2, “Supplying User Credentials”.

9. In the User Name/ID FieldName field, enter the label that the application uses for the user name field, for example, User Name.

10. In the Password FieldName field, enter the label that the application uses for the password field, for example Password.

11. From the Authentication Method list, select the application’s login method.
    Choose from the following:
    - **GET**
      Presents a page request to a server. Submits the login credentials as part of the login URL.
    - **POST**
      Submits login credentials within the body of a form.

12. Click Next.

13. On the Additional Fields page, enter the names and values of any additional fields that are submitted with the external application's login form:
    Click the Add Field button to create an input field:
    - **Field Name**
      Enter a unique name for any additional field that requires user input on the external application HTML login form.
    - **Field Value**
Enter a default value for the corresponding field name.

- **Display to User**
  Select to display the field on the external application login screen. If the field is not displayed (unchecked), then a default value must be specified, which will be used to login into the external application for all users. If the value is user-specific, then the field must be displayed to the user provisioning page.

---

**Note:** The Delete Field option can be used to delete selected rows.

---

14. Click **Next**.
15. On the Shared Credentials page, select the **Specify Shared Credentials** check box and specify a **Username** and **Password** if you want all authenticated users to access the external application using this credential. Authenticated users will not be challenged to provide their credentials when they access the external application.
16. Click **Next**.
17. On the Public Credentials page, select the **Specify Public Credentials** check box and specify a **Username** and **Password** to be used for all unauthenticated (public) users accessing the external application. This is required when the external application content is accessed through one of the WebCenter Services (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 custom WebCenter 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 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

---

37.2.3.1.2 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 37.2.3.1, "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.

37.2.3.1.3 Deleting External Application Registration Details in Oracle JDeveloper

To delete external application registration information, perform the following steps:

1. In the Application Navigator, from the Application Resources pane under the Connections node, right-click an external application and select Delete from the context menu.

   Alternatively, you can select an external application in the Applications Navigator and from the Edit menu, select Delete.

2. In the External Application Delete dialog box, select Yes.

   Oracle recommends that you remove any references to services, such as a portlet producer, with which the external application is associated. Failing to do so will likely result in runtime errors, because the components will attempt to communicate with the external application.

37.2.3.2 Working with External Applications in Enterprise Manager

Just as you can create, edit, and delete external application registration details in Oracle JDeveloper, you can also do this in Enterprise Manager. See the section "Working with External Applications" in the Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

37.2.3.3 Working with External Applications Using WLST

As well as Oracle JDeveloper and Enterprise Manager, you can also use WebLogic Scripting Tool (WLST) commands to create, edit, and delete external application registration details. See the section "Working with External Applications" in the Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

37.3 Setting Up Security for Your Application

Refer to the section on adding security to an ADF application in the Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework for information about securing your application by using Oracle ADF Security, which is based on JAAS.
37.4 Creating a Login Portlet

Refer to the *Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework* for more information on creating a login portlet for your application.

37.5 Creating Login Pages and a Login Component

Refer to the *Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework* for more information on creating login pages and login components.

37.6 Adding Portlets to a Login Page

The *Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework* describes how you can create an ADF Faces-based login page for your application. In this section, you will add some portlets to such a login page so that the login page becomes indistinguishable from the other pages in your WebCenter application.

Make sure your portlet producers have been registered before proceeding. See Section 34.2, "Registering Portlet Producers with a Custom WebCenter Application" for details.

To add portlets to the login page, perform the following steps:

1. Drag a PanelCustomizable onto the h:form tag that is inside the first cust:panelCustomizable tag you added to the login page.

2. From the Component Palette, select RichTextPortlet Producer, then select the Rich Text portlet from the list and drag it onto the PanelCustomizable component.

3. From the Component Palette, select ADF Faces Core and drag an ObjectSeparator below the Rich Text portlet on the PanelCustomizable component.

4. From the Component Palette, select OmniPortlet Producer, then select the OmniPortlet from the list and drag it onto the PanelCustomizable component.

5. Save the page.

Because the login page is called from the container as part of the login process, the request must be forwarded to the ADF binding filter to establish the appropriate portlet and security context. To do this, you must configure a mapping for the ADF Binding filter in the web.xml file. To do this, perform the following steps:

1. In the Applications Navigator, expand the WEB-INF node, right-click web.xml and select properties to open the property palette.

2. Select Filter Mappings in the left panel and click add to define a new mapping for the adfBindings Filter. This displays the Create Web Application Filter Mapping dialog box.

3. Specify adfBindings for the filter name and click the Servlet Name option and specify Faces Servlet as the servlet name. Ensure that the Forward and Include dispatcher types are selected as shown in Figure 37–3.
4. Click OK.

Follow the steps in the Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework to complete the creation of the login page.

### 37.7 Creating a Public Welcome Page for Your Application

WebCenter applications are not secured by default. If you have implemented ADF Security for your WebCenter application and intend to make part of the application public, you must provide a starting point or home page for unauthenticated users. To create this public welcome page, refer to the section on adding security to an ADF application in the Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework.

### 37.8 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:

1. Create a user and the role manager. See the Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework for information about creating users and roles.

2. Secure your application using the ADF Security Wizard. See the Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework for the steps to be performed. On the Login page of the wizard, select HTTP Basic Authentication (RFC 2617). This specifies that the application will use basic authentication.

3. Run the page in the integrated WLS and log in as a valid user and test your portlet’s edit mode.

When you are done testing your portlet’s Edit mode, you can quickly remove this test security by do the following:

---

**Note:** This procedure is useful for any portlet that has an Edit mode (Omniportlet, Web Clipping, JPS, and PDK-Java).
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.

37.9 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.

4. In the Security Alert dialog box, shown in Figure 37–4, 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 37–5. Select the root certificate and click View Certificate.

Figure 37–5 Certificate Dialog Box

7. Click the Details tab, and click Copy to File.

8. In the Certificate Export Wizard, accept the default settings and click Next until you reach the File to Export screen, shown in Figure 37–6.

Figure 37–6 File to Export Screen of the Certificate Export Wizard

9. In the File Name field, enter `<JDEV_HOME>/jdk/jre/lib/security/root.cer` and click Next.

10. Click Finish.

11. In the command prompt, set your default directory to `<JDEV_HOME>/jdk/jre/lib/security` and run the following command:
keytool -import -file root.cer -keystore cacerts -storepass changeit

By running this command, the root.cer certificate is imported into the keystore.

12. Enter y at the prompt to confirm that you trust this certificate.

13. Verify that the cacerts file is updated with the certificate.

37.10 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 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 37–1 and Example 37–2 and set the enableSecurity element in those sections to true.

Example 37–1 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 37–2 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.
Section 37.10.1, "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 11.5, "Applying Action-Level Restrictions on Panel Customizable and Show Detail Component Actions."

37.10.1 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

37.10.1.1 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 37-1 describes the different actionsCategory attributes and the portlet actions they support by default.
Example 37–3 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 37–3 actionsCategory Element in the Portlets Security Section**

```xml
<!--
==============================================================================
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>
</adfp:config-child>
```

### 37.10.1.2 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.

<table>
<thead>
<tr>
<th>Attribute Value</th>
<th>Actions Supported</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>viewActionsCategory</code></td>
<td>Render, <code>isHelpModeAvailable</code>, <code>isNormalModeAvailable</code>, <code>isAboutModeAvailable</code>, <code>isPreviewModeAvailable</code>, <code>isDetailModeAvailable</code>, <code>isLinkModeAvailable</code>, <code>isPrintModeAvailable</code></td>
</tr>
<tr>
<td><code>customizeActionsCategory</code></td>
<td><code>showMoveAction</code>, <code>showRemoveAction</code>, <code>isCustomizeModeAvailable</code>, <code>showMinimizeAction</code>, <code>showMaximizeAction</code>, <code>isConfigModeAvailable</code></td>
</tr>
<tr>
<td><code>personalizeActionsCategory</code></td>
<td><code>isPersonalizeModeAvailable</code></td>
</tr>
</tbody>
</table>

*Table 37–1 actionsCategory Attributes and Portlets Actions Mapping*
Example 37–4 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 37–4  actions Element in the Portlets Security Section**

```xml
<!--
==============================================================================
PORTLETS ACTIONS SECURITY
==============================================================================
-->
  <adfp:enableSecurity value="true"/>
  <adfp:actionsCategory>
    ..........................................
  </adfp:actionsCategory>
  <adfp:actions>
    <adfp:action name="Render" value="true"/>
    <adfp:action name="showMoveAction" value="true"/>
    <adfp:action name="isCustomizeModeAvailable" value="false"/>
    <adfp:action name="isPersonalizeModeAvailable" value="true"/>
  </adfp:actions>
</adfp:adf-config-child>
```

**Using EL to Prevent Customization of Portlets Outside of Business Hours**

An example to show when you may need to override inherited portlet security is an application that is configured to disable portlet customization outside of standard business hours. For this, you must first create a managed bean (for example, a managed bean called appBusinessRules), containing the method shown in Example 37–5.

**Example 37–5  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 37–6.

**Example 37–6  InsideBizHours Method Referenced in the adf-config.xml File**

```xml
  <adfp:enableSecurity value="true"/>
  <adfp:actionsCategory>
    <adfp:actionCategory name='customizeActionsCategory'
```
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 37–7 to the managed bean:

**Example 37–7 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 37–8.

**Example 37–8 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.InsideBizHours?"true":"false"}"/>
    <adfp:actionCategory name="personalizeActionsCategory"
        value="#{appBusinessRules.InsideBizHours?"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.

### 37.11 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 WebCenter application.

Whenever possible, WS-Security is the preferred means of identity propagation. Where WS-Security cannot be used due to legacy restrictions or pre-defined store-specific standards or specifications, the primary mechanism used is the credential mapping capability provided by the External Applications service to obtain the user's credentials for a remote application using a distinct security provider. For more information about WS-Security, see the Oracle Fusion Middleware Security and Administrator’s Guide for Web Services. For more information about External Applications, see Section 37.2, “Working with External Applications”.

<table>
<thead>
<tr>
<th>Service</th>
<th>Connection Type</th>
<th>Identity Propagation Mechanism</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conference</td>
<td>Webex</td>
<td>External Application</td>
</tr>
<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>Oracle 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>File System</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>Portal 10g/11g</td>
<td>JSR-170 (External Application)</td>
<td></td>
</tr>
<tr>
<td>Day Adapters</td>
<td>JSR-170 (External Application)</td>
<td></td>
</tr>
<tr>
<td>EMail</td>
<td>EMail Connection</td>
<td>External Application</td>
</tr>
<tr>
<td>LDAP Connection</td>
<td>External Application</td>
<td></td>
</tr>
<tr>
<td>Events - Personal</td>
<td>Exchange Server Connection</td>
<td>External Application</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 Communication Server (LCS)</td>
<td>SOAP, Web Services calls. External Application</td>
</tr>
<tr>
<td></td>
<td>Oracle WebLogic Communications Services (OWLCS)</td>
<td>Oracle WS-Security</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

37.12 Securing Identity Propagation Through WSRP Producers with WS-Security

The Web Services for Remote Portlets (WSRP) specification indicates that Web Services Security (WS-Security) can be leveraged for providing secure identity propagation between the consumer and the portlet producer. However, WSRP in and of itself does not provide secure identity propagation of the end user’s identity to the portlet producer. The WSRP specification explicitly defers to other security standards for secure identity propagation and does not go into the specific WS-Security profiles or options that may be employed. In the absence of a secure mechanism, WSRP defines the concept of user categories, which can be mapped to security roles like the ones used by the JSR168 portlets. By using a combination of WSRP and WS-Security, you can ensure end-to-end security.

This section covers the following:
- Identity Propagation Without WS-Security
- Identity Propagation with WS-Security
- Configuring Security for WSRP Portlets

37.12.1 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 37–9 shows how a sample portlet’s markup rendering code uses the **isUserInRole** API to decide what content to display.

**Example 37–9.isUserInRole(role) API**

```java
private void doViewHtml(RenderRequest
```
When WS-Security is leveraged with WSRP, the user’s identity is propagated outside of the SOAP message body in the WS-Security header. This is a user assertion, using the Username Token format, and is digitally signed to authenticate the consumer and to ensure the integrity of the assertion.

When this mechanism is used, the JSR 168 APIs `isUserInRole` and `getUserPrincipal` are established based on the security context resulting from the WS-Security authentication, rather than the information in the SOAP message’s `userContext`.

Also, when an Anonymous (i.e., PUBLIC) client is mapped to a default user using the producer’s Default User connection parameter, the WSRP producer must be configured with `strict-authentication` and a grant must be added to the Policy store. For more information, see "Adding a Grant to the Policy Store for a Mapped User Identity” in the Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter.

The use of WS-Security adds some complexity to the configuration and management of the WebCenter application and the set of producers it consumes. However, when the situation warrants its use, it becomes an important ingredient of the SOA architecture that ensures the security of the information being published by the WebCenter application.

Oracle WebCenter Framework supports the following token profiles (to digitally sign the security token and message body to ensure authenticity and integrity):

- Username token without password
- Username token with password
- SAML token (uses the `sender vouches` method that the producer uses to confirm the subject assertion)

Digitally signing the security token and the SOAP message body accomplishes the following objectives:

- Consumer Authentication
- Assertion and Message Integrity
Securing Identity Propagation Through WSRP Producers with WS-Security

- **Supported Producers**

- **Security Domain Implication**

**Consumer Authentication**
When a portlet producer is generating sensitive information, for example paystub information, it is imperative that it only responds to requests to show the information from a legitimate consumer.

By using WS-Security, and having the consumer digitally sign the security token and the message body, the producer can verify the signature using the public key of the legitimate consumer. If the signature cannot be verified, then it means that the request may have come from a fraudulent consumer. By requiring the verification of the digital signature, the sensitive information will only be sent to the legitimate consumer.

**Assertion and Message Integrity**
In addition to verifying the identity of the consumer making the Web Service requests, digitally signing the security token and the message body also ensures that the token and the message have not been tampered with. This prevents such problems as man-in-the-middle attacks where a legitimate request might be intercepted and the user name in the security token replaced with another user name to see the paystub information coming back for the other user. By digitally signing the token, it cannot be tampered with. Any modification to the token would result in the inability to verify the signature on the producer end, and would result in a SOAP fault to be returned instead of the requested paystub information.

**Supported Producers**
WS-Security implementation is supported by the Oracle WebCenter Suite WSRP container. Other WSRP vendors may also be able to support the WS-Security configuration of Username Token without password, with XML digital signature on the Username Token and the SOAP Message body.

**Security Domain Implication**
When using secure identity propagation as described in this section, the user name of the user authenticated to the consumer (WebCenter 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 37–7** summarizes the overall WSRP portlet security architecture.
37.12.3 Configuring Security for WSRP Portlets

Before you configure the producer for WS-Security, you must first deploy your standards-compliant portlet producer to the Oracle WebCenter Suite WSRP Container by performing the steps described in Chapter 33, "Testing and Deploying Your Portlets".

After you have deployed the producer, configure the producer for WS-Security by performing the following steps:

- Attaching a policy to the producer endpoint
- Creating the keystores
- Configuring the producer
- Configuring the consumer

These steps are described in Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter.

37.13 Implementing PDK-Java Portlet Security

This section describes the available security services for your PDK-Java portlet.

For more detailed information about the PDK classes referred to in this section, see the JavaDoc on OTN by clicking Java Doc API on the Portlet Development page available at

http://www.oracle.com/technology/products/ias/portal/portlet_development_10g1014.html
37.13.1 Assumptions

The following assumptions are made to perform the tasks in this section:

1. You have followed through and understood Section 31.2, "Creating Java Portlets".
2. You built a portlet using the wizard and successfully added it to a page.

37.13.2 Introduction to PDK-Java Portlet Security Features

This section introduces the major features that are available to secure your PDK-Java portlet producers.

37.13.2.1 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.

37.13.2.2 Authorization

Authorization determines if a particular user may view or interact with a portlet.

37.13.2.3 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 (WebCenter application) and the producer to digitally sign messages. See Section 37.13.5, "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 37.13.6, "User Input Escape" for more information.

37.13.3 Single Sign-On

Portlets may act as windows into an application. They display summary information and provide a way to access the full functionality of the application. Portlets display some portions of the application in the WebCenter 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.

37.13.3.1 External Application

An external application uses a different authentication server than the WebCenter application. This means that when a user is logged into the WebCenter application, you want to also log them into the external application without having to type in their user name or password.

Applications that manage the authentication of users can be loosely integrated with the WebCenter Framework if the administrator registers them as external applications. When a user who was previously authenticated by the WebCenter Framework accesses an external application for the first time, WebCenter Framework attempts to authenticate the user with the external application. The authentication process submits an HTTP request that combines the registration information and the user's user name and password for the application. If the user has not yet registered their user name and password for the external application, then the user is prompted for the required information before making the authentication request. When a user supplies a user name and password for an external application, WebCenter Framework maps the new user name and password to the WebCenter 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).

37.13.3.2 No Application Authentication

The producer trusts the WebCenter 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 application.

37.13.4 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 WebCenter 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:


37.13.4.1 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.

37.13.5 Message Authentication

Message authentication uses a digital signature. The signature is generated using a Hashed Message Authentication Code (HMAC) algorithm and is based on user information, the shared key, and a UTC (Universal Time, Coordinated) timestamp. The producer authenticates the message using its own copy of the shared key. This technique can be used in Secure Sockets Layer (SSL) communication with a producer instead of client certificates.

For caching purposes, show request signatures are calculated each time a session is set up so producers using message authentication must always have Producer Sessions enabled. This also means that there is a trade-off between performance and security. Shorter session timeouts mean less chance of a message being re-sent illegally, but there is a performance overhead associated with reestablishing a provider session.

A single producer instance cannot support multiple shared keys because it could cause security and administration problems. For instance, if one copy of the shared key is compromised in some way, then the producer administrator has to create a key and distribute it to all of the application clients, who then must update their producer definitions. The way around this problem is to deploy different producer services, specifying a unique shared key for each service. Each producer service has its own deployment properties file so that each service is configured independently of the others. The overhead of deploying multiple producer services within the same producer adapter is relatively small.

While the signature element provides protection against interception and resending of messages, it does nothing to prevent interception and reading of message contents. Messages are still transmitted in plain text. If you are concerned about the content of messages being read by unauthorized people, then this must used in conjunction with SSL to encrypt the message.

The advantage of message authentication is as follows:

■ Ensures that the message received by a producer comes from a legitimate WebCenter 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.

37.13.6 User Input Escape

By accepting user input without escaping it to text, you run the risk of an XSS attack, where an attacker attempts to pass in malicious scripts through user input forms. For example, if a portlet title is customizable, then an attacker might attempt to pass scripts or commands to the portlet through the title string. PDK-Java provides the following features to ensure that you can protect your portlets from such attacks:

■ Default Container Encoding
Escape Methods

37.13.6.1 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 32.2.4.2, "Setting JNDI Variable Values" for more information.

37.13.6.2 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:

```java
title = UrlUtils.escapeString(data.getPortletTitle());
```

37.14 Troubleshooting Security Issues
This section provides troubleshooting information to help diagnose security related issues for custom WebCenter applications.

This section includes the following troubleshooting notes:

- Section 37.14.1, "Error Message Appears When Running a Page with a Content Repository Data Control Method Being Consumed"

37.14.1 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
        UIViewHandler viewHandler = context.getApplication().getViewHandler();
        UIViewRoot dialog =
            viewHandler.createView(context,
                "/oracle/adfinternal/extapp/view/pages/CredentialProvisionerDialog.jspx*");

        HashMap<String, Object> properties = new HashMap<String, Object>();
```
```java
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 test custom WebCenter applications using the Integrated WebLogic Server (Integrated WLS) that comes packaged with Oracle JDeveloper and is typically configured as the default server (DefaultServer). This chapter also describes how to use JDeveloper to deploy applications to an Oracle WebLogic Managed Server configured to test application deployment or host applications for production.

For information about deploying custom WebCenter applications through other mechanisms, such as Oracle Enterprise Manager Fusion Middleware Control, Oracle WebLogic Administration Console, and WebLogic Scripting Tool (WLST) commands, see the chapter, "Deploying WebCenter Applications" in Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter.

This chapter includes the following sections:

- Section 38.1, "Introduction to Oracle WebLogic Servers"
- Section 38.2, "Testing a Custom WebCenter Application on the Integrated WebLogic Server"
- Section 38.3, "Packaging and Deploying a Custom WebCenter Application to a WebLogic Managed Server"
- Section 38.4, "Transporting Customizations Between Environments"

38.1 Introduction to Oracle WebLogic Servers

This section provides an overview of the Integrated WebLogic Server (or Integrated WLS) and Oracle WebLogic Managed Servers. Using Oracle JDeveloper, you can test your applications on the Integrated WLS and deploy applications to an Oracle WebLogic Managed Server that resides outside Oracle JDeveloper for staging and further testing or, if you have the required permissions, to an actual production environment.

Integrated WebLogic Server

The Integrated WebLogic Server (Integrated WLS) comes packaged with Oracle JDeveloper and is the default server (DefaultServer). The Integrated WLS connection appears as IntegratedWLSConnection in the Resource Palette under IDE Connections > Application Server. With Integrated WLS, you can quickly and easily test your application at design time without having to perform an enterprise deployment.

You can manually start the Integrated WLS by choosing the Run > Start Server Instance menu option in JDeveloper. Doing so creates a single server instance for all
Testing a Custom WebCenter Application on the Integrated WebLogic Server

Testing custom WebCenter applications on the Integrated WLS helps to improve design-time tasks by:

- Optimizing deployment: the requirement to archive and copy files to a separate server for design time testing is eliminated.
- Instantly refreshing changes to the application: you can directly run most files from project directories, which enables you to make selected changes and refresh these changes while the application is running. For example, you can modify a task flow on a JSPX page, then refresh the page in the browser to view your changes without redeploying the application.
- Removing the requirement to undeploy applications after testing and debugging. If you stop the Integrated WLS, however, applications running on it are automatically undeployed.

To learn more about testing a custom WebCenter application on the Integrated WLS, refer to Section 38.2, "Testing a Custom WebCenter Application on 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 custom WebCenter application to an Oracle WebLogic Managed Server, see Section 38.3, "Packaging and Deploying a Custom WebCenter Application to a WebLogic Managed Server."

38.2 Testing a Custom WebCenter Application on the Integrated WebLogic Server

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, then refresh your JSPX page in your browser to view the 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).

The Integrated WLS is preconfigured to run applications in Oracle JDeveloper, which means you do not need to create deployment profiles. To run a custom WebCenter application, in the Application Navigator, right-click the .jspx page in the project folder and choose Run. Alternatively, you can manually start the Integrated WLS by choosing Run > Start Server Instance. If you start an Integrated WLS instance manually, then all applications you subsequently run will run on this single instance of the Integrated WLS.

Running the application page triggers the packaging and deployment of your custom WebCenter application on an Integrated WLS instance named after the application. You can stop instances of the Integrated WLS by using the Run Manager panel (shown...
To stop an instance, just select **DefaultServer** and click the red Stop icon on the Run Manager tab.

![Run Manager](image)

Use the DefaultServer Log window (shown in **Figure 38–2**) to view how the activity is progressing.

![DefaultServer Log Window](image)

**Note:** When the Integrated WLS instance stops, the application is undeployed, and therefore, becomes unavailable.

For a more persistent testing scenario, you can deploy your application to the Integrated WLS by right-clicking the application and then selecting **Deploy**. This deploys the application to the DefaultServer instance and will always be available when the DefaultServer is running. If you choose this method, then you must first create deployment profiles, as described in Section 38.3.1.2, "Creating Deployment Profiles." If you deploy your application to the Integrated WLS, then the Deployment Configuration dialog displays to let you configure and customize deployment settings. The file system MDS repository pre-created by JDeveloper displays in the **Repository Name** field.

If you manually started the Integrated WLS, you can stop the server instance by choosing **Run > Stop Server Instance**.
Oracle WebCenter temporarily modifies mds-config in the adf-config.xml file to reconfigure the directory paths for the running application. To clean up any runtime customizations, choose the menu option Build > Clean Runtime MDS Customizations. For example, if you have included Oracle Composer in your application and changed the look and feel at runtime, you can reset these customizations back to your original application by removing the customizations stored in the MDS.

---

**Note:** The Integrated WLS creates an archive directory of your custom WebCenter application at `system_directory/o.j2ee/drs`.

Secure attributes and credentials are migrated by default when running a custom WebCenter application on the Integrated WLS.

### 38.3 Packaging and Deploying a Custom WebCenter Application to a WebLogic Managed Server

When you're ready to test your custom WebCenter 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 custom WebCenter applications using Fusion Middleware Control, Oracle WebLogic Administration Console, and WLST commands, see the chapter, "Deploying WebCenter Applications" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter*.

This section includes the following:

- Section 38.3.1, "Packaging a Custom WebCenter Application"
- Section 38.3.2, "Preparing the Target Environment for Deployment"
- Section 38.3.3, "Deploying a Custom WebCenter Application to a Managed Server"

#### 38.3.1 Packaging a Custom WebCenter Application

When you deploy custom WebCenter 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 custom WebCenter 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 38.3.1.1, "Packaging Database Connections and Application Security"
- Section 38.3.1.2, "Creating Deployment Profiles"

#### 38.3.1.1 Packaging Database Connections and Application Security

When you deploy custom WebCenter 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 custom WebCenter 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 38.3.1.1.1, "Packaging the Database Connections"
- Section 38.3.1.1.2, "Packaging Application Security Data"

38.3.1.1.1 Packaging the Database Connections

If your custom WebCenter application contains JDBC database connections, you'll need to choose how JDeveloper will migrate those database connections to the Oracle WebLogic Managed Server. You can choose from:

- **Global data source:**
  Oracle recommends that you choose this type of data source if you plan to deploy an application EAR file to a Managed Server running in production mode using Fusion Middleware Control, the Oracle WebLogic Administration Console, or using a WLST (wldeployer) command.
  
  To choose a global data source, deselect the *Auto Generate and Synchronize weblogic-jdbc.xml Descriptors During Deployment* checkbox when you create the EAR file so that JDeveloper does not create an application-level data source with password indirection. Instead, create the global data source using Oracle WebLogic Administration Console after deploying an application to a Managed Server in production mode. For more information on creating global data sources, see the section "Creating a JDBC Data Source" in *Oracle Fusion Middleware Configuring and Managing JDBC for Oracle WebLogic Server*.

- **Application-level data source with password indirection:**
  JDeveloper generates an application-level data source with password indirection by default when you run an application on the Integrated WLS. That is, the *Auto Generate and Synchronize weblogic-jdbc.xml Descriptors During Deployment* checkbox is selected by default in the Application Properties dialog, as shown in Figure 38–3.
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:
  
  ```xml
  <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:
  
  ```xml
  <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:
  
  ```xml
  <resource-ref>
  <description>WC-Connection</description>
  <res-ref-name>jdbc/WCDS</res-ref-name>
  <res-type>javax.sql.DataSource</res-type>
  <res-auth>Container</res-auth>
  </resource-ref>
  ```

If you choose to generate an application data source with password indirection, then you must add credential mappings using the Oracle WebLogic Administration Console to be able to activate the application. For more information on adding credential mappings, see the section "Creating a JDBC Data
Packaging and Deploying a Custom WebCenter Application to a WebLogic Managed Server

Source" in Oracle Fusion Middleware Configuring and Managing JDBC 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.

---

### 38.3.1.1.2 Packaging Application Security Data

The following is applicable if you have configured security in your application using the Configure ADF Security wizard, or the application contains connections with secure attributes such as a database connection password or external applications with shared or public credentials:

- **Packaging Credentials**
- **Packaging Identity Data**
- **Packaging Application Policies**

#### Packaging Credentials

If you do not intend to migrate credentials, then deselect the **Credentials** checkbox in the **Security Deployment Options** section in the Application Properties dialog (see Figure 38–4).

![Figure 38–4 Security Deployment Options](image)

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 custom WebCenter application to a Managed Server running in production mode, secure attributes of the connections packaged within your application and any shared or public credentials specified for external applications**
are not migrated to the domain-level credential store. This is because secure properties are likely to be different between development and production environments. Therefore, you must reconfigure secure attributes of connections and shared or public credentials for external applications using WLST commands or Fusion Middleware Control, as described in the Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter.

For more information on credential migration behavior, see the section "Configuring the Credential Store" in Oracle Fusion Middleware Security Guide.

■ When you deploy a custom WebCenter 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 38–4, in the Security Deployment Options section, the Users and Groups checkbox is selected by default. This indicates that the users and groups defined in the application's jazn-data.xml file will be migrated to the identity store configured for the WebLogic Managed Server, if the authenticator configured in the domain allows creation of users and groups. When migrating the 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 checkbox is selected, then the jps.policystore.migration parameter in the weblogic-application.xml file is set to OVERWRITE. This means that during application deployment and redeployment, the application policies packaged with the application will be overwritten on the domain-level policy store for the domain to which the application is being deployed.

If the Application Policies checkbox is not selected, then the jps.policystore.migration parameter is set to MERGE. This means that the application policies will be migrated during application deployment but not during redeployment, and the existing application policies will not be overwritten.

38.3.1.2 Creating Deployment Profiles

To deploy your application to a Managed Server that resides outside JDeveloper, you must first create deployment profiles. A deployment profile packages or archives a custom WebCenter application and its associated files so that the application can be deployed to an Oracle WebLogic Managed Server. You create deployment profiles at both the project and the application level.

At the project level, you create a Web Application Archive (WAR) deployment profile, which contains the projects to be deployed and any associated files, such as, pages, Java classes, weblogic.xml, web.xml, and so on.

At the application level, you create an Enterprise Archive (EAR) deployment profile that 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 custom WebCenter 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.

**Note:** You can deploy Oracle ADF applications, such as custom WebCenter applications, only as EAR files. Therefore, while creating deployment profiles, ensure that the WAR and MAR files of the application are included in the application’s EAR file.

This section includes the following sub-sections:

- Creating the WAR Deployment Profile
- Creating the EAR Deployment Profile

### 38.3.1.2.1 Creating the WAR Deployment Profile

To create a project-level WAR deployment profile:

1. In JDeveloper, in the Application Navigator, right-click ViewController and select New.
2. In the New Gallery, expand General, select Deployment Profiles, then WAR File, and then click OK.
3. In the Create Deployment Profile -- WAR File dialog, enter a name for your deployment profile and click OK.
4. In the WAR Deployment Profile Properties dialog, select the Specify Java EE Web Context Root option, enter a meaningful context root, and then click OK.

   The "context root" is the part of the Web application’s URL that is appended to the server ID and port number, and to which the servlet name is in turn appended. For example, in the URL http://www.hostname.example.com/myWebsite/HelloWorld, /myWebsite is the context root (or virtual path), and HelloWorld is the name of the servlet the user is accessing.
5. In the Project Properties dialog, click OK.

### 38.3.1.2.2 Creating the EAR Deployment Profile

To create an application-level EAR deployment profile:

1. In JDeveloper, in the Application Navigator, right-click the application and select New.
2. In the New Gallery, expand General, select Deployment Profile, then EAR File, and then click OK.
3. In the Create Deployment Profile – EAR File dialog, enter a name for your deployment profile and click OK.
4. In the Edit EAR Deployment Profile Properties dialog, select Application Assembly, then select the project WAR you want to make available in the project file and click OK.
5. In the Application Properties dialog, select the deployment options to use and click OK.

For a description of the **Auto Generate and Synchronize weblogic-jdbc.xml Descriptors During Deployment** option, see Section 38.3.1.1, "Packaging the
6. In the Application Navigator, right-click the application name, choose **Deploy**, *deployment profile name*, to, and then choose to **EAR file**. This creates the EAR file in the *deploy* folder located in `JDEV_HOME\mywork\application_name\deploy\`, as shown in **Figure 38–5**.

**Figure 38–5  EAR Status in Deployment - Log**

---

### 38.3.2 Preparing the Target Environment for Deployment

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

This section includes the following sub-sections:

- **Section 38.3.2.1, “Creating and Provisioning an Oracle WebLogic Managed Server Instance”**
- **Section 38.3.2.2, “Creating and Registering the Metadata Service Repository”**
- **Section 38.3.2.3, “Configuring the Target Environment”**
- **Section 38.3.2.4, “Creating a WebLogic Managed Server Connection”**

#### 38.3.2.1 Creating and Provisioning an Oracle WebLogic Managed Server Instance

Before deploying a custom WebCenter application, you must also create a new WebLogic Managed Server instance and provision it with a required set of shared libraries. You can create, provision a WebLogic Managed Server instance using the WLS Administration Console as described in the section “Creating and Provisioning a WebLogic Managed Server Instance” in *Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter*.

**Note:** Oracle does not recommend deploying custom WebCenter applications to any of the three pre-configured Managed Servers created during the installation, or to a domain Administration Server.

#### 38.3.2.2 Creating and Registering the Metadata Service Repository

After creating and provisioning the WebLogic Managed Server instance, you must create and register a Metadata Service Repository (MDS) schema for the application on the WebLogic Domain’s Administration Server instance, as described in the section "Creating and Registering the Metadata Service (MDS) Repository" in *Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter*. 
38.3.2.3 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 the data sources, see the section "Creating a JDBC Data Source" in Oracle Fusion Middleware Configuring and Managing JDBC for Oracle WebLogic Server.

38.3.2.4 Creating a WebLogic Managed Server Connection

Before you can deploy your applications using JDeveloper to an Oracle WebLogic Managed Server instance that reside outside JDeveloper, you must create a connection to the Managed Server instance where you will deploy the application. Before you create a connection to an Oracle WebLogic Managed Server instance, ensure that the target Managed Server instance is up and running, and has the required libraries.

To create a connection to an Oracle WebLogic Managed Server:

1. In JDeveloper, from the File menu, select New.
2. In the New Gallery, expand General, select Connections, and then Application Server Connection.
3. Click OK.
4. In the Create Application Server Connection wizard, for Step 1, enter a name for the new connection (for example, WLS_CustomApp), and then click Next.
5. At Step 2, specify the user name and password for authentication, and then click Next.
6. At Step 3, enter the host name of the WebLogic Managed Server (for example, webcenter.myserver.example.com) and the port number (for example, 7888).
7. In the WLS Domain field, specify the name of the domain in which the WebLogic Managed Server instance is created (for example, wl_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.

38.3.3 Deploying a Custom WebCenter Application to a Managed Server

After you have created the deployment profiles and server connection, you can deploy your custom custom WebCenter application to an Oracle WebLogic Managed server instance.

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 custom WebCenter application to an Oracle WebLogic Managed Server instance:

1. In JDeveloper, in the Application Navigator, open the application to be deployed.
2. Right-click the application name, choose **Deploy**, and then select the deployment profile name.

   The Deployment Action dialog displays (see **Figure 38–6**).

   **Figure 38–6  Deployment Action Dialog**

   ![Deployment Action Dialog](image)

3. Select **Deploy to Application Server** and click **Next**.

   The Select Server dialog displays (see **Figure 38–7**).

   **Figure 38–7  Select Server Dialog**

   ![Select Server Dialog](image)

4. Select the connection name of the Managed Server (for example, `WLS_CustomApp`), and click **Next**.
In most cases, you will want to deselect **Deploy to all server instances in the domain** and select the specific servers and clusters to deploy to using the Server Instances dialog. Refer to the online help for more information about using the Select Server dialog options.

5. On the Deployment Summary dialog, shown in **Figure 38–8**, check that all of the deployment options are correct and click **Finish** to start the deployment.

**Figure 38–8  Deployment Summary Dialog**

6. Continue by doing any post-deployment security or data source connection configurations that may be required as described in the section on "Post-Deployment Configuration" in the **Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter**.

### 38.4 Transporting Customizations Between Environments

When migrating a deployed application to a new environment, post-deployment customizations made to pages, WebCenter Web 2.0 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 chapter, "Managing Backup, Recovery, Export, and Import of WebCenter" in **Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter**.
This chapter contains information about managing files in Oracle JDeveloper. It includes information about source control systems, namely CVS and Subversion.

This chapter includes the following sections:

- Section 39.1, "Enabling Source Control on WebCenter Applications"
- Section 39.2, "Understanding WebCenter Application Files Affected by Developers"
- Section 39.3, "Implementing Common Requirements One Time"
- Section 39.4, "Portlet Producer Considerations"

### 39.1 Enabling Source Control on WebCenter Applications

You can use CVS or Subversion for source control in a WebCenter application, similar to the way you would use a source control system in any other development environment.

There are two ways to use CVS or Subversion: In Oracle JDeveloper, either click the Versioning menu or use the Versioning Navigator (click View - Versioning Navigator). The Versioning Navigator is shown in Figure 39–1.

![Versioning Navigator](image)

This section contains the following subsections:

- Section 39.1.1, "Importing Application Files into CVS"
- Section 39.1.2, "Creating a Subversion Repository"

For detailed information about CVS and Subversion, see the Oracle JDeveloper online help system and the "Using a Source Control System" section in the Oracle Application Development Framework Developer’s Guide.
39.1.1 Importing Application Files into CVS

If you do not have your application files source controlled in CVS, then you must import them into CVS before you can get started.

To import application files into CVS:

1. In Oracle JDeveloper, from the **Versioning** menu, select **CVS** and then **Check Out Module**.

   If you have a CVS connection, then you are taken directly to step 2. If you do not have a CVS connection, then you are prompted to create a connection, as follows:

   ![Figure 39–2 Confirm Create Connection Dialog](image)

   **Note:** An alternative way to create a CVS connection is to right-click CVS in the Versioning Navigator and select **New CVS Connection**. Any existing CVS connection is shown in the Versioning Navigator.

   a. If the Confirm Create Connection dialog appears, then click **OK** (Figure 39–2).

   ![Figure 39–3 Create CVS Connection - Connection](image)

   b. In the Create CVS Connection wizard, on the Connection page, enter your connection information for CVS, then click **Next** (Figure 39–3).

   c. On the Root page (Figure 39–4), in the **Value of CVSROOT** field, enter a value for the CVSROOT. Click **Next**.
d. On the Test page (Figure 39–5), click Test Connection. The Log In to CVS dialog displays, as shown in Figure 39–6.

**Figure 39–5 Create CVS Connection - Test Connection**
e. Enter your password, and then click OK.

f. If the test is successful, then you should see something similar to Figure 39–7. Click Next.

g. On the Name page, enter a value for **Connection Name**, as shown in Figure 39–8. Click Next.
h. The Summary page displays with the connection information. Click Finish (Figure 39–9).

2. On the Check Out from CVS page, enter the application path.

3. Click OK to execute the import operation. You can see the connection now in the Application Navigator.

39.1.2 Creating a Subversion Repository

To create a Subversion repository:
1. From the **Versioning** menu, choose Subversion, and then **Create Repository**. The Create Subversion Repository dialog displays (Figure 39–10).

**Figure 39–10 Create Subversion Repository**

2. In the **Create Subversion Connection** dialog, enter connection details, and then click **OK**. (Figure 39–11)

---

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

**Figure 39–11 Create Subversion Connection**

3. Later you can edit these pages on the Edit Subversion Connection page (Figure 39–12).
39.2 Understanding WebCenter Application Files Affected by Developers

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

This section contains the following subsections:

- Section 39.2.1, "Files Associated With Common Objects"
- Section 39.2.2, "Developer Actions Affecting Metadata Files"

39.2.1 Files Associated With Common Objects

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

- Pages
- Portlets
- Services
- Data controls

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

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

39.2.2 Developer Actions Affecting Metadata Files

Table 39–1 lists developer actions and the files that are affected by these actions. Take this information into account while managing your files in CVS or Subversion.
<table>
<thead>
<tr>
<th>Action</th>
<th>Files Affected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Creating or editing the following connection:</td>
<td>Addss or updates:</td>
</tr>
<tr>
<td>■ BPEL</td>
<td>■ app-root/.adf/META-INF/connections.xml with the connection</td>
</tr>
<tr>
<td>■ Content Repository; for example, Oracle Content Server, Oracle Portal, file system</td>
<td>■ app-root/projectname/projectname.jpr with connection libraries</td>
</tr>
<tr>
<td>■ Database</td>
<td>■ app-root/.adf/META-INF/adf-config.xml with service-level configuration, including the default connection</td>
</tr>
<tr>
<td>■ Discussions</td>
<td>An external application connection, and all connections that require an external application connection (including Mail, Content Repository, and Instant Messaging and Presence) also add or update the following files:</td>
</tr>
<tr>
<td>■ External Application</td>
<td>■ app-root/src/META-INF/jps-config.xml with the credential store provider</td>
</tr>
<tr>
<td>■ Instant Messaging and Presence</td>
<td>■ app-root/src/META-INF/cwallet.sso with the user name and password</td>
</tr>
<tr>
<td>■ Mail</td>
<td>Note: All secure data specified for connections is stored in cwallet.sso, referenced in jps-config.xml. For example, the password for a database connection is stored in cwallet.sso.</td>
</tr>
<tr>
<td>■ Oracle Secure Enterprise Search</td>
<td></td>
</tr>
<tr>
<td>■ Oracle PDK-Java Producer</td>
<td></td>
</tr>
<tr>
<td>■ WSRP Producer</td>
<td></td>
</tr>
<tr>
<td>Creating a content repository data control</td>
<td>Addss or updates following data control files under app-root/projectname/adfmsrc:</td>
</tr>
<tr>
<td>■ projectpackage/DataControls.dcx with the data control definition</td>
<td></td>
</tr>
<tr>
<td>■ projectpackage/dcname.xml with the data control structure</td>
<td></td>
</tr>
<tr>
<td>■ projectpackage/dcname/*.xml with data structure definitions used by the data control</td>
<td></td>
</tr>
<tr>
<td>■ app-root/projectname/adfmsrc/META-INF/adfm.xml with a pointer to the DCX file</td>
<td></td>
</tr>
<tr>
<td>■ app-root/projectname/adfmsrc/return.xml with the definition for default operations used on collections</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Updates app-root/projectname/projectname.jpr with content repository libraries</td>
</tr>
</tbody>
</table>
### Table 39–1 (Cont.) Developer Actions Affecting Metadata Files

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

When adding a portlet from a portlet producer in the Resource Palette, WebCenter first registers the producer with the application (if it is not registered already). See the "De-registering an Oracle PDK-Java or WSRP portlet producer" row for files affected. It then adds or removes the portlet from the page, affecting the files described in this row.

**Adds or updates:**
- `app-root/projectname/adfmsrc/projectpackagename/DataBindings.cpx` with a new PageDef mapping (if it does not exist already)
- `app-root/projectname/public_html/WEB-INF/faces-config.xml` (This happens whenever you add an Oracle ADF component, such as a portlet, to a page.)
- `adf-faces-config.xml` (This happens whenever you add an Oracle ADF component to a page.)
- `app-root/projectname/public_html/WEB-INF/oracle-webservices.xml` if the portlet is WSRP

**Adds:**
- A number of files pertaining to the newly cloned portlet instance to `app-root/mds/`
- A number of WSDL and XML files to `app-root/projectname/public_html/WEB-INF/wsdl` if the portlet is WSRP

**Updates:**
- `app-root/projectname/public_html/pagename.jsp` with the `<adfp:portlet>` tag (JSF faces tags are replaced with Oracle ADF faces tags.)
- `app-root/projectname/adfmsrc/projectpackagename/pageDefs/pagenamePageDef.xml` with the region binding and page variable definitions
- `app-root/projectname/public_html/WEB-INF/web.xml` with `secure-ref` entries if the portlet is WSRP

<table>
<thead>
<tr>
<th>Action</th>
<th>Files Affected</th>
</tr>
</thead>
</table>
| Adding or removing a portlet on a page from a portlet producer connection in the Resource Palette | When adding a portlet from a portlet producer in the Resource Palette, WebCenter first registers the producer with the application (if it is not registered already). See the "De-registering an Oracle PDK-Java or WSRP portlet producer" row for files affected. It then adds or removes the portlet from the page, affecting the files described in this row. Adds or updates:  
- `app-root/projectname/adfmsrc/projectpackagename/DataBindings.cpx` with a new PageDef mapping (if it does not exist already)  
- `app-root/projectname/public_html/WEB-INF/faces-config.xml` (This happens whenever you add an Oracle ADF component, such as a portlet, to a page.)  
- `adf-faces-config.xml` (This happens whenever you add an Oracle ADF component to a page.)  
- `app-root/projectname/public_html/WEB-INF/oracle-webservices.xml` if the portlet is WSRP  
Adds:  
- A number of files pertaining to the newly cloned portlet instance to `app-root/mds/`  
- A number of WSDL and XML files to `app-root/projectname/public_html/WEB-INF/wsdl` if the portlet is WSRP  
Updates:  
- `app-root/projectname/public_html/pagename.jsp` with the `<adfp:portlet>` tag (JSF faces tags are replaced with Oracle ADF faces tags.)  
- `app-root/projectname/adfmsrc/projectpackagename/pageDefs/pagenamePageDef.xml` with the region binding and page variable definitions  
- `app-root/projectname/public_html/WEB-INF/web.xml` with `secure-ref` entries if the portlet is WSRP |
### Table 39–1 (Cont.) Developer Actions Affecting Metadata Files

<table>
<thead>
<tr>
<th>Action</th>
<th>Files Affected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adding a task flow for the following service:</td>
<td>Updates:</td>
</tr>
<tr>
<td>■ Announcements</td>
<td>• app-root/projectname/public_html/pagename.jspx with appropriate ADF faces tags</td>
</tr>
<tr>
<td>■ Discussions</td>
<td>• app-root/projectname/adfmsrc/projectpackagename/pagedefs/pagenamePageDef.xml with the region binding</td>
</tr>
<tr>
<td>■ Documents</td>
<td>• app-root/projectname/adfmsrc/projectpackagename/DataBindings.cpx with a listing of the page definitions</td>
</tr>
<tr>
<td>■ Links</td>
<td>• app-root/projectname/adfmsrc/META-INF/adfm.xml with a pointer to the data bindings file</td>
</tr>
<tr>
<td>■ Mail</td>
<td>• app-root/.adf/META-INF/adf-config.xml with MDS configuration entries</td>
</tr>
<tr>
<td>■ Page</td>
<td>• app-root/projectname/projectname.jpr with service libraries, dependent libraries, and JSP tag libraries</td>
</tr>
<tr>
<td>■ People Connections</td>
<td>• app-root/projectname/public_html/META-INF/web.xml with ADF library, external application, and MDS configuration entries</td>
</tr>
<tr>
<td>■ Recent Activities</td>
<td>Adds or updates all connection-related files for any service that requires connections</td>
</tr>
<tr>
<td>■ RSS Viewer</td>
<td>Copies over any required connection that was not present in the application, which updates</td>
</tr>
<tr>
<td>■ Search</td>
<td>app-root/.adf/META-INF/connections.xml and app-root/projectname/projectname.jpr</td>
</tr>
<tr>
<td>■ Tags</td>
<td><strong>Note:</strong> The Recent Activities, Search, and Tags services add the search extension to adf-config.xml and create the sample service-definition.xml file. The service-definition.xml file lets developers define QueryManager implementations for their service.</td>
</tr>
<tr>
<td>■ Worklist</td>
<td></td>
</tr>
<tr>
<td>■ External Application</td>
<td></td>
</tr>
<tr>
<td>Dragging a file or folder from a content repository connection to a page</td>
<td></td>
</tr>
<tr>
<td>Dragging a content repository data control on a page</td>
<td></td>
</tr>
</tbody>
</table>
### Table 39–1  (Cont.) Developer Actions Affecting Metadata Files

<table>
<thead>
<tr>
<th>Action</th>
<th>Files Affected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adding the following component or service on a page:</td>
<td>Updates:</td>
</tr>
<tr>
<td>■ Customizable Components</td>
<td>• app-root/.adf/META-INF/ADF-config.xml with MDS configuration entries and resource catalog service</td>
</tr>
<tr>
<td>■ Instant Messaging and Presence</td>
<td>• app-root/projectname/adfmsrc/ADF-META-INF/adfmsrc.xml with the region binding</td>
</tr>
<tr>
<td>■ Links</td>
<td>• app-root/projectname/adfmsrc/META-INF/adfmsrc.xml with a pointer to the data bindings file</td>
</tr>
<tr>
<td>■ Tags</td>
<td>• app-root/projectname/adfmsrc/ADF-META-INF/adfmsrc/DataBindings.cpx with the task flow binding definition factory and pageDef entries</td>
</tr>
<tr>
<td>■ Oracle Composer</td>
<td>• app-root/projectname/public_html/pageDef/pageDefName/PageDef.xml with the ADF Controller Phase Listener</td>
</tr>
<tr>
<td></td>
<td>• app-root/projectname/public_html/Web-INF/faces-config.xml with the ADF library, external application, and MDS configuration entries</td>
</tr>
<tr>
<td></td>
<td>• app-root/projectname/public_html/Web-INF/web.xml with service libraries and dependent libraries</td>
</tr>
<tr>
<td></td>
<td>Note: The projectname.jpr and web.xml files are updated if the service is dependent on an external application.</td>
</tr>
<tr>
<td></td>
<td>Note: The Tags service has a design time dependency on the Search service, so it also adds Search service libraries.</td>
</tr>
<tr>
<td>Creating a JSR 168 portlet application or adding a portlet to an existing application that uses JSP file</td>
<td>Adds or updates:</td>
</tr>
<tr>
<td></td>
<td>• app-root/projectname/public_html/Web-INF/portlet.xml</td>
</tr>
<tr>
<td></td>
<td>• app-root/projectname/public_html/Web-INF/oracle-portlet.xml</td>
</tr>
<tr>
<td></td>
<td>• app-root/projectname/public_html/Web-INF/web.xml</td>
</tr>
<tr>
<td></td>
<td>Adds Java and JSP source files for the portlet</td>
</tr>
</tbody>
</table>
### Table 39-1  (Cont.) Developer Actions Affecting Metadata Files

<table>
<thead>
<tr>
<th>Action</th>
<th>Files Affected</th>
</tr>
</thead>
</table>
| Creating a JSR 168 portlet application or adding a portlet to an existing application that uses the ADF/JSPX option | A portlet application generated with this option uses the Oracle Portlet Bridge to access a Faces application.  
Add or updates:  
- `app-root/projectname/public_html/WEB-INF/portlet.xml` with configuration information for the portlet bridge  
- `app-root/projectname/public_html/WEB-INF/oracle-portlet.xml` with configuration information for the portlet bridge  
- `app-root/projectname/public_html/WEB-INF/web.xml` with configuration information for a Faces servlet  
- `app-root/src/META-INF/orion-application.xml` with shared library import statements for `adf.oracle.domain` and `oracle.webcenter.skin`  
Add:  
- Java and JSPX source files for the portlet  
- `app-root/projectname/public_html/WEB-INF/faces-config.xml` |
| Creating a PDK portlet producer application | Adds or updates:  
- `app-root/projectname/public_html/WEB-INF/web.xml` with producer servlet definitions and producer deployment details  
- `service_id.properties` with producer configuration attributes  
- `app-root/src/META-INF/orion-application.xml` with shared library import statements for `oracle.portal`  
Add:  
- Java and JSP files for the portlet |
| Registering a secure WSRP producer (WS-Security) | Updates `app-root/.adf/META-INF/connections.xml` with the secure producer connection definition  
**Notes:** The `connections.xml` entry for a secure producer looks different in terms of how the security policy is stored. Instead of the whole security policy description, the security policy is just a URI (that is, a “pointer” to the policy in the policy repository).  
The secure ref attributes in `connections.xml` are stored in the credential store (as defined by `app-root/src/META-INF/jps-config.xml`). The credential store configuration information (as defined by `jps-config.xml`) is stored in `app-root/.adf/META-INF/adf-config.xml`. At design time, the credential store is stored in an Oracle Wallet by default; that is, `META-INF/cwallet.sso`. |
39.3 Implementing Common Requirements One Time

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

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

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

39.4 Portlet Producer Considerations

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

- Section 39.4.1, "Portlet Producer Connections"
- Section 39.4.2, "Portlet Producer Name Clashes"
- Section 39.4.3, "Combining Portlets from Different Portlet Producers"

39.4.1 Portlet Producer Connections

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

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

39.4.2 Portlet Producer Name Clashes

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

39.4.3 Combining Portlets from Different Portlet Producers

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

- Portlet names and JSP paths could clash. Portlet developers should use prearranged class package names and JSP paths to avoid naming clashes when portlets are combined within one portlet producer in one application.

- When you create a JPS portlet in the Portlet wizard, the directory for the portlet modes defaults to \portlet\html\mode_name, where \n is a number that increments for each portlet you create. To avoid directory and file name clashes, portlet developers should change the directory name on the Content Type and Portlet Modes page of the Portlet wizard. Select the portlet mode and then change the directory name in the corresponding field to something unique.

- When you combine portlets into one portlet producer, you must manually merge the portlet descriptor files, avoiding identifier clashes as you do so as follows:
  - JPS portlet identifiers in the portlet.xml and oracle-portlet.xml files are generated automatically starting from portlet1. When you manually merge multiple portlet descriptor files into one, you must change any portlet identifiers that clash with one another.
  - Similarly, the PDK-Java portlet identifiers in provider.xml are automatically generated starting from 1. When you manually merge multiple provider.xml files, you must change any portlet identifiers that clash with one another.

- You must manually merge any web descriptor (web.xml) changes; for example, security role information.

- For PDK-Java portlets, you also might need to manually merge .properties files.

- When using many different portlets, parts of files might not be checked in properly because they are overwritten by someone else or because JDeveloper does not pick up the changed files correctly. For example, you might deploy an application with many types of portlets and see the following error in two of the three OmniPortlets:

  mdsId
  oracle/adf/portlet/OmniPortlet_1172537210873/ap/Portlet100_0b4156e9_0111_1000_8001_82235f273a39.pxml not found
mdsId
oracle/ADF/portlet/OmniPortlet_1172537210873/ap/Portlet100__250498fc_0111_1000_8002_a9fe7286a54e.pxml not found

If you receive error messages like these, then ensure that all *.pxml files from the development environment are copied over to the deployed environment.

- Your portlet customizations might not appear in your deployed environment. Make sure that the content of the oracle\ADF\portlet\<portletInstanceName>.pxml file is correct and that it refers to the proper *.pxml files.
Part X contains the following appendixes:

- Appendix A, "Files for WebCenter Applications"
- Appendix B, "Oracle Composer Component Properties and Files"
- Appendix C, "Resource Catalog Properties and Files"
- Appendix D, "Guidelines for Creating Task Flows to Be Used in Oracle Composer-Enabled Pages"
- Appendix E, "Calling Oracle SES to Search Data"
- Appendix F, "Additional Portlet Configuration"
- Appendix G, "Reuse of Oracle Portal Components"
This appendix provides reference information about the files that are created and modified as you build up your WebCenter application. This appendix includes the following sections:

- Section A.1, "About Files"
- Section A.2, "Files Overview"
- Section A.3, "Files Related to JSR 168 Portlets"
- Section A.4, "Files Related to PDK-Java Portlets"
- Section A.5, "Files Related to Pages"
- Section A.6, "Files Related to Security"
- Section A.7, "Files Related to WebCenter Web 2.0 Services"

For a complete reference for the Oracle Application Development Framework (Oracle ADF) metadata files that you create in your data model and user interface projects, see the Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework.

To learn more how files are affected by major actions, see Section 39.2.2, "Developer Actions Affecting Metadata Files."

A.1 About Files

When you use Oracle WebCenter Framework to build applications and components, several files are created as you perform such actions as building and consuming portlets. As you work with your application, you may find it useful to know a little bit about each of these files and how they relate to your application. You can group the files affected by the Oracle WebCenter Framework into two broad categories as follows:

- Files that are common to any Oracle ADF application, such as web.xml. For these files, it is useful to know what specific additions and changes are made for WebCenter applications. Those modifications are described in this appendix, but for more complete descriptions of these common files, see the Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework.

- Files that are unique to WebCenter 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 WebCenter application. Hence, the easiest way to discuss these files is by object as follows:

- Files Related to JSR 168 Portlets
- Files Related to PDK-Java Portlets
- Files Related to Pages
- Files Related to Security
- Files Related to WebCenter Web 2.0 Services

A.3 Files Related to JSR 168 Portlets
This section includes the following subsections that describe the files that are created for you when you build a JSR 168 portlet:

- Section A.3.1, "portlet.xml"
- Section A.3.2, "oracle-portlet.xml"
- Section A.3.3, "oracle-portlet-tags.jar"
- Section A.3.4, "portlet_mode.jsp"
- Section A.3.5, "portlet_name.java"
- Section A.3.6, "portlet_nameBundle.jar"
- Section A.3.7, "web.xml"
- Section A.3.8, "connections.xml"

A.3.1 portlet.xml
portlet.xml defines the characteristics of your JSR 168 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 168 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>
  </supports>
  <supported-locale>en</supported-locale>
  <resource-bundle>jsr.resource.MapPortletBundle</resource-bundle>
  <portlet-preferences>
    <preference>
      <name>portletTitle</name>
    </preference>
  </portlet-preferences>
</portlet>
For JSR 168 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 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.
- `<supports>` provides information about the portlet modes supported for each content type.
- `<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.
- `<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.3.2 oracle-portlet.xml

`oracle-portlet.xml` is an Oracle extension of `portlet.xml` to support WSRP 2.0 features. For example, a portlet's navigational parameters (a WSRP 2.0 feature) are defined in `oracle-portlet.xml`. In the future, when JSR 286 becomes available, this extension file will no longer be necessary.

The `oracle-portlet.xml` file is generated when:

- The `Enable inter-portlet communication using Oracle WSRP V2 extensions` checkbox is selected in the step 1, General Portlet Information of the Create JSR 168 Java Portlet wizard. For information, see Section 31.2.1, "How to Create a JSR 168 Java Portlet."

- A page or a task flow is converted into a JSF portlet using the Oracle JSF Portlet Bridge. For information, see Chapter 30, "Creating Portlets with the Oracle JSF Portlet Bridge."

This section includes the following subsections:

- Section A.3.2.1, "oracle-portlet.xml Syntax"
Section A.3.2.2, "oracle-portlet.xml Sample With Navigation Parameters"

A.3.2.1 oracle-portlet.xml Syntax

The top-level element of oracle-portlet.xml is <portlet-app-extension>.

```xml
<portlet-app-extension
    xsi:schemaLocation="http://xmlns.oracle.com/portlet/oracle-portlet-app"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://xmlns.oracle.com/portlet/oracle-portlet-app">

The <portlet-app-extension> element must include the child elements shown in Example A–2.

Example A–2 oracle-portlet.xml Element Hierarchy

```xml
<portlet-app-extension>
    <allow-export> ... </allow-export>
    <allow-import> ... </allow-import>
    <portlet-extension>
        <portlet-name> ... </portlet-name>
        <navigation-parameters>
            <name> ... </name>
            <type> ... </type>
            <label> ... </label>
            <hint> ... </hint>
            <usage> ... </usage>
            <aliases> ... </aliases>
        </navigation-parameters>
        <portlet-id> ... </portlet-id>
        <allow-export> ... </allow-export>
        <allow-import> ... </allow-import>
        <strict-authentication>... </strict-authentication>
        <hide-portlet> ... </hide-portlet>
        <require-iframe> ... </require-iframe>
        <minimum-wsrp-version>2</minimum-wsrp-version>
    </portlet-extension>
</portlet-app-extension>
```

The child elements have the following usages:

- <portlet-app-extension> indicates the start and end of the portlet application definition.
- <portlet-extension> indicates the start and end of a portlet definition.
- <portlet-name> identifies the portlet to which the extensions that follow it apply.
- <navigation parameters> defines the parameters for the previously identified portlet.
- <name> is the name of the navigation parameter. This name must be unique within the portlet.
- <type> is the type of the navigation parameter. Currently the only supported type is string.
- <label> is the label for the navigation parameter that end users see on the customization and personalization pages of the portlet.
- <hint> is currently not used.
■ <usage> is currently not used.
■ <aliases> is currently not used.
■ <portlet-id> is the unique identifier of the portlet.
■ <allow-export> is a flag that indicates whether the portlet supports the export of its customization data. This value can be true or false.
■ <allow-import> is a flag that indicates whether the portlet supports the import of its customization data. This value can be true or false.
■ <strict-authentication> is a flag to be set at the producer level. This value can be true or false. If this flag is set to true, it indicates that the portlet container must expose the current user and role membership information supplied by the local server. If false, the user and role membership information suggested by the consumer is used instead.
■ <hide-portlet> is a flag that indicates that the portlet must not appear under the producer connection in the Application Resources panel or Resource Palette. Only <require-iframe> and <minimum-wsrp-version> tags can appear after this tag. If any other tag is added after the <hide-portlet> tag, then deployment may fail.
■ <require-iframe> is used to indicate to the Oracle ADF Faces portlet consumer that the portlet must be rendered in an inline frame (IFRAME).
■ <minimum-wsrp-version> is the minimum WSRP version supported. The minimum version is 2.

A.3.2.2 oracle-portlet.xml Sample With Navigation Parameters

Example A–3 provides a sample of oracle-portlet.xml with two portlets with three navigational parameters each.

Example A–3 oracle-portlet.xml Sample With Navigational Parameters

<portlet-extension>
  <portlet-name>ParameterForm</portlet-name>
  <navigation-parameters>
    <name>ora_wsrp_navigparam_Parameter1</name>
    <type>xsi:string</type>
    <label xml:lang="en">First parameter</label>
    <hint xml:lang="en">First parameter set by portlet</hint>
    <usage/>
    <aliases/>
  </navigation-parameters>
  <navigation-parameters>
    <name>ora_wsrp_navigparam_Parameter2</name>
    <type>xsi:string</type>
    <label xml:lang="en">Second parameter</label>
    <hint xml:lang="en">Second parameter set by portlet</hint>
    <usage/>
    <aliases/>
  </navigation-parameters>
  <navigation-parameters>
    <name>ora_wsrp_navigparam_Parameter3</name>
    <type>xsi:string</type>
    <label xml:lang="en">Third parameter</label>
    <hint xml:lang="en">Third parameter set by portlet</hint>
    <usage/>
    <aliases/>
</portlet-extension>
A.3.3 oracle-portlet-tags.jar

oracle-portlet-tags.jar is the Oracle implementation of the JSP tag library defined by the Java Portlet Specification.

A.3.4 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 168 portlets, you can have the following JSP files for your portlet modes:

- about.jsp
- config.jsp
- edit_defaults.jsp
Files Related to JSR 168 Portlets

- edit.jsp
- help.jsp
- preview.jsp
- print.jsp
- view.jsp

For further explanation of portlet modes, see Section 31.2.5.1, "Guidelines for Portlet Modes."

A.3.5 portlet_name.java

portlet_name.java is the class that acts as the entry point for the portlet logic. This class must implement the javax.portlet.Portlet interface or extend the GenericPortlet abstract class. The portlet container uses this class when it invokes the portlet lifecycle methods.

A.3.6 portlet_nameBundle.jar

portlet_nameBundle.jar is a resource bundle class, containing translation of the strings used by the portlet.

A.3.7 web.xml

web.xml is a Java EE standard descriptor that contains details about Web applications. For more information about web.xml, see the Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework.

A.3.8 connections.xml

connections.xml contains WSRP producer connection information. Example A–4 shows a sample connections.xml file.

Example A–4 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/WSRP_v2_PortletManagement_Service'
          xmlns='http://oracle.com/ws/model'>
          <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'
                <soap
                  address='http://host/portletapp/portlets/WSRP_v2_PortletManagement_
Service
xmlns="http://schemas.xmlsoap.org/wsd1/soap/*"/>
<operation name="setPortletProperties">
<soap
<input name="setPortletProperties"/>
<output name="setPortletPropertiesResponse"/>
</operation>
<operation name="getPortletProperties">
<soap
<input name="getPortletProperties"/>
<output name="getPortletPropertiesResponse"/>
</operation>
<operation name="importPortlets">
<soap
<output name="importPortletsResponse"/>
<input name="importPortlets"/>
</operation>
<operation name="destroyPortlets">
<soap
<input name="destroyPortlets"/>
<output name="destroyPortletsResponse"/>
</operation>
<operation name="exportPortlets">
<soap
<input name="exportPortlets"/>
<output name="exportPortletsResponse"/>
</operation>
<operation name="releaseExport">
<soap
<input name="releaseExport"/>
<output name="releaseExportResponse"/>
</operation>
<operation name="getPortletsLifetime">
<soap
<input name="getPortletsLifetimeResponse"/>
<input name="getPortletsLifetime"/>
</operation>
<operation name="copyPortlets">
<soap
<input name="copyPortletsResponse"/>
<input name="copyPortlets"/>
</operation>
Files Related to JSR 168 Portlets

<operation>
<operation name="performBlockingInteraction">
<soap
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
<input name="performBlockingInteraction"/>
<output name="performBlockingInteractionResponse"/>
</operation>
<operation name="getResource">
<soap
xmlns="http://schemas.xmlsoap.org/wsdl/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/wsdl/soap/"/>
<input name="handleEvents"/>
<output name="handleEventsResponse"/>
</operation>
</port>
<port name="WSRP_v2_Registration_Service"
binding="{urn:oasis:names:tc:wsrp:v2:bind}WSRP_v2_Registration_Binding_SOAP">
<soap
addressUrl="http://host:port/portletapp/portlets/WSRP_v2_Registration_Service"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
<operation name="register">
<soap
soapAction="urn:oasis:names:tc:wsrp:v2:register"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
<input name="register"/>
<output name="registerResponse"/>
</operation>
<operation name="getRegistrationLifetime">
<soap
soapAction="urn:oasis:names:tc:wsrp:v2:getRegistrationLifetime"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
<input name="getRegistrationLifetime"/>
<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/"/>
A.4 Files Related to PDK-Java Portlets

This section includes the following subsections that describe the files that are created for you when you build a PDK-Java portlet:

- Section A.4.2, ".default.properties"
- Section A.4.3, "index.jsp"
- Section A.4.4, "portlet_name_modePage.jsp"
- Section A.4.1, "producer_name.properties"
- Section A.4.5, "provider.xml"
- Section A.4.6, "web.xml"
- Section A.3.8, "connections.xml"
A.4.1 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.4.2, "_default.properties".

A.4.2 _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.4.1, "producer_name.properties".

A.4.3 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.4.4 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 31.2.5.1, "Guidelines for Portlet Modes".

A.4.5 provider.xml

`provider.xml` is the definition file for your PDK-Java producer.

---

**Note:** PDK-Java producers were formerly referred to as providers.
This section includes the following subsections:

- Section A.4.5.1, "provider.xml Syntax"
- Section A.4.5.2, "provider.xml Sample"

**A.4.5.1 provider.xml Syntax**

For more information about the elements and syntax of `provider.xml`, see the Provider Definition Extensible Markup Language (XML) Tag Reference v2 on the Oracle Technology Network.


**A.4.5.2 provider.xml Sample**

Example A–5 provides a sample `provider.xml` file.

**Example A–5  provider.xml Sample**

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<providerDefinition version="3.1">
  <provider class="oracle.portal.provider.v2.DefaultProviderDefinition">
    <localePersonalizationLevel>none</localePersonalizationLevel>
    <session>true</session>
    <defaultLocale>en</defaultLocale>
    <preferenceStore class="oracle.portal.provider.v2.preference.FilePreferenceStore">
      <name>prefStore1</name>
      <useHashing>true</useHashing>
    </preferenceStore>
    <portlet class="oracle.portal.provider.v2.DefaultPortletDefinition">
      <id>1</id>
      <name>SampleRenderer</name>
      <title>SampleRenderer example</title>
      <shortTitle>SampleRenderer</shortTitle>
      <description>Example portlet rendered using the SampleRenderer</description>
      <timeout>40</timeout>
      <timeoutMessage>SampleRenderer example timed out</timeoutMessage>
      <acceptContentType>text/html</acceptContentType>
      <showEdit>true</showEdit>
      <showEditToPublic>false</showEditToPublic>
      <showEditDefault>true</showEditDefault>
      <showPreview>true</showPreview>
      <showDetails>true</showDetails>
      <hasHelp>true</hasHelp>
      <hasAbout>true</hasAbout>
      <renderer class="oracle.portal.sample.v2.devguide.samplerenderer.SampleRenderer"/>
      <personalizationManager class="oracle.portal.provider.v2.personalize.PrefStorePersonalizationManager">
        <dataClass>oracle.portal.provider.v2.personalize.NameValuePersonalizationObject</dataClass>
      </personalizationManager>
    </portlet>
  </provider>
</providerDefinition>
```
A.4.6 web.xml

web.xml is a Java EE standard descriptor that contains details about Web applications. For more information about web.xml, see the Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework.

A.4.7 connections.xml

connections.xml contains PDK-Java producer connection information. Example A–4 shows a sample connections.xml file.

A.5 Files Related to Pages

This section includes the following subsections that describe the files that are created or modified when you create or modify pages:

- Created at page design time:
  - Section A.5.1, "adf-config.xml"
  - Section A.5.2, "DataBindings.cpx"
  - Section A.5.3, "faces-config.xml"
  - Section A.5.4, "page_name.jspx"
  - Section A.5.5, "PageDef.xml"
  - Section A.5.6, "web.xml"

- Created at application deployment time:
  - Section A.5.7, "mds Subdirectory"

- Created upon consumption of JSR 168 portlets
  - Section A.5.8, "wsdl Subdirectory"

A.5.1 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 Services, including pages, portlets, and all the services such as search, worklist, forums, and so on, to set configuration information. This file, with connections.xml, contains the configuration information required for most of the services to function.

Table A–1 describes the child elements of <adf-portlet-config>. The init-param names in the first column correspond to the names of the servlet init-params used when the portlet client is accessed through the Web adapter.

<table>
<thead>
<tr>
<th>Element (init-param)</th>
<th>Description</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>parallelPoolSize (parallel.pool.size)</td>
<td>The number of threads to use for parallel execution of tasks.</td>
<td>10</td>
</tr>
<tr>
<td>parallelQueueSize (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>
</tbody>
</table>
Table A–1 (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>defaultTimeout (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>minimumTimeout (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>maximumTimeout (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>60</td>
</tr>
<tr>
<td>resourceProxyPath (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>supportedLocales (supported.locales)</td>
<td>The set of supported locales defined using strings of the form: language[_country[_variant]] Commented out by default. You should uncomment it if you have multiple locales. See Example A–6.</td>
<td></td>
</tr>
<tr>
<td>portletTechnologies (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>{o.p.c.i.web.WebPortletTechnologyConfig, o.p.c.i.wsrp.WSRPPortletTechnologyConfig}</td>
</tr>
<tr>
<td>cacheSettings (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. The cache is enabled and no size restrictions are imposed.</td>
<td></td>
</tr>
</tbody>
</table>

<adf-portlet-config> element Sample

Example A–6 illustrates the usage of the <adf-portlet-config> element.

Example A–6  <adf-portlet-config> element Sample

<adf-portlet-config xmlns="http://xmlns.oracle.com/adf/portlet/config">
  <supportedLocales>
    <value>en</value>
    <value>fr</value>
    <value>de</value>
    <value>es</value>
  </supportedLocales>
  <portletTechnologies>
    <value>oracle.portlet.client.containerimpl.web.WebPortletTechnologyConfig</value>
    <value>oracle.portlet.client.containerimpl.wsrp.WSRPPortletTechnologyConfig</value>
  </portletTechnologies>
</adf-portlet-config>
Files Related to Pages

A.5.2 DataBindings.cpx

DataBindings.cpx is a file common to Web applications. For more information about DataBindings.cpx, see the Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework.

A.5.3 faces-config.xml

faces-config.xml is a file common to JSF applications. It describes the page flow of your application. For more information about faces-config.xml, see the Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework.

A.5.4 page_name.jsp

page_name.jsp 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.5.5 PageDef.xml

PageDef.xml is a file common to Oracle ADF applications. This file holds information about portlet bindings. Also, portlet parameters can be tied to page variables in this file.

For more information about PageDef.xml, see the Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework.

PageDef.xml Sample

Example A–7 provides a sample PageDef.xml file.

Example A–7 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'
```
Files Related to Pages

Files for WebCenter Applications

A-17
A.5.6  web.xml

web.xml is a Java EE standard descriptor that contains details about Web applications. For more information about web.xml, see the Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework.

A.5.7  mds Subdirectory

When you deploy a WebCenter 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.5.8 wsd1 Subdirectory
When you consume JSR 168 portlets in an application, a wsd1 subdirectory is created in the WEB-INF directory. The files in this subdirectory are internal files created for portlets from a WSRP producer.

A.6 Files Related to Security
This section includes the following subsections that describe the files that are created or modified when you implement or modify security:

- Section A.6.1, "jazn-data.xml"
- Section A.6.2, "cwallet.sso"
- Section A.6.3, "jps-config.xml"
- Section A.6.4, "adf-config.xml"

A.6.1 jazn-data.xml
When you implement or modify security for your application, the jazn-data.xml file is created or modified.

The jazn-data.xml file is used to facilitate the deployment of realm and policy information for your application. In your development environment (JDeveloper), jazn-data.xml is located in your application's workspace/src/META-INF directory. After deployment, the contents of this file are merged in domain-level identity or policy stores. This file contains the policies that are created at design time, and is merged at deployment by JpsApplicationLifecycleListener into the system-jazn-data.xml file (or appropriately configured policy store) for the deployment target.

When you migrate security information with JpsApplicationLifecycleListener, the jazn-data.xml file can be used as the source file for the migration. For more information about migrating roles, see the Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework.

jazn-data.xml Sample
Example A–8 provides a sample jazn-data.xml file.

Example A–8 jazn-data.xml Sample
<?xml version = '1.0' encoding = 'UTF-8'?>
<jazn-data>
  <jazn-realm>
    <realm>
      <name>jazn.com</name>
    </realm>
  </jazn-realm>
  <policy-store>
    <applications>
      <application>
        <name>mdsapp-secure</name>
        <app-roles>
          <app-role>
            <name>unauthenticated-role</name>
            <display-name>anonymous-role</display-name>
          </app-role>
          ...
        </app-roles>
      </application>
      ...
    </applications>
  </policy-store>
</jazn-data>
<class>oracle.security.jps.service.policystore.ApplicationRole</class>
</app-role>

<app-role>
  <name>authenticated-role</name>
  <display-name>authenticated-users</display-name>
</app-role>

<app-role>
  <name>approle</name>
  <display-name>approle</display-name>
  <description>approle</description>
  <guid/>
  <class>oracle.security.jps.service.policystore.ApplicationRole</class>
  <members>
    <member>
      <class>oracle.security.jps.internal.core.principals.JpsXmlUserImpl</class>
      <name>orcladmin</name>
    </member>
  </members>
</app-role>

<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>
  <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>
A.6.2 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 workspacerooot/src/META-INF directory.

A.6.3 jps-config.xml

jps-config.xml is located at workspacerooot/src/META-INF. The jps-config.xml file packaged with the application is used only at design time. At runtime, the configurations stored in the domain-level jps-config.xml file become functional.

jps-config.xml configures OPSS services. For more information on the element hierarchy and attributes of jps-config.xml, see the appendix "OPSS Configuration File Reference" in the Oracle Fusion Middleware Security Guide.

The recommendation for migration is to create a new jps-config.xml file with the source and destination contexts defined as required by the migration commands. For details on migrating identities, see the section "Migrating Identities Manually" in the Oracle Fusion Middleware Security Guide.

The policy store is migrated in the following ways: entire policy store, application-specific policies, and global policies. For details, see the section "Migrating Policies Manually" in the Oracle Fusion Middleware Security Guide.

The credential store is migrated in the following ways: entire credential store or application-specific credentials. When migrating application-specific credentials, ensure that srcFolder and targetFolder are the same. srcFolder and targetFolder can be determined from appUID in the adf-config.xml file, see Example A–9. For details, see the section "Migrating Credentials Manually" in the Oracle Fusion Middleware Security Guide.

Example A–9 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.6.4 adf-config.xml

Security configurations are stored in adf-config.xml. This file is located at workspacerooot/.adf/META-INF/adf-config.xml. See Example A–10.

Example A–10 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"
  />
</adf-security-child>
```
A.7 Files Related to WebCenter Web 2.0 Services

This section includes sample connections.xml and adf-config.xml files for some WebCenter Web 2.0 Services.

Example A–11 shows the minimum required configuration for the SES Web service connection.

Example A–11  SES Web Service Sample Connection

<Reference name="GenericSesConnection"
className="oracle.adf.mbean.share.connection.webcenter.search.SesConnectionReferencable"
credentialStoreKey="GenericSesConnection" xmlns="">
  <Factory
className="oracle.adf.mbean.share.connection.webcenter.search.SesConnectionFactory"/>
  <RefAddresses>
    <XmlRefAddr addrType="SesConfig">
      <Contents>
        <SesConfig>
          <AttributeMap>
            <Attribute name="date">
              <SesAttribute>LastModifiedDate</SesAttribute>
            </Attribute>
            <Attribute name="person">
              <SesAttribute>Author</SesAttribute>
            </Attribute>
          </AttributeMap>
          <SoapUrl>http://host:port/search/query/OracleSearch</SoapUrl>
        </SesConfig>
      </Contents>
    </XmlRefAddr>
    <StringRefAddr addrType="appUsername">
      <Contents>wpadmin</Contents>
    </StringRefAddr>
    <SecureRefAddr addrType="appPassword"/>
  </RefAddresses>
</Reference>

Example A–12 shows a sample adf-config.xml for the Documents service.

Example A–12  ADF Configuration for the Documents Service

<doclibC:adf-doclib-config xmlns="http://xmlns.oracle.com/webcenter/doclib/config"
primaryConnectionName="Oracle Content Server">
  <doclibC:spaces-repository
    spacesRoot="/repository path"
  ></doclibC:spaces-repository>
</doclibC:adf-doclib-config>

Note: The credentialStoreLocation entry in the sample is valid at design time only.
Example A–13 shows a sample adf-config.xml for the Worklist service.

**Example A–13  ADF Configuration for the Worklist Service**

```xml
<worklistC:worklist-config>
  <worklistC:ConnectionName>WebCenter-Worklist</worklistC:ConnectionName>
</worklistC:worklist-config>
```
Oracle Composer Component Properties and Files

This appendix provides reference information about Oracle Composer-specific tags, configuration files, and style properties. This information is useful while performing the tasks described in Chapter 7, "Enabling Runtime Editing of Pages Using Oracle Composer" and Chapter 8, "Extending Runtime Editing Capabilities Using Oracle Composer."

This appendix contains the following sections:

- Section B.1, "Oracle Composer Component Properties"
- Section B.2, "Oracle Composer-Specific Files and Configurations"
- Section B.3, "Oracle Composer Default Add-Ons and Property Panels"
- Section B.4, "Oracle Composer Help Topic IDs"
- Section B.5, "Oracle Composer Components Style-Specific Properties"
- Section B.6, "Customizable Components (HTML) Properties"

B.1 Oracle Composer Component Properties

This section is a quick reference for all Oracle Composer tags. When you add Oracle 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 7.1, "Designing Editable Pages Using Oracle Composer Components."

Note: The Property Inspector in JDeveloper displays certain extended metadata attributes under the Customization Attributes and Annotations Attributes categories. You can use these properties to provide additional metadata information. For more information, see the section "Extended Metadata and Annotation Properties" in Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework.

This section contains the following subsections:

- Section B.1.1, "Page Customizable"
- Section B.1.2, "Change Mode Link and Change Mode Button"
B.1.1 Page Customizable

Use the Page Customizable component to enable page editing at runtime. The Page Customizable component denotes the customizable part of a page and shows the Oracle Composer toolbar in Edit mode at runtime. Components enclosed within a Page Customizable component can be customized and edited.

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

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.

The Page Customizable component stretches its first child component to fill the space available to it. It is recommended that the Page Customizable component contain only one child component.

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>
</tbody>
</table>

rendered | Boolean | Yes | Specifies whether the component is rendered. When set to false, no output is delivered for this component (the component is not in any way rendered, and cannot be made visible on the client). The default value is true. |

Table B–1 Attributes of a Page Customizable Component
**Oracle Composer Component Properties**

**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>sourceViewPosition</td>
<td>String</td>
<td>Yes</td>
<td>The position of the Source view pane displaying components in a tree structure. Valid values are bottom, end, start, and top. Default value is top. Alternatively, the Source Position submenu on the View menu on the Oracle Composer toolbar provides options to set this attribute at runtime. For more information, see the section, Section 5.5, &quot;Editing Capabilities in Source View in Page Edit Mode.&quot;.</td>
</tr>
<tr>
<td>sourceViewSize</td>
<td>String</td>
<td>Yes</td>
<td>The height or width of the Source view pane in pixels. Use this to specify height if the sourceViewPosition is set to top or bottom, or to specify width if sourceViewPosition is set to start or end. Since the default for sourceViewPosition is top, the pane has a default height of 200 pixels. Alternatively, the resize handler on the edge of the Source view pane enables users to alter the height or width of the pane at runtime.</td>
</tr>
<tr>
<td>toolbarLayout</td>
<td>String</td>
<td>Yes</td>
<td>The elements to be displayed on the Oracle 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 8.9, &quot;Customizing the Oracle Composer Toolbar.&quot;</td>
</tr>
<tr>
<td>view</td>
<td>String</td>
<td>Yes</td>
<td>The default view in which to open a page in Oracle Composer. Available options are Design and Source. Default value is Design. For more information, see Section 9.7, &quot;Configuring an Application Page to Display in Source View by Default.&quot;</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>
<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>
The Page Customizable component provides an editor facet, which is prepopulated with a Page Editor Panel component. To ensure that Oracle 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 ('_').</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></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 Attributes</strong></td>
<td></td>
<td></td>
<td></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.

**Disabling Runtime Editing in Your Application Pages**

To disable runtime editing of pages, you can revoke Edit privileges for users, or delete the Change Mode Link or Change Mode Button component from the page so that the user cannot navigate to the Edit mode of the page.

**B.1.3 Layout Customizable Component**

The Layout Customizable component applies a predefined layout to its child components. It is a container component that enables end users to lay out its child components in several predefined ways. Use the Layout Customizable component only to enable changing of the page layout at runtime. If you just want container components in which to layout components, but do not want to enable runtime layout changes, then it is recommended that you use ADF Faces layout components like Panel Stretch Layout and Panel Group Layout. It is recommended that you have just one Layout Customizable on the page.

A Layout Customizable component contains a direct child Panel Customizable component and two facets with Panel Customizable components by default to enable users to add content inside them at runtime. However, you can add any components inside the Layout Customizable component and its facets. The content in the three Panel Customizable components can be placed in different ways using predefined layouts. For more information, see the section, "Predefined Layout Types".
Table B–3 describes the attributes of a Layout Customizable component.

**Table B–3  Attributes of the Layout Customizable Component**

<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).</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>The rendering of a component can also be defined at runtime using the Show Component and Hide Component options.</td>
</tr>
<tr>
<td>type</td>
<td>String</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Default value is threeColumn.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>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 11, “Modifying Default Security Behavior of Oracle Composer Components.”</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>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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>For example: text=&quot;Change Layout&quot;</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>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. If the same access key appears in multiple input fields in the same page of output, the rendering user agent cycles among the elements accessed by the similar keys. Note that user agents are inconsistent about dealing with two links having same access key, and so the cycling behavior is dependent on what the user agent provides. This attribute is sometimes referred to as the &quot;mnemonic&quot;. The character specified by this attribute must exist in the Text attribute of this button instance. If it does not, the user receives no visual indication of the existence of the accessKey. The easiest, and most convenient way to specify both the text and the mnemonic together is to use textAndAccessKey. Note that the accessKey is triggered by browser-specific and platform-specific modifier keys. It even has browser-specific meaning. For example, Internet Explorer 7.0 sets focus when you press Alt+&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>
</tbody>
</table>
| showLayoutChanger      | Boolean    | Yes          | Specifies whether the Change Layout icon or text must be displayed. The default value is true. When set to true, the runtime behavior of the layout changer is as follows:  
- Displayed in View mode.  
- Displayed in Edit mode.  
  It is always displayed in Edit mode, even if the value of the showLayoutChanger attribute is false.  
- If there is a security or MDS restriction on the Layout Customizable component, then the layout changer is not displayed in View or Edit modes. |
**Table B–3 (Cont.) Attributes of the Layout Customizable Component**

<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>

**Behavior Attributes**

<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>

**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 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 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 in the two facets. When you select the layout type, the components in these Panel Customizable 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>
<thead>
<tr>
<th>Layout Type</th>
<th>Image Showing Content Arrangement</th>
</tr>
</thead>
<tbody>
<tr>
<td>oneColumn</td>
<td>![oneColumn Image]</td>
</tr>
<tr>
<td>threeColumn</td>
<td>![threeColumn Image]</td>
</tr>
<tr>
<td>threeColumnNarrow</td>
<td>![threeColumnNarrow Image]</td>
</tr>
<tr>
<td>twoColumn</td>
<td>![twoColumn Image]</td>
</tr>
<tr>
<td>twoColumnBottom</td>
<td>![twoColumnBottom Image]</td>
</tr>
<tr>
<td>twoColumnNarrowLeft</td>
<td>![twoColumnNarrowLeft Image]</td>
</tr>
<tr>
<td>twoColumnNarrowRight</td>
<td>![twoColumnNarrowRight Image]</td>
</tr>
<tr>
<td>twoColumnTop</td>
<td>![twoColumnTop Image]</td>
</tr>
</tbody>
</table>

*Note:* In a oneColumn layout, A and B are rendered only if they contain child components.

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 Oracle Composer's 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>Table B–6</th>
<th>Attributes of a Panel Customizable Component</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute</td>
<td>Type</td>
</tr>
<tr>
<td>-----------</td>
<td>-----------------</td>
</tr>
<tr>
<td>id</td>
<td>String</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>rendered</td>
<td>Boolean</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, and stretch. 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. |

<table>
<thead>
<tr>
<th>Style Attributes</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>
</tbody>
</table>
### Inline Style

**inlineStyle** | **String** | **Yes** | The CSS styles to use for this component. This is intended for basic style changes. The inlineStyle is a set of CSS styles that are applied to the root DOM element of the component. If the inlineStyle attribute's CSS properties do not affect the DOM element you want affected, then you must create a skin and use the skinning keys which are meant to target particular DOM elements, like ::label or ::icon-style.

### Behavior Attributes

**allowAction** | **Boolean** | **Yes** | Determines whether this component allows addition of child components, dragging and dropping Show Detail Frame components inside and from it, and rearranging child components at runtime. It also determines whether the Add Box icons (that enable users to add Box components adjacent to the component) must be rendered on the component.

Available values are none and all. The default value is all.

**Note:** You can also set this attribute while defining component security in the application's adf-config.xml file. For more information, see Section 11.5, "Applying Action-Level Restrictions on Panel Customizable and Show Detail Component Actions."
Supported Facets
The Panel Customizable component supports a separator facet, which can be used to render content once between each of its child components.

Disabling the Edit Option on a Panel Customizable Component
When you switch to the Edit mode of a page at runtime, in Design view you will notice an Edit icon on all Panel Customizable or Box components that can be edited. To disable property editing on a specific Panel Customizable or Box component in Design view, you must set the showEditAction attribute on the component to false at design time. The Edit icon is no longer displayed on the component in Design view. However, users can still select such a component in Source view and edit its properties.

B.1.5 Show Detail Frame Component
Add Show Detail Frame components inside Panel Customizable components to enable runtime customizations like move, minimize, restore, and delete of child components. You can move Show Detail Frame components only if they are added inside a Panel Customizable component.

Geometry Management
A Show Detail Frame component can be stretched by a parent layout component that stretches its children, for example, Panel Customizable.

A Show Detail Frame component stretches its child component to fill the height and width available to it. It is recommended that you add only one child component inside a Show Detail Frame component. However, if you have more than one child component, the first child component is stretched to the height and width of the Show Detail Frame's content area and subsequent child components are ignored.

### 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>showEditIcon</td>
<td>Boolean</td>
<td>Yes</td>
<td>While in Edit mode, in Design view, renders Edit and Delete icons 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 an Oracle 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. |
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>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>id</strong></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><strong>rendered</strong></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><strong>text</strong></td>
<td>String</td>
<td>Yes</td>
<td>A title for the Show Detail Frame component.</td>
</tr>
<tr>
<td><strong>icon</strong></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: icon=&quot;<a href="http://source-pc/images/accessability.gif">http://source-pc/images/accessability.gif</a>&quot;</td>
</tr>
</tbody>
</table>

Note: An image that is stored at the document root does not require a full path. For example: icon="detail.gif"
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>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. Available values are light, medium, and dark. The default value is medium.</td>
</tr>
<tr>
<td>displayHeader</td>
<td>Boolean</td>
<td>Yes</td>
<td>Indicates whether the header of the Show Detail Frame is displayed. The default value is true. If you choose to set displayHeader to false and if you have exposed some actions on the component, then a toolbar is displayed when you move the mouse over the component area. The toolbar contains a drop down icon, which displays a menu of available options. This toolbar displays only in Edit mode, if there are actions available on the component.</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. The default value is false.</td>
</tr>
<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>
</tbody>
</table>

Actions Attributes

| displayActions | String   | Yes          | Specifies when seeded interactions are displayed. Available options are onHover and always. The default is always.                                                                                                                                 |

Oracle Composer Component Properties
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>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 9.3.2, &quot;How to Define 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 displayed in the Actions menu. Available values are menu and none. The default value is menu.</td>
</tr>
<tr>
<td>showRemoveAction</td>
<td>String</td>
<td>Yes</td>
<td>Renders a Remove icon on the Show Detail Frame header. Clicking this icon removes the component from the page. Available values are chrome and none. The default value is none. Note: In page Edit mode, the Remove icon appears regardless of this setting, provided other restrictions have not been applied.</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 none and always. The default value is always.</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 chrome and none. The default is chrome.</td>
</tr>
<tr>
<td>showEditAction</td>
<td>Boolean</td>
<td>Yes</td>
<td>While in Edit mode, in Design view, renders an Edit icon on the Show Detail Frame header that enables you to edit component properties at runtime. Default value is true. This is relevant only in an Oracle 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 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>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>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.</td>
</tr>
</tbody>
</table>

**Th CSS styles in this attribute are commonly used to define the height and background color of the Show Detail Frame. For example, height:200px; background-color:green;**

**Note:** If the Show Detail Frame component is nested inside a Panel Customizable component with a horizontal layout, you must specify a width for the Show Detail Frame component to ensure that it is rendered in page View mode.

### Behavior Attributes
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>partialTriggers</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 - &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.</td>
</tr>
<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>
<tr>
<td>stretchContent</td>
<td>Boolean</td>
<td>Yes</td>
<td>Specifies whether the Show Detail Frame stretches its child component. 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.</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>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>attributeChangeLister</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>
<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 an Oracle 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>
<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>

Other Attributes

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 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:

```xml
<cust:showDetailFrame text="showDetailFrame 1" id="sdf1">
   <f:attribute name="sdf_selection_rule" value="sdf_for_edit_mode_only"/>
</cust:showDetailFrame>
```

Disabling the Edit Option on a Show Detail Frame Component

When you switch to the Edit mode of a page at runtime, in Design view you will notice an Edit icon on all Show Detail Frame components that can be edited. To disable property editing on a specific Show Detail Frame or Movable Box component in Design view, then you can do this by setting the showEditAction attribute on the component to false at design time. The Edit icon is no longer displayed on the component when you enter the Edit mode of the page at runtime. However, users can still select such a component in Source view and edit its properties.

B.1.6 Custom Action

Use Custom Action components to trigger navigational flow in a task flow when a region is included inside a Show Detail Frame component. You can define custom actions corresponding to the ADFc outcomes of the task flows. At runtime, these custom actions are displayed on the Show Detail Frame header as icons, menu options, or both. A Custom Action must always be defined as a child component of a Show Detail Frame and is useful only when the Show Detail Frame also contains a task flow as its child component.

You can add as many Custom Action components as there are ADFc outcomes in the task flow. If you add a Custom Action attribute without a corresponding ADFc outcome, or if the action attribute value does not match an ADFc outcome defined on the view that is currently being displayed by the task flow, then that action is not displayed at runtime.
Table B–9 describes the 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>action</td>
<td>String</td>
<td>Yes</td>
<td>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.</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>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>Represents the text of the menu item link.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>This is applicable for a custom action rendered as a menu item link.</td>
</tr>
</tbody>
</table>

Note: You can also define custom actions for Show Detail Frame components at the global level. For more information, see Section 9.3, “Enabling Custom Actions on a Show Detail Frame Enclosing a Task Flow.”
### 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/com/hp/oralogo_small.gif"  
- relative - a path located relatively to the source page, like "bullet.jpg"  
- context relative - a path relatively based on the web application's context root, like "/images/error.png"  
- server relative - a path relatively based on the web server by application name, like "//adf-richclient-demo-context-root/images/error.png"  
As icon does not allow alternative text to be provided for the image, in order to create an accessible product an icon must only be used when its use is purely decorative. |
| shortDesc   | String    | Yes         | Provides a short description of the component. This text is commonly used by user agents to display tooltip help text, in which case the behavior for the tooltip is controlled by the user agent, for example, Firefox 2 truncates long tooltips. For form components, the shortDesc is displayed in a note window. This is applicable only for a custom action rendered as a toolbar icon. |
| location    | String    | Yes         | Determines whether the custom action is rendered as a toolbar icon or a menu item link. Available values are chrome, menu, or both. Default value is chrome. |

Other Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actionComponent</td>
<td>String</td>
<td>Yes</td>
<td>Specifies the ID of the command component that must be queued for the action event. When the actionComponent attribute is specified, the Show Detail Frame component queues the action event on this component. For more information, see Section 9.3.3, &quot;How to Configure Custom Actions that Display Task Flow Views in a Separate Browser Window.&quot;</td>
</tr>
</tbody>
</table>
B.1.7 ImageLink

Use the Image Link component to include an image with a hyperlink on the page. The Image Link component makes the runtime customization experience easier by adding an image and a link in one step.

The Go Image Link component is an alternative to the Image Link component. For information about the Go Image Link component, see Oracle Fusion Middleware Tag Reference for Oracle ADF Faces.

Table B–10 describes the attributes of an Image Link 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>
<tr>
<td>source</td>
<td>String</td>
<td>Yes</td>
<td>Displays the location of the image.</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>destination</td>
<td>String</td>
<td>Yes</td>
<td>Specifies the URL this image component references.</td>
</tr>
<tr>
<td>targetFrame</td>
<td>String</td>
<td>Yes</td>
<td>Specifies the target frame for this component. You can specify values like _blank, _self, or any other value allowed by HTML, including the names of frames.</td>
</tr>
</tbody>
</table>

Link Attributes

Appearance Attributes
Table B–10  (Cont.) Attributes of an Image Link 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>String</td>
<td>Yes</td>
<td>Specifies a keyboard shortcut to gain quick access to this component.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Note: The key specified in this attribute must exist in the text attribute for this component instance.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>You can specify browser-specific access keys as follows:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ For Internet Explorer, specify</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Alt+&lt;key&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ For Firefox, specify</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Ctrl+Alt+&lt;key&gt;</td>
</tr>
<tr>
<td>longDescURL</td>
<td>String</td>
<td>Yes</td>
<td>Specifies the URL to a document that contains a long description of the image.</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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>This is intended for basic style changes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>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>disabled</td>
<td>Boolean</td>
<td>Yes</td>
<td>Specifies whether the Image Link component should be considered disabled.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The default value is false.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The disabled attribute is of significance only when the Image Link component is used in container components like global headers and tab bars.</td>
</tr>
</tbody>
</table>
Oracle Composer-Specific Files and Configurations

This section describes the various files that you can modify to extend Oracle Composer capabilities. While some files, for example the Oracle Composer extension file, `pe_ext.xml`, are entirely relevant to Oracle Composer only, others are ADF configuration files that can take entries for Oracle 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, "web.xml"

### Table B–10 (Cont.) Attributes of an Image Link Component

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Supports EL?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>partialTriggers</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 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.</td>
</tr>
<tr>
<td>binding</td>
<td>oracle.adf.view.page.editor.component.ImageLink</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>visible</td>
<td>Boolean</td>
<td>Yes</td>
<td>Specifies whether the component must be visible on the client. The default value is true. If Rendered was set to false, then the component is not rendered and cannot be made visible on the client even if visible is set to true.</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 events might include the width of a column that supported client-side resizing.</td>
</tr>
</tbody>
</table>
B.2.1 pe_ext.xml

You can create a pe_ext.xml file and add elements in it to register new Oracle 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 ViewController project; for example, in the APPLICATION_HOME\ViewController\src\META-INF directory. This section describes the different elements of the Oracle Composer extension file schema and explain when you may want to use the schema elements.

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, "event-handlers"
- Section B.2.1.4, "drop-handlers"
- Section B.2.1.5, "filter-config"
- Section B.2.1.6, "elbuilder-config"
- Section B.2.1.7, "Sample pe_ext.xml File"

B.2.1.1 addon-config

Use the addon-config element to include entries for new Oracle 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. Oracle 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 Oracle 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 Oracle 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.
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:

```xml
<addon-config>
    <panels>
        <panel name="oracle.fod.custom.panel" title="About FOD"
            icon="adf/webcenter/images/about.gif"
            taskflow-id="/WEB-INF/about-fod.xml#about-fod"
        />
    </panels>
</addon-config>
```

For more information about add-ons, see Section 8.2, “Creating Oracle Composer Add-Ons.”

### 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 8.3, “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 Oracle Composer. Within the property-panels element you can insert any number of individual property-panel elements to define custom property panels to display as tabs in the Component Properties dialog. To specify the component or task flow for which you want to register a custom property panel, you must add a component or taskflow-id element respectively within the property-panel.

Table B–12 describes the property-panel element attributes.

### Table B–11 panel Element Attributes

<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 adf-config.xml, or a property panel in the property-panels 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 Oracle 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>
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 <code>addon-config</code> 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 event-handlers
Use the event-handlers element to register handlers for events generated by Oracle Composer. Add the event-handlers element directly under the pe-extension tag. You can have only one event-handlers element in an extension file.

event-handler
Use the event-handler element to register an event handler in Oracle 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 Oracle 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 8.5, "Configuring Event Handlers for Oracle Composer UI Events."

B.2.1.4 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 Oracle 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 Oracle Composer by adding a drop-handler entry in the pe_ext.xml file as follows:

```
<drop-handlers>
  <drop-handler>test.TestDropHandler</drop-handler>
</drop-handlers>
```

For more information, see Section 8.6, "Configuring Drop Handlers in the Oracle Composer Catalog."

B.2.1.5 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 8.7, "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 attribute 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.</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="accessKey" />
    <attribute name="attributeChangeListener" />
  </global-attribute-filter>
</filter-config>
```
B.2.1.6  elbuilder-config

Use the elbuilder-config element to configure custom values for the expression builder. The expression builder invoked from Oracle 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 8.4, "Extending the Expression Builder."

Add the elbuilder-config element directly under the pe-extension tag. You can have only one elbuilder-config element in an extension file.

selector

Use selector elements to define and include custom options, as shown in the following example:

```
<elbuilder-config>
  <!-- define selector -->
  <selector id="CustomELParameter">
    view.CustomELParameter
  </selector>

  <!-- include selector -->
  <selectors>
    <selector id="CustomELParameter" />
  </selectors>
</elbuilder-config>
```

You can include many selector elements inside a selectors element.

B.2.1.7  Sample pe_ext.xml File

Example B–1 shows a sample pe_ext.xml file with different elements used to extend Oracle Composer capabilities.

```
Example B–1  Sample pe_ext.xml File

<pe-extension xmlns="http://xmlns.oracle.com/adf/pageeditor/extension">

  <addon-config>
    <!-- Oracle 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>

    <!-- Oracle Composer property panels configuration -->
    <property-panels>
      <property-panel name="cmdbtn">
```

```
<component>oracle.rich.CommandButton</component>
<panel name="prop.panel.cmdbtn" />
</property-panel>
</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 Oracle Composer. -->
<filter-config>
  <global-attribute-filter>
    <attribute name="accessKey" />
    <attribute name="attributeChangeListener" />
    <attribute name="autoSubmit" />
    <attribute name="binding" />
  </global-attribute-filter>
  <taglib-filter namespace="http://xmlns.oracle.com/adf/faces/rich">
    <tag name="commandButton">
      <attribute name="text" />
      <attribute name="icon" />
    </tag>
  </taglib-filter>
</filter-config>
</pe-extension>

B.2.2 adf-config.xml

The adf-config.xml file specifies application-level settings that are usually determined at deployment and are often changed at runtime. This file is created when you create your application and is located in the ADF META-INF folder under Descriptors in the Application Resources panel. The following example shows the minimal adf-config.xml file created when you create a WebCenter application:

<?xml version="1.0" encoding="windows-1252" ?>
  <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 Oracle Composer sandbox, creating custom Resource Catalogs, and disabling Source view. The following sections describe the various adf-config.xml elements used for Oracle Composer-specific configurations.

This section contains the following subsections:
Oracle Composer-Specific Files and Configurations

- 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, "rcv-config"
- Section B.2.2.6, "customizableComponentsSecurity"
- Section B.2.2.7, "mds-config"
- Section B.2.2.8, "resource-string-editor"
- Section B.2.2.9, "enable-source-view"
- Section B.2.2.10, "persistent-change-manager"
- Section B.2.2.11, "taglib-config"
- Section B.2.2.12, "Sample adf-config.xml File"

### B.2.2.1 page-editor-config

Use the `page-editor-config` element to include Oracle Composer-specific configurations such as add-on panel registration, sandbox configuration, resource string editor configuration, and so on. Add the `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">
  ...
  <page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">
  
  </page-editor-config>
</adf-config>
```

### B.2.2.2 addon-panels

Use the `addon-panels` element to register custom add-ons and property panels in Oracle Composer. Insert the `addon-panels` element within the `page-editor-config` element. You can have only one `addon-panels` element in the file. Within this you can have any number of `addon-panel` elements to register custom panels. For more information about Oracle Composer add-ons and property panels, see Section 8.2, "Creating Oracle Composer Add-Ons" and Section 8.3, "Creating Custom Property Panels."

**addon-panel**

Use an `addon-panel` element to register an add-on in the `adf-config.xml` file. An `addon-panel` element must only be included inside the `addon-panels` element. You can have any number of `addon-panel` elements to register custom panels.

The following example shows how you can use `addon-panels` and `addon-panel` elements to register a new add-on:

```xml
<page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">
  <addon-panels>
    <!-- Page Properties add-on -->
    <addon-panel name="oracle.adf.pageeditor.addonpanels.page-settings" />

    <!-- Page Reset add-on -->
  </addon-panels>
</page-editor-config>
```
The `addon-panel` element also supports the `rendered` attribute. This attribute can take an EL value and specifies whether the panel must be rendered in Oracle 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 Oracle Composer.

### B.2.2.3 sandbox-namespaces

Use the `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 10.2.1, "How to Enable Oracle Composer Sandbox Creation."

Insert the `sandbox-namespaces` element within the `adf-config` element. The following example shows how to use the `sandbox-namespaces` element:

```xml
<page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">
    . . .
    <sandbox-namespaces>
        <namespace path="/pages"/>
        <namespace path="/pageDefs"/>
    </sandbox-namespaces>
    . . .
</page-editor-config>
```

### B.2.2.4 session-options-factory

When creating customization layers in your application, use the `session-options-factory` element to register the `ComposerSessionOptionsFactory` implementation with Oracle Composer. This class contains the logic to save customizations in different layers based on the specified criteria. For more information, see Section 10.3, "Adding Customization Layers to View and Edit Modes: Example."

The following example shows how to register a class named `AppsSessionOptionsFactoryImpl`:

```xml
<page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">
    . . .
    <session-options-factory>view.AppsSessionOptionsFactoryImpl</session-options-factory>
    . . .
</page-editor-config>
```

### B.2.2.5 rcv-config

Use the `rcv-config` element to register custom Resource Catalogs with Oracle 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 Oracle 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 12.3,
"Creating a Custom Resource Catalog."

Insert the rcv-config element within the adf-config tag. The following example
shows how you can use the rcv-config element to register a custom catalog called
users-catalog:

```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 Oracle 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.2.2.6 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 11.5.2, "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:actionsCategory>
    <cust:actions>
      <cust:action name="showMinimizeAction" value="true"/>
      <cust:action name="showMoveAction" value="false"/>
    </cust:actions>
  </cust:customizableComponentsSecurity>
</adf-config>
```

For more information about applying action-level restrictions, see Section 11.5, "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 9.3, "Enabling Custom Actions on a Show Detail Frame Enclosing a Task Flow."

### B.2.2.7 mds-config

When defining type-level 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 11.1.1, "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.2.2.8 resource-string-editor

Use a resource-string-editor element to enable resource string editing in your application. Insert the resource-string-editor element within the page-editor-config section, as shown in the following example:

```xml
<page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">
  ...<resource-string-editor>
    <enabled>true</enabled>
  </resource-string-editor>
</page-editor-config>
```

For more information, see Section 9.11, "Configuring Runtime Resource String Editing."

### B.2.2.9 enable-source-view

Use the enable-source-view element to enable or disable Source view in Oracle Composer. Insert the enable-source-view element within the page-editor-config section. Source view is enabled by default in Oracle Composer. You can disable it by setting enable-source-view to false, as shown in the following example:

```xml
<page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">
  ...<enable-source-view>false</enable-source-view>
</page-editor-config>
```

For more information, see Section 9.8, "Disabling Source View for the Application."
B.2.2.10 persistent-change-manager

Use this to configure the change manager to be used for persisting changes made in Oracle 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 all runtime customizations to MDS:

```xml
<adf-faces-config xmlns="http://xmlns.oracle.com/adf/faces/config">
  <persistent-change-manager>
  </persistent-change-manager>
  ...
</adf-faces-config>
```

For more information about change persistence in Oracle Composer, see Section 9.10, "Configuring the Persistence Change Manager."

For more information about the persistent-change-manager element, see Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework.

B.2.2.11 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 Oracle Composer components to the page, the adf-config.xml file is populated with the taglib-config element containing a list of tags and attributes that must be persisted. You can add more elements and attributes to the default list.

For more information about this element, see Oracle Fusion Middleware Fusion Developer’s Guide for Oracle Application Development Framework.

Insert the taglib-config element within the adf-faces-config section. The taglib-config element appears as shown in the following example:

```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 9.10, "Configuring the Persistence Change Manager."
**B.2.2.12 Sample adf-config.xml File**

Example B–2 shows a sample adf-config.xml file with Oracle Composer-specific configurations.

**Example B–2 Sample adf-config.xml File with Oracle 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 Oracle Composer sandbox -->
          <!-- Your jspx customizations alone go here -->
          <namespace path="/pages">
            metadata-store-usage="WebCenterFileMetadataStore">
              <namespace-restriction type="CUSTOMIZATIONS"/>
            </namespace>
          <!-- Your pagedef customizations alone go here -->
          <namespace path="/pageDefs">
            metadata-store-usage="WebCenterFileMetadataStore">
              <namespace-restriction type="CUSTOMIZATIONS"/>
            </namespace>
        </metadata-namespaces>
        <metadata-store-usages>
          <metadata-store-usage id="WebCenterFileMetadataStore"
                                default-cust-store="true">
            <metadata-store class-name="oracle.mds.dt.persistence.stores.file.SrcControlFileMetadataStore">
              <property name="metadata-path" value="../../mds"/>
            </metadata-store>
          </metadata-store-usage>
        </metadata-store-usages>
      </persistence-config>

      <cust-config>
        <match>
          <customization-class name="oracle.adf.share.config.SiteCC"/>
        </match>
      </cust-config>

      <cache-config>
        <max-size-kb>100000</max-size-kb>
      </cache-config>

      <!-- Registration of a standalone XML file used for type-level customization policies -->
      <type-config>
        <standalone-definitions>
          <file>Directory_Name/standalone.xml</file>
        </standalone-definitions>
      </type-config>
    </mds-config>
  </mdsC:adf-mds-config>
</adf-config>
```
<page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">
  <!-- Oracle Composer add-on panel configuration -->
  <addon-panels>
    <addon-panel name="oracle.adf.pageeditor.addonpanels.page-settings" />
    <addon-panel name="oracle.adf.pageeditor.addonpanels.page-reset" />
    <addon-panel name="oracle.fod.custom.panel" />
  </addon-panels>
  <!-- Oracle Composer sandbox configuration -->
  <sandbox-namespaces>
    <namespace path="/pages"/>
    <namespace path="/pageDefs"/>
  </sandbox-namespaces>
  <!-- ComposerSessionOptionsFactory class registration for implementing customization layers -->
  <session-options-factory>view.AppsSessionOptionsFactoryImpl</session-options-factory>
  <!-- Resource string editor configuration -->
  <resource-string-editor>
    <enabled>true</enabled>
  </resource-string-editor>
  <!-- Switch to enable or disable Oracle Composer Source view -->
  <enable-source-view>false</enable-source-view>
</page-editor-config>

<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>

<!-- Oracle Composer and Customizable Components actions security configuration -->
  <cust:enableSecurity value="true"/>
  <cust:actionsCategory>
    <cust:actionCategory name="personalizeActionsCategory" value="false"/>
    <cust:actionCategory name="editActionsCategory" value="true"/>
  </cust:actionsCategory>
  <cust:actions>
    <cust:action name="showMinimizeAction" value="true"/>
    <cust:action name="showMoveAction" value="false"/>
  </cust:actions>
</cust:customizableComponentsSecurity>

<adf-faces-config xmlns="http://xmlns.oracle.com/adf/faces/config">
  <!-- Oracle Composer persistence change manager configuration -->
  <persistent-change-manager>
<persistent-change-manager-class>oracle.adf.view.rich.change.MDSDocumentChangeManager</persistent-change-manager-class>

<!-- Oracle Composer default persistence configuration -->
<taglib-config>
  <taglib uri="http://xmlns.oracle.com/adf/faces/customizable">
    <tag name="showDetailFrame">
      <persist-operations>all</persist-operations>
      <attribute name="expansionMode">
        <persist-changes>true</persist-changes>
      </attribute>
      <attribute name="contentStyle">
        <persist-changes>true</persist-changes>
      </attribute>
    </tag>
    <tag name="panelCustomizable">
      <persist-operations>all</persist-operations>
    </tag>
  </taglib>
</taglib-config>
</adf-faces-config>
</adf-config>

**B.2.3 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 Oracle Composer sandbox. The following sections describe the `web.xml` configurations specific to Oracle Composer.

**B.2.3.1 CHANGE_PERSISTENCE Context Parameter**

The `CHANGE_PERSISTENCE` context parameter in the application’s `web.xml` file specifies how end user customizations are persisted. To enable changes in an Oracle 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 WebCenter 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 Oracle Composer components to such an application, you
must ensure that the CHANGE_PERSISTENCE parameter is set to ComposerChangeManager.

For more information, see Section 9.10, "Configuring the Persistence Change Manager."

B.2.3.2 WebCenterComposerFilter

Use WebCenterComposerFilter in the web.xml file to register Oracle Composer's ComposerSessionOptionsFactory with Oracle ADF for every HTTP request. The request is then handled depending on your implementation of ComposerSessionOptionsFactory. This filter can be defined as shown in the following example:

```
<filter-mapping>
  <filter-name>WebCenterComposerFilter</filter-name>
  <url-pattern>/faces/*</url-pattern>
  <dispatcher>FORWARD</dispatcher>
  <dispatcher>REQUEST</dispatcher>
</filter-mapping>
```

You must use WebCenterComposerFilter if you are performing the following tasks:

- Creating customization layers. For information, see Section 10.3, "Adding Customization Layers to View and Edit Modes: Example."
- Enabling sandbox creation in Oracle Composer. For more information, see Section 10.2.1, "How to Enable Oracle Composer Sandbox Creation."

B.3 Oracle Composer Default Add-Ons and Property Panels

This section lists the add-ons and property panels available by default in Oracle Composer.

B.3.1 Default Add-Ons

This section lists the add-ons available by default and provides the names with which these add-ons are registered in the adf-config.xml file. These add-on names are useful if you choose to create custom add-ons or exclude default add-ons in Oracle Composer. 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 this, the default add-ons are not displayed in Oracle Composer. For information about add-ons, see Section 8.2, "Creating Oracle Composer Add-Ons."

- **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.
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. For information about creating property panels, see Section 8.3, "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.
  
  This panel is referenced with the name `oracle.adf.pageeditor.pane.generic-property-inspector` in the `pe_ext.xml` file.

- **Style**
  Used to define the appearance of the component instance. This panel is displayed in the Component Properties dialog for all components.
  
  This panel is referenced with the name `oracle.adf.pageeditor.pane.inline-style-editor` in the `pe_ext.xml` file.

- **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.
  
  This panel is referenced with the name `oracle.adf.pageeditor.pane.content-style-editor` in the `pe_ext.xml` file.

- **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.
  
  This panel is referenced with the name `oracle.adf.pageeditor.pane.events` in the `pe_ext.xml` file.

- **Region Parameters**
  Used to define parameters for the task flow region. This panel displays in the Component Properties dialog only for task flow regions.
  
  This panel is referenced with the name `oracle.adf.pageeditor.pane.region-param` in the `pe_ext.xml` file.

- **Portlet Parameters**
  Used to define portlet parameters. This panel displays in the Component Properties dialog only for portlets.
  
  This panel is referenced with the name `oracle.adf.pageeditor.pane.portlet-param` in the `pe_ext.xml` file.

- **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.

This panel is referenced with the name oracle.adf.pageeditor.pane.layout-cust-prop in the pe_ext.xml file.

- **Child Components**

  Used to rearrange and show or hide child components inside a Panel Customizable component. This panel displays only for Panel Customizable components.

  This panel is referenced with the name oracle.adf.pageeditor.pane.child-navigator in the pe_ext.xml file.

- **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.

  This panel is referenced with the name oracle.webcenter.page.pane.page-param in the pe_ext.xml file.

- **Page Security**

  Used to define page permissions. This panel displays in the Page Properties dialog for secured application pages.

  This panel is referenced with the name oracle.webcenter.page.pane.page-sec in the pe_ext.xml file.

### B.4 Oracle Composer Help Topic IDs

By default, context-sensitive help is not available for Oracle Composer panels and dialogs, and therefore Help icons are not displayed. These icons are displayed when you hook up help topics for the different panels and dialogs. Oracle Composer help topics are available in the pages_cs.jar file on the Oracle WebCenter Suite 11g Demonstrations and Samples page on the Oracle Technology Network (OTN) at:

http://webcenter.oracle.com

To integrate help using the JAR file, you must have created a help provider. For information about displaying help for components and creating a help provider, see the section, "Displaying Help for Components" in the Oracle Fusion Middleware Web User Interface Developer’s Guide for Oracle Application Development Framework.

Alternatively, you can extract the raw HTML files from pages_cs.jar and hook up those files directly from Oracle Composer panels or dialogs. However, in this case the reference links within these files do not work and must be deleted.

To integrate this help in the Oracle Composer UI in your custom WebCenter application, use the topic IDs in Table B–17, which describes the context-sensitive help topics available in pages_cs.jar.

<table>
<thead>
<tr>
<th>Topic ID to be Used</th>
<th>Corresponding Oracle Composer UI</th>
</tr>
</thead>
<tbody>
<tr>
<td>pages_cs_pgeditmodesv</td>
<td>Source view of a page in Edit mode</td>
</tr>
</tbody>
</table>
Table B–17  (Cont.) Oracle Composer Runtime Help Topics

<table>
<thead>
<tr>
<th>Topic ID to be Used</th>
<th>Corresponding Oracle Composer UI</th>
</tr>
</thead>
<tbody>
<tr>
<td>pages_cs_pgeditmodedv</td>
<td>Design view of a page in Edit mode</td>
</tr>
<tr>
<td>pages_cs_cpparams</td>
<td>Parameters tab in the Component Properties dialog</td>
</tr>
<tr>
<td>pages_cs_cpdisplayopts</td>
<td>Display Options tab in the Component Properties dialog</td>
</tr>
<tr>
<td>pages_cs_cpstyle</td>
<td>Style tab in the Component Properties dialog</td>
</tr>
<tr>
<td>pages_cs_cpcontstyle</td>
<td>Content Style tab in the Component Properties dialog</td>
</tr>
<tr>
<td>pages_cs_othercss</td>
<td>Other CSS option on the Style tab in the Component Properties dialog</td>
</tr>
<tr>
<td>pages_cs_cpendents</td>
<td>Events tab in the Component Properties dialog</td>
</tr>
<tr>
<td>pages_cs_compcat</td>
<td>Catalog dialog</td>
</tr>
<tr>
<td>pages_cs_ppgen</td>
<td>Display Options tab in the Page Properties dialog</td>
</tr>
<tr>
<td>pages_cs_ppsecurity</td>
<td>Security tab in the Component Properties dialog</td>
</tr>
<tr>
<td>pages_cs_pppparams</td>
<td>Parameters tab in the Component Properties dialog</td>
</tr>
<tr>
<td>pages_cs_cplayoutprops</td>
<td>Layout Customizable tab in the Component Properties dialog for a Layout Customizable component</td>
</tr>
<tr>
<td>pages_cs_resourceeditor</td>
<td>Select text resource dialog invoked on selecting the Select Text Resource option on the menu next to a field in the Component Properties dialog</td>
</tr>
</tbody>
</table>

B.5 Oracle Composer Components Style-Specific Properties

This section includes a series of tables that describe the style selectors associated with Oracle Composer components. Additionally, it describes how to use the background property to choose one of three skin-defined looks for these components.

This section contains the following subsections:

- **Section B.5.1, "Style Selectors for Oracle Composer Components"
- **Section B.5.2, "Style Attributes"

B.5.1 Style Selectors for Oracle 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**
- **Page Customizable Style Selectors**
- **Layout Customizable Style Selectors**
- **Panel Customizable Style Selectors**
- **Show Detail Frame Style Selectors**
B.5.1.1 Global Style Selectors

Use global style selectors to define styles for multiple components within the application. Table B–18 lists and describes the global style selectors relevant to Oracle Composer components.

<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 Oracle Composer, with a color scheme of dark.</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 “display: block” with height, width attributes to display in block mode.</td>
</tr>
</tbody>
</table>

B.5.1.2 Page Customizable Style Selectors

Use the style selectors in Table B–19 to skin Page Customizable components.

<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>
<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>
</tbody>
</table>
B.5.1.3 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</td>
<td>layoutCustomizable:menu:edit</td>
</tr>
<tr>
<td>af</td>
<td>commandImageLink::text</td>
</tr>
<tr>
<td>af</td>
<td>commandImageLink::text:hover</td>
</tr>
</tbody>
</table>

B.5.1.4 Panel Customizable Style Selectors

Use the style selectors listed in Table B–21 to skin Panel Customizable components.

**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.

<table>
<thead>
<tr>
<th>Style Selector</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>af</td>
<td>panelCustomizable</td>
</tr>
<tr>
<td>af</td>
<td>panelCustomizable.PESstretched</td>
</tr>
<tr>
<td>af</td>
<td>panelCustomizable:edit</td>
</tr>
<tr>
<td>af</td>
<td>panelCustomizable:edit:drop-target</td>
</tr>
</tbody>
</table>
B.5.1.5 Show Detail Frame Style Selectors

Use the style selectors listed in Table B–22 to skin portlets and Show Detail Frame components.

In WebCenter applications, each portlet is rendered with portlet chrome (for more information, see Section 29.1.1, "Portlet Anatomy"). The portlet chrome shares the same chrome rendering mechanism as a Show Detail Frame component. Thus, the style and icon selectors that apply to Show Detail Frame also apply to the portlet chrome. In other words, in addition to defining styles for Show Detail Frame components, use Show Detail Frame style and icon selectors to define styles for portlets.

**Note:** Some Show Detail Frame style selectors in Table B–22 have light, medium, and dark color scheme options. Using these you can define three distinct looks in your CSS and specify which one to use through the background property in the JDeveloper Property Inspector. Depending on which value is specified for a component instance’s background property, the skin applies the relevant style.

### Table B-21 (Cont.) Panel Customizable Style Selectors

<table>
<thead>
<tr>
<th>Style Selector</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>af</td>
<td>panelCustomizable::edit-mode-content-style</td>
</tr>
<tr>
<td>af</td>
<td>panelCustomizable::add-icon-style</td>
</tr>
<tr>
<td>af</td>
<td>panelCustomizable::add-icon-style:active</td>
</tr>
<tr>
<td>af</td>
<td>panelCustomizable::add-icon-style:hover</td>
</tr>
<tr>
<td>af</td>
<td>panelCustomizable::add-icon-style:rtl</td>
</tr>
<tr>
<td>af</td>
<td>panelCustomizable::edit-icon-style</td>
</tr>
<tr>
<td>af</td>
<td>panelCustomizable::edit-icon-style:active</td>
</tr>
<tr>
<td>af</td>
<td>panelCustomizable::edit-icon-style:hover</td>
</tr>
<tr>
<td>af</td>
<td>panelCustomizable::delete-icon-style</td>
</tr>
<tr>
<td>af</td>
<td>panelCustomizable::delete-icon-style:active</td>
</tr>
<tr>
<td>af</td>
<td>panelCustomizable::delete-icon-style:hover</td>
</tr>
<tr>
<td>.PanelCustomizableDropProxy</td>
<td>Specifies the style for the placeholder area inside this component when you drag another component on the page and hover over this component to drop it in.</td>
</tr>
</tbody>
</table>

### Table B–22 Show Detail Frame Style Selectors

<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.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>
</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::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-center</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-start</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::header-start:rtl</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::header-start:dark:rtl</td>
</tr>
<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:dark:rtl</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::icon-style</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::icon-style:active</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::icon-style:hover</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>
<tr>
<td>af</td>
<td>showDetailFrame::header-actions:active</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::header-actions:hover</td>
</tr>
<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::collapsable-icon-style</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::collapsable-icon-style:active</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::collapsable-icon-style:hover</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::discloseable-icon-style</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::discloseable-icon-style:active</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::discloseable-icon-style:hover</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-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>
</tbody>
</table>
Table B–23  (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::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-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:hover</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::resize</td>
</tr>
<tr>
<td>af</td>
<td>showDetailFrame::resize:rtl</td>
</tr>
</tbody>
</table>

B.5.2 Style Attributes

Table B–23 provides a quick reference for the style attributes available for WebCenter Customizable Components and Oracle 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 Oracle Composer and WebCenter 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>Attribute</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</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</td>
</tr>
<tr>
<td></td>
<td>ContentStyle property to turn off portlet borders. Using this attribute</td>
</tr>
<tr>
<td></td>
<td>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</td>
</tr>
<tr>
<td></td>
<td>inherit to determine the thickness of the border or inherit border width</td>
</tr>
<tr>
<td></td>
<td>from a parent component. Alterntively, click the Edit icon and define a</td>
</tr>
<tr>
<td></td>
<td>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</td>
</tr>
<tr>
<td></td>
<td>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</td>
</tr>
<tr>
<td></td>
<td>component text. Enter this value manually.</td>
</tr>
<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</td>
</tr>
<tr>
<td></td>
<td>preferred unit of measurement.</td>
</tr>
<tr>
<td></td>
<td>■ Choose Percentage: Specify font size as a percentage of the normal</td>
</tr>
<tr>
<td></td>
<td>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</td>
</tr>
<tr>
<td></td>
<td>parent component.</td>
</tr>
<tr>
<td></td>
<td>■ larger: Increase the font size to larger than the font associated with</td>
</tr>
<tr>
<td></td>
<td>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</td>
</tr>
<tr>
<td></td>
<td>component.</td>
</tr>
<tr>
<td></td>
<td>■ small: Set the font size to smaller than the default font size of the</td>
</tr>
<tr>
<td></td>
<td>parent component.</td>
</tr>
<tr>
<td></td>
<td>■ smaller: Set the font size to smaller than the default font size of the</td>
</tr>
<tr>
<td></td>
<td>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</td>
</tr>
<tr>
<td></td>
<td>not have a formal italic member.</td>
</tr>
<tr>
<td></td>
<td>■ inherit: Inherit font style from a parent component.</td>
</tr>
</tbody>
</table>
Oracle Composer Components Style-Specific Properties

Table B–23 (Cont.) Style Attributes of Oracle Composer and WebCenter Customizable Components

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<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>
<tr>
<td>list-style-type</td>
<td>The type of indicator to use for items on a list. Use this instead of specifying a list-style-image.</td>
</tr>
<tr>
<td></td>
<td>Select from among many styles offered on the list, including &lt;none&gt;, which means the browser’s default style is used, and inherit, which means the</td>
</tr>
<tr>
<td></td>
<td>list-style-type is inherited from a parent component.</td>
</tr>
<tr>
<td>margin</td>
<td>The border of space surrounding the component content. Choose from:</td>
</tr>
<tr>
<td></td>
<td>■ Choose Length: Specify an absolute value for all margins.</td>
</tr>
<tr>
<td></td>
<td>■ Choose Percentage: Set a value as a percentage of the margin of a parent component.</td>
</tr>
<tr>
<td></td>
<td>■ inherit: Inherit margin value from a parent component.</td>
</tr>
<tr>
<td></td>
<td>■ auto: Set the value automatically according to browser defaults.</td>
</tr>
<tr>
<td>padding</td>
<td>The amount of space between the component and its margin or, if there is a border, between the component and its border. Choose 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 the value as a percentage of the padding set for a parent component.</td>
</tr>
<tr>
<td></td>
<td>■ inherit: Inherit the value from a parent component.</td>
</tr>
</tbody>
</table>
Customizable Components (HTML) Properties

Table B–23  (Cont.) Style Attributes of Oracle Composer and WebCenter Customizable Components

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>text-align</td>
<td>The horizontal alignment of the component text. Choose from:</td>
</tr>
<tr>
<td></td>
<td>- &lt;none&gt;: Use browser default.</td>
</tr>
<tr>
<td></td>
<td>- left</td>
</tr>
<tr>
<td></td>
<td>- right</td>
</tr>
<tr>
<td></td>
<td>- center</td>
</tr>
<tr>
<td></td>
<td>- justify: Align text so that all lines start and end at the same point left and right.</td>
</tr>
<tr>
<td></td>
<td>- auto: Apply a value automatically, either left or right.</td>
</tr>
<tr>
<td></td>
<td>- inherit: Inherit alignment from a parent component.</td>
</tr>
<tr>
<td>text-decoration</td>
<td>The decorative value to apply to component text. Choose from:</td>
</tr>
<tr>
<td></td>
<td>- &lt;none&gt;: Use the browser default for hyperlinks; otherwise, no text decoration.</td>
</tr>
<tr>
<td></td>
<td>- none: No text decoration, including no underscore for hyperlinks.</td>
</tr>
<tr>
<td></td>
<td>- underline: Draw a line under all component text.</td>
</tr>
<tr>
<td></td>
<td>- overline: Draw a line over all component text.</td>
</tr>
<tr>
<td></td>
<td>- line-through: Draw a line through the middle of all component text.</td>
</tr>
<tr>
<td></td>
<td>- blink: Make component text blink.</td>
</tr>
<tr>
<td></td>
<td>- inherit: Inherit text decoration from a parent component.</td>
</tr>
<tr>
<td>vertical-align</td>
<td>The vertical alignment of component text. Choose 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>- inherit: Inherit the value from a parent component.</td>
</tr>
<tr>
<td></td>
<td>- baseline: Set the text on the baseline of the text row.</td>
</tr>
<tr>
<td></td>
<td>- middle: Set the text in the middle of the text row.</td>
</tr>
<tr>
<td></td>
<td>- text-bottom: Set the text on the bottom of the text row, lower than the baseline.</td>
</tr>
<tr>
<td></td>
<td>- text-top: Set the text at the top of the text row, above the baseline.</td>
</tr>
<tr>
<td></td>
<td>- sub: Set the text as subscript.</td>
</tr>
<tr>
<td></td>
<td>- super: Set the text as superscript.</td>
</tr>
<tr>
<td>width</td>
<td>The spacing between text. Specify a width 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>
</tbody>
</table>

B.6 Customizable Components (HTML) Properties

Customizable Components (HTML) are similar in function to customizable components from the 10.1.3.2.0 release and can be used to add more customizable components to your migrated 10.1.3.2.0 application page. These customizable components are available from the Customizable Components (HTML) tag library in
JDeveloper. When you add customizable components to the page, default values are assigned to certain attributes and you can define values for the remaining attributes as required.

Customizable components enable users to minimize or maximize, hide or show, or move any component on the page at runtime.

### B.6.1 Panel Customizable (HTML) Component

Use the Panel Customizable (HTML) component as a container for page content that can be customized at runtime. Components added inside a Panel Customizable (HTML) can be maximized, minimized, or rearranged.

Use this component only to perform runtime customizations on child components. If you just want a container to arrange components at design time, then it is recommended that you use an ADF Faces container like Panel Group Layout.

Table B–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>
<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>
### 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. 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.</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: binding=&quot;#{yourManagedBean.Binding}&quot;</td>
</tr>
</tbody>
</table>
B.6.2 Show Detail Frame (HTML) Component

Add Show Detail Frame (HTML) components inside Panel Customizable (HTML) components to enable runtime personalizations 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>
<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. |
| text      | String | Yes | A title for the Show Detail Frame (HTML) component. |
| icon      | String | Yes | If you want to add an icon on the header of the Show Detail Frame component, then this specifies the URI for the image to be used. For example:  
icon="http://source-pc/images/accessability.gif"  
Note: An image that is stored at the document root does not require a full path. For example:  
icon="detail.gif" |

Appearance Attributes
### Table B–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>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>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>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>Action 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>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>
</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>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></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Style Attributes</strong></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>
<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>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Behavior Attributes</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>partialTriggers</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 - :: pops out of the component's naming container (or itself if the component is a naming container) and begin the search from there, :::: pops out of two naming containers (including itself if the component is a naming container) and begin the search from there.</td>
</tr>
<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 disclosure state changes.</td>
</tr>
</tbody>
</table>
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 <strong>Show Detail Frame (HTML)</strong> 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 <strong>Show Detail Frame (HTML)</strong> header.</td>
</tr>
</tbody>
</table>

JDeveloper displays all facets available to the **Show Detail Frame (HTML)** component in the Structure window. However, only those that contain UI components appear activated.

### B.6.3 Customizable Components (HTML) Style Selectors

The tables in this section describe the style selectors that you can use to skin WebCenter Customizable Components.

This section contains the following subsections:

- Panel Customizable (HTML) Style Selectors
B.6.3.1 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.6.3.4, "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.6.3.2 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>
<tr>
<td>afh</td>
<td>showDetailFrame::header-medium</td>
</tr>
<tr>
<td>afh</td>
<td>showDetailFrame::header-dark</td>
</tr>
<tr>
<td>afh</td>
<td>showDetailFrame::content-light</td>
</tr>
<tr>
<td>afh</td>
<td>showDetailFrame::content-medium</td>
</tr>
<tr>
<td>afh</td>
<td>showDetailFrame::content-dark</td>
</tr>
<tr>
<td>afh</td>
<td>showDetailFrame::main-menu-container</td>
</tr>
<tr>
<td>afh</td>
<td>showDetailFrame::sub-menu-container</td>
</tr>
<tr>
<td>afh</td>
<td>showDetailFrame::menu-item</td>
</tr>
<tr>
<td>afh</td>
<td>showDetailFrame::menu-item:hover</td>
</tr>
<tr>
<td>afh</td>
<td>showDetailFrame::sub-menu-item</td>
</tr>
</tbody>
</table>
B.6.3.3 Property Keys

Use property keys to control the display of custom menu items and component action icons. Though you include property keys in your custom skin, they are not represented in the generated CSS that results from the skin.

To explain, skins go through a process that results in a generated CSS. In turn, the generated CSS is consumed by the application. Most style selectors for Panel Customizable and Show Detail Frame components are represented in the generated CSS. Property keys are the exception. Although they affect the application as much as any other component style selector, they are not represented in the final generated CSS.

Table B–29 lists and describes property keys relevant to Panel Customizable and Show Detail Frame components.
B.6.3.4 Icon Selectors

Icons are either displayed or not displayed depending on whether the component’s ora-menu-icon-display property key is set to true or false. Property keys are described in Section B.6.3.3, "Property Keys."

Each icon selector has a light, medium, and dark scheme. Using these you can define three distinct looks in your CSS and specify which one to use through the background property in the JDeveloper Property Inspector. Depending on which value is specified for a component instance’s background property, the skin applies the relevant style.

For easy, error-free portability, store all application icons under the WebCenter 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>
<tbody>
<tr>
<td>showDetailFrame {-ora-additional-actions-position-last:true}</td>
<td>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.</td>
</tr>
<tr>
<td>showDetailFrame {-ora-menu-icon-display:false}</td>
<td>Controls the display of icons next to their related commands on a Show Detail Frame Actions menu. Set to true to display icons to the left of each action on the Actions menu. By default, no icons are displayed to the left of individual actions. The default value is false. For information about specifying the icons to use when this property key is set to true, see Section B.6.3.4, &quot;Icon Selectors.&quot;</td>
</tr>
</tbody>
</table>

For easy, error-free portability, store all application icons under the WebCenter 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 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 Actions menu. The Actions menu lists the actions a user can perform on the component. In a WebCenter application, the Actions icon is rendered on the right corner of the component header.</td>
</tr>
<tr>
<td><code>panelCustomizable</code></td>
<td>This icon represents the Minimize option. Minimize collapses the view of the component like a window shade. In a WebCenter application, the Minimize icon is rendered on the left side of the component header. See also, <code>showDetailFrame::light-ExpandIcon:alias</code>.</td>
</tr>
<tr>
<td><code>showDetailFrame</code></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 WebCenter application, the Maximize icon is displayed to the left of the Maximize command on the component's Actions menu.</td>
</tr>
<tr>
<td><code>panelCustomizable</code></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 WebCenter application, the Maximize icon is displayed to the left of the Maximize command on the component's Actions menu.</td>
</tr>
<tr>
<td><code>showDetailFrame</code></td>
<td>This icon represents the Restore option, which restores maximized views to their default display modes. In a WebCenter application, the Restore icon is rendered to the left of the Restore command on the component's Actions menu.</td>
</tr>
<tr>
<td><code>panelCustomizable</code></td>
<td>This icon represents the Restore option, which restores maximized views to their default display modes. In a WebCenter application, the Restore icon is rendered to the left of the Restore command on the component's Actions menu.</td>
</tr>
</tbody>
</table>
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 WebCenter application, the Expand icon is displayed on the left side of the component header.

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 WebCenter application, the Move icon is displayed to the left of the Move command on the component’s Actions menu.

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 WebCenter application, the Move Left icon is displayed to the left of the Move Left submenu item on the component’s Actions menu.

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 WebCenter application, the Move Right icon is displayed to the left of the Move Right submenu item on the component’s Actions menu.

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 WebCenter 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 Customizable Components**

<table>
<thead>
<tr>
<th>Selector</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>showDetailFrame</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>panelCustomizable</td>
<td>In a WebCenter 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>showDetailFrame</td>
<td>This icon provides an image for the top-left corner of the component header.</td>
</tr>
<tr>
<td>panelCustomizable</td>
<td></td>
</tr>
<tr>
<td>showDetailFrame</td>
<td>This icon provides the image for the top-right corner of the component header.</td>
</tr>
<tr>
<td>panelCustomizable</td>
<td></td>
</tr>
<tr>
<td>showDetailFrame</td>
<td>This icon provides the left portion of the component's FadeIn-FadeOut toolbar.</td>
</tr>
<tr>
<td>panelCustomizable</td>
<td>The FadeIn-FadeOut toolbar comes into play when the adfp:portlet tag attribute isSeededInteractionAvailable is set to true and displayHeader is set to false.</td>
</tr>
<tr>
<td>showDetailFrame</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>panelCustomizable</td>
<td>If the page design is very simple, then the FadeIn-FadeOut toolbar may not display, even when displayHeader is set to false and isSeededInteractionAvailable is set to true.</td>
</tr>
<tr>
<td>showDetailFrame</td>
<td>This icon provides the right portion of the component's FadeIn-FadeOut toolbar.</td>
</tr>
<tr>
<td>panelCustomizable</td>
<td>See the description for showDetailFrame::light-ToolbarLeftIcon:alias for an explanation of the FadeIn-FadeOut toolbar.</td>
</tr>
</tbody>
</table>
### Table B–30  (Cont.) Icon Selectors for WebCenter Customizable Components

<table>
<thead>
<tr>
<th>Selector</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>showDetailFrame</code></td>
<td>This icon provides the center portion of the component’s FadeIn-FadeOut toolbar. See the description for <code>showDetailFrame::light-ToolbarLeftIcon:alias</code> for an explanation of the FadeIn-FadeOut toolbar.</td>
</tr>
<tr>
<td><code>panelCustomizable</code></td>
<td>This icon represents the Edit option on the component menu. In a WebCenter application, the Edit icon is displayed to the left of the Edit menu item on the component’s Actions menu.</td>
</tr>
<tr>
<td><code>showDetailFrame</code></td>
<td>This icon represents the Help option on the component menu. In a WebCenter application, the Help icon is displayed to the left of the Help menu item on the component’s Actions menu.</td>
</tr>
</tbody>
</table>

This appendix provides reference information about the configuration, location, and attributes of the default catalog definition file. This information is useful while performing the tasks described in Chapter 7, "Enabling Runtime Editing of Pages Using Oracle Composer."

This appendix includes the following sections:

- Section C.1, "Configuration and Location of Catalog Definitions"
- Section C.2, "Default Catalog Definition"
- Section C.3, "XML Schema"
- Section C.4, "Catalog Definition Attributes"

## 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-config xmlns="http://xmlns.oracle.com/adf/rcs/viewer/adf-config">
  <catalog-selector class-name=""/>
  <default-catalog catalog-name="default-catalog"/>
</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 `mds-package-prefix` attribute in the `adf-rcs-config` element in your `adf-config.xml`:

```xml
<adf-rcs-config xmlns="http://xmlns.oracle.com/adf/rcs/adf-config">
  <rcs-config mds-package-prefix="/oracle/adf/rc/metadata"/>
</adf-rcs-config>
```

The default Resource Catalog root is defined as:

```
APPLICATION_ROOT/mds/oracle/adf/rc/metadata
```

When you create or access catalogs using the Resource Catalog service APIs, the fully qualified MDS reference for the catalog (its storage location in the MDS repository) is dynamically constructed by appending the catalog identifier to the `resource-catalog-root` as follows:
If the value of `resource-catalog-root` is `/oracle/adf/rc/metadata` then all your catalog definitions must be stored within the `/oracle/adf/rc/metadata` MDS package.

If you create a catalog definition file and save it in MDS at `/oracle/adf/rc/metadata/myFirstCatalog.xml`, then that catalog’s identifier would be `myFirstCatalog`. If this is to be the default catalog for your application, then `catalog-name` should be set to `myFirstCatalog` in `adf-config.xml`.

If your application requires multiple catalogs, you can organize them into sub packages. In this case, you might create a catalog definition file in MDS at `/oracle/adf/rc/metadata/hr/personalization.xml`. That catalog’s identifier would be `hr/personalization`. If this is to be the default catalog for your application, then `catalog-name` should be set to `hr/personalization`.

### C.2 Default Catalog Definition

Example C–1 shows the default catalog definition file to illustrate the most common elements.

#### Example C–1 Example of a Catalog Definition

```xml
<?xml version = '1.0' encoding = 'UTF-8'?>
<catalogDefinition xmlns="http://xmlns.oracle.com/adf/rcs/catalog"
    id="catalogDefinition"
    name="Default Resource Catalog"
    description="Default resource catalog definition containing sample entries">
    <contents>
        <!-- **********************************************************************
        * Custom folder exposing ADF Faces components                            *
        * Comment out this element if you want to remove it from the catalog.   *
        * ********************************************************************** -->
        <customFolder id="facesComponents" name="ADF Faces Components"
            description="ADF Faces components you can add to application pages"
            factoryClass="oracle.adfinternal.view.page.editor.componentcatalog.adapter.ComponentObjectFactory"/>
        <!-- **********************************************************************
        * Dynamically include the portlets custom folder into the catalog at runtime. The custom folder will only be included if the portlet runtime jar files are included in the application classpath.  *
        * ********************************************************************** -->
        <customContent id="portletContent" contentProviderClass="oracle.adf.rc.webcenter.WebCenterContentProvider"/>
        <!-- **********************************************************************
        * Uncomment the following <customFolder> element if you have configured the Document Library service in your application and want to include documents in the catalog.  *
        * ********************************************************************** -->
        <customFolder id="doclibDocuments"
            name="Documents"
            description="Documents from the Document Library Service"
            factoryClass="oracle.webcenter.content.model.rc.CustomFolderContextFactory"/>
    </contents>
</catalogDefinition>
```

C-2 Oracle Fusion Middleware Developer's Guide for Oracle WebCenter
C.3 XML Schema

Example C–2 provides the catalog definition XML schema for your reference.

Example C–2   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:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:jaxb="http://java.sun.com/xml/ns/jaxb" jaxb:version="1.0">
  <xsd:annotation>
    <!--- *************** -->
    * Uncomment the following <resource> element if you have configured the *
    * Document Library service in your application and want to include the *
    * Document Library main view in the catalog. *
    * *
    * NOTE: application.classpath is an implicitly defined "repository" for *
    * accessing ADF Libraries that are included in your classpath *
    ************************************************************-->

  <resource id="doclibMainView"
    name="Document Library Task Flow"
    description="Main view of the Document Library Service"
    repository="application.classpath"
    path="doclib-service-view.jar/ADF_TaskFlow/oracle+webcenter+doclib+view+jsf+taskflows+mainView.xml#doclib-document-library"/>

  <!--- *************** -->
  * To create a link to a task flow in an ADF Library: *
  * 1) add a <resource> tag to your catalog definition & set the *
  *    "repository" attribute to "application.classpath" *
  *    (see doclibMainView above) *
  * 3) set the "repository" attribute to the name of your file system *
  *    connection. *
  * 4) set the "path" attribute for the task flow you are interested in. *
  *    The path is of the following form: *
  *    path_to_jar/ADF_TaskFlow/task_flow_path *
  *    *
  *    To obtain the task_flow_path, use the file system connection from (1) *
  *    and navigate to the task flow in your ADF Library. Mouse-over your task *
  *    flow so its tool-tip is displayed. The tool-tip shows he fully qualified* *
  *    ID of the task flow. Take this value and replace all occurrences of "/" *
  *    with "+". For example: *
  *    *
  *    Tool-tip: *
  *    oracle/webcenter/collab/announcement/view/taskflows/main-view-definition.xml#announcement-main-view *
  *    Task_flow_path: *
  *    oracle+webcenter+collab+announcement+view+taskflows+main-view-definition.xml#announcement-main-view *

  </xsd:annotation>
  </xsd:schema>
```
<xsd:appinfo>
  <jaxb:schemaBindings>
    <jaxb:package name="oracle.adfinternal.rc.catalog.xml"/>
  </jaxb:schemaBindings>
</xsd:appinfo>
</xsd:annotation>
<xsd:complexType name="descriptorType">
  <xsd:attribute name="attributeId" type="xsd:string" use="required"/>
  <xsd:attribute name="labelKey" type="xsd:string" use="required"/>
  <xsd:attribute name="shortLabelKey" type="xsd:string"/>
  <xsd:attribute name="searchable" type="xsd:boolean" default="true"/>
  <xsd:attribute name="multivalue" type="xsd:boolean" default="false"/>
  <xsd:attribute name="endUserVisible" type="xsd:boolean" default="true"/>
  <xsd:attribute name="resourceBundle" type="xsd:string"/>
</xsd:complexType>
<xsd:complexType name="schemaType">
  <xsd:sequence>
    <xsd:element name="descriptor" type="rcs:descriptorType" minOccurs="1" maxOccurs="unbounded"/>
  </xsd:sequence>
  <xsd:attribute name="resourceBundle" type="xsd:string"/>
</xsd:complexType>
<xsd:complexType name="attributeType">
  <xsd:attribute name="attributeId" type="xsd:string" use="required"/>
  <xsd:attribute name="value" type="xsd:string" use="required"/>
  <xsd:attribute name="isKey" type="xsd:boolean" use="required"/>
  <xsd:attribute name="resourceBundle" type="xsd:string"/>
</xsd:complexType>
<xsd:complexType name="attributesType">
  <xsd:sequence>
    <xsd:element name="attribute" type="rcs:attributeType" minOccurs="1" maxOccurs="unbounded"/>
  </xsd:sequence>
  <xsd:attribute name="resourceBundle" type="xsd:string"/>
</xsd:complexType>
<xsd:complexType name="itemType">
  <xsd:sequence>
    <xsd:element name="attributes" type="rcs:attributesType" minOccurs="0" maxOccurs="1"/>
  </xsd:sequence>
  <xsd:attribute name="id" type="xsd:string"/>
  <xsd:attribute name="name" type="xsd:string"/>
  <xsd:attribute name="description" type="xsd:string"/>
</xsd:complexType>
<xsd:complexType name="resourceType">
  <xsd:complexContent>
    <xsd:extension base="rcs:itemType">
      <xsd:attribute name="repository" type="xsd:string"/>
      <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="repositoryType">
  <xsd:complexContent>
    <xsd:extension base="rcs:itemType">
      <xsd:attribute name="repository" type="xsd:string"/>
      <xsd:attribute name="path" type="xsd:string"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexContent>
  <xsd:extension base="rcs:itemType">
    <xsd:sequence>
      <xsd:element name="source" type="rcs:dynamicFolderSourceType"
        minOccurs="1" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:extension>
</xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="dynamicFolderType">
  <xsd:complexContent>
    <xsd:extension base="rcs:itemType">
      <xsd:sequence>
        <xsd:element name="source" type="rcs:dynamicFolderSourceType"
          minOccurs="1" maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:group name="folderContentGroup">
  <xsd:choice>
    <xsd:element name="folder" type="rcs:folderType"/>
    <xsd:element name="repository" type="rcs:repositoryType"/>
    <xsd:element name="resource" type="rcs:resourceType"/>
    <xsd:element name="dynamicFolder" type="rcs:dynamicFolderType"/>
  </xsd:choice>
</xsd:group>

<xsd:complexType name="contentsType">
  <xsd:group ref = "rcs:folderContentGroup" minOccurs="0" maxOccurs="unbounded"/>
</xsd:complexType>

<xsd:complexType name="folderType">
  <xsd:complexContent>
    <xsd:extension base="rcs:itemType">
      <xsd:sequence>
        <xsd:element name="contents" type="rcs:contentsType"
          minOccurs="1" maxOccurs="1"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="catalogType">
  <xsd:complexContent>
    <xsd:extension base="rcs:folderType">
      <xsd:sequence>
        <xsd:element name="schema" type="rcs:schemaType" minOccurs="0"
          maxOccurs="1"/>
      </xsd:sequence>
      <xsd:attribute name="contact" type="xsd:string"/>
      <xsd:attribute name="definitionFilter" type="xsd:string"/>
      <xsd:attribute name="resourceBundle" type="xsd:string"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:element name="catalogDefinition" type="rcs:catalogType"/>
</xsd:schema>
C.4 Catalog Definition Attributes

The catalog definition schema provides a rich set of elements through which you can specify the structure and content of your resource catalogs.

This section contains the following subsections:

- Section C.4.1, "catalogDefinition"
- Section C.4.2, "dynamicFolder"
- Section C.4.3, "folder"
- Section C.4.4, "attributes"
- Section C.4.5, "attribute"
- Section C.4.6, "resource"
- Section C.4.8, "schema"

For an example of creating a custom catalog definition file, see Section 12.3, "Creating a Custom Resource Catalog."

C.4.1 catalogDefinition

The catalogDefinition element defines the overall characteristics of the catalog, such as the location of translated strings.

Example C–3 illustrates the use of catalogDefinition.

Example C–3 Example of catalogDefinition

```xml
<catalogDefinition xmlns="http://xmlns.oracle.com/adf/rcs/catalog"
   id="PersonalSpaceCatalog"
   name="Personal WebCenter Catalog"
   description="Resource Catalog definition for Personal WebCenter"
   resourceBundle="oracle.webcenter.webcenterapp.internal.model.
                  rc.resource.PersonalSpaceCatalogBundle"
   definitionFilter="oracle.webcenter.webcenterapp.internal.model.
                  rc.PersonalSpaceCatalogFilter">
```

Table C–1 describes the keywords of the catalogDefinition element. For information about including attributes for elements, see Section C.4.4, "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>name</td>
<td>The displayed name of the catalog definition.</td>
</tr>
<tr>
<td>description</td>
<td>Is text that describes the catalog definition.</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>
</tbody>
</table>
C.4.2 dynamicFolder

The `dynamicFolder` element defines a folder that is dynamically populated based upon the results of a query.

Dynamic folders have attributes just like any other entry. For information about including attributes for elements, see Section C.4.4, "attributes."

Example C–4 illustrates the use of `dynamicFolder`.

**Example C–4  Example of dynamicFolder**

```xml
<dynamicFolder id="dynamicFolderId"
    name="dynamicFolderName"
    description="folderDescription"
    expression="jndiSearchExpression">
    <source repository="repositoryId" path="repositoryPath"
        includeSubfolders="true"/>
    ...
</dynamicFolder>
```

Table C–2 describes the keywords of the `dynamicFolder` element.

**Table C–2  dynamicFolder Keywords**

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The identifier used when referring to this dynamic folder.</td>
</tr>
<tr>
<td>name</td>
<td>The displayed name of the dynamic folder.</td>
</tr>
<tr>
<td>description</td>
<td>Text that describes the dynamic folder.</td>
</tr>
<tr>
<td>expression</td>
<td>A JNDI query expression. This query is executed against each of the source repositories and the results merged to form the dynamic folder contents.</td>
</tr>
</tbody>
</table>

**source**

The `source` element defines a repository or repository folder that should be included in a search as part of a dynamic folder. Dynamic folders can have one or more `source` repositories.

**Example C–5  Example of source element**

```xml
<source repository="repositoryId" path="repositoryPath" includeSubfolders="true"/>
```

Table C–3 describes the keywords of the `source` element.
C.4.3 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.4.4, "attributes."

Example C–6 illustrates the use of the `folder` element.

Example C–6  Example of folder

```xml
<folder id="groupSpaceFolder">
  <attributes>
    <attribute value="GROUP_SPACE_FOLDER.TITLE" attributeId="Title" isKey="true"/>
    <attribute value="GROUP_SPACE_FOLDER.DESCRIPTION" attributeId="Description" isKey="true"/>
    <attribute value="GROUP_SPACE_FOLDER.KEYWORDS" attributeId="Subject" isKey="true"/>
  </attributes>
  <contents>
    ...
  </contents>
</folder>
```

Note: In addition to `folder` and `contents` elements, `customFolder` and `customContent` elements are displayed in the default catalog definition file. These elements are used to define content generated using Resource Catalog adapters.

You can comment out existing `customFolder` and `customContent` elements in your catalog definition file if required, but not add new ones. This is because you must use Resource Catalog APIs to define content in these elements and the required APIs are not exposed in this release.

Table C–4 describes the keywords of the `folder` element.

Table C–4  folder Keywords

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The identifier used when referring to this folder.</td>
</tr>
<tr>
<td>name</td>
<td>The displayed name of the folder.</td>
</tr>
<tr>
<td>description</td>
<td>Text that describes the folder.</td>
</tr>
</tbody>
</table>
C.4.4 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–8 illustrates the use of the attributes element.

Example C–7  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–5 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.4.5 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.4.4, "attributes." Oracle 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.4.8, "schema."

Example C–8 illustrates the use of the attribute element.

**Example C–8 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–6 describes the keywords of the attribute element.

**Table C–6 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 12.2.4, "How to Define Task Flow Parameters and Attributes of the Enclosing Show Detail Frames for Task Flows."

**C.4.6 resource**

The resource element is used to declare an individual resource, such as a task flow, in the catalog. Example C–9 illustrates the use of the resource element.
**Example C–9  Example of resource**

```xml
<resource id='announcements-main'  repository='application.classpath'
path='announcement-view.jar/ADF_TaskFlow/oracle+webcenter+collab+announcement+view+taskflows+main-view-definition.xml#announcement-main-view'/>
<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>
</resource>
```

Table C–7 describes the keywords of the resource element. For information about including attributes for elements, see Section C.4.4, "attributes."

### Table C–7 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>name</td>
<td>The name of the element.</td>
</tr>
<tr>
<td>description</td>
<td>The text describing the element.</td>
</tr>
<tr>
<td>repository</td>
<td>Name of the connection used to lookup the resource.</td>
</tr>
</tbody>
</table>
| path         | The path to the resource. For a task flow in an ADF Library, the path is of the following form:  

`path_to_jar/ADF_TaskFlow/task_flow_path`

To obtain the `task_flow_path`, create a file system connection in Application Connections and navigate to the task flow in your ADF Library. The tooltip for the task flow shows the fully qualified ID of the task flow. Take this value and replace all occurrences of "/" with "+". For example:

**Tooltip:**

oracle/webcenter/collab/announcement/view/taskflows/main-view-definition.xml#announcement-main-view

**task_flow_path:**

oracle+webcenter+collab+announcement+view+taskflows+main-view-definition.xml#announcement-main-view

### C.4.7 component

The component element is used to include a custom XML resource, such as a simple ADF Faces component or a YouTube video, in the catalog. The component element must be located within the contents section. You can add more than one component element 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, within a CDATA (character data) tag. To
generate an ADF Faces ID for the component at runtime, set the id on the component to # (hash symbol).

**Example C–10** shows the code to add a Panel Customizable component to the Resource Catalog.

**Example C–10  Panel Customizable Component in the Catalog Definition File**

```xml
<!-- panelCustomizable -->
<component id="pc" factoryClass="oracle.adf.rc.component.XmlComponentFactory">
    <attributes>
        <attribute attributeId="Title" value="Box"/>
        <attribute attributeId="Description" value="Container for other objects"/>
        <attribute attributeId="IconURI" value="/adf/webcenter/panelcustomizable_qualifier.png"/>
    </attributes>
    <parameters>
        <parameter id="xml">
            <![CDATA[
            ]]>
        </parameter>
    </parameters>
</component>
```

Table C–8 describes the keywords of the component element. For information about including attributes for elements, see Section C.4.4, "attributes."

**Table C–8  component 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 definition.</td>
</tr>
<tr>
<td>factoryClass</td>
<td>The name of a Java class that implements the oracle.adf.rc.component.ComponentFactory interface and generates a resource catalog item based on a set of parameters. Each ComponentFactory implementation defines the set of parameters it requires. For example, XmlComponentFactory requires a single parameter named xml and the value of the parameter is a block of XML representing a component or components that can be inserted into a page.</td>
</tr>
</tbody>
</table>

For more information, see Section 12.4, "Adding XML Content to the Resource Catalog."

### C.4.8 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.

**Example C–11** illustrates the use of the schema element.

**Example C–11  Example of schema**

```xml
<schema>
    <descriptor searchable="true" labelKey="TITLE.PROMPT_KEY" endUserVisible="true" shortLabelKey="TITLE.SHORT_PROMPT_KEY" attributeId="Title" multivalue="false"/>
</schema>
```
endUserVisible="true" shortLabelKey="DESCRIPTION.SHORT_PROMPT_KEY"
attributeId='Description' multivalue='false'/>
<descriptor searchable="true" labelKey='SUBJECT.PROMPT_KEY' endUserVisible='true'
shortLabelKey='SUBJECT.SHORT_PROMPT_KEY' attributeId='Subject'
multivalue='true'/>
<descriptor searchable="true" labelKey='WEBCENTER_SERVICE_ID.LABEL'
endUserVisible='false' shortLabelKey='WEBCENTER_SERVICE_ID.SHORT_LABEL'
attributeId='WEBCENTER_SERVICE_ID' multivalue='false'/>
<descriptor searchable="false" labelKey='TOOL_TIP.LABEL' endUserVisible='false'
shortLabelKey='TOOL_TIP.SHORT_LABEL' attributeId='ToolTip' multivalue='false'/>
<descriptor searchable="false" labelKey='ICON_URI.LABEL' endUserVisible='false'
shortLabelKey='ICON_URI.SHORT_LABEL' attributeId='IconURI' multivalue='false'/>
</schema>

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

Task flows are reusable building blocks in Oracle ADF. Oracle WebCenter provides task flows for you, or you can build your own task flows and add them to your WebCenter application pages. When you add task flows to customizable pages (that is, Oracle 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 publications. Failure to do so can result in problems when the task flows are consumed in WebCenter pages, such as too much white space, too many scroll bars, or slow, jerky, and unpredictable responses when displaying large data sets.

To prevent these issues and provide a consistent user experience in terms of look and feel, layout design, and interaction, follow the instructions in this document for:

- Creating highly performant, cost-effective task flows
- Consuming these task flows effectively in your application by implementing task flow parameters and customization and personalization options

This appendix contains the following topics:

- Section D.1, "Guidelines for Effective Geometry Management and Pagination"
- Section D.2, "Guidelines for Efficient Use of Task Flow Parameters and Customization and Personalization Options"
- Section D.3, "Example of a WebCenter Application Containing a Task Flow Created By Following the Guidelines"
- Section D.4, "Conclusion"

D.1 Guidelines for Effective Geometry Management and Pagination

Effective geometry management results in efficient sizing and placement of task flows on the page. If the task flows are not designed properly, you end up with pages that have a slow response time or lots of unnecessary scroll bars or white spaces between components. For example, if you include a task flow containing very little data inside a container that stretches it (for example, a Show Detail Frame component), there is a lot of white space below the data in the task flow. Poor design can also cause scrolling to be slow and unpredictable in a task flow with a table displaying a large amount of data. To avoid such issues, be sure to design your task flows so that they flow and use a conventional pagination model when displaying large data sets.
Follow the guidelines in this section when designing new task flows for WebCenter applications. Also, be sure to adhere to these guidelines when you edit existing WebCenter 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 *stretched* 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](image)

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**

```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"
    autoHeightRows="#{pageFlowScope.recentPagesBean.number_of_items_to_show_at_a_time}"
    contentDelivery="immediate"
    fetchSize="#{pageFlowScope.recentPagesBean.number_of_items_to_query}"
/>

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 WebCenter Application Containing a Task Flow Created By Following the Guidelines."

To learn more about the `autoHeightRows` attribute, including requirements such as setting the `contentDelivery` attribute to `immediate`, see the tag documentation for `<af:table>` in the Oracle Fusion Middleware Tag Reference for Oracle ADF Faces.

**D.1.1.2 Example 2: Oracle ADF Faces Iterator Component Inside a Panel Group Layout with a Scrolling Layout**

Add a Panel Group Layout component within the task flow and set its `layout` attribute to `scroll`. Then add an Iterator inside the Panel Group Layout. Add your content inside the Iterator. The content from the Iterator flows inside the Panel Group Layout.

Since the height of the Panel Group Layout depends on the content from the Iterator, specify a minimum and maximum height for the component by using `min-height` and `max-height` in the Panel Group Layout's `inlineStyle` attribute.

**D.1.2 Guideline 2: Limit the Number of Records Displayed in a Task Flow**

To avoid a slow and jerky response when scrolling large data sets in your task flow, you must limit the number of records to be fetched from the data source at a time. For example, if you have included an ADF faces Table component inside your task flow,
limit the number of records to be displayed in the table. Use the `fetchSize` attribute on the `Table` to specify the number of records to be fetched from the data source in a single query. If you specify a fetch size of $N$, only $N$ records are sent to the client from the data source.

The `fetchSize` and `autoHeightRows` attributes together control the number of rows displayed in the table. The relationship between these attributes is as follows:

- If `fetchSize` equals `autoHeightRows`, all records are displayed in the table.
- If `fetchSize` is less than `autoHeightRows`, the table is cropped at the `fetchSize` value.
- If `fetchSize` is greater than `autoHeightRows`, you can decide whether to display only as many records as the `autoHeightRows` value or provide pagination controls to display the remaining records. For example, you might include a More link at the bottom of the task flow or include pagination using Previous and Next links. For more information, see Section D.3, "Example of a WebCenter Application Containing a Task Flow Created By Following the Guidelines."

Figure D–2 shows the difference between two task flows containing tables with equal `autoHeightRows` and `fetchSize` values of 10 and 5. The task flows wrap to the number of records displayed.

Figure D–2 Difference Between Task Flows with Different Fetch Size Values

Instead of specifying a fixed value for `fetchSize`, you can provide an EL value so that the attribute is populated based on some logic you specify. You can enable customization and personalization on the task flow and provide an option for users to specify a value for this attribute. A value specified using the personalization option overrides a value specified using the customization option.

The following example shows the attributes of a sample ADF Faces `Table` component that is included inside a task flow called `recent-pages-task-flow-definition`. The `fetchSize` attribute is set to an EL expression that references a Java bean, `recentPagesBean`. This bean contains the logic to populate this attribute with a customization value provided by the end user.

```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'
```
autoHeightRows="#{pageFlowScope.recentPagesBean.number_of_items_to_show_at_a_time}"
    contentDelivery="immediate"
        fetchSize="#{pageFlowScope.recentPagesBean.number_of_items_to_query}">
</af:table>

For a detailed description of the sample task flow displaying a customization option for specifying the fetchSize value, see Section D.3, "Example of a WebCenter Application Containing a Task Flow Created By Following the Guidelines."

To learn more about the fetchSize attribute, see the tag documentation for <af:table> in the Oracle Fusion Middleware Tag Reference for Oracle ADF Faces.

D.1.3 Guideline 3: Specify a Minimum Height for the Task Flow

You can ensure that the task flow wraps to the height of the content by adhering to the first guideline, described in Section D.1.1, "Guideline 1: Create Task Flows that Flow."

To prevent the task flow from collapsing completely when there are no records, or seeming too short when there are only one or two records, specify a minimum height for the content.

For example, if your task flow contains a table with an autoHeightRows value specified, you can specify a min-height value using the table's inlineStyle attribute.

Figure D–3 shows a task flow displayed with a reasonable height even though it has only two rows.

Figure D–3  Task Flow with a Minimum Height Specified

D.1.4 Guideline 4: Include UI for Accessing Data Beyond the Display Limit

If a task flow is showing ten records, but the data source contains 100 records, you can design the task flow to do one of the following depending on the nature of data in the task flow:

- Display only ten records.
  
  For example, a Popular Discussion Topics task flow may use this first option if it is considered sufficient to show the ten most popular topics and not necessary to let users see even the eleventh most popular topic.

- Show a More link at the bottom of the task flow that opens a view with the ability to show more records.
  
  For example, the People Connections task flow on a WebCenter Spaces page displays a More link at the bottom.
Guidelines for Effective Geometry Management and Pagination

- 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**

<table>
<thead>
<tr>
<th>Watched Topics</th>
<th>Popular Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bicycle Parking</td>
<td>WebCenter Release Date</td>
</tr>
<tr>
<td>Discount tickets to the Fair</td>
<td>Bicycle Parking</td>
</tr>
<tr>
<td>New England Pros</td>
<td>Suggestions for external ha...</td>
</tr>
<tr>
<td>Things to do in the Bay Area</td>
<td>Where to find good food inside</td>
</tr>
<tr>
<td>Suggestions for external ha...</td>
<td>Safar 4-release issues</td>
</tr>
</tbody>
</table>

**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 WebCenter 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 Oracle 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.
### 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>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 Spaces, you can use a parameter for a context like group space name. Precedence: 3 This value overrides built-in values in the task flow.</td>
<td>Power users (or users with Edit or Customize permission) can see parameters in Oracle Composer's Component Properties dialog while editing a page. In WebCenter Spaces, moderators can see and edit task flow parameters. When a WebCenter task flow is exposed as a portlet in Oracle Portal, 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.

---

**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/Prefere nces</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, group space name), customization overrides that value.</td>
<td>Business users (or users with Edit or Customize permission) editing a page in Oracle Composer can see the customization or preferences options. Typically, customization or preferences options are available on the chrome surrounding the task flow in Edit mode. However, users with Edit or Customize permission can also see these options on the chrome in View mode of the page. When a WebCenter task flow is exposed as a portlet in Oracle Portal, 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 task flows are exposed as portlets in Oracle Portal, 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>
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 Oracle Composer. To provide a good user experience, you must adhere to the following guidelines when creating parameters:

■ Provide a display name that is meaningful.
■ Provide a description for each parameter, along with examples and the possible values it can take.

The following example shows the code to define a parameter called `layoutStyle` in the task flow definition file:

```
<input-parameter-definition>
  <name>layoutStyle</name>
  <description>Used to control whether the documents must display as a list, with icons, or with document details. Allowed values are list, icon, and detail.</description>
  <display-name>Layout Style</display-name>
</input-parameter-definition>
```

D.2.2 Guidelines for Implementing Customization

Users with Edit or Customize permission can access customization options from the chrome surrounding the task flow.

You must adhere to the following guidelines while implementing customization in your task flow:

■ Customization options must be a subset of the task flow's parameters.
■ Implement a view fragment within the task flow and expose an outcome to open this view in a separate dialog. For example, the control flow rule to invoke a view called `advancedEditPopupView` in a dialog is as follows:

```
<control-flow-case>
  <from-outcome>dialog:advancedEditPopup</from-outcome>
  <to-activity-id>advancedEditPopupView</to-activity-id>
</control-flow-case>
```

Adhering to this guideline is mandatory for new task flows.

■ Implement a view fragment within the task flow and expose an outcome to transition inline to this view. For example, the control flow rule to invoke a view called `advancedEditView` within the same screen is as follows:

```
<control-flow-case>
  <from-outcome>advancedEdit</from-outcome>
  <to-activity-id>advancedEditView</to-activity-id>
</control-flow-case>
```

Although you can implement a task flow in this way, it is recommended that you expose an outcome to open the view in a dialog.

■ Ensure that customizations are saved in the MDS level set up by the application. Unlike task flow parameters, which are stored in the task flow definition itself, customizations are saved in the Metadata Storage (MDS). If you do not explicitly specify the MDS layer to be used to save task flow customizations, customization metadata and the consuming application metadata may get stored in different layers.
To ensure that customizations are saved to the same layer as application metadata, you must clone the application’s MDS session in the task flow’s preference file and reference that while configuring customization. For an example of how this is implemented, see Section D.3, "Example of a WebCenter Application Containing a Task Flow Created By Following the Guidelines."

D.2.3 Guidelines for Implementing Personalization

All users viewing the page can access personalization options from the chrome surrounding the task flow in View mode of the page.

You must adhere to the following guidelines while implementing personalization in your task flow:

- Ensure that personalization options are a subset of the customization options you define on the task flow.

- If a key aspect of the task flow is to show objects—for example, documents, users, or requests—then you must include a personalization option for users to set the maximum number of objects to display.

- Implement a view fragment within the task flow, and expose an outcome to open this view in a dialog. For example, the control flow rule to invoke a view called simpleEditPopupView in a dialog is as follows:

  ```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 "Example of a WebCenter 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 WebCenter Application Containing a Task Flow Created By Following the Guidelines

As a companion to this document, a sample application was developed using Oracle WebCenter Framework. This sample application, called RecentPages, is used as an example to illustrate how you can build a near-perfect task flow by adhering to the guidelines in this document. The RecentPages application contains a customizable page, Welcome.jspx, which contains a task flow named recent-pages-taskflow-definition created by following all the required guidelines.

This section only provides a high-level description of how the task flow was implemented. For detailed information about using task flows and Oracle Composer, see the following guides:

- Oracle Fusion Middleware User’s Guide for Oracle WebCenter, for information about editing, customizing, and personalizing task flows in Oracle Composer.

The RecentPages sample application is available for download from the Oracle WebCenter Suite 11g Demonstrations and Samples page on OTN at:

http://webcenter.oracle.com

The RecentPages WebCenter application contains the following projects:

- A Model project
- A ViewController project named RecentPagesTaskFlow that contains the recent-pages-task-flow-definition task flow
- Another ViewController project named WebPages that contains the JSPX page in which the task flow is consumed

D.3.1 The RecentPagesTaskFlow Project

This project contains the task flow created by following the guidelines. The files of significance in this project are as follows:

- The recent-pages-task-flow-definition.xml file
- The mainView.jsff fragment
- The simpleEditPopupView.jspx page
- The advancedEditPopupView.jspx page
- The recentPagesBean Java bean
- The Preference Java bean
D.3.1.1 The recent-pages-task-flow-definition.xml File

This is an Oracle ADF task flow definition containing three views: mainView, simpleEditPopupView, and advancedEditPopupView. The task flow is designed such that mainView displays a list of recently created pages in the application, simpleEditPopupView displays a personalization option to all users, and advancedEditPopupView displays customization options to users with Edit or Customize permission.

The following table breaks down the code in recent-pages-task-flow-definition.xml and explains each snippet.

The outcomes from mainView to simpleEditPopupView and advancedEditPopupView are called dialog:simpleEditPopupview and dialog:advancedEditPopupView respectively. This is to ensure that the views open in separate dialogs, not inline. In addition, this task flow provides Refresh, Previous, and Next outcomes on mainView. The simpleEditPopupView and advancedEditPopupView views provide Save and Cancel outcomes.

The following table describes significant sections of code in the recent-pages-task-flow-definition.xml file.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;default-activity&gt;mainView&lt;/default-activity&gt;</code></td>
<td>Defining the default view for the task flow:mainView</td>
</tr>
<tr>
<td><code>&lt;managed-bean&gt;</code></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><code>&lt;managed-bean-name&gt;recentPagesBean&lt;/managed-bean-name&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;managed-bean-class&gt;view.RecentPagesBean&lt;/managed-bean-class&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;managed-bean-scope&gt;pageFlow&lt;/managed-bean-scope&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;/managed-bean&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;view id=&quot;mainView&quot;&gt;</code></td>
<td>Mapping task flow views to pages or page fragments. For example, mainView is associated with mainView.jsff.</td>
</tr>
<tr>
<td><code>&lt;page&gt;/mainView.jsff&lt;/page&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;/view&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;view id=&quot;simpleEditPopupView&quot;&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;page&gt;/simpleEditPopupView.jspx&lt;/page&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;/view&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;view id=&quot;advancedEditPopupView&quot;&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;page&gt;/advancedEditPopupView.jspx&lt;/page&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;/view&gt;</code></td>
<td></td>
</tr>
</tbody>
</table>
**Example of a WebCenter Application Containing a Task Flow Created By Following the Guidelines**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
</table>
| <control-flow-rule>
  <from-activity-id>mainView</from-activity-id>
  <control-flow-case>
    <from-outcome>refresh</from-outcome>
    <to-activity-id>mainView</to-activity-id>
  </control-flow-case>
  <control-flow-case>
    <from-outcome>dialog:-simpleEditPopup</from-outcome>
    <to-activity-id>simpleEditPopupView</to-activity-id>
  </control-flow-case>
  <control-flow-case>
    <from-outcome>next</from-outcome>
    <to-activity-id>mainView</to-activity-id>
  </control-flow-case>
  <control-flow-case>
    <from-outcome>previous</from-outcome>
    <to-activity-id>mainView</to-activity-id>
  </control-flow-case>
  <control-flow-case>
    <from-outcome>dialog:advancedEditPopup</from-outcome>
    <to-activity-id>advancedEditPopupView</to-activity-id>
  </control-flow-case>
</control-flow-rule> | Control flow rules from `mainView`. Each control flow case defines an outcome and a corresponding target view. |
| <control-flow-rule>
  <from-activity-id>simpleEditPopupView</from-activity-id>
  <control-flow-case>
    <from-outcome>save</from-outcome>
    <to-activity-id>mainView</to-activity-id>
  </control-flow-case>
  <control-flow-case>
    <from-outcome>cancel</from-outcome>
    <to-activity-id>mainView</to-activity-id>
  </control-flow-case>
</control-flow-rule> | Control flow rules from `simpleEditPopupView`. Each control flow case defines an outcome and a corresponding target view. |
| <control-flow-rule>
  <from-activity-id>advancedEditPopupView</from-activity-id>
  <control-flow-case>
    <from-outcome>save</from-outcome>
    <to-activity-id>mainView</to-activity-id>
  </control-flow-case>
  <control-flow-case>
    <from-outcome>cancel</from-outcome>
    <to-activity-id>mainView</to-activity-id>
  </control-flow-case>
</control-flow-rule> | Control flow rules from `advancedEditPopupView`. Each control flow case defines an outcome and a corresponding target view. |

**Table D–2 (Cont.) Sections Of Code in the Task Flow Definition File**

---

**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>
</table>
| `<af:panelGroupLayout id="pg11" layout="vertical">  
  <af:outputText value="Shows recent pages in the application" id="ot1"/>` | A single top-level component-Panel Group Layout. This component is designed to ensure that the task flow content flows vertically and always stretches horizontally. This allows geometry management. |
### Table D-3 (Cont.) The mainView Fragment Code

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;af:table value=&quot;#{pageFlowScope.recentPagesBean.pages}&quot; var=&quot;row&quot;</td>
<td>A table component that is a child of the Panel Group Layout.</td>
</tr>
<tr>
<td>rowBandingInterval=&quot;0&quot; id=&quot;t1&quot; rowSelection='none'</td>
<td>Table attributes:</td>
</tr>
<tr>
<td>columnBandingInterval=&quot;0&quot; width=&quot;100%&quot;</td>
<td>■ autoHeightRows (N): The number of items to display in the table at one time. An EL value is specified here such that a Java bean, recentPagesBean, populates this attribute with a value specified by the user at runtime. If there are more than N items to display, a scrollbar appears on the task flow.</td>
</tr>
<tr>
<td>inlineStyle=&quot;border:none;width:100%;&quot;</td>
<td>■ fetchSize (M): The maximum number of items to be fetched from the data source. An EL value is specified here such that a Java bean, recentPagesBean, populates this attribute with a value specified by the user at runtime. If M is greater than N, pagination controls appear on the task flow to enable users to view the remaining items.</td>
</tr>
<tr>
<td>columnStretching=&quot;last&quot;</td>
<td></td>
</tr>
<tr>
<td>horizontalGridVisible=&quot;true&quot; verticalGridVisible=&quot;false&quot;</td>
<td></td>
</tr>
</tbody>
</table>
|     autoHeightRows="#{pageFlowScope.recentPagesBean.number_of_items_to_show_at_a_time}" contentDelivery="immediate" fetchSize="#{pageFlowScope.recentPagesBean.number_of_items_to_query}"
| <af:column sortable="false" headerText="#{null}" align="start" id="c1" | 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. |
|     <af:panelGroupLayout id="pg11" styleClass="nowrap"           |             |
|     inlineStyle="padding-top:2px; font-weight:bold;"             |             |
|     <af:outputText value="#{row.title}" id="ot4" />             |             |
|     <af:goLink destination="/faces#{row.contentMRef}" text=">>" targetFrame="_blank" id="gl1" /> |             |
|     </af:panelGroupLayout>                                        |             |
|     <af:panelGroupLayout id="pg10" styleClass="AFFieldTextDisabled nowrap" inlineStyle="padding-top:2px"> |             |
|     <af:outputText value="Created on " id="ot7" />               |             |
|     <af:outputText value="#{row.createDate}" id="ot8" />        |             |
|     <af:outputText value=" by #{row.createdBy}" id="ot9" />    |             |
|     </af:panelGroupLayout>                                        |             |
| </af:column>                                                      |             |
Table D–3  (Cont.) The mainView Fragment Code

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;af:panelGroupLayout id=&quot;pgl2&quot; halign=&quot;end&quot; layout=&quot;vertical&quot;</td>
<td>Pagination controls to display records beyond those that are displayed in the table.</td>
</tr>
<tr>
<td>inlineStyle=&quot;margin:1px&quot; rendered=&quot;#{pageFlowScope.recentPagesBean.showPageControl}&quot;</td>
<td>The recentPagesBean code defines what is displayed to users at runtime.</td>
</tr>
<tr>
<td>&lt;af:panelGroupLayout id=&quot;pgl3&quot;&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;af:outputText value=&quot;#{pageFlowScope.recentPagesBean.footerText}&quot;</td>
<td></td>
</tr>
<tr>
<td>id=&quot;ot2&quot;/&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;af:commandLink id=&quot;cl1&quot; actionListener=&quot;#{pageFlowScope.recentPagesBean.previous}&quot;</td>
<td></td>
</tr>
<tr>
<td>disabled=&quot;#{not pageFlowScope.recentPagesBean.showPrevious}&quot; action=&quot;previous&quot;&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;af:image source=&quot;#{pageFlowScope.recentPagesBean.showPrevious ?</td>
<td></td>
</tr>
<tr>
<td>'/adf/webcenter/shuttleleft_ena.png' : '/adf/webcenter/shuttleleft_dis.png'}&quot; id=&quot;il&quot; shortDesc=&quot;previous&quot;/&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;/af:commandLink&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;af:commandLink id=&quot;cl2&quot; actionListener=&quot;#{pageFlowScope.recentPagesBean.next}&quot;</td>
<td></td>
</tr>
<tr>
<td>disabled=&quot;#{not pageFlowScope.recentPagesBean.showNext}&quot; action=&quot;next&quot;&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;af:image source=&quot;#{pageFlowScope.recentPagesBean.showNext ?</td>
<td></td>
</tr>
<tr>
<td>'/adf/webcenter/shuttleright_ena.png' : '/adf/webcenter/shuttleright_dis.png'}&quot; id=&quot;il2&quot; shortDesc=&quot;next&quot;/&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;/af:commandLink&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;/af:panelGroupLayout&gt;</td>
<td></td>
</tr>
</tbody>
</table>

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.
Example of a WebCenter Application Containing a Task Flow Created By Following the Guidelines

Figure D–6  The simpleEditPopupView Page

The following table describes significant sections of code in the simpleEditPopupView.jspx page.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;af:panelStretchLayout id=&quot;psl1&quot;&gt;</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>&lt;f:facet name=&quot;center&quot;&gt;</td>
<td>The Panel Form Layout contains an Input Text component, which is populated with a value returned by recentPagesBean.</td>
</tr>
<tr>
<td>&lt;af:group id=&quot;group1&quot;&gt;</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>&lt;af:panelFormLayout id=&quot;pfl1&quot;&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;af:inputText label=&quot;Number of Items to show at a time&quot;</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>&lt;/af:panelFormLayout&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;af:panelGroupLayout id=&quot;pgl2&quot;</td>
<td></td>
</tr>
<tr>
<td>layout=&quot;horizontal&quot;</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>action=&quot;#{pageFlowScope.recentPagesBean.saveSettings()}&quot;</td>
<td></td>
</tr>
<tr>
<td>actionListener=&quot;#{pageFlowScope.recentPagesBean.cancel()}&quot;</td>
<td></td>
</tr>
<tr>
<td>&lt;/af:panelGroupLayout&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;/af:group&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;/f:facet&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;/af:panelStretchLayout&gt;</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.

The following table describes significant sections of code in the advancedPopupView.jspx page.
Example of a WebCenter Application Containing a Task Flow Created By Following the Guidelines

**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.

### 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.</td>
</tr>
<tr>
<td><code>&lt;f:facet name=&quot;center&quot;&gt;</code></td>
<td>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;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; id=&quot;it1&quot; value=&quot;#{pageFlowScope.recentPagesBean.number_of_items_to_show_at_a_time}&quot; columns=&quot;3&quot;/&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;af:inputText label=&quot;Number of Items to Query&quot; id=&quot;it2&quot; value=&quot;#{pageFlowScope.recentPagesBean.number_of_items_to_query}&quot; columns=&quot;3&quot;/&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;af:selectBooleanCheckbox label=&quot;Show Page Control&quot; id=&quot;sbc1&quot; value=&quot;#{pageFlowScope.recentPagesBean.showPageControl}&quot;/&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;af:panelGroupLayout id=&quot;pgl2&quot; layout=&quot;horizontal&quot; halign=&quot;right&quot; inlineStyle=&quot;width:500px;&quot;&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;af:commandButton text=&quot;Save&quot; id=&quot;cb1&quot; action=&quot;#{pageFlowScope.recentPagesBean.saveSettings}&quot;/&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;af:commandButton text=&quot;Cancel&quot; id=&quot;cb2&quot; action=&quot;cancel&quot; actionListener=&quot;#{pageFlowScope.recentPagesBean.cancel}&quot;/&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;/af:panelFormLayout&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>
You can find the code for this bean in the RecentPages application that is available for download from the Oracle WebCenter Suite 11g Demonstrations and Samples page on OTN at:

http://webcenter.oracle.com

The following table describes significant sections of code in the recentPagesBean.java file.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>private String _getPreferenceFileName()</td>
<td>Logic for identifying the task flow and retrieving the instance-level preference file for the currently running task flow.</td>
</tr>
<tr>
<td>String taskflowId =</td>
<td></td>
</tr>
<tr>
<td>ControllerContextFwk.getInstanceFwk().getCurrentViewPortFwk().getClientId();</td>
<td></td>
</tr>
<tr>
<td>if (taskflowId == null)</td>
<td></td>
</tr>
<tr>
<td>taskflowId =</td>
<td></td>
</tr>
<tr>
<td>ControllerContextFwk.getInstanceFwk().getCurrentViewPort().getParentViewPort().getClientId();</td>
<td></td>
</tr>
<tr>
<td>String preferenceFileName =</td>
<td></td>
</tr>
<tr>
<td>PREFERENCE_BASE + taskflowId.replace(“.”, “_”)+ “.xml”;</td>
<td></td>
</tr>
<tr>
<td>return preferenceFileName;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>private static PageServiceBean</td>
<td></td>
</tr>
<tr>
<td>getPageServiceBean()</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>String beanName = “pageServiceBean”;</td>
<td></td>
</tr>
<tr>
<td>Object bean = null;</td>
<td></td>
</tr>
<tr>
<td>FacesContext ftx =</td>
<td></td>
</tr>
<tr>
<td>FacesContext.getCurrentInstance();</td>
<td></td>
</tr>
<tr>
<td>bean =</td>
<td></td>
</tr>
<tr>
<td>ftx.getExternalContext().getRequestMap().get(beanName);</td>
<td></td>
</tr>
<tr>
<td>if (bean == null)</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>ELContext elContext = ftx.getELContext();</td>
<td></td>
</tr>
<tr>
<td>ExpressionFactory exFactory =</td>
<td></td>
</tr>
<tr>
<td>ftx.getApplication().getExpressionFactory();</td>
<td></td>
</tr>
<tr>
<td>ValueExpression expr =</td>
<td></td>
</tr>
<tr>
<td>exFactory.createValueExpression(elContext,</td>
<td></td>
</tr>
<tr>
<td>“#{“ + beanName + “”},PageServiceBean.class);</td>
<td></td>
</tr>
<tr>
<td>bean = expr.getValue(elContext);</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>return (PageServiceBean) bean;</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 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>{</td>
<td></td>
</tr>
<tr>
<td>if (number_of_items_to_query != null &amp;&amp; !number_of_items_to_query.isEmpty()) { }</td>
<td></td>
</tr>
<tr>
<td>Preference.setNumberOfItemsToQuery(_getPreferenceFileName(), Integer.parseInt(number_of_items_to_query));</td>
<td></td>
</tr>
<tr>
<td>if (this.number_of_items_to_show_at_a_time != null &amp;&amp; !this.number_of_items_to_show_at_a_time.isEmpty()) { }</td>
<td></td>
</tr>
<tr>
<td>Preference.setNumberOfItemsToShow(_getPreferenceFileName(), Integer.parseInt(number_of_items_to_show_at_a_time));</td>
<td></td>
</tr>
<tr>
<td>if (_showPageControl != null) { }</td>
<td></td>
</tr>
<tr>
<td>Preference.setShowPageControl(_getPreferenceFileName(), _showPageControl);</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>AdfFacesContext.getCurrentInstance().returnFromDialog(null, null);</td>
<td></td>
</tr>
<tr>
<td>return &quot;save&quot;;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>public void setNumber_of_items_to_show_at_a_time(String number_of_items_to_show_at_a_time) { }</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>this.number_of_items_to_show_at_a_time = number_of_items_to_show_at_a_time;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>public String getNumber_of_items_to_show_at_a_time() { }</td>
<td></td>
</tr>
<tr>
<td>String query =</td>
<td></td>
</tr>
<tr>
<td>Integer.toString(Preference.getNumberOfItemsToShow(_getPreferenceFileName()));</td>
<td></td>
</tr>
<tr>
<td>return query;</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>
Table D–6 (Cont.) The recentPagesBean Java Bean Code

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
</table>
| `public int getTableSize()`  ```
   {
   int tableSize = -1;
   try
   {
   PageServiceBean psb = getPageServiceBean();
   List<PageDef> pages = psb.getPages();
   tableSize = pages.size();
   }
   catch (Exception e)
   {
   System.out.println(e.toString());
   }
   return tableSize;
   }```
| Logic to set the table size to the number of records retrieved. |
| `public int getStart()`  ```
   {
   String items_to_query =
   this.getNumber_of_items_to_query();
   int offset = Integer.parseInt(items_to_query);
   return (getPageNumber() - 1) * offset;
   }```
|  |
| `public int getPageNumber()`  ```
   {
   Object page_num =
   RequestContext.getCurrentInstance().getPageFlowScope().get("page_num");
   int local_page_num = 1;
   if (page_num == null || (Integer) page_num <= 0)
   {
   local_page_num = 1;
   }
   else
   {
   local_page_num = (Integer) page_num;
   }
   return local_page_num;
   }```
|  |
| `public void cancel(ActionEvent actionEvent)`  ```
   {
   AdfFacesContext.getCurrentInstance().returnFromDialog(null, null);
   }```
| Logic to define behavior of Cancel button in the personalization or customization dialog. |
### Table D–6  (Cont.) The recentPagesBean Java Bean Code

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
</table>
| public void previous(ActionEvent actionEvent) {  
  int page_num = getPageNumber();  
  page_num--;  
  if (page_num <= 0)  
  {  
    page_num = 1;  
  }  
  RequestContext.getCurrentInstance().getPageFlowScope().put("page_num", page_num);  
} | Logic to define behavior of Previous action in the task flow's pagination control. |
| public void next(ActionEvent actionEvent) {  
  int page_num = getPageNumber();  
  page_num++;  
  String items_to_query = this.getNumber_of_items_to_query();  
  int offset = Integer.parseInt(items_to_query);  
  int total_page_num = getTotalPageNumber();  
  if (page_num > total_page_num)  
  {  
    page_num = this.getTableSize() / offset;  
  }  
  RequestContext.getCurrentInstance().getPageFlowScope().put("page_num", page_num);  
} | Logic to define behavior of Next action in the task flow's pagination control. |
| public boolean isShowPrevious() {  
  int page_num = getPageNumber();  
  if (page_num <= 1)  
  {  
    return false;  
  }  
  else  
  {  
    return true;  
  }  
} | Logic to render Previous icon |
| public boolean isShowNext() {  
  int page_num = getPageNumber();  
  if (page_num >= getTotalPageNumber())  
  {  
    return false;  
  }  
  else  
  {  
    return true;  
  }  
} | Logic to render Next icon. |
D.3.1.6 The Preference Bean

This Java bean is a wrapper that provides APIs to get and set attributes in the task flow preference file. It contains the logic for saving personalizations and customizations on this task flow instance in the same MDS layer as that used by the application. In this sample application, site-level customizations are stored in the mds/mdssys/cust/site/site directory. Therefore, the code in the Preference bean ensures that the application's MDS session is cloned and the task flow personalizations and customizations are stored in the same directory.

You can find the code for this bean in the RecentPages application that is available for download from the Oracle WebCenter Suite 11g Demonstrations and Samples page on OTN at:

http://webcenter.oracle.com

The following table describes significant sections of code in the Preference.java file.

Table D–7  The Preference Java Bean Code

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>private static MDSMessage _getClonedMDSMessage()</td>
<td>Logic to clone the application's MDS session.</td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>MDSInstance instance = (MDSInstance)</td>
<td></td>
</tr>
<tr>
<td>ADFContext.getCurrent().getMDSInstanceAsObject();</td>
<td></td>
</tr>
<tr>
<td>MDSMessage mdsSession = (MDSMessage)</td>
<td></td>
</tr>
<tr>
<td>ADFContext.getCurrent().getMDSMessageAsObject();</td>
<td></td>
</tr>
<tr>
<td>mdsSession =</td>
<td></td>
</tr>
<tr>
<td>instance.createSession(mdsSession.getSessionOptions(),</td>
<td></td>
</tr>
<tr>
<td>mdsSession.getUserStateHandler());</td>
<td></td>
</tr>
<tr>
<td>return mdsSession;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
</tbody>
</table>
Logic to read the preference file in the MDS session or create one, if required.
Example of a WebCenter Application Containing a Task Flow Created By Following the Guidelines

**Table D–7** (Cont.) The Preference Java Bean Code

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>public static void setNumberOfItemsToShow(String preferenceFileName, int numberOfItemsToShow)</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td>Logic to get and set the number of items to show in the table at a time.</td>
</tr>
<tr>
<td>MDSSession session = _getClonedMDSSession();</td>
<td></td>
</tr>
<tr>
<td>mo.getDocument().getElement().setAttribute(&quot;numberOfItemsToShow&quot;, Integer.toString(numberOfItemsToShow));</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>public static int getNumberOfItemsToShow(String preferenceFileName)</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>MDSSession session = _getClonedMDSSession();</td>
<td></td>
</tr>
<tr>
<td>mo.getDocument().getElement().getAttribute(&quot;numberOfItemsToShow&quot;);</td>
<td></td>
</tr>
<tr>
<td>return Integer.parseInt(show);</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>public static void setNumberOfItemsToQuery(String preferenceFileName, int _numberOfItemsToQuery)</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td>Logic to get and set the number of items to query from the data source.</td>
</tr>
<tr>
<td>MDSSession session = _getClonedMDSSession();</td>
<td></td>
</tr>
<tr>
<td>mo.getDocument().getElement().setAttribute(&quot;numberOfItemsToQuery&quot;, Integer.toString(_numberOfItemsToQuery));</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>public static int getNumberOfItemsToQuery(String preferenceFileName)</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>MDSSession session = _getClonedMDSSession();</td>
<td></td>
</tr>
<tr>
<td>mo.getDocument().getElement().getAttribute(&quot;numberOfItemsToQuery&quot;);</td>
<td></td>
</tr>
<tr>
<td>return Integer.parseInt(query);</td>
<td></td>
</tr>
<tr>
<td>}</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.recentpagestaskflowdefinition1.regionModel}" id="r2"/>
                                </cust:showDetailFrame>
                            </cust:panelCustomizable>
                        </pe:pageCustomizable>
                    </f:facet>
                    <f:facet name="start"/>
                    <f:facet name="end"/>
                </af:panelStretchLayout>
            </af:form>
        </af:document>
    </f:view>
</jsp:root>
```
At runtime, the page appears as shown in Figure D–7.

**Figure D–7  The Welcome.jspx Page**

D.3.2.2 The adf-config.xml File

You can add a task flow inside a Show Detail Frame component and display the task flow’s outcomes as actions on the chrome of the parent Show Detail Frame component. This is possible by defining custom actions on the parent Show Detail Frame component. You can define custom actions at the instance level where Custom Action components are added as child components of the Show Detail Frame, or at the global level where Custom Action elements are defined in the application’s adf-config.xml file. Global custom actions are enabled on all Show Detail Frame instances in the application, but are displayed only when a Show Detail Frame instance contains the task flow as its child.

In this application, custom actions corresponding to the task flow’s outcomes, dialog:simpleEditPopupView and dialog:advancedEditPopupView, are defined globally in the adf-config.xml file as follows:

```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 WebCenter 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 WebCenter application users, follow the guidelines in this document while implementing task flows to be consumed in the application.
In custom WebCenter applications, you can provide users with the ability to search the data, or a portion of the data, presented to them on a page. To achieve this type of functionality, use a web service data control to call the searching function of Oracle Secure Enterprise Search (SES).

For more information on Oracle SES, see the Oracle Secure Enterprise Search documentation that comes with the product.

E.1 Calling Oracle SES to Search Data

This appendix demonstrates how to create a web service data control to search SES data.

This involves the following steps:

- Section E.1.1, "Creating a Data Control"
- Section E.1.2, "Adding the Data Control on a Page"
- Section E.1.3, "Formatting the Output"

E.1.1 Creating a Data Control

You can leverage Oracle SES APIs using a data control to build custom UIs. To build a data control that uses Oracle Secure Enterprise Search:

1. After installing Oracle Secure Enterprise Search, create a custom WebCenter application in Oracle JDeveloper.
2. Right-click the project where you want to incorporate searching and select New from the context menu.
3. In the New Gallery, make sure to filter by All Technologies.
4. Expand Business Tier and select Web Services.
5. In the Items list, select Web Service Data Control and click OK.
6. Enter a name for the data control. The wizard page should look similar to Figure E–1.
7. Enter a Web Services Description Language (WSDL) URL. The WSDL URL for Oracle Secure Enterprise Search has the following form:


**Note:** When Oracle Secure Enterprise Search is installed, the endpoint listed in the WSDL points to a placeholder location. After the data control is created, you must correct the endpoint URL using the Structure panel. This step is described later in this procedure.

8. If the service is not automatically provided, then click **Services**.

9. Click **Next**.

10. From the Data Control Operations page, select the methods you want to expose by moving them from the **Available** list to the **Selected** list. For the purposes of this example, move `doOracleSimpleSearch` to the **Selected** list. When you are done, the page should look similar to Figure E–2.
11. Click **Finish**. To review the other pages of the wizard first, click **Next** until you reach the end of the wizard, and then click **Finish**.

12. In the Application Navigator, expand **Application Sources** and then **view**.

13. Click **DataControls.dcx** and its structure will appear in the Structure panel below the Application Navigator.

14. Right-click your data control in the Structure panel and select **Edit Web Service Connection** from the context menu.

15. The Edit Web Service Connection dialog appears as shown in **Figure E–3**. As noted earlier, the endpoint listed in the WSDL now points to a placeholder location. Enter again the same endpoint URL.
16. Click OK.

E.1.2 Adding the Data Control on a Page

This section describes how to drop your new data control onto a page.

1. To create a new page, right-click your project and select New from the context menu. Under Web Tier, select JSF. Under Items, select JSF Page, and click OK. On the Create JSP Page, accept the default settings for everything except File Name. Select a meaningful file name, and click OK.

2. You should see your data control listed under the Data Controls section of the Application Navigator. Expand the top level node of your data control and then expand Return and Return. Now you can see all the data that is returned by the data control.

3. For the purposes of this example, drag and drop resultElements onto your page.

4. Select Tables, ADF Read-only Table from the context menu that appears.

5. The Edit Table Columns dialog appears. (Figure E–4) Select lastModified and click the Delete icon. Click OK.
6. For the sake of simplicity, you can hard code the values for each parameter as shown in Table E–1.

**Table E–1 Parameter Values for Action Binding Editor**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>query</td>
<td>The string you want to search for, for example, oracle</td>
</tr>
<tr>
<td>startIndex</td>
<td>1</td>
</tr>
<tr>
<td>docsRequested</td>
<td>20</td>
</tr>
<tr>
<td>dupRemoved</td>
<td>false</td>
</tr>
<tr>
<td>dupMarked</td>
<td>false</td>
</tr>
<tr>
<td>returnCount</td>
<td>false</td>
</tr>
</tbody>
</table>

7. The dialog should look like Figure E–5. Click OK, and then click OK again.
8. Now you can run your page. In the Application Navigator, right-click the page and select Run from the context menu.

### E.1.3 Formatting the Output

At runtime, you can browse through your output and see how occurrences of a word, for example "oracle," are surrounded by `<b>` tags.

**Figure E–6** shows a small sample fragment of this output. Currently, the search results are not formatted because you have not yet converted the view component that renders this column to `OutputFormatted`.

**Figure E–6 Sample of Searched Output, Hits Unformatted**

<table>
<thead>
<tr>
<th>description</th>
<th>url</th>
</tr>
</thead>
</table>

To format the search output:

1. To format the output, return to Oracle JDeveloper and display the page containing the data control by clicking its tab in the Design view.

2. In the Structure panel, find and expand one of the columns that contains `oracle`; for example, the description column.

3. Right-click `af:outputText - #{row.snippet}` for that column.

4. In the Source view, change `af:outputText` to `af:outputFormatted`.

---

**Calling Oracle SES to Search Data**

**Figure E–5  Action Binding Editor**

![Action Binding Editor](image)

Select a data collection and the action you want your control to initiate. The control initiates the action on the data objects of the selected collection.

Data Collection:

![Data Collection](image)

Select an Iterator:

![Select an Iterator](image)

Operation:

![Operation](image)

Parameters:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Value</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>query</td>
<td>java.lang.String</td>
<td>oracle</td>
<td></td>
</tr>
<tr>
<td>startIndex</td>
<td>java.lang.Integer</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>docRequested</td>
<td>java.lang.Integer</td>
<td>20</td>
<td></td>
</tr>
</tbody>
</table>

Add Help | Ok | Cancel
5. Run the page again to see the changes. You should see output similar to that in Figure E–7. Notice how the `<b>` around the word `oracle` is now interpreted so that search hits appear in bold.

![Figure E–7 Sample of Searched Output, Hits Formatted](http://www.oreilly.com-python/C/C++/Basic/Oracle Business Accelerator for JD Edward/Oracle Magazine publishes all articles in printed form/Oracle's Applications Integration Architect/Oracle Data Integrator delivers unique new features)

6. Using the Structure panel, right-click and delete any extraneous columns of your table.

7. Repeat steps 1-5 for each of the remaining columns on your page.

8. Save and run the page again.

**E.1.3.1 Additional Formatting**

The rest of this section describes some other formatting features you can include.

1. You might see automatic pagination in the upper right corner of the table. The default number of rows for an Oracle ADF table is 10, but you chose to return 20 results.

   To enhance the page, you can provide an estimated count. From the Application Navigator, under Data Controls, drag and drop `estimatedHitCount` onto the page, before the table. Select Texts, ADF Output Text w/ Label from the context menu.

2. Another useful feature for this table would be a link to open the document. Drag a Go Link from under ADF Faces in the Component Palette and drop it right in front of `outputFormatted` of the description column. Note that you can perform this step a little more accurately in the Structure panel.

3. In the Property Inspector, change the Text attribute to Open. For the Destination attribute, enter the following: `#{row.url}`.

4. Click the Source tab. You should see something similar to the following in the source for your page:

   ```af:goLink text="Open" destination="#{row.url}" />

5. You could also add an input box and a Submit button to enable users to perform their own custom searches. Click the Design tab.

6. From the Application Navigator, under Data Controls, expand the Parameters node under your data control.

7. Drag `query` and drop it just above the left corner of your table. Select Texts, ADF Input Text w/ Label from the context menu.

8. Drag `doOracleSimpleSearch` and drop it just below the parameter input box you just created. Select Methods, ADF Button from the context menu.

9. To avoid getting an error when someone first runs the page, you must set a default value for the query parameter. Go to the page definition, `PageDef.xml`.
10. In the Structure panel, expand variables under executables and select 
doOracleSimpleSearch_query.

11. In the Property Inspector, select the DefaultValue property and enter a default search; for example, oracle.

12. Run the page. Enter a different search term in the box and click the button you just created. The results should change.

---

**Note:** Because you previously limited the number of results returned, the hit count you added at the bottom of the page is now artificially limited. To get an accurate hit count, find the following line in your page definition file:

```
<NamedData NDName="returnCount" NDValue="false"
    NDTYPE="java.lang.Boolean"/>
```

and change it to:

```
<NamedData NDName="returnCount" NDValue="true"
    NDTYPE="java.lang.Boolean"/>
```
This appendix discusses configuration information for the portlet technologies available with Oracle WebCenter.

This chapter includes the following sections:

- Section F.1, "Java Portlet Configuration Tips"
- Section F.2, "OmniPortlet Configuration Tips"
- Section F.3, "Web Clipping Portlet Configuration Tips"
- Section F.4, "Setting Up a Preference Store"
- Section F.5, "Portlet Preference Store Migration Utilities"
- Section F.6, "Exporting and Importing Portlet Producers at Design Time"

For more information about using the portlet technologies at design time, see their respective chapters throughout this guide:

- Chapter 31, "Creating Portlets with the Portlet Wizard"
- Chapter 32, "Coding Portlets"
- Chapter 35, "Creating Portlets with OmniPortlet"
- Chapter 36, "Creating Content-Based Portlets with Web Clipping"

For more information about using these technologies at runtime, see their respective chapters in Oracle Fusion Middleware User’s Guide for Oracle WebCenter.

**F.1 Java Portlet Configuration Tips**

This section contains configuration information for Java portlets.

**Disabling a WSRP Test Page**

To disable your WSRP test pages:

1. In Oracle JDeveloper, go to the Application Navigator and expand the Web Content and WEB-INF folders.
2. Double-click the web.xml file to open it.
3. In the Source view, look for the following element and comment it out:

```xml
<servlet-mapping>
    <servlet-name>WSRPTestPage</servlet-name>
    <url-pattern>/info</url-pattern>
</servlet-mapping>
```
5. Deploy the portlet application.
6. Run your test page in a browser to confirm that the WSRP test page has been disabled.

F.2 OmniPortlet Configuration Tips

This section contains configuration information for OmniPortlet. To learn more about the OmniPortlet wizard, see Chapter 35, "Creating Portlets with OmniPortlet." This section contains configuration information for the following areas:

- Section F.2.1, "Configuring the OmniPortlet Producer to Access Data Outside a Firewall"
- Section F.2.2, "Configuring the OmniPortlet Producer to Access Other Relational Databases"
- Section F.2.3, "Configuring Portal Tools and Web Producers (Optional)"

Note: In this section, OmniPortlet_WAR_DIR indicates the directory where omniPortlet.war is unwrapped 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 custom WebCenter 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_11.1.1.2.0/RANDOMLY_GENERATED_DIRECTORY/war/WEB-INF/providers/omniPortlet/provider.xml
```

F.2.1 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
```
To set the proxy information in the command line when starting the WebLogic server, set the parameters as described in Table F–1, if you are using an HTTP Proxy Host, or Table F–2, if you are using an HTTPS Proxy Host.

**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.

### Table F–1 HTTP Proxy Information Command Line Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>http.proxyHost</code></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><code>http.proxyPort</code></td>
<td>The port number for the HTTP Proxy Host.</td>
</tr>
<tr>
<td><code>http.nonProxyHosts</code></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=localhost=localhost.localdomain. Hosts can be fully qualified host names or can be IP addresses.</td>
</tr>
<tr>
<td><code>http.proxyUser</code></td>
<td>The user to log in to the proxy server if the proxy server requires authentication.</td>
</tr>
<tr>
<td><code>http.proxyPassword</code></td>
<td>The password to log in to the proxy server if the proxy server requires authentication.</td>
</tr>
<tr>
<td><code>http.proxyAuthType</code></td>
<td>The authentication type of the proxy server. Acceptable values: Basic</td>
</tr>
<tr>
<td><code>http.proxyAuthRealm</code></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 F–2 HTTPS Proxy Information Command Line Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>https.proxyHost</code></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><code>https.proxyPort</code></td>
<td>The port number for the HTTPS Proxy Host.</td>
</tr>
<tr>
<td><code>https.nonProxyHosts</code></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=localhost=localhost.localdomain. Hosts can be fully qualified host names or can be IP addresses.</td>
</tr>
<tr>
<td><code>https.proxyUser</code></td>
<td>The user to log in to the proxy server if the proxy server requires authentication.</td>
</tr>
<tr>
<td><code>https.proxyPassword</code></td>
<td>The password to log in to the proxy server if the proxy server requires authentication.</td>
</tr>
<tr>
<td><code>https.proxyAuthType</code></td>
<td>The authentication type of the proxy server. Acceptable values: Basic</td>
</tr>
<tr>
<td><code>https.proxyAuthRealm</code></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:
To configure the proxy information in the `provider.xml` file, see Table F–3 for a list of parameters and their descriptions.

### Table F–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>
<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 the names of a business, or organization, or government agency, for example: <code>*.company.com, *.us.company.com</code>. 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>httpProxyHost</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>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. 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>proxyUserName</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 Source tab and Personalize page. If false, then page designer must log in using the Proxy Authentication section on the Source tab when they define the portlet. The end user must log in using the Proxy Authentication section on the Personalize screen. If <code>&lt;proxyUsername&gt;</code> and <code>&lt;proxyPassword&gt;</code> are given, then they are only used for public users.</td>
</tr>
<tr>
<td>proxyPassword</td>
<td>Enter the user name to log in to the proxy server.</td>
</tr>
<tr>
<td>httpProxyHost</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> file for protection when the producer starts.</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>
  <proxyUseGlobal>false</proxyUseGlobal>
</proxyInfo>
```

**F.2.2 Configuring the OmniPortlet Producer to Access Other Relational Databases**

The OmniPortlet SQL data source is preconfigured to access Oracle databases using the Oracle JDBC drivers, and ODBC data sources using Sun Microsystem’s JDBC-ODBC driver. Oracle allows developers to access other relational databases using DataDirect JDBC drivers.

*See Also:* For a list of supported databases, Certification Matrix for Oracle Application Server and DataDirect JDBC available on the Oracle Technology Network (http://www.oracle.com/technology/index.html).

This section contains the following steps:

- **Section F.2.2.1, "Installing DataDirect JDBC Drivers"**
- **Section F.2.2.2, "Registering DataDirect Drivers in OmniPortlet"**

**F.2.2.1 Installing DataDirect JDBC Drivers**

The following DataDirect JDBC drivers are included with the WebLogic Server installation:

- `wlinformix.jar`
- `wlsqlserver.jar`
- `wlutil.jar`
- `wldb2.jar`
- `wlresource.jar`
- `wlspy.jar`
- `wlbase.jar`

If you do not plan to use these DataDirect drivers, you can instead download DataDirect JDBC drivers to access the desired database. These drivers are packaged in a single ZIP, which you can download from the following location:


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`. 
F.2.2.2 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**:
  ```xml
  <driverInfo class="oracle.webdb.reformlet.data.jdbc.JDBCDriverInfo">  
  <name>Microsoft SQL Server</name>  
  <sourceDataBase>other</sourceDataBase>  
  <subProtocol>weblogic:sqlserve</subProtocol>  
  <connectString>mainProtocol:subProtocol://databaseName</connectString>  
  <driverClassName>weblogic.jdbc.sqlserver.SQLServerDriver</driverClassName>  
  <dataSourceClassName>com.oracle.ias.jdbcx.sqlserver.SQLServerDataSource</dataSourceClassName>  
  <connHandlerClass>oracle.webdb.reformlet.data.jdbc.JDBCConnectionHandler</connHandlerClass>  
  <connPoolSize>5</connPoolSize>  
  <loginTimeOut>30</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>
  ```
Informix:

```xml
<driverInfo class="oracle.webdb.reformlet.data.jdbc.JDBCDriverInfo">
  <name>Informix</name>
  <sourceDataBase>other</sourceDataBase>
  <subProtocol>weblogic:informix</subProtocol>
  <connectString>mainProtocol:subProtocol://databaseName</connectString>
  <driverClassName>weblogic.jdbc.informix.InformixDriver</driverClassName>
  <connHandlerClass>
    oracle.webdb.reformlet.data.jdbc.JDBCODBCConnectionHandler
  </connHandlerClass>
  <connPoolSize>5</connPoolSize>
  <loginTimeOut>30</loginTimeOut>
</driverInfo>
```

Table F–4 describes the parameters in the `driverInfo` property.

### Table F–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 <code>other</code>.</td>
</tr>
<tr>
<td>subProtocol</td>
<td>JDBC subprotocol name used by OmniPortlet to create the connection string, for example <code>sqlserver</code>, <code>sybase</code>, or <code>db2</code>. 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: <code>mainProtocol:subProtocol://databaseName</code></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 F–5 for the right data source class name for your driver.</td>
</tr>
<tr>
<td>connHandlerClass</td>
<td>Class used by OmniPortlet to manage the driver and connection pooling. The value is either of the following:</td>
</tr>
<tr>
<td></td>
<td>- For OmniPortlet version 9.0.4.1 or later: <code>oracle.webdb.reformlet.data.jdbc.JDBCODBCConnectionHandler</code></td>
</tr>
<tr>
<td></td>
<td>- For OmniPortlet versions before 9.0.4.1: <code>oracle.webdb.reformlet.data.jdbc.JDBCODBCConnectionHandler</code></td>
</tr>
<tr>
<td>connPoolSize</td>
<td>Minimum number of connections that are opened by the connection pool.</td>
</tr>
</tbody>
</table>
Table F–5 lists the values for the driverClassName and dataSourceClassName properties for specific DataDirect JDBC drivers.

Table F–5 Parameters and Values for driverClassName and dataSourceClassName

<table>
<thead>
<tr>
<th>DataDirect Drivers Supported</th>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microsoft SQL Server</td>
<td>Parameter: driverClassName</td>
</tr>
<tr>
<td></td>
<td>Value: weblogic.jdbc.sqlserver.SQLServerDriver</td>
</tr>
<tr>
<td>Sybase</td>
<td>Parameter: driverClassName</td>
</tr>
<tr>
<td></td>
<td>Value: weblogic.jdbc.sybase.SybaseDriver</td>
</tr>
<tr>
<td>DB2</td>
<td>Parameter: driverClassName</td>
</tr>
<tr>
<td></td>
<td>Value: weblogic.jdbc.db2.DB2Driver</td>
</tr>
<tr>
<td>Informix</td>
<td>Parameter: driverClassName</td>
</tr>
<tr>
<td></td>
<td>Value: weblogic.jdbc.informix.InformixDriver</td>
</tr>
</tbody>
</table>

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 35, "Creating Portlets with OmniPortlet."

F.2.3 Configuring Portal Tools and Web Producers (Optional)

To ensure that the OmniPortlet and Web Clipping producers, locally built, and custom built Web producers work properly in the middle-tier environment, some additional configuration may be needed. If OmniPortlet or any other Web producers have customizations in the file system, then PDK-Java provides a Preference Store Migration/Upgrade Utility that you can use to migrate the existing customizations to the database and upgrade customizations from earlier releases. See Section F.5, "Portlet Preference Store Migration Utilities" for more information about the PDK Preference Store Migration Utility.

Configuring Portal Tools Producers in the Multiple Middle-Tier Environment

By default, the OmniPortlet producer uses the Database Preference Store. It can work in a multiple middle-tier environment without additional configuration.
You can find more information about configuring the database Preference Store in Section F.5, "Portlet Preference Store Migration Utilities" and in the PDK article titled "Installing the DBPreferenceStore Sample (V2)", (with the filename installing.db.preference.store.v2.html) located in the pdksoftware10132.zip file on the Oracle Technology Network (http://www.oracle.com/technology/products/webcenter/index.html).

1. If you have created an OmniPortlet instance with customizations in the file system, then you must migrate these customizations to the database using the Preference Store Migration Utility.

   To run the migration utility:

   1. Navigate to the WebCenter 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 Preference Store (FilePreferenceStore) to the database Preference Store (DBPreferenceStore):
      
      ```bash
      ```

      See Section F.5, "Portlet Preference Store Migration Utilities" for more information about the PDK Preference Store Migration Utility.

2. Typically, you perform the HTTP Proxy configuration for OmniPortlet and Web Clipping before you configure the Load Balancing Router (LBR). To do it after the LBR is configured, perform the following steps:

   a. The Portal Tools configuration information is stored in the `provider.xml` file on the middle-tier server. You must update the configuration directly on one middle tier (for example, M1) and then propagate it across all middle tiers front-ended by the LBR. You must first shut down all middle tiers except M1.

   b. You can change the HTTP Proxy settings in the `provider.xml` file. For more information, see Section F.2.1, "Configuring the OmniPortlet Producer to Access Data Outside a Firewall."

   c. Propagate the changes made to the `provider.xml` file to middle tier M2:
      
      - Copy `OmniPortlet_WAR_DIR/WEB-INF/providers/omniPortlet/provider.xml` from M1 to M2.
      
      - Copy `WebClipping_WAR_DIR/WEB-INF/providers/webClipping/provider.xml` from M1 to M2.


5. Verify that OmniPortlet and the Web Clipping producers work properly through the LBR, by going to the test pages at the following URLs.

- **OmniPortlet Producer:**
  http://lbr.abc.com/portalTools/omniPortlet/producers/omniPortlet
  
  If you see the "No Portlets Available" message under the Portlet Information section in the OmniPortlet Producer test page, then you may not have configured OmniPortlet correctly in Step 1. If OmniPortlet is configured correctly, then the OmniPortlet and Simple Parameter Form portlets are available on the test page.

- **Web Clipping Producer:**
  http://lbr.abc.com/portalTools/webClipping/producers/webClipping

---

**Note:** To use the Web Clipping portlet, or the Web Page Data Source for OmniPortlet, then you must also enable session binding in Oracle Web Cache.

---

### F.3 Web Clipping Portlet Configuration Tips

Before you use Web Clipping, you must perform some administrative tasks, including the following:

- **Section F.3.1, "Web Clipping Repository Configuration"**
- **Section F.3.2, "HTTP or HTTPS Proxy Configuration"**
- **Section F.3.3, "Web Clipping Producer Security"**

---

#### F.3.1 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 F-1 shows the Provider Test Page of Web Clipping.

**Figure F–1 Web Clipping - Provider Test Page**

### Producer Test Page: Web Clipping

#### Portlet Information

Your producer contains the following portlets:

- WebClippingProvider

#### Producer Initialization Parameters

The following parameters are defined in the Web application configuration file (web.xml):

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>invalidation_caching</td>
<td>true</td>
</tr>
</tbody>
</table>

#### Producer Configuration

You can configure each of the following settings. For more information, see the "Web Clipping Portlet Configuration Tips" section in the Additional Portlet Configuration appendix of the Oracle WebCenter Framework Developer's Guide. [Learn More](#).

<table>
<thead>
<tr>
<th>Setting</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Clipping Repository</td>
<td>Configured</td>
</tr>
<tr>
<td>HTTP Proxy</td>
<td>Not Configured</td>
</tr>
<tr>
<td>Portal Caching</td>
<td>Use Portal Cache (Validation)</td>
</tr>
</tbody>
</table>

You cannot change the Web Clipping configuration information by using the Provider Test Page. You can configure the Web Clipping repository by setting appropriate values in the `provider.xml` file. In this file, you use the `repositoryInfo` tag to specify the Web Clipping repository settings.

---

**Note:** To determine the path of `provider.xml`, search for `webClipping/provider.xml` in the file system. For example, run the following command in UNIX:

```
> find DOMAIN_DIR -name "provider.xml" | grep -i webClipping
```

In Oracle JDeveloper's Integrated WLS, Web Clipping's `provider.xml` is located here:

```
JDEV_SYSTEM_DIRECTORY/DefaultDomain/servers/DefaultServer/tmp/_WL_user/portalTools_11.1.1.2.0/RANDOMLY_GENERATED_DIRECTORY/war/WEB-INF/providers/webClipping/provider.xml
```

Note that, on a Windows platform, pages in custom WebCenter 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 subsections:
Section F.3.1.1, "Using Oracle Metadata Services (MDS) as the Web Clipping Repository"

Section F.3.1.2, "Using an Oracle Database as the Web Clipping Repository"

Section F.3.1.3, "Configuring Web Clipping Repository in provider.xml"

Section F.3.1.4, "Attributes and Child Tags of the repositoryInfo Tag"

F.3.1.1 Using Oracle Metadata Services (MDS) as the Web Clipping Repository

By default, in Oracle JDeveloper, the Web Clipping producer hosted on Integrated WLS, the default server, is configured to use MDS, which is file-based, as the Web Clipping repository.

**Note:** In a full Oracle Fusion Middleware installation, the Web Clipping portlet producer is also included within the WLS_Portlets managed server in the default domain. By default, this Web Clipping portlet producer is configured to use the Oracle database, which is installed as part of Oracle WebLogic Server, as the Web Clipping repository.

Example F–1 shows MDS as the default repository in provider.xml.

**Example F–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 custom WebCenter applications are not rendered if there is a space in the path to the system directory in JDeveloper. Therefore, ensure that the JDEV_SYSTEM_DIRECTORY path does not contain spaces.

The mds-config.xml file specifies the location of the repository in a property tag:

```
<property name="metadata-path" value="portletdata/tools/webClipping"/>
```

The location specified for value is relative to JDEV_HOME/portal. Any relative path specified is interpreted to be relative to JDEV_HOME/portal. To use another location, for example, a location outside the Oracle JDeveloper home, then specify an absolute path, such as c:\mds.

**Note:** For a multiple middle tier deployment, change the metadata path to a shared file system.
F.3.1.2 Using an Oracle Database as the Web Clipping Repository

Although MDS is the default repository in Oracle JDeveloper, you can alternatively select to use a database schema for your Web Clipping repository.

Note: If you use a database as your Web Clipping repository, any customizations are retained even if the Web Clipping producer or the consuming application is redeployed.

You can use either of the following database schemas for Web Clipping:

- The default PORTLET database schema created by RCU for Oracle WebLogic Server
- A user-defined database schema for Oracle 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 subsections:

- Section F.3.1.2.1, "Using the Database Schema Created by RCU"
- Section F.3.1.2.2, "Creating a Database Schema"

F.3.1.2.1 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 Oracle WebCenter. For more information, see the "Installing WebCenter" chapter in the Oracle Fusion Middleware Installation Guide for Oracle WebCenter.

F.3.1.2.2 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 F–2.

Example F–2 Java Command for Creating a Schema for Web Clipping Portlet Definitions and Clippings

```
java -classpath WC_ORACLE_HOME/lib/xmlparserv2.jar:
    WC_ORACLE_HOME/jdbc/lib/ojdbc14.jar:WC_ORACLE_HOME/portal/jlib/wce.jar
```

Where:

- `WC_ORACLE_HOME` is the path to your WebCenter Oracle home directory
- `dbuser` is the database user for the schema

Consider using the same database user you use to create the WSRP and PDK-Java preference store database schema. If you do not use the same user, then you must create a new user and grant it connect and resource privileges.
- `dbpassword` is the specified user's password
- `dburl` is the URL for the database

This is the database that contains the schema that you create for Web Clipping portlet definitions and clippings by using the following syntax:

```
jdbc:oracle:thin:@//dbhost:dbport/service_name
```

For example:

```
jdbc:oracle:thin:@//shobeen:1521/sales_us
```

**Note:** The classpath in Example F–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.

### F.3.1.3 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 F–3.

**Example F–3 Oracle Database as the Web Clipping Repository**

```
<repositoryInfo class="oracle.portal.wcs.provider.info.JdbcDbInformation">
  <connectionName>jdbc/portletPrefs</connectionName>
  <useRAA>false</useRAA>
  <useASO>false</useASO>
</repositoryInfo>
```

For information about tag parameters, see Section F.3.1.4, "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 F–4.

**Example F–4 Setting Oracle Database 9i or Later as Web Clipping Repository**

```
<repositoryInfo class="oracle.portal.wcs.provider.info.DatabaseInformation">
  <useRAA>false</useRAA>
  <databaseHost>dbhost.mycompany.com</databaseHost>
  <databasePort>1521</databasePort>
  <databaseSid>iasdb</databaseSid>
  <databaseUsername>scott</databaseUsername>
  <databasePassword>!tiger</databasePassword>
  <useASO>false</useASO>
</repositoryInfo>
```

For information about tag parameters, see Section F.3.1.4, "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 F.3.3, "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.

F.3.1.4 Attributes and Child Tags of the repositoryInfo Tag

Table F–6 lists and describes the attributes and child tags of the `repositoryInfo` tag.

**Note:** The attributes and child tags of the `repositoryInfo` tag are also described in the comments in the Web Clipping `provider.xml` file.

In the comments in the `provider.xml` file, the example provided for Oracle9i Database or later using Oracle infrastructure database is specific to Oracle Portal and its infrastructure database and application programming interfaces. That example must not be used for custom WebCenter application implementations.
## Table F-6  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>
</tbody>
</table>
|                      |              | ■ `oracle.portal.wcs.provider.info.MdsInformation`  
|                      |              |    This value signifies that MDS is used to store Web Clipping definitions and that MDS configuration is pushed to the `mds-config.xml` file. |
|                      |              | ■ `oracle.portal.wcs.provider.info.DatabaseInformation`  
|                      |              |    This value signifies that an Oracle9i Database or later is used to store Web Clipping definitions and that the database connection details are included as children in the `repositoryInfo` tag. |
|                      |              | ■ `oracle.portal.wcs.provider.info.JdbcDbInformation`  
|                      |              |    This value signifies that Oracle database installed as part of Oracle WebLogic Server is used to store Web Clipping definitions. |
| **mdsConfigLocation**| MDS          | Use the `mdsConfigLocation` tag when the value for the `class` attribute indicates an MDS repository. It points to the MDS configuration file `mds-config.xml`, which specifies the actual MDS location. The `mds-config.xml` file is located at: |
|                      |              | `MW_HOME/wlsServer_10.3/wc_domain/servers/WLS_Portlet/tmp/_WL_user/portalTools_11.1.1.2.0/yyggl7/war/WEB-INF` |
| **connectionName**   | Database     | Specifies the JNDI name of the data source where the Web Clipping repository has been installed using the RCU. By default, the connection name is `jdbc/portletPrefs`, which points to the `PORTLET` schema in Oracle WebLogic Server. In an interoperability scenario where an Oracle WebLogic Server 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. |
| **useASO**           | Database     | Set to `true` or `false`:
|                      |              | ■ Specify `true` to use Oracle Advanced Security Option (ASO) to encrypt the communication channel between the Web Clipping producer and the database. This is provided for introducing added security if case-sensitive data is contained in the clipped content. |
|                      |              | ■ Specify `false` to omit this option. |
| **useRAA**           | Database     | Set to `true` or `false`:
|                      |              | ■ Specify `true` if Repository Access APIs are used to access the database connection parameters. Specifying `true` is equivalent to making the Web Clipping producer use the default Oracle infrastructure database as the repository. Specifying `true` removes the need for other `repositoryInfo` child tags. |
|                      |              | ■ Specify `false` to omit this option. |
| **databaseHost**     | Database     | Specify the host name of the Oracle database. Use only version 9i or later. For example: |
|                      |              | `mycompany.dbhost.com` |
F.3.2 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.

WebCenter application administrators can set proxy settings manually according to the HTTP or HTTPS configuration. Edit the appropriate entries in the `provider.xml` file.

Example F–5 shows the relevant portion of `provider.xml`.

Example F–5  Proxy Settings

```
- <!--
  proxy information: Fill the following up if you have a proxy server between the provider and external sites.
  <proxyInfo class="oracle.portal.provider.v2.ProxyInformation">
    <httpProxyHost>proxy.mycompany.com</httpProxyHost>
    <httpProxyPort>80</httpProxyPort>
    <dontProxyFor>*.mycompany.com</dontProxyFor>
    <proxyUseAuth>true</proxyUseAuth>
    <proxyType>Basic</proxyType>
    <proxyRealm>realm1</proxyRealm>
    <proxyUseGlobal>false</proxyUseGlobal>
    <proxyUser>scott</proxyUser>
    <proxyPassword>!tiger</proxyPassword>
  </proxyInfo>
-->

It is optional to specify values for the following tags: `<proxyUseAuth>`, `<proxyType>`, `<proxyRealm>`, `<proxyUseGlobal>`, `<proxyUser>`, and `<proxyPassword>`.

Table F–3, available earlier in this appendix, describes the proxy settings you specify in the `provider.xml` file. The descriptions in the table are applicable for Web Clipping producers also.
The preceding sections described the administrative tasks that must be performed before you are able to use the Web Clipping producer. The following sections describe some security configuration options that you must implement to enable the Web Clipping producer to access Secure Sockets Layer (SSL)-enabled web sites and encrypt the channel between itself and the database:

- Section F.3.3.1, "Adding Certificates for Trusted Sites"
- Section F.3.3.2, "Configuring Oracle Advanced Security for the Web Clipping Producer"

### F.3.3.1 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.

### F.3.3.2 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 F–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.

---

### F.4 Setting Up a Preference Store

The portlet preference store is used for persisting consumer registration handles and portlet preference data. Portlet producers can use one of two types of preference store: *File* and *Database*.

In a clustered environment, Oracle recommends the use of a database preference store. If you prefer to use a file-based preference store, then you must also use a shared file system.

If you configure the WebLogic Server to run portlet applications, then the database preference store is configured for you. For more information, see the *Oracle Fusion Middleware Installation Guide for Oracle WebCenter*.

This section contains configuration information for the following areas:

- **Section F.4.1, "WSRP Producers"**
- **Section F.4.2, "PDK-Java Producers"**

### F.4.1 WSRP Producers

WSRP producers have a JNDI preference value that determines which preference store is used, and is set in the web.xml file of the portlet application. Table F-7 lists and describes the JNDI variables used to specify the preference store for WSRP producers.
Setting Up a Preference Store

Example F–6 Configuring web.xml to Use a Database Preference Store

```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>
```

Example F–7 Configuring web.xml to Use a File-Based Preference Store

```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>
```

When you have updated the relevant `web.xml` files, restart your WebLogic Server.

### F.4.2 PDK-Java Producers

The type of preference store used for PDK-Java producers is determined by the `preferenceStore` tag in the `provider.xml` file. Table F-8 lists and describes the attributes and parameters used with the `preferenceStore` tag when a database preference store is specified.

---

### Table F–7 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 or Database) is used for persisting a portlet application's consumer registration handles and portlet preferences.</td>
</tr>
<tr>
<td>oracle/portal/wsrp/server/persistentStore</td>
<td>File</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. Absolute paths are interpreted relative to the file system root. Relative paths are interpreted relative to the \WC_ORACLE_HOME\portal directory. Note that all producers running within the same WebLogic Server must use the same path for this variable. Otherwise, you get a Portlet unavailable error for some portlets.</td>
</tr>
</tbody>
</table>

---

The following examples show the entries in a producer's `web.xml` file.
A preference store is a mechanism for storing information like user preference data, portlet and producer settings, or even portlet data. Preferences can be stored in a database, which is recommended for high availability configurations, a file system, or with the consuming application. You can migrate the following stores for your WebCenter applications:

- Section F.5.1, "JPS Portlet Preference Store - PersistenceMigrationTool"

Table F–8 Attributes and Parameters of the preferenceStore Tag

<table>
<thead>
<tr>
<th>Attribute/Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| class               | This required attribute specifies the Java class that defines the location and other details of portlet preferences. For example, a file-based preference store might use: |<preferenceStore class="oracle.portal.provider.v2.preference.FilePreferenceStore"/>
A database preference store might use: |<preferenceStore class="oracle.portal.provider.v2.preference.DBPreferenceStore">
OmniPortlet uses its own class, for example: |<preferenceStore class="oracle.webdb.reformlet.ReformletFilePreferenceStore">
| name                | This required parameter provides a name for the preference store. Use any value you choose. For example: |<preferenceStore class="oracle.portal.provider.v2.preference.DBPreferenceStore">
|                     | <name>MyPDKProducerPreferenceStore</name> |
|                     | </preferenceStore> |
| connection          | This required parameter for a database preference store points to a JNDI connection that connects with a schema containing the portlet preference store. For example: |<preferenceStore class="oracle.portal.provider.v2.preference.PCODBPreferenceStore">
|                     | <name>MyPDKProducerPreferenceStore</name> |
|                     | <connection>java:comp/env/jdbc/portletPrefs</connection> |
|                     | </preferenceStore> |
| rootDirectory       | This optional parameter specifies the location where the file-based preference store preferences are stored. When this parameter is not included with the preferenceStore tag, the default file-based preference store location is used. This is the same folder where the producer’s provider definition file is stored. |
| useHashing          | This optional parameter is used when specifying a file based preference store, and accepts the value true or false. When it is set to true, each preference data file is stored in an extra subdirectory with a name determined by hashing the data file name. Using this parameter can improve file system performance by limiting the number of preference data files stored in a single directory. For example: |<preferenceStore class="oracle.portal.provider.v2.preference.FilePreferenceStore">
|                     | <name>PDKProducerPreferenceStore</name> |
|                     | <useHashing>true</useHashing> |
|                     | </preferenceStore> |
F.5.1 JPS Portlet Preference Store - PersistenceMigrationTool

A WSRP container preference store is a mechanism used for persisting consumer registration and portlet preference data. Currently, there are two preference store implementations, database preference store and file-based preference store. A database preference store persists data using a relational database. A file-based preference store persists data using the file system. A file-based preference store may be used for testing, since it removes the dependency on a database. But, for highly available configurations, Oracle recommends that you use the database preference store.

The WSRP container preference store migration utility, PersistenceMigrationTool, enables you to migrate existing data between different preference stores (for example, from a database preference store to a file preference store). This utility also enables upgrading users to ensure that their existing locale-specific portlet preference data uses a naming format compatible with the latest JPS release. You can also use this utility to migrate between source and destination stores of the same type, enabling data to be moved from one database store to another.

The syntax of the PersistenceMigrationTool is:

```
java oracle.portlet.server.containerimpl.PersistenceMigrationTool
    -sourceType file | db
    -destType file | db
    { -sourcePath dir |
    -sourceUsername username -sourcePassword password -sourceDatabase db}
    { -destPath dir | destUsername username -destPassword password -destDatabase db}
    [-debug]
```

where

sourceType indicates whether the source store is in a file or database. You can have source and destination stores of the same type. Hence, you can migrate from one database to another or one file system to another.

destType indicates whether the destination store is in a file or database. You can have source and destination stores of the same type. Hence, you can migrate from one database to another or one file system to another.

sourcePath is the location of a file-based preference store. This argument is required when sourceType is file.

Note: You must set the classpath when running this tool to include wsrp-container.jar, oracle.portlet-api.jar and ojdbc6.jar. For example:

```
java  -classpath \
    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_HOME/server/lib/ojdbc6.jar
    oracle.portlet.server.containerimpl.PersistenceMigrationTool
```

```
java oracle.portlet.server.containerimpl.PersistenceMigrationTool
    -sourceType file | db
    -destType file | db
    { -sourcePath dir |
    -sourceUsername username -sourcePassword password -sourceDatabase db}
    { -destPath dir | destUsername username -destPassword password -destDatabase db}
    [-debug]
```

where

sourceType indicates whether the source store is in a file or database. You can have source and destination stores of the same type. Hence, you can migrate from one database to another or one file system to another.

destType indicates whether the destination store is in a file or database. You can have source and destination stores of the same type. Hence, you can migrate from one database to another or one file system to another.

sourcePath is the location of a file-based preference store. This argument is required when sourceType is file.

Note: You must set the classpath when running this tool to include wsrp-container.jar, oracle.portlet-api.jar and ojdbc6.jar. For example:

```
java  -classpath \
    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_HOME/server/lib/ojdbc6.jar
    oracle.portlet.server.containerimpl.PersistenceMigrationTool
```

```
java oracle.portlet.server.containerimpl.PersistenceMigrationTool
    -sourceType file | db
    -destType file | db
    { -sourcePath dir |
    -sourceUsername username -sourcePassword password -sourceDatabase db}
    { -destPath dir | destUsername username -destPassword password -destDatabase db}
    [-debug]
```

where

sourceType indicates whether the source store is in a file or database. You can have source and destination stores of the same type. Hence, you can migrate from one database to another or one file system to another.

destType indicates whether the destination store is in a file or database. You can have source and destination stores of the same type. Hence, you can migrate from one database to another or one file system to another.

sourcePath is the location of a file-based preference store. This argument is required when sourceType is file.
sourceUsername is the database user name for a preference store database. This argument is required when sourceType is db.

sourcePassword is the database password for a preference store database. This argument is required when sourceType is db.

sourceDatabase is the name of a preference store database. This argument is required when sourceType is db.

destPath is the location of a file-based preference store. This argument is required when destType is file.

destUsername is the database user name for a preference store database. This argument is required when destType is db.

destPassword is the database password for a preference store database. This argument is required when destType is db.

destDatabase is the name of a preference store database. This argument is required when destType is db.

debug turns on full logging through standard output to enable users to diagnose issues that arise when the tool runs.

Example F–8 demonstrates running the PersistenceMigrationTool utility. In this example, preferences from a database store are copied to a file store.

Example F–8  Running the PersistenceMigrationTool Utility

```java
java  -classpath 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_HOME/server/lib/ojdbc6.jar \
oracle.portlet.server.containerimpl.PersistenceMigrationTool
-sourceType db \
-sourceUsername scott \
-sourcePassword tiger \
-sourceDatabase abc_mycompany.com:1521:yourdatabase \
-destType file \
-destRoot /data/prefs
```

F.5.2 PDK-Java Portlet Preference Store - Migration and Upgrade Utilities


If you have installed Oracle PDK, then you can manage the information stored in the preference store by using the Preference Store Migration and Upgrade Utility, which is included in the pdkjava.jar file. Note that you must run this tool from WC_ORACLE_HOME. The syntax of the migration utility is:
Portlet Preference Store Migration Utilities

Note: You must set the classpath when running this tool to include pdkjava.jar, ptlshare.jar, and ojdbc6.jar. For example:

```
java -classpath WC_ORACLE_HOME\jdeveloper\modules\oracle.dms_11.1.1\dms.jar;
WC_ORACLE_HOME\wlserver_10.3\server\lib\ojdbc6.jar;WC_ORACLE_HOME\jdeveloper\webcenter\modules\oracle.portlet.server_11.1.1\pdkjava.jar;
WC_ORACLE_HOME\jdeveloper\webcenter\modules\oracle.portlet.server_11.1.1\ptlshare.jar
oracle.portal.provider.v2.preference.MigrationTool
```

```
java oracle.portal.provider.v2.preference.MigrationTool
-mode [file | db | filetodb | filetofile | dbtodb]
[-remap language | locale]
[-countries iso_country_code]
[-pref1UseHashing true | false]
[-pref1RootDirectory directory]
  -pref1User username -pref1Password password -pref1URL url)
[-pref1UseHashing true | false]
[-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**, **dbtodb**, or **dbtodb** indicates to run in migration mode. See Section F.5.2.1, "Migration Mode" for more information about this mode.
- **file** or **db** indicates to run in upgrade mode. See Section F.5.2.2, "Upgrade Mode" for more information about this mode.
- **-remap** is the **localePersonalizationLevel** (language or locale). Note that you only must use this option to change **localePersonalizationLevel** as part of your upgrade or migration.
- **-countries** specifies a prioritized list of ISO country codes, indicating your order of preference in a collision between remapped preferences for different countries. **-countries** is only meaningful if you also specified the **-remap** option.
- **-pref1UseHashing** specifies whether you want to employ hashing on the source for this operation.
- **-pref1RootDirectory** is the path of a source file system, for example, j2ee/home/applications/jpdk/jpdk/WEB-INF/providers/sample.
- **-pref1User** is the user name for a source database.
- **-pref1Password** is the password for a source database.
- **-pref1URL** is the URL to a source database, for example, jdbc:oracle:thin:@myserver.mydomain.com:1521:mysid.
- **-pref2UseHashing** specifies whether you want to employ hashing on the destination for this operation.
Portlet Preference Store Migration Utilities

- **-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, 

- **-upfixwpi** indicates a log file for the operation.

This section includes the following subsections:

- Section F.5.2.1, "Migration Mode"
- Section F.5.2.2, "Upgrade Mode"

### F.5.2.1 Migration Mode

Use a migration mode to copy data from a source preference store to a target
preference store. When the utility is run in this mode, the preference stores for all the
portlet definitions are updated.

Table F–9 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 FilePreferenceStore to a DBPreferenceStore.</td>
</tr>
<tr>
<td>filetofile</td>
<td>Use when data must be copied from one FilePreferenceStore to another FilePreferenceStore that is in a different location.</td>
</tr>
<tr>
<td>dbtotime</td>
<td>Use when data must be copied from a DBPreferenceStore to a FilePreferenceStore.</td>
</tr>
<tr>
<td>dbtodb</td>
<td>Use when data must be copied from one DBPreferenceStore to another DBPreferenceStore 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 portlet template and the appropriate
schemas have been created using the RCU. For more information, see Oracle Fusion
Middleware Installation Guide for Oracle WebCenter.

When using a migration mode, you can also use the -remap and -countries
options to specify that the data must be upgraded while being migrated. Specifically,
you use these options to ensure that the locale-specific preferences are appropriately
remapped.

You can use the other options accepted by the utility to specify the properties of the
preference stores involved in the upgrade or migration process. These options must
correspond to the tags you specify in provider.xml to describe the preference
stores. For more information about the properties you can set on a preference store,
see the "PDK-Java XML Provider Definition Tag Reference" document on Portal
Studio:

c/apidoc/oracle/portal/provider/v2/ProviderDefinition.html
Properties beginning with the prefix -pref1 correspond to properties of the source preference store (in an upgrade mode, this is the only preference store). For example, specifying -pref1UseHashing true -pref1RootDirectory j2ee/home/applications/jpdk/jpdk/WEB-INF/providers/sample sets the useHashing and rootDirectory properties of a source FilePreferenceStore.

When a migration basic mode is selected, properties beginning with the prefix -pref2 correspond to properties of the target preference store. For example, specifying -pref2User portlet_prefs -pref2Password portlet_prefs -pref2URL jdbc:oracle:thin:@myserver.mydomain.com:1521:mysid sets the database connection details on a target DBPreferenceStore.

Example F–9  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
```

F.5.2.2 Upgrade Mode

Use an upgrade mode to upgrade data in place, and to modify existing locale-specific preferences in the preference store so that the naming format used is compatible with the current version of Oracle Portal and a given localePersonalizationLevel setting.

Table F–10 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 F–10  PDK-Java Migration Utility Command Line, Upgrade**

```
java -classpath WC_ORACLE_HOME/webcenter/modules/oracle.portlet.server_11.1.1/pdkjava.jar:
  WC_ORACLE_HOME/webcenter/modules/oracle.portlet.server_11.1.1/ptlshare.jar:
  WC_ORACLE_HOME/ucm/shared/classes/ojdbc14.jar \
  oracle.portal.provider.v2.preference.MigrationTool \ 
  -mode file -remap language \\
  -countries GB,US -pref1UseHashing true \\
  -pref1RootDirectory j2ee/home/applications/jpdk/jpdk/WEB-INF/providers/sample
```

**F.5.3 Web Clipping Repository**

Web Clipping does not have a preference store as such, but it stores Web Clipping definitions and associated metadata. By default, it uses MDS, which is file-based, for this purpose. But you can configure Web Clipping to use a database. To migrate this repository for WebCenter 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 WebCenter applications that use the Web Clipping producer. For more information, see Section 32.1.3, "How to Implement Export/Import of Customizations (WSRP 2.0)."

2. Update the producer to use a different repository. For information, see Section F.3.1.3, "Configuring Web Clipping Repository in provider.xml."

3. Run the Predeployment tool in import mode on all WebCenter applications that use the Web Clipping producer. For more information, see Section 32.1.3, "How to Implement Export/Import of Customizations (WSRP 2.0)."

**F.5.4 Moving a Producer**

In some situations, you may want to move a portlet producer. To do so, you must:

1. Install the new producer.

2. Make the preference store of the original producer available to the new producer by performing one of the following tasks:
   - Migrate the preference store using the Preference Store Migration and Upgrade Utilities (see Section F.5.2, "PDK-Java Portlet Preference Store - Migration and Upgrade Utilities" for more information), or
   - Configure the new producer to use the same preference store as the original producer. If the database preference store is being used, then you must point the WebLogic Server data source to the same database as the original producer.

3. Update the URL of the producer registration by using Enterprise Manager Fusion Middleware Control or the WLST commands: `setWSRPPProducerRegistration` or `setPDKJavaProducerRegistration`.  

---

---

---

---

---
F.6 Exporting and Importing Portlet Producers at Design Time

You can export and import portlet producers at design time.

This section includes the following subsections:

- Section F.6.1, "Exporting Portlet Producers at Design Time"
- Section F.6.2, "Importing Portlet Producers at Design Time"

F.6.1 Exporting Portlet Producers at Design Time

When you package an application for deployment, any portlet producers referenced by the application are contacted so that the producer data can be included in the MAR file. If any of the producers are not running, they cannot be contacted and the data cannot be included.

By creating an export archive of the producer data at design time instead, you can ensure that the producers are running, therefore all the producer data, including remote customizations and client-side metadata, can be retrieved. WebCenter 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. In JDeveloper, open the application that contains the producers to export.

3. From the menu, choose Application and then Export Portlet Producers.
   
   This menu option appears only if the current application contains producers.

4. 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.

5. Click OK.

   When the application is packaged for deployment, because you set the disableRemoteExport flag to true, WebCenter Framework checks for the presence of the export archive at the location specified in the dialog. If the export archive exists, the contents of the archive are included in the MAR file instead of contacting the remote producers to retrieve producer data.

   **Note:** If the disableRemoteExport flag is set and there is no export archive, a default export archive, without the remote producer data, is created and included in the MAR file.

F.6.2 Importing Portlet Producers at Design Time

You can import producer data, including customizations made at runtime, from a deployed application into your application at design time.
For information about how to create an export archive of producer metadata from a deployed application, see the section "Exporting Portlet Client Metadata (Custom WebCenter Application)" in the Oracle Fusion Middleware Administrator’s Guide for Oracle WebCenter.

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. In JDeveloper, open the application into which you want to import the producers.

3. From the menu, choose Application and then Import Portlet Producers.

4. 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.

5. 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.
Reuse of Oracle Portal Components

This appendix describes how to reuse components from your existing Oracle Portal application in a custom WebCenter application built with Oracle WebCenter Framework.

This appendix includes the following sections:
- Section G.1, "Introduction to Oracle Portal Components"
- Section G.2, "Reusing Portlets"
- Section G.3, "Reusing Items"

G.1 Introduction to Oracle Portal Components

The two most basic building blocks of Oracle Portal are as follows:

- Portlets are the fundamental building blocks of a portal page. Portlets can be deployed to Oracle Portal through three producer (known in Oracle Portal as provider) types:
  - Web (or Oracle PDK-Java) producers use open standards, such as XML, SOAP, HTTP, or J2EE for deployment, definition, and communication with applications. The Oracle PDK-Java provides a framework that simplifies the task of building Web producers. For more information, see Section 29.2.3.1, "PDK-Java Producers."
  - WSRP producers serve as the containers for standards-based (JSR 168) portlets. For more information, see Section 29.2.3.2, "WSRP Producers."
  - Database producers own one or more database portlets. Examples of database portlets include, portlets built using the Oracle PDK-PL/SQL, Portlet Builder portlets (charts, forms, reports, and so on), and Oracle Portal page portlets.

In Oracle WebCenter Framework, you can make Web and WSRP portlets available by registering their producers with the application. For more information, see Section G.2.1, "How to Reuse JSR 168 and Oracle PDK-Java Portlets."

You can also make database portlets available to custom WebCenter applications by using the Federated Portal Adapter. For more information, see Section G.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 Oracle WebCenter Framework. For more information, see Section G.3, "Reusing Items."

Given the technical differences between Oracle Portal and WebCenter, there is no direct upgrade available across the platform. However, key components of your portal can be exposed within WebCenter.

G.2 Reusing Portlets

You can make the portlets used in your portal available to your custom WebCenter applications.

This section includes the following subsections:

- Section G.2.1, "How to Reuse JSR 168 and Oracle PDK-Java Portlets"
- Section G.2.2, "What You May Need to Know About Events"
- Section G.2.3, "What You May Need to Know About Mobile Portlets"
- Section G.2.4, "What You May Need to Know About the Portlet Chrome"
- Section G.2.5, "What You May Need to Know About Personalizations and Customizations"
- Section G.2.6, "What You May Need to Know About Oracle Portal System Resources"
- Section G.2.7, "What You May Need to Know About Partner and External Applications"
- Section G.2.8, "How to Use the Federated Portal Adapter to Reuse Database Portlets"
- Section G.2.9, "What You May Need to Know About Troubleshooting the Federated Portal Adapter"
- Section G.2.10, "What You May Need to Know About Limitations of the Federated Portal Adapter"
- Section G.2.11, "How to Reuse Oracle PDK-Java Producers from Earlier Oracle Application Server Versions"

G.2.1 How to Reuse JSR 168 and Oracle PDK-Java Portlets

To reuse JSR 168 and Oracle PDK-Java portlets in your custom WebCenter application, simply register the producers in the same way as any other producer.

To reuse JSR 168 and Oracle PDK-Java portlets:

1. In Oracle JDeveloper, access the WSRP or Oracle PDK-Java Producer Registration wizard.
2. Step through the wizard providing information about the producer, including the URL endpoint.
3. Open the page to which you to add the portlet.
4. Drag the portlet from the registered producer to the appropriate part of the page.

For more detailed information about consuming portlets in a custom WebCenter application, see Chapter 34, "Consuming Portlets."
G.2.2 What You May Need to Know About Events

Oracle PDK-Java events are not supported in Oracle WebCenter Framework. If you have Oracle PDK-Java portlets that use events and you deploy them in an Oracle WebCenter Framework environment, then the events are ignored.

G.2.3 What You May Need to Know About Mobile Portlets

Mobile portlets are not supported in Oracle WebCenter Framework. If you have Oracle PDK-Java mobile portlets, then they do not work in a Oracle WebCenter Framework environment.

G.2.4 What You May Need to Know About the Portlet Chrome

In Oracle WebCenter 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 Oracle WebCenter Framework environment.

As part of this filtering process, Oracle WebCenter 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 Oracle WebCenter 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.

G.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 custom WebCenter application. In a custom WebCenter application, your Oracle Portal portlets are treated as new instances. Hence, previous personalizations and customizations are not available.

G.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 Oracle WebCenter 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 Oracle WebCenter 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 Oracle WebCenter Framework environment by using the proper portlet APIs.

**G.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.

**G.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 custom WebCenter applications.

**Note:** You can also expose Oracle Portal pages in WebCenter 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 custom WebCenter 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 custom WebCenter application.

For more information about the Federated Portal Adapter, see the chapter "Using the Federated Portal Adapter" in the *Oracle Fusion Middleware Administrator’s Guide for Oracle Portal.*

**To register a Federated Portal Adapter producer with Oracle WebCenter:**

1. Set up the Oracle Portal environment for the Federated Portal Adapter. For more information, see the section "Setting Up the Environment to Use the Federated Portal Adapter" in the *Oracle Fusion Middleware Administrator’s Guide for Oracle Portal.*

2. To register an instance of Oracle Portal through the Federated Portal Adapter, the portal must have a user named "PORTAL" in Oracle Internet Directory. To create this user:
   a. Log in to the Oracle Portal instance as the portal administrator user.
   b. In the **User** portlet, click **Create New Users**.
      
      By default, the **User** portlet is on the **Administer** tab of the **Portal Builder** page, on the **Portal** subtab.
c. In the **Last Name** field, enter PORTAL.
d. In the **User ID** field, enter PORTAL.
e. In the **Password** and **Confirm Password** fields, enter an appropriate password.
f. In the **Email Address** field, enter an appropriate email address.
g. Click **Submit**, then **Done**.
h. In the **Portal User Profile** portlet, enter PORTAL in the **Name** field, and click **Edit**.

By default, the **Portal User Profile** portlet is on the **Administer** tab of the **Portal Builder** page, on the **Portal** subtab.
i. In the **Default Group** field, enter PORTLET_PUBLISHERS.
j. Click **OK**.

3. If you want to display Oracle Portal pages in your custom WebCenter applications, you must first publish each of those pages as portlets. To display Oracle Portal pages, edit the page properties and select **Publish As Portlet** on the **Optional** tab. For more information, see the section "Placing One Page Onto Another" in the [Oracle Fusion Middleware User’s Guide for Oracle Portal](#).

4. Confirm that the users of the custom WebCenter 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 custom WebCenter 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 custom WebCenter application and Oracle Portal.

5. In Oracle JDeveloper, register the producer as an Oracle PDK-Java producer using the Federated Portal Adapter URL and Service ID:

http://host:port/adapter/dad/schema

To verify the URL, enter it in a browser. If the URL is correct you should see the following message:

**Congratulations - you got to the adapter test page**

If you are logged in you should see a list of producers, their service IDs, and the portlets each provides. If you are logged in as an Oracle Portal administrator, you should also see additional details about page portlets and Oracle Portal HMAC registrations.

When you register the producer, ensure that you select the **Establish Producer Sessions** checkbox on the Specify Connection Details page of the wizard. This is so that the WebCenter user information is correctly propagated to Oracle Portal.

For more information, see Section 34.2.3, "How to Register an Oracle PDK-Java Portlet Producer."

6. You can drag and drop the producer’s portlets onto a page.

7. To secure communication between the Oracle Portal instance and Oracle WebCenter, 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:
Reusing Portlets

@wwc/proadssr.sql http://www.oracle.com/adapter/portal

Note that, for custom WebCenter 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.

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 custom WebCenter 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.

G.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 Oracle WebCenter.

This section includes the following subsections:

- Section G.2.9.1, "On Registration"
- Section G.2.9.2, "During Runtime"

G.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.

G.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 G.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 custom WebCenter 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.

G.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 Oracle WebCenter:

■ 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. For a full list, see "Oracle Portal - Limitations" in Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter.

G.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 custom WebCenter 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 Oracle WebCenter Framework you must manually ensure that the images can be found by portlets.

This section includes the following subsections:

- Section G.2.11.1, "Consuming a Portlet from Oracle Portal"
- Section G.2.11.2, "Redeploying PDK-Java Producers from Oracle Portal"

G.2.11.1 Consuming a Portlet from Oracle Portal

In Oracle WebCenter, 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 in Example G–1.

Example G–1  Web Servlet Initialization Parameters

```xml
<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 G–2.

Example G–2 Servlet Definition for Resource Servlet

```xml
<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 G–3.

Example G–3 Servlet Mapping for Resource Servlet

```xml
<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 34.2.3, "How to Register an Oracle PDK-Java Portlet Producer."

G.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 G–4.

Example G–4 Web Servlet Initialization Parameter

```xml
<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 G–5.

Example G–5 Servlet Definition for Resource Servlet

```xml
<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 G–6.
Example G–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 34.2.3, "How to Register an Oracle PDK-Java Portlet Producer."

G.3 Reusing Items

Items from Oracle Portal have no equivalent in Oracle WebCenter Framework. As such, creating, maintaining, and publishing items is not supported in Oracle WebCenter Framework. To replicate the behavior of items with Oracle WebCenter Framework, you can use JCR data controls to include content from Oracle Portal. Oracle Portal is a content repository that Oracle WebCenter 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 13, "Integrating Content."
About mode
A **portlet mode** that typically displays information such as copyright, version, and author of the portlet.

Activity Stream
In the **People Connections service**, a feature for viewing the application activities tracked for you and other users.

ADF
Application Development Framework. A range of technologies aimed at making **Java EE** application development faster and simpler for developers while at the same time taking advantage of proven software patterns to ensure that the developed application is scalable, performant, and the like.

administrator
In WebCenter Spaces there are two types of administrator:

- Fusion Middleware administrator: Also referred to as systems administrator. A user with complete administrative capabilities. This administrator can perform the complete range of security-sensitive administrative duties, and all installation, configuration, and audit tasks.

- WebCenter Spaces administrator: A WebCenter Spaces user who is responsible for customizing WebCenter Spaces out of the box, managing and granting application roles, and maintaining the application when it is in use.

Ajax
A combination of asynchronous JavaScript, dynamic HTML (DHTML), XML, and XmlHttpRequest communication channel that enables requests to be made to the server without fully re-rendering the page. Ajax enables rich client-like applications to use standard internet technologies.

Announcements service
A WebCenter service that offers a quick, convenient way to create and widely distribute messages instantly or at a specific time.

API
Application Programming Interface. A set of exposed data structures and functions that an application can use to invoke services on an application object, such as a portlet.
Application Development Framework
See ADF.

application lifecycle
The process of creating and testing an application in a design time environment, deploying it to a production system, and then performing routine maintenance, such as monitoring performance and migrating customization data. The lifecycle of an application also includes performing further enhancements, restaging, and then redeploying the application to the production system.

Application Programming Interface
See API.

application role
Roles that are specific to a particular application and are stored in an application-specific stripe of the policy store.

application skin
Specifies the WebCenter Spaces application background color, screen fonts, and, with some skins, the shapes and images used for application buttons and icons. The WebCenter Spaces administrator chooses the default application skin. WebCenter users may change the application skin on the General tab of the Preferences dialog.

Applications pane
An area of the WebCenter Spaces Sidebar that provides convenient access to your frequently used applications. See Sidebar.

authenticated user
A user who is logged into a custom WebCenter application. Credentials of this user are verified against the identity store. By default, an authenticated user can access public information. To access secured information, such as pages and portlets, this user must be authorized through the policy and credential store.

Contrast with public users, who are not logged in, and can access public content only.

authentication
Identification of a user through an identity management system. You can require ADF authentication to enforce credentials for users to access the WebCenter application only (all ADF resources in the application remain accessible), or authentication and authorization to enforce credentials for users to access the WebCenter application and any ADF resources that have been secured in the application.

authorization
The policies that define the access rights of an individual or group to a secured resource. This resource may be a page or component within a page.

authorized user
An individual who has access to a secured resource. For non-public resources, this individual is also an authenticated user.

Blog service
A WebCenter service for integrating blogs into WebCenter applications.
**blog page**
A page that provides a personal record of an individual user’s experience and opinions. There are two kinds of blog: personal blogs are written by an individual; group blogs are written by several users.

**Box layout component**
An Oracle Composer layout component. A container that enables the placement of content on a WebCenter Spaces page. In Oracle Composer, a Box is rendered as a rectangle comprised of dashed lines. For designers of custom WebCenter applications, this is the runtime equivalent of a Panel Customizable component.

**BPEL**
Business Process Execution Language. An XML-based markup language for composing a set of discrete web services into an end-to-end process flow.

**business role page**
A page, created by the WebCenter Spaces administrator, specifically provided for a given role in an organization. Business role pages provide a targeted environment for users of a particular role by delivering information that is timely and relevant to individual roles without the noise of irrelevant information from other lines of business. Business role pages appear in the personal spaces of users classified under the specified role.

**caching**
The act of storing frequently accessed information, typically web pages, in a location where it can be accessed quickly to avoid frequent content generation.

See also **expiry-based caching** and **validation-based caching**.

**calendar overlay**
The ability to display multiple calendars in a single Events task flow.

**Change Mode Button component**
A component provided in the Oracle Composer tag library that enables users to change from page View mode to page Edit mode.

**Change Mode Link component**
A component provided in the Oracle Composer tag library that enables users to change from page View mode to page Edit mode.

**check out/check in**
A mechanism that enables a user to lock information, by checking it out, so that other users cannot modify that same piece of information. This prevents users from overwriting each other’s changes. After making modifications, the user releases it by checking it back in, making it available again for other users to modify.

**Child Components**
The components contained within a parent component. For example, the task flows contained within a Box layout component are the child components of the Box.

See also **Box layout component** and **parent component**.
chrome
Visual elements surrounding a portlet or task flow that provide an access point for actions, such as those on the Actions menu and those embedded in the chrome itself, such as the minimize icon or resize handles.

Community of Interest group space
A **group space** created using the Community of Interest template. This type of group space provides an optimal structure for supporting communities of people, joining together to learn more about a subject area through the sharing of expertise, ideas, and content.

component
An individual piece of an application, for example, a task flow, portlet, page, or layout element such as a box or image.

Component Catalog
A dialog, accessed from Oracle Composer, that provides access to all the content you can add to a WebCenter application page.

component developer
The developer who builds components (such as portlets, **JavaServer Faces** components, and web services).

Component Properties
A dialog, accessed from Oracle Composer, that provides access to a component’s parameters, display options, child components, style settings, and associated events.

Connections
In the **People Connections service**, a feature for establishing a social network with other application users.

container
An application program or subsystem in which the program building block, known as a **component**, is run.

content integration services
Services provided by **Oracle WebCenter** to enable developers to display content from a **content repository**, such as by creating **data controls**.

Content Presenter
A feature of the **Documents service** that enables end users to select and search content items and then display those items using available display templates. Oracle WebCenter provides out-of-the-box templates for displaying single and multiple content items on your pages. You can also define custom templates for the content that you want to display in your **custom WebCenter application**, or for selection by end users at runtime.

content repository
A specialized storage and management mechanism that provides such features as author-based versioning, full text searching, and content categorization and attribution. A content repository is optimized for storing unstructured information, which differentiates it from a data repository.
content repository data control

A data control sourced through a content repository. In a custom WebCenter application, you can create content repository data controls for the following content repositories: Oracle Portal, Oracle Universal Content Management, third-party repositories that support the Java Content Repository (JCR) standard, and your local file system.

credential provisioning page

A JSF (*.jspx) page used for authenticating to an external application. At runtime, the Credential Provisioning page displays login data fields consisting of the data fields specified through external application registration. Login information is passed to the producer, which in turn passes the login values to the external application. The application provides the producer with the requested portlets.

After authentication, the user's login credentials are preserved in a credential store, which subsequently supplies that information at future sessions. Unless his information changes, the user supplies his credentials only one time.

credential store

Provides storage for login credentials for its associated domain. It also preserves the login credentials that a user provides for authentication to an external application. Credential store is usually combined with the policy store as a single logical store.

Although the credentials stored in the credential store are used during subsequent logins for authentication, the main function of this store is to provide authorization for those accounts.

CSS

Cascading Style Sheet. A simple mechanism for ensuring a consistent look and feel or adding style, such as fonts, colors, and spacing, to web documents.

custom action

Icons or menu items rendered on the header or the Actions menu of a Show Detail Frame component surrounding a task flow. Custom actions can represent actions that were defined in the task flow when it was created. For example, at design time a developer can build a task flow with custom personalization settings. At runtime, users can access these settings through icons or Actions menu items provided in the task flow's surrounding chrome (or Show Detail Frame).

custom attribute

Specifies group space information in addition to that provided by the built-in attributes. Custom attributes can be used to determine the content of the components in a group space based on the parameter passed in. For example, a component can display data for a specific customer by passing in the customer ID. A custom attribute is simply a name value pair, for example customerId=400, orderId=11, userName=Smith, and so on. Custom attributes are stored within the group space template.

custom page

Any page created by a user rather than one provided out of the box.

custom resource catalog

A resource catalog that has been customized to control the components that are visible to specific users.
custom role
A user role created by an administrator or a group space moderator to meet a specific personal space or group space requirement.

custom WebCenter application
A custom WebCenter application is built on top of the ADF using the WebCenter Extension for Oracle JDeveloper. This application combines web content, portlets, content integration, and collaborative services for the end user. Developers and administrators can create a custom WebCenter application based on their roles and skill levels in the organization. See WebCenter application.

Customize mode
A portlet mode that enables users to set the default values for portlet preferences for all users.

customizable component
A WebCenter component that can be added to a page at runtime to enable end users to perform personalizations such as move, minimize, restore, or remove on content within those components. Customizable components are the Panel Customizable component and the Show Detail Frame component.

customization
An update that affects all users.
Contrast with personalization.

data control
A mechanism that provides an abstraction of the business service’s data model. The ADF data controls provide a consistent mechanism for clients and web application controllers to access data objects, collections, methods, and operations.
See also content repository data control.

default language (application-level)
A display language specified by the WebCenter Spaces administrator that is used when users log in to WebCenter Spaces. The WebCenter Spaces administrator sets the application-level default language on the General tab of the Administration page. Individual users can set their own user-level default language on the General tab of the Preferences dialog.

default language (user preference)
A user-specified display language that is rendered when the user logs in to WebCenter Spaces. This language selection lasts until the user specifies a different default language. It can be overridden by a session language, but returns as the default when the session cookie is purged or expires. This value is set on the General tab of the Preferences dialog.

default resource catalog
The resource catalog that is provided by default for an application. It contains all of the Oracle ADF components and portlets available to the application.
Contrast with custom resource catalog.
Default Server
See Integrated WLS.

deployment profile
A file used in application deployment that specifies the following types of information:

- The source files, deployment descriptors, and other auxiliary files that are packages
- The type and name of the archive file to be created
- Dependency information
- Platform-specific instructions
- Other information

WebCenter services provides a special deployment profile, the custom WebCenter application WAR deployment profile, that includes an option to export project metadata.

Design view (JDeveloper)
A view, in Oracle JDeveloper, that provides a WYSIWYG representation of a file.
See also Source view (JDeveloper).

Design view (WebCenter Spaces)
A view, in Oracle Composer, that provides a WYSIWYG representation of a page and its components.
See also Source view (WebCenter Spaces).

discoverable group space
A group space that can be found by anyone logged into WebCenter Spaces, for example through a search. A group space is made discoverable when the group space moderator enables the Discoverable setting. Discoverable group spaces are listed in My Group Spaces. Users wishing to join the group space can request membership through self-subscription (if enabled) or by contacting the group space moderator.

Discussions service
A WebCenter service that provides a means of creating and participating in text-based discussions with members of a particular group space.

display language
Controls the language in which application user interface elements, such as buttons, field labels, and screen text, are rendered in the browser. The order of precedence for WebCenter application display language settings from weakest to strongest is: browser setting, application setting, user preference setting, session setting, group space setting.

Document List Viewer task flow
A Documents service task flow that exposes a list of documents and, optionally, folders. The list is comprised of the contents of a specific folder or the results of a document search.
**Document Manager task flow**

A Documents service task flow that exposes all the folders and files available from the default content repository connection and default folder. Use to create, upload, and manage library content; to manage file versions; and to check files out and in.

**Documents page**

A predefined page provided in every WebCenter Spaces group and personal space that includes the Document Manager task flow for managing content.

**Documents service**

A WebCenter service that provides features for accessing, adding, and managing files; creating and managing file folders; configuring file and folder properties; and searching file and folder content.

The Documents service provides four task flows: Content Presenter, Document List Viewer, Document Manager, and Recent Documents.

**domain**

Any tree or subtree within the Domain Name System (DNS) namespace. Domain most commonly refers to a group of computers whose host names share a common suffix: the domain name.

**dynamically-generated page**

A page that displays as the result of a user action, such as a search or a click on a tag. As the name suggests, dynamically-generated pages are not stored, but rather are created as and when needed.

**EAR**

Enterprise Archive file. A Java EE archive file that is used in deploying applications on a Java EE application server. custom WebCenter applications are deployed using both a generic EAR file, which contains the application and the respective runtime customization, and a targeted EAR file, which contains only the application for deployment to the application server. EAR files simplify application deployment by reducing the possibility of errors when moving an application from development to test, and test to production.

See also **WAR**.

**ECMA-262 specification**

A standardization of scripting programming languages, such as ECMAScript and JavaScript.

**ECMAScript**

A scripting programming language, standardized by Ecma International according to the ECMA-262 specification. Frequently referred to as JavaScript or JScript, which are both extensions of the ECMA-262 specification.

**Edit Defaults mode**

(JSR 168 portlets only.) A portlet mode that enables personalization of a JSR 168 portlet. Edit Defaults mode is a display mode for the JSR 168 portlet's properties. In a custom WebCenter application, the Edit Defaults mode displays on the portlet's Actions menu as the Customize command.

See also **Edit mode**.
Edit mode
A portlet mode that enables personalization of the portlet for each user, for each instance.
See also Edit Defaults mode.

edit mode
A view mode that enables users to modify the content, style, and layout of a page. Access edit mode by choosing Edit Page from the Page Actions menu.

EL
Expression Language. Provides a shorthand way of working with web application data by providing operators for retrieving and manipulating application data residing in a Java EE web container. In a custom WebCenter application, EL expressions are encapsulated in the characters "#{" and "}" and typically come in the form #{object.data} where object represents any Java object or ADF component placed in the Java EE web container's page, request, session, or application's scope.

Enterprise Archive file
See EAR.

enterprise mashup
An application that enables users to bring all sorts of content and services together in a single place.

Events service
A WebCenter service that provides group calendars, which you can use to schedule meetings, appointments, and so on. You can integrate the Events service with a Microsoft Exchange Server to provide personal calendars for individual users. This service is available only in WebCenter Spaces, and not in custom WebCenter applications.

expiry-based caching
A caching method that uses a retention period to specify how long the item is valid in the cache before a refresh is required. When there is a request for the item beyond the retention period, it is refreshed in the cache.
See also validation-based caching.

Expression Language
See EL.

external application
Applications that do not delegate authentication to the single sign-on server. Instead, they display HTML login forms that ask for application user names and passwords. At the first login, users can choose to have the single sign-on server retrieve these credentials for them. Thereafter, they are logged in to these applications transparently.

farm
A collection of components managed by Fusion Middleware Control. A farm can contain a Managed Server domain and other Oracle Fusion Middleware system components that are installed, configured, and running on the domain.
favorites
A personal list of links to favorite WebCenter Spaces pages and external web sites.

Federated Portal Adapter
See FPA.

Feedback
In the People Connections service, a feature for posting informal appraisals for and receiving informal appraisals from other application users.

FOD
Fusion Order Demo. An enterprise application built using Oracle Fusion Middleware, including Oracle WebCenter, used to provide examples of WebCenter functionality.

FPA
Federated Portal Adapter. A component of Oracle Portal that enables Oracle Portal instances to share their database portlets through the web portlet interface. Using the FPA, Oracle Portal database portlets, including PL/SQL portlets, Portlet Builder portlets, and page portlets can be made available for use in WebCenter applications.

Full Screen Mode (WebCenter Spaces)
A view mode that opens the group space to occupy the entire screen, thus maximizing the display space. The Sidebar is not displayed in Full Screen Mode.

Full Screen mode (Portlets)
(PDK-Java portlets only.) A portlet mode that provides more content than can be shown in the portlet when it is sharing a page with other portlets.

Fusion Middleware Control
A browser-based management application that is deployed when you install Oracle WebCenter. From Fusion Middleware Control, you can monitor and administer a farm (such as Oracle WebCenter).

Fusion Order Demo (FOD)
See FOD.

Group Project group space
A group space created using the Group Project template. This type of group space provides an optimal structure for supporting a core project team where each member might come from a different department but all members contribute toward meeting a common goal.

group space
A work area within WebCenter Spaces that supports a group of people of any size that is organized around an area of interest or a common goal.

group space icon
An image displayed alongside group space names on the Group Spaces page in My Group Spaces to help other users with identification and location.
**group space logo**
An image displayed on the group space Home page to provide a visual identity for the group space. Group space logos also display alongside the group space name at the top of the page in Full Screen Mode.

**group space member**
A user who is participating in a group space. Members can be added or invited to a group space, or they can subscribe to a group space themselves if self-registration is enabled.

**group space owner**
A user who initially created a group space. The group space owner is automatically also a moderator of the group space.

**group space template**
A starting point for group space creation. WebCenter Spaces includes three templates to get you started: Group Project, Community of Interest, and Blank, but you can turn any group space into a template to use it as the starting point for other similar group spaces.

**Group Space Unavailable page**
A predefined page that displays when a group space member tries to open a group space that is temporarily offline. Moderators can customize this page.

**HA**
High Availability. A collection of solutions to ensure that your applications meet the required availability to achieve your business goals, eliminating single points of failure with no or minimal outage in service.

**Help mode**
A portlet mode that displays usage information about the functionality of the portlet.

**High Availability**
See HA.

**HTML Markup layout component**
An Oracle Composer layout component. A simple HTML component that renders raw HTML and JavaScript mark-up inline on the page.

**Hyperlink layout component**
An Oracle Composer layout component. A link to an internal or external web page. For designers of custom WebCenter applications, this is the runtime equivalent of a Go Link component.

**IDE**
Integrated Development Environment. A visual application development tool containing editors, debuggers, screen painters, object browsers, and the like. Oracle JDeveloper is an example of an IDE.

**Identity Propagation**
For a custom WebCenter application and associated content repositories, selecting this option allows propagation of current user's identity across the application and
processes. The propagated identity is verified on the receiver’s side, and then it is used to make decisions such as assigning role based access control.

**Image layout component**
An Oracle Composer layout component. An illustration that can include a hyperlink. For designers of custom WebCenter applications, this is the runtime equivalent of an Image Link component.

**IMP service**
See [Instant Messaging and Presence service](#).

**initialization parameters**
The parameters initialized upon the start-up of a standard JSR 168 portlet. Initialization parameters provide an alternative to JNDI (Java Naming and Directory Interface) variables. Use initialization parameters instead of JNDI to configure the behavior of all of the different components of the portlet—for example, servlets and other portlets—in a compatible way. In Oracle WebCenter, initialization parameters are entered into the `portlet.xml` file.

**Instant Messaging and Presence service**
A WebCenter service that enables users to observe the presence status of other authenticated users and provides instant access to interaction options, such as instant messages, emails, and phone calls.

**Integrated Development Environment**
See IDE.

**Integrated WLS**
Integrated WebLogic Server. A WLS instance used as a platform for pretesting WebCenter application deployments on a local computer. Integrated WLS also contains preconfigured portlet producers and several useful prebuilt portlets.

**JAAS**
Java Authentication and Authorization Service (JAAS) is a Java package that enables applications to authenticate and enforce access controls upon users. JAAS is designed to complement Java 2 security and implements a Java version of the standard Pluggable Authentication Module (PAM) framework. This enables an application to remain independent from the authentication service, and supports the use of custom authentication modules.

JAAS extends the access control architecture of the Java 2 Security Model to support subject-based authorization. It also supports declarative security settings in deployment descriptors instead of being limited to code-based security settings.

**Java Authentication and Authorization Service**
See JAAS.

**Java Content Repository**
See JCR 1.0.

**Java EE**
Also known as Java EE 5. Java Enterprise Edition 5 Platform. A platform that enables application developers to develop, deploy, and manage multitier, server-centric, enterprise-level applications. The Java EE platform offers a multitiered distributed
application model, integrated XML-based data interchange, a unified security model, and flexible transaction control. You can build your own Java EE portlets and expose them through web producers.

**Java Enterprise Edition 5 Platform**
See Java EE.

**Java Portlet Specification**
Standardizes how components for portal servers are to be developed. This specification defines a common portlet API and infrastructure that provides facilities for personalization, presentation, and security. Portlets using this API and adhering to the specification are product-agnostic, and can be deployed to any portal product that conforms to the specification. See also JSR 168.

**Java Specification Request**
See JSR 168.

**JavaServer Faces**
See JSF.

**JavaServer Page**
See JSP.

**JCR 1.0**
Java Content Repository 1.0. Also known as JSR 170. It proposes a standard access and interaction API for content repositories, much like JDBC does for databases.

**JDeveloper**
See Oracle JDeveloper.

**JSF**
JavaServer Faces. A standard Java framework for building web applications. It simplifies development by providing a component-centric approach to developing Java web user interfaces. JSF offers rich and robust APIs that provide programming flexibility and ensures that applications are well designed with greater maintainability by integrating the Model-View-Controller (MVC) design pattern into its architecture. As JSF is a Java standard developed through Java Community Process, development tools like Oracle JDeveloper are fully empowered to provide easy to use, visual, and productive development environments for JSF.

**JSF JSP**
JavaServer Faces JavaServer Page. JSF JSPs differ from plain JSPs through their support of Oracle ADF Faces components for the user interface and JSF technology for page navigation. JSF JSP pages leverage the advantages of the Oracle Application Development Framework (Oracle ADF) by using the ADF Model binding capabilities for the components in the pages.

**JSP**
JavaServer Page. An extension to servlet functionality that provides a simple programmatic interface to web pages. JSPs are HTML pages with special tags and embedded Java code that is executed on the web or application server. JSPs provide dynamic functionality to HTML pages. They are actually compiled into servlets when first requested and run in the servlet container.
See also JSP tags.

**JSP tags**
Tags that can be embedded in JSPs to enclose Java code. These tags use the `<jsp:` syntax and enclose action elements in the JSP with `begin` and `end` tags similar to XML elements.

**JSR 168**
Java Specification Request (JSR) 168. Defines a set of APIs for building standards-based portlets using Java. Portlets built to this specification can be rendered to a portal locally or deployed to a WSRP container for rendering portlets remotely. For more information, see [http://jcp.org/en/jsr/detail?id=168](http://jcp.org/en/jsr/detail?id=168).

**JSR 170**
See JCR 1.0

**JSR 301**
See Oracle JSF Portlet Bridge.

**keystore**
A file that provides information about available public and private keys that are used for authentication and data integrity. User certificates and the trust points needed to validate the certificates of peers are also stored securely in the keystore.

**layout box**
A container that enables placement of content on a WebCenter Spaces page.

**layout component**
An object for enhancing the usefulness and appearance of a given page. Layout components include layout boxes, a rich text editor, images, hyperlinks, and so on.

**Layout Customizable component**
A component provided in the Oracle Composer tag library that enables users to select from a set of predefined layouts (for example, two column, three column, two row, and so on) and apply it to the page. Users can apply these layouts to a particular area of the page or to the entire page.

**LDAP**
Lightweight Directory Access Protocol. A standard, extensible directory access protocol. It is a common language that LDAP clients and servers use to communicate. The framework of design conventions supporting industry-standard directory products, such as the Oracle Internet Directory.

**lifecycle**
See application lifecycle.

**Lightweight Directory Access Protocol (LDAP)**
See LDAP.

**Links service**
A WebCenter service that provides a means of creating a bidirectional association between two objects, thus setting up easy access between those objects.
List Manager
A task flow of the Lists service that provides access to all the tools for creating and revising lists and list content and to all of a group space's current lists.

Lists page
A predefined page that displays the group space's current lists.

Lists Viewer
A task flow of the Lists service that provides a means of placing a particular list on a group space page.

Mail service
A WebCenter service for exposing familiar email functionality in WebCenter applications.

Managed Server
In a production environment, a Managed Server hosts applications and the resources needed by those applications. A domain, which is a logically related group of Oracle WebLogic Server resources, can have any number of Managed Servers. An Administration Server manages these servers.

mashup
A web application that enables end users to pull information from different sources to create a personalized application that exactly meets their individual requirements.

See also enterprise mashup.

MBean Browser
In Fusion Middleware Control, MBean browsers enable the administrator to perform specific monitoring and configuration tasks and browse MBeans for an Oracle WebLogic Server or a selected application.

MDS
Oracle Metadata Services. A core technology of the Application Development Framework. MDS provides a unified architecture for defining and using metadata in an extensible and customizable manner.

MDS repository
An application server and Oracle relational database that keep metadata in these areas: a file-based repository, dictionary tables accessed by build-in functions, and a metadata registry. One of the primary uses of MDS is to store customizations and persisted personalization for Oracle applications.

Message Board
In the People Connections service, a feature for posting messages to and receiving messages from other application users.

Model-View-Controller
See MVC.

moderator
A WebCenter Spaces user who is responsible for managing a particular group space. A group space moderator can add and remove members, invite new members, enable
self registration, provide and update group space metadata, and manage the services available to the group space.

**Movable Box layout component**

An Oracle Composer layout component. A container that enables the placement of content on a WebCenter Spaces page. Movable Boxes, along with their content, can be moved around on the page. For designers of custom WebCenter applications, this is the runtime equivalent of Show Detail Frame component.

**MVC**

Model-View-Controller. A classic design pattern often used by applications that need the ability to maintain multiple views of the same data. The MVC pattern hinges on a clean separation of objects into one of three categories: models for maintaining data, views for displaying all or a portion of the data, and controllers for handling events that affect the model or views. Because of this separation, multiple views and controllers can interface with the same model. Even new types of views and controllers that never existed before, such as portlets, can interface with a model without forcing a change in the model design.

**My Group Spaces page**

A predefined page that displays a list of all the group spaces and group space templates available to the currently logged in user. This includes group spaces of which the user is a member, group spaces marked as discoverable, and group spaces that are public and available to everyone.

**navigation parameter**

Parameters in a WSRP container that map to the render parameters with the same name in JSR 168 portlet code. Navigation parameters are exposed by the portlet to the consumer. The consumer stores and manages parameter values and sends them on every invocation to the portlet. Navigation parameters are a WSRP version 2 feature.

**Notes service**

A WebCenter service that provides useful features for writing personal notes and reminders. This service is available only in WebCenter Spaces, and not in custom WebCenter applications.

**OAM**

See Oracle Access Manager (OAM).

**OHS**

See Oracle HTTP Server (OHS).

**OmniPortlet**

A component of Oracle WebCenter that enables you to inject portal-like capabilities, such as portlets, content integration, and customization, into your Oracle ADF Faces applications.

**Oracle Access Manager (OAM)**

Part of Oracle's enterprise class suite of products for identity management and security, Oracle Access Manager provides a wide range of identity administration and security functions, including several single sign-on options for WebCenter Spaces and WebCenter custom applications. OAM is the recommended single sign-on solution for Oracle WebCenter 11g installations.
**Oracle ADF Faces**

Oracle ADF Faces is a rich set of user interface components based on the new JavaServer Faces JSR (JSR 127). Oracle ADF Faces provide various user interface components with built-in functionality, such as data tables, hierarchical tables, and color and date pickers, that can be customized and reused in an application.

**Oracle Composer**

A seamlessly integrated environment for populating, revising, and configuring WebCenter application pages. It enables users to easily build or revise page layout and content. It also provides the means of adding different components, such as task flows, portlets, content, and other objects, onto a page and then linking those components for a more relevant or personalized view of the information.

**Oracle Content Server**

A content repository for building secure business libraries with check in and check out, revision control, and automated publishing in web-ready formats. Current information is available to authorized users anytime, anywhere. Oracle Content Server is a component of Oracle Universal Content Management.

**Oracle WebCenter Discussions Server**

Enables integration of discussion forums and announcements into WebCenter application.

**Oracle Enterprise Manager**

A component that enables administrators to manage Oracle Fusion Middleware services through a single environment. The Fusion Middleware administrator uses Enterprise Manager to configure, manage, and monitor WebCenter applications.

**Oracle HTTP Server (OHS)**

Software that processes web transactions that use the Hypertext Transfer Protocol (HTTP). Oracle uses HTTP software developed by the Apache Group.

**Oracle Internet Directory**

Oracle’s LDAP V3 compliant LDAP server. It is used as a repository for provisioning users and groups. By default, the Oracle Single Sign-On (OSSO) authenticates user credentials against Oracle Internet Directory information about dispersed users and network resources. Oracle Internet Directory combines LDAP version 3 with the high performance, scalability, robustness, and availability of the Oracle database.

**Oracle JDeveloper**

Oracle JDeveloper is an integrated development environment (IDE) for building applications and web services using the latest industry standards for Java, XML, and SQL. Developers can use Oracle JDeveloper to create Java portlets.

**Oracle JSF Portlet Bridge**

Based on and conforming to JSR 301, the Oracle JSF Portlet Bridge enables application developers to expose a JSF application or task flow as a JSR 168 portlet for consumption in another application.

**Oracle Metadata Services**

See MDS.
Oracle Portal
A component used for the development, deployment, administration, and configuration of enterprise class portals. Oracle Portal incorporates a portal building framework with self-service publishing features to enable you to create and manage information accessed within your portal.

Oracle SES
Oracle Secure Enterprise Search (SES) provides easy-to-use search for public and secure data. It is included with Oracle WebCenter. You can override the default search adapters in WebCenter Spaces and use Oracle SES, which provides unified ranking results. The results are listed together, instead of being clustered into separate sections for Documents, Discussions, and so on, with the most relevant items appearing first.

Oracle Single Sign-On (OSSO)
A component that enables users to log in to all features of the Oracle Fusion Middleware product suite, and to other web applications, using a single user name and password.

Oracle SOA Suite
A middleware component of Oracle Fusion Middleware. Oracle SOA Suite enables services to be created, managed, and orchestrated into SOA composite applications. Composites enable you to easily assemble multiple technology components into one SOA composite application. Oracle SOA Suite plugs into heterogeneous infrastructures and enables enterprises to incrementally adopt SOA.

Oracle Technology Network
See OTN.

Oracle Universal Content Management
A consolidated content management application that provides multisite web content management, document management, digital asset management and records management.

Oracle WebCenter
A suite of services that enables you to build custom WebCenter applications. Oracle WebCenter reduces the front-end labor historically required to bring necessary business components to the user by capitalizing on the notion of Service Oriented Architecture (SOA). The suite includes a wide range of plug-and-play products, tools, and services that make it easy to build the applications your users need. Oracle WebCenter includes:

- WebCenter services
- Oracle WebCenter Framework
- content integration services
- ADF
- Oracle SES
- Oracle WebCenter Discussions Server
- Mobile Services
- Portlet Pack
Oracle WebCenter Framework
A set of features provided by Oracle WebCenter that augments the Java Server Faces (JSF) environment by providing additional integration and runtime customization options. It is the basis of Oracle WebCenter and supports the creation and execution of context-rich applications, which can come in the form of human interaction, files, and documents, or a clear representation of where the user is within a complex work process. It includes such features as:

- Portlet support
- Content integration services
- Oracle JSF Portlet Bridge
- Search framework
- Customizable components

Oracle WebLogic Server Administration Console
A browser-based, graphical user interface to manage a WebLogic Server domain. Use to:

- Configure, start, and stop WebLogic Server instances
- Configure WebLogic Server clusters
- Configure WebLogic Server services, such as database connectivity (JDBC) and messaging (JMS)
- Configure security parameters, including creating and managing users, groups, and roles
- Configure and deploy your applications
- Monitor server and application performance
- View server and domain log files
- View application deployment descriptors
- Edit selected run-time application deployment descriptor elements

Oracle WebCenter Wiki and Blog Server
Supports integration of wikis and blogs into WebCenter applications. It also supports features that enable application users to create their own wikis and blogs.

OTN
Oracle Technology Network. The online Oracle technical community that provides a variety of technical resources for building Oracle-based applications. You can access OTN at http://www.oracle.com/technology/.

Page Customizable component
A component provided in the Oracle Composer tag library that defines the editable area of a page at runtime. Within this area, users can edit properties for a component, add content to the page, arrange content, and so on.

Page parameter
A parameter associated with a page that can be used to store values that can then be passed to the components on the page. It also enables your page to take values through its URL. Page parameters are defined using the  tag at the top of your PageDef.xml. You can bind page parameters to your page variables.
Page Properties
A dialog, accessed from Oracle Composer, that provides access to a page's display options, security settings, and parameters.

page scheme
Determines the background image used in the page. WebCenter Spaces provides several default page schemes and an option for specifying a custom page scheme.

Page service
A service for creating new pages and task flows in your application at runtime.

page style
Determines the initial page structure, for example one column or two column. Some default page styles also include the task flows, components, and page properties useful for a particular purpose. For example, a page created using the Text page style includes a Text layout component.

page variable
A variable that binds your public portlet parameter to the page. Page variables are defined within the <variableIterator> of your PageDef.xml. One page variable can be bound to multiple public portlet parameters.

Panel Customizable component
A component provided in the Oracle Composer tag library that provides a container region for a group of Oracle ADF components and portlets that are customizable at runtime. Any Show Detail Frame components and portlets added as child components to a Panel Customizable component can be moved or maximized with the Panel Customizable component.

parameter
A variable that controls the default behavior of task flow content and facilitates the wiring of a task flow to page parameters and page definition variables.

parent component
A component that contains other components, such as a Box layout component that contains task flows. The Box is the parent component of the task flows. In contrast, the task flows are the Box's child components.

See also Child Components.

participant
A WebCenter Spaces user who can manipulate the content of a group space. A participant can upload and share documents, initiate and take part in chats with other members, create discussion topics, create new or view existing lists.

PDK-Java
Java Portlet Developer Kit. The development framework used to build and integrate web content and applications with Oracle WebCenter. It includes toolkits, samples, and technical articles that help make portal development simple. You can take existing Java servlets, JSPs, URL-accessible content and web services and turn them into portlets. It is typically used by external developers and vendors to create portlets and services.
People Connections service
A WebCenter service that provides social networking tools for creating, interacting with, and tracking the activities of one's enterprise connections.

See also, Activity Stream, Connections, Feedback, Message Board, and Profile.

personalization
An update that affects only the user who made it.

personal page
A page created by a user in his or her personal space. Personal pages are viewable by other users only if specifically granted access by the user who created the page.

personal profile
A page that displays a user's personal information such as email address, phone number, office location, department, manager, direct reports, and so on.

See also, Profile.

personal space
A work area within WebCenter Spaces that provides individual users with a private space for storing personal content, keeping notes, viewing and responding to assignments, maintaining a list of online buddies, and performing many other tasks relevant to their unique working day. Users can also extend this environment by creating additional personal pages and custom content.

portal
A common interface (that is, a web page) that provides a personalized, single point of interaction with web-based applications and information relevant to individual users or class of users.

Portal Developer Kit
See PDK-Java.

portlet
A reusable web component that can draw content from many different sources. Portlets can display excerpts of other web sites, generate summaries of key information, perform searches, and access assembled collections of information from a variety of data sources. Because different portlets can be placed on a common page, the user receives a single-source experience, even though the content may be derived from multiple sources. Portlet resources include the many prebuilt portlets available out of the box and programmatic portlets built through WebCenter's JSR 168, PDK-Java Portlet wizards, and other portlet building tools.

portlet mode
The ways by which a portlet can be called to display information. These methods include:

- Shared Screen mode or View mode
- Edit mode or Edit Defaults mode
- Customize mode
- Help mode
- About mode
Full Screen mode (Portlets) or Show Details Page mode

Portlet Producer Application template
An application template, provided by JDeveloper, for creating an application with the recommended projects and technology scopes required for developing portlets. The Portlet Producer Application template consists of a single project scoped for portlet creation (Portlets).

See also WebCenter Application template.

predefined page
A page created by WebCenter Spaces to perform a specific function. Examples of predefined pages include, Welcome pages, Search pages, and Documents pages.

Predeployment Tool
A utility for custom WebCenter applications that assists you in configuring your target system with the new producer registrations you have added to your application in Oracle JDeveloper. You must run this utility before deploying your application. You can also use this utility after deployment to migrate metadata from stage to production, for example, to export and import your customizations. This tool also enables you to define the MDS repository location to allow run-time customizations to be migrated.

pretty URL
A shortened version of a page’s URL that hides the complexity of the real web address.

private parameter
A portlet parameter that is known only to the portlet itself and has no connection to the page on which the portlet resides.

Contrast with public parameter.

producer
A communication link between portlet consumers (such as a custom WebCenter application or a portal). When a consumer application renders a portlet, it calls the producer of that portlet, which in turn executes the portlet and returns the results in the form of portlet content. A producer can contain one or more portlets. A portlet can be contained by only one producer.

Oracle WebCenter supports two types of producers:

- Oracle PDK-Java producers: Deployed to a Java EE application server, which is often remote and communicates through Simple Object Access Protocol (SOAP) over HTTP.
- Web Services for Remote Portlets (WSRP): A web services standard that enables the plug-and-play of visual, user-facing web services with portals or other intermediary web applications. Being a standard, WSRP enables interoperability between a standards-enabled container based on a particular language (such as JSR 168, .NET, Perl) and any WSRP portal. A portlet (regardless of language) deployed to a WSRP-enabled container can be rendered in any application that supports this standard.

Profile
In the People Connections service, a feature for viewing and managing information about yourself, such as your contact information, manager, and direct reports, and for viewing this information about other application users.
programmatic portlets
Portlets constructed in a non-declarative manner using APIs. Also referred to as hand- or manually-coded portlets.

public group space
A group space that is available to all users, even those who are not logged in to WebCenter Spaces.

public page
A page within WebCenter Spaces that is available to all users, even those who are not logged in to WebCenter Spaces.

public parameter
A portlet parameter that is known to the page and bound to it by way of a page variable.
Contrast with private parameter.

public user
A user who can access, but is not logged into, a WebCenter application. A public user can view any page that has been marked as public, but cannot personalize or edit any content, or view pages that have any form of access control.
Contrast with authenticated user.

Recent Activities service
A WebCenter service that provides a means of tracking recent activities in a WebCenter application.

Recent Documents task flow
A Documents service task flow that exposes the files most recently modified in some way.

resize handle
A user interface element in a task flow chrome increasing or decreasing the height of the task flow.

Resource Action Handling framework
Enables services that expose custom resources to be viewed, searched, and tagged.

Resource Catalog
A catalog that provides a federated view of one or more otherwise unrelated repositories in a unified search and browse user interface. Resources are created and published in their source repository and are then exposed to the developer in JDeveloper’s Resource Palette and to the end user in the Resource Catalog Viewer. Resource catalogs can contain layout components, Oracle ADF components, portlets, service task flows, and documents.

Resource Index
The starting point for accessing WebCenter REST APIs. Sending a GET request to the Resource Index URI returns a list of links to entry points for all available services.
**resource type**
Defines the type of resource that a WebCenter REST API link identifies. Use resource types to determine the response bodies for GET requests and allowable request bodies for POST and PUT. Also use `resourceType` attributes on entities to uniquely identify their type.

**REST APIs**
Oracle WebCenter provides a set of web-based REST (REpresentational State Transfer) APIs for retrieving and modifying server data dynamically from the client. REST APIs are available for Discussions service, People Connections service, and WebCenter Spaces.

**Reverse Proxy Server**
A server process that hides the physical location of internal servers by exposing the servers as a single public site. Requests to the public site are routed to the appropriate internal server.

**Rich Text portlet**
A portlet, based on the WSRP standard, offering browser-based rich text editing at runtime on a deployed Oracle ADF JavaServer Faces JSP.

**RSS service**
A WebCenter service that provides a means of publishing content from other services as news feeds. The RSS service supports both RSS 2.0 and Atom 1.0 formats.

**Search page**
A predefined page for running searches, creating and managing saved searches, and viewing and refining search results.

**Search service**
A WebCenter service that enables the discovery of information and people in a WebCenter application, returning only the results users are authorized to see.

**Secure Enterprise Search**
See Oracle SES.

**secured application page**
A page created by a user that has not been made available to public users.

**Self-Registration page**
A predefined page where users can register with WebCenter Spaces, thus creating themselves an identity store login account. Administrators can customize certain aspects of this page.

**Self-Subscription page**
A predefined page where users can register to become members of a group space. Moderators can customize certain aspects of this page.

**service ID**
In Expression Language, the string that identifies a particular service. For example, the string oracle.webcenter.collab.announcement is the service ID for the Announcements service.
A PDK-Java producer’s unique identifier. PDK-Java enables you to deploy multiple producers under a single adapter servlet. Different producers are identified by their unique service IDs. A service ID is required only when a service ID/producer name is not appended to the URL endpoint.

**Service Oriented Architecture**

See [SOA](#).

**servlet**

A Java program that usually runs on a [Web server](#), extending the web server’s functionality. HTTP servlets take client HTTP requests, generate dynamic content (such as through querying a database), and provide an HTTP response.

**session language**

A display language specified by the user that remains in effect for the life of the session cookie (from log on to log off). If the user clears browser cookies, the display language reverts to the user-level default language, if specified, then to the application-level default language set by the WebCenter Spaces administrator. Set the session language in the Change Language pop-up, accessible from the Welcome page.

**Shared Screen mode**

A [portlet mode](#) that renders the body of the portlet and enables you to display a portlet on a page that can contain other portlets. Every portlet must have at least a Shared Screen mode.

See also [View mode](#).

**Show Detail Frame component**

A component provided in the Oracle Composer tag library that renders a border or chrome around the child component. It provides a header with an Actions menu and thereby provides user interface (UI) controls to customize the display of the child component. However, to customize the display of the child component, the Show Detail Frame component must be included inside a Panel Customizable component.

**Show Details Page mode**

A [portlet mode](#) that provides full-browser display of the portlet. For example, a portlet in Show Page mode could be limited to displaying only the ten most recently submitted expense reports, while the same portlet in Show Details Page mode could show all submissions.

Contrast with [Show Page mode](#).

**show modes**

Types of [portlet modes](#) encompassing [Show Page mode](#) and [Show Details Page mode](#).

**Show Page mode**

A [portlet mode](#) that provides a smaller portlet display to allow space for additional portlets and other objects in the browser window. For example, a portlet in Show Page mode could be limited to displaying only the ten most recently submitted expense reports, while the same portlet in Show Details Page mode could show all submissions.

Contrast with [Show Details Page mode](#).
Sidebar
A panel in WebCenter Spaces that provides quick access to tools and information essential to personal productivity, including mail, personal contacts, and so on.

**skin**
A style sheet based on the CSS 3.0 syntax specified in one place for an entire application. Instead of providing a style sheet for each component, or inserting a style sheet on each page, you can create one skin for the entire application.

**SOA**
Service Oriented Architecture. A design methodology aimed at maximizing the reuse of application services.

**Source view (JDeveloper)**
A view, in Oracle JDeveloper, that provides a way to directly edit the source code of a file.

**Source view (WebCenter Spaces)**
A view, in Oracle Composer, that provides a selectable structural representation of a page and its components.
See also Design view (WebCenter Spaces).

**struts**
A development framework for Java servlet applications based upon the MVC design paradigm.

**style properties**
Used to override the style information from the skin CSS to set specific instances of component display.

**Tags service**
A WebCenter service that enables users to apply their own terms to application objects, making it possible to search for those objects using personally meaningful terms.

**task flow**
A set of ADF Controller activities, control flow rules, and managed beans that interact to allow a user to complete a task. Task flows provide a modular approach for defining control flow in an application. Instead of representing an application as a single JSF page flow, developers can break it up into a collection of reusable task flows.

**task flow header**
An area at the top of a task flow that displays the task flow name and various tools for interacting with the task flow.

**template**
See group space template.

**Text layout component**
An Oracle Composer layout component. A rich text editor for providing static page text. For designers of custom WebCenter applications, this is the runtime equivalent of a Rich Text Editor component.
Unauthorized Access page
A predefined page that is shown when someone without access permission tries to open a page.

URL parameter
See private parameter.

validation-based caching
A caching method that uses a validation check to determine if the cached item is still valid.
Contrast with expiry-based caching.

viewer
WebCenter Spaces users who can look at the information in a group space but cannot contribute any of their own.

View mode
(JSR 168 portlets only.) A portlet mode that enables you to display a JSR 168 portlet on a page that can contain other portlets. It is the only required mode for JSR 168 portlets.
See also Shared Screen mode.

WAR
Web application archive file. This file is used in deploying applications on a Java EE application server. WAR files encapsulate in a single module all of the components necessary to run an application. WAR files typically contain an application's servlet, JSP, and JSF JSP components.
See also EAR.

Web 2.0
Technologies, such as wiki, RSS, and blogs, that enable the construction of highly interactive web applications.
See also WebCenter services.

Web Application Archive file
See WAR.

Web Clipping
A browser based declarative tool that enables developers and page designers to integrate any Web content with a WebCenter application. It is designed to provide quick integration by leveraging the existing user interface.

Web Clipping portlet
A browser-based declarative tool that enables you to integrate any web application with your custom WebCenter application. It is designed to give you quick integration by leveraging the web application's existing user interface. You can drag and drop Web Clipping portlets onto a *.jspx page.

Web Page layout component
An Oracle Composer layout component. A means of embedding another web site, wiki, or blog within the context of a WebCenter Spaces page. For designers of custom WebCenter applications, this is the equivalent of an Inline Frame component.
**Web server**
A program that delivers web pages.

**Web Services for Remote Portlets**
See [WSRP](#).

**WebCenter**
See [Oracle WebCenter](#).

**WebCenter application**
WebCenter applications encompass both custom WebCenter applications and WebCenter Spaces.

**WebCenter application administrator**
The administrator responsible for maintaining the WebCenter application. For example, in WebCenter Spaces, the administrator performs tasks such as implementing the branding for the WebCenter application, making new content available, modifying pages, and granting and revoking privileges.

Contrast with systems administrator who has administrative rights for entire Fusion Middleware functions. The Fusion Middleware administrator is also responsible for deploying, setting up, and configuring the WebCenter application, and performing on-going administrative tasks for the WebCenter application and other WebCenter components through Fusion Middleware Control.

**WebCenter application developer**
The developer who plans, builds, and maintains a custom WebCenter application using the Oracle Application Development Framework, Oracle JDeveloper, and Oracle WebCenter.

**WebCenter application end user**
The WebCenter application end user is the runtime user of the WebCenter application, who accesses pages, portlets, and content, and personalizes portlets (assuming the appropriate privileges).

**WebCenter Application template**
An application template, provided by JDeveloper, for creating an application with the recommended projects and technology scopes required for developing a WebCenter application. The WebCenter Application template consists of a project for the data model (Model) and a project for consuming portlets, components, and data controllers (ViewController).

See also [Portlet Producer Application template](#).

**WebCenter Extension for Oracle JDeveloper**
An extension available through the Oracle JDeveloper Update Wizard that installs the necessary libraries, templates, wizards, and dialogs needed to build and deploy custom WebCenter applications in Oracle JDeveloper.

**WebCenter Framework**
See [Oracle WebCenter Framework](#).
**WebCenter services**
A collection of Web 2.0 services that expose social networking and personal productivity features through various services.

- Announcements service
- Blog service
- Discussions service
- Documents service
- Events service
- Instant Messaging and Presence service
- Links service
- Mail service
- Notes service
- People Connections service
- Recent Activities service
- RSS service
- Search service
- Tags service
- Wiki service
- Worklist service

**WebCenter Spaces**
A web-based application that offers the very latest technology for social networking, communication, collaboration, and personal productivity. WebCenter Spaces uses services and applications to bring everything together that users require to exchange ideas with others, keep track of personal and work-related tasks, interact with critical applications, and zero in on projects and interests; all within a single integrated environment.

WebCenter Spaces is a **WebCenter application**.

**WebCenter Spaces application administrator**
See administrator.

**WebCenter Spaces RSS reader**
An RSS reader provided with WebCenter Spaces that incorporates public news feeds from external sources onto application pages. This RSS reader is available only in WebCenter Spaces, and not in custom WebCenter applications.

**WebCenter systems administrator**
See administrator.

**WebLogic Server**
See **WLS**.

**Welcome page**
There are two types of Welcome page:
- Public Welcome page: A predefined page that users encounter before logging in to WebCenter Spaces.
- Personal Welcome page: A predefined page that introduces users to their personal space.

**wiki page**
A page that provides in-place editing using HTML or a simple mark-up language. Any user with sufficient privileges can add, revise, and remove wiki content.

**Wiki service**
A WebCenter service for integrating wiki pages in WebCenter applications.

**WLS**
WebLogic Server. A scalable, enterprise-ready Java Platform, Enterprise Edition (Java EE) application server. The WebLogic Server infrastructure supports the deployment of many types of distributed applications and is an ideal foundation for building applications based on Service Oriented Architectures (SOA).

See also [Integrated WLS](#)

**WLST**
WebLogic Scripting Tool. A command line tool for managing Oracle Fusion Middleware components, such as Oracle WebCenter.

**Worklist service**
A WebCenter service that provides access to notifications, alerts, and BPEL tasks assigned to the current user.

**WSRP**
Web Services for Remote Portlets (WSRP) is a web services standard that allows the plug-and-play of visual, user-facing web services with portals or other intermediary web applications. Being a standard, WSRP enables interoperability between a standards-enabled container based on a particular language (such as JSR 168, .NET, Perl) and any WSRP portal. A portlet (regardless of language) deployed to a WSRP-enabled container can be rendered on any portal that supports this standard.

**XSL**
Extensible Stylesheet Language (XSL) is the language used within style sheets to transform or render XML documents.
A

About mode, 31-27
AboutPage.jsp file, A-12
Accordion View display template, 14-15
Actions menu
   Documents service task flows, 14-9
actionsCategory, B-36
Activity Stream
   about, 20-2
   compared to Recent Activities service, 20-3
   REST APIs, 20-36, 28-18
   supported site-level settings, 20-20
   task flows, 20-8
   Activity Stream, 20-8
   Activity Stream - Quick View, 20-8
   tracked activities, 20-2
adapters
   Documents service, 14-24
   Search service, 24-11
addMember API, 27-18
addon-config, B-27
addon-panels, B-34
add-ons
   about, 8-2
   creating, 8-4
   Customization Manager, 8-9
   excluding, 8-7
   registering in adf-config.xml, 8-7
   registering in extension file, 8-6
   rendering selectively, 8-10
address portable contact type, 28-12
ADF
   about, 1-2
   ADF data controls, 3-7
   adfc-config.xml file, 3-6
   adf-config.xml, 8-4, B-33
ADFm events
   cascading events, 34-44
   linking portlets, 34-42
adfp:portlet tag
   allModesSharedScreen attribute, 34-25
   attributes, setting, 34-20
   background attribute, 34-26
   binding attribute, 34-30
   contentInlineStyle attribute, 34-30
   customizationAllowed attribute, 34-30
   customizationAllowedBy attribute, 34-30
   displayActions attribute, 34-27
   displayHeader attribute, 34-26
   displayScrollBar attribute, 34-26
   displayShadow attribute, 34-26
   example, 34-18
   expansionMode attribute, 34-25
   height attribute, 34-24
   icon attribute, 34-24
   id attribute, 34-23
   iframeDtd attribute, 34-31
   inlineStyle attribute, 34-30
   isAboutModeAvailable attribute, 34-28
   isConfigModeAvailable attribute, 34-28
   isCustomizeModeAvailable attribute, 34-28
   isDetailModeAvailable attribute, 34-29
   isHelpModeAvailable attribute, 34-29
   isNormalModeAvailable attribute, 34-29
   isPersonalizeModeAvailable attribute, 34-29
   isPreviewModeAvailable attribute, 34-29
   isPrintModeAvailable attribute, 34-29
   partialTriggers attribute, 34-24, 34-28
   placement, 34-19
   rendered attribute, 34-26
   renderPortletInIFrame attribute, 34-25
   required attributes, 34-20
   shortDesc attribute, 34-26
   showMinimizeAction attribute, 34-27
   showMoveAction attribute, 34-27
   showRemoveAction attribute, 34-27
   showResizer attribute, 34-27
   submitUrlParameters attribute, 34-28
   title attribute, 34-23
   usage, 34-18
   width attribute, 34-23
adfp:portlet tag, 34-19
Announcements - Sidebar View task flow, 16-9
Announcements service
   about, 16-1
   adding to custom applications, 16-9
   Announcements - Sidebar View task flow, 16-9
   Announcements task flow, 16-8
   connection, 16-4
   customizing views, 16-12
   Discussions server, 16-4
Announcements task flow, 16-8
Apache Struts, creating portlets, 32-35
APIs
Discussions service, 17-18
group spaces, 27-4
Page service, 6-22, 6-36
People Connections service, 20-35 to 20-57
REST, 20-35 to 20-57
Search service, 24-11
Web Clipping Transport, 36-2, 36-7
Application Development Framework
See ADF
application links, 31-28
Application Resources panel
adding files and folders to page, 14-4
applications
consuming in WebCenter Spaces, 27-46
creating JSF portlets, 30-2
authentication
external application for PDK-Java, 37-26
message for PDK-Java, 37-25, 37-28
none for PDK-Java, 37-27
PDK-Java, 37-25
server for PDK-Java, 37-25
single sign-on PDK-Java, 37-26
BACK_LINK URL, 32-18
C
Caching
activating for PDK-Java portlets, 32-33
adding for PDK-Java portlets, 32-33, 32-34
expiry-based, 29-21
for Oracle PDK-Java portlets, 32-32
Java portlets, 29-22
OmniPortlet, 29-21
PDK-Java portlets, 32-32
portlets, 29-21
validation-based, 29-21
for Oracle PDK-Java portlets, 32-32
Web Clipping, 29-21
capabilities attribute, 28-6
Carousel View display template, 14-15
catalog definition, C-2
attributes, C-6
schema, C-3
catalog definition attributes
attribute, C-9
attributes, C-9
catalogDefinition, C-6
component, C-11
dynamicFolder, C-7
doctor, C-8
resource, C-10
schema, C-12
catalogDefinition, C-6
category tag, 13-42
Change Mode Button
adding to a page, 7-4
definition, 5-48
properties, B-4
Change Mode Link
adding to a page, 7-4
definition, 5-48
properties, B-4
CHANGE_PERSISTENCE, B-42
character sets
PDK-Java portlets, 31-16
Web Clipping, 36-9
communication
security for PDK-Java, 37-25
component, C-11
ComposerSessionOptionsFactory, 10-14
configuration
custom validators, 3-10
faces-config.xml file, 3-10
JSF servlet settings, 3-9
managed beans, 3-10
navigation rules, 3-10
servlet mapping, 3-9
web.xml file, 3-7
configuration parameters
connection to Oracle WebCenter adapter for
Microsoft SharePoint, 13-17
file system connection, 13-15
example, 2-6
requirements, 16-2
security, 16-10
Apache Struts, creating portlets, 32-35
APIs
Discussions service, 17-18
group spaces, 27-4
Page service, 6-22, 6-36
People Connections service, 20-35 to 20-57
REST, 20-35 to 20-57
Search service, 24-11
Web Clipping Transport, 36-2, 36-7
Application Development Framework
See ADF
application links, 31-28
Application Resources panel
adding files and folders to page, 14-4
applications
consuming in WebCenter Spaces, 27-46
creating JSF portlets, 30-2
authentication
external application for PDK-Java, 37-26
message for PDK-Java, 37-25, 37-28
none for PDK-Java, 37-27
PDK-Java, 37-25
server for PDK-Java, 37-25
single sign-on PDK-Java, 37-26
BACK_LINK URL, 32-18
best practices
About mode, 31-27
CSS for portlets, 31-23
Edit Defaults mode, 31-25
Edit mode, 31-24
for Java portlets, 31-21
Help mode, 31-26
HTML for portlets, 31-22
portlet navigation, 31-27
Preview mode, 31-26
bidirectional links, 22-3
blogs
adding, 21-3
adding by using a portlet, 21-3
adding by using IFRAME, 21-4
adding by using Web Services, 21-5
sample portlets, 21-3
URL formats for exposing blogs, 21-5
Web Services client stub generation, 21-14
Web Services example, 21-15
Web Services interface, 21-8
Web Services methods, 21-8
Web Services security, 21-13
bridge
See Oracle JSF Portlet Bridge
Buddies task flow, 18-16
Bulleted View display template, 14-15
Bulleted with Folder Label View display

Index-2
Oracle Content Server connection, 13-8
Oracle Portal connection, 13-11
configuring basic authentication, 37-11
connectionListName binding parameter, 20-31
connectionName parameter, 14-16, 14-19
Connections
about, 20-4
example site-level setting, 20-27
REST APIs, 20-40
supported site-level settings, 20-22
task flows, 20-9
Connections, 20-9
Connections - Card, 20-10
Connections - Quick View, 20-11
connections
accessing connection wizards, 3-27
Announcements service, 16-4
data source repositories, 3-27
Discussions service, 17-3
Documents, 14-2
files affected, 39-8
IMP service, 18-2
Links service, 22-4
Mail service, 19-4
Oracle Secure Enterprise Search (SES), 24-8
People Connections service, 20-5
portlet producers, 3-27
Recent Activities service, 25-3
RSS service, 15-3
Tags service, 23-4
constructResourceURL, 31-29
content
adding to a page, 13-28
content data controls
about, 13-4
advancedSearch method, 13-5
configuring, 13-49
creating, 13-49
default attributes, 13-4
editing, 13-53
getAttributes method, 13-6
getItems method, 13-4
getURI method, 13-6
methods, 13-4
parameters, 13-4
search method, 13-5
securing, 13-54
use case examples, 13-54
enable advanced search, 13-77
enable simple search, 13-76
file names as hyperlinks, 13-73
publish as links, 13-55
publish as tables, 13-62
publish as tree, 13-69
content integration
about, 1-7
Content Presenter task flow
adding runtime Edit action, 14-9
adding to page, 14-3
custom templates, 13-28
display templates, 13-3, 13-28, 14-12
EL expressions, 13-29
importing display templates, 13-46
modifying parameters at runtime, 14-11
multiple content templates, 13-38
overview, 13-2
parameters, 14-12
runtime behavior, 13-47
single content item display templates, 13-28
templates, 13-28
generating valid list, 13-48
importing, 13-46
testing, 13-46
templates registry, 13-3, 13-41
testing display templates, 13-46
content repositories
adding content to a page, 13-28
connecting, 14-2
Documents service, 14-2
content repository connections
editing application-specific connections, 13-26
editing common connections, 13-25
external application, 13-21
file system-based, 13-14
identity propagation, 13-21
Oracle Content Server-based, 13-7
Oracle Portal-based, 13-11
using existing repository connections, 13-27
custom actions
task flows, 9-14
custom actions
adding to a Show Detail Frame, 9-7
enabling at global level, 9-8
enabling at instance level, 9-7
custom actions
task flows, 9-14
custom certificates, 37-12
custom display templates
  see Content Presenter task flow, 13-28
custom property panels
  about, 8-2
  creating, 8-10
  excluding, 8-14
  overriding, 8-14
  registering, 8-11
custom Resource Catalog, 12-12
custom WebCenter applications
  about, 1-4
  consuming in WebCenter Spaces, 27-46
  consuming portlets, 34-1
  design considerations, 1-12
  planning, 1-1
custom-actions, B-37
customizable components
  about, 1-8
customizable page
  adding content, 7-16
  example, 7-13
  template, 7-10
customizableComponentsSecurity, B-36
customization, 5-2
  component properties, 5-17
  editing page properties, 5-29
  customization layers, 10-8
  Customization Manager, 5-33
  deleting customizations, 5-36
  displaying in Oracle Composer, 8-9
  downloading customizations, 5-35
  enabling parameter support, 8-30
  parameters, 8-30
  promoting customizations from a saved label, 5-36
  rendering, 8-9
  uploading customizations, 5-35
  customization manager, 8-3
  about, 8-3
  customization policies, 11-1
  customization restrictions, 11-4
  customizationAllowed, 11-13
  customizations
    exporting for WSRP 2.0, 32-7
    importing for WSRP 2.0, 32-7
  CVS
    importing files, 39-2
    using, 39-2

D

  data controls, 3-5
    ADF data controls, 3-7
    Page service, 6-1, 6-18
  data sources
    supported by portlets, 29-12
  database connection
    setting up for People Connections service, 20-5
    WebCenter schema, 22-4, 23-4
  DataDirect JDBC drivers
  registering with OmniPortlet, F-6
  datasource parameter, 14-13
  datasourceType parameter, 14-13
  Default Document Details View display template, 14-14
  Default List Item View display template, 14-14
  Default List Item View for Folders display template, 14-15
  Default View display template, 14-14
  deleteGroupSpace API, 27-18
  deployment
    packaging credentials, 38-7
    packaging identity data, 38-8
    policy migration, 38-8
    WebLogic Managed Server, creating, 38-10
    WebLogic Managed Server, provisioning, 38-10
  deployment descriptor file, 3-6, 3-7
  description tag, 13-42, 13-43
  design questions, 1-12
  design time at runtime, 5-1
  design time at runtime portlets, 29-12, 29-23
  DESIGN_LINK URL, 32-18
  develop first, add later portlets, 29-23
developer
  design considerations, 1-14
  disabling, Oracle Composer, B-6
  Discussion Forums task flow, 17-7
discussions
  access
    in custom WebCenter applications, 17-10
    categories, 17-1, 17-10
    forums, 17-1, 17-10
    watching, 17-14
    watching topics, 17-14
  Discussions - Popular Topics task flow, 17-7
  Discussions - Recent Topics task flow, 17-8
  Discussions - Sidebar View task flow, 17-8
  Discussions - Watched Forums task flow, 17-8
  Discussions - Watched Topics task flow, 17-8
  Discussions server, 17-1
  credentials, 17-10
  Discussions service
  about, 17-1
  adding to custom applications, 17-7
  APIs, 17-18
  connections, 17-3
customization views, 17-11
  Discussion Forums task flow, 17-7
  Discussions - Popular Topics task flow, 17-7
  Discussions - Recent Topics task flow, 17-8
  Discussions - Sidebar View, 17-8
  Discussions - Watched Forums task flow, 17-8
customizations
  example, 2-2
  requirements, 17-2
customizable components
Content Presenter, 14-12
see Content Presenter task flow, 13-28
display_maxConnections binding parameter, 20-31
display_numberOfColumns binding parameter, 20-31
display_numberOfRows binding parameter, 20-31
display_profileFormat binding parameter, 20-31
display_removalAllowed binding parameter, 20-32
display_sortedBy binding parameter, 20-32
displayCount binding parameter, 20-32
displayMessageSize binding parameter, 20-32
Document List View task flow
adding runtime Edit action, 14-9
adding to page, 14-3
parameters, 14-16
Document Manager task flow
adding to page, 14-3
parameters, 14-16
Documents service
about security, 14-20
adapters, 14-24
adding files and folders to a page, 14-4
adding task flows, 14-3
adding to Oracle Composer Catalog, 14-3
allowing runtime customization, 14-9
connections, 14-2
content repositories, 14-2
example, 2-2
integrating, 14-1
task flow parameters, 14-11
with other WebCenter services, 14-22
Documents Service task flows
adding files and folders, 14-4
Documents service task flows
adding to page, 14-2
adding to page from Application Resources panel, 14-4
adding to page from Oracle Composer, 14-8
adding to page from Resource Panel, 14-6
drop handlers, 8-22
about, 8-3
creating, 8-22
registering, 8-25
drop-handler, B-30
drop-handlers, B-30
dynamicFolder, C-7

E
Edit action
adding to Content Presenter task flow, 14-9
adding to Document List View task flow, 14-9
Edit Defaults mode, 31-25
Edit mode, 5-3, 31-24
EditDefaultsPage.jsp file, A-12
EditPage.jsp file, A-12
EL expressions
Content Presenter task flow, 13-29
e1builder-config, B-32
enableSecurity, B-36
encryption
message for PDK-Java, 37-25
escapes
for user input, PDK-Java, 37-28
event-handler, B-30
event-handlers, B-30
expiry-based caching, 29-21
exporting
customizations for WSRP 2.0, 32-7
expression builder
extending, 8-15
Expression Language
see EL expressions, 13-29
Ext, using with REST APIs, 28-20
external applications
associating with a PDK-Java producer, 34-13
associating with a WSRP producer, 34-8
authenticating, 29-29
authentication for PDK-Java, 37-26
deleting registration, 37-9
demote registration details, 37-8
registering, 37-6
security, 37-6
Web Clipping, 29-8
external links, 31-28

F
faces-config.xml file, 3-6, 3-10
Federated Portal Adapter
about, G-4
limitations, G-7
registering, G-4
troubleshooting, G-6
Feedback
about, 20-4
REST APIs, 20-51
supported site-level settings, 20-23
task flows, 20-11
Feedback, 20-11
Feedback - Quick View, 20-12
file
adding to a page, 14-4
file extensions, 3-9
files, 3-9
about, A-2
actions affecting, 39-7
adfc-config.xml, 3-6
deployment descriptor file, 3-6, 3-7
faces-config.xml file, 3-6, 3-10
for WebCenter applications, A-1
index.html file, 3-27
jsf-impl.jar, 3-6
managing, 39-7
provider.xml, F-11
WAR files, 3-23
WebCenter Preconfigured Server readme file, 3-25
web.xml file, 3-6, 3-7, 3-9
filter-config, B-30
Index-5
filterPattern binding parameter, 20-32
folder
adding to a page, 14-4
folder hierarchy in application, 3-6
form tag, IFRAME, 34-32
forms
building with URL types, 32-19
usesUpload attribute, 20-16, 20-17
FPA (Federated Portal Adapter)
about, G-4
FPA (Federated Portal Adapter)
limitations, G-7
registering, G-4
troubleshooting, G-6
frame sets in Web Clipping, 36-8
fromDate binding parameter, 20-32
Full Screen mode, 31-26
Fusion Order Demo
about, 2-1
WebCenter components, 2-2

G
getCreatedBy API, 27-22, 27-25
getCustomAttributes API, 27-22
getDescription API, 27-22, 27-25
getDisplayName API, 27-22, 27-25
getGroupSpaceMetadata API, 27-22
getGroupSpaces API, 27-21
getGroupSpaceState API, 27-23
getGroupSpaceTemplateMetadata, 27-25
getGroupSpaceTemplates API, 27-24
getGroupSpaceURL API, 27-26
group spaces, about, 27-2
group space parameter, 14-19
groupSpaceReference data type, 28-11
GroupSpaceWSClient
getGroupSpaces API, 27-21
GroupSpaceWSClient class, 27-14, 27-17
addMember API, 27-18
createGroupSpace API, 27-16
createGroupSpaceTemplate API, 27-17
deleteGroupSpace, 27-18
getGroupSpaceMetadata, 27-22
group spaces APIs
adding members, 27-18
creating group space attributes, 27-17
creating group spaces, 27-16
creating templates, 27-17
deleting group spaces, 27-18
inviting members, 27-19
removing members, 27-20
retrieving group space metadata, 27-22
retrieving list of public group spaces, 27-22
retrieving list of templates, 27-24
retrieving roles, 27-20
retrieving template metadata, 27-25
functionality in custom WebCenter applications, 27-4

G
getCreatedBy API, 27-22, 27-25
group spaces, about, 27-2
getCustomAttributes API, 27-22
getDescription API, 27-22, 27-25
getDisplayName API, 27-22, 27-25
group spaces API documentation, 27-29
groupSpace parameter, 14-19
GroupSpaceReference data type, 28-11
GroupSpaceWSClient
getGroupSpaces API, 27-21
GroupSpaceWSClient class, 27-14, 27-17
addMember API, 27-18
createGroupSpace API, 27-16
createGroupSpaceTemplate API, 27-17
deleteGroupSpace, 27-18
getGroupSpaceMetadata, 27-22
group spaces APIs
adding members, 27-18
creating group space attributes, 27-17
creating group spaces, 27-16
creating templates, 27-17
deleting group spaces, 27-18
inviting members, 27-19
removing members, 27-20
retrieving group space metadata, 27-22
retrieving list of public group spaces, 27-22
retrieving list of templates, 27-24
retrieving roles, 27-20
retrieving template metadata, 27-25
functionality in custom WebCenter applications, 27-4

G
getCreatedBy API, 27-22, 27-25
group spaces, about, 27-2
getCustomAttributes API, 27-22
getDescription API, 27-22, 27-25
getDisplayName API, 27-22, 27-25
group spaces API documentation, 27-29
groupSpace parameter, 14-19
GroupSpaceReference data type, 28-11
GroupSpaceWSClient
getGroupSpaces API, 27-21
GroupSpaceWSClient class, 27-14, 27-17
addMember API, 27-18
createGroupSpace API, 27-16
createGroupSpaceTemplate API, 27-17
deleteGroupSpace, 27-18
getGroupSpaceMetadata, 27-22
getGroupSpaces API, 27-24
group spaces APIs
adding members, 27-18
creating group space attributes, 27-17
creating group spaces, 27-16
creating templates, 27-17
deleting group spaces, 27-18
inviting members, 27-19
removing members, 27-20
retrieving group space metadata, 27-22
retrieving list of public group spaces, 27-22
retrieving list of templates, 27-24
retrieving roles, 27-20
retrieving template metadata, 27-25
functionality in custom WebCenter applications, 27-4

G
getCreatedBy API, 27-22, 27-25
group spaces, about, 27-2
getCustomAttributes API, 27-22
getDescription API, 27-22, 27-25
group spaces API documentation, 27-29
groupSpace parameter, 14-19
GroupSpaceReference data type, 28-11
GroupSpaceWSClient
getGroupSpaces API, 27-21
GroupSpaceWSClient class, 27-14, 27-17
addMember API, 27-18
createGroupSpace API, 27-16
createGroupSpaceTemplate API, 27-17
deleteGroupSpace, 27-18
getGroupSpaceMetadata, 27-22
getGroupSpaces API, 27-24
group spaces APIs
adding members, 27-18
creating group space attributes, 27-17
creating group spaces, 27-16
creating templates, 27-17
deleting group spaces, 27-18
inviting members, 27-19
removing members, 27-20
retrieving group space metadata, 27-22
retrieving list of public group spaces, 27-22
retrieving list of templates, 27-24
retrieving roles, 27-20
retrieving template metadata, 27-25
functionality in custom WebCenter applications, 27-4

G
getCreatedBy API, 27-22, 27-25
group spaces, about, 27-2
getCustomAttributes API, 27-22
getDescription API, 27-22, 27-25
group spaces API documentation, 27-29
groupSpace parameter, 14-19
GroupSpaceReference data type, 28-11
GroupSpaceWSClient
getGroupSpaces API, 27-21
GroupSpaceWSClient class, 27-14, 27-17
addMember API, 27-18
createGroupSpace API, 27-16
createGroupSpaceTemplate API, 27-17
deleteGroupSpace, 27-18
getGroupSpaceMetadata, 27-22
getGroupSpaces API, 27-24
group spaces APIs
adding members, 27-18
creating group space attributes, 27-17
creating group spaces, 27-16
creating templates, 27-17
deleting group spaces, 27-18
inviting members, 27-19
removing members, 27-20
retrieving group space metadata, 27-22
retrieving list of public group spaces, 27-22
retrieving list of templates, 27-24
retrieving roles, 27-20
retrieving template metadata, 27-25
functionality in custom WebCenter applications, 27-4
getDescription API, 27-22, 27-25
getDisplayName API, 27-22, 27-25
getGroupSpaceState API, 27-23
getGuid API, 27-23, 27-25
getIconURL API, 27-23, 27-25
getKeywords API, 27-23, 27-25
getLastUpdated API, 27-23
getLogoURL API, 27-23, 27-25
getMailingList API, 27-23
getMembers API, 27-23
getName API, 27-23, 27-25
isDiscoverable API, 27-23
isPublic API, 27-23

guidelines
About mode, 31-27
CSS for portlets, 31-23
Edit Defaults mode, 31-25
Edit mode, 31-24
Help mode, 31-26
HTML for portlets, 31-22
portlet navigation, 31-27
Preview mode, 31-26
task flow creation, D-1

H
Help mode, 31-26
HelpPage.jsp file, A-12
hideActions binding parameter, 20-32
hideConfigure binding parameter, 20-33
hideFooter binding parameter, 20-33
hideHeader binding parameter, 20-33
HTML, guidelines for portlets, 31-22
htmlFormHiddenFields utility, 32-20
HTTP methods
REST APIs, 28-9
HTTP response status codes
REST APIs, 28-9
HTTPS
Web Clipping, F-18

I
Icon View display template, 14-15
IFRAME, 34-32
Image Link
definition, B-24
properties, B-24
images
enabling Profile photo upload, 20-16, 20-17
resource proxy, 31-29
imageSize binding parameter, 20-33
IMP (Instant Messaging and Presence) service
about, 18-1
adding to custom application, 18-15
Buddies task flow, 18-16
connections, 18-3
customizing views, 18-20
Presence component, 18-16
Presence Data component, 18-16
requirements, 18-2
security, 18-17
troubleshooting, 18-21
importing
custom display templates, 13-46
customizations for WSRP 2.0, 32-7
inline frames, 29-2
inline parameter, 21-6
input
escapes for PDK-Java, 37-28
Instant Messaging and Presence (IMP)
See IMP
Integrated WLS
deploying JSR 168 portlets, 33-3
deploying PDK-Java portlets, 33-4
determining if it is running, 3-24
starting, 3-24
stopping, 3-24
testing JSR 168 portlets, 33-3
testing PDK-Java portlets, 33-4
Integrated WLS Server
testing WebCenter applications, 38-2
Integrated WLS Server (DefaultServer), 38-1
internal links, 31-28
intraportlet links, 31-28
URL parameters, 32-16
inviteMember API, 27-19
isDiscoverable API, 27-23
ISO-8859-1 character set, 36-9
isPublic API, 27-23
items
reusing from Oracle Portal, G-10

J
J2EE security roles, mapping to WSRP
producer, 34-10
Java Community Process, 29-15
Java EE application server, 29-17
Java Object Cache, disabling for preference
store, 32-9
Java portlets, 29-10
authenticating to external applications, 29-29
authentication security manager, 29-28
caching style, 29-22
capturing content, 29-25
charting, 29-26
configuring, F-1
creating JSR 168 portlets, 31-5
creating PDK-Java portlets, 31-14
creation style, 29-23
design-time flexibility, 29-24
devlopment tools, 29-22
expertise required, 29-22
group security manager, 29-28
guidelines, 31-21
hiding and showing, 29-28
intended users, 29-10
multilingual support, 29-28
pagination support, 29-28
parameter support, 29-27
rendering inline, 29-26
usage suitability, 29-15
use case, 29-10
JavaScript
   Web Clipping, 36-8
JavaServer Faces, See JSF
javax.faces.DEFAULT_SUFFIX parameter, 3-9
JCR 1.0, 13-2
JCR adapters, 1-7, 13-2
JDBC data source
   application-level data source, 38-5
   global data source, 38-5
JDeveloper, about, 1-3
JNDI variables, 32-25
   declaring, 32-25
   naming conventions, 32-25
   persistentStore, F-21
   provided by PDK-Java, 32-28
   retrieving, 32-27
   setting values, 32-26
types, 32-25
JPS portlets files, A-2
   about.jsp, A-6
   config.jsp, A-6
   connections.xml, A-7
   edit_defaults.jsp, A-6
   edit.jsp, A-7
   faces-config.xml, A-16
   help.jsp, A-7
   JSPX file, A-16
   MDS subdirectory, A-18
   oracle-portlet-tags.jar, A-6
   oracle-portlet.xml, A-3
   oracle-portlet.xml sample, A-5
   oracle-portlet.xml syntax, A-4
   PageDef.xml, A-16
   PageDef.xml sample, A-16
   portlet_mode.jsp, A-6
   portlet_nameBundle.jar, A-7
   portlet_name.java, A-7
   portlet.xml, A-2
   portlet.xml sample, A-2
   preview.jsp, A-7
   print.jsp, A-7
   view.jsp, A-7
   web.xml, A-7, A-18
   wsdl subdirectory, A-19
JSF applications, creating portlets, 30-2
JSF portlets
   about, 29-4
   example, 30-15
   general guidelines, 30-10
   intended users, 29-4
   JSF guidelines, 30-12
   navigational parameters, 30-3, 30-6
   Oracle ADF guidelines, 30-12
   portlet entry for task flow, 30-7
   portlet guidelines, 30-11
   security guidelines, 30-11
testing, 30-8
use case, 29-5
JSF servlet settings, 3-9
jsf-impl.jar file, 3-6
JSP servlet, 3-9
JSR 168
   about, 31-2
   and WSRP, 31-2
JSR 168 portlets
   adding personalization, 32-2
   creating, 31-5
   deploying, 33-3
   guidelines, 31-21
   security, 32-10
   testing, 33-3

L

lastModifiedAfter parameter, 14-17, 14-19
lastModifiedBefore parameter, 14-17, 14-20
lastModifier parameter, 14-18, 14-19
Layout Customizable
   adding to a page, 7-7
   definition, 5-49
   layout types, B-10
   properties, B-6
   style selectors, B-48
lifecycle
   overview, 1-9
links
   bidirectional links, 22-3
   building with URL parameters, 32-18
   from objects, 22-2
   portlet, 31-27
   to objects, 22-2
   with other WebCenter services, 22-2
Links service, 22-3
   about, 22-1
   adding to custom applications, 22-7
   connection, 22-4
   example, 2-2
   Links Detail Button component, 22-1
   Links Detail Menu Item component, 22-2
   Links Status Icon component, 22-2
   RelationshipPermission, 22-8
   requirements, 22-3
   security, 22-8
   troubleshooting, 22-8
List View display template, 14-15
List with Details Panel View display template, 14-15

M

mail
   configuring the number of mails displayed, 19-13
   filters, 19-2
   recent messages, 19-2
Mail Compose page, 19-12
mail server, 19-3
Microsoft Exchange, 19-3
Mail service
  about, 19-1
  adding to a custom application, 19-10
  connection, 19-4
  customizing views, 19-12
  distribution lists, 19-3
  Mail Compose page, 19-12
  Mail task flow, 19-10
  requirements, 19-3
  security, 19-11
  troubleshooting, 19-13
Mail task flow, 19-10
managed server
  creating, 38-10
  provisioning, 38-10
maxDocuments parameter, 14-20
maxResults parameter, 14-13
MDS
  subdirectory, A-18
MDS (Metadata Storage)
  configurations, 10-1
mds-config, B-38
Message Board
  about, 20-4
  REST APIs, 20-54
  supported site-level settings, 20-23
  task flows, 20-13
  Message Board, 20-13
  Message Board - Quick View, 20-14
messages
  authentication for PDK-Java, 37-28
  encryption for PDK-Java, 37-25
mod_osso
  Web Clipping, 36-8
Model project, 3-4, 3-5
modes
  See portlet modes
mostRecentFirst parameter, 14-20
multilingual support, portlets, 29-28
multiple Resource Catalogs, 12-28

N
name portable contact type, 28-12
name tag, 13-42, 13-43
navigation
  link API, 32-18
  linking portlets, 32-21, 34-38
  parameters for WSRP 2.0, 32-4
  within a Java portlet, 31-27
  within JSF portlets, 30-3
  within JSF portlets, 30-6

O
OCS Global Profile
  node properties, 13-47
OmniPortlet, 3-25
  about, 29-9, 35-1
  adding to a page, 35-3
AllModesSharedScreen, 34-19
  authenticating to external applications, 29-29
  caching style, 29-21
  capturing content, 29-25
  charting, 29-26
  configuring, F-2
  configuring to access data outside a firewall, F-2
  configuring to access relational databases, F-5
  creation style, 29-23
  customizing, 35-3
  data sources, 29-16
  design-time flexibility, 29-24
  development tools, 29-22
  expertise required, 29-16
  hiding and showing, 29-28
  installing DataDirect JDBC drivers, F-5
  intended users, 29-9
  multilingual support, 29-28
  pagination support, 29-28
  parameter support, 29-27
  provider.xml file, F-6
  registering DataDirect JDBC drivers, F-6
  registering the producer, 35-2
  rendering inline, 29-26
  troubleshooting, 35-4
  usage suitability, 29-15
  use case, 29-9
Oracle Composer
  about, 1-5, 5-1
  adding Box components, 5-12
  adding content, 5-9
  adding Documents service, 14-3
  adding Documents service task flows, 14-8
  add-ons
    about, 8-2
    creating, 8-4
    excluding, 8-7
    rendering selectively, 8-10
    adf-config.xml configurations, B-33
    applying styles to components, 9-21
    Change Mode Button component, 5-48, B-4
    Change Mode Link component, 5-48, B-4
    changing layouts, 5-28
    components, 5-46
    concurrency behavior, 10-6
    configuration files, 8-3
    creating labels to save customizations, 5-32
    Custom Action component, B-21, B-22
    custom property panels
      about, 8-2
      creating, 8-10
      excluding, 8-14
      overriding, 8-14
      registering, 8-11
    customization, 5-2
    Customization Manager, 5-33
    customizing the toolbar, 8-32
    cut and paste options, 5-40
    default add-ons, B-43
    default configurations, 7-11
default property panels, B-43
deleting components, 5-28
design considerations, 7-12
design time at runtime, 5-1
Design view, 5-7
designing editable pages, 7-1
design-time components, 5-46
disabling, B-6
disabling Source view, 9-20
displaying source view by default, 9-19
drop handlers, 8-3, 8-22
creating, 8-22
registering, 8-25
displaying source view, 5-37
disabling Source view, 5-45
Edit mode capabilities, 5-7
editing component properties, 5-17
editing page properties, 5-29
editing resource strings, 5-21
editing task flow content, 5-41
enabling runtime customizations, 7-3
error handling, 5-32
events, 8-2
configuring, 8-17
registering, 8-20
events, supported, 8-18
example, 2-13
extending framework, 8-1
extending the expression builder, 8-15
files, B-26
global style selectors, B-47
hiding page components
Design view, 5-26
Source view, 5-45
Image Link component, B-24
Layout Customizable component, 5-49, B-6
managing customizations, 5-33
MDS configurations, 10-1
adding customization layers, 10-8
MDS customization restrictions, 5-54
overriding component customizations, 5-20
overriding default security behavior, 11-1
Page Customizable component, 5-47, B-2
page privileges mapping, 5-52
page-level security, 5-51
Panel Customizable component, 5-48, B-10
pe_ext.xml, B-27
persistance change manager
about, 9-23
configuring, 9-25
personalization, 5-2
property filters, 8-3
defining, 8-28
removing, 8-29
rarranging components
using cut and paste options, 5-40
rarranging content, 5-16
resetting component customizations, 5-20
resetting customizations, 5-32
resetting page customizations, 5-30
resetting task flow customizations, 5-43
resource string editor
about, 9-27
configuring, 9-27
sandbox, 10-2
destroying stale sandboxes, 10-7
disabling, 10-7
enabling, 10-3
save and label, 5-32
saving customizations, 10-5
security
action-level restrictions, 11-16
attribute-level restrictions, 11-14
component-level restrictions, 11-1
customization policies, 11-1
task flow-level restrictions, 11-21
security considerations, 5-50
security, customizationAllowed attribute, 11-13
Show Detail Frame component, 5-49, B-14
showing page components
Design view, 5-26
Source view, 5-45
Source view, 5-37
task flow-level security, 5-52
View mode capabilities, 5-3
web.xml configurations, B-42
Oracle Discoverer, charting, 29-26
Oracle JSF Portlet Bridge, 30-1
authenticating to external applications, 29-29
caching style, 29-21
capturing content, 29-24
charting, 29-26
design-time flexibility, 29-24
development tool, 29-22
design-time flexibility, 29-24
development tool, 29-22
eample, 2-6, 30-15
expertise required, 29-16
general guidelines, 30-10
general suitability, 29-14
hiding and showing, 29-28
JSF portlets, 29-4
Oracle ADF guidelines, 30-12
pagination support, 29-28
parameter support, 29-27
portlet creation style, 29-23
portlet guidelines, 30-11
rendering inline, 29-25
security guidelines, 30-11
testing portlets, 30-8
See also JSF portlets, 30-1
Oracle PDK-Java
creating struts portlets, 32-35
Struts integration, 32-35
Oracle PDK-Java portlets
public parameters, 32-12
session storage
checking for valid session, 32-31
enabling in provider.xml, 32-31
implementing, 32-29
Oracle Portal
FPA (Federated Portal Adapter), G-4
reusing components in Oracle WebCenter, G-1
reusing items, G-10
reusing portlets, G-2
Oracle Portal pages
Web Clipping limitation, 36-8
Oracle Reports, charting, 29-26
Oracle Secure Enterprise Search (SES), 24-8, E-1
connection, 24-8
data control, E-1
using with WebCenter, 24-8
Oracle SES
See Oracle Secure Enterprise Search (SES)
Oracle WebCenter
about, 1-1
Oracle WebCenter Framework
portlet technologies, 29-11
Struts integration, 32-35
WSRP support, 31-3
Oracle WebCenter Spaces
about, 27-1
adding portlets, 27-4
API documentation, 27-29
APIs, 27-4
registering producers, 27-4
Oracle WebCenter Suite
components in the Fusion Order Demo, 2-2
Oracle WebCenter Wiki and Blog Server
adding wikis and blogs, 21-3
adding wikis or blogs by using a portlet, 21-3
adding wikis or blogs by using IFRAME, 21-4
adding wikis or blogs by using Web Services, 21-5
sample portlets, 21-3
URL formats for exposing wikis and blogs, 21-5
Web Services client stub generation, 21-14
Web Services example, 21-15
Web Services interface, 21-8
Web Services methods, 21-8
Web Services security, 21-13
Oracle WebLogic Server
Integrated WLS Server, 38-1
managed servers, 38-2
oracle-portlet.xml file
JSF portlets, 30-4, 30-7
syntax, A-3
organization portable contact type, 28-12
overview
security, 37-1
WSRP producers security, 37-21
adding to custom applications, 6-3
APIs, 6-1, 6-22
using, 6-37
data control, 6-1
using, 6-18
Page - Create New task flow, 6-2
samples, 6-41
scope, 6-1
security, 6-4
page variables
deleting for portlets, 34-36
PAGE_LINK URL, 32-18
pages
adding portlets, 34-16
creating, 3-12, 6-1
creating portlets, 30-2
Edit mode, 5-3
managing, 6-18
portlet entry, 30-3
requirements for WebCenter applications, 3-12
schemes, 6-2
styles, 6-2
custom, 6-22
default, 6-22
templates, 6-22
creating, 6-23
View mode, 5-3
pages files, A-14
adf-config.xml, A-14
adf-portlet-config sample, A-15
DataBindings.cpx, A-16
pagination support, portlets, 29-28
Panel Customizable
adding to a page, 7-6
applying action-level restrictions, 11-16
definition, 5-48
disabling Edit option, B-14
properties, B-10
style selectors, B-48
Parameter Form portlet
about, 29-6
intended users, 29-7
use case, 29-7
parameters
adding for WSRP 2.0, 32-4
building links with URL types, 32-18
context parameters, 3-9
Documents service task flows, 14-11
in Oracle PDK-Java portlets, 32-12
in PDK-Java portlets, 32-16
javax.faces.DEFAULT_SUFFIX, 3-9
navigational for WSRP 2.0, 32-4
portlets, 29-26, 29-27
private portlet parameters, 29-27
public portlet parameters, 29-27
URL, 32-16
partner portlets, 29-6
PDK
Preference Store Migration/Upgrade Utility, F-8,
F-22

PDK-Java
about, 31-3
adding portlet modes, 32-10
JNDI variables, 32-28
portlets
building links with URL types, 32-18
caching, 32-32 to 32-35
creating, 31-14
guidelines, 31-21
session storage, 32-29 to 32-32
security, 37-24
PDK-Java portlet files, A-11
_default.properties, A-12
connections.xml, A-14
index.jsp, A-12
portlet_nameAboutPage.jsp, A-12
portlet_nameEditDefaultsPage.jsp, A-12
portlet_nameEditPage.jsp, A-12
portlet_nameHelpPage.jsp, A-12
portlet_nameShowDetailsPage.jsp, A-12
portlet_nameShowPage.jsp, A-12
producer_name.properties, A-12
provider.xml file, A-12
provider.xml sample, A-13
provider.xml syntax, A-13
web.xml, A-14
PDK-Java portlets
private parameters, 32-16
PDK-Java producers, 3-27, 29-17
deleting portlets, 34-36
deregistering, 34-16
editing registration values, 34-15
enabling producer sessions, 34-13
external applications, 34-13
refreshing, 34-16
registering, 34-11
service ID, 34-13
syntax of URL endpoint, 34-12
testing connection, 34-15
pe_ext.xml, 8-4, B-27
People Connections service, 20-1 to 20-57
about, 20-1 to 20-5
Activity Stream, 20-2, 20-3
REST APIs, 20-36
supported site-level settings, 20-20
configuring, 20-5
Connections, 20-4
example site-level settings, 20-27
REST APIs, 20-40
supported site-level settings, 20-22
database set-up, 20-5
Feedback
about, 20-4
REST APIs, 20-51
supported site-level settings, 20-23
Message Board
about, 20-4
REST APIs, 20-54
supported site-level settings, 20-23
Profile
about, 20-4
field names, 20-26
REST APIs, 20-40
section names, 20-26
supported site-level settings, 20-24
requirements, 20-4
REST APIs, 20-35 to 20-57, 28-20, 28-24
security, 20-17
site-level settings
preparing and running
settings-insert.sql, 20-18
preparing and running
settings-update.sql, 20-19
scripts, 20-18
supported settings, 20-20
task flow binding parameters, 20-28
connectionListName, 20-31
display_maxConnections, 20-31
display_numberOfColumns, 20-31
display_numberOfRows, 20-31
display_profileFormat, 20-31
display_removalAllowed, 20-32
display_sortedBy, 20-32
displayCount, 20-32
displayMessageSize, 20-32
filterPattern, 20-32
fromDate, 20-32
hideActions, 20-32
hideConfigure, 20-33
hideFooter, 20-33
hideHeader, 20-33
imageSize, 20-33
photoUploadAllowed, 20-33
resourceld, 20-34
revising, 20-29
showConnectionsActivities, 20-34
showSummarizedView, 20-34
showUserActivities, 20-34
statusUpdateAllowed, 20-35
toDate, 20-35
userId, 20-35
task flows, 20-7 to 20-17
Activity Stream, 20-8
Connections, 20-9
Feedback, 20-11
Message Board, 20-13
Profile, 20-14
performance
caching for PDK-Java portlets, 32-32
permissions
RelationshipPermission, 22-8
persit-changes, 11-15
persistence change manager
about, 9-23
configuring, 9-25
personalization, 5-2
adding to JSR 168 portlets, 32-2
testing for portlets, 32-35
personReference data type, 28-11
Index
navigation, 31-27
Preview mode, 31-26
headers, displaying, 34-26
hiding and showing, 29-28
icon, 34-24
IFRAME, 34-32
implementing internal navigation, 32-21
in custom WebCenter applications, 34-1
inline content rendering, 29-13
inline frames, 29-2
Java portlets, 29-10
JSF portlets, 29-4, 30-1
JSP files for PDK-Java, A-12
linking, 34-37
example for cascading events, 34-44
example with ADFm events, 34-42
example with navigational parameters, 34-39
links
applications, 31-28
external, 31-28
internal, 31-28
intraportlet, 31-28
resource, 31-28
types, 31-27
maximized display, 34-25
Minimize icon, 34-27
minimized display, 34-25
modes
See portlet modes
Move command, 34-27
multilingual support, 29-13, 29-28
navigation link API, 32-18
normal display, 34-25
OmniPortlet, 3-25, 29-9
Oracle JSF Portlet Bridge, 30-1
overriding inherited security, 37-16
pagination support, 29-13, 29-28
Parameter Form portlet, 29-6
PDK-Java security, 37-24
personalization, 29-3, 32-35
portlet entry for page, 30-3
portlet parameters
private, 29-13, 29-27, 32-16
public, 29-13, 29-26, 32-12
prebuilt, 3-23, 29-6
preconfigured server and, 3-25
producers, 29-17
programmatic, 29-11
refreshing portlet producers, 34-16
registering portlet producers, 34-2
Remove icon, 34-27
rendering inline, 29-25
resize handle, 34-27
resource bundle, A-7
reusing from Oracle Portal, G-2
Rich Text portlet, 3-26, 29-5
running, 34-20
sample portlets, 3-26, 3-27
scroll bar, displaying, 34-26
security, 29-3, 29-13, 37-16
security at actions category level, 37-16
security at actions level, 37-17
security managers for PDK-Java, 37-27
shadow, displaying, 34-26
skin style selectors, B-49
style, 34-30
supported data sources, 29-12
testing, 33-1
title, setting, 34-23
URL types, 32-18
WAR deployment profile, 33-8
Web Clipping, 29-7
Web Clipping portlet, 3-25
Web content capture, 29-13
prebuilt portlets, 29-6
Preconfigured Server
index.html file, 3-27
preference store
database (WSRP producer), F-21
disabling Java Object Cache, 32-9
migration utility, F-23
WSRP container, F-23
Preference Store Migration/Upgrade Utility, F-8, F-22
preferenceStore tag
class attribute, F-22
connection parameter, F-22
name parameter, F-22
Presence component, 18-16
presence server
LCS, 18-2
OWLCS, 18-2
Preview mode, 31-26
guidelines, 31-26
printStackTrace API, 27-29
private parameters
in PDK-Java portlets, 32-16
restricting navigation, 32-24
producers
name clashes, 39-15
PDK-Java, 3-27
preconfigured, 3-23
preconfigured server and, 3-25
registering with WebCenter Spaces, 27-4
WSRP producer’s database preference store
settings, F-21
WSRP producers, 29-19
Profile
about, 20-4
enabling photo upload, 20-16, 20-17
field names, 20-26
REST APIs, 20-40
section names, 20-26
supported site-level settings, 20-24
task flows, 20-14
Profile, 20-14
Profile - Snapshot, 20-15
Profile Gallery, 20-16
projection parameter (REST APIs), 28-8
projects
data controls, 3-5
Model, 3-4, 3-5
pages, 3-5
portlet producers, 3-5
ViewController, 3-4, 3-5

properties
content items, 13-46
content type, 13-36

property filters
about, 8-3
defining, 8-28
removing, 8-29

property-panel, B-28
property-panels, B-28

providers
See portlet producers

provider.xml file
OmniPortlet, F-6
session storage, 32-31
Web Clipping, F-11

proxy
resource, 31-29
resource proxy for WSRP, 32-8
stateless resource proxying, 32-9
Web Clipping, F-17

public parameters
in Oracle PDK-Java portlets, 32-12

public welcome page, 37-11

Q
q parameter (REST APIs), 28-8
query string parameters, 21-5, 21-6

R
rcv-config, B-35

Recent Activities service
about, 25-2
adding, 25-3
compared to Activity Stream, 20-3
connections, 25-3
editing task flow parameters, 25-7
element, 2-6
security, 25-7
task flow, 25-3

Recent Documents task flow
adding to page, 14-3
parameters, 14-18

registering
JPS portlet, 33-11
PDK-Java portlet, 33-11
portlet producers, 34-2
resource viewers, 4-10
rel attribute, 28-6
removeMember API, 27-20
renderProperty tag, 13-34
repositoryInfo tag, F-15
class, F-16
connectionName, F-16
databaseHost, F-16
databasePassword, F-17
databasePort, F-17
databaseSid, F-17
databaseUsername, F-17
mdsConfigLocation, F-16
useASO, F-16
useRAA, F-16

resource, C-10
bundle file for JPS portlets, A-7
Resource Action Handling framework, 23-13
about, 4-9
custom components, 4-9
resource viewers, 4-10

Resource Catalog
about, 1-7, 12-1
adding XML content, 12-16
configuring multiple catalogs, 12-28
contents, 12-3
creating a custom catalog, 12-12
default catalog definition, C-2
enabling task flows, 12-7
element, 2-13
example, 12-18
example for configuring multiple catalogs, 12-29
example for creating a custom catalog, 12-13
example for filtering items, 12-25
example for modifying default catalog, 12-5
filtering items, 12-25
location, C-1
modifying contents, 12-4
ResourceCatalogSelector, 12-28

Resource Index, 28-3
resource links, 31-28

Resource Palette
adding Documents service task flow, 14-6
resource proxy, 31-29
for WSRP, 32-8
stateless resource proxying, 32-9
resource string editor
about, 9-27
configuring, 9-27
resource viewer, 4-10, 23-13
ResourceCatalogSelector, 12-28
resourceld binding parameter, 20-34
resourceType attribute, 28-6

REST APIs
about, 28-1
capabilities attribute, 28-6
elements, 28-13, 28-18, 28-20, 28-24
Ext, 28-20
HTTP methods, 28-9
HTTP response status codes, 28-9
hypermedia links, 28-4
hypermedia, navigating, 28-8, 28-13
link model, 28-2

People Connections service, 20-35 to 20-57
Activity Stream, 20-36
Connections, 20-40
Feedback, 20-51
Message Board, 20-54
Profile, 20-40
projection parameter, 28-8
q parameter, 28-8
query parameters, 28-8
rel attribute, 28-6
Resource Index, 28-3
resourceType attribute, 28-6
security, 28-10
stoken parameter, 28-10
template attribute, 28-7
type attribute, 28-7
Rich Text portlet, 3-26, 29-5
intended users, 29-6
use case, 29-6
rootFolderPath parameter, 14-18
RSS
about, 15-1
RSS service
about, 15-1
adding task flows, 15-4
connections, 15-3
security, 15-5
task flows, 15-4
RSS Viewer service
example, 2-6
runtime customizations, clearing, 33-1
runtime persistence, turning on and off, 34-31

S
sample application
about the Fusion Order Demo, 2-1
Documents service, Discussions service, Links
service, Tags service, 2-2
features, 2-15
Oracle Composer, 2-13
Oracle WebCenter Fusion Order Demo components, 2-2
Resource Catalog, 2-13
Search service, Recent Activities service, RSS viewer service, Tags service, Oracle JSF Portlet Bridge, 2-6
sandbox
about, 10-2
concurrency behavior, 10-6
destroying stale sandboxes, 10-7
disabling, 10-7
enabling, 10-3
saving customizations, 10-5
sandbox-namespaces, B-35
scope parameter, 21-7
scopes
about, 6-1
scripts
People Connections service
settings-insert.sql, 20-18
settings-update.sql, 20-18
search
implementing as a Web service, E-1
Search - Saved Searches task flow, 24-4
search adapters, 24-11
attributes, 24-17
registering, 24-17
Search Preferences task flow, 24-4
Search service
about, 24-1
adapters, 24-11
adding a search source, 24-11
adding to custom applications, 24-3
APIs, 24-11
connection, 24-3
example, 2-6
including Oracle SES results, 24-8
Search - Saved Searches task flow, 24-4
Search Preferences task flow, 24-4
Search task flow, 24-4
Search Toolbar task flow, 24-4
security, 24-5
troubleshooting, 24-20
with documents, 24-1
with tags, 24-1
Search task flow, 24-4
Search Toolbar task flow, 24-4
security
adding test users, 20-17
and PDK-Java, 37-24
authentication for PDK-Java, 37-25
basic configuration for WebCenter applications, 3-15
communication for PDK-Java, 37-25
configuring basic authentication, 37-11
creating public welcome page, 37-11
customizable components, 37-25
escaping user inputs for PDK-Java, 37-28
external applications, 37-6
features for PDK-Java, 37-25
JSR 168 portlets, 32-10
managers for PDK-Java, 37-27
message authentication for PDK-Java, 37-28
Oracle Composer, 5-50
overview, 1-8, 37-1
People Connections service and, 20-17
portlets, 34-20, 37-14
registering custom certificates, 37-12
securing Documents service, 14-20
securing Recent Activities service, 25-7
securing RSS service, 15-5
security Oracle WebCenter Wiki and Blog Server, 21-13
WebCenter Customizable Components, 5-50
WSRP producers, 37-21
security files, A-19
jazn-data.xml, A-19
jazn-data.xml sample, A-19
service IDs, 20-21
service-definition.xml file, 4-10, 4-14
services
about Worklist, 26-1
See WebCenter Services, 4-9
servlets
JSF servlet settings, 3-9
servlet mapping, 3-9
session storage
about, 32-29
checking for valid session, 32-31
enabling in provider.xml, 32-31
implementing in Oracle PDK-Java portlets, 32-29
session-options-factory, B-35
setCustomAttribute API, 27-17
setGroupSpacesWebServiceEndpoint API, 27-11
settings-insert.sql script, 20-18
settings-update.sql script, 20-18
Shared Screen mode (PDK-Java), 31-22
Show Detail Frame
action-level security, 5-54
adding to a page, 7-9
additional actions facet example, 9-2
applying action-level restrictions, 11-16
custom actions example, 9-12
definition, 5-49
disabling Edit option, B-21
enabling custom actions, 9-7
facets, 9-1
properties, B-14
style selectors, B-49
show modes
See portlet modes
showConnectionsActivities binding parameter, 20-34
showDetailsPage.jsp file, A-12
showFolders parameter, 14-18
showPage.jsp file, A-12
showSummarizedView binding parameter, 20-34
showUserActivities binding parameter, 20-34
Simple Object Access Protocol (SOAP), 29-17
single sign on
Web Clipping, 29-8
single sign-on
external application for PDK-Java, 37-26
single sign-on PDK-Java, 37-26
site administrator
design considerations, 1-13
skins
global style selectors, B-47
Layout Customizable style selectors, B-48
Page Customizable style selectors, B-47
Panel Customizable style selectors, B-48
portlet style selectors, B-49
Show Detail Frame style selectors, B-49
style attributes, B-51
SOAP, 29-17
social networking, 20-1 to 20-57
linking and, 22-3
Sortable Table View display template, 14-15
source control systems
CVS, 39-1
Subversion, 39-1
standalone Oracle WebLogic Server
deploying portlet applications, 33-9
testing portlet applications, 33-9
startFolderPath parameter, 14-16
stateless resource proxying, 32-9
statusUpdateAllowed binding parameter, 20-35
stoken parameter (REST APIs), 28-10
Struts portlets
creating, 32-35
Oracle WebCenter Framework integration, 32-35
Subversion
using, 39-5
T
Tabbed View display template, 14-15
Tag Center
about, 23-1
tag clouds
about, 23-2
Tagging - Dialog task flow, 23-6
Tagging - Personal View task flow, 23-6
Tagging - Related Links task flow, 23-6
Tagging - Tag Cloud task flow, 23-6
Tagging - Tagged Items task flow, 23-6
tags
exposing resources to be tagged, 23-13
resource viewer, 23-13
searching, 24-2
Tags service
about, 23-1
adding to custom applications, 23-5
connection, 23-4
example, 2-2, 2-6
requirements, 23-2, 23-3
Resource Action Handling framework, 23-13
searching tags, 24-2
security, 23-9
Tagging - Dialog task flow, 23-6
Tagging - Personal View task flow, 23-6
Tagging - Related Links task flow, 23-6
Tagging - Tag Cloud task flow, 23-6
Tagging - Tagged Items task flow, 23-6
Tagging Button component, 23-6
Tagging Menu Item component, 23-6
using Tag Cloud and Tagged Items together, 23-14
with a page template, 23-9
task flow parameters
Documents service, 14-11
task flows
Activity Stream, 20-8
adding Documents service, 14-3
adding Recent Activities, 25-3
adding RSS service, 15-4
Announcements service
Announcements - Sidebar View task flow, 16-9
Announcements task flow, 16-8
Connections, 20-9
creating, 6-1
creating portlets, 30-5
Index-17
Discussions service
  Discussion Forums task flow, 17-7
  Discussions - Popular Topics task flow, 17-7
  Discussions - Recent Topics task flow, 17-8
  Discussions - Sidebar View task flow, 17-8
  Discussions - Watched Forums task flow, 17-8
  Discussions - Watched Topics task flow, 17-8
  editing content in Oracle Composer, 5-41
  editing Recent Activities parameters, 25-7
  enabling custom actions, 9-14
Feedback, 20-11
  files affected, 39-11
  guidelines for creating, D-1
IMP service
  Buddies task flow, 18-16
Mail service
  Mail task flow, 19-10
  Message Board, 20-13
Page service
  Page - Create New task flow, 6-2
People Connections service
  Activity Stream, 20-8
  Activity Stream - Quick View, 20-8
  adding to a page, 20-16
  Connections, 20-9
  Connections - Card, 20-10
  Connections - Quick View, 20-11
  Feedback, 20-11
  Feedback - Quick View, 20-12
  Message Board, 20-13
  Message Board - Quick View, 20-14
  Profile, 20-14
  Profile - Snapshot, 20-15
  Profile Gallery, 20-16
portlet entry, 30-7
Profile, 20-14
Recent Activities, 25-3
RSS service, 15-4
Search service
  Search - Saved Searches task flow, 24-4
  Search Preferences task flow, 24-4
  Search task flow, 24-4
  Search Toolbar task flow, 24-4
Tags service
  Tagging - Dialog task flow, 23-6
  Tagging - Personal View task flow, 23-6
  Tagging - Related Links task flow, 23-6
  Tagging - Tag Cloud task flow, 23-6
  Tagging - Tagged Items task flow, 23-6
taskFlowInstId parameter, 14-14, 14-18
    teams
      developer actions affecting metadata files, 39-7
      developing in, 39-1
      implementing common requirements, 39-14
      managing files, 39-7
      template attribute, 28-7
      templateCategoryId parameter, 14-14
      template-def tag, 13-42, 13-43
    templates
      Content Presenter, 14-12
Content Presenter task flow, 13-3
  see also Content Presenter task flow, 13-28
  templates registry
    Content Presenter task flow, 13-3, 13-41
    tags for multiple content item templates, 13-42
    tags for single content item templates, 13-41
    templates-registry.xml, 13-3, 13-41
    templateViewId parameter, 14-14
themes
  wiki themes, 21-7
toDate binding parameter, 20-35
type attribute, 28-7

U

UPLOAD_MAX_DISK_SPACE, 14-23
UPLOAD_MAX_MEMORY, 14-23
UPLOAD_TEMP_DIR, 14-23
uploading
  setting parameters, 14-23
URLs
  in forms, 32-19
  parameters, 32-16
  rewriting for WSRP resource proxy, 32-8
  stateless resource proxying, 32-9
  UrlUtils.constructLink API, 32-18
  use case examples
    additional actions facet on Show Detail Frame, 9-2
    enabling custom actions on a task flow, 9-12
    user
      adding test users, 20-17
      design considerations, 1-12
      input escapes for PDK-Java, 37-28
      session, 32-29
      userId binding parameter, 20-35
usesUpload, 14-24
usesUpload attribute on form, 20-16, 20-17
UTF-8 character sets
  PDK-Java portlets, 31-16
utility
  Preference Store Migration/Upgrade, F-8, F-22

V

validation-based caching, 29-21
value portable contact type, 28-13
Versioning Navigator, 39-1
View mode, 5-3
View mode (JSR 168), 31-22
view tag, 13-42, 13-43
ViewController project, 3-4, 3-5

W

WAR files, 3-23
Web archive, See WAR
Web Clipping
  about, 29-7, 36-1
  adding certificates for trusted sites, F-18
  adding to a page, 36-3
advanced features, 36-6
authenticating to external applications, 29-29
authentication, 29-8
browser support, 36-9
caching style, 29-21
capturing content, 29-24
charting, 29-26
configuring Oracle Advanced Security Option (ASO), F-19
configuring proxy settings, F-17
configuring repository, F-10
configuring security, F-18
creation style, 29-23
design time flexibility, 29-24
development tools, 29-22
expertise required, 29-16
features, 36-1
fuzzy matching, 29-8
hiding and showing, 29-28
inline rendering, 29-8
intended users, 29-8
limitations, 36-8
multilingual support, 29-28
navigation, 29-8
open transport API, 29-8
pagination support, 29-28
parameter support, 29-27
personalization, 29-8
proxy authentication, 29-8
rendering inline, 29-25
resource tunneling, 29-8
restricting clipping from unauthorized external web sites, F-18
reusing Web content, 29-8
rewriting image links, 36-8
single sign on, 29-8
URL rewriting, 29-25
usage suitability, 29-15
use case, 29-8
using cascading style sheets (CSS), 36-8
Web Clipping Transport API, 36-7
Web Clipping portlet, 3-25
Web Clipping repository configuring, F-10, F-14
creating database schema, F-13
JNDI data source, F-13
provider.xml path, F-11
repositoryInfo tag, F-15
using MDS, F-12
using Oracle database, F-13
Web Clipping Transport API using, 36-7
Web service search, E-1
Web Services for Remote Portlets See WSRP
Web Services methods
domain-related methods, 21-9
WebCenter applications, 38-1
ADF applications and, 3-21
application-level deployment profile, 38-9
creating, 3-2
creating application manually, 3-21
creating pages, 3-12
creating through WAR files, 3-23
deploying, 38-1
deploying to standalone WebLogic Server, 38-11
Deployment profiles, 38-8
Enterprise Archive (EAR) file, 38-8
folder hierarchy, 3-6
implementing security, 3-15
mobile support, 3-29
Model project, 3-4, 3-5
project-level deployment profile, 38-9
security, 37-1
technology scopes, 3-4
testing, 38-1
ViewController project, 3-4, 3-5
Web Application Archive (WAR) file, 38-8
WebCenter Customizable Components
security considerations, 5-50
WebCenter Extension, about, 1-3
WebCenter extension, installing, 3-1
WebCenter Framework
about, 1-4
JSR 168, 29-12
WSRP, 29-12
WebCenter Preconfigured Server
readme file, 3-25
WebCenter Services
about, 1-10
Announcements service, 16-1
Discussions service, 17-1
Documents service, 14-1
testing, 38-1
examples in the sample application, 2-15
files, A-22
adf-config.xml for the Documents service, A-22
adf-config.xml for the Worklist service, A-23
SES Web service connection example, A-22
IMP service, 18-1
integration, 1-11
Links service, 22-1
Mail service, 19-1
People Connections service, 20-1 to 20-57
about, 20-1 to 20-5
Activity Stream, 20-2, 20-3, 20-8
configuring, 20-5
Connections, 20-4, 20-9
database set-up, 20-5
Feedback, 20-4, 20-11
Message Board, 20-4, 20-13
Profile, 20-4, 20-14
requirements, 20-4
task flows, 20-7 to 20-17
Recent Activities service, 25-2
RSS service, 15-1
Search service, 24-1
service IDs, 20-21
Tags service, 23-1
Wiki and Blog services, 21-3
Worklist service, 26-1
WebCenter Spaces
  about, 27-1
  adding portlets, 27-4
  API documentation, 27-29
  APIs, 27-4
  consuming information from custom applications, 27-46
  registering producers, 27-4
WebCenterComposerFilter, 10-14, B-43
WebLogic Managed Server
  creating, 38-10
  provisioning, 38-10
WebLogic Server connection, 38-11
web.xml, B-42
web.xml file, 3-6, 3-7, 3-9
wikis
  adding, 21-3
  adding by using a portlet, 21-3
  adding by using IFRAME, 21-4
  adding by using Web Services, 21-5
  sample portlets, 21-3
  URL formats for exposing wikis, 21-5
Web Services client stub generation, 21-14
Web Services example, 21-15
Web Services interface, 21-8
Web Services methods, 21-8
Web Services security, 21-13
window.location, iframes, 34-32
Worklist service
  about, 26-1
WSDL
  subdirectory, A-19
WSRP, 29-12
  about, 31-1
  and JSR 168, 31-2
  architecture, 31-2
  navigational parameters (WSRP 2.0), 32-4, 34-38
  Oracle WebCenter Framework architecture, 31-3
WSRP producers, 29-18, 29-19
  database preference store settings, F-21
  deleting portlets, 34-36
  deregistering, 34-16
  editing registration values, 34-15
  external applications, 34-8
  linking portlets, 34-37
  mapping J2EE security roles, 34-10
  navigational parameters, 34-38
  producer test page, 34-5
  refreshing, 34-16
  registering, 34-3
  security, 37-21
  security attribute configuration, 34-6
  supported versions, 34-3
  syntax for WSDL URL, 34-4
  testing connection, 34-15

Index-20