Oracle® WebCenter Content

Deployment Guide for Content Portlet Suite

11*g* Release 1 (11.1.1)

E10637-02

November 2011

Oracle WebCenter Content Deployment Guide for Content Portlet Suite, 11*g* Release 1 (11.1.1)

E10637-02

Primary Author: Will Harris

Contributors: Anand Vaidyanathan, Chandrasekhar Atla, Tarun Sharma , Mridhula Shetty, Naresh Sivaraman, Ron van de Crommert, Sean Cearley

Contents

Preface

Oracle WebCenter Content Deployment Guide for Content Portlet Suite (CPS) provides an overview of the CPS architecture as well as information on using the CPS Software Development Kit (SDK).

Audience

This document is intended for application developers and programmers. It offers an overview of the portlets, a presentation of their framework and architecture, and information on using the CPS Software Development Kit (SDK).

Documentation Accessibility

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

Access to Oracle Support

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

Related Documents

For more information, see the following documents in the Oracle WebCenter Content documentation set:

- Remote Intradoc Client (RIDC) chapter of the *Oracle WebCenter Content Developer's Guide for Content Server*
- *Oracle WebCenter Content Remote Intradoc Client (RIDC) Java API Reference*

Conventions

The following text conventions are used in this document:

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

1 General Information

This chapter provides general information on Content Portlet Suite (CPS) including a list of supported servers, descriptions of the portlets, recommended access settings, and information on the portlet interface language. It covers the following topics:

- Section 1.1, "What's New"
- Section 1.2, "Servers Supported in this Release"
- Section 1.3, "Portlets Provided with this Release"
- Section 1.4, "Recommended Access Settings"
- Section 1.5, "Portlet Interface Language"

1.1 What's New

CPS 11*g* Release 1 (11.1.1) includes the following new features and enhancements:

Additional WSRP Support

IBM Websphere Portal 6.1 is now supported as both a WSRP Producer and WSRP Consumer. See Section 3.3, "IBM WebSphere Portal as Producer" for more information.

CPS Architecture

CPS now uses the Remote Intradoc Client (RIDC) API to communicate with Oracle WebCenter Content Server. CPS portlet *actions* are mapped to a custom MVC framework that uses the RIDC API to perform tasks. This architecture enables CPS portlets to call RIDC methods and execute the requested content server service. See Appendix A, "CPS Software Development Kit" for more information.

1.2 Servers Supported in this Release

This version of CPS provides Web Services for Remote Portlets (WSRP) support. CPS is deployed as a WSRP producer and can be consumed from other WSRP-enabled consumers.

Supported WSRP Producers and Consumers

Oracle WebCenter Portal 11*g* Release 1 (11.1.1.3 or higher) (as *producer*)

- Oracle WebCenter Spaces 11*g* Release 1 (supported *consumer*)

Oracle WebLogic Portal 10*g* Release 3 (as *producer*)

- Oracle WebCenter Interaction 10*g* Release 3 (supported *consumer*)
- Oracle WebLogic Portal 10*g* Release 3 (supported *consumer*)

IBM WebSphere Portal Version 6.1 (as *producer*)

- IBM WebSphere Portal Version 6.1 (supported *consumer*)

1.3 Portlets Provided with this Release

CPS enables you to manage content creation and the distribution process through a set of easy-to-use portlets. By providing access to Oracle WebCenter Content Server, CPS enables users to update, search, and view content in an easy, efficient way.

These are the available portlets for this release:

- **Oracle Guest Library**: Presents content to users based on their role in the organization.
- **Oracle Guest Search**: Enables the user to perform a keyword or full-text search on the content server and permits read-only access to the returned content.
- **Oracle Saved Search**: Enables the user to save frequently used queries.
- **Oracle Contribution**: Enables the user to submit content to the content server.
- **Oracle Library**: Presents content to users based on their role in the organization, and provides read/write access to the returned content.
- **Oracle Search**: Enables the user to perform a selected metadata and keyword search on the content server and provides read/write access to the returned content.
- **Oracle Workflow Queue**: Notifies users of their outstanding workflow tasks.
- **Oracle Metadata Admin**: Enables the administrator to modify the properties of custom metadata.

1.4 Recommended Access Settings

This section provides the recommended access settings for the portlets. By default, portlet security is set to allow access by only the system administrator. You must also configure security to allow other users to access the portlets and assign permissions based on the access requirements you establish for users. However, some security settings are recommended for authenticated users and for anonymous users.

Authenticated Users

These security settings are recommended for authenticated users:

- Oracle Guest Library: **edit**
- Oracle Guest Search: **edit**
- Oracle Saved Search: **edit**
- Oracle Guest Library: **edit**
- Oracle Contribution: **edit**
- Oracle Library: **edit**
- Oracle Search: **edit**
- Oracle Workflow Queue: **view**
- Oracle Metadata Admin: **view**

  The Metadata Admin portlet is an administration portlet and should be restricted to admin-level users.

Anonymous Users

These security settings are recommended for anonymous users:

- Oracle Guest Library: **view**
- Oracle Guest Search: **view**

1.5 Portlet Interface Language

The portlet interface language is set to the client's locale. That is, if the client is using, say, a German language version of an operating system (assuming support by the JVM running the portal server), the CPS portlets will display a German interface.

2 Architecture and Request Handling

This chapter provides an overview of the Content Portlet Suite (CPS) architecture and the high-level sequence of events for portlet request-handling. It covers the following topics:

- Section 2.1, "CPS Architecture Overview"
- Section 2.2, "Model-View-Controller Framework"
- Section 2.3, "Request Handling"

2.1 CPS Architecture Overview

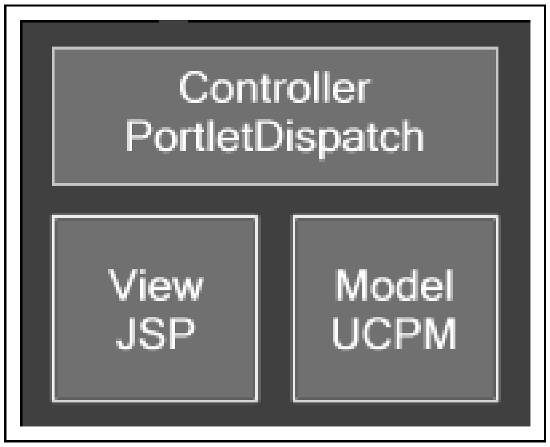
The CPS Portlets use the Remote Intradoc Client (RIDC) API to communicate with Oracle WebCenter Content Server. The Portlet API facade abstracts the common operations within portlet containers thus allowing our framework to work on a variety of platforms using the same handler code. Portlet *actions* are mapped to a custom MVC framework that uses the RIDC API to perform the desired task.

The portlets are consumers of standard content server services such as CHECKIN_UNIVERSAL and GET_SEARCH_RESULTS. These services are called by the dispatch handlers from the portlet controller using the RIDC API. Refer to the Remote Intradoc Client (RIDC) chapter of the *Oracle WebCenter Content Developer's Guide for Content Server* for more information.

2.2 Model-View-Controller Framework

CPS uses a Model-View-Controller design pattern based on the open source Struts and Tiles framework. The presentation of data is separated from the logic that obtains and organizes the data. The *model* is the UCPM layer that encapsulates the data access layer, the *view* is the JSP pages that render the model as a user interface element, and the *controller* is the PortletDispatch handler that processes and responds to events, typically user actions, and invokes changes on the model.

Figure 2–1 MVC Framework

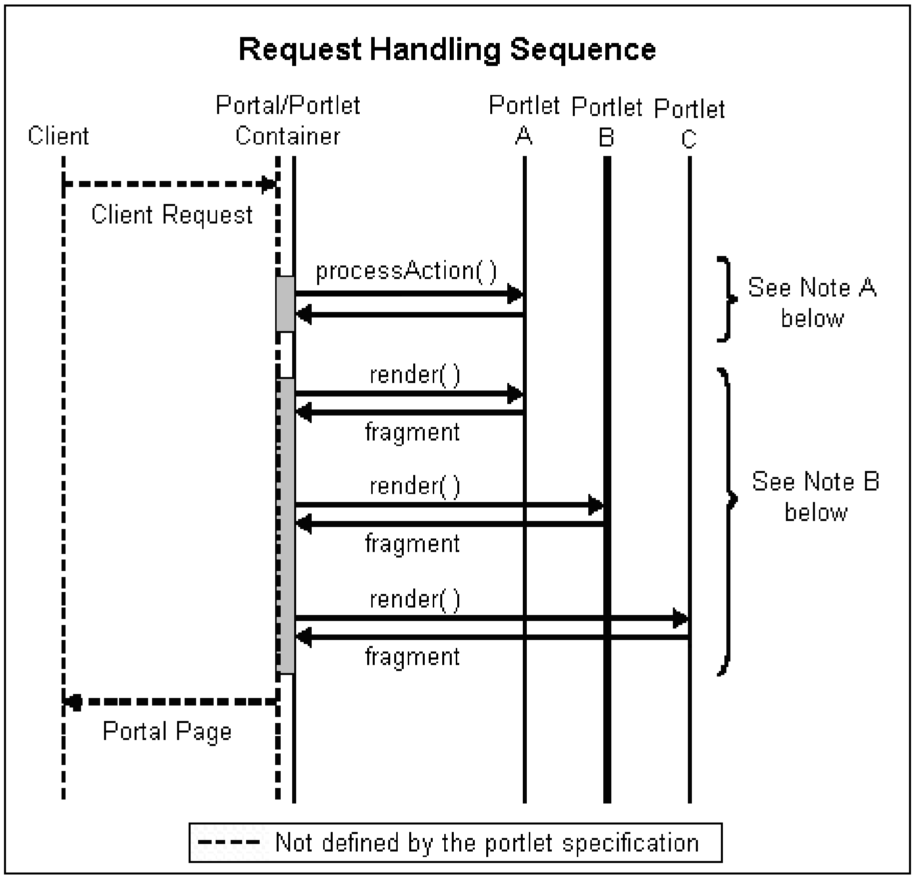


2.3 Request Handling

This is the high-level sequence of events for portlet request-handling (based on the Search portlet):

1. A user enters a query and clicks the Search button.
2. An *action* URL is built and routed to the portlet container, which, in turn, routes the command to the appropriate portlet (in this case, the search portlet).
3. A *processAction* is called on the Search portlet.
4. The Search portlet retrieves the search parameters (they are part of the URL that was built), and calls the RIDC methods which execute the content server SEARCH services and returns the search response back to the portlet.
5. The portlet container calls *render* on each of the portlets on the page (including the Search portlet), and each portlet uses the received data, or refreshes the data, and displays HTML fragments to the user.

Figure 2–2 Portlet Request Handling



Note A: The *action* request must finish before the render requests begin.

Note B: The *render* requests are not triggered in any specific order. They may be executed sequentially or simultaneously.

Portlet API Facade

Because portal vendors implement the standard differently, each action handler has access to a facade object that provides an interface that protects the user of the facade from code incompatibilities between various portal vendors.

3 Deploying CPS as a WSRP Producer

This chapter provides information required for the deployment of Oracle Content Portlet Suite (CPS) as a WSRP producer. Once CPS is deployed, the portlets can be consumed from other WSRP-enabled portal servers. It covers the following topics:

- Section 3.1, "Oracle WebCenter Portal as Producer"
- Section 3.2, "Oracle WebLogic Portal as Producer"
- Section 3.3, "IBM WebSphere Portal as Producer"
- Section 3.4, "Configuring Oracle WebCenter Content Server for CPS Communication"

Important: User IDs on the content server must correspond with user IDs on your portal server.

3.1 Oracle WebCenter Portal as Producer

This section provides information on the deployment of Content Portlet Suite (CPS) as a WSRP producer on Oracle WebCenter Portal:

- Section 3.1.1, "Installation Requirements"
- Section 3.1.2, "Supported WSRP Consumers"
- Section 3.1.3, "Creating a Custom Managed Server"
- Section 3.1.4, "Deploying CPS Portlets on WebCenter Portal"

3.1.1 Installation Requirements

These servers and components are required for installation:

- Oracle WebCenter Portal 11*g* Release 1 (11.1.1.3 or higher) (as *producer*)
- Oracle WebCenter Content Server 11*g* Release 1, Oracle Content Server 11*g* Release 1, or Oracle Content Server 10*g* Release 3
- Oracle Content Portlet Suite (CPS) distribution file

3.1.2 Supported WSRP Consumers

These WSRP consumers are supported with Oracle WebCenter Portal as producer:

- Oracle WebCenter Spaces 11*g* Release 1. See Section 4.1, "Oracle WebCenter Spaces as Consumer" for more information.

3.1.3 Creating a Custom Managed Server

To create a new managed server, follow the steps in "Creating and Provisioning a WebLogic Managed Server Using a Jython Script" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

Note: While modifying `targetServer.properties`, set `serverType=Portlet` and give a name to your server by setting the `mgdServerName` property; for example, `mgdServerName=CPS_Portlet`.

Targeting Libraries

The following libraries need to be targeted to the newly created managed server (for example, CPS_Portlet). Note that the libraries for 11*g*R1 (11.1.1.3) installations of Oracle WebCenter Portal are different from the libraries for 11*g*R1 (11.1.1.4 and higher) installations.

Libraries for 11*g*R1 (11.1.1.3) installations of WebCenter Portal:

- oracle-ridc-client-app-lib(10.3.2,10.3.2)
- oracle-ucm-spi-app-lib(10.3.2,10.3.2)
- p13n-app-lib-base(10.3.2,10.3.2)
- vcr-app-lib(10.3.2,10.3.2)
- oracle.webcenter.framework(11.1.1,11.1.1)
- oracle.webcenter.spaces.fwk(11.1.1,11.1.1)

Libraries for 11*g*R1 (11.1.1.4) and higher installations of WebCenter Portal:

- oracle.ucm.ridc.app-lib(11.1.1,11.1.1)
- p13n-app-lib-base(10.3.2,10.3.2)
- oracle.webcenter.framework(11.1.1,11.1.1)
- oracle.webcenter.spaces.fwk(11.1.1,11.1.1)
- oracle.webcenter.framework.core(11.1.1,11.1.1)

Follow these steps to target the libraries:

1. Go to the WebLogic Server Administration Console of your WebCenter Portal instance, and then open the **Deployments** section.
2. Click on the library name you want to target; for example, `oracle.webcenter.framework(11.1.1,11.1.1).`
3. On the settings page for the selected library, open the Targets tab.
4. Select the option for the newly created server (for example, CPS_Portlet) and then save and activate the changes.
5. Repeat this for each of the libraries to be targeted.

Targeting Data Sources

Target the `mds-SpacesDS` data source to the newly created managed server (for example, CPS_Portlet):

1. Go to the Oracle WebLogic Server Administration Console of your WebCenter Portal instance. In the Domain Structure window, choose **Services**, then **JDBC**, and then **Data Sources**.
2. In the Summary of JDBC Data Sources section, click **mds-SpacesDS**.
3. On the settings page for mds-SpacesDS, open the Targets tab.
4. Select the option for the newly created server (for example, CPS_Portlet) and then save and activate the changes.

3.1.4 Deploying CPS Portlets on WebCenter Portal

This section describes how to configure WebCenter Portal as a producer using Web Services Remote Portlets (WSRP) for CPS.

Note: When consuming Oracle Content Portlet Suite 11*g*R1 (11.1.1.4) on Oracle WebCenter Portal 11*g*R1 (11.1.1.4) or higher, users will see the exception:

```
java.lang.IllegalArgumentException: IdcContext.setUser() method not
valid, do not use this method,
```

To resolve the issue you may adopt either one of the following two workarounds:

Workaround 1: Edit the weblogic-application.xml

1. Deploy the library oracle-ridc-client-app-lib(10.3.2,10.3.2) and target it only at the managed server created for CPS. If you do not have this library, please contact Oracle Support.
2. Extract weblogic-application.xml file from StellentPortlets.ear file.
3. Edit the file and add the following entry as the first library reference:

```
<library-ref>
<library-name>oracle-ridc-client-app-lib</library-name>
</library-ref>
```

4. Update the StellentPortlets.ear file with the updated weblogic-application.xml file..
5. Deploy StellentPortlets.ear to your managed server.

Workaround 2: Use StellentPortlets.ear from Oracle UCM 11gR1 (11.1.1.5) or Oracle WebCenter Content 11gR1 (11.1.1.6)

The StellentPortlets.ear file from the Oracle UCM 11*g*R1 (11.1.1.5) and Oracle WebCenter Content 11*g*R1 (11.1.1.6) has the fix for this issue. Thus, you can use this EAR file instead of the EAR file bundled with Oracle UCM 11*g*R1 (11.1.1.4).

Deploying the EAR File

1. Uncompress the CPS distribution file and extract the CPS zip file to your local directory.

   For example:

   ```
   /oracle/apps/webcenter/cps
   ```

2. Launch the WebLogic Server Administration Console of your WebCenter Portal instance.
3. Click **Lock and Edit**.
4. In the Domain Structure section, click **Deployments**.
5. In the Summary of Deployments section, open the Control tab and click **Install**.
6. In the Install Application Assistant section, click the **Upload your file(s)** link.
7. In the Deployment Archive section, click the **Browse** button.
8. Browse to the /portlets directory of the extracted CPS zip file and select the StellentPortlets.ear file.
9. Click **Next**.
10. The CPS EAR file is uploaded and displays in the list of available files.
11. Enable the option button corresponding to the CPS EAR file.
12. Click **Next**.
13. Select the **Install this deployment as an application** option.
14. Click **Next**.
15. Select the new custom managed server that you created in Section 3.1.3, "Creating a Custom Managed Server" (for example, CPS_Portlet).
16. Click **Finish** to deploy the application.
17. Access the WSDL URL. Note that after deploying the CPS EAR file to WebCenter Portal as a producer, the WSDL URL for the 11*g*R1 (11.1.1.3) installation of WebCenter Portal is different from the WSDL URL for 11*g*R1 (11.1.1.4) or higher installations.

    WSRP URL 11*g*R1 (11.1.1.3) installation of WebCenter Portal:

    ```
    http://Host_Name:Port/cps/portlets/wsrp1?wsdl
    ```

    WSRP URL for the 11*g*R1 (11.1.1.4) or higher installations of WebCenter Portal:

    ```
    http://Host_Name:Port/cps/portlets/wsrp2?WSDL
    ```

Registering your WSDL

1. From a command-line, launch the WebLogic Scripting Tool (WLST), for example:

   ```
   WLS_HOME/as11r1wc/common/bin/wlst.sh
   ```

Note: The WebLogic Scripting Tool (WLST) is a command-line scripting environment that you can use to create, manage, and monitor WebLogic Server domains. Refer to the *Oracle Fusion Middleware Oracle WebLogic Scripting Tool* document for more information.

2. After the WebLogic Scripting Tool initializes, type `connect()` and press **Enter**.
3. You are prompted to enter the username, password and Admin server URL. Enter these values and press **Enter**.
4. Once connected to the server, run the following command to register your WSDL:

   ```
   registerWSRPProducer('webcenter','Provide_Any_Name_for_WSDL_Handle', 'Your_
   WSDL>',timeout=500)
   ```

   For example:

   ```
   registerWSRPProducer('webcenter','StellentWSRP','http://Host_Name:Port/cps/
   portlets/wsrp1?WSDL',timeout=500
   ```

Referencing Your Content Server

The following WLST commands should be used to create, delete, and reset an Oracle WebCenter Content Server connection:

> **Note:** The custom managed server on which CPS is deployed and the WLS_Spaces server must be restarted after executing the following commands for the changes to take effect.

- Run the following WLST command to **create** an Oracle WebCenter Content Server connection:

  ```
  createJCRContentServerConnection('CPS_Application_Name', 'CSConnection',
  socketType='socket', serverHost='Content_Server_Host', serverPort='Content_
  Server_Port')
  ```

  For example (if the CPS application name is StellentPortlets):

  ```
  createJCRContentServerConnection('StellentPortlets', 'CSConnection',
  socketType='socket', serverHost='192.0.2.250', serverPort='4444')
  ```

  For the CPS Portlets to work, the connection name must be 'CSConnection' as other connection names are not recognized
- Run the following WLST command to **delete** an Oracle WebCenter Content Server connection:

  ```
  deleteConnection(appName='CPS_Application_Name', name='CSConnection')
  ```

- Run the following WLST command to **reset** an existing Oracle WebCenter Content Server connection:

  ```
  setJCRContentServerConnection('CPS_Application_Name', 'CSConnection',
  socketType='socket', serverHost='New_Content_Server_Host', serverPort='New_
  Content_Server_Port')
  ```

- Run the following WLST command to **list** the existing Oracle WebCenter Content Server connections:

  ```
  listJCRContentServerConnections(appName='CPS_Application_Name')
  ```

Enable Your Content Server to Communicate with CPS

You must enable your Oracle WebCenter Content Server instance to communicate with CPS. See Section 3.4, "Configuring Oracle WebCenter Content Server for CPS Communication" for instructions.

3.2 Oracle WebLogic Portal as Producer

This section provides information on the deployment of Content Portlet Suite (CPS) as a WSRP producer on Oracle WebLogic Portal.

- Section 3.2.1, "Installation Requirements"
- Section 3.2.2, "Supported WSRP Consumers"
- Section 3.2.3, "Deploying CPS Portlets on WebLogic Portal"

3.2.1 Installation Requirements

These servers and components are required for installation:

- Oracle WebLogic Portal 10*g* Release 3 (as *producer*)
- Oracle WebCenter Content Server 11g Release 1, Oracle Content Server 11*g* Release 1, or Oracle Content Server 10*g* Release 3
- Oracle Content Portlet Suite (CPS) distribution file

3.2.2 Supported WSRP Consumers

These WSRP consumers are supported with Oracle WebLogic Portal as producer:

- Oracle WebCenter Interaction 10*g* Release 3. See Section 4.2, "Oracle WebCenter Interaction as Consumer" for more information.
- Oracle WebLogic Portal 10*g* Release 3. See Section 4.3, "Oracle WebLogic Portal as Consumer" for more information.

3.2.3 Deploying CPS Portlets on WebLogic Portal

This section describes how to configure WebLogic Portal as a producer using Web Services Remote Portlets (WSRP) for CPS.

Important: Before deploying CPS portlets, select an existing WebLogic domain or create a new WebLogic domain (for example, cps_domain). You must configure this domain to support WebLogic Server, Workshop for WebLogic, and WebLogic Portal.

Reference Your Content Server

1. Uncompress the CPS distribution file and extract the WebLogic CPS zip file to your local directory.
2. Locate the /WEB-INF/config directory.
3. Open the **cps-initialization.properties** file in a text-only editor. Edit the host property value to reference the IP address or host name of your Content Server instance. Edit the port property value if your content server is running on a port other than the default port of 4444. Do not remove the new line character at the beginning of the file.

   For example:

   ```
   Host=192.0.2.254
   Port=4448
   ```

Create a Portal Web Project

1. Launch Workshop for WebLogic.
2. Select **File**, then **New**, and then **Project**.
3. In the New Project dialog, expand the WebLogic Portal node.
4. Select **Portal Web Project**.
5. Click **Next**.

   The New Portal Web Project wizard displays.
6. In the Create a New Oracle WebLogic Portal Web Project step, enter a project name (for example, CPS).
7. Click **Finish**.
8. If the Open Associated Perspective dialog displays, click **Yes**.

Create a Portal EAR Project

1. In Workshop for WebLogic, select **File**, then **New**, and then **Project**.
2. In the New Project dialog, expand the WebLogic Portal node.
3. Click **Portal EAR Project**.
4. Click **Next**.

   The New Portal EAR Project wizard displays.
5. In the Create a New Oracle WebLogic Portal EAR Project step, enter a project name (for example, CPS Consumer-EAR).
6. Click **Next**.
7. In the Project Facets step, select **WebLogic Portal EAR Project Facets** from the drop-down list.
8. Click **Next**.
9. In the J2EE Modules to Add to the EAR step, select the portal web project you created (for example, CPS).
10. Click **Finish**.
11. If the Open Associated Perspective dialog displays, click **Yes**.

Import the CPS Portlets

1. Select Project Explorer view and expand the new Portal Web Project node (for example, CPS).
2. Right-click the WebContent node, and select **Import**.

   The Import wizard displays.
3. In the Select an Import Source step, expand the General node.
4. Select **File System**.
5. Click **Next**.
6. In the File System step, click **Browse** associated with the From Directory field.
7. Browse to the /portlets directory of the uncompressed CPS distribution file and click **OK**.
8. In the Import Resources from Local File System step, enable the **portlets** option.

9. Click **Finish**.
10. If prompted to overwrite Manifest.MF file, click **Yes**.
11. After the portlets folder is imported, you might notice several errors in the web project. Error files are shown with a red cross prefixed with the file in Project Explorer view. Select the files showing errors and delete them.

Publish the EAR File

1. In Workshop for WebLogic, select Servers view.
2. Right-click in Servers view, click **New**, then **Server**.

   The New Server wizard displays.
3. In the Choose Type of Server to Create step, accept the default settings and click **Next**.
4. In the Specify a WebLogic Domain Directory step, click **Browse**.
5. Browse to the new CPS domain directory and click **OK**.

   For example (on Windows):

   ```
   C:/oracle_weblogic/user_projects/domains/cps_domain
   ```
6. Click **Next**.
7. In the Add and Remove Projects step, select the CPS EAR file from the Available Projects section and click **Add** to add it to Configured Projects.
8. Click **Finish**.
9. In Servers view, right-click on the new server and select **Publish** to start the server and publish the EAR file.
10. The WSRP-enabled CPS portlets are available via the WSDL URL.

    The URL will be of the format:

    ```
    http://host_server:host_port/context-root-specified-in-portal-web-project/
    producer?wsdl
    ```

    where *host_server*, *host_port*, and *context-root-specified-in-portal-web-project* is replaced with appropriate values.

Enable Your Content Server to Communicate with CPS

You must enable your content server instance to communicate with CPS. See Section 3.4, "Configuring Oracle WebCenter Content Server for CPS Communication" for instructions.

3.3 IBM WebSphere Portal as Producer

This section provides information on the deployment of Content Portlet Suite (CPS) as a WSRP producer on IBM WebSphere Portal.

- Section 3.3.1, "Installation Requirements"
- Section 3.3.2, "Supported WSRP Consumers"
- Section 3.3.3, "Deploying CPS Portlets on WebSphere Portal"

3.3.1 Installation Requirements

These servers and components are required for installation:

- IBM WebSphere Portal Version 6.1 (as *producer*)
- Oracle WebCenter Content Server 11*g* Release 1, Oracle Content Server 11*g* Release 1, or Oracle Content Server 10*g* Release 3
- Oracle Content Portlet Suite (CPS) distribution file

3.3.2 Supported WSRP Consumers

These WSRP consumers are supported with IBM WebSphere Portal as producer:

- IBM WebSphere Portal Version 6.1. See Section 4.4, "IBM WebSphere Portal as Consumer" for more information.

3.3.3 Deploying CPS Portlets on WebSphere Portal

This section describes how to configure WebSphere Portal as a producer using Web Services Remote Portlets (WSRP) for CPS.

Deploy the EAR File

1. Uncompress the CPS distribution file and extract the WebSphere CPS zip file to your local directory.
2. Start the server and launch WebSphere Portal.
3. From the main menu, click **Administration**.
4. In the Portlet Administration section, click **Web Modules**.
5. In the Manage Web Modules section, click **Install**.
6. Browse to the /portlets directory of the extracted WebSphere CPS zip file and select the StellentPortlets.ear file.
7. Click **Next**.
8. Provide an enterprise application display name and context root.
9. Make sure the **Start Application** option is enabled (default).
10. Click **Finish**.

    A confirmation message will display in the Manage Web Modules section after successful installation.

Reference Your Content Server

1. Locate the StellentPortlets WEB-INF/config directory:

    ```
    WebSphere_Portal_Home\wp_profile\installedApps\Node_Name\CPS_App_Display_Name\
    StellentPortlets.war\WEB-INF\config
    ```

2. Open the **cps-initialization.properties** file in a text-only editor. Edit the host and port properties to reference your content server instance.

    For example:

    ```
    Host=192.0.2.254
    Port=4448
    ```

3. Stop and restart the server.

Locate the CPS Portlets

1. Launch WebSphere Portal and click **Administration** from the main menu.
2. In the Portlet Management section, click **Portlets**.
3. Search for the CPS portlets in the list of available portlets.

Assign Resource Permissions

For each of the CPS portlets perform the following steps:

1. Click the **Provide Portlet** icon.
2. Click **OK**.
3. Click the **Assign Access to Portlet** icon.

   The Resource Permissions page displays.
4. In the Roles column, click the **Edit Role** icon for the User role.
5. Click **Add**.
6. Enable the **Anonymous Portal User** option.
7. Click **OK**.
8. In the Add section, click the portlet name.
9. Repeat the steps above for the Privileged User role: click the **Edit Role** icon, click **Add**, enable the **Anonymous Portal User** option, and click **OK**.
10. When you have completed these steps for each of the CPS portlets, click **Done**.

Create the Producer

1. In the Portlet Management section, click **Web Services**.
2. Click **New Producer** and provide a title for the producer. Select this Web Service Producer when consuming the CPS portlets. See Section 4.3, "Oracle WebLogic Portal as Consumer" and Section 4.4.3, "Consumption Steps" for more information.
3. Confirm that the entries for your configuration are correct:
   - Review the URL to WSDL Service Definitions field and edit as needed.
   - Review the host name. The default host name is `localhost`. It is recommended that the IP address or machine name be used.
   - Review the port number. The default port number is `10040`. The port number should match the port where your WebSphere Portal is deployed (as displayed in the address bar of your web browser).
4. After you have confirmed your entries, click **Next**.
5. Click **Next**.

   The confirmation message "Created the Producer successfully" displays.
6. The WSDL for your producer is:

   ```
   Host_Name:Port/wps/wsdl/service.wsdl
   ```

   For example:

   ```
   http://localhost:10039/wps/wsdl/service.wsd
   ```

Enable Your Content Server to Communicate with CPS

You must enable your content server instance to communicate with CPS. See Section 3.4, "Configuring Oracle WebCenter Content Server for CPS Communication" for instructions.

3.4 Configuring Oracle WebCenter Content Server for CPS Communication

To enable your content server instance to communicate with CPS, you must define the IP address of your portal server and edit the content server configuration file (config.cfg) to allow server communication with that IP address.

This section covers the following topics:

- Section 3.4.1, "Enabling the IP Address on Content Server"
- Section 3.4.2, "Editing the Configuration File for Content Server"

3.4.1 Enabling the IP Address on Content Server

You must enable the IP address of the application server. This enables the content server to listen for connections from the application server.

1. Launch the System Properties editor.

   Select **Content Server**, then **instance name**, then **Utilities**, and then **System Properties**.

2. Open the **Server** tab.
3. Enter your application server specific information for the Hostname Filter and IP Address Filter.

   The wildcard (*) is accepted, but IP addresses must take the form x.x.x.x regardless of wildcards, and must be separated by a vertical bar.

   For example:

   ```
   12.34.56.*|12.34.57.*|12.35.*.*
   ```

4. You must restart the content server for these changes to take effect.

3.4.2 Editing the Configuration File for Content Server

You must edit the content server configuration file (config.cfg). The file is located at *MW_HOME*/user_projects/domains/*cs_domain_name*/ucm/cs/config.

Add or edit the `IntradocServerPort` configuration variable to include the port number used by CPS. This entry is present by default in Content Server 10*g* but must be added to the configuration file for Content Server 11*g*.

Follow these steps to edit the configuration file:

1. Log in to the content server as an administrator.
2. Launch the Admin Server.
3. Click **General Configuration**.
4. In the Additional Configuration Variables field, add or edit the `IntradocServerPort` entry and the port number.

For example:

```
IntradocServerPort=4444
```

5. Click **Save**.
6. Restart the content server.

4 Consuming CPS Portlets

This chapter provides information required for consuming Content Portlet Suite (CPS) portlets. It covers the following topics:

- Section 4.1, "Oracle WebCenter Spaces as Consumer"
- Section 4.2, "Oracle WebCenter Interaction as Consumer"
- Section 4.3, "Oracle WebLogic Portal as Consumer"
- Section 4.4, "IBM WebSphere Portal as Consumer"

Important: User IDs on the content server must correspond with user IDs on your portal server.

4.1 Oracle WebCenter Spaces as Consumer

This section provides information on consuming CPS portlets in WebCenter Spaces:

- Section 4.3.1, "Installation Requirements"
- Section 4.3.2, "Supported WSRP Producers"
- Section 4.3.3, "Consumption Steps"

4.1.1 Installation Requirements

These servers and components are required for the installation:

- Oracle WebCenter Spaces 11*g* Release 1 (as *consumer*)
- Oracle WebCenter Content Server 11*g* Release 1, Oracle Content Server 11*g* Release 1, or Oracle Content Server 10*g* Release 3

4.1.2 Supported WSRP Producers

These WSRP producers are supported with WebCenter Spaces as consumer:

- Oracle WebCenter Portal 11*g* Release 1. See Section 3.1, "Oracle WebCenter Portal as Producer" for more information.

4.1.3 Consumption Steps

This section describes how to consume CPS portlets on Oracle WebCenter Spaces.

1. Login to WebCenter Spaces at:

   ```
   http://Host_Name:Port/webcenter/spaces/
   ```

2. Create a new page. Select **Page Actions**, then **Edit Page**, and then click **Add Content**.
3. A new dialog displays. Click on **Portlets** and then select the WSRP Producer that you registered.
4. Select the portlets you want to include.

4.2 Oracle WebCenter Interaction as Consumer

This section provides information for consuming CPS portlets on Oracle WebCenter Interaction 10*g* Release 3.

- Section 4.2.1, "Installation Requirements"
- Section 4.2.2, "Supported WSRP Producers"
- Section 4.2.3, "Consumption Steps"

4.2.1 Installation Requirements

These servers and components are required for the installation:

- Oracle WebCenter Interaction 10*g* Release 3 (as *consumer*)
- WSRP Consumer component

  The WebCenter Interaction WSRP Consumer component must be installed to configure WebCenter Interaction as a WSRP consumer. Refer to your WebCenter Interaction installation guide for information on installing this component.
- JSR 168 Container

  Refer to the WebCenter Interaction Installation Guide for information on installing this component.
- Oracle WebCenter Content Server 11*g* Release 1, Oracle Content Server 11*g* Release 1, or Oracle Content Server 10*g* Release 3

4.2.2 Supported WSRP Producers

These WSRP producers are supported with Oracle WebCenter Interaction as consumer:

- Oracle WebLogic Portal 10*g* Release 3. See Section 3.2, "Oracle WebLogic Portal as Producer" for more information.

4.2.3 Consumption Steps

This section describes how to consume CPS portlets on Oracle WebCenter Interaction.

1. Start WebCenter Interaction and log in to the portal as an administrator.
2. Click **Administration**.
3. Click on the folder where you want to store the remote portlets.
4. From the Create Object drop-down list, select **Portlet**.
5. In the Choose Template or WebService dialog, enable the **WSRP consumer** option.

   The WSRP Consumer component must be installed for this option to display. See Section 4.2.1, "Installation Requirements" for more information.

6. Click **OK**.
7. In the Create Portlet dialog, enable the **Suppress Portlet Title Bar** option.
8. In the Configure this Portlet section, click **Edit**.
9. Enter the WSDL link of the WSRP-enabled CPS producer.
10. Click **Import Service WSDL**.
11. Click **Next**.
12. Choose the CPS remote portlet and click **Finish**.
13. In the Create Portlet dialog, click **Finish**.
14. In the Save Object dialog, provide a name and description and then save the portlet object to the desired folder.
15. Repeat these steps to add the other CPS remote portlets.

4.3 Oracle WebLogic Portal as Consumer

This section provides information for consuming CPS portlets on Oracle WebLogic Portal 10*g* Release 3.

- Section 4.3.1, "Installation Requirements"
- Section 4.3.2, "Supported WSRP Producers"
- Section 4.3.3, "Consumption Steps"

4.3.1 Installation Requirements

These servers and components are required for the installation:

- Oracle WebLogic Portal 10*g* Release 3 (as *consumer*)
- Oracle WebCenter Content Server 11*g* Release 1, Oracle Content Server 11*g* Release 1, or Oracle Content Server 10*g* Release 3

4.3.2 Supported WSRP Producers

These WSRP producers are supported with WebLogic Portal as consumer:

- Oracle WebLogic Portal 10*g* Release 3. See Section 3.2, "Oracle WebLogic Portal as Producer" for more information.

4.3.3 Consumption Steps

This section describes how to consume CPS portlets on WebLogic Portal.

Create a Portal Web Project

1. Launch Workshop for WebLogic.
2. Select **File**, then **New**, and then **Project**.
3. In the New Project dialog, expand the WebLogic Portal node.
4. Select **Portal Web Project**.
5. Click **Next**.

   The New Portal Web Project wizard displays.

6. In the Create a New Oracle WebLogic Portal Web Project step, enter a project name (for example, CPS Consumer).
7. Click **Finish**.
8. If the Open Associated Perspective dialog displays, click **Yes**.

Create a Portal EAR Project

1. In Workshop for WebLogic, select **File**, then **New**, and then **Project**.
2. In the New Project dialog, expand the WebLogic Portal node.
3. Click **Portal EAR Project**.
4. Click **Next**.

   The New Portal EAR Project wizard displays.
5. In the Create a New Oracle WebLogic Portal EAR Project step, enter a project name (for example, CPS Consumer-EAR).
6. Click **Next**.
7. In the Project Facets step, select **WebLogic Portal EAR Project Facets** from the drop-down list.
8. Click **Next**.
9. In the J2EE Modules to Add to the EAR step, select the portal web project you created (for example, CPS Consumer).
10. Click **Finish**.
11. If the Open Associated Perspective dialog displays, click **Yes**.

Import the CPS Remote Portlets

1. Select Project Explorer view and expand the new Portal Web Project node (for example, CPS Consumer).
2. Right-click the WebContent node, select **New**, and then **Portlet**.
3. Select the directory where you want the portlet to be available and provide a name for the portlet.
4. Click **Next**.

   The Portlet wizard displays.
5. In the Select Portlet Type step, enable the **Remote Portlets** option.
6. Click **Next**.
7. In the Producer step, enable the **Find Producer** option.
8. Enter the WSDL link of the WSRP-enabled CPS producer.
9. Click **Retrieve**.
10. Click **Register** to register the producer.
11. In the registration dialog, provide a Producer Handle name and enable the **Store registration properties in local registry** option.
12. Click **OK**.
13. Click **Next**.
14. Select the CPS remote portlet from the list of available portlets and click **Next**.

15. Provide proxy portlet details and click **Create**. A new remote (proxy) portlet is created.
16. From the File menu select **Save All** to save the remote portlet.
17. Repeat these steps to add the other CPS remote portlets. If you enabled the option to store registration properties in the local registry you can add the WSDL link (step 8) by clicking **Producer** and selecting it from the **Registered Producer** drop-down list.

Follow these additional steps to deploy the CPS portlets to your existing portal instance:

1. Select your Portal EAR project and open the Portal Administration Console by selecting **Run** and then **Open Portal Administration Console**.
2. Log in using your administration credentials.
3. In the left navigation pane, click **Portal Management**.
4. Within the resource tree, select **Portals**
5. Select the CPS remote portlets and add them to your existing portal.

4.4 IBM WebSphere Portal as Consumer

This section provides information for consuming CPS portlets on IBM WebSphere Portal.

- Section 4.3.1, "Installation Requirements"
- Section 4.3.2, "Supported WSRP Producers"
- Section 4.3.3, "Consumption Steps"

4.4.1 Installation Requirements

These servers and components are required for the installation:

- IBM WebSphere Portal Version 6.1 (as *consumer*)
- Oracle WebCenter Content Server 11*g* Release 1, Oracle Content Server 11*g* Release 1, or Oracle Content Server 10*g* Release 3

4.4.2 Supported WSRP Producers

These WSRP producers are supported with WebSphere Portal as consumer:

- IBM WebSphere Portal Version 6.1. See Section 3.3, "IBM WebSphere Portal as Producer" for more information.

4.4.3 Consumption Steps

This section describes how to consume CPS portlets on WebSphere Portal:

Consume the CPS Portlets

1. Launch WebSphere Portal and click **Administration** from the main menu.
2. In the Portlet Management section, click **Web Modules**.
3. Click **Consume**.

4. From the list of Web Service Producers, select the producer created to provide CPS portlets.

   See Section 3.3, "IBM WebSphere Portal as Producer" and Section , "Create the Producer" more information.

5. Select the CPS portlets you want to consume in your portal. Select all if you want to consume all the portlets.

6. Click **OK**.

Locate the CPS Remote Portlets

1. In the Portlet Management section, click **Portlets**.

2. Search for the CPS remote portlets in the list of available portlets.

   Remote portlets are distinguished from local portlets by the title of the Web Service Producer that appears under the 'Remote Portlet' column for that portlet.

Assign Resource Permissions

For each of the CPS remote portlets perform the following steps:

1. Click the **Assign Access to Portlet** icon.

   The Resource Permissions page displays.

2. In the Roles column, click the **Edit Role** icon for the User role.

3. Click **Add**.

4. Enable the **Anonymous Portal User** option.

5. Click **OK**.

6. In the Add section, click the portlet name.

7. Repeat the steps above for the Privileged User role: click the **Edit Role** icon, click **Add**, enable the **Anonymous Portal User** option, and click **OK**.

8. When you have completed these steps for each of the CPS remote portlets, click **Done**.

Add the Portlets to Your Page

1. In the Portal User Interface section, click **Manage Pages**.

2. In the Title column, click **Content Root** and then **Home**.

3. Click **New Page**.

   The Page Properties page displays.

4. Provide a title and unique name.

5. Click **OK**.

6. Click the **Home** link at the top left of the page and then click the tab with your newly created page.

7. Hover your cursor over the title, click the displayed icon, and select **Edit Page Layout**.

8. Click **Add Portlets**.

9. Select each of the CPS remote portlets from the list of available portlets and click **OK**.

10. When you have completed these steps for each of the CPS remote portlets, click **Done**.

> **Note:** If users are using Internet Explorer 8, it is recommended that they view the portlet pages in compatibility mode. If they do not, the portlet icon menu options do not render correctly and are not available.

5 CPS Portlet Functionality

This chapter provides information on the CPS portlets, including a short tutorial on using the CPS portlets, and instructions on how to customize the Library and Guest Library portlets. It covers the following topics:

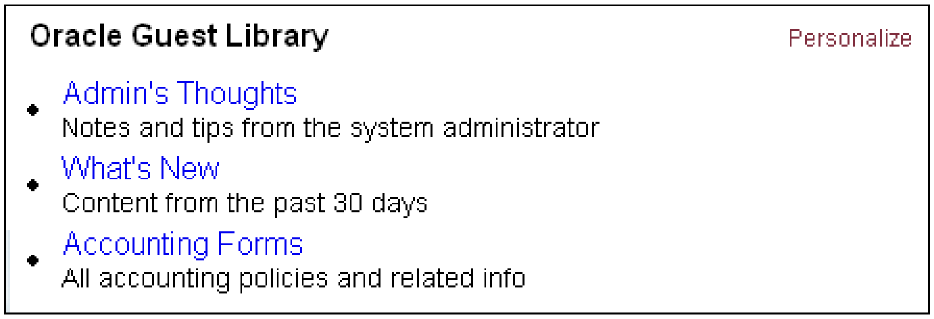
The portlet images in this section are examples only. The actual look (icons, colors, and so on) may be different on your portal server instance.

Note: If the portlets are on WebSphere and users are using Internet Explorer 8, it is recommended that they view the portlet pages in compatibility mode. If they do not, the portlet icon menu options do not render correctly and are not available.

5.1 Guest Library Portlet

The Guest Library portlet enables portal users to view content that matches criteria predefined by the portlet administrator. The results of the predefined search are delivered to the user in a list format. By selecting a hyperlink item in the list, the user can view the content. The user can also view the content item information page. The content displayed to the user is defined by the portal administrator.

Figure 5–1 Guest Library Portlet


Oracle Guest Library

Personalize

- Admin's Thoughts
  Notes and tips from the system administrator
- What's New
  Content from the past 30 days
- Accounting Forms
  All accounting policies and related info

Using the Guest Library Portlet

1. Click **Personalize**.

   Personalization enables users to modify the displayed results from a search.

   - **Results Per Page**: Enter the number of results to display per page. Default is 20.
   - **Sort By**: Select the field to sort by. Options are Date or Title.
   - **Sort Order**: Select the sort order. Options are Ascending or Descending.

2. Modify the settings as desired and click **Save** when edits are complete.
3. Select a search query (for example, What's New).

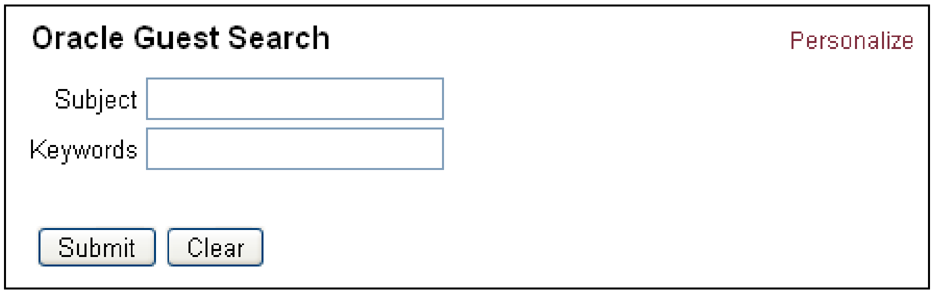
   The results are displayed in a list format.

4. Click **Back**.

5.2 Guest Search Portlet

The Guest Search portlet enables portal users to search the content server for business content. Users can search by subject and/or keyword. The results of the search are delivered to the user in a list format. By selecting a hyperlink item in the list, the user can view the content.

Figure 5–2 Guest Search Portlet



Using the Guest Search Portlet

1. Click **Personalize**.

   Personalization enables users to modify the displayed results from a search.

   - **Results Per Page**: Enter the number of results to display per page. Default is 20.

- **Sort By**: Select the field to sort by. Options are Date or Title.
- **Sort Order**: Select the sort order. Options are Ascending or Descending.

2. Modify the settings as desired and click **Save** when edits are complete.
3. Enter a subject or keyword and click **Search**.

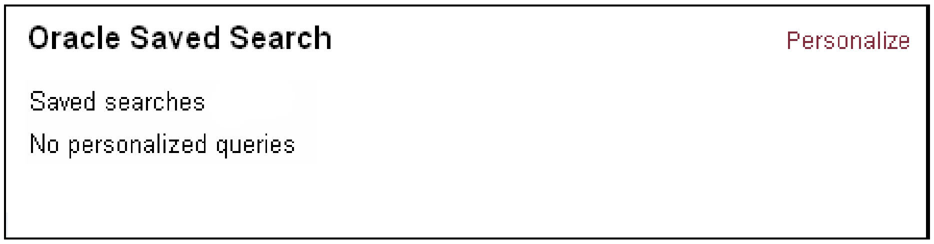
   There must be content checked into your content server that contains the specified subject or keyword for items to be retrieved.
4. In the search results screen, click **Save**.

   The save search query screen is displayed. This functionality saves a search to the saved searches list of the Saved Search portlet.
5. The keyword used to perform the search is displayed in the Keywords field. To save a search, enter a title for the query, the content item title, and click **Add New Query**.
6. Click **Back** to return to the search results page.

5.3 Saved Search Portlet

The Saved Search portlet enables portal users to view previously saved searches. Saved searches are delivered to the user in a list format. By selecting a hyperlink item in the list, the user can view the content of the saved search. Saved searches can be saved to the list by performing a search using the Search portlet and saving that search or by directly defining search parameters on the Saved Search portlet.

Figure 5–3 Saved Search Portlet



Oracle Saved Search

Personalize

Saved searches

No personalized queries

Using the Saved Search Portlet

Searches can be saved to the Saved Searches list by performing a search using the Search portlet and saving that search, or by directly defining search parameters on the Saved Search portlet.

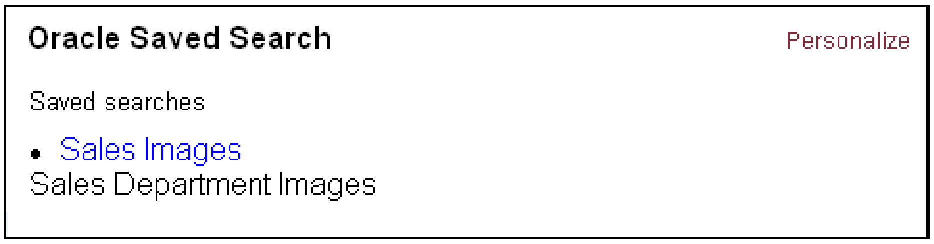
1. Click **Personalize**.

   The search criteria page is displayed. On this page you can edit your standard search parameters, view or delete items from your saved search list, and create a new query.
2. Enter the information for the new query:
   - Query Title (for example, Sales Images).
   - Query Description (for example, Sales Department Images).
   - Query Text (for example, dDocTitle <substring> `salesimage`).

   Rather than entering a query string, you can perform a standard search using the Search portlet, click **Save**, and then click **Add New Query.**

3. Click **Add New Query**.

Figure 5–4 Saved Search Portlet - Saved Query



Oracle Saved Search

Personalize

Saved searches

- Sales Images

Sales Department Images

4. Click the saved search title.

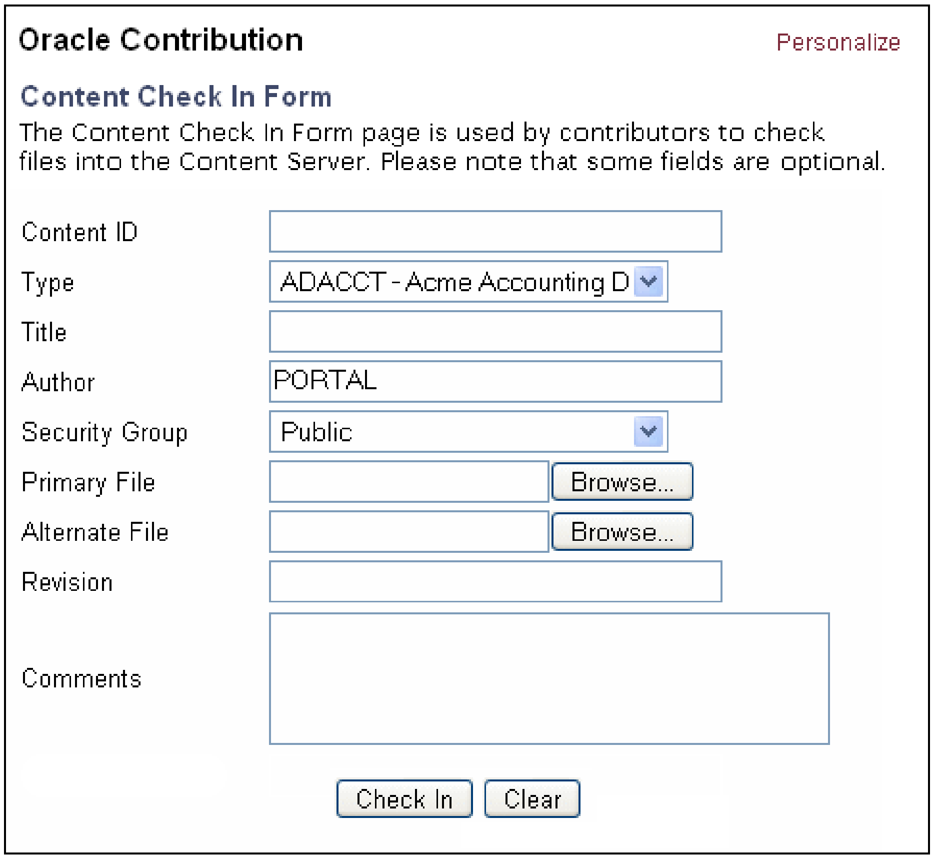
   The saved search results are displayed in a list format.

5. Click **Back**.

5.4 Contribution Portlet

The Contribution portlet enables portal users to submit native business content for management and publishing. Content submission can trigger automatic indexing, workflow routing, subscription services, and security. Additionally, content server add-on modules can enable publishing in a variety of formats.

Figure 5–5 Contribution Portlet



Oracle Contribution

Personalize

Content Check In Form

The Content Check In Form page is used by contributors to check files into the Content Server. Please note that some fields are optional.

Content ID	
Type	ADACCT - Acme Accounting D
Title	
Author	PORTAL
Security Group	Public
Primary File	
Alternate File	
Revision	
Comments	

Using the Contribution Portlet

1. In the Contribution portlet, enter required metadata and any desired optional metadata.

   The Contribution portlet enables users to submit business content created in native applications for publishing and revision control. Metadata fields are defined by the content server administrator.

2. In the Primary File field, enter the complete path to the content or click **Browse** to select the content for check in.

3. Click **Check In** to submit the content.

Note: The contribution portlet has some known issues when the component for Oracle WebCenter Content: Records or Oracle Universal Records Management (Oracle URM) has been enabled on the content server, including:

- Rendering the CPS contribution portlet takes a long time. Please ensure that your WSRP request timeout on the consumer is set to a sufficiently high value.
- The CPS contribution portlet displays two additional fields: 'Folder ID' and 'Category ID'. These fields are not visible on the content server check-in page.
- The 'Category or Folders' field is a required field for checking items in to the content server. However, the contribution portlet does not highlight this field as a required field. If users attempt to check in a document from the contribution portlet without providing a value for this field, check-in will fail.

5.5 Workflow Queue Portlet

The Workflow Queue portlet enables portal users to quickly access content that is waiting for their review. This portlet provides users direct access to their workflow in-boxes for the review and approval of business content. A workflow must be defined for this portlet to provide functionality. The portlet lists the content item title, workflow step information, the last action performed, and the actions allowed, and provides a link to content item information.

Figure 5–6 Workflow Queue Portlet

A workflow must be defined for this portlet to provide functionality. Refer to the workflow chapter in the *Oracle WebCenter Content Application Administrator's Guide for Content Server* for instructions on defining a workflow.

This example assumes a criteria workflow with these properties:

- Name: WorkflowTest

- Security Group: Public
- Has Criteria Definition: ENABLED
- Field: Type
- Operator: Matches
- Value: ADSALES

Also, it assumes a workflow step with these properties:

- Name: StepOne
- Type: Reviewer
- User: User1

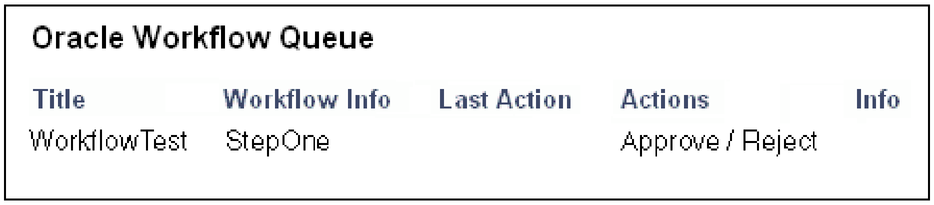
Using the Workflow Queue Portlet

1. Locate the Workflow Queue portlet.
2. After creating a workflow, log in as a workflow reviewer (for example, user1) and check in a content item matching the specified criteria.
3. Review the workflow item.

   The Workflow Queue portlet provides this functionality:

   - **Title**: Lists the content item. Click to download file.
   - **Workflow Info**: Lists the active step. Click to view workflow step information.
   - **Last Action**: Lists the last action the user performed on the workflow.
   - **Action**: Users defined as reviewers can approve or reject content. Users defined as reviewer/contributor can approve, reject, or contribute content.
   - **Info**: Provides content item information.

Figure 5–7 Workflow Queue Portlet - Workflow Defined



Oracle Workflow Queue

Title	Workflow Info	Last Action	Actions	Info
WorkflowTest	StepOne		Approve / Reject	

4. Click **Workflow Info** step name to view the workflow step information.
5. Click **Back**.
6. Click **Approve** or **Reject**.

   Depending on how your approval criteria is defined, clicking Approve may send the content item to the next workflow step. However, each workflow can include multiple review and notification steps, and multiple reviewers can be assigned to approve or reject the content at each step. Thus, a workflow step may require approval from a number of reviewers to go to the next step in the workflow.

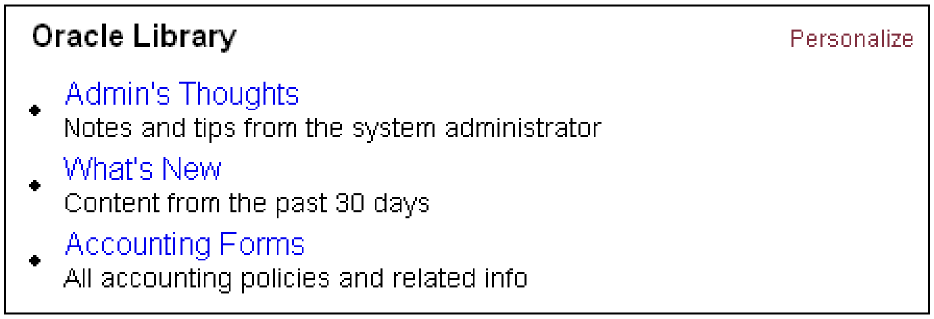
   Clicking Reject sends the content item back to the most recent Review/Edit Revision or Review/New Revision step. If there is no such step, the content goes back to the original author.

Refer to the workflow chapter in the *Oracle WebCenter Content Application Administrator's Guide for Content Server* for more information.

5.6 Library Portlet

The Library portlet enables portal users to perform a selected metadata and keyword search on the content server, and provides read/write access to the returned content. This portlet also enables users to save queries for display in the Saved Search portlet.

Figure 5–8 Library Portlet


Oracle Library

Personalize

- Admin's Thoughts
  Notes and tips from the system administrator
- What's New
  Content from the past 30 days
- Accounting Forms
  All accounting policies and related info

Using the Library Portlet

1. Click **Personalize**.

   Personalization enables users to modify the displayed results from a search.

   - **Results Per Page**: Enter the number of results to display per page. Default is 20.
   - **Sort By**: Select the field to sort by. Options are Date or Title.
   - **Sort Order**: Select the sort order. Options are Ascending or Descending.

2. Modify the settings as desired and click **Save** when edits are complete.
3. Select a search query (for example, What's New).

   The search results are displayed.

4. Click the **Info** icon.
5. In the Content Info page, these options can be selected from the list:

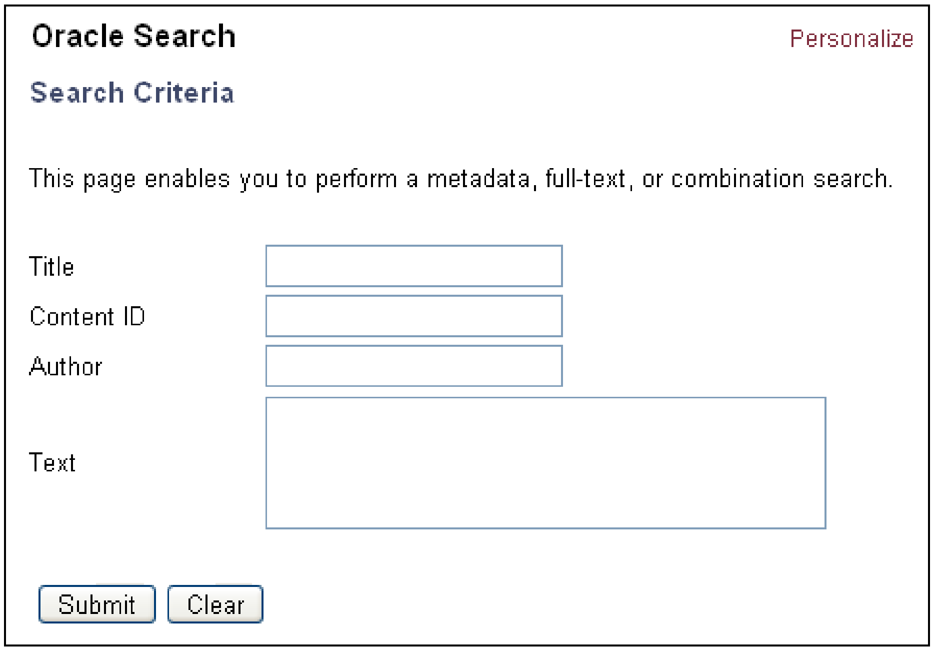
   - Content Information
   - Check Out
   - Check in Similar
   - Send Link by Email

6. Click **Back**.

5.7 Search Portlet

The Search portlet enables portal users to view content that matches user-defined query criteria, and provides read/write access to the returned content. This portlet also enables users to save queries for display in the Saved Search portlet.

Figure 5–9 Search Portlet


Oracle Search

Personalize

Search Criteria

This page enables you to perform a metadata, full-text, or combination search.

Title	
Content ID	
Author	
Text	

Using the Search Portlet

1. Enter a search query (for example, Sample) and click **Submit**.

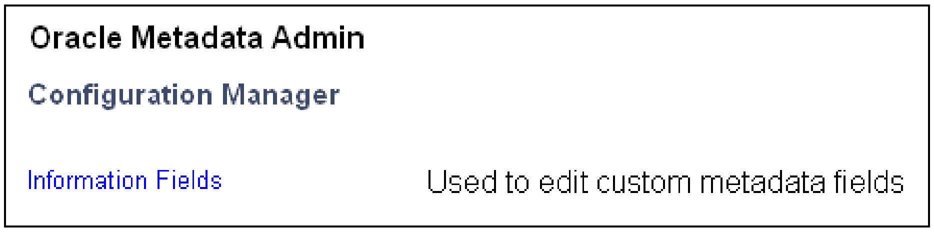
   The results from the search query are displayed.

2. Click the **Info** icon.
3. In the Content Info page, these options can be selected from the list:
   - Content Information
   - Check Out
   - Check in Similar
   - Send Link by Email
4. Click **Back**.

5.8 Metadata Admin Portlet

The Metadata Admin portlet enables administrators to modify the properties of custom metadata.

Figure 5–10 Metadata Admin Portlet


Oracle Metadata Admin

Configuration Manager

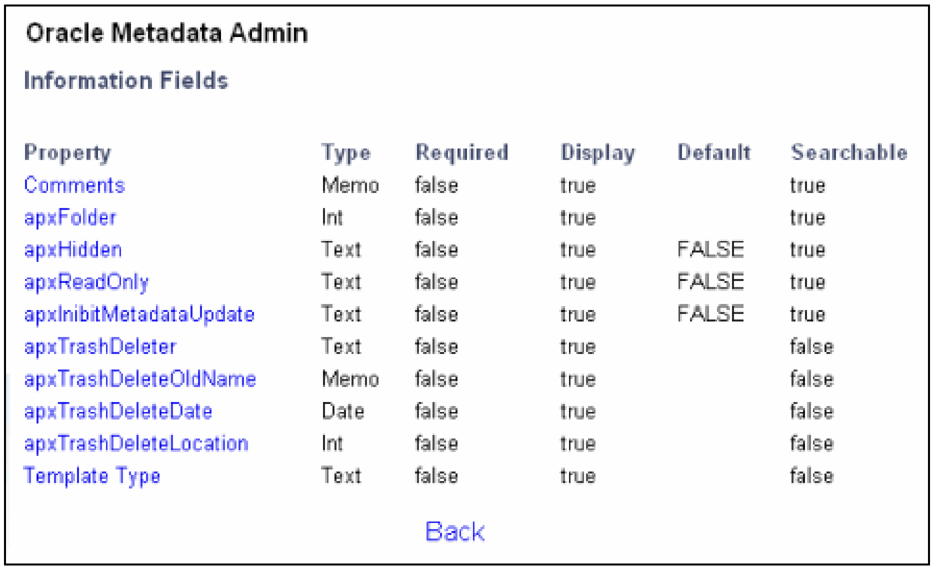
Information Fields	Used to edit custom metadata fields

Using the Metadata Admin Portlet

1. Click **Information Fields**.

   The Information Fields list is displayed. Information fields are defined by the content server administrator.

Figure 5–11 Metadata Admin Portlet - Information Fields



Oracle Metadata Admin

Information Fields

Property	Type	Required	Display	Default	Searchable
Comments	Memo	false	true		true
apxFolder	Int	false	true		true
apxHidden	Text	false	true	FALSE	true
apxReadOnly	Text	false	true	FALSE	true
apxInibitMetadataUpdate	Text	false	true	FALSE	true
apxTrashDeleter	Text	false	true		false
apxTrashDeleteOldName	Memo	false	true		false
apxTrashDeleteDate	Date	false	true		false
apxTrashDeleteLocation	Int	false	true		false
Template Type	Text	false	true		false

Back

2. Click an information field (for example, Comments).

   The Edit Field page is displayed.

3. Make changes as desired and click **Apply** or **Cancel**.

5.9 Customizing the Portlets

This section provides information on customizing the portlets including customizing the Library and Guest Library portlets that display queries predefined by the administrator.

These procedures are performed by an administrator and are not user configurable.

Administrators can define the business content displayed to users in portlets by editing the portlet.xml file. A query string defines the parameters to search on, and any metadata field can be used to define the search query.

1. Uncompress the CPS distribution file or locate the deployed portlet EAR file on your Portal Server instance.
2. Uncompress the WAR file (for example, cps-oracle-war-*version*.war).
3. In the WEB-INF directory, open the portlet.xml file in a text-only editor.
4. Locate the <portlet> entries for the Library and Guest Library portlets.

```
<portlet>
  <description>Oracle Library Portlet</description>
  <portlet-name>stellentLibrary</portlet-name>

<portlet>
  <description>Oracle Guest Library Portlet</description>
  <portlet-name>stellentGuestLibrary</portlet-name>
```

5. For either the Library or the Guest Library portlet, locate the <preference> entries that provide the query definitions. To customize the portlet, you must edit the <value> entries.

```
<preference>
  <name>query1.title</name>
  <value>What&apos;s New</value>
</preference>
<preference>
  <name>query1.description</name>
  <value>Content from the past 30 days</value>
</preference>
<preference>
  <name>query1.query</name>
  <value>dInDate &gt; `&lt;$dateCurrent(-30)$&gt;`</value>
</preference>
```

6. Edit the title and description entries. The title is the search title displayed to the user and the description is the short search description displayed to the user.

   For example: Important Notices and Important messages from the company president.

```
<name>query3.title</name>
<value>Important Notices</value>

<name>query3.description</name>
<value>Important messages from the company president.</value>
```

7. Edit the query entry defined by a query string using standard XML notation. To code a query string, use a predefined metadata parameter, a query string modifier, and an appropriate metadata value.

   For example, define a query string that returns content items checked in by author president (dDocAuthor matches president):

```
<name>query3.query</name>
<value>dDocAuthor &lt;matches&gt; &apos;president&apos;</value>
```

   Any predefined metadata parameter can be used to create commands, for example:

   - **dInDate**: The content item release date (that is, the date the content item was released to the web).
   - **dDocAuthor**: The content item author (for example, user1 or sysadmin).
   - **dDocType**: The content item type (for example, ADACCT or FILES).

   You must use the metadata parameter, not the field title (for example, dDocAuthor not Author).

   These query string modifiers can be used:

   - contains
   - starts
   - ends
   - matches
   - substring

8. When your edits are complete, save the portlets.xml file.
9. Recompress the WAR file.
10. Deploy the updated version of the edited portlet.

A CPS Software Development Kit

This section covers these topics:

A.1 Portlet Development Tips

Whenever possible, CPS follows web standards and uses the Model-View-Controller design pattern. By following these best practices your portlets will be portable and maintainable. Portlet developers should use these coding guidelines when designing and developing portlets.

This is not intended as a primer for portlet development, as it does not address the fundamentals of portlet programming. Instead, use these guidelines as a checklist during design and code reviews to help promote consistent and quality portlet implementations.

Use these practical recommendations when developing portlets:

- **Use taglibs whenever possible**. Encapsulating Java code within JSP Tag Libraries allows you to more easily reuse common functions and makes the JSP pages easier to update.
- **Do not give portlets and servlets the same name**. Some portal servers use the portlet name to identify the portlet within a web application and may cause errors if encounters a servlet with the same name.
- **Do not use head or body tags**. The portlet JSP page contributes to the content of a larger page. Because the HTML fragment is being added to a table cell <td></td> in the portal, it should not include <html>, <head>, or <body> tags.

- **Avoid client-side JavaScript**. Using JavaScript executed on the browser makes your portlets browser-dependent and requires additional cross-browser testing.
- **Follow the Model-View-Controller design pattern**. CPS uses a Model-View-Controller design pattern based on the open source Struts and Tiles framework. Thus, the presentation of data should be separated from the logic that obtains and organizes the data.
- **Use the JavaServer Pages Standard Tag Library (JSTL)**. The JSTL defines many commonly needed tags for conditions, iterations, formatting, and so on. When you see the c: prefix in the code of JSP pages, these tag libraries are being used. You can find more information about these tag libraries at:

  `http://jakarta.apache.org/taglibs/`

A.2 CPS SDK Directory Structure

The CPS SDK can be found in the /sdk directory of the CPS distribution package. It consists of these subdirectories:

- **ReferencePortlets**: Contains the source code for the CPS Portlets, including the Java code, JSP pages, and the Ant build.xml file used to create customized portlets. This allows those who want to customize the portlets to have access to the source code for the portlet JSP pages and the Model-View-Controller framework.
- **PortletBuilder**: Provides the structure for creating new portlets using the Model-View-Controller framework. It includes an Ant build.xml file that can be used to create custom portlets for a target platform.
- **lib**: Contains the CPS SDK tag libraries bundled in JAR files.
- **sample**: Contains a sample development portlet as an example on how to use the PortletBuilder directory to create a custom portlet.

Apache Ant is a Java-based build tool that must be installed in order to build customized portlets. This tool is available at:

`http://ant.apache.org`

A.3 Building CPS Portlets

Any portlet you build with the CPS SDK contains a dispatch configuration file, a set of JavaServer Pages, and a set of action handlers. When the user clicks a link in a specific portlet, the associated action handler is executed and the result of the action is placed on the request. The tile configured to be the destination after the action is executed is then retrieved and the associated JSP pages are inserted. The JSP page then models the data that was the result of the action. See Section A.7, "Creating a Dispatch Configuration" for more information.

A.4 Using ReferencePortlets and PortletBuilder

The CPS SDK includes the ReferencePortlets and PortletBuilder directories. The ReferencePortlets directory contains source code and the PortletBuilder directory contains the portlet build files. These directories share a similar build environment and Ant scripts.

This directory structure is used by the supplied Ant file to build a portlet distribution. The PortletBuilder Ant script builds a single portlet as an example of how to package the needed portlet files for a portal container. Developers wanting to build many portlets should adapt the scripts accordingly.

Directory Structure	Definition
lib/compile/$portalvendor	Contains the libraries needed for building the portlets.
lib/deploy/$portalvendor	Contains the libraries needed for deploying the portlets.
resources/$portalvendor	Contains global files and portal vendor specific files needed for portlet packaging.
src	The source files for the new portlet.
build/$appserver	The directory generated during the build to hold the classes and other build-related files.
build/$appserver	The directory generated after the build is run to hold the built portlet.
dist/$appserver	Contains the libraries needed for building the portlets.

A.5 Using Ant to Build Portlet Distributions

Both the PortletBuilder and ReferencePortlets root directories contain an Ant file that performs the compilation and packaging of the portlet. This root directory will be referred to as $workingdirectory. The distribution process is invoked by the following commands:

```
cd $workingdirectory
ant dist
```

For this distribution to work correctly, the following two environment variables should be set in the build.properties file in the $workingdirectory directory.

Property Name	Definition
portal.vendor	The name of the portal vendor that is the target for the current distribution.
portlet.name	The name the user wants to use for the current build. This name will be used in the generation of the descriptor files for the portlet.

The newly built portlet can be found in the $workingdirectory/dist/$portal.vendor directory. Apache Ant must be installed for this process to work properly. This tool is available at:

`http://ant.apache.org`

A.6 Using the CPS Tag Libraries

The CPS Tag Libraries includes several tags that may be useful when building customized portlets. The CPS Tag Libraries are located in the /lib directory and are bundled in JAR files.

This section covers the following topics:

- Section A.6.1, "URI Creation"
- Section A.6.2, "Error Handling"
- Section A.6.3, "Portlet Preferences"

A.6.1 URI Creation

Creates a URL through the PortletAPIFacade. The `mode` parameter is optional and, if used, the created URL will cause the portlet mode to be switched to the user-specified value. This tag is often used in conjunction with the following two nested tags:

```
<SCS:CreateURI mode=("edit" | "help" | "view"">
```

Modify the created URL by specifying an action to perform. This action name is defined in the PortletDispatch.xml file (see Portlet Dispatch Framework).

```
<SCS:URIAction name="$actionName">
```

Add the name/value pair to the generated URL.

```
<SCS:URIParameter name="$paramName" value="$paramValue">
```

Code Sample

```
<a href="
<scsportlet:createURI>
<scsportlet:URIAction value="checkOut"/>
<scsportlet:URIParameter name="documentID" value='<%=id%>' />
</scsportlet:createURI>">
Check Out File
</a>
```

A.6.2 Error Handling

Determines if an error is present. If an error is found, the body of the tag is evaluated and the error object variable is set.

```
<SCS:Error id="$errorObject">
```

Code Sample

```
<scsportlet:error id="error">
<div class="portlet-msg-error">
<%=error.getMessage ("%></div>
</scsportlet:error>
```

A.6.3 Portlet Preferences

Retrieves the specified portlet preferences and stores the result in the specified variable name.

```
<SCS:getPreference preference="$prefName" result="$resultVar">
```

Code Sample

```
<scsportlet:getPreference preference="maxResults" result="maxResultsVar" />
```

A.7 Creating a Dispatch Configuration

A dispatch configuration file defines each action handler, each tile, and information about the portlet itself. By default, the naming convention is stellent<portletname>dispatch (for example, stellentactivesearchdispatch.xml).

The entry point to CPS Portlets is the SCSPortlet class (there may be different implementations of this per container). This class extends the GenericPortlet class. At initialization, this class looks for a configuration file in the /WEB-INF/config directory.

This section covers the following topics:

- Section A.7.1, "Keywords"
- Section A.7.2, "Active Search Dispatch Configuration"
- Section A.7.3, "Types of Child Nodes"

A.7.1 Keywords

These special keywords can be used as view targets:

- **default**: Renders the default page for the portlet as defined by the default-action node; if the user is in edit mode, the default edit mode page is displayed.
- **previous**: Renders the previous page in the stack.
- **login**: Specifying an action node with this name will cause the framework to execute the action handler upon detection of a new login.
- **error**: CPS displays a default error page that assumes a throwable error has been placed on the request, you may override this error page by creating a new action definition using the `error` keyword.

A.7.2 Active Search Dispatch Configuration

Here is an example of the active search dispatch configuration coding:

```
<portletdispatch-config>
<!--
Default action parameters, name for the default action, cacheResult is a
boolean that specifies whether the default behavior is to cache the action
result on the session. If the value is set to false, the action will be
performed each time the portlet is rendered, the result data is discarded
each time.

The cacheResult value here can be overriden by the action definition itself,
it the action does not specify, the default value is used.
-->

<default-action view="showHome" edit="showEdit" cacheResult="true"/>

<!--
Portlet-id is used to ensure that unique HTML form, javascript names are
used, this value will be available on the request object as
ISCSAction.PORTLET_ID
-->

<portlet-id value="active_search_portlet"/>

<!--
Definitions for all the action types available to this portlet
-->

<action-mappings>
  <forward name="showHome" authRequired="true" path="active.search.main.page"/>
  <forward name="showEdit" authRequired="true" path="active.search.edit.page"/>
    <location path="/WEB-INF/actions/active_search_actions.xml"/>
    <location path="/WEB-INF/actions/active_document_actions.xml"/>
</action-mappings>

<!--
```

```
Definitions for UI components available to this portlet
-->

<tiles-definitions>
  <definition name=".mainLayout" path="/stellent/ui/layouts/mainlayout.jsp">
    <put name="header" value="/stellent/ui/fragment/header.jsp"/>
    <put name="footer" value="/stellent/ui/fragment/footer.jsp"/>
    <put name="content" value="/stellent/ui/layouts/defaultContent.jsp"/>
  </definition>
  <location path="/WEB-INF/tiles/active_search_tiles.xml"/>
  <location path="/WEB-INF/tiles/active_document_tiles.xml"/>
</tiles-definitions>
</portletdispatch-config>
```

A.7.3 Types of Child Nodes

The top-level node, <portletdispatch-config>, can have these types of child nodes: default-action, portlet-id, location, action-mappings, and the Tiles-definition node.

This section covers the following topics:

- Section A.7.3.1, "Default Action Node"
- Section A.7.3.2, "Portlet ID Node"
- Section A.7.3.3, "Location Node"
- Section A.7.3.4, "Action Mappings Node,"
- Section A.7.3.5, "Tiles-Definitions Node"

A.7.3.1 Default Action Node

The <default-action> node is used to specify the action to execute or tile to display when the portlet or the edit mode is first visited. It will also be the action executed when the keyword `default` is the target of another action.

view/edit attribute: Specifies the view/edit default; the values are the name of a defined action and the default edit action. For example, the defined action of `showHome` and the default edit action of `showEdit`.

cacheResult attribute: Indicates whether or not the result of the action should be cached on the session or re-executed each time the render () method is called. When users perform actions on other portlets it generates a render call which asks the portlet to redraw itself. If `cacheResult` is set to `true`, this redraw will not re-execute the action but instead uses the cached result. In the case of the Active Search portlet, it is set to true by default. Individual actions can override the default for the portlet, this value is only used when not specified by the action definition.

A.7.3.2 Portlet ID Node

The <portlet-id> node is used to specify a unique name for the portlet. The string value specified here is made available on the request with the parameter name ISCSAction.PORTLET_ID. This ID is mainly used to uniquely identify HTML elements such as forms and JavaScript functions so they do not conflict with other portlets on the same page.

A.7.3.3 Location Node

The <location> node is used to specify another dispatch configuration file in which to load definitions from. It takes a path attribute and indicates where to look for the configuration file to be loaded.

This node can be a child of the Action Mappings node or the Tile Definitions node. If there is a name conflict, the Action Mappings definitions in the current configuration XML takes precedence over the loaded action definitions.

A.7.3.4 Action Mappings Node

The <action-mappings> node is the container for action definitions, which are usually defined by an Action node; two exceptions are the Forward node and the Location node.

Action Node

The <action> node specifies several attributes, which are used to perform the desired action. Here is an example of an action definition:

```
<!--
Shows the form to add new saved search.
-->

<action
  name="active.search.showAddSavedSearch"
  class="com.stellent.portlet.components.search.active.handlers.
        ShowAddSavedSearchHandler"
  bean="com.stellent.portlet.components.search.active.forms.
        AddSavedSearchForm"
  authRequired="true"
  addToStack="false">
  <forward name="success" path="active.search.savedsearch.add.page"/>
</action>
```

The attributes are:

- **name**: The name of the action. This is used when executing this action within a JSP page.
- **class**: The fully qualified class name of the class that implements the ISCSActionHandler interface. This class is where you will add your action handler code. Both the name and class attributes are required to define an action.
- **bean**: The fully qualified class name of a class that implements the ISCSActionForm interface. This is passed into your action handler when the handleAction method is called. This bean can be populated through an HTML form post or by explicit definition through a special CPS portlet tag. This is an optional attribute, if the action does not need any input parameters this attribute can be omitted, the ISCSActionForm passed into the handleAction will be null.
- **authRequired**: Controls whether the framework will execute the action if an unauthenticated portal user tries to perform an action. It defaults to false and is an optional parameter. If it is set to `true` and an unauthenticated user attempts to execute the action, a special system JSP page will displayed asking the user to login before attempting to use the portlet.
- **addToStack**: Defines whether the portlet framework will execute this action again or cache the result when `render` is called for a redraw. It defaults to true so that portlets will show the last state when redrawing. However, some actions should not be performed more than once. Thus, you can define that the framework not save the result or remember it as the last action.

The <action> node is also a container for any number of forward actions, which specify which tile or JSP page the portlet should display upon completion of the action. You may specify as many different `forwards` as you want in this list, as long as they have unique names. The action handler code itself specifies which `forward` to use upon completion of the action. If the handler does not explicitly state the view name to forward to upon completion of the action, it defaults to the success forward.

In addition to these framework properties, you may specify an arbitrary number of custom attributes for an action definition. These attributes will be available to the action handler via the ISCSActionHandler `getAttributes()` method. For example, the Contribution portlet adds the custom property `async` that indicates whether contributions should leverage the Java Messaging Service (JMS) to perform document contributions asynchronously.

Forward Node

The <forward> node is a special type of action that does not actually execute any code but rather automatically forwards the display to the specified tile definition or explicit JSP page location. The path attribute accepts either of these values.

A.7.3.5 Tiles-Definitions Node

The Tiles and Struts design pattern tells the framework how to render a particular view by specifying a main JSP page, various regions of content, and an optional controller class that are all used to create the final view.

Example:

```
<definition name=".mainLayout" path="/stellent/ui/layouts/mainlayout.jsp">
  <put name="header" value="/stellent/ui/fragment/header.jsp"/>
  <put name="footer" value="/stellent/ui/fragment/footer.jsp"/>
  <put name="content" value="/stellent/ui/layouts/defaultContent.jsp"/>
</definition>
```

This defines a tile of the name `.mainLayout` and specifies the JSP page `/stellent/ui/layouts/mainlayout.jsp` to be the main JSP page to use when rendering this view. Notice the Put nodes, which specify the regions of content available to the main JSP page. In this example, three regions are available: a header, footer, and content region. Each of these Put nodes specify a name for the region and the corresponding JSP page to use to render the region.

You can also specify a controller for tile definitions and specify inheritance, as in the following definition:

```
<definition name="active.search.edit.page" extends=".mainLayout"
  controllerClass="com.stellent.portlet.components.search.active.
  controllers.EditController">

<put name="content" value="/stellent/ui/layouts/search/active/
  active_search_edit.jsp"/>
</definition>
```

This tile extends the `.mainLayout` tile that we defined earlier and inherits its configuration. We add a controllerClass to this tile which is an object that implements the ISCSController interface and provides a hook to execute Java code before the tile is rendered in situations where processing is required. Notice that this tile definition overrides the `content` region and changes the JSP page that is used to render this region.

A.8 Getting a Reference to the Portlet API Facade

An action definition is invoked through a JSP page, like the following:

```
<form name="subAuthSearch" method="POST" onSubmit="prepareAuthScsSearch()"
    action='<scsportlet:createURI><scsportlet:URIAction
    value="active.search.doSearch"/>
    </scsportlet:createURI>'>
```

In this example, the CPS tag `createURI` invokes the `active.search.doSearch` action when the form is submitted. The action `active.search.doSearch` maps to an action definition created in the configuration file. See Section A.7.3.4, "Action Mappings Node" for additional information.

The action definition specifies a class name. The class name should be an object that implements the ISCSActionHandler interface. This interface has a variety of methods that can be implemented which the framework uses to exercise the object. However, by extending the abstract base class SCSActionHandler, the developer need only implement one method:

```
/**
  * Handle an action from the portlet
  * @param portletRequest
  * @throws com.stellent.portlet.dispatcher.PortletDispatcherException
*/
public ISCSActionResult handleAction (ISCSActionForm form,
  Object portletRequest)
  throws PortletDispatcherException, IdcClientException, RemoteException;
```

This method will be called each time the action is invoked through the portlet framework. The method is passed in an ISCSActionForm, which is a bean that represents the parameters that are made available to this action.

This class will be of the type specified in the Action node.

The already initialized IdcClient object will be available to the action handler through the `getIdcClient()` method when inside the handleAction method, as will any other attributes specified on the Action node via the `getAttributes()` method. You may also access a unique ID for this handler via the `getID()` method. This can be used to store information on the session without conflicts.

The return type is an action result object. Usually, this is simply a container for the result parameters that are to be stored on the request for access within the JSP page. However, you may specify other parameters to this result, such as the view that should be used upon return. It defaults to `success` if you use the base class SCSActionResult.

Sample Action Handler

The action definition specifies the action `active.document.checkOut`, the ISCSActionHandler class that performs the action, the ISCSActionForm, which is a bean that represents the parameters passed into the handler, and the resulting tile that is displayed upon completion of the action.

```
<!--
Attempts to check out the specified document.
-->

<action
  name="active.document.checkOut"
  class="com.stellent.portlet.components.document.
```

```
      active.handlers.CheckOutHandler"
  bean="com.stellent.portlet.components.document.
       active.forms.CheckOutForm"
  authRequired="true" >

  <forward name="success" path="active.document.checkout.page"/>
</action>
```

The SCSActionForm code represents the parameters the checkout handler needs to complete its action. In the following example, checkout only requires the document ID of the document we are planning to check out:

```
public class CheckOutForm extends SCSActionForm {
  private String m_documentID;

  public String getDocumentID () {
    return m_documentID;
    }

  public void setDocumentID (String documentID) {
     m_documentID = documentID;
  }
}
```

The SCSActionHandler code first checks to see if the passed-in form is an instance of CheckOutForm. It errors out if this is not the case. Otherwise, it checks out the file through the RIDC API by calling the content server CHECKOUT service.
The resulting response object is put on the request by calling `result.setVariable(name,object)`. These objects will now be available to the JSP page rendering the view.

```
public class CheckOutHandler extends SCSActionHandler {
/**
* Checks out the specified content.
*
* @param portletRequest
* @throws com.stellent.portlet.dispatcher.PortletDispatcherException
*/
public ISCSActionResult handleAction (ISCSActionForm form, Object portletRequest)
throws PortletDispatcherException, IdcClientException, RemoteException {
ISCSActionResult result = new
SCSActionResult ();

// Checking of Form Instance.
if (form instanceof CheckOutForm) {
CheckOutForm cof = (CheckOutForm)form;

//The IdcClient Object and the Context.
IdcClient client = getIdcClient();
IdcContext ctx = SCSSession.getSCSContext(portletRequest);

// Content Server Service executed using RIDC API for Checking Out a document.
DataBinder chkoutBinder = client.createBinder ();
chkoutBinder.putLocal ("IdcService", "CHECKOUT");
chkoutBinder.putLocal ("dID", cof.getDocumentID ());
ServiceResponse response = client.sendRequest (ctx, chkoutBinder);
DataBinder checkoutBinder = response.getResponseAsBinder ();

// Service Executed for getting Document Information.
DataBinder docBinder = client.createBinder ();
docBinder.putLocal ("IdcService", "DOC_INFO");
docBinder.putLocal ("dID", cof.getDocumentID ());
ServiceResponse docresponse = client.sendRequest (ctx, docBinder);
DataBinder docDataBinder = docresponse.getResponseAsBinder ();
DataResultSet docData = docDataBinder.getResultSet ("DOC_INFO");
List docDataRows = docData.getRows ();
DataObject dobj = (DataObject)docDataRows.get(0);

//Result put on the request and sent to the JSP Page.
result.setVariable ("checkoutResponse", checkoutBinder);
result.setVariable ("infoResponse", dobj);
} else {
throw new PortletDispatcherException ("Unexpected form type,
expected 'CheckOutForm', got " + form);
}
return result;
}
```

A.9 Creating a Tile

A tile consists of a definition, an optional controller class, and a collection of JSP classes that will make up a portlet view. The definition section contains XML code that identifies the main layout JSP page.

The following example specifies three different regions that reference three JSP pages:

```
<%@ include file="/stellent/ui/fragment/jspimport.inc" %>
<scsportlet:insert name="header"/>
<scsportlet:insert name="content"/>
<scsportlet:insert name="footer"/>
```

The `include` at the top includes commonly defined imports and taglib definitions. This is an example of the file for the portlets:

```
<%@ page import="com.stellent.portlet.api.IPortletAPIFacade" %>
<%@ page import="com.stellent.portlet.api.PortletAPI" %>
<%@ include file="/stellent/ui/fragment/page.inc" %>
<%@ taglib uri="/WEB-INF/tlds/i18n.tld" prefix="i18n" %>
<%@ taglib uri="/WEB-INF/tlds/scsportlet.tld" prefix="scsportlet" %>
<%@ taglib uri="/WEB-INF/tlds/c.tld" prefix="c" %>
<%@ taglib uri="/WEB-INF/tlds/scs-databinder.tld" prefix="db" %>

<%
  //the api facade class
  IPortletAPIFacade apiFacade = PortletAPI.getInstance ().getPortletAPIFacade ();
%>
```

In the JSP page, three lines use the `insert` tag to tell the view to put the header first, the content next, and the footer last. For each region, the insert tag tells the framework to look up the definition and to include the JSP page specified by the insert tag.

A.10 Creating a Controller

A controller is a hook that allows the tile author to execute Java code before the tile itself is rendered. To create a controller, you need to implement the ISCSController class, which requires several methods that the portlet framework uses to control its lifetime.

In most cases, the abstract base class SCSController performs all the operations you need, with this one exception:

```
/**
  * Method is called before a Tile is rendered.
  *
  * @param portletRequest The portlet request that generated the Tile render.
  * @param portletResponse The portlet response associated with Tile render
  * @throws ServletException If a portlet container error occurs.
  * @throws IOException If a portlet container error occurs.
  * @throws IdcClientException If a RIDC framework error occurs.
  * @throws RemoteException If a RIDC communication error occurs.
  */

public void perform (Object portletRequest,
       Object portletResponse)
       throws ServletException, IOException, IdcClientException, RemoteException;
```

This method will get called immediately before the tile is rendered and any objects you place on the request will be available to the resulting JSP page.

Index