



BEA WebLogic Workshop™ Help

Version 8.1 SP2
November 2003

Copyright

Copyright © 2003 BEA Systems, Inc. All Rights Reserved.

Restricted Rights Legend

This software and documentation is subject to and made available only pursuant to the terms of the BEA Systems License Agreement and may be used or copied only in accordance with the terms of that agreement. It is against the law to copy the software except as specifically allowed in the agreement. This document may not, in whole or in part, be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine readable form without prior consent, in writing, from BEA Systems, Inc.

Use, duplication or disclosure by the U.S. Government is subject to restrictions set forth in the BEA Systems License Agreement and in subparagraph (c)(1) of the Commercial Computer Software–Restricted Rights Clause at FAR 52.227–19; subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227–7013, subparagraph (d) of the Commercial Computer Software—Licensing clause at NASA FAR supplement 16–52.227–86; or their equivalent.

Information in this document is subject to change without notice and does not represent a commitment on the part of BEA Systems. THE SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. FURTHER, BEA Systems DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE SOFTWARE OR WRITTEN MATERIAL IN TERMS OF CORRECTNESS, ACCURACY, RELIABILITY, OR OTHERWISE.

Trademarks or Service Marks

BEA, Jolt, Tuxedo, and WebLogic are registered trademarks of BEA Systems, Inc. BEA Builder, BEA Campaign Manager for WebLogic, BEA eLink, BEA Liquid Data for WebLogic, BEA Manager, BEA WebLogic Commerce Server, BEA WebLogic Enterprise, BEA WebLogic Enterprise Platform, BEA WebLogic Enterprise Security, BEA WebLogic Express, BEA WebLogic Integration, BEA WebLogic Personalization Server, BEA WebLogic Platform, BEA WebLogic Portal, BEA WebLogic Server, BEA WebLogic Workshop and How Business Becomes E–Business are trademarks of BEA Systems, Inc.

All other trademarks are the property of their respective companies.

Table of Contents

Developing Personalized Applications.....	1
Overview of Content Management.....	2
Creating Content.....	6
Overview of My Content Portlet.....	7
Setting Up My Content Portlet.....	8
Creating and Modifying Content with My Content Portlet.....	13
Searching for Content with My Content Portlet.....	17
Supporting Additional Mime Types.....	19
Setting up Users.....	21
Setting up Unified User Profiles.....	24
Creating User Profile Properties.....	34
Creating User Segments.....	37
Tracking Anonymous Users.....	38
Enabling Anonymous User Tracking.....	39
Creating Anonymous User Profiles.....	40
Designing Interaction Management.....	41
Creating Personalization Conditions.....	43
Creating User Profile Properties.....	45
Creating User Segments.....	48
Creating Session Properties.....	49
Creating Request Properties.....	51
Creating Catalog Structure Properties.....	53
Registering Custom Events.....	55

Table of Contents

Personalizing Portal Applications.....57

Creating Placeholders.....59

Creating Content Selectors.....61

Creating Discounts.....63

Creating Campaigns.....65

Preparing to Use Campaigns.....66

Building Campaigns.....67

Using Session, Request, and Event Properties in Campaigns.....71

Personalization Conditions Reference.....73

Developing Personalized Applications

WebLogic Portal provides powerful tools for building personalized portal applications. These interaction management tools let you develop personalization and campaigns.

Personalization and Campaigns – With personalization and campaigns, you can target users with personalized content and actions. Based on conditions such as user profile properties, user segment membership, HTTP session or request data, date/time conditions, or events, each user is dynamically served personalized Web content, automatic e-mails, and discounts with pinpoint accuracy.

Steps for Adding Interaction Management to Your Applications

Interaction management development involves setting up interrelated pieces. The following sections describe the steps needed to implement interaction management functionality. Each section contains links to different implementation tasks depending on your needs. Use this topic as an overall roadmap for developing personalized applications.

1. Overview of Content Management
2. Setting up Users
3. Designing Interaction Management
4. Creating Personalization Conditions
5. Personalizing Portal Applications

Related Topics

Portal JSP Tags

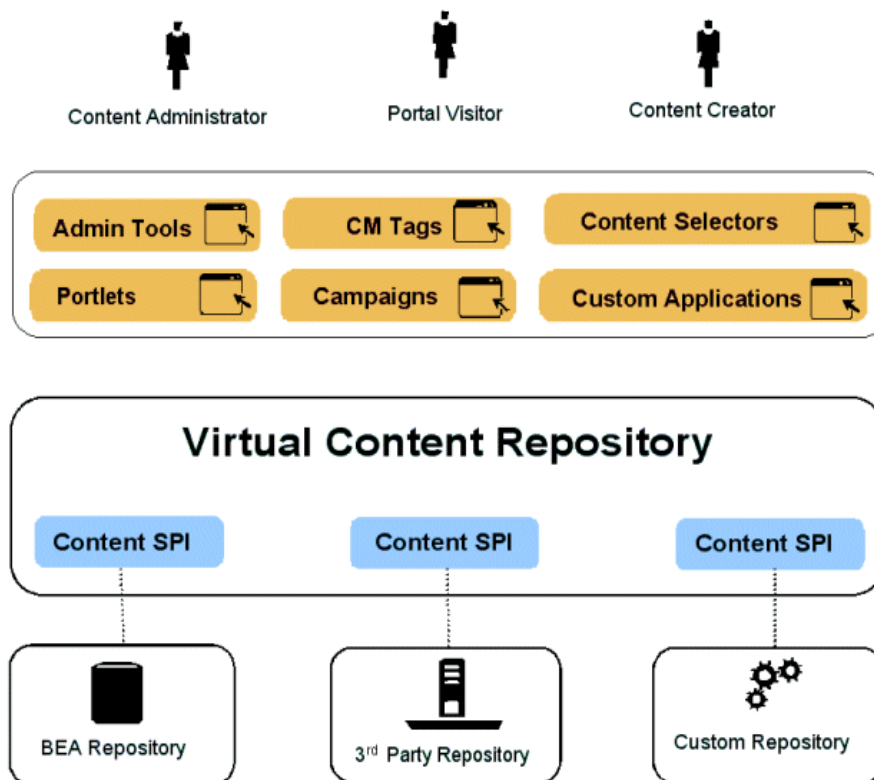
Overview of Content Management

The content you want to show users, whether it is a single line of text, an HTML file, a graphic, or an animation file can be stored in a content repository. BEA's Virtual Content Repository, included with WebLogic Portal, provides a single interface that lets you store content in BEA repositories as well as seamlessly incorporate BEA-compatible third-party content management systems. This overview provides information on the following subjects:

- The Virtual Content Repository
- Content Hierarchy
- Content Types
- Creating and Modifying Content
- Using Content in Personalized Applications

The Virtual Content Repository

The Virtual Content Repository can contain multiple content repositories. It provides services such as federated search (a search that returns a result set from all the relevant content across the plugged in repositories), content lifecycle management, Delegated Administration and content type management. Many Portal subsystems interact with the Virtual Content Repository. Content Management tags execute queries to deliver dynamic content to end users. Content Selectors and Campaigns deliver dynamic, personalized content to user based upon personalization rules or conditions.



The Content Hierarchy

WebLogic Portal Content Management is organized hierarchically. The Virtual Content Repository (VCR) is the top-level node in the content management system. Repositories are the immediate children of the VCR. These repositories can be made up of multiple BEA Systems repositories, multiple third-party repositories, or custom content repositories.

Hierarchy Nodes and Content Nodes comprise the next level of the hierarchy tree and are organized much like a file system. Hierarchy Nodes can contain both Hierarchy Nodes and Content Nodes. Content Nodes can only contain other Content Nodes. Nodes can be created based upon Content Types. For example:

Virtual Content Repository

Repository 1

Hierarchy Node

ContentNode (index.htm)

ChildContent1 (logo.gif)

ChildContent2 (photo.jpg)

Content Repositories provide the storage mechanism for content, and they comprise the second-level of the Virtual Content Repository hierarchy. Content Repositories may include multiple instances of BEA repositories, 3rd party repositories, or customer repositories. To plug into the Virtual Content Repository, you must implement the BEA Content Management Service Provider Interface the CM SPI.

Hierarchy Nodes are organizational mechanisms that help you organize and group content in the hierarchy, much like folders in a file system. Hierarchy Nodes can contain other Hierarchy Nodes as well as Content Nodes. They can also be typed so that they function similarly to Content Nodes.

Content Nodes represent content stored in the repository. A complete content node comprises a set of data property values defined by a content type. This data structure may include files such as a word processing document, HTML file, spreadsheet or image. It may also include metadata such as the author, version number or summary. Content Nodes can also have child Content Nodes. For example, The Content Node for an HTML document may have child Content Nodes for the images used by the HTML document.

Content Types

Content Types define the set of properties that make up a Content Node or Hierarchy Node. This may include any combination of the supported data types, such as date and time, number, text (string), Boolean (true/false), or binary (file).

For example, the Content Type for image content may have a number property "width" and a number property "height," while the Content Type for news article content may have a text property "Author", a text property "Summary", a date property "Published Date", and a binary property "Article" for a file containing the formatted article. Types do not have to include a binary, although a common example of a type is a single binary with a set of non-binary properties that describe the document.

Repository 1

Content Type 1

Property 1 = Binary

Property 2 = String

Content Type 2

Content Types also define the available values for a given property, including whether it can contain multiple values. For example, a property called "Priority" may only allow a single choice among the values "High", "Medium", and "Low", while a property called "Favorite Color" may allow multiple pre-defined values to be chosen.

Each repository has its own set of content types. You can create types in BEA repositories and third-party repositories that support this feature.

Creating and Modifying Content

After you connect a BEA-compatible content management system to the Virtual Content Repository you can continue to add and modify content directly in your BEA-compatible content management system. Changes appear automatically in the Virtual Content Repository. You can create and manage content in the Administration Portal, in the My Content Portlet, or with the bulkloader. For more information, see "Creating Content."

Using Content in Personalized Applications

WebLogic Workshop extensions support development of personalized applications, while the WebLogic Administration Portal enables portal administrators to adapt site interaction to fit the needs of the audience. The core of the Personalization system is the underlying rules engine that matches users with appropriate content. Content Selectors, Placeholders and Campaigns are the aspects of content management visible to administrators. Also, User Segments contain the criteria that define the target visitor, such as gender or browser type.

The Content Management component provides the run-time API by which content is queried and retrieved. The functionality of this component is accessible via tags. The content retrieval functionality is provided using either the provided reference implementation or third-party content retrieval products.

Related Topics

Creating Content

Setting up Users

Designing Interaction Management

Creating Personalization Conditions

Personalizing Portal Applications

Creating Content

After you connect a BEA-compatible content management system to the Virtual Content Repository you can continue to add and modify content directly in your BEA-compatible content management system. Changes appear automatically in the Virtual Content Repository. You have three primary options for creating content for your portal:

- ***WebLogic Administration Portal***

BEA's Virtual Content Repository, available in the WebLogic Administration Portal under "Content Management," lets you add and modify content nodes and assign types (metadata) to those nodes. Through these tools, you can manage content in all the repositories plugged into the Virtual Content Repository that support management functions.

- ***Bulkloader***

The BulkLoader is a command-line application that is capable of loading document metadata into the reference implementation database from a directory and file structure. The BulkLoader parses the document base and loads all the document metadata so that the Content Management component can search for documents.

- ***My Content Management Portlet***

My Content portlet provides you with tools to manage your content in the BEA Virtual Content Repository. You can create, update, and delete content directories and nodes as well as browse content hierarchies and search for content.

Related Topics

Creating Campaigns

Creating Users and Groups

Delegated Administration

Overview of My Content Portlet

My Content portlet provides you with tools to manage your content in the BEA Virtual Content Repository. You can create, rename, update, and delete content directories and nodes as well as browse content hierarchies and search for content. My Content portlet supports delegated administration, letting users view and manage only the content nodes delegated to them.

Setting Up My Content Portlet

My Content portlet must be set up in a portal, and you must have delegated administration rights before you can use it to manage your content.

Creating and Managing Content with My Content Portlet

The My Content portlet supports full create, read, update, and delete (CRUD) operations for content. You can edit content properties and create content (directory nodes and content nodes). You cannot create new content types in the My Content portlet.

You can have multiple binary properties for a content node. For placeholders and content selectors, only the binary marked as the primary property is displayed.

Searching Content

In the My Content portlet, you can search for content in the virtual repository.

Related Topics

[Content Management Overview](#)

[Creating a New BEA Content Repository](#)

[Portal Samples](#)

Setting Up My Content Portlet

To make My Content portlet available for administrators to create and manage content, you must perform the following tasks:

- Step 1: Add the Portlet to a Portal Application
- Step 2: Add the Portlet to a Page in your Portal
- Step 3: Set Up User Access Rights to Content

Step 1: Add the My Content Portlet to a Portal Project (Web Application)

Before you begin this process, it is assumed that you already have a portal application with a portal project (web application) that it is running on a WebLogic Server. For more information on creating a portal application, see Creating a Portal Application and Portal Web Project.

1. Start WebLogic Workshop and open your application.
2. Create a directory in your portal project titled `portlets`, if one does not already exist.
3. Right-click on your `portlets` folder and choose **Import**, or click the `portlets` folder and choose **File -> Import Files**.
4. In the **Import Files to Project** dialog, navigate to the folder called **content**.

Import or copy this	to this directory (create if necessary)
<WEBLOGIC_HOME>\samples\portal\portalApp\sampleportal\portlets\content\	<PORTAL_APP>\<project>\portlets\

5. With the **content** folder selected, click **Import**.

Note: The My Content portlet requires that users be logged in to determine the rights each user has on certain content. Consider importing the Login to Portal portlet as well. To do this, import the folder `login` and the file `includes/Login.portlet` from the same location as the `content` folder above.

Step 2: Add the Portlet to a Page in your Portal

1. In WebLogic Workshop, Add the My Content portlet to a page in your .portal based Portal.
Or
In the Administration Portal, add the My Content portlet to a page in your streaming Portal.

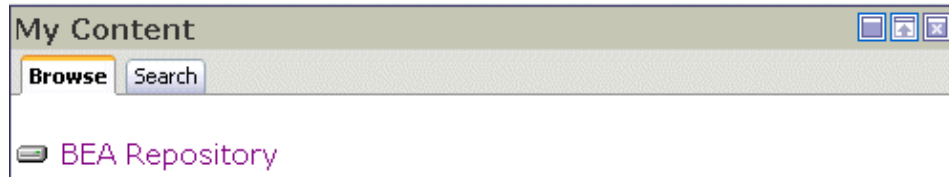
Note: The .portal file you create in WebLogic Workshop is a template. In this template you create books, pages and portlets and define defaults for them. When you view the .portal file with your browser the portal is rendered in "single file mode," meaning that you are viewing the portal from your file system as opposed to a database.

The .portal file's XML is parsed and the rendered portal is returned to the browser. The creation and use of a .portal is intended for development purposes. Because there is no database involved you cannot take advantage of things such as user customization. Once you have created a .portal file you can use it to create desktops for a production environment.

2. View your portal with the WebLogic Test Browser or with your default browser. Login to the portal as **webLogic**, **webLogic** so you can see the portlet.
 - ♦ **WebLogic Test Browser** – In the WebLogic Workshop toolbar, click the **Start** button (or

press **Ctrl+F5**).

- ◆ **Default Browser** – In the WebLogic Workshop menu, choose **Portal-->Open Current Portal**.



Note: Only the BEA Repository is visible until you add content.

Step 3: Set Up User Access Rights to Content

You can set up user access to the My Content Portlet two ways:

- Set Up Delegated Administration rights in the Administration Portal
- Set Up Portlet Preferences

Set Up Delegated Administration Rights in the Administration Portal

Delegated Administration provides a way for WebLogic Administration Portal to propagate privileges down a hierarchy of roles. A Delegated Administration role is a dynamic classification of users based on user name, group membership or by the user's characteristics (or expressions), such as user profile values or time.

The rights you set up in the Administration Portal determine the content users can see and manage in My Content Portlet. Users must be in the PortalSystemAdministrators group to manage content with My Content portlet, and you can set this up two ways:

- If you create a Delegated Administration role based on users, or a user group, the role members are automatically added to the PortalSystemAdministrators group.
- If you define a Delegated Administration Role using only an expression, such as a date or user profile properties, you may need to manually add users to the PortalSystemAdministrator group.

To set up Delegated Administration Rights for content in the Administration Portal you will follow this general process:

1. Create the Delegated Administration Role in the Administration Portal.
2. Apply the role to a node/content using the Content Mananagement tools in the Administration Portal.

Note: To grant Delegated Administration authority in the Administration Portal, be sure you select Can Manage for the content node to empower users with access to see and manage all of the content below the selected content node.

Set Up Portlet Preferences

This section contains information on the following subjects:

- What is a Portlet Preference?
- What Kind of Preferences Can I Set Up
- Steps for Setting Up Preferences

What is a Portlet Preference?

A *portlet preference* is a property in a portlet that can be customized by either an administrator or a user. The My Content Portlet provides you with a way to grant users access to content by using a special Portlet Preference called `cm_homefolders`. You can modify values of the `cm_homefolders` Portlet Preference using either the Administration Portal or WebLogic Workshop.

For example, your company provides each employee with his or her own private space for content on the corporate network. If you've got 10,000 employees, setting up Delegated Administration rights on each of those 10,000 folders individually would be very time consuming. The `cm_homefolders` Portlet Preference provides a way to quickly grant rights to content based on string substitution of users' username, group names, or role names.

If each user's private content area is represented by a folder that matches their username:

Corporate Repository

Employees

&
jan_h
jane_d
joe_p
john_d
&

To grant each user access to their folder, you add a value to the `cm_homefolders` Portlet Preference like `/Corporate Repository/Employees/%username%`. Now when *jan_h* logs in, she will have access to `/Corporate Repository/Employees/jan_h` and when *joe_p* logs in he will have access to `/Corporate Repository/Employees/joe_p`.

What Kind of Preferences Can I Set Up?

The My Content portlet supports three primary types of string substitution variables:

- Users (`%username%`)
- Groups (`%groupname%`)
- Roles (`%rolename%`)

The `%username%` variable can be used to provide individual users access to the appropriate part of the content directory. The access is granted by adding `%username%` to a path in a repository. For example, `/BEA Repository/%username%`. If **BEA Repository** contains immediate children named Bob and Jane, the user named **Jane** would have access to `/BEA Repository/Jane` and its children while the user **Bob** would have access to `/BEA Repository/Bob` and its children.

Note: Users and groups are scoped to the users and groups that are set up in the Administration Portal. Roles are scoped to visitor entitlements.

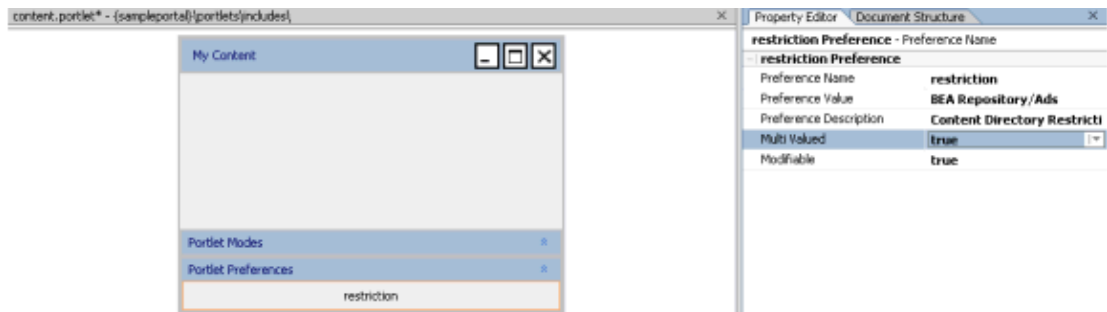
Steps for Setting Up Preferences

You can set up portlet preferences in the Administration Portal or you can start the process in the Administration Portal and finish it in WebLogic Workshop.

Setting up Preferences in WebLogic Workshop

Use the following instructions to set up portlet preferences:

1. In the Administration Portal, create the necessary Users and Groups,
OR
Create the necessary Visitor Entitlement Roles.
2. Set up nodes for the Users, Groups, or Entitlements in the Content Management Hierarchy in the Administration Portal.
3. In WebLogic Workshop, set up the Preferences to point to the nodes you created: (%username%, %groupname%, %rolename%)
 - a. Open the portlet called content.portlet in the sampleportal\portlets\includes directory. The portlet appears in the Portlet Designer.
 - b. Drag a **New Preference** from the Palette window and drop it on the body of the portlet.
 - c. In the Portal Designer, expand the **Portlet Preferences** bar and select **New Preference**.
 - d. In the Property Editor, enter preference values.
4. After this restriction is set, the portlet will have access starting only at the path defined for both browsing and searching. You can create additional restrictions by creating additional portlet preferences using the previous steps. When you use multiple path restrictions, only those paths are available for browsing and searching.



Setting up Preferences in the Administration Portal

Use the following instructions to set up portlet preferences:

1. In the Administration Portal, select the My Content portlet in the **Library Portal Resources** -> **Library** -> **All Portlets** -> **My Content**.
2. Select the **Portlet Preferences** tab.
3. Click **Edit** next to the cm_homefolders Portlet Preference.
4. Enter a new value in the **Add new value:** box, such as /BEA Repository/employees/%username%.
5. Click **Add** for each new value that you want to add to the Portlet Preference.
6. Click **the trash can icon** to remove any unwanted values.
7. Click **Save Portlet Preference** to save your changes.

Developing Personalized Applications

NOTE: Do **NOT** to change the **Preference Name** field. Make sure that **Is Multi-Valued** is checked on. If you wish to make sure that all occurrences of this portlet use your new values, check **Propagate value(s) to all instances of this Portlet** before saving.

For more information about portlet preferences in the Administration Portal, see [Creating Portlet Preferences](#).

Related Topics

[My Content Portlet Sample](#)

[Creating Content with My Content Portlet](#)

[Searching with My Content Portlet](#)

Creating and Modifying Content with My Content Portlet


The My Content Portlet provides you with a way to manage content directly from a portal. The My Content Portlet does not provide as much flexibility or power as the Administration Portal's content management tools, but it does allow you to create, rename, update, and delete content and folders.

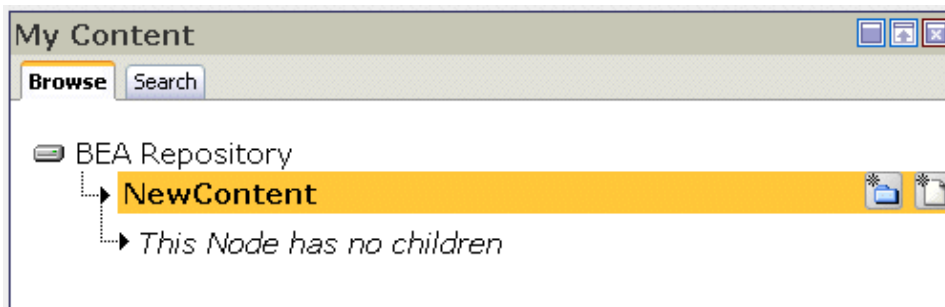
To get access to the My Content portlet, you need delegated administration rights on content or rights granted by the cm_homefolders portlet preference. You must also be logged in to use the portlet. For complete setup instructions, see "Setting Up My Content Portlet."


- Creating Content
- Modifying Content

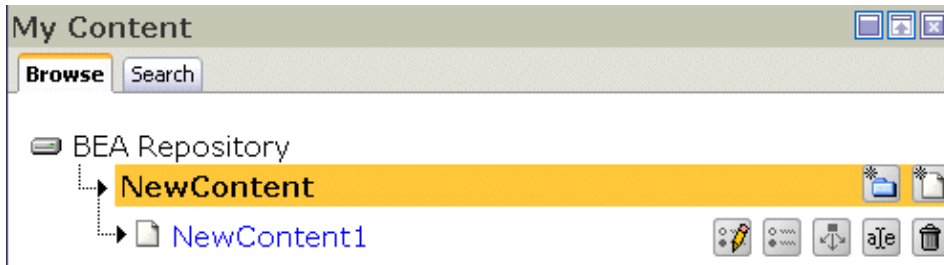
Creating Content

To create content with the My Content portlet:

1. Login to the Portal that contains the My Content portlet.
2. Navigate to the My Content portlet. The content for which you have security rights will be visible in the portlet.
3. Navigate to the node under which you want to create new content.
4. Create a new folder (node) to hold the new content if one does not already exist:
 - a. Click the **Create New Folder icon**. 
 - b. Name the new folder.
 - c. Click **Create**.



5. Create the content:
 - a. Click the **Create Content icon**. 
 - b. Enter the Name of the content (can be anything)
 - c. Select a type of content from the drop-down menu.
 - d. Click **Create**.



6. Click the **Edit Properties icon** for the new content item.
7. Update values for the properties. The editable properties vary depending on the type of content you selected.

You cannot change the content types or add new content types in the My Content Portlet, but you can update and add values for existing properties. For information about creating content types (schemas), see the Content Management Documentation for the BEA WebLogic Administration Portal.

Edit Properties of "NewContent1"	
Property Name	Property Value(s)
height:	<input type="text"/>
adMap:	<input type="text"/>
adTargetUrl:	<input type="text"/>
adClickTarget:	<input type="text"/>
adWinTitle:	<input type="text"/>
adAltText:	<input type="text"/>
adWinClose:	<input type="text"/>
content:	Replace with: <input type="text"/> <input type="button" value="Browse..."/>

8. Save your changes.
9. To view your content, click the View Properties icon.

Managing Content

You can manage content with My Content Portlet in the following ways:

- Manage Properties
- Browse Children
- Rename a Content Node or Folder
- Delete a Content Node or Folder

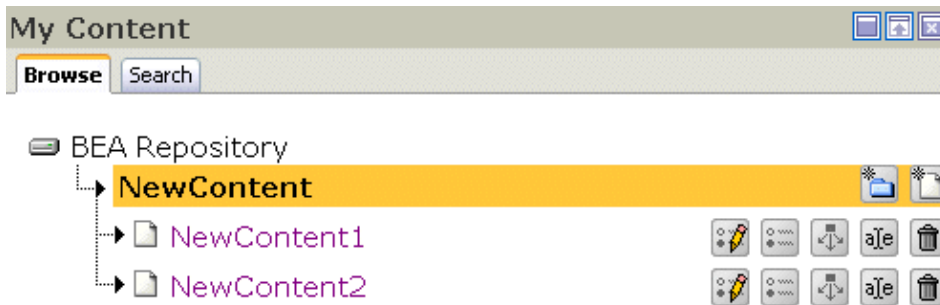
Manage Properties



The My Content portlet allows you to view and edit content property values.

Note: If you want to change the properties themselves, see Content Management for the BEA WebLogic Administration Portal.

To view and edit content node properties:

1. In the Content tree, double click the **content folder** that holds the content whose properties you want to update.




2. Select the **View Properties icon**  to see the properties associated with this content.
or
Select the **Edit Properties icon**  to open the property editor.
3. Update the **properties**, and save your changes.

Note: The editable properties vary depending on the type of content.

Browse Children


A "child" is a dependent of another content node. The My Content Portlet allows you to quickly list the child nodes of a selected Content Node.

1. Select a **Content Node**.
2. Select the **Browse Children Icon**. 

Rename a Content Node or Folder

You can rename a Content node or folder.


Warning: Carefully consider and test the effects that renaming nodes will have on other users of your system!

1. Select a **Content Node** or **Folder** to display its children nodes.
2. Select the **Rename icon** next to the child node you want to rename. 
3. Enter the new name, and select **Rename**.

Delete a Content Node or Folder

You can quickly and easily delete a Content Node or Folder. If you delete a parent node, all of the child nodes will be deleted as well.

Developing Personalized Applications

1. Select a ***Content Node*** or ***Folder*** to display its children nodes.
2. Select the ***Delete icon*** next to the child node you want to delete. 

Related Topics

Browsing and Searching with My Content Portlet

Creating Content

Overview of Content Management

My Content Portlet Sample

Searching for Content with My Content Portlet

You can use the portlet to search the Virtual Content Repository.

1. In My Content Portlet, select the *Search tab*.

The screenshot shows the 'My Content' portlet interface. At the top, there are two tabs: 'Browse' and 'Search', with 'Search' being the active tab. Below the tabs, there is a search configuration area. It includes a 'Content Type' dropdown menu set to 'Standard'. Below that is an 'Expression' field with a dropdown set to 'cm_createdBy' and a comparison operator dropdown set to 'Equals'. To the right of the comparison operator is an empty text input field. Below these is a 'Multiple Expressions' section with a dropdown set to 'And' and the text 'this expression'. There is an 'Add Expression' button below this section. At the bottom of the configuration area is a large text input field labeled 'Search Expression(s):'. Below this field is a 'Search' button.

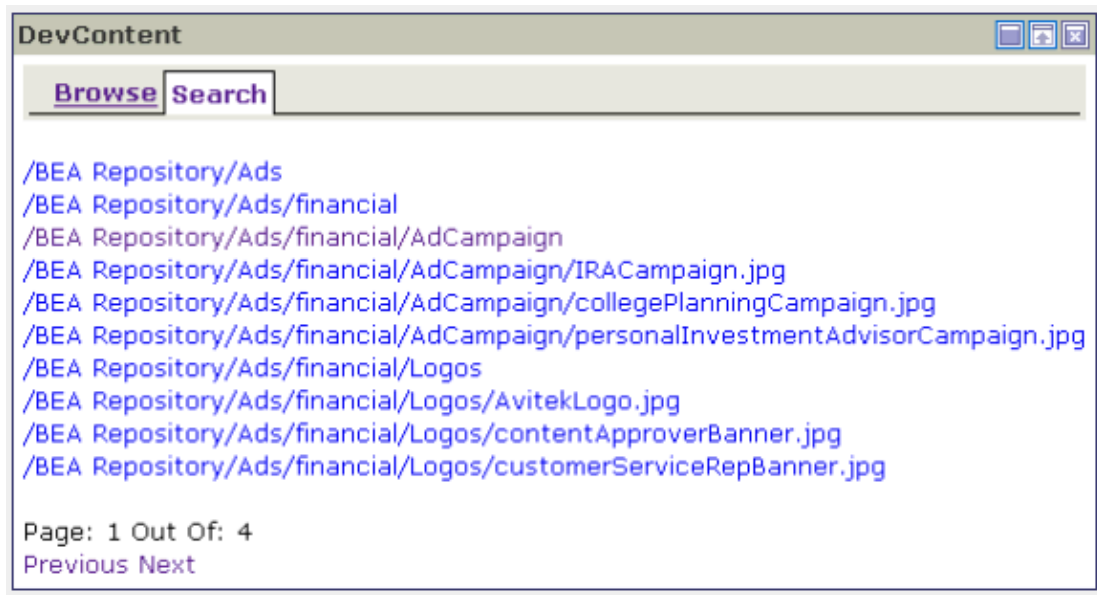
2. Set the Search Expression.

For example, the search expression `cm_path` like `*Ads*` brings back any node with the string `Ads` in its path. You can develop a search query based on comparisons with values from multiple types. The task order for building a query includes:

- a. Select the *Content Type*.
- b. Select the *Property*.
- c. Select the *Conditional* value.
- d. Type the *comparison value*, for example, `*Ads*`.
- e. Select *Add Expression*.
- f. Click *Search*.

NOTE: If you want a more complex expression, repeat steps a through e and choose `And` or `Or` from *Multiple Expressions* before clicking *Add Expression*.

3. When the results display, you can then select the link to see the content in the Browse tab.



Related Topics

Setting Up My Content Portlet

Creating and Modifying Content with My Content Portlet

Portal Samples

Supporting Additional Mime Types

To display content, Placeholders refer to a document's MIME type and then generate the HTML tags that a browser requires for the specific document type. For example, to display an image-type document, an ad placeholder must generate the tag that a browser requires for images. By default, ad placeholders can generate the appropriate HTML only for the following MIME types:

- HTML, XML, plain text – For this type of content, a Placeholder passes the text directly to the JSP.
- Images – For this type of content, a Placeholder generates an tag with attributes that the browser needs to display the image. If you want images to be clickable, you must specify the target URL and other link-related information as content properties in the Virtual Content Repository.
- Shockwave files – For this type of content, a Placeholder generates the <OBJECT> tag, which Microsoft Internet Explorer on Windows uses to display the file, and the <EMBED> tag, which browsers that support the Netscape-compatible plug-in use to display the file. In Virtual Content Repository you can specify attributes for the <OBJECT> and <EMBED> tags.

If you are familiar with basic Java programming, you can write classes that enable placeholders to generate HTML for additional MIME types. To support additional MIME types, you must complete the following tasks:

Create and Compile a Java Class to Generate HTML

Register the New Class

Create and Compile a Java Class to Generate HTML

To generate the HTML that the browser requires to display the MIME type, use WebLogic Workshop to create a Java Project in your application, then create and compile a Java class in that Java Project that implements the `com.bea.p13n.ad.AdContentProvider` interface. For information on this interface, see the WebLogic Portal Javadoc.

After you compile the class, make sure the class is available to the application. One way to do this is to add the class appropriately to one of the deployed JAR files, such as `p13n_ejb.jar` or your own jar file. Another way to make the class available to the application is to save it under a directory that is specified in the system's CLASSPATH environment variable. For example, create a <PORTAL_APP>\classes directory and add it to the set-environment script.

Register the New Class

After you save the class in a directory that is in your classpath, you must notify WebLogic Portal of its existence:

1. Stop the server.
2. Create a backup copy of your application's META-INF\application-config.xml file.
3. Open application-config.xml in a text editor and find the <AdService> element.
4. Add the following as a subelement of <AdService>:

```
<AdContentProvider
  Name="MIME-type"
  Provider="YourClass.class"
  Properties="optional-properties-for-your-class"
```

Developing Personalized Applications

```
>  
</AdContentProvider>
```

Provide the following values for the attributes of the AdContentProvider element:

Name – The name of the MIME type that you want to support.

Provider – The name of the compiled Java file. If you saved the file below a directory that your CLASSPATH environment variable names, you must include the file's pathname, starting one directory level below the directory in classpath.

Properties – Any additional properties or parameters want to pass to your object. For example, if you added <PORTAL_APP>/classes to the system classpath, save your class to support AVI files as <PORTAL_APP>/classes/myclasses/MimeAvi.class.

For example:

```
<AdContentProvider  
Name="video/x-msvideo"  
Provider="myclasses.MimeAvi"  
Properties=" "  
>  
</AdContentProvider>
```

5. Save your modifications to application-config.xml.
6. Restart the server.
7. Activate the content provider in the WebLogic Administration Portal Service Administration tools.
 - a. Launch the WebLogic Administration Portal by entering the following URL in a browser:
http://<server>:<port>/<portal_app>Admin. For example,
http://localhost:7001/myPortalAppAdmin. Log in with a portal or system administrator username and password. The default login is weblogic/weblogic.
 - b. In the WebLogic Administration Portal, click **Service Administration**.
 - c. In the left resource tree, click **Add/remove configurable item**.
 - d. In the list of configurable items that appears, select the checkbox next to the content provider you added.
 - e. Click **Update** at the bottom of the window.

Related Topics

Creating Placeholders

Developing Personalized Applications

Setting up Users

For the purposes of interaction management development, users are people who visit your portals. (System and portal administrators are also users.) Users can be authenticated (logged in) or they can be anonymous visitors. Either way, you can target users with personalized content and campaigns and capture their actions in your portals with behavior tracking.

There are different ways of setting up or identifying users in your portals. You are likely to use many of these ways:

- Adding Users with the WebLogic Administration Portal
- Adding Users with the WebLogic Administration Console
- Letting Users Add Themselves
- Enabling Anonymous User Tracking
- Handling Anonymous Users

Adding Users with the WebLogic Administration Portal

You can add users and organize them into groups in your domain using the WebLogic Administration Portal.

For instructions on creating users and groups, see the WebLogic Administration Portal help system.

To access the WebLogic Administration Portal:

- **On the production server:** With the server running, enter the following URL in a browser: `http://<server>:<port>/<portalApp>Admin`. For example, `http://localhost:7001/greatAppAdmin`.
- **On your development server:** You can use the same method as used on the production server, or you can use WebLogic Workshop. With your portal application open and the development server running, choose **Portal-->Open Portal Administration**.

Adding Users with the WebLogic Administration Console

You can add users and organize them into groups in your domain using the WebLogic Administration Console. Add users this way for convenience if you work frequently in the WebLogic Administration Console rather than in the WebLogic Administration Portal.

For instructions on creating users and groups, see Security in the WebLogic Administration Console help system documentation.

To access the WebLogic Administration Console:

- **On the production server:** With the server running, enter the following URL in a browser: `http://<server>:<port>/console`. For example, `http://localhost:7001/console`.
- **On your development server:** You can use the same method as used on the production server, or you can use WebLogic Workshop. With the development server running, choose **Tools-->WebLogic Server-->WebLogic Console**.

Letting Users Add Themselves

Developing Personalized Applications

You can build functionality into your portal application that lets users add themselves to the domain. Use any of the following features:

- `<um:createUser>` – The WebLogic Workshop Portal Extensions `<um:createUser>` JSP tag lets you add users to the domain.
- Create User Control – The WebLogic Workshop Portal Extensions Create User Control lets you add user–creation functionality to Java Page Flows and surface that functionality in portlets.
- WebLogic Portal API – The WebLogic Workshop Portal Extensions API provides user management classes for creating users.

Enabling Anonymous User Tracking

WebLogic Portal lets you identify and retain information about non–authenticated visitors to your portals. When you enable anonymous user tracking, non–authenticated users receive a cookie after a predetermined time (30 seconds by default), and preference information is persisted in a database rather than in memory.

Each time an anonymous tracked user returns to the portal, the ID in their cookie matches the primary key in the tracked anonymous user database, their previous user properties are maintained, and your personalization and campaign rules will work for those users. When anonymous tracked users register in your portal, their user profile is moved from the anonymous tracked user database to the user database.

If users do not have cookies enabled or if they delete cookies frequently, there is no way to match the users with their existing records in the anonymous tracked user database, and on subsequent returns to the portal these users are treated as new anonymous users.

See Tracking Anonymous Users for instructions.

Handling Anonymous Users

You can target users with personalized content and campaigns and track their behavior in your portals even if they remain anonymous; that is, even if they have not logged in or if you do not have anonymous user tracking enabled.

The main difference between completely anonymous users and authenticated/anonymous tracked users is how long their profile information is retained.

For example, if an anonymous user visits a portal and sets her preferences to "favorite color=purple" and "favorite hobby=reading," those values are persisted in memory and can be used to display personalized content and trigger campaigns while the browser session lasts. However, if the user closes the browser and revisits the portal, she has to re–enter her user preferences. Registered and anonymous tracked users have their preferences saved in a database.

While anonymous users do not retain a consistent store of user preferences from session to session, you can still provide a fair amount of interaction management functionality. For example, many personalization and campaign conditions are based on generic HTTP session and request properties you define, dates and times, and events that have no relationship to user profile properties. For example, you could create a successful campaign based on users accessing your portal with a specific type of browser during a specific timeframe.

Related Topics

Developing Personalized Applications

Setting up Users

Developing Personalized Applications

Overview of Content Management

Designing Interaction Management

Creating Personalization Conditions

Personalizing Portal Applications

Setting up Unified User Profiles

A Unified User Profile provides the capability to leverage user data from external sources such as LDAP servers, legacy systems and databases. This allows for the use of a single profile to access user data from many different sources.

Note: If you are using WebLogic's default user store, unified user profile functionality is already configured.

To create a UUP to retrieve user data from external sources, complete the following tasks:

Create an EntityPropertyManager EJB to Represent External Data

Deploy a ProfileManager That Can Use the New EntityPropertyManager

Retrieving User Profile Data from LDAP

Create an EntityPropertyManager EJB to Represent External Data

To incorporate data from an external source, you must first create a stateless session bean that implements the methods of the `com.bea.p13n.property.EntityPropertyManager` remote interface. `EntityPropertyManager` is the remote interface for a session bean that handles the persistence of property data and the creation and deletion of profile records.

In addition, the stateless session bean should include a home interface and an implementation class. For example:

```
MyEntityPropertyManager  
extends com.bea.p13n.property.EntityPropertyManager
```

```
MyEntityPropertyManagerHome  
extends javax.ejb.EJBHome
```

Your implementation class can extend the `EntityPropertyManagerImpl` class. However the only requirement is that your implementation class is a valid implementation of the `MyEntityPropertyManager` remote interface. For example:

```
MyEntityPropertyManagerImpl extends  
com.bea.p13n.property.internal.EntityPropertyManagerImpl
```

or

```
MyEntityPropertyManagerImpl extends  
javax.ejb.SessionBean
```

Recommended EJB Guidelines

We recommend the following guidelines for your new EJB:

- Your custom `EntityPropertyManager` is not a default `EntityPropertyManager`. A default

EntityPropertyManager is used to get/set/remove properties in the Portal schema. Your custom EntityPropertyManager does not have to support the following methods. It can throw `java.lang.UnsupportedOperationException` instead:

- ◆ `getDynamicProperties()`
- ◆ `getEntityNames()`
- ◆ `getHomeName()`
- ◆ `getPropertyLocator()`
- ◆ `getUniqueId()`
- If you want to be able to use the portal framework and tools to create and remove users in your external data store then you must support the `createUniqueId()` and `removeEntity()` methods. However, your custom EntityPropertyManager is not the default EntityPropertyManager so your `createUniqueId()` method does not have to return a unique number. It must create the user entity in your external data store and then it can return any number, such as `-1`.
- The following recommendations apply to the EntityPropertyManager() methods that you must support:
 - ◆ `getProperty()` – Use caching. You should support the `getProperties()` method to retrieve all properties for a user at once, caching them at the same time. Your `getProperty()` method should use `getProperties()`.
 - ◆ `setProperty()` – Use caching.
 - ◆ `removeProperties()`, `removeProperty()` – After these methods are called, a call to `getProperty()` should return null for the property. Remove properties from the cache too.
- Your implementations of the `getProperty()`, `setProperty()`, `removeProperty()`, and `removeProperties()` methods must include any logic necessary to connect to the external system.
- If you want to cache property data, the methods must be able to cache profile data appropriately for that system. (See the `com.bea.p13n.cache` package in the WebLogic Portal Javadoc at <http://edocs.bea.com/wlp/docs81/javadoc/index.html>.)
- If the external system contains read-only data, any methods that modify profile data must throw a `java.lang.UnsupportedOperationException`. Additionally, if the external data source contains users that are created and deleted by something other than your WebLogic Portal `createUniqueId()` and `removeEntity()` methods can simply throw an `UnsupportedOperationException`.
- To avoid class loader dependency issues, make sure that your EJB resides in its own package.
- For ease of maintenance, place the compiled classes of your custom EntityPropertyManager bean in your own JAR file (instead of modifying an existing WebLogic Portal JAR file).

Before you deploy your JAR file, follow the steps in the next section.

Deploy a ProfileManager That Can Use the New EntityPropertyManager

A "user type" is a mapping of a ProfileType name to a particular ProfileManager. This mapping is done in the UserManager EJB deployment descriptor.

To access the data in your new EntityPropertyManager EJB, you must do *one* of the following:

- In most cases you will be able to use the default deployment of ProfileManager, the UserProfileManager. You will modify the UserProfileManager's deployment descriptor to map a property set and/or properties to your custom EntityPropertyManager. If you support the `createUniqueId()` and `removeEntity()` methods in your custom EntityPropertyManager, you can use WebLogic Administration Portal to create a user of type "User" with a profile that can get/set properties using your custom EntityPropertyManager. For more information, refer to Modifying the

Existing ProfileManager Deployment Configuration.

- In some cases you may want to deploy a newly configured ProfileManager that will be used instead of the UserProfileManager. This new ProfileManager is mapped to a ProfileType in the deployment descriptor for the UserManager. If you support the createUniqueId() and removeEntity() methods in your custom EntityPropertyManager, you can use the WebLogic Administration Portal (or API) to create a user of type "MyUser" (or anything else you name it) that can get/set properties using the customized deployment of the ProfileManager that is, in turn, configured to use your custom EntityPropertyManager. For more information, refer to Configuring and Deploying a New ProfileManager.

ProfileManager is a stateless session bean that manages access to the profile values that the EntityPropertyManager EJB retrieves. It relies on a set of mapping statements in its deployment descriptor to find data. For example, the ProfileManager receives a request for the value of the "DateOfBirth" property, which is located in the "PersonalData" property set. ProfileManager uses the mapping statements in its deployment descriptor to determine which EntityPropertyManager EJB contains the data.

Modifying the Existing ProfileManager Deployment Configuration

If you use the existing UserProfileManager deployment to manage your user profiles, perform the following steps to modify the deployment configuration.

Under most circumstances, this is the method you should use to deploy your UUP. An example of this method is the deployment of the custom EntityPropertyManager for LDAP property retrieval, the LdapPropertyManager. The classes for the LdapPropertyManager are packaged in ldapprofile.jar. The deployment descriptor for the UserProfileManager EJB is configured to map the "ldap" property set to the LdapPropertyManager. The UserProfileManager is deployed in p13n_ejb.jar.

1. Back up the p13n_ejb.jar file in your enterprise application root directory.
2. From p13n_ejb.jar, extract META-INF/ejb-jar.xml and open it for editing.
3. In ejb-jar.xml, find the <env-entry> element, as shown in the following example:

```
<!-- map all properties in property set ldap to ldap server -->
<env-entry>
  <env-entry-name>PropertyMapping/ldap</env-entry-name>
  <env-entry-type>java.lang.String</env-entry-type>
  <env-entry-value>LdapPropertyManager</env-entry-value>
</env-entry>
```

and add an <env-entry> element after this to map a property set to your custom EntityPropertyManager, as shown in the following example:

```
<!-- map all properties in UUPExample property set to MyEntityPropertyManager -->
<env-entry>
  <env-entry-name>PropertyMapping/UUPExample</env-entry-name>
  <env-entry-type>java.lang.String</env-entry-type>
  <env-entry-value>MyEntityPropertyManager</env-entry-value>
</env-entry>
```

4. In ejb-jar.xml, find the <ejb-ref> element shown in the following example:

```
<!-- an ldap property manager -->
<ejb-ref>
  <ejb-ref-name>ejb/LdapPropertyManager</ejb-ref-name>
  <ejb-ref-type>Session</ejb-ref-type>
  <home>com.bea.p13n.property.EntityPropertyManagerHome</home>
  <remote>com.bea.p13n.property.EntityPropertyManager</remote>
</ejb-ref>
```

Developing Personalized Applications

and add an <ejb-ref> element after this to map a reference to an EJB that matches the name from the previous step with ejb/ prepended as shown in the following example:

```
<!-- an example property manager -->
<ejb-ref>
  <ejb-ref-name>ejb/MyEntityPropertyManager</ejb-ref-name>
  <ejb-ref-type>Session</ejb-ref-type>
  <home>examples.usermgmt.MyEntityPropertyManagerHome</home>
  <remote>examples.usermgmt.MyEntityPropertyManager</remote>
</ejb-ref>
```

The home and remote class names match the classes from your EJB JAR file for your custom EntityPropertyManager.

5. If your EntityPropertyManager implementation handles creating and removing profile records, you must also add Creator and Remover entries. For example:

```
<env-entry>
  <env-entry-name>Creator/Creator1</env-entry-name>
  <env-entry-type>java.lang.String</env-entry-type>
  <env-entry-value>MyEntityPropertyManager</env-entry-value>
</env-entry>

<env-entry>
  <env-entry-name>Remover/Remover1</env-entry-name>
  <env-entry-type>java.lang.String</env-entry-type>
  <env-entry-value>MyEntityPropertyManager</env-entry-value>
</env-entry>
```

This instructs the UserProfileManager to call your custom EntityPropertyManager when creating or deleting user profile records. The names "Creator1" and "Remover1" are arbitrary. All Creators and Removers will be iterated through when the UserProfileManager creates or removes a user profile. The value for the Creator and Remover matches the ejb-ref-name for your custom EntityPropertyManager without the ejb/ prefix.

6. From p13n_ejb.jar, extract META-INF/weblogic-ejb-jar.xml and open it for editing.
7. In weblogic-ejb-jar.xml, find the elements shown in the following example:

```
<weblogic-enterprise-bean>
  <ejb-name>UserProfileManager</ejb-name>
  <reference-descriptor>
    <ejb-reference-description>
      <ejb-ref-name>ejb/EntityPropertyManager</ejb-ref-name>
      <jndi-name>${APPNAME}.BEA_personalization.EntityPropertyManager</jndi-name>
    </ejb-reference-description>
  </reference-descriptor>
</weblogic-enterprise-bean>
```

and add an ejb-reference-description to map the ejb-ref for your custom EntityPropertyManager to the JNDI name. This JNDI name must match the name you assigned in weblogic-ejb-jar.xml in the JAR file for your customer EntityPropertyManager. It should look like the following example:

```
<weblogic-enterprise-bean>
  <ejb-name>UserProfileManager</ejb-name>
  <reference-descriptor>
    <ejb-reference-description>
      <ejb-ref-name>ejb/EntityPropertyManager</ejb-ref-name>
      <jndi-name>${APPNAME}.BEA_personalization.EntityPropertyManager</jndi-name>
    </ejb-reference-description>
    <ejb-reference-description>
      <ejb-ref-name>ejb/MyEntityPropertyManager</ejb-ref-name>
      <jndi-name>${APPNAME}.BEA_personalization.MyEntityPropertyManager</jndi-name>
    </ejb-reference-description>
  </reference-descriptor>
</weblogic-enterprise-bean>
```

Developing Personalized Applications

Note the `${APPNAME}` string substitution variable. The WebLogic EJB container automatically substitutes the enterprise application name to scope the JNDI name to the application.

8. Update `p13n_ejb.jar` for your new deployment descriptors. You can use the `jar uf` command to update the modified `META-INF/` deployment descriptors.
9. Edit `META-INF/application.xml` for your enterprise application to add an entry for your custom `EntityPropertyManager` EJB module as shown in the following example:

```
<module>
    <ejb>UUPEXample.jar</ejb>
</module>
```
10. If you are using an application-wide cache, you can manage it from the WebLogic Administration Console if you add a `<Cache>` tag for your cache to the `META-INF/application-config.xml` deployment descriptor for your enterprise application like this:

```
<Cache Name="UUPEXampleCache" TimeToLive="60000"/>
```
11. Verify the modified `p13n_ejb.jar` and your custom `EntityPropertyManager` EJB JAR archive are in the root directory of your enterprise application and start WebLogic Server.
12. Use the WebLogic Server Administration Console to verify your EJB module is deployed to the enterprise application and then use the console to add your server as a target for the EJB module. You need to select a target to have your domain's `config.xml` file updated to deploy your EJB module to the server.
13. Use the WebLogic Workshop Property Set Designer to create a User Profile (property set) that matches the name of the property set that you mapped to your custom `EntityPropertyManager` in `ejb-jar.xml` for the `UserProfileManager` (in `p13n_ejb.jar`). You could also map specific property names in a property set to your custom `EntityPropertyManager`.

Your new Unified User Profile type is ready to use. You can use the WebLogic Administration Portal to create a user of type "User," and it will use your UUP implementation when the "UUPEXample" property set is being modified. When you call `createUser("bob", "password")` or `createUser("bob", "password", null)` on the `UserManager`, several things will happen:

- A user named "bob" is created in the security realm.
- A WebLogic Portal Server profile record is created for "bob" in the user store.
- If you set up the Creator mapping, the `UserManager` will call the default `ProfileManager` deployment (`UserProfileManager`) which will call your custom `EntityPropertyManager` to create a record for Bob in your data source.
- Retrieving Bob's profile will use the default `ProfileManager` deployment (`UserProfileManager`), and when you request a property belonging to the "UUPEXample" property set, the request will be routed to your custom `EntityPropertyManager` implementation.

Configuring and Deploying a New ProfileManager

If you are going to deploy a newly configured `ProfileManager` instead of using the default `ProfileManager` (`UserProfileManager`) to manage your user profiles, perform the following steps to modify the deployment configuration. In most cases, you will not have to use this method of deployment. Use this method only if you need to support multiple types of users that require different `ProfileManager` deployments that allow a property set to be mapped to different custom `EntityPropertyManagers` based on `ProfileType`.

An example of this method is the deployment of the custom `CustomerProfileManager` in `customer.jar`. The `CustomerProfileManager` is configured to use the custom `EntityPropertyManager` (`CustomerPropertyManager`) for properties in the "CustomerProperties" property set. The `UserManager` EJB in `p13n_ejb.jar` is configured to map the "WLCS_Customer" `ProfileType` to the custom deployment of the `ProfileManager`, `CustomerProfileManager`.

Developing Personalized Applications

To configure and deploy a new ProfileManager, use this procedure.

1. Back up the p13n_ejb.jar file in your enterprise application root directory.
2. From p13n_ejb.jar, extract META-INF/ejb-jar.xml, and open it for editing.
3. In ejb-jar.xml, copy the entire <session> tag for the UserProfileManager, and configure it to use your custom implementation class for your new deployment of ProfileManager.

In addition, you could extend the UserProfileManager home and remote interfaces with your own interfaces if you want to repackage them to correspond to your packaging (for example., examples.usermgmt.MyProfileManagerHome, examples.usermgmt.MyProfileManager).

However, it is sufficient to replace the bean implementation class:

You must create an <env-entry> element to map a property set to your custom EntityPropertyManager. You must also create a <ejb-ref> element to map a reference to an EJB that matches the name from the PropertyMapping with ejb/ prepended. The home and remote class names for your custom EntityPropertyManager match the classes from your EJB JAR file for your custom EntityPropertyManager.

Also, if your EntityPropertyManager implementation handles creating and removing profile records, you must also add Creator and Remover entries. This instructs your new ProfileManager to call your custom EntityPropertyManager when creating or deleting user profile records.

Note: The name suffixes for the Creator and Remover, "Creator1" and "Remover1", are arbitrary. All Creators and Removers will be iterated through when your ProfileManager creates or removes a user profile. The value for the Creator and Remover matches the <ejb-ref-name> for your custom EntityPropertyManager without the ejb/ prefix.

4. In ejb-jar.xml, you must add an <ejb-ref> to the UserManager EJB section to map your ProfileType to your new deployment of the ProfileManager, as shown in the following example:

```
<ejb-ref>
  <ejb-ref-name>ejb/ProfileType/UUPExampleUser</ejb-ref-name>
  <ejb-ref-type>Session</ejb-ref-type>
  <home>com.bea.p13n.usermgmt.profile.ProfileManagerHome</home>
  <remote>com.bea.p13n.usermgmt.profile.ProfileManager</remote>
</ejb-ref>
```

The <ejb-ref-name> must start with ejb/ProfileType/ and must end with the name that you want to use as the profile type as an argument in the createUser() method of UserManager.

5. From p13n_ejb.jar, extract META-INF/weblogic-ejb-jar.xml and open it for editing.
6. In weblogic-ejb-jar.xml, copy the <weblogic-enterprise-bean> tag, shown in the following example, for the UserProfileManager and configure it for your new ProfileManager deployment:

```
<weblogic-enterprise-bean>
  <ejb-name>MyProfileManager</ejb-name>
  <reference-descriptor>
    <ejb-reference-description>
      <ejb-ref-name>ejb/EntityPropertyManager</ejb-ref-name>
      <jndi-name>${APPNAME}.BEA_personalization.EntityPropertyManager</jndi-name>
    </ejb-reference-description>
    <ejb-reference-description>
      <ejb-ref-name>ejb/PropertySetManager</ejb-ref-name>
      <jndi-name>${APPNAME}.BEA_personalization.PropertySetManager</jndi-name>
    </ejb-reference-description>
    <ejb-reference-description>
      <ejb-ref-name>ejb/MyEntityPropertyManager</ejb-ref-name>
      <jndi-name>${APPNAME}.BEA_personalization.MyEntityPropertyManager</jndi-name>
    </ejb-reference-description>
  </reference-descriptor>
  <jndi-name>${APPNAME}.BEA_personalization.MyProfileManager</jndi-name>
```

Developing Personalized Applications

```
</weblogic-enterprise-bean>
```

You must create an `<ejb-reference-description>` to map the `<ejb-ref>` for your custom `EntityPropertyManager` to the JNDI name. This JNDI name must match the name you assigned in `weblogic-ejb-jar.xml` in the JAR file for your custom `EntityPropertyManager`. Note the `${APPNAME}` string substitution variable. The WebLogic Server EJB container automatically substitutes the enterprise application name to scope the JNDI name to the application.

7. In `weblogic-ejb-jar.xml`, copy the `<transaction-isolation>` tag for the `UserProfileManager`, shown in the following example, and configure it for your new `ProfileManager` deployment:

```
<transaction-isolation>
  <isolation-level>TRANSACTION_READ_COMMITTED</isolation-level>
  <method>
    <ejb-name>MyProfileManager</ejb-name>
    <method-name>*</method-name>
  </method>
</transaction-isolation>
```

8. Create a temporary `p13n_ejb.jar` for your new deployment descriptors and your new `ProfileManager` bean implementation class. This temporary EJB JAR archive should not have any container classes in it. Run `ejbc` to generate new container classes.
9. Edit `META-INF/application.xml` for your enterprise application to add an entry for your custom `EntityPropertyManager` EJB module, as shown in the following example:

```
<module>
  <ejb>UUPEXample.jar</ejb>
</module>
```

10. If you are using an application-wide cache, you can manage it from the WebLogic Server Administration Console if you add a `<Cache>` tag for your cache to the `META-INF/application-config.xml` deployment descriptor for your enterprise application as shown in the following example:

```
<Cache Name="UUPEXampleCache" TimeToLive="60000"/>
```

Verify the modified `p13n_ejb.jar` and your custom `EntityPropertyManager` EJB JAR archive are in the root directory of your enterprise application and start your server.

11. Use the WebLogic Server Administration Console to verify your EJB module is deployed to the enterprise application and add your server as a target for the EJB module. You must select a target to have your domain's `config.xml` file updated to deploy your EJB module to the server.
12. Use the WebLogic Workshop Property Set Designer to create a User Profile (property set) that matches the name of the property set that you mapped to your custom `EntityPropertyManager` in `ejb-jar.xml` for the `UserProfileManager` (in `p13n_ejb.jar`). You could also map specific property names in a property set to your custom `EntityPropertyManager`.

Your new Unified User Profile type is ready to use. You can use the WebLogic Administration Portal to create a user of type `"UUPEXampleUser,"` and it will use your UUP implementation when the `"UUPEXample"` property set is being modified. That is because you mapped the `ProfileType` using an `<ejb-ref>` in your `UserManager` deployment descriptor, `ejb/ProfileType/UUPEXampleUser`.

Note: Tell your administrators that when they create a user in the WebLogic Administration Portal, they must select the new user type.

Now, when you call `createUser("bob", "password", "UUPEXampleUser")` on the `UserManager`, several things will happen:

- A user named `"bob"` is created in the security realm.

- A WebLogic Portal Server profile record is created for "bob" in the WebLogic Portal RDBMS repository.
- If you set up the Creator mapping, the UserManager will call your new ProfileManager deployment, which will call your custom EntityPropertyManager to create a record for Bob in your data source.
- Retrieving Bob's profile will use your new ProfileManager deployment, and when you request a property belonging to the "UUPEXample" property set, the request will be routed to your custom EntityPropertyManager implementation.

Retrieving User Profile Data from LDAP

The LdapRealm security realm and the LdapPropertyManager unified user profile (UUP) for retrieving user properties from LDAP are independent of each other. They do not share configuration information and there is no requirement to use either one in conjunction with the other. A security realm has nothing to do with a user profile. A security realm provides user/password data, user/group associations, and group/group associations. A user profile provides user and group properties. A password is not a property.

In order to successfully retrieve the user profile from the LDAP server, ensure that you've done the following:

1. If you have already deployed the application on a WebLogic Portal instance, stop the server.
2. Deploy the ldaprofile.jar component within your application.

The LdapPropertyManager EJB in ldaprofile.jar allows for the inspection of the LDAP schema to determine multi-valued versus single-value LDAP attributes, to allow for multiple userDN/groupDN, and to allow for SUBTREE_SCOPE searches for users and groups in the LDAP server. Following are more detailed explanations:

The determination of multi-value versus single-value LDAP attributes allows a developer to configure the ejb-jar.xml deployment descriptor for the LdapPropertyManager EJB to specify that the LDAP schema be used to determine if a property is single- or multi-value.

This is done by editing ejb-jar.xml to setting the following env-entries:

```
<!-- Flag to specify if LDAP attributes will be determined to be single value
or multi-value via the schema obtained from the attribute. If false,
then the attribute is stored as multi-valued (a Collection) only if it has
more than one value. Leave false unless you intend to use multi-valued LDAP
attributes that may have only one value. Using true adds overhead to check
the LDAP schema. Also, if you use true beware that most LDAP attributes are
multi-value. For example, iPlanet Directory Server 5.x uses multi-value for
givenName, which you may not expect unless you are familiar with LDAP schemas. -->

<env-entry>
  <env-entry-name>config/detectSingleValueFromSchema</env-entry-name>
  <env-entry-type>java.lang.Boolean</env-entry-type>
  <env-entry-value>true</env-entry-value>
</env-entry>

<!-- Value of the name of the attribute in the LDAP schema that is used
to determine single value or multi-value (RFC2252 uses SINGLE-VALUE).
This attribute in the schema should be true for single value and false
or absent from the schema otherwise. The value only matters if
config/detectSingleValueFromSchema is true. -->

<env-entry>
  <env-entry-name>config/singleValueSchemaAttribute</env-entry-name>
```

Developing Personalized Applications

```
<env-entry-type>java.lang.String</env-entry-type>
<env-entry-value>SINGLE-VALUE</env-entry-value>
</env-entry>
```

It is not recommended that true be used for config/detectSingleValueFromSchema unless you are going to write rules that use multi-valued LDAP attributes that have a single value. Using config/detectSingleValueFromSchema = true adds the overhead of checking the LDAP schema for each attribute instead of the default behavior (config/detectSingleValueFromSchema = false), which only stores an attribute as multi-valued (in a Collection) if it has more than one value.

This feature also implements changes that allow you to use SUBTREE_SCOPE searches for users and groups. It also allows multiple base userDN and groupDN to be specified. The multiple base DN can be used with SUBTREE_SCOPE searches enabled or disabled.

A SUBTREE_SCOPE search begins at a base userDN (or groupDN) and works down the branches of that base DN until the first user (or group) is found that matches the username (or group name).

To enable SUBTREE_SCOPE searches you must set the Boolean config/objectPropertySubtreeScope env-entry in the ejb-jar.xml for ldaprofile.jar to true and then you must set the config/userDN and config/groupDN env-entry values to be equal to the base DNs from which you want your SUBTREE_SCOPE searches to begin.

For example, if you have users in ou=PeopleA,ou=People,dc=mycompany,dc=com and in ou=PeopleB,ou=People,dc=mycompany,dc=com then you could set config/userDN to ou=People,dc=mycompany,dc=com and properties for these users would be retrieved from your LDAP server because the user search would start at the "People" ou and work its way down the branches (ou="PeopleA" and ou="PeopleB").

You should not create duplicate users in branches below your base userDN (or duplicate groups below your base groupDN) in your LDAP server. For example, your LDAP server will allow you to create a user with the uid="userA" under both your PeopleA and your PeopleB branches. The LdapPropertyManager in ldaprofile.jar will return property values for the first userA that it finds.

It is recommended that you do not enable this change (by setting config/objectPropertySubtreeScope to true) unless you need the flexibility offered by SUBTREE_SCOPE searches.

An alternative to SUBTREE_SCOPE searches (with or without multiple base DNs) would be to configure multiple base DNs and leave config/objectPropertySubtreeScope set to false. Each base DN would have to be the DN that contains the users (or groups) because searches would not go any lower than the base DN branches. The search would cycle from one base DN to the next until the first matching user (or group) is found.

The new ejb-jar.xml deployment descriptor is fully commented to explain how to set multiple DNs, multiple usernameAttributes (or groupnameAttributes), and how to set the objectPropertySubtreeScope flag.

3. Start the server and deploy the application.
4. Start the WebLogic Server Administration Console for the active domain.

You can set a default profile type for each Web project by setting a context parameter in web.xml for DEFAULT_USER_PROFILE_TYPE. For example:

```
<context-param>
```

Developing Personalized Applications

```
<param-name>DEFAULT_USER_PROFILE_TYPE</param-name>  
  <param-value>WLCS_Customer</param-value>  
</context-param>
```

To create a custom user profile property set, see [Creating User Profile Properties](#).

Creating User Profile Properties

User profile properties are name/value pairs attached to users and used for entering personal information about users. For example, you could create a property set called "human resources" that contains properties such as "gender," "hire date," and "social security number." User profile properties appear as input fields in the Administration Portal when you edit a user's profile values. The properties you create are also used to define queries when you develop personalization functionality for your applications. Users can have multiple profiles.

Note: The WebLogic Workshop Portal Extensions provide a default user profile property set called `CustomerProperties.usr` that contains many common properties you may want to use.

Properties you create are surfaced automatically in the WebLogic Administration Portal Users & Groups tools. Portal administrators can assign property values to each user. Properties you create can also serve as placeholders for programs that need to store user profile information. For example, if you add My Yahoo! Enterprise Edition to a portal Web project, it includes a property set that automatically stores a Yahoo ID for users when they link from the portal to their Yahoo! account.

To create a User Profile Property Set

Be sure you have added a Data Sync project to your Portal Application. See [Creating a Portal Application and Portal Web Project](#) for more information.

1. In the Application window, right-click the **data\userprofiles** folder and choose **New-->User Profile Property Set**.
2. In the New File window, enter a name for the User Profile property set in the **File name** field. Make sure you keep the file extension.
3. Click **Create**. The User Profile Property Set designer appears.
4. Use the next procedure to add properties to the property set.

To add properties to a property set

After you create a property set, you add the properties you want to it.

1. In the Palette window, drag one of the types of properties into the designer window.

The type defines the number of values that can be entered for the property. Following are descriptions of each type.

Single Unrestricted – A single unrestricted property can have only one value, but you can enter any value.

Single Restricted – A single restricted property can have only one value, and you are restricted to selecting that value from a predefined list.

Multiple Unrestricted – A multiple unrestricted property can have multiple values, and you can enter any values.

Multiple Restricted – A multiple restricted property can have multiple values, and you are restricted to selecting the values from a predefined list.

Developing Personalized Applications

2. In the Property designer window:

- ◆ Enter a name and description for the property
- ◆ Select the **Data Type** for the property value. For example, if you select Boolean, your property value can be only true or false. (Properties with a Boolean data type are automatically set to "single restricted.")
- ◆ In the **Selection Mode** and **Value Range** fields, you can change the type of property. For example, you can change a property from "single unrestricted" to "multiple restricted."
Note: Any change to Data Type, Selection Mode, or Value Range removes anything previously entered in the Values field.
- ◆ Use the **Values** field to enter values for "restricted" types or to set the default value(s) for "unrestricted types." Click the ellipsis icon (...) to enter values. (In the Enter Property Value dialog box that appears, click **Add** after each entry, and click **OK** when all values are entered.)

3. Save the file after you have added all the properties you want.

Note: You can also use the User Profile Control to create property sets. However, property sets created with this control are not surfaced in the WebLogic Administration Portal. They must be modified and updated programmatically.

To modify properties and their values

To modify properties and their values, double-click the property set file in the the Application window, click the property you want to modify, and change the values in the Property designer window.

You can also use the <um:setProperty> JSP Tag in your JSPs to modify existing property values for users.

To delete properties

You can delete individual properties from a property set, and you can delete property sets.

To delete a property from a property set, open the property set file, select the property, and press the **Delete** key.

To delete a property set, select the property set file in the Application window and press the **Delete** key.

Related Topics

Tutorial: Showing Personalized Content in a Portlet

Property Set Designer window

Creating Segments

Creating Content Selectors

Creating Campaigns

Creating Request Properties

Creating Session Properties

Creating User Profile Properties

Developing Personalized Applications

Creating Catalog Structure Properties

Registering Custom Events

Creating User Segments

You can target visitors with Web content, automatic e-mails, and discounts by defining and using groups called User Segments (as in segments of a population). Instead of being groups of hard-coded users, Segments are groupings of characteristics, such as gender, the type of browser being used, and date or time information. If users match the characteristics, they are automatically and dynamically members of that User Segment and are targeted with the Web content, e-mail, or discounts you determine.

After you create a user segment, you can use it to when you define your campaigns or content selectors.

To create a User Segment

Be sure you have added a Data Sync project to your Portal Application. See [Creating a Portal Application and Portal Web Project](#) for more information.

1. In the Application window, right-click the *data\segments\GlobalClassifications* folder and choose *New-->User Segment*.
2. In the New File window, enter a name for the User Segment in the *File name* field. Make sure you keep the file extension.
3. Click *Create*. The User Segment designer appears.
4. In the *Available Conditions* section of the designer window, select the types of conditions under which a user will be a member of the User Segment.

As you select conditions, corresponding links appear in the top of the designer window.

7. Click the corresponding links to create the conditions you selected, and enter the appropriate information.
8. Save the file.

To modify and delete User Segments

To modify a User Segment, open its file by double-clicking it in the Application window.

To delete a User Segment, select it in the Application window, press the *Delete* key, and click OK in the confirmation dialog box.

Related Topics

[Personalization Conditions Reference](#)

[User Segment Designer](#)

[Preparing to Use Campaigns](#)

[Creating Campaigns](#)

[Creating Content Selectors](#)

Tracking Anonymous Users

This feature enables you to track users before they register with your site, meaning you can present personalized content to an anonymous user over multiple sessions. You can also persist information about a user's behavior, provided cookies are enabled on the user's browser. This feature can be parameterized to exclude those who spend less than a designated period of time on your site, avoiding the expense of storing data on irrelevant users. Finally, should a tracked anonymous user choose to register with your site, the data collected on that user is persisted in the account of the newly-registered user.

Five User Profile Types for User Tracking

If Anonymous User Tracking is not enabled, users who visit the site without registering will be considered **ANONYMOUS**. If it is enabled, such users would be considered **TRACKABLE** on the first visit.

If a **TRACKABLE** anonymous user remains at a site longer than the specified duration, the user is converted automatically into a **TRACKED** anonymous user, and a cookie with the tracking id is given to the user. That id is used as the primary key of the tracking EJBs associated with that user, so that when the session ends, the user's properties are persisted and available for the next visit.

If this **TRACKED** anonymous user returns, the tracking id in the user's cookie is used to retrieve the user properties from the database, and those properties may be used to enable campaigns to be run against the user.

Finally, if a **TRACKED** anonymous user registers at the site, the tracking data is transferred to the newly created user (now considered a **REGISTERED** user) and deleted from the anonymous user.

The final user profile type is **UNKNOWN**, which designates users whose status cannot be determined.

Tracking Based on Visit Duration

Because users don't always remain at a site long enough to demonstrate interest, the duration of the initial visit is used as a configurable trigger. If this value is set to 30 seconds, a user that leaves after 25 seconds would not be remembered on the next visit to the same site. A 30-second or more user would be remembered, even without having registered.

This parameter is set by editing the web.xml file.

Related Topics

Creating Anonymous User Profiles

Adding Personalization and Interaction Management to a Portal

Enabling Anonymous User Tracking

Here are specific instructions for enabling your Web application to track anonymous users. Using this feature requires that cookies be enabled on the user's browser.

Anonymous User Tracking is not enabled by default. It must be enabled in order to function by completing the following tasks:

- Enable Anonymous user Duration Checking
- Configure Visit Duration

To Enable Tracked Anonymous Users

Anonymous User Tracking is implemented by configuring the PortalServletFilter by adding the following entry to the web.xml file inside your Web application:

```
<!-- Portal Servlet Filter, always required for Portal --> <filter> <filter-name> PortalServletFilter
</filter-name> <filter-class> com.bea.p13n.servlets.PortalServletFilter </filter-class> <init-param>
<param-name>fireSessionLoginEvent</param-name> <param-value>>false</param-value> <description>
Option to fire SessionLoginEvent , defaults to false if not set</description> </init-param> <init-param>
<param-name>createAnonymousProfile</param-name> <param-value>true</param-value> <description>
Filter will create an anonymous profile for every session. Defaults to true if not set</description>
</init-param> <init-param> <param-name>enableTrackedAnonymous</param-name>
<param-value>true</param-value> <description> Option to track anonymous users , defaults to false if not
set. 'createAnonymousProfile' is ignored if this is true</description> </init-param> <init-param>
<param-name>trackedAnonymousVisitDuration</param-name> <param-value>5</param-value>
<description> Length in seconds visitor must be on site before we start tracking them . Defaults to 60 seconds
if not set</description> </init-param> </filter>
```

Related Topics

[Creating Anonymous User Profiles](#)

[Tracking Anonymous Users](#)

[Adding Personalization and Interaction Management to a Portal](#)

Creating Anonymous User Profiles

If a user visits your site and then registers within the same session, any data collected on that user is persisted in the account of the newly-registered user.

The Create Anonymous Profile parameter does not enable Anonymous Users to be tracked from one session to the next without registering. This would require the Tracked Anonymous User feature.

NOTE: This feature defaults to True.

Disabling the Create Anonymous Profile Feature

This feature is configured with the PortalServletFilter settings in the web.xml for the enterprise application.

To disable this feature, set the init-param node called createAnonymousProfile to False.

```
<init-param>
    <param-name>createAnonymousProfile</param-name>
    <param-value>true</param-value>
    <description>
        Filter will create an anonymous profile for every session. Defa
    </description>
</init-param>
```

Related Topics

Tracked Anonymous Users

Designing Interaction Management

Use the following list of scenarios to help you determine which type of interaction management you want to develop. Use the guidance in the "do this" column to help you identify the steps to take in the next phases of the interaction management development process: Creating Personalization Conditions, and Personalizing Portal Applications.

<i>If you want to...</i>	<i>do this:</i>
<p>Display a binary property from a single content node from the Virtual Content Repository that can change each time a user visits your portal or clicks the browser refresh button.</p> <p>For example, each time an employee visits the Intranet portal, display a different picture from the company picnic.</p>	<p>For creating a generic rotation of content for all users, create a Placeholder and add a default query for the Placeholder that displays the range of content you want.</p> <p>For creating a targeted rotation of content for each user based on each user's characteristics, create a Placeholder and a campaign. Set up the campaign with content actions that put different types of content in the Placeholder for different types of users. Define the necessary conditions and rules to be used in the campaign.</p> <p>You can also use the <ad:adTarget> JSP tag as an alternative to a Placeholder to manually embed a content query in a JSP.</p>
<p>Display a binary property from a single content node from the Virtual Content Repository that shows the same content node for each type of user.</p> <p>For example, if a user of type "manager" views the portal, always show the manager the "performance review reminder" graphic. If a user of type "regular employee" views the portal, always show the employee the "benefits open enrollment" graphic.</p>	<p>As in the previous scenario, you can create a Placeholder with a default query for all users or a Placeholder used by a campaign to target specific users differently. There are two keys to showing the same content node without content rotation:</p> <ul style="list-style-type: none"> • Set up your content with properties and values that can uniquely identify each piece of content. • Create highly focused content queries in the Placeholder or the campaign to retrieve those single unique content nodes.
<p>Display multiple content nodes and properties from the Virtual Content Repository simultaneously.</p> <p>For example, show each user a unique list of recommended books based on the user's characteristics.</p>	<p>To show multiple content nodes from the Virtual Content Repository simultaneously, create content selectors and add them to your JSPs.</p>
<p>Display personalized content from an inline section of a JSP.</p>	<p>To display personalized inline JSP content, create user segments and use the <pz:div></p>

Developing Personalized Applications

For example, provide three different sections of HTML content in a JSP but show users only the sections that match their characteristics.	<p>JSP tag to wrap personalized content.</p> <p>You can also use the following JSP tags to display inline JSP content based on the device that is viewing the content (for example, a handheld device or a PC):</p> <pre><cscm:default>, <cscm:not-default>, <cscm:recognized>, <cscm:not-recognized>, <cscm:when>, and <cscm:when-not>.</pre>
Send users automatic e-mails.	Create and store e-mail message files (see Preparing to Use Campaigns) and create a campaign that uses an e-mail action.
Give users automatic discounts.	<p>Perform the following tasks:</p> <ul style="list-style-type: none"> • Add commerce services to your portal application. • Set up a shopping cart using the WebLogic Portal commerce API. • Create a catalog in the Virtual Content Repository. • Use the WebLogic Portal catalog classes in the commerce API to surface catalog items from the Virtual Content Repository and identify them with "categories" and "SKU" numbers. • Create discounts and use the commerce API to surface the discounts in your shopping cart. If desired, use the API to surface the discount's description next to the discount amount displayed in the shopping cart.

Related Topics

Developing Personalized Applications

Overview of Content Management

Setting up Users

Creating Personalization Conditions

Personalizing Portal Applications

Creating Personalization Conditions

While you can target users with personalization and campaigns based on generic factors such as date and time, you can develop more finely targeted personalization and campaigns by creating fine-grained conditions that determine precisely when and under what conditions users see their personalized content or the effects of the campaign targeting them.

For example, you can display personalized content to users who visit your portal between January 1 and April 15. However, you can define more fine-grained conditions that display personalized content only to users identified (by their user profile) as managers belonging of a specific division of the organization.

This topic provides links to instructions on creating personalization conditions.

Note: Many interaction management development tasks are dependent on other tasks. For example, if you are creating a user segment based on user profile properties, you must first create the user profile properties you want; and if you are creating a campaign and you want to trigger campaign actions based on user segments, you must first create the user segments you want. The following tasks highlight typical usage patterns.

Creating User Segments

You must create user segments if you want to use them in content selectors and campaigns.

Creating User Profile Properties

You must create user profile properties if you want to use them in user segments, content selectors, and campaigns.

Creating Session Properties

You must create session properties if you want to use them in user segments, content selectors, and campaigns.

Creating Request Properties

You must create request properties if you want to use them in user segments, content selectors, and campaigns.

Related Topics

[Developing Personalized Applications](#)

[Overview of Content Management](#)

[Setting up Users](#)

[Designing Interaction Management](#)

[Personalizing Portal Applications](#)

[Personalization Conditions Reference](#)

[Creating Personalization Conditions](#)

Creating User Profile Properties

User profile properties are name/value pairs attached to users and used for entering personal information about users. For example, you could create a property set called "human resources" that contains properties such as "gender," "hire date," and "social security number." User profile properties appear as input fields in the Administration Portal when you edit a user's profile values. The properties you create are also used to define queries when you develop personalization functionality for your applications. Users can have multiple profiles.

Note: The WebLogic Workshop Portal Extensions provide a default user profile property set called `CustomerProperties.usr` that contains many common properties you may want to use.

Properties you create are surfaced automatically in the WebLogic Administration Portal Users & Groups tools. Portal administrators can assign property values to each user. Properties you create can also serve as placeholders for programs that need to store user profile information. For example, if you add My Yahoo! Enterprise Edition to a portal Web project, it includes a property set that automatically stores a Yahoo ID for users when they link from the portal to their Yahoo! account.

To create a User Profile Property Set

Be sure you have added a Data Sync project to your Portal Application. See [Creating a Portal Application and Portal Web Project](#) for more information.

1. In the Application window, right-click the **data\userprofiles** folder and choose **New-->User Profile Property Set**.
2. In the New File window, enter a name for the User Profile property set in the **File name** field. Make sure you keep the file extension.
3. Click **Create**. The User Profile Property Set designer appears.
4. Use the next procedure to add properties to the property set.

To add properties to a property set

After you create a property set, you add the properties you want to it.

1. In the Palette window, drag one of the types of properties into the designer window.

The type defines the number of values that can be entered for the property. Following are descriptions of each type.

Single Unrestricted – A single unrestricted property can have only one value, but you can enter any value.

Single Restricted – A single restricted property can have only one value, and you are restricted to selecting that value from a predefined list.

Multiple Unrestricted – A multiple unrestricted property can have multiple values, and you can enter any values.

Multiple Restricted – A multiple restricted property can have multiple values, and you are restricted to selecting the values from a predefined list.

Developing Personalized Applications

2. In the Property designer window:

- ◆ Enter a name and description for the property
- ◆ Select the **Data Type** for the property value. For example, if you select Boolean, your property value can be only true or false. (Properties with a Boolean data type are automatically set to "single restricted.")
- ◆ In the **Selection Mode** and **Value Range** fields, you can change the type of property. For example, you can change a property from "single unrestricted" to "multiple restricted."
Note: Any change to Data Type, Selection Mode, or Value Range removes anything previously entered in the Values field.
- ◆ Use the **Values** field to enter values for "restricted" types or to set the default value(s) for "unrestricted types." Click the ellipsis icon (...) to enter values. (In the Enter Property Value dialog box that appears, click **Add** after each entry, and click **OK** when all values are entered.)

3. Save the file after you have added all the properties you want.

Note: You can also use the User Profile Control to create property sets. However, property sets created with this control are not surfaced in the WebLogic Administration Portal. They must be modified and updated programmatically.

To modify properties and their values

To modify properties and their values, double-click the property set file in the the Application window, click the property you want to modify, and change the values in the Property designer window.

You can also use the <um:setProperty> JSP Tag in your JSPs to modify existing property values for users.

To delete properties

You can delete individual properties from a property set, and you can delete property sets.

To delete a property from a property set, open the property set file, select the property, and press the **Delete** key.

To delete a property set, select the property set file in the Application window and press the **Delete** key.

Related Topics

Tutorial: Showing Personalized Content in a Portlet

Property Set Designer window

Creating Segments

Creating Content Selectors

Creating Campaigns

Creating Request Properties

Creating Session Properties

Creating User Profile Properties

Developing Personalized Applications

Creating Catalog Structure Properties

Registering Custom Events

Creating User Profile Properties

Creating User Segments

You can target visitors with Web content, automatic e-mails, and discounts by defining and using groups called User Segments (as in segments of a population). Instead of being groups of hard-coded users, Segments are groupings of characteristics, such as gender, the type of browser being used, and date or time information. If users match the characteristics, they are automatically and dynamically members of that User Segment and are targeted with the Web content, e-mail, or discounts you determine.

After you create a user segment, you can use it to when you define your campaigns or content selectors.

To create a User Segment

Be sure you have added a Data Sync project to your Portal Application. See [Creating a Portal Application and Portal Web Project](#) for more information.

1. In the Application window, right-click the *data\segments\GlobalClassifications* folder and choose *New-->User Segment*.
2. In the New File window, enter a name for the User Segment in the *File name* field. Make sure you keep the file extension.
3. Click *Create*. The User Segment designer appears.
4. In the *Available Conditions* section of the designer window, select the types of conditions under which a user will be a member of the User Segment.

As you select conditions, corresponding links appear in the top of the designer window.

7. Click the corresponding links to create the conditions you selected, and enter the appropriate information.
8. Save the file.

To modify and delete User Segments

To modify a User Segment, open its file by double-clicking it in the Application window.

To delete a User Segment, select it in the Application window, press the *Delete* key, and click OK in the confirmation dialog box.

Related Topics

[Personalization Conditions Reference](#)

[User Segment Designer](#)

[Preparing to Use Campaigns](#)

[Creating Campaigns](#)

[Creating Content Selectors](#)

Creating Session Properties

Session Properties are name/value pairs that let you capture and use specific HTTP session information to trigger personalization and campaigns. Session properties are associated with a session and are not persisted between sessions.

To create a Session Property Set

Be sure you have added a Data Sync project to your Portal Application. See [Creating a Portal Application and Portal Web Project](#) for more information.

1. In the Application window, right-click the **data\session** folder and choose **New-->Other File Types**.
2. In the New File window, select the **Portal** folder in the left pane.
3. In the right pane, select **Session Property Set**.
4. In the **File name** field, enter a name for the Session property set. Make sure you keep the file extension.
5. Click **Create**. The Session Property Set designer appears.
6. Use the next procedure to add properties to the property set.

To add properties to a property set

After you create a property set, you add the properties you want to it.

1. In the Palette window, drag one of the types of properties into the designer window.

The type defines the number of values that can be entered for the property. Following are descriptions of each type.

Single Unrestricted – A single unrestricted property can have only one value, but you can enter any value.

Single Restricted – A single restricted property can have only one value, and you are restricted to selecting that value from a predefined list.

Multiple Unrestricted – A multiple unrestricted property can have multiple values, and you can enter any values.

Multiple Restricted – A multiple restricted property can have multiple values, and you are restricted to selecting the values from a predefined list.

2. In the Property designer window:

- ◆ Enter a name and description for the property
- ◆ Select the **Data Type** for the property value. For example, if you select Boolean, your property value can be only true or false. (Properties with a Boolean data type are automatically set to "single restricted.")
- ◆ In the **Selection Mode** and **Value Range** fields, you can change the type of property. For example, you can change a property from "single unrestricted" to "multiple restricted."
Note: Any change to Data Type, Selection Mode, or Value Range removes anything previously entered in the Values field.

Developing Personalized Applications

- ◆ Use the **Values** field to enter values for "restricted" types or to set the default value(s) for "unrestricted types." Click the ellipsis icon (...) to enter values. (In the Enter Property Value dialog box that appears, click **Add** after each entry, and click **OK** when all values are entered.)
- 3. Save the file after you have added all the properties you want.

To modify properties and their values

To modify properties and their values, double-click the property set file in the the Application window, click the property you want to modify, and change the values in the Property designer window.

Use code in a JSP or Java Page Flow to populate the session.

To delete properties

You can delete individual properties from a property set, and you can delete property sets.

To delete a property from a property set, open the property set file, select the property, and press the **Delete** key.

To delete a property set, select the property set file in the Application window and press the **Delete** key.

Related Topics

Guide to Building Page Flows

Property Set Designer window

Creating Segments

Creating Content Selectors

Creating Campaigns

Creating User Profile Properties

Creating Request Properties

Creating Catalog Structure Properties

Registering Custom Events

Creating Request Properties

Request Properties are name/value pairs that let you capture and use specific HTTP request information to trigger personalization. Request properties are associated with a request and are not persisted between requests.

To create a Request Property Set

Be sure you have added a Data Sync project to your Portal Application. See [Creating a Portal Application and Portal Web Project](#) for more information.

1. In the Application window, right-click the **data/request** folder and choose **New-->Other File Types**.
2. In the New File window, select the **Portal** folder in the left pane.
3. In the right pane, select **Request Property Set**.
4. In the **File name** field, enter a name for the Request property set. Make sure you keep the file extension.
5. Click **Create**. The Request Property Set designer appears.
6. Use the next procedure to add properties to the property set.

To add properties to a property set

After you create a property set, you add the properties you want to it.

1. In the Palette window, drag one of the types of properties into the designer window.

The type defines the number of values that can be entered for the property. Following are descriptions of each type.

Single Unrestricted – A single unrestricted property can have only one value, but you can enter any value.

Single Restricted – A single restricted property can have only one value, and you are restricted to selecting that value from a predefined list.

Multiple Unrestricted – A multiple unrestricted property can have multiple values, and you can enter any values.

Multiple Restricted – A multiple restricted property can have multiple values, and you are restricted to selecting the values from a predefined list.

2. In the Property designer window:

- ◆ Enter a name and description for the property
- ◆ Select the **Data Type** for the property value. For example, if you select Boolean, your property value can be only true or false. (Properties with a Boolean data type are automatically set to "single restricted.")
- ◆ In the **Selection Mode** and **Value Range** fields, you can change the type of property. For example, you can change a property from "single unrestricted" to "multiple restricted."
Note: Any change to Data Type, Selection Mode, or Value Range removes anything previously entered in the Values field.

Developing Personalized Applications

- ◆ Use the **Values** field to enter values for "restricted" types or to set the default value(s) for "unrestricted types." Click the ellipsis icon (...) to enter values. (In the Enter Property Value dialog box that appears, click **Add** after each entry, and click **OK** when all values are entered.)
- 3. Save the file after you have added all the properties you want.

To modify properties and their values

To modify properties and their values, double-click the property set file in the the Application window, click the property you want to modify, and change the values in the Property designer window.

Request properties can also store values that are populated programmatically. For example, the <project>\data\request\DefaultRequestPropertySet.req property set included with every portal Web project contains properties called "User-Agent" and "Client Classification." These properties are populated automatically when a device accesses a portal, providing a key component of the multichannel framework in delivering appropriate Web content to mobile devices.

To delete Properties

You can delete individual properties from a property set, and you can delete property sets.

To delete a property from a property set, open the property set file, select the property, and press the **Delete** key.

To delete a property set, select the property set file in the Application window and press the **Delete** key.

Related Topics

Property Set Designer window

Creating Segments

Creating Content Selectors

Creating Campaigns

Creating User Profile Properties

Creating Session Properties

Creating Catalog Structure Properties

Registering Custom Events

Creating Catalog Structure Properties

The Catalog Structure Properties designer lets you add properties to your catalog. To use catalog structure properties, you must enable the WebLogic Portal Administration Tools that support catalog management.

Catalog Structure Properties are name/value pairs that lets you define the information you want to enter about items in your catalog, such as "SKU," "Description," and "Price."

To create a Catalog Structure Property Set

1. Enable catalog management, which provides the catalog tools that use the catalog structure properties. See Enabling Catalog Management.
2. In the Application window, right-click the `data\catalog` folder and choose **File-->New-->Other File Types**.
3. In the New File window, select **Catalog Structure Property Set** in the right pane.
4. In the **File name** field, enter a name for the property set. Make sure you keep the file extension.
5. Click **Create**. The Property Set designer appears.
6. Use the next procedure to add properties to the property set.

To add properties to a property set

After you create a property set, you add the properties you want to it.

1. In the Palette window, drag one of the types of properties into the Property Set Designer.

The type defines the number of values that can be entered for the property. Following are descriptions of each type.

Single Unrestricted – A single unrestricted property can have only one value, but you can enter any value.

Single Restricted – A single restricted property can have only one value, and you are restricted to selecting that value from a predefined list.

Multiple Unrestricted – A multiple unrestricted property can have multiple values, and you can enter any values.

Multiple Restricted – A multiple restricted property can have multiple values, and you are restricted to selecting the values from a predefined list.

2. In the Property Editor window:

- ◆ Enter a name and description for the property
 - ◆ Select the **Data Type** for the property value. For example, if you select Boolean, your property value can be only true or false. (Properties with a Boolean data type are automatically set to "single restricted.")
 - ◆ In the **Selection Mode** and **Value Range** fields, you can change the type of property. For example, you can change a property from "single unrestricted" to "multiple restricted."
- Note:** Any change to Data Type, Selection Mode, or Value Range removes anything previously entered in the Values field.

Developing Personalized Applications

- ◆ Use the **Values** field to enter values for "restricted" types or to set the default value(s) for "unrestricted types." Click the ellipsis icon (...) to enter values. (In the Enter Property Value dialog box that appears, click **Add** after each entry, and click **OK** when all values are entered.)
- 3. Save the file after you have added all the properties you want.

To modify properties and their values

To modify properties and their values, double-click the property set file in the the Application window, click the property you want to modify, and change the values in the Property designer window.

To delete properties

You can delete individual properties from a property set, and you can delete property sets.

To delete a property from a property set, open the property set file, select the property, and press the **Delete** key.

To delete a property set, select the property set file in the Application window and press the **Delete** key.

Related Topics

Property Set Designer window

Building a Commerce Application

Creating Segments

Creating Content Selectors

Creating Campaigns

Creating User Profile Properties

Creating Session Properties

Creating Request Properties

Registering Custom Events

Registering Custom Events

After you create custom events for your portal application you must register those events so that your application recognizes them. After you register your events, you can use them to trigger personalization and campaigns and track user behavior in your portals.

To register the event listener

Events need event listeners to listen for them. Register the event listeners for your custom event in <PORTAL_APP>\META-INF\application-config.xml.

To create an Event Property Set

Be sure you have added a Data Sync project to your Portal Application. See *Creating a Portal Application and Portal Web Project* for more information.

1. In the Application window, right-click the *data\events* folder and choose *New-->Other File Types*.
2. In the New File window, select the *Portal* folder in the left pane.
3. In the right pane, select *Event Property Set*.
4. In the *File name* field, enter a name for the Event property set. Make sure you keep the file extension.
5. Click *Create*. The Event Property Set designer appears.
6. Use the next procedure to add properties to the property set.

To adding properties to a property set

After you create a property set, you add the properties you want to it.

1. In the Palette window, drag one of the types of properties into the designer window.

The type defines the number of values that can be entered for the property. Following are descriptions of each type.

Single Unrestricted – A single unrestricted property can have only one value, but you can enter any value.

Single Restricted – A single restricted property can have only one value, and you are restricted to selecting that value from a predefined list.

Multiple Unrestricted – A multiple unrestricted property can have multiple values, and you can enter any values.

Multiple Restricted – A multiple restricted property can have multiple values, and you are restricted to selecting the values from a predefined list.

2. In the Property designer window:

- ◆ Enter a name and description for the property
- ◆ Select the **Data Type** for the property value. For example, if you select Boolean, your property value can be only true or false. (Properties with a Boolean data type are automatically set to "single restricted.")
- ◆ In the **Selection Mode** and **Value Range** fields, you can change the type of property. For example, you can change a property from "single unrestricted" to "multiple restricted."

Developing Personalized Applications

Note: Any change to Data Type, Selection Mode, or Value Range removes anything previously entered in the Values field.

- ◆ Use the **Values** field to enter values for "restricted" types or to set the default value(s) for "unrestricted types." Click the ellipsis icon (...) to enter values. (In the Enter Property Value dialog box that appears, click **Add** after each entry, and click **OK** when all values are entered.)
3. Save the file after you have added all the properties you want.

To modify properties and their values

To modify properties and their values, double-click the property set file in the the Application window, click the property you want to modify, and change the values in the Property designer window.

To delete properties

You can delete individual properties from a property set, and you can delete property sets.

To delete a property from a property set, open the property set file, select the property, and press the **Delete** key.

To delete a property set, select the property set file in the Application window and press the **Delete** key.

Related Topics

Property Set Designer window

Creating Segments

Creating Content Selectors

Creating Campaigns

Creating User Profile Properties

Creating Session Properties

Creating Request Properties

Creating Catalog Structure Properties

Personalizing Portal Applications

After you create all necessary personalization conditions, you can create interaction management functionality and surface it in your portals. For example, after you create a user segment to trigger personalization, you can create a content selector that defines the content that is shown to users who are members of the user segment.

Note: Many interaction management development tasks are dependent on other tasks. For example, if you are creating a user segment based on user profile properties, you must first create the user profile properties you want; and if you are creating a campaign and you want to trigger campaign actions based on user segments, you must first create the user segments you want. The following tasks highlight any applicable dependencies.

Creating Placeholders

Dependencies: Placeholders rely on no other file types for creation. However, to set up a placeholder query, the types and content you want to display in the placeholder must be set up in the Virtual Content Repository.

Creating Content Selectors

Dependencies: Content selectors can use user segments, user profile properties, and session and request properties.

Creating Discounts

Dependencies: Discounts can be triggered by and applied to product categories and items. Categories and items are defined in the Virtual Content Repository.

Preparing to Use Campaigns

Dependencies: Campaigns can use user segments, placeholders, and properties from property sets (user profile, HTTP session and request, catalog structure, and events).

Creating Campaigns

Dependencies: Campaigns can use user segments, placeholders, and properties from property sets (user profile, HTTP session and request, catalog structure, and events).

Using the <pz:div> JSP Tag

Dependencies: The <pz:div> tag requires the use of user segments.

After you create interaction management resources, such as placeholders, content selectors, and campaigns, those resources are available to portal administrators for modification in the WebLogic Administration Portal.

Related Topics

Developing Personalized Applications

Overview of Content Management

Setting up Users

Developing Personalized Applications

Designing Interaction Management

Creating Personalization Conditions

Personalization Conditions Reference

Creating Placeholders

A placeholder is a predefined location in a JSP that displays a single piece of Web content retrieved from the BEA Virtual Content Repository. A placeholder uses queries to retrieve and display content.

By default, placeholders support the following MIME types: HTML, XML, plain text, images, and Shockwave files. To display additional MIME types in placeholders, see Supporting Additional MIME Types.

When a user views a Portal Desktop or Portlet containing a placeholder, the placeholder's rules and back-end logic run a query, retrieve zero or more pieces of content matching the query, and display one of the returned items. If no content is retrieved, none is displayed.

The queries for a placeholder come from two different places:

- From the placeholder itself – When you create a placeholder in the placeholder designer, you can also define default queries to run in the placeholder. Default queries run for all visitors, anonymous and authenticated.
- From a Campaign – When you create a campaign in the Campaign Designer, one option in the campaign is displaying targeted, personalized content in a placeholder for a specific type of visitor.

Because a placeholder can contain multiple queries, the placeholder serves as a content bucket that can display a different piece of content each time a user accesses the JSP containing the placeholder. The following procedure includes a step that lets you set a query's priority to increase or decrease the chances that the query is run instead of other queries present in the placeholder.

Placeholders are scoped to the enterprise application, so you can include Placeholders in any JSPs within the enterprise application.

To create a placeholder

1. If your server is not running, start it by choosing **Tools-->WebLogic Server-->Start WebLogic Server**.
2. In the Application window, right-click the **data\placeholders** folder and choose **New-->Placeholder**.
3. In the New File window, enter a name for the placeholder in the **File name** field. Make sure you keep the file extension.
4. Click **Create**. The placeholder designer appears.
5. In the Palette window, drag the **New Query** icon into the placeholder designer to define a default query.
6. In the Property designer window, set values for the following:
 - ◆ **Name** – Enter a name for the query.
 - ◆ **Priority** – Select the relative priority for the query. A query with a higher priority is more likely to be run.
7. In the designer window, click the query's **empty content search** link to define the query. This requires a connection to the Virtual Content Repository, which is set up in the Administration Portal.
8. In the Content Search window, select a Property Set, select a Property within the Property Set, and click **Add**.

The properties you select are content properties (types) rather than property set properties such as user profile or session properties.

Developing Personalized Applications

9. In the Content Search Values window that appears, use one of the following tabs:
 - ◆ **Values** – To define the query based on a comparison to a value you enter. For example, the query could be set to retrieve content with an "investorRiskLevel" property that is marked as "high."
 - ◆ **Properties** – To define the content query based on the property value that is dynamically fed in from another type of property, such as a User Profile property. For example, instead of creating a query based on static content properties, you can create a query that reads in the value of the current user's "investorRiskLevel" to populate the query. The query would be different for each user.
10. Click **Add**. The query descriptor is added in the Content Search window.
11. You can add more value phrases to the query, then set the appropriate option in the **For multiple descriptors** area at the bottom of the window.
12. Click **OK** in the Content Search window.
13. Create additional queries as needed.
14. You can preview the content that will be retrieved by the query in the Content Preview window below the Placeholder Designer. In the **Preview User** field, enter the username of an existing user, such as weblogic.
15. Save the placeholder file.
16. To use the placeholder, add the <ph:placeholder> tag in the relevant JSPs.

Note: As an alternative to Placeholders, you can use the <ad:adTarget> JSP tag to hard-code a content query within the tag.

To modify a placeholder

To modify a placeholder, open its file by double-clicking it in the Application window. Select and modify the appropriate query.

To delete a query or placeholder

To delete a query in a placeholder, open the placeholder file, select the query in the Document Structure window, and press the **Delete** key.

To delete a placeholder, select it in the Application window, press the **Delete** key, and click OK in the confirmation dialog box. Also delete any <ph:placeholder> tags in your JSPs for the deleted Placeholder.

Related Topics

Placeholder Designer window

<ph:placeholder> Tag

Creating Campaigns

Creating Content Selectors

A Content Selector targets users with personalized Web content taken from BEA's Virtual Content Repository. Content Selectors can return and display multiple content nodes.

When a user views a Portal Desktop or a Portlet containing a Content Selector, the Content Selector's rules and back-end logic look for a match of properties such as the the user's profile information. If the properties match the Content Selector rules, the Content Selector runs a query and retrieves and displays all content matching the query.

Content Selectors are scoped to the enterprise application, so you can include Content Selectors in any JSPs within the enterprise application.

To create a Content Selector

Be sure you have added a Data Sync project to your Portal Application. See *Creating a Portal Application and Portal Web Project* for more information.

1. If your server is not running, start it by choosing **Tools**—>**WebLogic Server**—>**Start WebLogic Server**.
2. In the Application window, right-click the **data|contentselectors|GlobalContentSelectors** folder and choose **New**—>**Content Selector**.
3. In the New File window, enter a name for the Content Selector in the **File name** field. Make sure you keep the file extension.
4. Click **Create**. The Content Selector designer appears.
5. In the Property designer window, you can change the display **Name** for the Content Selector.
6. In the **Available Conditions** section of the designer window, select the types of conditions under which the Content Selector will run.

As you select conditions, corresponding links appear in the top of the designer window.

7. Click the corresponding links to create the conditions you selected, and enter the appropriate information.
8. In the designer window, click the query's **empty content search** link to define the query. This requires a connection to the content management system, which is set up in the Administration Portal.
9. In the Content Search window, select a Property Set, select a Property within the Property Set, and click **Add**.
10. In the Content Search Values window that appears, use one of the following tabs:
 - ◆ **Values** – To define the query based on a comparison to a value you enter. For example, the query could be set to retrieve content with an "investorRiskLevel" property that is marked as "high."
 - ◆ **Properties** – To define the content query based on the property value that is dynamically fed in from another type of property, such as a User Profile property. For example, instead of creating a query based on static content properties, you can create a query that reads in the value of the current user's "investorRiskLevel" to populate the query. The query would be different for each user.
11. Click **Add**. The query descriptor is added in the Content Search window.
12. You can add more query descriptors to the query, then set the appropriate option in the **For multiple descriptors** area at the bottom of the window.
13. Click **OK** in the Content Search window.

Developing Personalized Applications

14. Save the Content Selector file.
15. To use the Content Selector, add the `<pz:contentSelector>` tag in the relevant JSPs. The examples in the `<pz:contentSelector>` topic show how to display text content (including text binaries such as HTML files) and non-text binaries (such as graphics).

To modify a Content Selector

To modify a Content Selector, open its file by double-clicking it in the Application window. Modify the appropriate conditions. Click the query link and modify the queries.

To delete a Content Selector query

1. Open the Content Selector file.
2. Click the query link in the Content Selector designer.
3. Select the query in the Content Search window.
4. Click Remove.
5. Click **OK**.

To delete a Content Selector

To delete a Content Selector, select it in the Application window, press the **Delete** key, and click **OK** in the confirmation dialog box. Also delete any `<pz:contentSelector>` tags in your JSPs that reference the Content Selector you deleted.

Related Topics

Tutorial: Showing Personalized Content in a Portlet

How Do I: Create a Personalized Portlet?

Content Selector Designer window

`<pz:contentSelector>` JSP Tag

Personalization Conditions Reference

Creating Campaigns

Developing Personalized Applications

Creating Discounts

Use the WebLogic Workshop Portal Extensions Discount Designer to create discounts that can be used on the items in your catalog or shopping cart.

You can create discounts to globally apply to all users or to be used exclusively in campaigns.

Before You Create Discounts

To create discounts that can be incorporated into your commerce application, you must enable commerce functionality. Also, you are going to create discounts based on product categories or items, you must set up a product catalog against which discounts can be applied. Use the following topics for guidance:

Adding Commerce Services to an Application

Enabling Catalog Management

Creating Catalog Structure Properties

To create a Discount

1. If your server is not running, start it by choosing **Tools-->WebLogic Server-->Start WebLogic Server**.
2. In the Application window, right-click the **data\discounts\DefaultDiscountSet** folder and choose **New-->Other File Types**.
3. In the New File window, select the **Discount** in the right pane.
4. In the **File name** field, enter a name for the Discount. Make sure you keep the file extension.
5. Click **Create**. The Discount Designer appears.
6. In the Discount Designer, click the **Wizard** icon to use the Discount Wizard to create the discount.

OR

7. If you create the discount manually (without the wizard), select the **Discount type** option you want (discount on a single item, on a set of items, or on an entire order. If you select **Set-based discount**, determine whether there is a limit to the number of discounts in the order.
8. In the **Discount terms** pane, click **Add a Trigger** to set the items or order properties for which the discount will be used.
9. Click **Add a Discount** to define a discount percentage or dollar amount.
10. Click **Add a Target** to set the items to which the discount is applied.
11. In the **Property Editor**, set the following properties:

Description	For campaign discounts. The text you enter can be displayed in the shopping cart when the discount is applied.
Usage	Select whether the discount will apply to all users (Stand-alone) or whether it will be used in campaigns (Campaign)
Explanation	For stand-alone discounts only. The text you enter can be displayed in the shopping cart when the discount is applied.
Priority	

Developing Personalized Applications

	The number that determines which discount is used if there are competing discounts (1 is the highest priority).
Overall limit	You can set the number of times a customer can receive the discount. Enter 0 if there is no limit.

Required

Start Date and Stop Date	Set the range of time for the discount to be in effect. Start and stop date must be set to be able to finalize the discount.
---------------------------------	--

12. When the discount is final and ready to use, select the **Finalize this discount** option.
13. Save the discount.

If you created a campaign discount, the discount will be available for use when you create campaigns.

Related Topics

Discount Designer window

Building a Commerce Application

Creating Campaigns

Creating Campaigns

Campaigns are powerful tools for personalization, letting you target users with specific Web content, e-mails, and discounts based on fine-grained rules.

The following topics guide you through the campaign creation process:

Preparing to Use Campaigns

This topic leads you through setting up the infrastructure used by campaigns.

Building Campaigns

This topic shows you how to build a campaign in your portal application.

Related Topics

Personalization Conditions Reference

Creating User Segments

Creating Placeholders

Creating User Profile Properties

Registering Custom Events

Creating Session Properties

Creating Request Properties

Creating Catalog Structure Properties

Building a Commerce Application

Campaign Designer window

Preparing to Use Campaigns

Before you create Campaigns, ensure that you have done the following:

1. Create a Portal Application.
2. If you are using a BEA-compatible third-party content management system, make sure you have connected it to the BEA Virtual Content Repository using the WebLogic Administration Portal.

If you are using the Virtual Content Repository directly, define types and add the content you want to use in your campaigns.

3. Create Placeholders – If you will display personalized Web content with Campaigns, create the placeholders that will retrieve and display the Web content.
4. Create User Segments – If you want to trigger Campaigns based on users who are grouped dynamically based on specific characteristics, create User Segments.
5. Create Property Sets – If you will trigger Campaigns based on properties of a user, event, HTTP session, or HTTP request, perform any of the following relevant procedures:
 - ◆ Creating User Profile Properties
 - ◆ Registering Custom Events
 - ◆ Creating Session Properties
 - ◆ Creating Request Properties
6. Create and Store E-mail Messages.

If you are using your own e-mail server to send Campaign e-mails, perform the following steps:

- ◆ In the WebLogic Administration Portal Server Administration tools, select Server Admin, add a mail service configurable item, and enter the SMTP Host Name for your mail server's outgoing mail.
 - ◆ Create your predefined e-mail messages. E-mail messages can be in any of the following formats: TXT, HTML, JSP, or XML (with style sheets). Store them in the following location: `<Web-app-project>\campaigns\emails`.
7. If you are going to create discount campaign actions, perform the following steps:
 - ◆ Add commerce services to your portal application.
 - ◆ Set up a shopping cart using the WebLogic Portal commerce API.
 - ◆ Create a catalog in the Virtual Content Repository.
 - ◆ Use the WebLogic Portal catalog classes in the commerce API to surface catalog items from the Virtual Content Repository and identify them with "categories" and "SKU" numbers.
 - ◆ Create discounts and use the commerce API to surface the discounts in your shopping cart. If desired, use the API to surface the discount's description next to the discount amount displayed in the shopping cart.

Related Topics

Building Campaigns

Campaign Designer

Building Campaigns

After you have set up all the necessary pieces used in campaigns (as described in Preparing to Use Campaigns), you are ready to build campaigns.

The primary functional units in campaigns are actions. Actions perform specific personalization tasks and are triggered by specific conditions you set. Actions are grouped into scenarios.

For example, a campaign action can be triggered by the following conditions: "When a user logs in between January 1 and January 31, and that user is a member of the 'Non-manager' user segment, trigger the campaign to do something." The action could then do the following: "When the campaign is triggered, send an automatic e-mail reminding the user to complete an annual performance review."

A campaign can contain multiple scenarios, each of which can contain multiple actions. When each action runs depends on its conditions. You can also trigger all actions in a scenario to run by assigning user segments to the scenario. The actions run when users belonging to those user segments log in.

Developers create campaigns and administrators use those campaigns as templates to modify campaign characteristics and create new campaigns with similar characteristics.

The following steps guide you through building a campaign.

1. Creating a Campaign File and Setting Campaign Properties

1. If your server is not running, start it by choosing **Tools**—>**WebLogic Server**—>**Start WebLogic Server**.
2. In the Application window, right-click the **data\campaigns** folder and choose **New**—>**Campaign**.
3. In the New File window, enter a name for the campaign in the **File name** field. Make sure you keep the file extension.
4. Click **Create**. The Campaign Designer appears.
5. In the Property Editor window, set the following properties:
 - ◆ **Name** – Read-only. The name of the Campaign (the campaign filename).
 - ◆ **Description** – Enter a detailed description of the Campaign. The description is appended to the text in the Discription window.
 - ◆ **Sponsor** – Enter the name of the organization or person sponsoring the Campaign.
 - ◆ **Active** – Set the value to **true** if you want the campaign to be run. Set the value to **false** if you do not want the Campaign to run.
 - ◆ **Start Date** – Click the ellipsis icon [...] and set the date you want the Campaign to start (in your time zone).
 - ◆ **Stop Date** – Click the ellipsis icon [...] and set the date you want the Campaign to end (in your time zone).
 - ◆ **Description (Goal)** – Enter a description about the goals that will end a campaign prior to its stop date.
6. In the Property Editor window, optionally use the **Goal Definitions** property to end a campaign prior to the campaign stop date when specific images are viewed or clicked in a portal. Click the ellipsis icon [...] to launch the Edit Campaign Goals window and select the content and the number of viewings (impressions) or clicks (click-throughs) required to end the campaign.

In the **Add Path to Goal** field, enter the repository path to the campaign content that can end the campaign. For example, /BEA Repository/campaigns/ad1. Click **Add**. Add paths for all content nodes

Developing Personalized Applications

you want to use. Click **OK** when you are finished.

2. Adding a Scenario to a Campaign

1. From the Palette window, drag the **New Scenario** item into the Campaign designer.

Note: There are also four scenario templates you can use that contain predefined actions and conditions. If you drag one of these scenarios into the Campaign designer, see the Description window for details on each.

2. In the Property designer window, set the following properties:
 - ◆ **Name** – Enter a name for the scenario.
 - ◆ **Description** – Click the ellipsis icon [...] and enter a detailed description of the scenario. Your description is appended to the text in the Description window.
 - ◆ **Segments** – If you want all actions in the scenario to run if the user is a member of one or more User Segments, click the ellipsis icon [...] and select the User Segments you want to use.
3. Add more scenarios as needed.

3. Adding an Action to a Scenario

Add the actions you want a scenario to perform. You can add multiple actions in each scenario.

Adding a Content Action

Content actions retrieve Web content from a content repository and display the selected piece of content in a predefined placeholder on a JSP.

1. From the Palette window, drag the **New Content Rule** icon onto the appropriate scenario icon in the Campaign designer.
2. In the Property designer window, enter a name for the action.
3. In the action, click the **all** link to toggle back and forth between **any** and **all** to determine which conditions will trigger the action.
4. Click the **[empty content search]** link to define the query that will determine which content is retrieved.
5. In the Content Search window, select a Property Set, select a Property within the Property Set, and click **Add**.
6. In the Content Search Values window that appears, use one of the following tabs:
 - ◆ **Values** – To define the query based on a comparison to a value you enter. For example, the query could be set to retrieve content with an "investorRiskLevel" property that is marked as "high."
 - ◆ **Properties** – To define the content query based on the property value that is dynamically fed in from another type of property, such as a User Profile property. For example, instead of creating a query based on static content properties, you can create a query that reads in the value of the current user's "investorRiskLevel" to populate the query. The query would be different for each user.
7. Click **Add**. The query descriptor is added in the Content Search window.
8. Click **OK** in the Content Search window.
9. Click the **[placeholder name]** link in the action, and select the placeholder(s) that will display content when the scenario is triggered.
10. If you want the content to stop being displayed prior to the end of the campaign, click **the campaign ends** link to determine when the content will stop being displayed.

Developing Personalized Applications

11. To increase the chances that content from the query will be displayed, click the ***Do not remove any other content*** link and determine the existing content that will be removed from the designated placeholder(s) when the action runs.
12. To set the priority that the query will be run compared to other queries that may exist in the placeholder(s), click the ***Normal*** link and select a priority. Higher priorities give the query a greater chance of being run.
13. In the Available Conditions pane, select the conditions you want to trigger the action. When you select a condition, a corresponding link appears in the action area. Click the link to define the condition.
14. Save the Campaign file.

Adding an E-mail Action

E-mail actions send a predefined e-mail to the user.

1. From the Palette window, drag the ***New E-mail Rule*** icon onto the appropriate scenario icon in the Campaign designer.
2. In the Property designer window, enter a name for the action.
3. In the action, click the ***all*** link to toggle back and forth between ***any*** and ***all*** to determine which conditions will trigger the action.
4. Click the ***[server url]*** link to select the e-mail message to send, and enter a ***Subject*** and optional default e-mail address. Click ***Preview*** if you want to preview the e-mail.
5. Click ***OK***.
6. In the Available Conditions pane, select the conditions you want to trigger the action. When you select a condition, a corresponding link appears in the action area. Click the link to define the condition.
7. Save the Campaign file.

Adding a Discount Action

Discount actions give the user a discount on items, orders, or shipping.

1. From the Palette window, drag the ***New Discount Rule*** icon onto the appropriate scenario icon in the Campaign designer.
2. In the Property designer window, enter a name for the action.
3. In the action, click the ***all*** link to toggle back and forth between ***any*** and ***all*** to determine which conditions will trigger the action.
4. Click the ***[discount]*** link and select the Discount(s) you want to apply. Enter an option ***Discount Explanation*** that can appear in the shopping cart next to the displayed discount.
5. Click ***OK***.
6. In the Available Conditions pane, select the conditions you want to trigger the action. When you select a condition, a corresponding link appears in the action area. Click the link to define the condition.
7. Save the Campaign file.

Related Topics

Preparing to Use Campaigns

Personalization Conditions Reference

Developing Personalized Applications

Creating User Segments

Creating Placeholders

Creating User Profile Properties

Registering Custom Events

Creating Session Properties

Creating Request Properties

Creating Catalog Structure Properties

Building a Commerce Application

Campaign Designer window

Using Session, Request, and Event Properties in Campaigns

Campaign scenario rules are evaluated only when a single event occurs for which the campaign listener is configured. When the event is triggered, the event takes a snapshot of the current session properties, the single request property (contained in the session), and the event properties (contained in the request). The snapshot taken by the event is in the form of a Request object, which the event passes to the campaign service for evaluation. If the values in that snapshot evaluate to true against any campaign action rules, those campaign actions are triggered.

When campaign actions are not triggered as expected using session, request, and event properties, one or more of the following is usually to blame:

- No event was fired that the Campaign service was listening for.
- The session, request, or event properties contained in the Campaign rule were not part of the Request object snapshot taken when the event was fired.
- In Campaign rules that are defined so that all conditions must apply for the campaign action to be triggered, one or more of the conditions evaluated to false.

Consider the following Campaign action rules as created in the E-Business Control Center:

When ***all*** of these conditions apply:

an HTTP request has the following properties:

RequestPropertyOne is equal to 'success'

any of the following events has occurred:

SessionLoginEvent

Do the following [Ad action, e-mail action, or discount action].

The rule will be evaluated only if an event for which the campaign service is listening occurs. (This event need not be used directly in the campaign rule.) For example, if the campaign service is configured to listen for the BEA-provided UserRegistrationEvent (which it is by default), then when a UserRegistrationEvent occurs the event takes a snapshot of the Request object and the Campaign rules are evaluated.

Here is how the previous Campaign action rules would be evaluated:

- Is there a request property called RequestPropertyOne with a value of success?
- Am I a SessionLoginEvent?
- Are all of these conditions true?

Because a UserRegistrationEvent woke up the campaign service and took a snapshot of the request object, the campaign action will not be triggered, because the rule requires that all of its conditions evaluate to true. The SessionLoginEvent rule is false (because it was the UserRegistrationEvent that woke up the campaign service).

Developing Personalized Applications

If the rule was defined differently so that **any** of the conditions evaluating to true would trigger the action (rather than **all** conditions), the campaign action would have fired if the request property evaluated to true.

To use session or request properties to trigger campaign actions, make sure you do the following:

- In the JSP containing the event to be fired, get the request attribute through a variable or set it directly in the JSP.
- In the JSP containing the event to be fired, get/set any event properties you want to use.
- If you want to use session properties to trigger campaign actions, make sure the firing event is in the same session containing the session properties you want to use.

Related Topics

Preparing to Use Campaigns

Building Campaigns

Personalization Conditions Reference

When you create User Segments, Content Selectors, and Campaigns to target users with personalized Web content, send them automatic e-mail messages, or give them discounts automatically, you select the conditions that trigger the personalization actions to occur.

The follow table describes conditions you can select in the User Segment, Content Selector, and Campaign designers. Some conditions appear only in the Campaign designer.

When you select a condition in one of those designer windows, a corresponding hyperlink appears in the designer, letting you set the values for the condition.

The visitor is a member of a predefined user segment	If the visitor to your Web site belongs to a predefined user segment, execute the specified action. For example, if the visitor is a Gold Customer (user segment), show the visitor a specific piece of Web content (action).
The visitor has specific characteristics	If a visitor's characteristics are compared to values you specify and those comparisons evaluate to true, execute a specified action. For example, if the visitor's salary (characteristic) is greater than or equal to (comparison) \$50,000 (value), send the visitor an e-mail (action).
The HTTP session has specific properties	If the HTTP session's properties are compared to values you specify and those comparisons evaluate to true, execute a specified action. An HTTP session is information your organization may want to track about a visitor's browsing session on the Web site.
An HTTP request has specific properties	If the HTTP request's properties are compared to values you specify and those comparisons evaluate to true, execute a specified action.
An event has occurred (Campaigns only)	If a specified event occurs, execute the specified action. For example, if the user has logged in (event), display an ad that matches their interests (action). Events may be selected from a number of predefined event types.
An event has specific characteristics (Campaigns only)	If the event's characteristics are compared to values you specify, and those comparisons evaluate to true, execute a specified action. For example, if a user adds more than 5 items to their shopping cart (event), give the user a 10% discount (action).
The date is	If the current date is equal to the one you specify, execute a specified action. For example, if the date is equal to July 22, 2001, send users an e-mail about an upcoming sale (action). The current date refers to the date at the point that the condition is evaluated for a given user visiting the Web site. See About Dates and Times.
It is after a given date	If the current date is after a date you specify, execute a specified action. For example, if the date is after December 18, 2000, offer users a discount (action). The current date

Developing Personalized Applications

	<p>refers to the date at the point that the condition is evaluated for a given user visiting the Web site.</p> <p>See About Dates and Times.</p>
It is after a given date and time	<p>If the current date and time is after a date and time you specify, execute a specified action. For example, if the date and time is after July 22, 2001 at 3 p.m., send users an e-mail about an upcoming sale (action). The current date and time refers to the date and time at the point that the condition is evaluated for a given user visiting the Web site.</p> <p>See About Dates and Times.</p>
It is between two times	<p>If the current time falls within a range of times you specify, execute a specified action. For example, if the time is between 3 p.m. and 5 p.m., offer users a discount (action). The current time refers to the time at the point that the condition is evaluated for a given user visiting the Web site.</p> <p>See About Dates and Times.</p>
It is between two dates	<p>If the current date falls within a range of dates you specify, execute the specified action. For example, if the date is between December 18, 2000 and December 18, 2001, show users a sale ad (action). The current date refers to the date at the point that the condition is evaluated for a given user visiting the Web site.</p> <p>See About Dates and Times.</p>
It is between two dates/times	<p>If the current date and time fall within a range of dates and times you specify, execute the specified action. For example, if the date and time is between July 22, 2000 at 3 p.m. and July 22, 2001 at 3 p.m., show users a sale ad (action). The range of dates is inclusive. The current date and time refers to the date and time at the point that the condition is evaluated for a given user visiting the Web site.</p> <p>See About Dates and Times.</p>
There is a specific item in the shopping cart (Campaigns only)	<p>If one or more items (identified by SKU number) are in the shopping cart, execute the specified action. For example, if the user adds a specific item to the shopping cart, give the user a discount on another item. To use this condition your portal application must have commerce services enabled and a shopping cart configured. For information commerce services, see Building a Commerce Application.</p>
There is an item from a given category in the shopping cart (Campaigns only)	<p>If an item in the shopping cart belongs to one or more categories in the catalog, execute the specified action. For example, if an item in the shopping cart belongs to the "books," "toys," or "food" categories, give the user a discount. To use this condition your portal application must have commerce services enabled and a shopping cart</p>

Developing Personalized Applications

	configured. For information commerce services, see Building a Commerce Application .
The value of items in the cart is a certain amount (Campaigns only)	If the value of the items in the shopping cart is equal to, less than, or greater than a certain value, execute the specified action. For example, if the items in the shopping cart total more than \$100, give the user a discount on shipping. To use this condition your portal application must have commerce services enabled and a shopping cart configured. For information commerce services, see Building a Commerce Application .
A random number falls within a given range (Campaigns only)	When someone visits your Web site, the system assigns him a random number from 1 to 100. If the visitor's random number falls within the numeric range you set, execute this action. For example, if you specify a range of 1 to 50, the action will be executed for approximately 50% of the target visitor population.

About Dates and Times

When you set date and time conditions, the dates/times you set represent the time in your region. For example, if you are creating a campaign action that will be triggered at 8 p.m., that means 8 p.m. in your region. For a time zone that is two hours behind you, the action will be triggered at 6 p.m. in that time zone.

This also affects dates you set. The date you set becomes effective at midnight in your time zone. In a time zone that is six hours ahead of yours, that date becomes effective for that time zone at 6 p.m. your time the day before.

Time changes also affect time-triggered actions. For example, say you create a campaign that begins October 1 at noon and ends October 31 at noon. If a change to standard time (one hour earlier) occurs on October 29, the campaign will actually end on October 31 at 11 a.m. So if you want the campaign to end at noon on the new standard time, set the end time to 1 p.m.

Because of the different dates and times on which actions will be triggered around the country or world, it is important to tell users that dates and times are effective for your time zone. This type of information allows users to calculate when in their time zone they can take advantage of your promotions.

Related Topics

[Creating User Segments](#)

[Creating User Profile Properties](#)

[Creating Campaigns](#)

[Creating Content Selectors](#)

[Building a Commerce Application](#)

