BEA WebLogic Portal®

User Management Guide

Version: 10.0
Revised: May 2007

Contents

9. Adding and Managing Users

10.Editing User Profile Property Values

Part IV. Production

CHAPTER 1

Introduction

This guide describes how to add and manage users and groups in your portal application. A user can be a portal administrator or a portal end user (a visitor or a registered user). Each user is given a unique identity within a security realm. A group is a collection of users who usually have something in common, such as working in the same department. Create users and groups to simplify user management and administration.

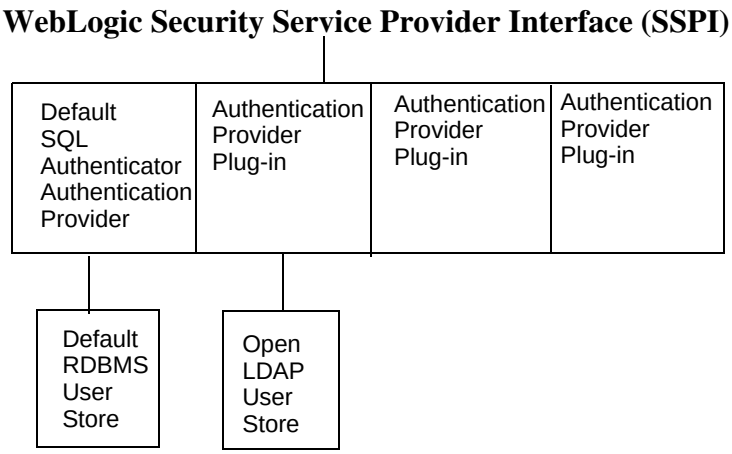
You can add and manage portal users and groups using any of the following methods:

- Connect to your own external user store to retrieve existing users and groups
- Use the default user store contained in BEA WebLogic Server
- Let users register and add themselves to your portal
- Create new users and groups

These internal and external user stores are accessed from BEA Web Logic Portal through the WebLogic Security Service Provider Interface (SSPI) Authentication Providers. An authentication provider uses the username and password to find and authenticate a corresponding user in the user store.

Figure 1-1 shows how authentication providers and user stores work together.

Figure 1-1 Authentication Providers and User Stores



The guide also contains instructions for creating user profiles to retrieve other information about users, such as phone number, e-mail address, country, and so on. This user profile information is used to create personalization features that display customized content or guide users through a process.

WebLogic Portal uses the default PointBase Relational Database Management System (RDBMS) database that ships with Weblogic Server. During the installation, WebLogic Portal installs demonstration user accounts and data into the WebLogic Server default database. You can also store user information externally and manage these attributes through a Unified User Profile (UUP). You can also track anonymous users (users who have not yet registered) to get information about these portal visitors and display customized content for them.

This chapter includes the following sections:

- Introducing Users and Groups
- User Management in the Portal Life Cycle
- Getting Started

Introducing Users and Groups

This section contains the following topics:

Understanding Users

BEA WebLogic Portal supports two types of users:

- **Portal administrators** – Manage portal content and portal users, and build portals with existing portal resources
- **Portal end users (visitors)** – Use the portals assembled by the portal administrator

A portal administrator can simplify management tasks by combining users into groups and subgroups. An administrator with full management rights can add, edit, remove, or move users and user profile properties programmatically in BEA Workshop for WebLogic Platform or in the WebLogic Portal Administration Console.

You can access existing users through internal or external user stores. An internal RDBMS user store is included when you install WebLogic Server and WebLogic Portal, and it uses the *SQLAuthenticator* as the authentication provider. You can also access more than one external user store that contains your users, so you can select users and groups from multiple user stores. For instructions on setting up external user stores, see the *Security Guide*.

In addition to retrieving users stored in the external user store, you can also add users to the user store (if the user store is writable) using WebLogic tools or you can add users directly to the user store itself. See "Accessing Users" on page 1-5 for more information on authentication providers and user stores.

When you add a user, WebLogic Portal creates a user profile that contains the user's identity (name and password). Other user properties (such as address, phone number, e-mail, hobbies, and so on) can be added to the user profile to set up personalization and define rules for delegated administration and visitor entitlement. Personalization allows you to target users with customized content that enhances the user's interaction with your portal. Delegated administration and visitor entitlement control what the user can see or do in your portal.

If you have external data stores that contain additional information about users and groups, accessing those properties involves separate development steps to create a UUP. For more

information on integrating these external systems with user profiles, see Chapter 6, "Configuring a UUP".

WebLogic Portal also contains tracking tools to find out more about visitors to your portal, such as the date and time they visit and information in the request or session. This anonymous user tracking helps you present personalized content to an unregistered user over multiple sessions.

WebLogic Portal uses the following terminology to describe users:

- **User ID** – For an anonymous user, a user ID is a unique number that follows the actions of tracked anonymous profiles. A User ID is used because the user is not registered.
- **Credential –** An authentication token, such as a password or certificate.
- **Principal** – When capitalized, Principal is a Java object. For example:
  - A Principal object represents the abstract notion of a principal, which can represent an entity, such as an individual, corporation, or a login ID.
  - A Subject object (which maps to a user) can hold multiple Principals.
  - In WebLogic Security, Principal objects map to groups and also IDs.
- **Username** – A user attribute, such as a user's name. It can also be a string (for example, `username = String` to display on the Welcome page).

Understanding User Profile Types

A user profile can store and retrieve additional information about users, such as phone number, e-mail address, country, and so on. You can use this information to create personalization features that display customized content or guide users through a process.

WebLogic Portal supports three kinds of user profiles:

- **Registered** – A fully-registered user who can be authenticated with the system.
- **Anonymous** – A user who does not have an identity with the system.
- **Tracked Anonymous** – A user who is recognized by WebLogic Portal but is not registered. The user has an identity but cannot authenticate with the system.

The user profile is stored in the session by default. The profile is usually initialized on the user's first visit to the portal. At the same time, an anonymous profile or a tracked anonymous profile is initialized in the session on first access (if user tracking is enabled and the request has a valid tracking cookie). See Chapter 7, "Setting Up Anonymous User Tracking" for instructions on setting up anonymous users.

If the user logs in (authenticates), the session initialization process switches the profile with the user's registered profile. If the user registers with the system, a registered profile is created that is initialized with values from the tracked anonymous profile, and the profile is placed in the session.

Implementing Security and User Profiles

In addition to storing valuable information (attributes) about users, you can also use user profiles to implement portal security. When you use visitor entitlement to limit user access to portal resources (desktops, books, pages, portlets, and other resources), the visitor entitlement policy can be defined with user properties. For example, you could create a visitor entitlement role called *manager* that determines if a user who logs in has an *employee_type* property with a value of *manager***,** then that user belongs to the *manager* role. You could then select a portlet and set its visitor entitlement to the *manager* role. When a manager logs in, the manager sees the portlet. If a non-manager logs in, the user does not see the portlet.

You can also create delegated administration roles using user profile properties. See the *Security Guide* for instructions.

Accessing Users

When you add a user to WebLogic Portal, you add the user to the user store. WebLogic Portal supports two places (both of which use SSPI) to store users:

1. **Internal User Store** – You can use the default *SQLAuthenticator* authentication provider to access the RDBMS user store that comes with WebLogic Server. The default RDBMS user store contains users and group membership. Security roles and policies are stored separately in the Lightweight Directory Access Protocol (LDAP) server.

   The default RDBMS user store supports the following access and storage functions:

   – Performs read and write access by the WebLogic Security providers

   – Creates, reads, updates, and deletes entries in the RDBMS server

Figure 1-2 The SQLAuthenticator is the Default Authentication Provider

Note: The RDBMS user store that shipped with WebLogic Portal prior to version 9.2 has been deprecated and it not automatically upgraded to 9.2 or 10.0. If you

installed and used the Portal version of the RDBMS user store, you can upgrade it with a migration script. See the *Upgrade Guide* for instructions on running the script.

2. **External User Store** – WebLogic Portal also supports using external user stores that contain users and group information. External user stores can include OpenLDAP, Netscape iPlanet, Novell NDS, or Microsoft Active Directory, and other RDBMS user stores.

   Using more than one user store can help you perform the following portal administration tasks:

   – Users in external user stores can log into your portal desktops.

   – When external users and groups appear in the Administration Console, you can define access to applications through personalization rules that are based on users and groups and their profile properties.

   – You can entitle users in an external user store to specific portal resources when they log in. You can define visitor entitlement rules based on external users and groups.

Note: WebLogic Portal supports the use of multiple user stores or authenticators. When you use more than one authenticator, you can configure each authenticator differently to allow flexibility in authenticating the users. The authenticators can require the user to be authenticated in one authenticator, some of the authenticators, or all of the defined authenticators (depending how they were configured).

You can add users to an external user store with WebLogic Server Administration Console, WebLogic Portal Administration Console, or by adding a user directly to the user store.

Using Groups and Group Hierarchy

A group is a collection of users. Creating groups and placing users in groups can help the portal administrator simplify user management and administration. A subgroup (child group) is a subset of a higher-level group (parent). Organizing users into groups and subgroups that look like your organization can make it easier to manage users.

Creating groups and subgroups can help you perform the following tasks:

- Organize a related group of users, such as a department, team, or regional office
- Locate users
- Set up delegated administration and visitor entitlement to control what users can see in your portal and the actions they can perform
- Work with large sets of users at one time

Administrators with full group management rights can add, remove, move, and change group profile properties in the WebLogic Portal Administration Console. For more information, see "Editing a Group Profile" on page 8-8.

See Chapter 2, "Planning a User and Group Strategy" for detailed information on developing a user management strategy that can save you time during the Development, Staging, and Production phases.

User Management in the Portal Life Cycle

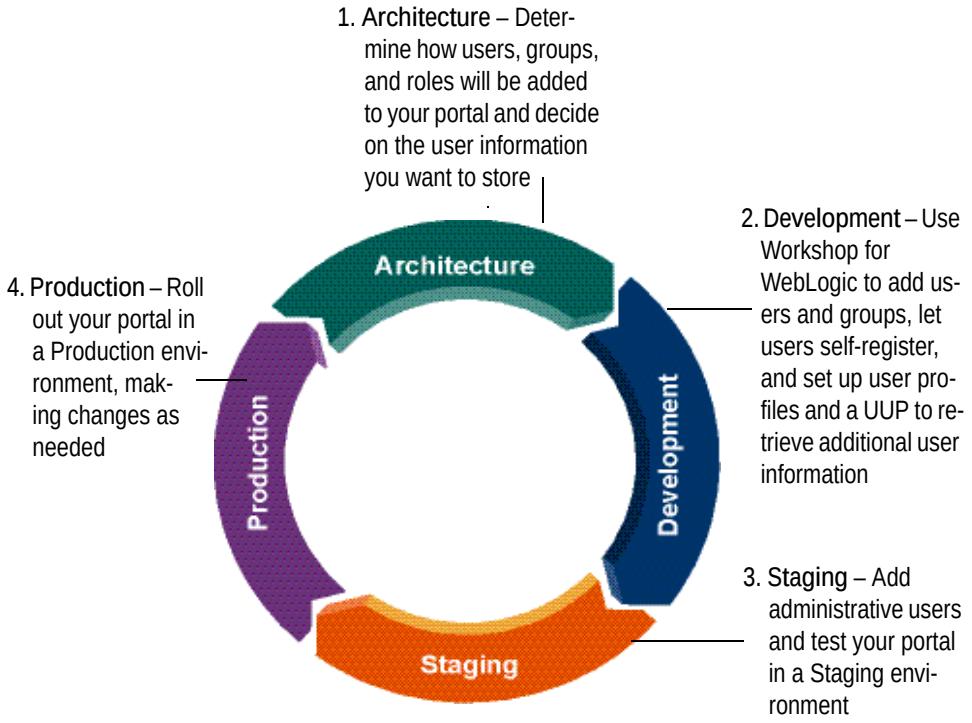
This section contains the following topics:

- Architecture
- Development
- Staging
- Production

The tasks in this guide are organized according to the portal life cycle. The portal life cycle contains four phases: Architecture, Development, Staging, and Production. Adding and managing users and groups is part of the portal life cycle. For more information about the portal life cycle, see the *WebLogic Portal Overview*.

Figure 1-3 lists some of the user and group management tasks in each phase.

Figure 1-3 Users and Groups in the Four Phases of the Portal Life Cycle



Architecture

In the Architecture phase, you plan how to organize and access users and groups stored in internal or external user stores. You can decide if you want users to register themselves when they visit your portal application. If you plan to retrieve user profile information from multiple systems, you can use UUP to perform this integration. If you want to edit that user profile information in the Administration Console, you must make the information writable in the user store. Your strategy for organizing users and groups is different from the strategy you use to create roles and visitor entitlement. Roles and entitlements are used to control what users can view and do in your portal.

The following chapter provides guidance on Architecture tasks:

- Chapter 2, "Planning a User and Group Strategy"

Development

In the Development phase, developers use Workshop for WebLogic to create JSP tags and controls that allow you to add users or let users add themselves to your portal. Developers can also programmatically create groups and subgroups, set up user profiles and properties, create a UUP, and track anonymous users. Developers can choose to implement user roles with declarative security with roles and entitlement (using the isAccessAllowed() method) or programmatic security (with the isUserInRole() method). Declarative security is easy to change and manage. Programmatic security is often coded into portlets, books, and pages and is more difficult to modify.

Tools: BEA Workshop for WebLogic Platform and the Java API.

The following chapters provide guidance on Development tasks:

- Chapter 3, "Adding Groups with JSP Tags and Controls"
- Chapter 4, "Adding and Updating Users with JSP Tags and Controls"
- Chapter 5, "Creating and Updating User Profiles"
- Chapter 6, "Configuring a UUP"
- Chapter 7, "Setting Up Anonymous User Tracking"

Staging

In the Staging phase, you can add administrative users to a staging environment to test the functionality you created in the Development phase. You can also test user profiles and UUP in this phase to verify that they are collecting additional information about users. The Development and Staging phases often occur simultaneously. You might move iteratively between these two phases, developing and then testing what you created. If you return to the Development phase and make changes, you must redeploy your portal application to see the changes in the Staging phase. If you plan to use more than one development team to create your portal, determine how you will merge and assemble the code.

The following chapters provide guidance on Staging tasks:

- Chapter 8, "Adding and Managing Groups"
- Chapter 9, "Adding and Managing Users"
- Chapter 10, "Editing User Profile Property Values"

Production

After you test your portal application in the Staging phase, use the Production phase to perfect your production environment. In the Production phase, for example, you might use the Administration Console to fine tune your portal application, such as adding or moving a group or an administrative user, removing a user or group, adding a desktop, and creating community users and groups.

Getting Started

If you are new to portal development, see the *WebLogic Portal Overview* for more information about the portal life cycle.

You can also consult the following information:

- *Security Guide*
- *Interaction Management Guide*

Part I Architecture

Part I includes the following chapter:

- Chapter 2, “Planning a User and Group Strategy”

This section contains guidelines to help you plan the structure of your users and groups, determine where user and group information is stored, and decide how to retrieve this information. Developing a user management strategy early in your planning can save you time during the other phases of the portal life cycle.

Many portals access users and groups that are stored in existing user stores. WebLogic Portal can plug into these user stores and you can develop roles based on the existing groups in the user store. Making groups role-based to control what each group can do and see in your portal will save you time later. See the *Security Guide* for more information on creating roles and how to manage roles for more than one development environment (offices in New York City and Bangalore, for example).

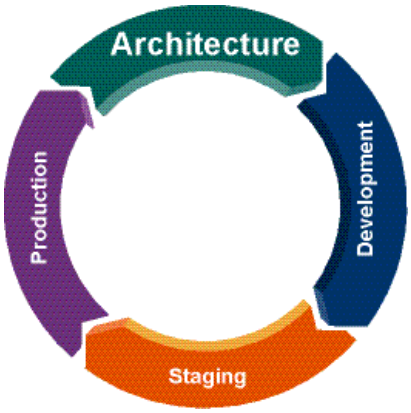
You can also use JSP tags and controls to create the ability for users to register themselves and add themselves to a group when they visit your portal. You might decide you want to add new groups and users in the Administration Console. Portal administrators can also perform these tasks with a WebLogic Server command-line scripting tool called WebLogic Scripting Tool, or they can add users and groups with the Administration Console.

You should also determine if you want to be able to collect and edit information about a user (user profile properties), and if those properties should be read-only. If additional user information already exists in other user stores, you can create a UUP to manage that information.

You should decide if you want to track anonymous users to learn more about their actions in your portal.

The users and groups that you implement in a test or staging environment might be different than the users you access on a production system.

For a description of the Architecture phase of the portal life cycle, see the *WebLogic Portal Overview*. The portal life cycle is shown in the following graphic:



CHAPTER 2

Planning a User and Group Strategy

Developing a plan to structure your users and groups can save you time later during other phases in the portal life cycle. You must determine where your existing user and group information is stored, how to retrieve this information, if you want to allow users to self-register, and if you will need to add new users to your portal.

You must also determine if you want to store and retrieve user profile information to use in personalization. You should consider what each group can see or do in your portal, and plan your roles accordingly. If you decide you want to be able to edit user profile information later, you must configure the user store to be writable.

This chapter includes the following sections:

Developing a User and Group Management Strategy

You should plan your role structure at the same time you plan your strategy for users and groups. Making groups role-based to control what each group can do and see in your portal will save you time later.

Use the following steps as guidelines to develop a user and group strategy:

1. Plan the structure of your groups and subgroups. If your users are located in an external user store and it contains groups, you can bring those existing groups into WebLogic Portal. See Planning Your Groups and Group Hierarchy for more information on categorizing groups and users. Plan the roles for your groups (see the *Security Guide* for more information on roles).
2. Determine where your users are stored and how to access them. Users can be stored in an internal user store (the default RDBMS user store) or in an external user store. For a detailed description of authentication providers and user stores, see "Accessing Users" on page 1-5.
3. Decide if you will create groups and add users programmatically or if you will use the Administration Console. You can use the WebLogic Scripting Tool to retrieve large numbers of existing users and groups. For instructions, see the *BEA WebLogic Server WebLogic Scripting Tool Guide*. You can also create JSP tags and controls that let users register themselves when they go to your portal application.
4. If necessary, you can perform the following tasks with the Administration Console or with the Server Administration Console:

   a. Create groups and subgroups

   b. Add users

   c. Add users to groups

   For instructions on creating groups and users in the Administration Console, see Chapter 8, "Adding and Managing Groups" and Chapter 9, "Adding and Managing Users".
5. Determine if you want to collect and integrate information about users (user profile properties) into your portal.You can use user profile information to target users with personalized content, e-mails, pre-populated forms, and discounts based on the rules you set up. Personalized content can also help guide your users through a process in your portal. For instructions, see Chapter 5, "Creating and Updating User Profiles" and Chapter 10, "Editing User Profile Property Values".
6. If user profile properties are stored externally, create a UUP to retrieve additional user profile information from the external user stores. You can use the properties to define rules for personalization, delegated administration, and visitor entitlement. For more information on when to use UUP, see "Planning to Use a UUP" on page 2-6. For instructions on setting up UUP, see Chapter 6, "Configuring a UUP".
7. After your users and groups appear in the Administration Console, you can place users and groups in roles for delegated administration and assign visitor entitlement. If you are using more than one user store, you might have a group in one user store with a name that is the same as a group in another user store. When you set delegated administration on a group, an

administrator in that delegated administration role can manage that group in all providers that contain that group (if the administrator also has delegated administration rights to those other providers). See the *Security Guide* for instructions on assigning roles and entitlement.

Planning Your Groups and Group Hierarchy

You can structure your users into groups and subgroups that look like your organization, which makes it easier to manage users. A group can be a related collection of users, such as a department, team, or regional office. A user can belong to more than one group, and groups can belong to other groups. Plan the roles you plan to associate with each group, to control what each group can see and do.

If you are using an external user store and it contains groups, you can build a group hierarchy tree to retrieve its group structure and view it in the Administration Console. See Chapter 8, "Adding and Managing Groups" for instructions.

Administrators can also create and change a group profile, which is a schema that determines the data you collect and store about a specific group of users. The properties in the group profile contain information about the group that is stored in editable fields in the Administration Console.

You can edit properties for groups, but users belonging to those groups do not automatically inherit the group properties you specify. If a user belongs to two groups, for example, and you have edited each group's properties, you must specify which set of group properties the user should inherit. For more information, see Chapter 8, "Adding and Managing Groups".

After you determine your group structure, you can programmatically create, move, and delete groups in or the Administration Console.

After you create your groups, you can tie the groups to roles and visitor entitlement to control what users see and experience in your portal.

This section contains the following topics:

- Organizing Users into Groups
- Using an Existing Group Structure

Organizing Users into Groups

For instructions on putting users into groups, see "Placing Users in Groups" on page 4-4 and "Placing Users in Groups" on page 9-7.

The *Everyone* group is a built-in group in Administration Console that you cannot modify. All users, including anonymous users, belong to this group.

The first step in setting up your groups is to determine your group hierarchy. For example, you might have a top-level group called *AllEmployees* that contains all of the employees in your company. The *AllEmployees* group might then contain other groups, each of which contains just the employees of offices in different geographic locations.

This categorization is shown in the following example:

AllEmployees (group containing all employees)

- *NewYork (group containing employees in New York)*
  - *Executive*
  - *Marketing*
  - *Human Resources*
  - *Sales*
- *SanFrancisco (group containing employees in San Francisco)*
- *Seattle (group containing employees in Seattle)*
- *Denver (group containing employees in Denver)*

When you create a group, you create an empty group to which you can add an infinite number of users. You can then classify the users in the group by mapping the group to an entitlement role; see the *Security Guide* for instructions.

You can use the Administration Console to add a user to a group; see "Placing Users in Groups" on page 9-7. You can also use BEA Workshop for WebLogic Platform to add a user to a group; see "Adding a User to a Group with a JSP Tag" on page 4-5.

If you are using an external user store, you should determine if there are existing groups in that user store and match those groups to the groups you are creating.

Using an Existing Group Structure

If the user store you are accessing already has users organized into groups, you can build a hierarchical tree in the Administration Console that matches the existing group structure. If you are using WebLogic Portal Server's default RDBMS user store, you do not need to build a group hierarchy tree. See "Using a Group Hierarchy Tree" on page 8-2 for instructions.

Working with a User Store

You can use the default RDBMS user store that comes with WebLogic Portal, or you can use a user store outside of WebLogic Portal. Examples of an external user store include OpenLDAP or Netscape's iPlanet. You can connect that provider to WebLogic Portal and the users in that external user store can log into your portal.

This section contains the following topics:

Editing User Information

If you want to be able to edit information about your users, you must configure your external user store to allow write access. If the user store is writable, you can add users, groups, and user profiles that are stored in that user store. See the *Security Guide* for instructions on setting up external user stores and making them writable.

If the user store is read-only, the properties are visible in the portal but you cannot change them. The default configuration for supported external user stores is read-only access to users and groups from the Administration Console or the WebLogic Server Administration Console. The default RDBMS user store provides write access by default.

Accessing Multiple User Stores

If you have users stored in more than one external user store, you might want to access each of these user stores. WebLogic Portal supports using more than one authentication provider and more than one user store.

Accessing multiple user stores can help you perform the following portal administration tasks:

- Let users in external user stores log into your portal desktops.
- Provide personalized applications to users stored in an external user store. When those external users and groups appear in the Administration Console, you can define personalization rules based on the users and groups and their profile properties.

- Entitle users in an external user store to specific portal resources when they log in. You can define visitor entitlement rules based on external users and groups.
- Give users in an external user store administrative access to portal applications. You can define delegated administration roles based on external users and groups.

Tip: Do not store identical user names or group names in more than one user store. If you are storing users, passwords, and groups in an external user store (such as OpenLDAP or Netscape's iPlanet), you can connect that user store to WebLogic Server (assuming it is a supported type), and the users in that external user store can log into your portals. For example, user *dsmith* can reside in the default RDBMS user store and in an external RDBMS user store. In this case, WebLogic Portal uses only one user profile for user *dsmith*.

After you follow the instructions in the *Security Guide* to develop and configure the external user store, the authentication provider for the user store appears in a drop-down list in the Administration Console. Two authentication providers appear in Figure 2-1.

Figure 2-1 Choose the Authentication Provider from the Drop-Down List

Note: If your external user store contains additional properties for users and groups (for example, e-mail and phone numbers), accessing those properties involves separate development steps to create a UUP. See Chapter 6, "Configuring a UUP".

Planning to Use a UUP

Use UUP to retrieve additional user profile information from other user stores, such as OpenLDAP servers and databases, legacy applications, and flat files. Using a UUP to integrate existing systems with user profiles provides the following benefits:

- Integrates with most other systems (because of the UUP open interface), so you can access existing user information without migrating that data into the portal schema.
- Retrieves data from multiple systems and matches the data to one user profile type.
- Uses multiple profile types in the same instance of a portal server.

- Retrieves the same attribute from two different systems for two different users. For example, you can obtain a customer address property from SAP for Customer 1 and the same information from Oracle Financial for Customer 2.
- Triggers personalization, sets role-based administration, and creates entitlements to limit access to portal resources.
- You can create a UUP to retrieve additional user information stored in other external data stores.

Use a UUP to perform the following tasks:

- Use WebLogic Portal's JSP tags, controls, or APIs to retrieve property values from that external user store.
- View external properties in the Administration Console to define rules for personalization, delegated administration, or visitor entitlement. Users and groups are not considered properties.

You do not need to use an external data store's UUP to perform any of the following tasks:

- Provide authentication for users in the external user store.
- Define rules for personalization, delegated administration, or visitor entitlement based only on users or groups in an external user store, rather than on user properties.
- Define rules for personalization, delegated administration, or visitor entitlement based on the user profile properties you create in BEA Workshop for WebLogic Platform, which are kept in the portal schema.

Using a Sample UUP Scenario

Use the following scenario to understand how and when to use a UUP:

1. You created a new portal application and you want users to be able to log into it.
2. Your users are stored in an RDBMS user store outside of the WebLogic Portal. You could connect WebLogic Server (your portal application's domain server instance) to your RDBMS system, and your users could log into your portal application as if their user names and passwords were stored in WebLogic Server. If authentication was all you wanted to provide through your RDBMS user store, you could stop here without using a UUP.
3. You also stored e-mail and phone number information (properties) for users in your RDBMS user store, and you want to access those properties in your portal applications.You must create a UUP for your RDBMS user store to access those additional properties from your code.

For instructions on how to set up a UUP, see Chapter 6, "Configuring a UUP".

Upgrading Users and Groups from Portal 8.1

WebLogic Portal Server 9.2 contained a new default *SQLAuthenticator* authentication provider, which contains an RDBMS user store for users and group membership. The same default authentication provider is used in 10.0.

When you run the BEA WebLogic Upgrade Wizard, you can choose to upgrade your WebLogic Portal 8.1 SP4, SP5, or SP6, or Portal 9.2 or 9.2 MP1 users and groups or continue to use your existing RDBMS user store in Portal 8.1:

- Upgrade users and groups – Choose to automatically upgrade your users and groups from WebLogic Portal 8.1 or 9.2 to the new RDBMS user store, which is part of the default *SQLAuthenticator* authentication provider in the WebLogic Portal 9.2 and 10.0 installation. When you run the BEA WebLogic Upgrade Wizard and it detects your Portal 8.1 *RDBMSAuthenticator*, you can select the **Upgrade RDBMSAuthenticator** option. Selecting this option replaces the existing authentication provider with the new *SQLAuthenticator* authentication provider and upgrades its contents, including users and groups.
- Do not upgrade users and groups – Choose to continue to use the RDBMS user store from the default *RDBMSAuthenticator* in your WebLogic Portal 8.1 or 9.2 installation. When you run the BEA WebLogic Upgrade Wizard and it detects your Portal 8.1 *RDBMSAuthenticator*, you can select the **Do not upgrade RDBMSAuthenticator** option. You can choose to manually upgrade your users and groups to the Portal 9.2 or 10.0 RDBMS user store later. See the *Upgrade Guide* for instructions.

For step-by-step upgrade instructions on running the BEA WebLogic Upgrade Wizard, see the *Upgrade Guide*.

Designing Users and Groups for Optimal Performance

Following are guidelines to follow when you create users and groups:

- Plan and create your groups according to the roles you will create. If you do not consider roles in your group planning, you might have to redo your groups to accommodate roles.
- Adjust your user management cache settings.

Part II Development

Part II includes the following chapters:

- Chapter 3, “Adding Groups with JSP Tags and Controls”
- Chapter 4, “Adding and Updating Users with JSP Tags and Controls”
- Chapter 5, “Creating and Updating User Profiles”
- Chapter 6, “Configuring a UUP”
- Chapter 7, “Setting Up Anonymous User Tracking”

This section contains instructions for developers on how to create JSP tags and controls that provide the capability to add users or let users add themselves to your portal. Developers can also create groups and subgroups, add user profiles and properties, track anonymous users, and set up a UUP to integrate with other external user stores.

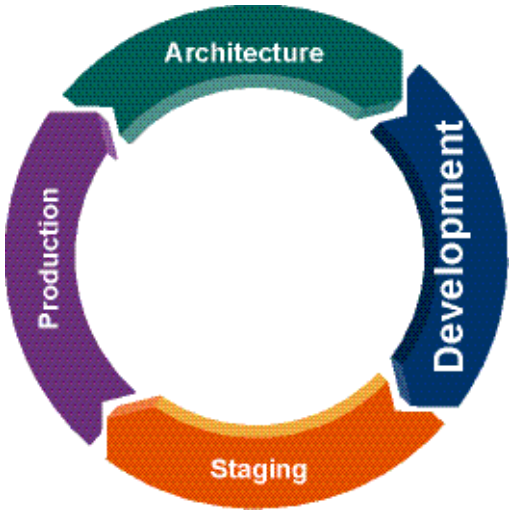
The decisions you made during the Architecture phase shape what you do in the Development phase. If you decided to use group profiles, then users that you add to a group in the Development phase can inherit the group property values. The Development phase allows you to create user profiles to gather user information and then edit that information later (if the user store has write access).

When you finish the Development phase you can proceed to the Staging phase. Consider setting up a common development environment for the Development phase and the Staging phase. You can use the Staging phase as a test environment to add users and groups to test the functionality you created in the Development phase. You might move iteratively between these two phases,

developing and then testing what you created. The users and groups you add in the staging environment might be different than the users and groups you use in your production system.

If you have moved on to the Production phase and then go back to make changes that affect the Development phase, you must redeploy your portal application in order to view your changes.

For a detailed description of the Development phase of the portal life cycle, see the *WebLogic Portal Overview*. The portal life cycle is shown in the following graphic:



CHAPTER 3

Adding Groups with JSP Tags and Controls

Developers can use Workshop for WebLogic to provide the capability to create groups, add group profiles, and delete groups with JSP tags and controls. Portal administrators might prefer to add or edit a small number of groups in the WebLogic Portal Administration Console.

If you are using an external user store, determine if there are existing groups in that user store. You can use the WebLogic Scripting Tool to retrieve existing groups, as well as users. See the *BEA WebLogic Server WebLogic Scripting Tool Guide* for more information.

You should plan your role structure before you create groups. Making groups role-based to control what each group can do and see in your portal will save you time later.

If you are using an external user store and it does not contain groups, developers can use the following tools to create and manage groups:

- **JSP Tags** – Use the `createGroup` JSP tag to create a new group. Other JSP tags let you add a subgroup, remove a group or subgroup, and create a group profile.
- **Controls** – Use the `createGroup` action in the group provider control to create a new group. Other controls allow you to add a subgroup, remove a group or subgroup, and create a group profile. See the *Javadoc* for more information.
- **Java** – Work directly with the `com.bea.p13n.security.management.authentication.AtnManagerProxy` Java class to add groups, create group profiles, and remove groups. When you use the API to create a user, a user profile is not automatically created. See the *Javadoc* for more information on the Java class.

Portal administrators can use the following tools to create and manage groups:

- **WebLogic Portal Administration Console** – Add and edit groups here if you work primarily in the Administration Console. See Chapter 8, "Adding and Managing Groups" for instructions.
- **WebLogic Server Administration Console** – Add and edit groups here if you work primarily in the WebLogic Server Administration Console. See the *BEA WebLogic Server Securing WebLogic Resources Guide* for instructions.

Tip: Roles and entitlements are stored in the LDAP server by default. The BEA Propagation Utility does not propagate this data from your staging environment to production. See the *Production Operations Guide* for more information.

This chapter includes the following sections:

Creating Groups

If the user store you are accessing already has users organized into groups, you can build a hierarchical tree in the Administration Console to match that group structure. See "Using Existing Groups" on page 8-2 for instructions. If you are using WebLogic Server's default RDBMS user store, you do not need to build a group hierarchy tree.

When you create a group, you are creating an empty group to which you can add any number of users. A subgroup exists under a parent group.

Tip: Create groups before you add users. If the group exists before you add a user, you can then immediately add the user to the group.

This section contains the following topics that explain how to add a group programmatically with JSP tags and controls:

- Creating a Subgroup with a Control

Creating a Group with a JSP Tag

The <ugm:createGroup> tag adds a new group in the Security realm, and a corresponding group profile in the personalization database. The logged-in user must have administrator rights to execute this tag. You can use the JSP you create alone, in a page flow, or in a web project.

You can determine which portal administrator can manage each user group by assigning delegated administration roles to those groups.

The <ugm:createGroup> tag automatically creates a group profile. You can use this tag with the <ugm:addUserToGroup> tag to add a user to your new group and with the <ugm:addGroupToGroup> tag to add a subgroup to the group you just created. Verify your results with the <ugm:getTopLevelGroups> tag.

See the *JSP Tag Javadoc* for more information on the Java class.

Tip: If you test your JSP tag or control in the Staging phase and the new group does not appear, verify that you built a group hierarchy tree for the user store. If you built the tree but you still do not see a list of groups, the user store probably does not allow read access. See the *Security Guide* for instructions on giving the provider read access.

Creating a Subgroup with a JSP Tag

The <um:addGroupToGroup> JSP tag adds a child group to a parent group. A group can have more than one parent. Both the parent group and the child group must already exist.

After you create the subgroup, you can use the <profile:createProfile> tag to create a group profile for the new group. You might want to use the <ugm:addUserToGroup> tag to add users to the new subgroup. You can verify your results with the <ugm:getChildGroupNames> and <ugm:getUsernamesForGroup> tags.

The steps to create the <ugm:addGroupToGroup> tag are similar to the <um:createGroup> tag.

See the *JSP Tag Javadoc* for more information on the Java class.

Creating a Group with a Control

The group provider control contains an action called createGroup, which adds a new parent group in the Security realm, and a corresponding group profile in the personalization database. The logged-in user must have administrator rights to execute this control.

The `createGroup` action provides a pre-built form (a form bean) you can add to your JSPs to add a new parent group. Using a form can save you development time because it simplifies data entry and group management.

The `createGroup` action fails if the group you are adding already exists in WebLogic Portal or if you give the group an invalid name. Use the `createGroup` action instead of the JSP tag if you want to add a parent group in a page flow where a successful action forwards the user to another JSP.

After you use the `createGroup` action, you can use the `createGroupProfile` action in the Profile control to create a group profile for the new group. You can also use the `addGrouptoGroup` action to create a subgroup.

For more information on using the Group Provider control and its properties, see the *Javadoc*.

Creating a Subgroup with a Control

The `addGroupToGroup` action in the Group Provider control adds a child group to a parent group. You must log in as a user with administrator rights to add a group to a group.

The `addGroupToGroup` action provides a pre-built form (form bean) to add to your JSPs to add an existing group to another group. Using a form can save you development time because it simplifies data entry and group management.

Use the `addGroupToGroup` action instead of the JSP tag if you want to add a subgroup in a page flow where a successful action forwards the user to another JSP.

After you use the `addGroupToGroup` action, you can use the `createGroupProfile` action in the Profile control to create a group profile for the new group.

For more information on using the Group Provider control and its properties, see the *Javadoc*.

Removing Groups

When you delete a group, you permanently remove the group and all of its subgroups from the user store. You must re-create all of the affected groups if you want to use them again. Deleting a group does not remove the users contained within it.

You can remove a subgroup from a group, and you can move a group within the group hierarchy to change its relationship to other groups. If the group was listed in a delegated administration or visitor entitlement role, you must also remove that group from the role definition.

In addition to deleting a group with a JSP tag or control in Workshop for WebLogic, you can remove a group in the Administration Console. See Chapter 8, "Adding and Managing Groups" for instructions. If you are using an external user store, you can also remove the group there.

This section contains the following topics that explain how to remove groups programmatically with JSP tags and controls:

- Removing a Group with a JSP Tag
- Removing a Subgroup with JSP Tag
- Removing a Group with a Control
- Removing a Subgroup with a Control

Removing a Group with a JSP Tag

The `<ugm:removeGroup>` JSP tag deletes a specific group. The logged-in user must have administrator rights to execute this tag.

The `<ugm:removeGroup>` tag is often used with the `<ugm:removeGroupFromGroup>` tag and the `<ugm:removeUserFromGroup>` tag. After you remove the group, you can verify the results with the `<ugm:getTopLevelGroups>` tag to view a list of all top groups.

See the *JSP Tag Javadoc* for more information on the Java class.

Removing a Subgroup with JSP Tag

The `<ugm:removeGroupFromGroup>` JSP tag removes a child group from a parent group. This tag does not delete the group.The logged-in user must have administrator rights to execute this tag.

The `<ugm:removeGroupFromGroup>` tag is often used with the `<ugm:removeGroup>` tag and the `<ugm:removeUserFromGroup>` `tag`. After you remove the subgroup, you can verify the results with the `<ugm:getChildGroupNames>` tag.

See the *JSP Tag Javadoc* for more information on the Java class.

Removing a Group with a Control

The `removeGroup` action in the Group Provider control removes a specific group. The logged-in user must have administrator rights to execute this tag.

Use the `removeGroup` action instead of the JSP tag if you want to remove a parent group in a page flow where a successful action forwards the user to another JSP.

You can also use another action, `removeGroupFromGroup`, to remove a group's subgroups. After you use the `removeGroup` action, you can verify the results with the `getTopLevelGroupNames` action.

For more information on using the Group Provider control and its properties, see the *Javadoc*.

Removing a Subgroup with a Control

The `removeGroupFromGroup` action in the Group Provider control removes a child group from a group. This tag does not delete the group. The logged-in user must have administrator rights to execute this tag.

Use the `removeGroupFromGroup` action instead of the JSP tag if you want to remove a child group in a page flow where a successful action forwards the user to another JSP.

After you use the `removeGroupFromGroup` action, you can verify the results with the `getAllGroupNames` action. You can also use the `removeGroup` action to remove a parent group.

For more information on using the Group Provider control and its properties, see the *Javadoc*.

Creating a Group Profile

After you create groups, an optional step is to create group profiles. You can use BEA Workshop for WebLogic Platform to create a group profile and its default property values. You can edit the profile's default values in Workshop for WebLogic or in the WebLogic Portal Administration Console.

A group profile contains information about the group that is stored in editable fields in the Administration Console. Group profile properties could include whatever group information you want, such as the group's location, a group e-mail alias, or the name of the group's administrator.

You can edit properties for groups, but users belonging to those groups do not automatically inherit the group properties you specify. If a user belongs to two groups, for example, and you have edited each group's properties, you must specify which set of group properties the user should inherit by configuring a `ProfileWrapper` successor at runtime. A `ProfileWrapper` is a lightweight object that can access the correct `ProfileManager` session beans based on the profile identity with which it is initialized.

For instructions on editing group profile values in the Administration Console, see Chapter 8, "Adding and Managing Groups".

This section contains the following topics that explain how to create a group profile programmatically with JSP tags and controls:

- Creating a Group Profile with a JSP Tag
- Creating a Group Profile with a Control

Creating a Group Profile with a JSP Tag

You can use the `<profile:createProfile>` tag to create a new group profile that corresponds to the `profilekey`. Creating a profile also creates designated successors for property inheritance. The logged-in user must have administrator rights to execute this tag.

You can also create a profile for a group in a user store that does not allow read access. When the user logs in, the user is paired with the profile. You cannot edit the group profile properties, but you can still use those properties to set up personalization, delegated administration, and visitor entitlement.

The `<profile:createProfile>` tag is often used with the `<ugm:createGroup>` tag and the `<profile:getProfile>` tag.

See the *JSP Tag Javadoc* for more information on the Java class.

Creating a Group Profile with a Control

The `createGroupProfile` action in the Profile control creates a new group profile that corresponds to the `profilekey`. Creating a profile also creates designated successors for property inheritance. The logged-in user must have administrator rights to execute this tag.

Developers use this action in the Profile control to retrieve a group's profile, use the Property control to put properties in working memory, and then use the Rules Executor control to evaluate and filter the group’s profile properties in order to trigger actions based on that evaluation.

You can use the `createGroupProfile` action with the `createGroup` action to create a group profile for a new group. After you create the group profile, use the `getGroupProfile` action to access the ProfileWrapper for the group.

Use the `createGroupProfile` action instead of the `<profile:createProfile>` JSP tag if you want to create a group profile in a page flow where a successful action forwards the user to another JSP.

For more information on using the Profile control and its properties, see the *Javadoc*.

CHAPTER 4

Adding and Updating Users with JSP Tags and Controls

Developers can use JSP tags and controls in Workshop for WebLogic to provide the capability to add users to your portal application or allow users to add themselves. Portal administrators might prefer to add or edit a small number of users in the Administration Console.

Developers can also use JSP tags and controls to create and edit user profiles to store additional information about users. Portal administrators might prefer to perform these tasks in the Administration Console.

If you have a large number of users stored in an external user store, you can use the WebLogic Scripting Tool to retrieve those users. You could use JSP tags and controls in BEA Workshop for WebLogic Platform to add the ability to your portal for users to add themselves. If you want to add a portal administrator with special privileges to manage portal content and users, use the Administration Console. See Chapter 9, "Adding and Managing Users" for instructions.

Developers can use the following tools to add and manage large numbers of users:

- **JSP tags** – Use the `createUser` JSP tag to create a new JSP tag to add a new user. You can also use the tag to create the ability for visitors to your portal to register themselves and be added to the user store. Other JSP tags let you place users in groups, change passwords, and delete users.
- **Controls** – Use the `createUser` action in the User Provider control to add a new user. You can also use the User Provider control to create the ability for visitors to your portal to register themselves and be added to the user store. Other controls let you place users in groups, change passwords, and delete users. See the *Javadoc* for more information.

- **Java** – Work directly with the `com.bea.p13n.security.management.authentication.AtnManagerProxy` Java class to add users, change passwords, and delete users. When you use the API to create a user, a user profile is not automatically created. See the *Javadoc* for more information.

Portal administrators can use the following tools to create and manage a small number of users:

- **WebLogic Portal Administration Console** – Add and edit users here if you work primarily in the Administration Console. See Chapter 9, "Adding and Managing Users" for instructions.
- **WebLogic Server Administration Console** – Add and edit users here if you work primarily in the WebLogic Server Administration Console. See the *BEA WebLogic Server Securing WebLogic Resources Guide* for instructions.
- **External user store** – Add a user to an external user store (such as openLDAP) and then use an external tool to configure that repository to be writable.You can access more than one user store to select users and groups. See "Accessing Users in an External User Store" on page 4-4 for more information.

Adding a user with any of these methods (except the Java API) adds the user to the user store and creates a basic user profile that contains the user's identity (username and password). The Java API does not automatically create a user profile when you add a user.

You can set up other user properties (such as address, phone number, e-mail, and so on) to enable personalization and define rules for delegated administration and visitor entitlement.

Tip: Users and groups are stored in the RDBMS server by default. The BEA Propagation Utility does not propagate this data from your staging environment to production.

This chapter includes the following sections:

Creating Users

Developers can build functionality into your portal application that lets users add themselves to the domain. This self-registration is often used by visitors to your portal. If you have a large number of users stored in a user store, you can use Workshop for WebLogic to access those users. If you want to add a few users or a portal administrator, use the Administration Console.

If a tracked anonymous user registers at your portal, the tracking data is transferred to the new user. The new user is now considered a registered user and is no longer an anonymous user. See "Setting Up Anonymous User Tracking" on page 7-1 for instructions on setting up this user tracking.

When you create a new user, WebLogic Portal creates a user profile containing the user's identity (name and password) in the user store. Other user properties (such as address, phone number, e-mail, and so on) can be used to set up personalization and define rules for delegated administration and visitor entitlement.

WebLogic Portal does not support multiple RDBMS authenticators under a single Security realm.

This section contains the following topics that explain how to add a user programmatically with JSP tags and controls:

- Creating a User with a JSP Tag
- Creating a User with a Control

Creating a User with a JSP Tag

The `<ugm:createUser>` tag adds the ability to create a user and sets the user's password. You can add this JSP tag to a Java Page Flow (JPF) and show that functionality in a portlet. If the user already exists, the `<ugm:createUser>` tag does not create another user.

You can determine the roles that can perform this action in the Administration Console. See the *Security Guide* for instructions on defining roles.

Note: You must have WebLogic Server running in the cluster for self-registration to work.

If you set the `saveAnonymous` attribute for this tag to **true** and an `AnonymousProfile` exists, any properties defined in the `AnonymousProfile` are set for the new user.

After you create the user, you can use the `<profile:createProfile>` tag to create a user profile for the new user. You can verify your results with the `<ugm:getUsernames>` tag.

See the *JSP Tag Javadoc* for more information on the Java class.

Creating a User with a Control

The User Provider control contains an action called `createUser`, which adds the ability to create a user and sets the user's password. You can add this control to a Java Page Flow (JPF) and show that functionality in a portlet.

The `createUser` action provides a pre-built form (a form bean) you can add to your page flow to add a new group. Using a form can save you development time because it simplifies data entry and group management.

If you use the `createUser` action, you can also use the Profile control to create a user profile for the new user.

Note: You must have WebLogic Server running in the cluster for self-registration to work.

Use the `createUser` action instead of the JSP tag if you want to add a user in a page flow where a successful action forwards the user to another JSP. You can also use the `createProfile` action to create a user profile for the new user.

For more information on using the User Provider control and its properties, see the *Javadoc*.

Accessing Users in an External User Store

If you decide to use multiple user stores (not the default RDBMS user store built into WebLogic Server), most of the effort is setting up and configuring those providers and then connecting WebLogic Server to those providers. See the *Security Guide* for instructions on setting up user stores. You can access multiple user stores.

Two ways exist to add more users to an external user store (such as openLDAP):

- Adding a user directly to an external provider
- Adding a user to an external provider with an outside tool, such as the Administration Console

See "Accessing Users in an External User Store" on page 9-4 for instructions on adding users to an external user store.

Placing Users in Groups

You can add a user to one or more groups using a JSP tag, a control, or the Administration Console. See Chapter 9, "Adding and Managing Users" for instructions on using the Administration Console to place users into groups.

Tip: The WebLogic Portal's internal RDBMS user store can store large numbers of users and groups. The internal LDAP is sufficient for storing policies for a small number of roles, as well as entitlements.

This section contains the following topics that explain how to locate users programmatically with JSP tags and controls:

- Adding a User to a Group with a JSP Tag
- Adding a User to a Group with a Control
- Viewing a User's Group Membership with a JSP Tag
- Viewing a User's Group Membership with a Control
- Removing a User from a Group with a JSP Tag
- Removing a User from a Group with a Control

Adding a User to a Group with a JSP Tag

The `<ugm:addUserToGroup>` tag adds a user to a group. The tag adds the username you provide to the specified group name. Both the user and the group must previously exist for proper tag behavior. The logged-in user must have administrator rights to execute this tag.

After you add the user to a group, you can verify your results with the `<ugm:getUsernamesForGroup>` tag.

Note: If a user store does not allow write access to users and groups, you will not be able to add users to groups with the Administration Console. You must add users to groups in the user store directly.

See the *JSP Tag Javadoc* for more information on the Java class.

Adding a User to a Group with a Control

The `addUserToGroup` action in the Group Provider control adds an existing user to an existing group. The `addUserToGroup` action provides a pre-built form (a form bean) you can add to your page flow to add a user to a group. The logged-in user must have administrator rights to execute this tag.

Use the `addUserToGroup` action instead of the JSP tag if you want to add a user to a group in a page flow where a successful action forwards the user to another JSP.

The `addUserToGroup` action is often used with the `isMemberofGroup` action, which determines if a user belongs to a group. You can also verify that the user was added to the group with the `getUsernamesForGroupLimited` action.

For more information on using the Group Provider control and its properties, see the *Javadoc*.

Viewing a User's Group Membership with a JSP Tag

A user can belong to more than one group. The `<ugm:getGroupNamesForUser>` JSP tag retrieves a String array that contains the group names to which the user belongs. The logged-in user must have administrator rights to execute this tag.

Developers use the `<ugm:getGroupNamesForUser>` tag to verify the results of the `<ugm:addUserToGroup>` tag.

If you are using an RDBMS user store, be aware of case sensitivity when looking up users and groups. For example, *Bob* is different than *bob*.

See the *JSP Tag Javadoc* for more information on the Java class.

Viewing a User's Group Membership with a Control

A user can belong to more than one group. The `getGroupNamesForUser` action in the Group Provider control retrieves a string array that contains the group names to which the user belongs. The logged-in user must have administrator rights to execute this tag.

Developers use the `getGroupNamesForUser` action instead of the JSP tag to view a user's group membership in a page flow where a successful action forwards the user to another JSP.

You can use the `getGroupNamesForUser` action to verify the results of the following actions: `createUser`, `createGroup`, and `addUserToGroup`.

For more information on using the Group Provider control and its properties, see the *Javadoc*.

Removing a User from a Group with a JSP Tag

The `<ugm:removeUserFromGroup>` tag removes a user from a group. The tag does not delete the user. The logged-in user must have administrator rights to execute this tag.

Developers often use the `<ugm:removeUserFromGroup>` tag with the following tags: `<ugm:removeGroup>`, `<ugm:removeUser>`, and `<ugm:removeGroupFromGroup>`.

See the *JSP Tag Javadoc* for more information on the Java class.

Removing a User from a Group with a Control

The `removeUserFromGroup` action in the Group Provider control removes a user from a group. The tag does not delete the user. The logged-in user must have administrator rights to execute this tag.

Developers use the `removeUserFromGroup` action instead of the JSP tag to remove a user from a group in a page flow where a successful action forwards the user to another JSP.

The `removeUserFromGroup` action is often used with the following actions: `removeGroup` and `removeGroupFromGroup`. You can also verify that the user was removed from the group with the `getUsernamesForGroupLimited` action, which retrieves a list of users in a group.

For more information on using the Group Provider control and its properties, see the *Javadoc*.

Searching for Users

WebLogic Portal support two ways to locate users by username:

- **WebLogic Portal Administration Console** – If you need to perform administrative tasks, such as editing user profiles, removing users from a group, or deleting users from the system, you must first locate those users in the system. See Chapter 9, "Adding and Managing Users" for instructions on searching the Administration Console for a single user or multiple users.
- **BEA Workshop for WebLogic Platform** –Programmatically locate users with JSP tags and controls.

Tip: If you are using an RDBMS user store, be aware of case sensitivity when looking up users and groups. For example, *Bob* is different than *bob*.

This section contains the following topics that explain how to locate users programmatically with JSP tags and controls:

Finding a Single User with a JSP Tag

The `<ugm:userExists>` JSP tag determines if a user exists in WebLogic Portal by searching for a specific username.

Developers often use the `<ugm:userExists>` tag with the `<ugm:createUser>` tag to verify that a user does exist in the system.

See the *JSP Tag Javadoc* for more information on the Java class.

Finding Multiple Users with a JSP Tag

The `<ugm:getUserNames>` tag lets you use a wildcard search expression to retrieve a list of all user names in the WebLogic Portal. The search expression supports only the asterisk (*) wildcard character, and is case insensitive. You can use as many asterisks as you want in the search expression. You can limit the number of items that the tag returns; only usernames that match the search expression are returned.

Developers often use the `<ugm:getUserNames>` tag with the `<ugm:getUserNamesForGroup>` tag to retrieve a list of all users that belong to a specific group.

See the *JSP Tag Javadoc* for more information on the Java class.

Finding a Single User with a Control

The `userExists` action in the User Provider control determines if a user exists in WebLogic Portal by searching for a specific username.

Developers use the `userExists` action instead of the JSP tag to locate a user in a page flow where a successful action forwards the user to another JSP. The `userExists` action can be used with the `createUser` action to verify that a new user was successfully added.

For more information on using the User Provider control and its properties, see the *Javadoc*.

Finding Multiple Users with a Control

The `getUserNames` action in the User Provider control lets you use a wildcard search expression to retrieve a list of all user names in the WebLogic Portal. The search expression supports only the asterisk (*) wildcard character, and is case insensitive. You can use as many asterisks as you want in the search expression. You can limit the number of items that the tag returns; only usernames that match the search expression are returned.

Developers use the `getUserNames` action instead of the JSP tag to find users in a page flow where a successful action forwards the user to another JSP. You can use the `getUserNames` action with the `getUserNamesForGroup` action to retrieve a list of all users that belong to a specific group.

For more information on using the User Provider control and its properties, see the *Javadoc*.

Changing a User's Password

You might need to reset a password if a user loses or cannot remember a password. If you have the appropriate delegated administration rights, you can change any user's password. See the *Security Guide* for instructions on setting up delegated administration.

Ensure that your user knows the new password, because once the password is changed there is no way to find out what it is. If a user forgets a password, the portal administrator must change it again.

This section contains the following topics that explain how to update a user's password programmatically:

- Changing a Password with a JSP Tag
- Changing a Password with a Control

Changing a Password with a JSP Tag

The `<ugm:setPassword>` JSP tag updates the password for the user you specify in the Security realm. Due to Security constraints, only administrators should use this method. Users must be logged in to change their passwords.

The new password must be at least eight characters. Developers often use the `<ugm:setPassword>` tag with the `<ugm:getUserNames>` tag.

See the *JSP Tag Javadoc* for more information on the Java class.

Changing a Password with a Control

The `setPassword` action in the User Provider controls updates the password for the user you specify in the Security realm. Due to Security constraints, only administrators should use this method. Users must be logged in to change their passwords.

The new password must be at least eight characters. Developers often use the `setPassword` action instead of the JSP tag to update a user's password in a page flow where a successful action

forwards the user to another JSP. You can use the `setPassword` action with the `getUserNames` action to update a user's password.

For more information on using the User Provider control and its properties, see the *Javadoc*.

Removing a User

Removing a user from a group does not delete the user from the system or change a user profile's properties. A group does not own a user, so you can add and remove users from groups without affecting the user's properties. Removing a user from a group deletes the user from any delegated administration or visitor entitlement roles based on that group. For example, if you remove a user from the *Administrators* group, that user may no longer have full administrative access to the Administration Console.

When you delete a user, you remove the user from the user store. The deleted user is no longer available in any other group or subgroup, and the user will not be able to log into your portal application. To get the user back in the system, you must create the user again.

If you want to remove the user from a group but not remove the user from the entire system, see "Searching for Users" on page 4-7 or "Removing Users" on page 9-14.

Note: If you are using an external user store to store users and groups (one that is not the default RDBMS provider built into WebLogic Server), and you want to remove a user from that provider, the provider might be configured to prevent user removal from an outside tool, such as the Administration Console. See "Adding a User to an External User Store with an Outside Tool" on page 9-5 the instructions.

This section contains the following topics:

Removing a User with a JSP Tag

The `<ugm:removeUser>` JSP tag removes a user from the system. This tag does not remove both the Security realm and user profile records for this user. (You can use the <profile:removeProperty> JSP tag or the removeProperty action in the Property control in your page flows to remove user profiles. See Deleting a Property Set and Properties for more information.) The tag can remove any type of extended user that has its `profileType` set in the database. Due to security constraints, only administrators should use this method.

Developers often use the `<ugm:removeUser>` tag with the `<ugm:removeUserFromGroup>` tag to remove the user from its group.

See the *JSP Tag Javadoc* for more information on the Java class.

Removing a User with a Control

The `removeUser` action in the User Provider control removes a user from the system. This control does not remove both the Security realm and user profile records for this user. (You can use the <profile:removeProperty> JSP tag or the removeProperty action in the Property control in your page flows to remove user profiles. See Deleting a Property Set and Properties for more information.) The control can remove any type of extended user that has its `profileType` set in the database. Due to security constraints, only administrators should use this method.

Use the `removeUser` action instead of the JSP tag if you want to delete a user in a page flow where a successful action forwards the user to another JSP. The `removeUser` action can be used with the `removeUserFromGroup` action to remove the user from its group.

For more information on using the User Provider control and its properties, see the *Javadoc*.

CHAPTER 5

Creating and Updating User Profiles

Developers can use Workshop for WebLogic to create JSP tags and controls that add and edit user profiles. A user profile consists of a username and any additional properties you collect and store about a user. These properties can be used to personalize the user's experience in your portal.

Properties can consist of personal data, work-related data, geographic data, or something else that logically categorizes your users. For example, you could create a property set in BEA Workshop for WebLogic Platform called **human resources** that contains properties such as **gender**, **hire date**, and **e-mail address**.

You must use Workshop for WebLogic to programmatically create user profiles and edit the profile's default property values. Administrators can edit the profile's property values in the Administration Console. See Chapter 10, "Editing User Profile Property Values" for instructions.

When users log into a portal, the portal can access the property values and target them with personalized content, e-mails, pre-populated forms, and discounts based on the personalization rules you set up. See the *Interaction Management Guide* for more information.

Developers can use the following Workshop for WebLogic to programmatically create and edit user profile default property values:

- **JSP Tags** – Use the `createProfile` JSP tag to create a portal JSP tag that adds a user profile. Other JSP tags let you retrieve the user profile and add or edit its properties.
- **Controls** – Use the `createProfile` action in the Profile control to add a user profile. Other actions in the User Provider control let you retrieve the user profile and add or edit its properties. You can use the Profile control to retrieve a user's profile, use the Property control to put properties in working memory, then use the Rules Executor control to

evaluate and filter the user profile's properties in order to trigger actions based on that evaluation.

- **Java** – Work directly with the `com.bea.p13n.usermgmt.profile.ProfileFactory` Java class to create a user profile. You can change user properties by calling the `ProfileWrapper` object directly. For more information on the Java class, see the *Javadoc*.

This chapter includes the following sections:

Creating a User Profile

A user profile is a collection of user property values for a user from all available user property sets. Each piece of metadata in a user profile is called a user property. A user profile property set organizes the properties that it contains and provides a convenient way to name a group of properties for a specific purpose. The properties you create can be used to define rules for personalization, delegated administration, or visitor entitlement.

User properties can range from statically-defined properties (such as a user's phone number and e-mail address) to dynamically-created and persisted properties (web site tracking information for the user, for example). A property set called *personal* could contain properties, such as age, gender, marital status, and address. Another property set called *preferences* could contain properties, such as *hobby*, *favorite color*, and *news preference*.

You must create user profiles and edit the profile's default values in Workshop for WebLogic. You can edit the profile's values in Workshop for WebLogic or in the Administration Console.

WebLogic Portal provides a default user profile property set called `CustomerProperties.usr` that contains common properties.

Note: You can also create an application-defined property set to store profile data for entities that are not users or groups. These entities include communities and Web Services for Remote Portlets (WSRP), or a custom entity created by an application programmer. See the *Interaction Management Guide* for instructions on creating this type of property set.

This section contains the following topics:

Creating a User Profile Property Set

You can create a user profile and a property set in Workshop for WebLogic. You can edit the profile's default values in Workshop for WebLogic or in the Administration Console.

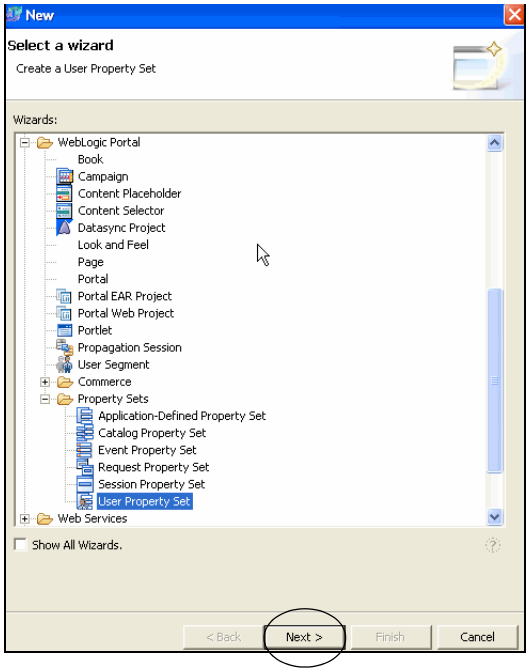
To create a property set for a user profile:

1. In the **BEA Workshop for WebLogic Platform Application** window, right-click the data\src\userprofiles folder, and choose **New > Other**.

   Tip: You can customize the menu so that **Property Sets** appears as a choice on the **New** menu. See the *Portal Development Guide* for instructions.

2. In the New dialog box, expand **WebLogic Portal** and then expand **Property Sets**.
3. Select **User Property Set** and click **Next**, as shown in Figure 5-1.

Figure 5-1 Create a User Profile Property Set



4. In the **New User Property Set** window, enter a name for the user profile property set in the **File name** field. Keep the `.usr` file extension. For example, **SalesRegion.usr**.
5. Click **Finish**. The user profile editor appears.

Adding Properties to a Property Set

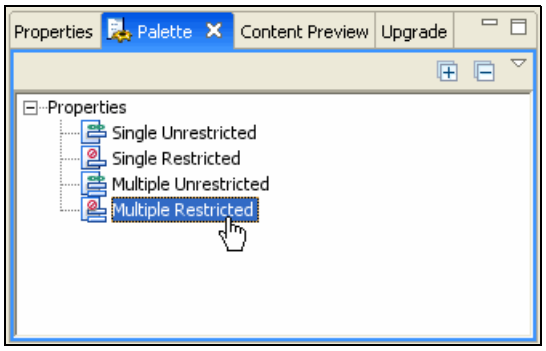
After you create a property set by following the instructions in "Creating a User Profile Property Set" on page 5-3, you can capture user information by adding properties to the profile.

To add properties to a user profile property set:

1. Select the user profile in the Navigator window.
2. In the Palette window, drag one of the following property types into the editor window:
   - **Single Unrestricted** – A single unrestricted property can have only one value, but you can enter any value. For example, **Country**, **Last Name**, or **Age**.
   - **Single Restricted** – A single restricted property can have only one value, and you are restricted to selecting that value from a predefined list. For example, a **Browser** property could have possible values of **Internet Explorer**, **Netscape**, **Opera**, or **Mozilla**.
   - **Multiple Unrestricted** – A multiple unrestricted property can have multiple values, and you can enter any values. For example, an **email** property could contain one or more e-mail addresses.
   - **Multiple Restricted** – A multiple restricted property can have multiple values, and you are restricted to selecting the values from a predefined list. For example, a **Forms** property could allow a user to select a document, such as **1040EZ**, **1040A**, or **1040**.

     Figure 5-2 shows the property types for a property set.

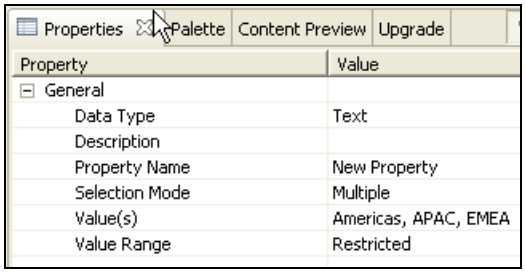
Figure 5-2 Drag a Property Type, Such as Multiple Restricted, to the User Profile Editor



3. Select the **Properties** tab and select the **Data Type** for the property value. Select one of the following values from the drop-down list: **Text**, **Numeric**, **Float** (decimal), **Boolean** (true or false), or **Date/Time.** A **Date/Time** property must be `java.sql.Timestamp` type. Your selection determines the dialog box you see when you edit the **Value** field. For example, properties with a Boolean data type are automatically set to single restricted. If you edit the **Data Type**, the change removes anything previously entered in the **Value** field, because the types of values change.

   Figure 5-3 shows how to configure a multiple restricted type to reflect three sales regions called **Americas**, **APAC**, and **EMEA**. You could use this **Sales Region** property to target sales employees with personalized content.

Figure 5-3 Enter User Profile Details in the Property Editor



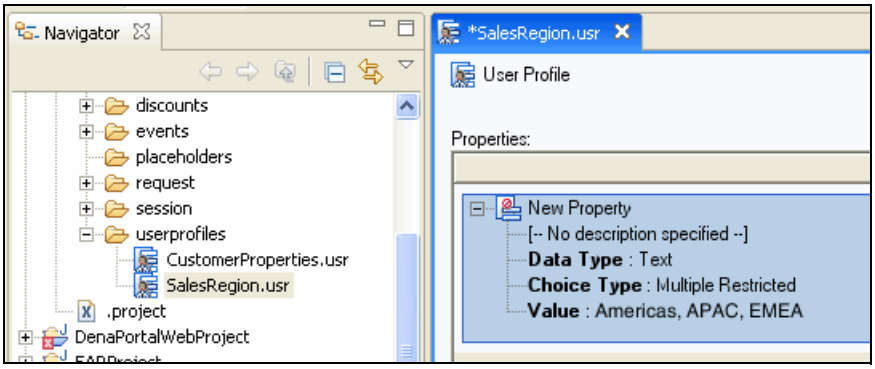
4. In the **Selection Mode** and **Value Range** fields in the property editor in Workshop for WebLogic, you can change the type of property. This field will already be populated, based on the type of property you dragged from the **Palette** window, but you can change a property from single unrestricted to multiple restricted.

   Note: Any change to **Data Type**, **Selection Mode**, or **Value Range** fields replaces anything previously entered in the **Value** field because the number of allowed values changes.

Use the **Value** field to enter values for restricted types or to set the default value for unrestricted types. Click the ellipsis icon (**...**) to enter values. (If you picked **Restricted** in the **Value Range** field, enter the value in the **Enter Property Value** dialog box that appears and click **Add** after each entry. Click **OK** after you enter all values. If you picked **Unrestricted** in the **Value Range** field, enter the value in the **Enter Property Value** dialog box and click **OK**.) Any values you enter in this field will be removed if you change the **Data Type**, **Selection Mode**, or **Value Range**.)

The properties you enter in the property editor appear in the user profile editor, as shown in Figure 5-4.

Figure 5-4 View the Properties in the User Profile Editor



5. After you add all the properties, save the file by choosing **File > Save**.
6. Use the Administration Console to view the user profile you set up in Workshop for WebLogic.

WARNING: You can also use the Property control to programmatically create and manage properties. However, properties created with this control do not appear in the Administration Console. You must modify and update them programmatically.

Editing Properties and Values

You can use Workshop for WebLogic to create user profiles and the profile's default values. You can edit the profile's values in Workshop for WebLogic or in the Administration Console. See Chapter 10, "Editing User Profile Property Values" for instructions on editing the values for property sets in the Administration Console.

This section contains the following topics:

Editing Properties and Values in the Property Editor

Developers use the user profile editor in Workshop for WebLogic to create a user profile and add the profile's properties. Then you can edit properties and their default values that are part of each user's profile.

To modify properties and their values in Workshop for WebLogic:

1. Double-click the property set file in the Navigator window.
2. Select the property in the user profile editor that you want to modify.
3. Change the property or its value in the property editor window.

Tip: You can edit property values in the Administration Console.

Editing Properties and Values with JSP Tags

The `<profile:getProperty>` JSP tag retrieves property values for a specified property set. The `<profile:setProperty>` JSP tag updates a property value for either the session's current profile or for the anonymous user profile.

Typically, the `<profile:getProperty>` tag is used after the `<profile:getProfile>` tag is invoked to retrieve a profile for session use. The `<profile:getprofile>` JSP tag retrieves a user profile and its properties. The `<profile:getProperty>` and `<profile:setProperty>` JSP tags let developers retrieve and rapidly edit properties for a large number of users. If the `<profile:getProfile>` tag was not used after you used the `<profile:getProperty>` tag, the specified property value is retrieved from the anonymous user profile.

See the *JSP Tag Javadoc* for more information on the Java class.

You can retrieve an authenticated user profile by using the `<profile:getProfile>` JSP tag in a page flow as shown in the code sample in Listing 5-1.

Listing 5-1 Retrieve an Authenticated User Profile with the <profile:getProfile> Tag

```
<%@ page import="com.bea.p13n.usermgmt.SessionHelper"%>
<%@ taglib uri="http://www.bea.com/servers/p13n/tags/usermanagement"
   prefix="profile"%>
<%@ taglib uri="netui-tags-databinding.tld" prefix="netui-data"%>
<%@ taglib uri="netui-tags-html.tld" prefix="netui"%>
Profile is: [<code><%= SessionHelper.getProfile(request) %></code>]<br>
<%-- This tag works for authenticated users. --%>
<profile:getProfile profileKey="<%=request.getUserPrincipal().getName()%>"
   profileId="profile"/>
Profile is: [<code><%= profile %></code>]<br>

<%-- You would generally want to do this in your PageFlow, not your JSP. --%>
<netui-data:declareControl controlId="profileControl"
   type="com.bea.p13n.controls.profile.UserProfileControl"/>
<netui-data:callControl resultId="getProfileFromRequestResult"
   controlId="profileControl" method="getProfileFromRequest">
<netui-data:methodParameter
   value="{request}"></netui-data:methodParameter>
</netui-data:callControl>
Profile is: [<code><netui:label value="
   {pageContext.getProfileFromRequestResult}"></netui:label> </code>]<br>
```

If the user is registered, then the profile can be retrieved without a reference to the session, as shown in the code sample in Listing 5-2. This method is useful if you do not have access to the session object.

Tip: To retrieve a user's profile using this programmatic technique, the user must be logged in and authenticated. If you call `com.bea.p13n.security.Authentication.login()` to perform the login, the user profile is automatically created. You can also call the WebLogic Server method `weblogic.servlet.security.ServletAuthentication.login()`; however, note that the user profile is only created after the next access (usually after the first page

refresh). Before this subsequent access, you will receive a `ProfileNotFound` exception when you try to retrieve the user's profile.

Listing 5-2 Retrieve a Registered User's Profile Without a Session Reference

```
import com.bea.p13n.usermgmt.profile.ProfileFactory;
import com.bea.p13n.usermgmt.profile.ProfileNotFoundException;
import com.bea.p13n.usermgmt.profile.ProfileWrapper;
import java.rmi.RemoteException;
public class MyHelper
{
    public static String helperMethod(String username)
    {
        try
        {
            ProfileWrapper profile =
               ProfileFactory.getProfile(username,null);
            // do something helpful here.
            return profile.toString();
        }
        catch (RemoteException ex)
        {
        }
        catch (ProfileNotFoundException ex)
        {
        }
        return null;
   }
```

For anonymous and tracked anonymous users, you must retrieve the profile from the session. Anonymous profiles have no identity. Tracked anonymous profiles have an identity that is not valid for authentication. A safe way to retrieve the identity for a user, based upon the user's profile type, is shown in Listing 5-3. This code sample retrieves the current `ProfileWrapper` and gets the username associated with the wrapper.

Listing 5-3 Retrieve a User's Identity by Retrieving the Current ProfileWrapper

```
<%@ page import="com.bea.p13n.usermgmt.SessionHelper"%>
Profile Id is: [<code><%= SessionHelper.getUserId(request) %></code>]<br>
```

Use the following returned values to determine the user type:

- **Null** – Anonymous profiles
- **UserId** – Tracking number (which cannot be used for authentication) for tracked anonymous profiles
- **Username** – User Principal name for registered (authenticated) profiles

Editing Properties and Values with Controls

Developers use the `getProperty` and `setProperty` actions in the Property control to let users retrieve property values for a property set and update property values for either the session's current profile or for the anonymous user profile.

WARNING: Properties created with this control do not appear in the WebLogic Portal Administration Console, and you must modify and update them programmatically.

For more information on using the Property control and its properties, see the *Javadoc*.

Listing 5-4 shows how a user can use the `setProperty` action in the Property control to edit a `Profile Wrapper`. An example page flow (and associated JSP) that uses controls to offer a form for the user to set a favorite color is shown in the code sample. This example requires a `Generalnfo.usr` user profile property set file to exist in the \userprofiles folder of the data project, with a single-valued, restricted, text **FavoriteColor** property. For more information, see the help in Workshop for WebLogic.

Listing 5-4 Use this index.jsp file with the Page Flow

```
<%@ page language="java" contentType="text/html;charset=UTF-8"%>
<%@ taglib uri="netui-tags-databinding.tld" prefix="netui-data"%>
<%@ taglib uri="netui-tags-html.tld" prefix="netui"%>
<%@ taglib uri="netui-tags-template.tld" prefix="netui-template"%>
<netui:html>
   <body>
```

```
      <netui:form action="setColor">
         <table>
            <tr valign="top">
               <td>Favorite Color:</td>
               <td>
               <netui:select dataSource="{actionForm.color}"
                  defaultValue="{pageFlow.usersColor}"
                  optionsDataSource="{pageFlow.possibleColors}">
               </netui:select>
               </td>
            </tr>
         </table>
         <br/> 
         <netui:button value="Set Color" type="submit"/>
      </netui:form>
   </body>
</netui.html>
```

Editing Properties and Values with the ProfileWrapper Object

Developers can change user profile property values by calling the `ProfileWrapper` object directly. For more information, see the *Javadoc*.

Retrieving User Profiles

You must specify which set of user or group properties the user should inherit by configuring a `ProfileWrapper` successor at runtime. A `ProfileWrapper` is a lightweight object that can access the correct `ProfileManager` session beans based on the profile identity with which it is initialized. The `ProfileManager` has a `getAllProfileNames` method and a `listAllProfiles` method. The `listAllProfiles(int pageSize)` method is new to WebLogic Portal 10.0 and efficiently retrieves all user profiles or group profiles. See the Javadoc for more detail.

Deleting a Property Set and Properties

You can use Workshop for WebLogic to delete individual properties from a property set, or you can delete an entire property set.

This section includes the following topics:

- Deleting a Property
- Deleting a Property Set
- Using Properties from an External User Store

Deleting a Property

To delete individual properties from a property set:

1. In the **Workshop for WebLogic Navigator** window, expand the data\src\userprofiles folder, and double-click the user profile property set you created.
2. Select the property in the user profile editor window.
3. Right-click the property and choose **Delete**. The property is deleted from the user profile property set.

Deleting a Property Set

To delete a property set:

1. In the **Workshop for WebLogic Application** window, right-click the data\src\userprofiles folder, and select the user profile you created.
2. Right-click the property set and choose **Delete** to remove the property set.
3. Click **Yes** to confirm the deletion.

You can also use the `<profile:removeProperty>` JSP tag or the `removeProperty` action in the Property control in your page flows to remove existing properties or profiles for users. See the *Javadoc* for more information.

Using Properties from an External User Store

If you created a UUP to access external user or group properties, you can use those properties to define rules for personalization, delegated administration, or visitor entitlement.

After you create a UUP to access these properties in the external user store (for example, an openLDAP server) you can access those external properties only through WebLogic Portal's JSP tags, controls, or APIs. Those external properties are not yet accessible in the Administration Console.

You must surface those external properties in the Administration Console if you want to use those properties in defining rules for personalization, delegated administration, or visitor entitlement.

Note: If the properties you surface from an external user store are read-only, you cannot update them in the Administration Console. To make those properties writable, your custom UUP would have to become writable.

To use properties from an external user store:

1. Create a UUP for the external user store. See Chapter 6, "Configuring a UUP" for instructions.
2. In Workshop for WebLogic, create a user profile property set for the external user store. The name you give the property set *must* match the name of the provider's `PropertyMapping`. To find the name of the property set, perform the following steps:
   a. Look in your enterprise application root directory and open the META-INF/p13n-profile-config.xml file.
   b. In the `<!-- User Profile Manager -->` section, locate the name entry for your external user store, such as:

      ```
      <property-adapter>
         <name>MyExternalStore</name>
         <property-mapping>MyExternalPropertySet</property-mapping>
         <ejb-jndi>my_uup.jar#ExternalEntityPropertyManager</ejb.jndi>
      </property-adapter>
      ```

      The name following the `property-adapter` line is the name you must give to the new property set. The name is case sensitive. For example, in the example shown, the property set would be named `MyExternalStore.usr`.
   c. If you are using the LDAP UUP provided by WebLogic Portal, name the property set `ldap.usr`.
3. Add properties to the property set that exactly match the property names in the external store you want to surface.
4. Save the property set file.

Note: After you have deployed your portal application to production, any modifications you make to user profile properties in Workshop for WebLogic must be pushed to the running server. For more information, see the *Production Operations Guide*.

CHAPTER 6

Configuring a UUP

WebLogic Portal includes a Unified User Profile (UUP) service that lets you add and manage users and their properties in a single logical location—even if the user data is stored in external systems, such as an LDAP server. You can add this additional data to a user's profile.

A UUP extension lets WebLogic Portal read property values stored in external data stores, such as openLDAP servers, legacy applications, Netscape iPlanet user stores, and flat files. If you have an existing provider with users, groups, and additional properties (such as address, e-mail address, phone number, and so on), you can use a UUP to bring those user properties into WebLogic Portal.

A UUP allows you to access existing user information without migrating that data into the portal schema. See "Planning to Use a UUP" on page 2-6 to determine when to use UUP.

Whether or not you have additional properties stored in your external user store, the external users and groups you connect to WebLogic Portal are automatically assigned the default user property values you have set up in WebLogic Portal (without using a UUP). With the Administration Console, you can change the WebLogic Portal property values for those users. These values are stored in WebLogic Portal's RDBMS user store using the portal schema.

In WebLogic Portal, you can retrieve and edit user property values and use those property values to set up personalization, delegated administration, or visitor entitlement. See the *Security Guide* for instructions on setting up delegated administration and visitor entitlement.

Note: In WebLogic Portal 8.1, you configured a UUP by creating an EJB and then adding the appropriate descriptors to the `ejb-jar.xml` file and the `weblogic-ejb-jar.xml` file in the `p13n-ejb-jar` file. In WebLogic Portal 10.0, you can use Workshop for WebLogic during portal application development, or you can use the Administration Console during

portal runtime to register a UUP EJB. See "Choosing a Method to Configure a UUP" on page 6-4 and "Creating a UUP EJB" on page A-1.

Figure 6-1 shows where a UUP fits between an external user store and the WebLogic product environment.

Figure 6-1 Unified User Profile

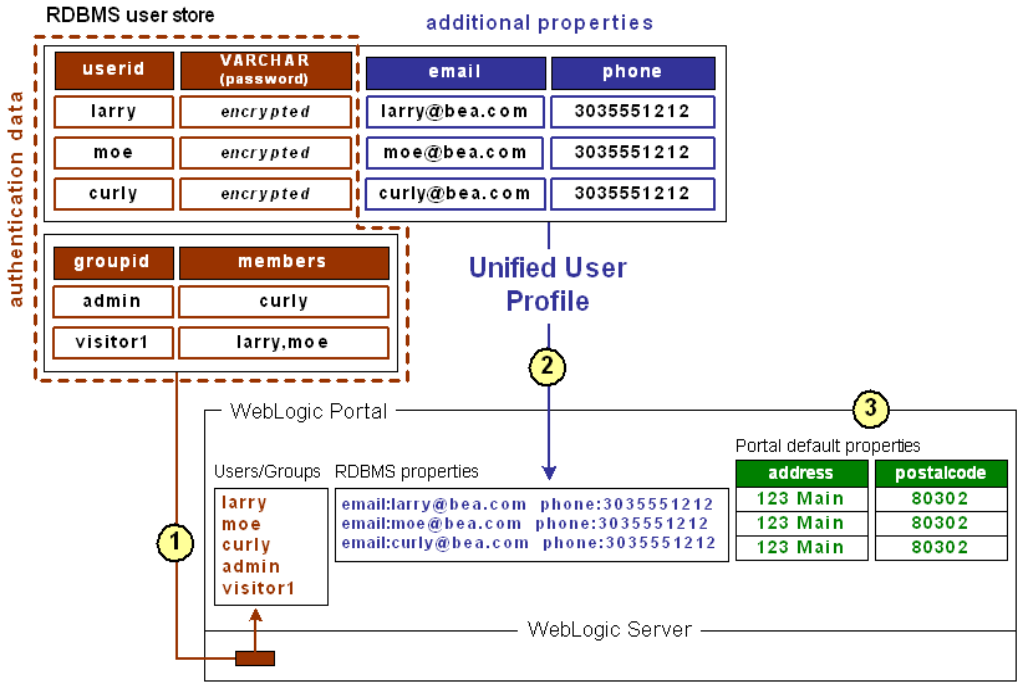


Table 1 Unified User Profile

1 **External RDBMS User Store** – The user store supports authentication, and contains users and passwords in one database table and groups in another. Giving a user store authentication capabilities (as an authentication provider or identity asserter) involves configuration steps that are not associated with the UUP configuration process. UUP configuration is not dependent on the user store configuration and vice versa.

   After the RDBMS user store is connected to WebLogic Server, both WebLogic Server and WebLogic Portal can see those users and groups. Those users can log into your portal applications, and you can include those users and groups in your rules for personalization, delegated administration, and visitor entitlement. Also, WebLogic Portal's `ProfileWrapper` maps the principals to properties kept in the portal schema, thereby establishing the user profile.

2 **UUP** – The same external table that contains users and passwords also contains additional properties (e-mail and phone) for each user. These additional properties are not part of authentication, but they can be part of each user's profile. If you want to access these properties in your portal applications (with the WebLogic Portal JSP tags, controls, or API), you must configure a UUP for the RDBMS user store. When you configure the UUP, the `ProfileWrapper` includes the external properties in the user profile. The UUP extension consists of a stateless session bean and associated classes that you create.

   If you want to surface any of these properties in the Administration Console to define rules for personalization, delegated administration, or visitor entitlement, create a user profile property set for the external user store in addition to implementing your UUP session bean. The property set provides metadata about your external properties so that Workshop for WebLogic and the Administration Console can display them.

   Properties from an external data store are typically read-only in the Administration Console.

3 **BEA Workshop for WebLogic Platform** and **Administration Console** – You can create default user and group properties and set default values for those properties. Any user or group in WebLogic Server, whether created in the default RDBMS user store or brought in through a connection to an external user store, is automatically assigned those default property values. You can change the default values for each user or group, either programmatically or in the Administration Console. This does not involve a UUP, because the properties being retrieved are WebLogic Portal properties, so they are not stored in an external user store.

   After the user store or identity asserter provides the users and groups, as shown in Figure 6-1, the `ProfileWrapper` combines the users and groups with the external properties of e-mail and phone (retrieved by the UUP) and the default WebLogic Portal properties of address and postal code, all of which make up the complete user profile.

The chapter includes the following sections:

Choosing a Method to Configure a UUP

In WebLogic Portal 8.1, you could configure a UUP manually, but that method is no longer supported. See the WebLogic Portal 8.1 documentation for more information.

WebLogic Portal now provides two ways to configure a UUP:

1. **Workshop for WebLogic** – Edit the `p13n-profile-config.xml` descriptor file in Workshop for WebLogic to configure the UUP extension during the portal development phase. The descriptor file is deployed with the packaged enterprise application. A UUP that you configured in the Administration Console at runtime uses the deployment plan and takes precedence over the same UUP that you create in Workshop for WebLogic during portal development.

2. **UUP in the Administration Console** – After you deploy your portal, you can still configure a UUP in the Service Administration menu in the Administration Console. A UUP that you create in the Administration Console modifies the deployment plan for the application and requires a redeployment. See the *Production Operations Guide* and the *WebLogic Server* documentation for more information. A UUP that is configured in the Administration Console takes precedence over a UUP created in Workshop for WebLogic or a manual configuration from EJB environment properties.

   See "Verifying the UUP" on page 6-8. Existing UUPs that you configured manually in previous versions of WebLogic Portal can co-exist with new UUPs that you configure in the Administration Console or in Workshop for WebLogic. See "Creating a UUP EJB" on page A-1 for instructions for manually configuring a UUP for Portal 9.2.

Configuring a UUP in the Administration Console

You can retrieve data from an external source by configuring a UUP in the Administration Console. The steps below assume that you have created a new portal domain in Workshop for WebLogic. This section contains the following topics:

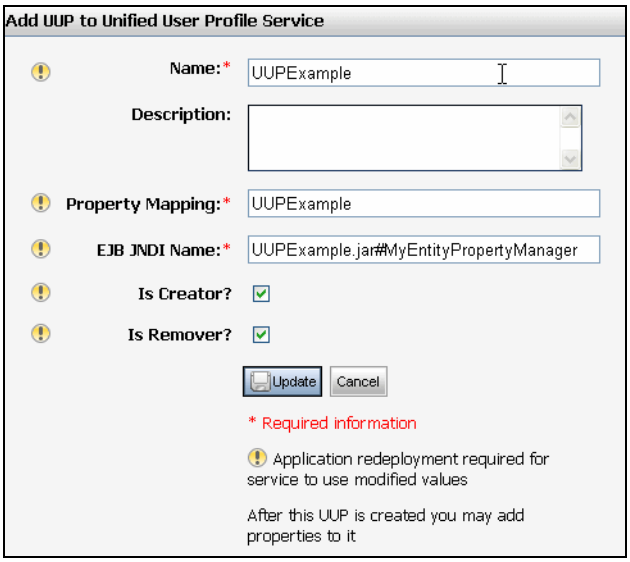
- Configuring a UUP in the Administration Console
- Editing a UUP in the Administration Console

Before you configure the UUP, you must create an `EntityPropertyManager` Enterprise Java Bean (EJB) to represent the external data. See "Creating and Configuring an EntityPropertyManager EJB" on page A-2 for instructions.

To configure a UUP and add properties to it using the Administration Console:

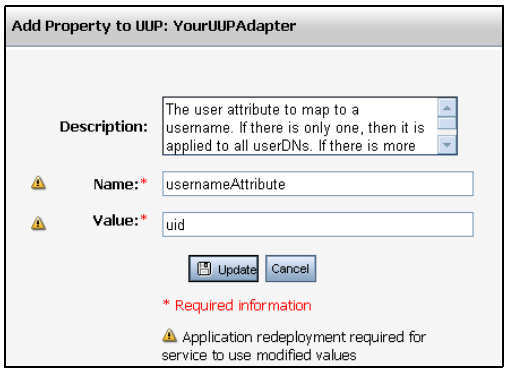
1. After you create the `EntityPropertyManage` EJB, open the Administration Console by launching a web browser and entering the URL for the application.
2. After you log into the Administration Console, select **Configuration & Monitoring** > **Service Administration**.
3. In the Resource Tree, select **Unified User Profiles**.
4. In the Browse tab, click **Add UUP**.
5. In the **Name** field, enter an appropriate name for this new UUP and complete the **Description** field. For example, enter `UUPExample`. See Figure 6-2.
6. In the **Property Mapping** field, enter the name of the property set that represents a namespace of the external data. This field is the name of a property set or a single property. For example, enter `UUPExample`.
7. In the **EJB JNDI Name** field, enter the name of the new EJB Java Naming and Directory Interface (JNDI) that accesses the data in the **Property Mapping** field that you defined in step 6. The EJB JNDI directs the **Profile Manager EJB** to find the new `PropertyManager` at runtime. The format of this field is *<UUP JAR file name>#<EJB name defined in UUP ejb-jar.xml>*. For example, `UUPExample.jar#MyEntityPropertyManager`.
8. Select the **Is Creator** and the **Is Remover** check boxes if the `PropertyManager` you created earlier implements certain methods and receives callbacks from the `ProfileManager` when profiles are created and removed. The default for these fields is unchecked. See Figure 6-2.

Figure 6-2 Configure A New UUP



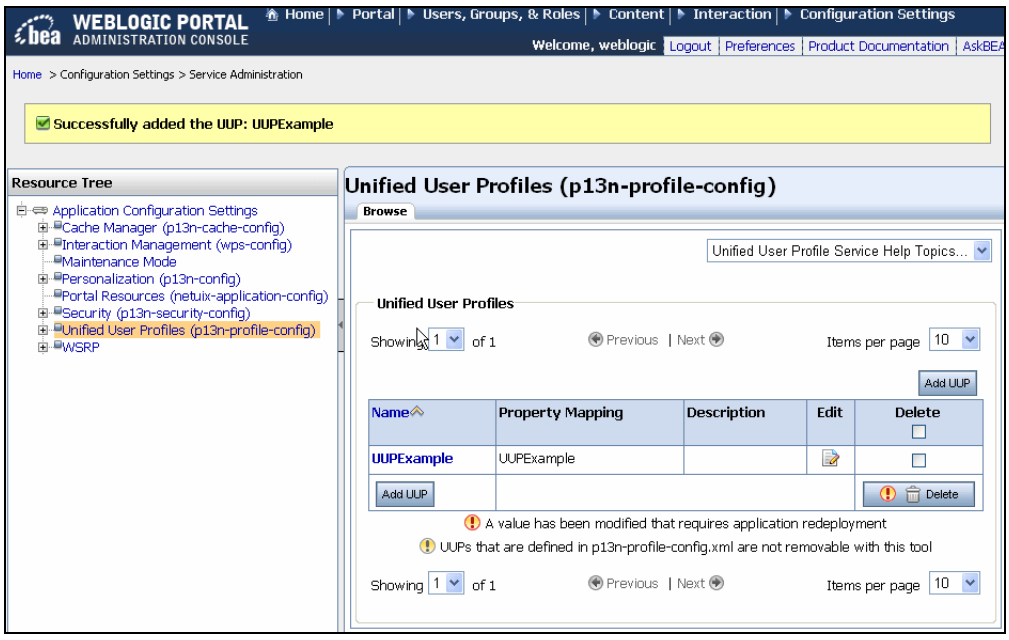
9. Click **Update**.
10. After you create a UUP, you can add specific UUP adapter configuration parameters (called adapter properties) to it. Adding configuration parameters is optional. On the Browse UUPs tab, click the name of your new UUP.
11. In the **UUP Details** page, click **Add Property**.
12. In the **Description** field, enter text to represent each UUP adapter.
13. In the **Name** field, enter a unique name that matches the name of the deployed custom `PropertyManager` in the EJB descriptor file.
14. In the **Value** field, define a value for the new property. A new property is shown in Figure 6-3.

Figure 6-3 Add Properties to a New UUP



15. Click **Update**. The new UUPExample appears in the Browse tab, as shown in Figure 6-4.

Figure 6-4 The New UUP



16. Log out of the Administration Console and stop the WebLogic Server, if it is running. Stopping the server results in minimal down time using this method to configure the UUP.

17. You must redeploy your application for the new UUP and new properties to take effect. During the deployment, the UUP adapter configurations are merged from the deployment plan and the `p13n-profile-config` descriptor file stored in the META-INF directory of the portal application. Then the `ProfileManager` resolves the EJB references to the deployed custom `PropertyManager` using the name in the **EJB JNDI Name** field. If the references resolve successfully, property set mappings are created according to the property-adapter section specifications in the descriptor bean. At runtime, the properties in this property set are routed to the new `PropertyManager`. The new `PropertyManager` can also use descriptor beans to retrieve adapter-specific configuration parameters stored in one or more of the `<adapter-property>` fields.

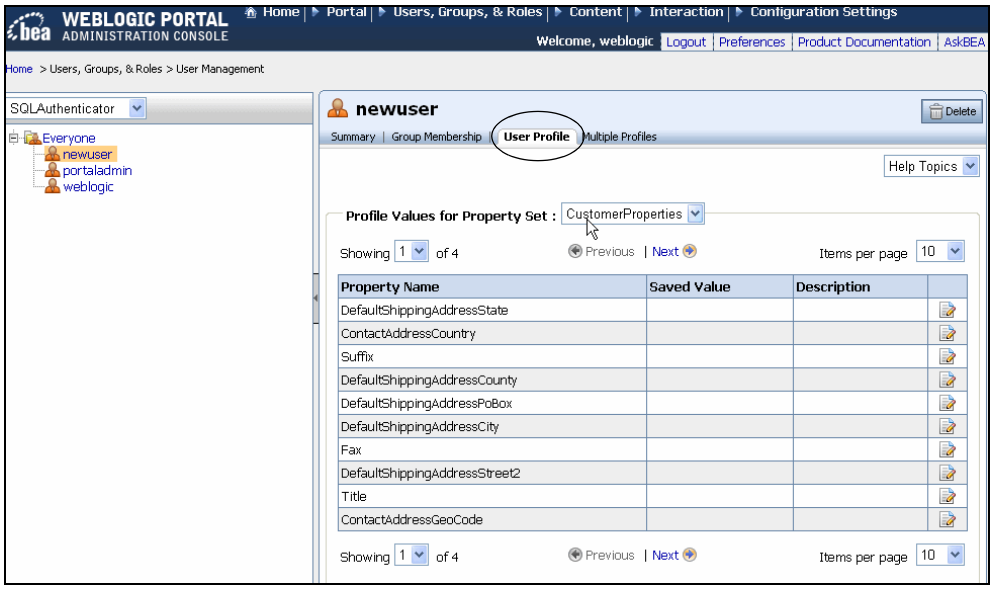
Verifying the UUP

After you configure the UUP in the Administration Console (or Workshop for WebLogic), you should verify the UUP in the Administration Console.

To verify the UUP:

1. In the Administration Console, choose **Users, Groups, & Roles > User Management**.
2. In the User Tree, select **Everyone**.
3. In the Browse tab, click **Create New Users**.
4. Enter the user's Name and a Password, and click **Create User**.
5. In the User Tree, select the name of the new user and select the **User Profile** tab. Figure 6-5 appears.

Figure 6-5 User Profile Tab for the New User



6. From the **Profile Values for Property Set** field, select your UUP from the drop-down list. For example, select `UUPExample`.
7. Click the **Edit** icon next to the **attribute1** property.
8. In the Edit Profile dialog, enter a value for the field and click **Update**. The new value appears.

Note: You can also verify that the data exists by checking your database.

Editing a UUP in the Administration Console

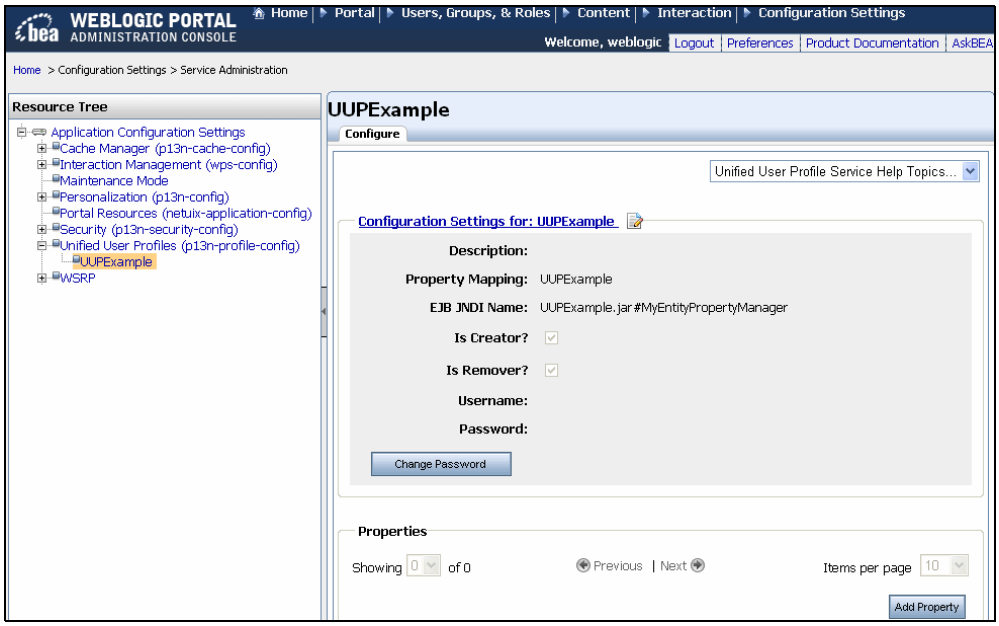
You can use the Administration Console to change the configuration settings or properties for your UUP.

To edit a UUP:

1. In the Administration Console, choose **Configuration & Monitoring > Service Administration**.
2. Select **Unified User Profiles.**
3. In the Browse UUPs tab, locate the UUP you want to change and click **Edit**.

4. Click the **Configuration Settings for: *UUPName*** link, as shown in Figure 6-6.

Figure 6-6 Click the Configuration Settings for: UUPExample Link to Edit It



5. Enter your edits and click **Update**.

   Note: You can also edit the properties in your UUP by locating the property name in the UUP Browse tab and clicking **Edit**.

6. In the Properties section, click **Edit**.
7. In the Edit Property dialog, enter a new **Description** or a new **Value**, and click **Update**.
8. You must redeploy your application for the changes to take effect.

Configuring a UUP in Workshop for WebLogic

A second way to retrieve data from an external source is to configure a UUP in the Workshop for WebLogic. (The other method is through the Administration Console; see "Configuring a UUP in the Administration Console" on page 6-4.) A descriptor file called `p13n-profile-config.xml` lets you define the mappings for the UUP.

Before you configure the UUP, you must create an `EntityPropertyManager` Enterprise Java Bean (EJB) to represent the external data. See "Creating the EJB" on page A-2 for instructions.

To create a UUP in Workshop for WebLogic:

1. After you create the `EntityPropertyManage` EJB, start Workshop for WebLogic, open the Portal Perspective, and open the Merged Projects View.
2. Open an existing Portal EAR project or create one (see the *Portal Development Guide* for instructions on creating an EAR project). Locate the `p13n-profile-config.xml` file in the <UUPApp>/EARContent/META-INF directory.
3. If the `p13n-profile-config.xml` file is italicized, it exists in a library module and is not in the project itself. Select the file, right-click, and choose **Copy To Project**. The filename changes from italics to a normal font.
4. Change to the Package Explorer View, and navigate to the <UUPApp>/EARContent/META-INF directory.
5. Open the `p13n-profile-config.xml` file by double-clicking it.
6. Add the following entry to the file:

```
<property-adapter>
     <name>UUPExample</name>
     <description>UUP EJB</description>
     <property-mapping>UUPExample</property-mapping>
     <ejb-jndi>UUPExample.jar#MyEntityPropertyManager</ejb-jndi>
     <is-creator>true</is-creator>
     <is-remover>true</is-remover>
</property-adapter>
```

7. Save the file.
8. Create a new WebLogic Server definition and associate the new server with a domain. For instructions, see the *Portal Development Guide*.
9. Associate your UUP application with your server by select the server in the Servers tab, right-clicking, and choosing **Add and Remove Projects**. Select the UUP application from the **Available Projects** section, click **Add**, and then click **Finish**.
10. Build and publish your application. Verify the application by starting the WebLogic Server Administration Console and clicking **Deployments**. Verify that the UUP application is active. Then open the UUP application by expanding the tree and verifying that the UUP JAR file appears as an EJB.

11. Verify the UUP by following the instructions in "Verifying the UUP" on page 6-8.

Editing a UUP in Workshop for WebLogic

You can use Workshop for WebLogic to change the configuration settings or properties for your UUP.

To edit a UUP in Workshop for WebLogic:

1. In Workshop for WebLogic, change to the Package Explorer View and navigate to the <UUPApp>/EARContent/META-INF directory.
2. Open the `p13n-profile-config.xml` file by double-clicking it.
3. Edit the `<property-adapter>` entry for your UUP.
4. Save the file.
5. If y ou have already associated your UUP application with the server, build and republish the application.
6. Verify the UUP by following the instructions in "Verifying the UUP" on page 6-8.

Upgrading a UUP

Your WebLogic Portal 9.2 or 9.2 MP1 UUP automatically works in WebLogic Portal 10.0. You do not need to upgrade your Portal 9.2 UUP.

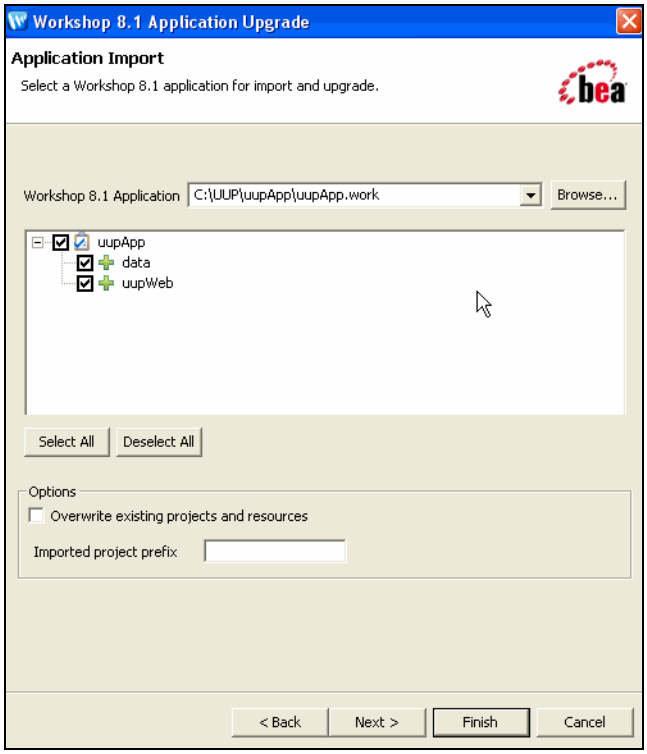
When you upgrade a UUP from WebLogic Portal 8.1, the `p13n_ejb.jar` file is deleted and replaced with a new WebLogic Portal version of this file. The new `p13n_ejb.jar` file is packaged in the library modules that ship with WebLogic Portal.

To upgrade a UUP configured in WebLogic Portal 8.1 to Portal 9.2:

1. Start BEA Workshop for WebLogic Platform and create a new Workspace.
2. Create a new portal domain. Do not create a Portal EAR Project. For instructions on creating a new domain, see the *Portal Development Guide*.
3. Import your Portal 8.1 UUP application into your new environment by choosing **File > Import**.
4. In the Select dialog, open the **Other** folder, and select **Workshop 8.1 Application**, and click **Next**.

5. In the Application Import dialog, click **Browse** and locate your 8.1 UUP application. Select the `.work` file and click **Open**. Verify that the check boxes for the UUP application are selected and click **Next**, as shown in Figure 6-7.

Figure 6-7 Locate the 8.1 UUP Application



6. In the Source Upgrade dialog, click **NetUI Project Upgrader options** and select the **Use WebLogic 9.0 J2EE Shared Libraries** check box. You can also select the **Replace BEA NetUI tags with Apache Beehive tags** check box (if desired) in JSP File Migrator options and click **Finish**.

7. After the upgrade finishes, verify that the following actions occurred:

   – The `p13n-ejb.jar` file was removed from the EARContent directory of the UUP application.

   – The UUP EJB JAR file (for example, `UUPExample.jar`) exists in the EARContent directory of the UUP application.

- The UUP EJB JAR file is referenced in a module entry in the `application.xml` file in the <UUPApplication>/EARContent/META-INF/ directory.
- As an example, the cache entry below was added to the `p13n-cache-config.xml` file in the <UUPApplication>/EARContent/META-INF/ directory:

```
<p13n:cache>
     <p13n:name>UUPExampleCache</p13n:name>
     <p13n:description>Cache for UUP Example</p13n:description>
     <p13n:time-to-live>60000</p13n:time-to-live>
     <p13n:max-entries>100</p13n:max-entries>
</p13n:cache>
```

- Verify that the user profile file (for example, `UUPExample.usr`) file exists in the data/src/userprofiles/ directory (or where your Datasync folder exists).

8. Associate your portal application with your WebLogic Server by selecting the server in the Servers tab, right-clicking the server, and choosing **Add and Remove Projects**. Select the UUP application from the **Available Projects** section, click **Add**, and then click **Finish**.
9. Build and publish your application. Verify the application by starting the WebLogic Server Administration Console and clicking **Deployments**. Verify that the UUP application is active. Then open the UUP application by expanding the tree and verifying that the UUP JAR file appears as an EJB.

For more information about upgrading other non-portal applications from WebLogic Portal 8.1, see the *Upgrade Guide*.

CHAPTER 7

Setting Up Anonymous User Tracking

An anonymous user is someone who visits your portal without logging in. Working with anonymous users can be limiting because you have less information about them. However, you can use tracking tools to find out more about these users, such as the date and time they visit and information in the request or session.

Anonymous user tracking can help you present personalized content to an anonymous user over multiple sessions. For example, you can set up visitor entitlement, delegated administration, and some types of personalization based on that information. Anonymous user tracking gives anonymous users records in a database, where you can store meaningful information about them, such as personal preferences.

Developers use Workshop for WebLogic to programmatically set up anonymous user tracking. You cannot use the Administration Console to set up anonymous user tracking.

Note: Campaigns and behavior tracking are not currently supported for anonymous, non-trackable users.

This chapter includes the following sections:

Using 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 (the default is 30 seconds), and preference information is persisted in a database rather than in memory.

Each time an anonymous tracked user returns to your portal, the ID in the cookie matches the primary key in the tracked anonymous user database, the previous user properties are maintained, and your personalization and campaign rules work for this user. When an anonymous tracked user registers in your portal, the 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.

Anonymous user tracking can be customized to exclude users who spend less than a designated period of time on your site, avoiding the expense of storing data on irrelevant users.

This section includes the following topic:

Choosing a User Tracking Profile Type

WebLogic Portal supports the following types of user tracking profiles:

- **Anonymous Users** – If anonymous user tracking is not enabled, users who visit the site without registering are considered anonymous, non-tracked users. There is no ID associated with the user.
- **Trackable Anonymous Users** (also called Trackable Users) – If you enable anonymous user tracking, anonymous users are considered trackable on the first visit. If a trackable anonymous user remains at a site longer than the specified duration, the user is automatically converted into a tracked anonymous user. An ID is associated with the user, but the ID is not persisted.
- **Tracked Anonymous Users** (also called Tracked Users) – If a trackable anonymous user remains at a site longer than the specified duration, the user is automatically converted 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 when the session ends, the user's properties are persisted and available for the next visit. If a 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 can be used to enable personalization and run campaigns against the user.

- **Registered Users** – If a tracked anonymous user registers at your site, the data collected on that user is transferred to the new user (now considered a registered user) and is deleted from the anonymous user database.
- **Unknown Users** – Unknown users include users whose status cannot be determined. When the unknown user's session begins, the `ProfileWrapper` is not yet created.

Setting Up Anonymous User Tracking

You must configure anonymous user tracking by editing the `web.xml` file in your web application directory in Workshop for WebLogic. You will enable cookies on the user's browser in order to track anonymous users. You must also enable anonymous user tracking for your portal, because the tracking is not enabled by default.

When you configure anonymous user tracking, you perform the following tasks:

- **Enable Anonymous User Tracking** – Use this feature to identify and retain information about non-authenticated visitors to your portals.
- **Configure visit duration for anonymous users** – Users do not always remain at a site long enough to demonstrate interest, so 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 user that stays 30 seconds or more is remembered, even if the user does not register.
- **Create anonymous user profiles** – User profiles can store personal information about anonymous users to use later for personalization or campaigns.

To edit the `web.xml` file to enable anonymous user tracking:

1. In Workshop for WebLogic, locate the `web.xml` file, which is typically located in the /WEB-INF directory in your web application.
2. Double-click the `web.xml` file to open it. You can also edit the `web.xml` file in a text editor.
3. Locate the `PortalServletFilter` section, which begins with the following code sample:

```
<filter-name> PortalServletFilter </filter-name>
<filter-class> com.bea.p13n.servlets.PortalServletFilter
   </filter-class>
```

4. Change the `fireSessionLoginEvent` parameter value to `false`, as shown in the following code sample:

```
<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>
```

5. To create a user profile that can store information about each anonymous user, change the `createAnonymousProfile` parameter value to `true`, as shown in the following code sample.

```
<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>
```

   The default is `true`. To disable the creation of anonymous user profiles, set the value to `false`. If the `enableTrackedAnonymous` parameter is set to `true`, the `createAnonymousProfile` parameter is ignored.

   Note: The `createAnonymousProfile` parameter does not enable anonymous user tracking from one session to the next without registering; this would require the `enableTrackedAnonymous` value to be changed to `true`. 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 that registered user.

6. To begin tracking anonymous users, change the `enableTrackedAnonymous` parameter value to `true`, as shown in the following code sample:

```
<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>
```

7. Change the `trackedAnonymousVisitDuration` parameter value to the number of seconds before tracking begins. The following code sample shows that tracking is configured to begin after five seconds:

```
<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>
```

8. Save your edits to the `web.xml` file.

Tip: Tracking anonymous users can potentially generate a large amount of data. You might want to develop a strategy to remove old data. For example, you could delete entries from the database that are older than two months. If a tracked anonymous user is converted to a registered user, you could remove the tracked anonymous user EJB and its associated property set EJBs from the database.

Targeting Anonymous Non-Tracked Users

You can target anonymous non-tracked users with personalized content using a personalization tool, such as default (non-campaign) placeholders and content selectors. Campaigns and behavior tracking are not currently supported for anonymous, non-tracked users.

Tip: Campaigns and behavior tracking are not officially supported for anonymous users that have not logged in, but your BEA Professional Services representative can help you create a campaign that displays personalized content for this type of visitor.

The difference between completely anonymous non-tracked users and registered or anonymous tracked users is how long their profile information is retained.

For example, if an anonymous user visits a portal and sets 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, the user preferences must be re-entered. Registered and anonymous tracked user preferences are saved in a database.

While anonymous users might not retain a consistent store of user preferences from session to session, you can still provide some Interaction Management functionality. Many personalization conditions are based on generic HTTP session and request properties that you define, dates and times, or personal session preferences that have no relationship to registered users and their profile properties. For example, you could display personalized content for anonymous users accessing your portal with a specific type of browser during a specific time frame.

See the *Interaction Management Guide* for instructions on setting up placeholders and content selectors for anonymous non-tracked users.

Part III Staging

Part III includes the following chapters:

- Chapter 8, “Adding and Managing Groups”
- Chapter 9, “Adding and Managing Users”
- Chapter 10, “Editing User Profile Property Values”

This section contains instructions for adding administrative users to your portal application and testing the functionality you created in the Development phase. This Staging environment simulates a Production environment.

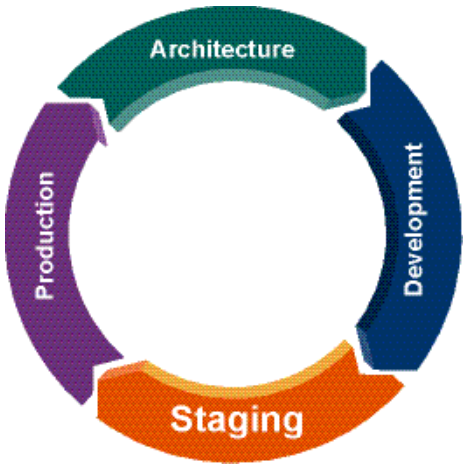
Consider setting up a common development environment for the Development phase and the Staging phase. You might move iteratively between these two phases, developing and then testing what you created in the Administration Console.

Use the Staging phase to test the connection to the user stores that you will use in the Production phase. The Staging phase is also a good place to verify that delegated administration and visitor entitlement that you set up are working. See the *Security Guide* for instructions on setting up delegated administration and visitor entitlement. You can also review your personalization logic (see the *Interaction Management Guide*) and content management in this phase (see the *Content Management Guide*).

When you move to a Production environment in the Production phase, your users and groups from the Staging phase are not automatically moved there. BEA propagation tools do not move user and group data because of the hashed passwords in the WebLogic Server domain. Hashed passwords provide security to a system’s user so that others cannot use the password to log into

other sites the user frequents. See the *Production Operations User Guide* for more information on deployment and propagation.

For a description of the Staging phase of the portal life cycle, see the *WebLogic Portal Overview*. The portal life cycle is shown in the following graphic:



CHAPTER 8

Adding and Managing Groups

After you use Chapter 2, "Planning a User and Group Strategy" to determine your group structure, you can create, move, and delete a small number of groups in the Administration Console. Developers might prefer to do these same tasks for a large number of users in Workshop for WebLogic with JSP tags and controls.

You can also use groups to set up delegated administration and visitor entitlement; see the Administration Console. If the group exists before you add a user, you can then immediately add the user to the group.

Portal administrators with full group management rights can use the following tools to create and manage a small number of groups:

- **WebLogic Portal Administration Console** – Add and edit groups here if you work primarily in the Administration Console.
- **WebLogic Server Administration Console** – Add and edit groups here if you work primarily in the WebLogic Server Administration Console. See the *BEA WebLogic Server Securing WebLogic Resources Guide* for instructions.

Developers can use the following tools to create and manage a large number of groups:

- **JSP tags** – Use the `createGroup` JSP tag to create a new group. Other JSP tags let you add a subgroup, remove a group or subgroup, and create a group profile. See "Creating a Group with a JSP Tag" on page 3-3.
- **Controls** – Use the `createGroup` action in the Group Provider control to create a new group. Other controls allow you to add a subgroup, remove a group or subgroup, and create a group profile. See "Creating a Group with a Control" on page 3-3.

- **Java** – Work directly with the `com.bea.p13n.security.management.authentication.AtnManagerProxy` Java class to add groups, create group profiles, and remove groups. See the *Javadoc* for more information.

Tip: Users and groups are stored in the RDBMS user store server by default. The BEA Propagation Utility does not propagate this data from your staging environment to production.

The chapter includes the following sections:

Using Existing Groups

If the user store you are accessing already has users organized into groups, you can build a hierarchical tree in the Administration Console to organize and manage those existing groups. If you are using WebLogic server's default RDBMS user store, you do not need to build a group hierarchy tree.

This section contains the following topics:

Using a Group Hierarchy Tree

A tree view of groups provides a visual way to change group properties, find users in groups, and add users and groups to rules for delegated administration and visitor entitlement. Figure 8-1 shows a group hierarchy tree for the default *SQLAuthenticator*.

Figure 8-1 Group Hierarchy Tree

If you are connecting WebLogic Server to an external user store, your ability to see a hierarchy tree in the Administration Console depends on how the user store is supported. If the provider does not allow read access by external tools (such as the Administration Console), you will not be able to see the tree representation of the groups. See the *Security Guide* to determine if your user store allows read access. If the provider does not allow read access to its groups, you can still use the Administration Console to enter the names of known users and groups.

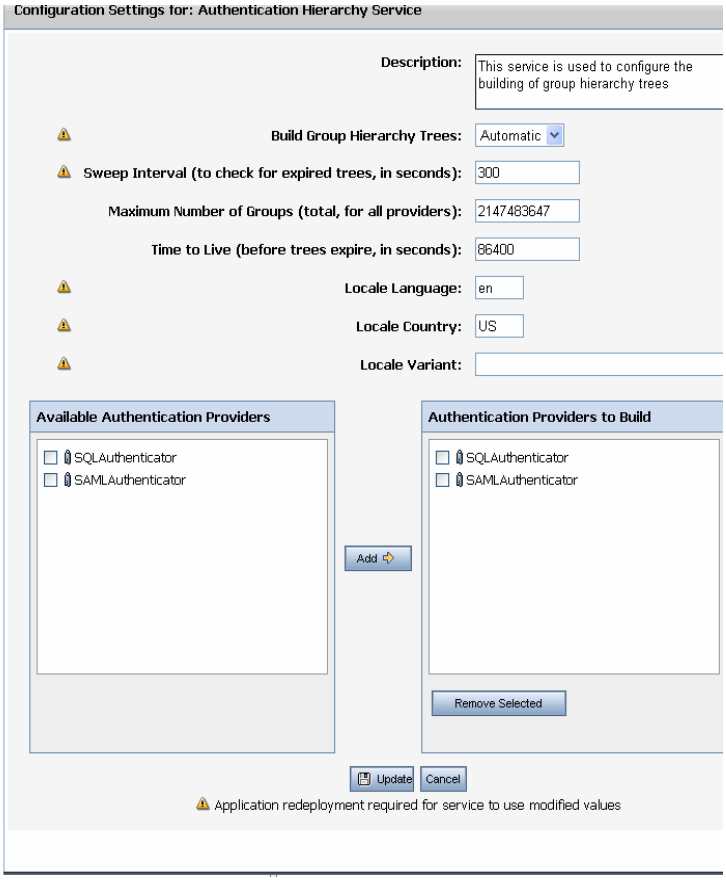
Building a Group Hierarchy Tree

Use the **Authentication Hierarchy Service** page to build and configure a group hierarchy tree for a read-access user store connected to WebLogic Server.

To build a group hierarchy tree:

1. Start the Administration Console.
2. In the Administration Console, choose **Configuration & Monitoring > Service Administration**.
3. In the Resource Tree, select **Security**.
4. In the Browse tab, click **Authentication Hierarchy Service**.
5. Click **Configuration Settings for: Authentication Hierarchy Service** to display the provider's settings, as shown in Figure 8-2.

Figure 8-2 Retrieve Existing Groups by Building a Group Hierarchy Tree



6. In the **Description** field, enter text that describes this group hierarchy tree. For example, **iPlanetAuthenticator**.
7. In the **Build Group Hierarchy Trees** field, select one of the following choices:
   - **Automatic** – After the server starts or the application redeploys, group hierarchy trees are automatically built for the user stores listed in the **Authentication Providers to Build** list.
   - **Manual** – Group hierarchy trees for the user stores listed in the **Authentication Providers to Build** list are built when you click **Update**, letting you determine when the processing overhead for building the tree occurs.

Note: When you change the value of the **Build Group Hierarchy Trees** field, you must redeploy your enterprise application or restart the server.

8. In the **Sweep Interval** field, specify how often the hierarchy trees are refreshed to show changes to users and groups. The **Sweep Interval** works with the **Time to Live** setting. The **Sweep Interval** determines how often (in seconds) the hierarchy trees are checked to see if they have expired (the time specified in the **Time to Live** field has ended). The default value is 300 seconds (five minutes).

   If a sweep finds the trees expired, the trees are cleared from memory and are not rebuilt until you try to access them in the Administration Console. Refreshing the trees more frequently can impact performance, but changes to users and groups are picked up more frequently.

   Tip: Set the **Sweep Interval** to the same value as **Time to Live** if you want the trees to be cleared from memory as soon as they expire. When you change this value, you must redeploy your enterprise application or restart the server.

9. In the **Maximum Number of Groups** field, enter the highest possible number of groups that will be built and added to memory for all user stores.

10. In the **Time to Live** field, enter how often the hierarchy trees should be refreshed to show changes to users and groups. The **Time to Live** field works with the **Sweep Interval** to determine how often (in seconds) the trees should be cleared from memory (expired). Set the **Time to Live** field to the same value as the **Sweep Interval** field if you want the trees to be cleared from memory as soon as they expire.

11. In the **Locale Language** field, enter the two-digit language abbreviation (for example, **en** for English). The locale settings determine how the lists of users and groups are sorted. When you change this value, you must redeploy your enterprise application or restart the server.

    Tip: You can configure the default **Locale Language** in the `netuix-config.xml` file in the `WEB-INF/` directory and also in the user's preferred language setting in the browser. If the user's preferred language setting is empty, some browsers will automatically choose the default **Locale Language** on the client machine.

12. In the **Locale Country** field, enter the two-letter country abbreviation. When you change this value, you must redeploy your enterprise application or restart the server.

13. In the **Locale Variant** field, enter a vendor or browser-specific code. For example, use **WIN** for Windows, **MAC** for Macintosh, and **POSIX** for POSIX. Where there are two variants,

separate them with an underscore, and put the most important one first. For example, a Traditional Spanish might use a locale with parameters for language, country and variant as: **es**, **ES**, **Traditional_WIN**. When you change this value, you must redeploy your enterprise application or restart the server.

14. In the **Available Authentication Providers** field, select the user store, and click **Add** to build a hierarchy tree. User stores must allow read access before you can build a hierarchy tree.
15. The **Authentication Providers to Build** field shows the user stores for which hierarchy trees are built. The **Build Group Hierarchy Trees** field described in step 7 determines when the trees are built.

    To remove a hierarchy tree for a user store in the Administration Console, select the name of the user store you want to remove, click **Remove Selected**, and click **Update**. The providers you can remove are listed in the **Authentication Providers to Build** field. After you remove the ability to build a hierarchy tree for a provider, you can still use a text entry field in the Administration Console to select users and groups for that provider.
16. Click **Update**.
17. Repeat these steps for each user store that has users and groups you want to see in the Administration Console.
18. To view the group hierarchies for the user store, select **Users, Groups, & Roles > Group Management**. Select the user store you want to view in the drop-down box above the Group tree. You can also see the group hierarchy trees in the Groups in Role tab for a specific delegated administration or visitor entitlement role.

Tip: If a list of groups does not appear, verify that you built a group hierarchy tree for the user store. If you built the tree but you still do not see a list of groups, the user store probably does not allow read access. See the *Security Guide* for instructions on giving the provider read access.

Creating a Group or Subgroup

You can use the Administration Console to add a group during Staging. If you have a large number of groups to add, you can add them programmatically with JSP tags and controls during the Development phase. Typically, you will add just a few groups or make small changes to your group structure using the Administration Console during the Staging phase.

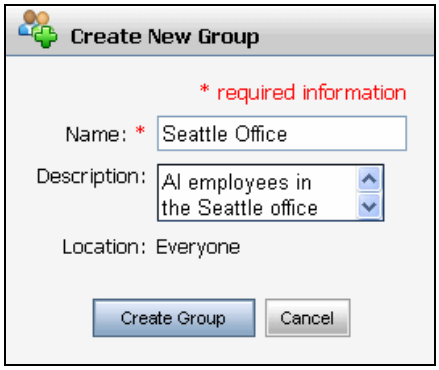
When you create a group, you are creating an empty group to which you can add an infinite number of users. You can also add a group profile, which contains additional information about a group.

Tip: Create groups before you add users to the Administration Console. If the group exists before you add a user, you can then immediately add the user to the group.

To create a group in the Administration Console:

1. In the Administration Console, choose **Users, Groups, & Roles > Group Management**.
2. Select an authentication provider from the drop-down list above the group tree. If you are using an RDBMS user store, group names are case sensitive. A group called **Managers** is different than **managers**.
3. Select **Everyone** to create a top-level group, or select a group in which you want to create a subgroup. If you do not see a list, verify that you built a group hierarchy tree as described in "Building a Group Hierarchy Tree" on page 8-3.
4. Click **Create Group**.
5. In the Create New Group dialog, enter the group name and description, and click **Create Group** as shown in Figure 8-3. The group name cannot contain special characters, such as \, <>, #, |, &, ~, ?, (), {}, %, or *.

Figure 8-3 Create a New Top-Level Group



Note: If you create a large number of groups, you can press **PgDn** to sort through the list of groups. Pagination improves performance by returning a smaller number of items.

If you add a subgroup to a group, the **Remove From Group** button appears. Clicking **Remove From Group** removes the subgroup's membership from the parent group. The subgroup is not deleted from the system.

Editing a Group Profile

When you create a group, a group profile is automatically created for the group. A group profile contains information about the group that is stored in editable fields in the Administration Console. You can edit properties for groups, but users belonging to those groups do not automatically inherit the group properties you specify. Group properties might include **group manager**, **location**, **department**, and so on.

If a user belongs to two groups, for example, and you have edited each group's properties, you must specify which set of group properties the user should inherit by configuring a `ProfileWrapper` successor at runtime. A `ProfileWrapper` is a lightweight object that can access the correct `ProfileManager` session beans based on the profile identity with which it is initialized.

To edit the values in a group profile in the Administration Console:

1. In the Administration Console, choose **Users, Groups, & Roles > Group Management**.
2. Select an authentication provider from the drop-down list above the Group tree.
3. Select the group you want to edit. If you do not see a list of groups, you might need to build a group hierarchy tree for the user store. If you built a group hierarchy tree and still do not see a list of groups, the user store probably does not allow read access. See the *Security Guide* for instructions on providing access.
4. Select the **Group Profile** tab.
5. From the drop-down list, select the property set whose values you want to edit.
6. Click **Edit** next to the property name to change the property's value, as shown in Figure 8-4.

Figure 8-4 Edit a Property's Value

7. After you enter the changes to the property's value, click **Update**. If the properties belong to a read-only external property database, you cannot modify property values. Even though you cannot edit the properties themselves, you can still use these properties in setting up personalization, delegated administration, and visitor entitlement. For information on personalization, see the *Interaction Management Guide*. For information on delegated administration and visitor entitlement, see the *Security Guide*.

   Tip: If you delete the saved value, a successor value is returned if there is one defined. A successor value specifies the value to use if more than one value exists. If a user belongs to more than one group, you can choose which group properties to inherit. If you do not define a successor, the default value is returned. If there is no defined successor value or a default value, the property value is null.

8. Repeat these steps to edit additional properties.

   If you cannot modify property values because they are read-only, the properties might originate from an external property database set up by your development team. You cannot edit the properties, but you can still use those properties to set up personalization, delegated administration, and visitor entitlement.

You can determine which portal administrators can manage each user group by assigning delegated administration roles to those groups. See the *Security Guide* for instructions on assigning delegated administration.

Tip: If you are using more than one user store, you might have a group in one user store with a name that is the same as a group in another user store. When you set delegated administration on a group, an administrator in that delegated administration role can manage that group in all providers that contain that group (if the administrator also has delegated administration rights to those other providers).

Moving a Group

To change the hierarchy of your groups and subgroups, you can move an existing subgroup from one group to another or change it from a parent group to a child group.

WARNING: Moving a group can affect delegated administration and visitor entitlement roles. For example, if a group called **Managers** is used in a delegated administration role that grants full administrative access to the Administration Console, and you move a group under the **Managers** group, all users in that new subgroup will have full administrative access to the Administration Console. This is a convenient way to grant users delegated administration or visitor entitlement privileges.

If you are using an external user store for users and groups (one that is not the default RDBMS user store built into WebLogic Server), and you want to move a group in that user store, the provider may be configured to prevent moving a group from an outside tool (such as the WebLogic Portal Administration Console). If the **Group Editor** field for the user store is set to `No`, you cannot move a group in that provider with the Administration Console. You must move the group directly in that provider.

To move a group in the group hierarchy:

1. In the Administration Console, choose **Users, Groups, & Roles > Group Management**.
2. Select an authentication provider from the drop-down list above the Group tree.
3. In the Group tree, select the group you want to move. If you do not see a list of groups, verify that you built a group hierarchy tree for the user store. If you built the tree but you still do not see a list of groups, the user store probably does not allow read access. See the *Security Guide* for instructions on giving the provider read access.
4. Right-click the group, and choose **Move**.
5. Click **OK** on the dialog and then right-click the group where you want to place the new group, and choose **Paste**. You can also select **Everyone** to move the group to the top level.

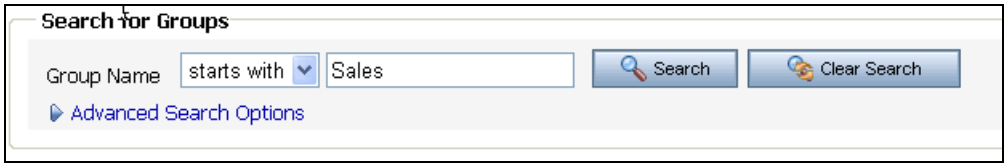
Searching for a Group

If a user store contains thousands of groups, you might get better performance by not building a group hierarchy tree for the provider.

To find a specific group if you did not build a group hierarchy tree:

1. In the Administration Console, choose **Users, Groups, & Roles > Group Management**.
2. Select the top-level group and then enter the name of the group you want to find in the Search for Groups section, as shown in Figure 8-5.

Figure 8-5 Type the Group Name and Click Search to Retrieve It



3. Click **Search**. You can refine your search by selecting `contains` or `ends with`.

When you select a group this way, you can add and edit users and set up delegated administration for the group. However, without a group hierarchy tree, you cannot create, delete, or rearrange groups.

Note: You can disable all group hierarchy trees by adding the following code to the `JAVA_OPTIONS` line in your server startup script:

```
-Dcom.bea.jsptools.disableGroupTree=true
```

Removing a Group

When you delete a group, you permanently remove the group and all of its subgroups from the user store. You must re-create all of the affected groups if you want to use them again. If you delete a group, the users that belong to the group are not deleted.

You can remove a subgroup from a group, and you can move a group within the group hierarchy to change its relationship to other groups. If the group was listed in a delegated administration or visitor entitlement role, you must also remove that group from the role definition.

You can delete a group in the Administration Console or if you use an external user store, you can remove the group there.

This section contains the following topics:

Removing a Group from an Internal User Store

To delete a group:

1. In the Administration Console, choose **Users, Groups, & Roles > Group Management**.
2. Select an authentication provider from the drop-down list above the Resource Tree.
3. Select the group you want to delete. If you do not see a list of groups, verify that you built a group hierarchy tree for the user store. If you built the tree but still do not see the groups, the user store probably does not allow read access. See the *Security Guide* for instructions on making the user store writable.
4. Click **Delete**.
5. If the group was listed in a delegated administration or visitor entitlement role, remove the group from the role definition on the delegated administration or visitor entitlement pages.

Removing a Group from an External User Store

If you are using an external user store to hold users and groups (one that is not the default RDBMS user store built into WebLogic Server), you can delete groups that were created in that provider.

If the external user store you are using does not support group deletion from an outside tool (the **Group Remover** field in WebLogic Server for the provider is set to `No`) you cannot delete a group for that provider with the Administration Console. You must delete the group directly in that provider.

Tip: If you are using an RDBMS user store, group names are case sensitive. For example, the **Managers** group is different than **managers**.

To remove a group from an external user store with the WebLogic Server Administration Console:

1. To verify that your user store supports using an outside tool to remove groups, open the WebLogic Server Administration Console.
2. In the left navigation pane, select **Security Realms**.

3. Select your security realm.
4. Select the **Providers** tab and then the **Authentication** tab.
5. Select your user store.
6. Select the **Provider Specific** tab and review the settings in the **Group Remover** field for the user store. If the **Group Remover** field appears and is set to `Yes`, you can use WebLogic Server or WebLogic Portal to remove groups in the user store. If you need to make the user store writable, follow the instructions in the *Security Guide*.
7. In the WebLogic Server Administration Console, remove the group. Follow the instructions in "Removing a Group from an Internal User Store" on page 8-12.

Note: If you make changes to any user store configuration setting in the WebLogic Server Administration Console, restart the server. Restarting the server prevents exceptions in the WebLogic Portal Administration Console.

CHAPTER 9

Adding and Managing Users

Portal administrators can use the WebLogic Portal Administration Console to add other administrators and portal end-users. Developers might prefer to perform these tasks with JSP tags and controls in Workshop for WebLogic if the portal will have a large number of users. See Chapter 4, "Adding and Updating Users with JSP Tags and Controls" for instructions on adding users with JSP tags and controls. You should set up groups before you add users.

Note: See the *Interaction Management Guide* for instructions on setting up personalization. See the *Security Guide* for instructions on setting up delegated administration and visitor entitlement.

Administrators with full user management rights can use the following tools to create and manage a small number of users:

- **WebLogic Portal Administration Console** – Add and edit portal administrators and small numbers of users here if you work primarily in the Administration Console.
- **WebLogic Server Administration Console** – Add and edit portal administrators and small numbers of users here if you work primarily in the Server Administration Console. See the *BEA WebLogic Server Securing WebLogic Resources* for instructions.
- **External user store** – Add a user to an external user database (such as Netscape's iPlanet) and then configure that repository to be writable.You can access more than one user store to select users and groups. See "Accessing Users in an External User Store" on page 9-4 and for more information.

Developers can use the following tools to create and manage a large number of users:

- **JSP tag** – Add a new user with the `createUser` JSP tag or create a new JSP tag in Workshop for WebLogic. You can also use this tag to create the ability for visitors to your portal to register themselves. Other JSP tags let you let you place users in groups, change passwords, and delete users.
- **Control** – Add a new user with the `createUser` action in the User Provider control in Workshop for WebLogic. You can also use the User Provider control to create the ability for visitors to your portal to register themselves. Other controls let you place users in groups, change passwords, and delete users.
- **Java** – Work directly with the `com.bea.p13n.security.management.authentication.AtnManagerProxy` Java class to add users, change passwords, and delete users. When you use the API to create a user, a user profile is not automatically created. For more information, see the *Javadoc*.

Adding a user with any of these methods (except the Java API) adds the user to the user store and creates a basic user profile that contains the user's identity (name and password). The Java API does not automatically create a user profile when you add a user. You can use other user properties (such as **address**, **phone number**, **e-mail**, and so on) to set up personalization and define rules for delegated administration and visitor entitlement.

This chapter includes the following sections:

Creating Users

This section contains the following topic:

You can add users to WebLogic Portal through internal or external user stores. The default *SQLAuthenticator* authentication provider and RDBMS user store is included when you install WebLogic Server. You can also access other user stores, such as openLDAP, that already contain your users.

If you have a large number of users stored in a user store, you might want to use the WebLogic Scripting Tool to retrieve those users. You can also use Workshop for WebLogic to programmatically get access to the users. Use the Administration Console if you want to add a small number of administrators with special privileges to manage portal content and users.

Tip: You can use WebLogic Portal's internal RDBMS user store for large numbers of users and groups. The internal LDAP is sufficient for storing policies for roles and entitlement.

Adding a User

The WebLogic Portal Administration Console lets you access more than one user store, so you can select users and groups from multiple user stores. The Administration Console contains a list of available user stores. For instructions on adding a new external user store, see the *Security Guide*.

To create a new user:

1. In the Administration Console, choose **Users, Groups, & Roles > User Management**.
2. Select an authentication provider from the drop-down list above the User tree.

   If you are storing users, passwords, and groups in a user store outside of WebLogic Server (such as an OpenLDAP server or Novell NDS), you can connect that provider to WebLogic Server (assuming it is a supported type), and the users in that external provider can log into your portal. In addition to the default RDBMS user store, you can use multiple external user stores in WebLogic Server and WebLogic Portal.

   Tip: WebLogic Portal does not support multiple RDBMS authenticators under a single Security realm.

   ---

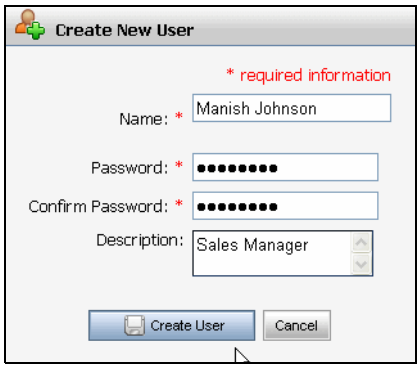
3. In the User tree, select **Everyone.** You can now create a user without assigning the user to a group or you can select a group to which you want to add the user.

   Note: If you do not see a list of groups, verify that you built a group hierarchy tree for the user store. If you built a group hierarchy tree and still do not see a list of groups, the user store probably does not allow read access. You can enable read access to the user store by following the instructions in the *Security Guide*.

4. Click **Create New User**.

5. In the Create New User dialog, enter the new user's **Name**, **Password**, and a **Description**. The name cannot contain special characters, such as \, <>, #, |, &, ~, ?, (), {}, %, or *. The password must be at least eight characters and is encrypted later. See Figure 9-1.

Figure 9-1 Complete the Fields in the Create New User Dialog Box



6. Click **Create User**.

After you create a new user, you can create a user profile to capture more information about the user. See Chapter 10, "Editing User Profile Property Values" for more information.

Accessing Users in an External User Store

If you decide to use multiple user stores (not the default RDBMS user store built into WebLogic Server), most of the effort is setting up and configuring those providers and then connecting WebLogic Server to those providers. You can configure that repository to be writable in the WebLogic Server Administration Console. See the *Security Guide* for instructions on setting up a single user store or multiple user stores.

After your external user stores are connected to WebLogic Server, you can view its existing groups by building a group hierarchy tree in WebLogic Portal. A tree view of groups provides a convenient visual way to change profile values, find users in groups, and add users and groups to rules for delegated administration and visitor entitlement. See "Building a Group Hierarchy Tree" on page 8-3. After you build the group hierarchy tree, you should see the provider's users and groups in WebLogic Portal.

This section contains two topics that explain how to add additional users to an external user store (such as openLDAP):

- Adding a User Directly to an External User Store

Adding a User Directly to an External User Store

See the *Security Guide* for instructions on setting up a user store and connecting it to WebLogic Server. The default configuration for supported external user stores is read-only access to users and groups from the WebLogic Server Administration Console. If the provider does not allow write access, you must add users in the user store itself.

To add a user directly to the external user store:

1. Open the WebLogic Server Administration Console.
2. In the left navigation pane, select **Security Realms**.
3. Click the name of the security realm.
4. Select the **Providers** tab and then select the **Authentication** tab.
5. Select the user store and select the **Provider Specific** tab.
6. Review the settings in the **Create User** field for the user store. If the **Create User** field does not appear or it is set to `No`, you cannot use WebLogic Portal to create a user in this user store. You must create the user or group directly in that provider. (If the User Provider field is set to `Yes`, you can use WebLogic Portal to create a user.)
7. Add the user to the user store itself. You might need to contact your development team to determine the best way to do this.

If your external user store contains additional properties for users and groups (for example, e-mail and phone), accessing those properties involves separate development steps for creating a UUP. See Chapter 6, "Configuring a UUP" for instructions.

Adding a User to an External User Store with an Outside Tool

See the *Security Guide* for instructions on setting up a user store and connecting it to WebLogic Server. The default configuration for supported external user stores is read-only access to users and groups from the Administration Console. If the provider allows write access, you can add additional users to an external user store (such as RDBMS) by adding users in the user store itself.

To use an outside tool to add a user to an external user store:

1. To verify that your user store supports using an outside tool to add users, open the WebLogic Server Administration Console.

2. In the left navigation pane, select **Security Realms**.
3. Click the name of the security realm.
4. Select the **Providers** tab and then select the **Authentication** tab.
5. Select the user store and then select the **Provider Specific** tab.
6. Review the settings in the **User Editor** field for the user store. If the **User Editor** field appears and is set to `Yes`, you can use WebLogic Server or WebLogic Portal to create a user in this user store. If you need to make the user store writable, follow the instructions in the *Security Guide*. (If the **User Editor** field does not appear or is set to `No`, you cannot use WebLogic Portal to create a user.)
7. In the Administration Console, create the new user. Follow the instructions in "Adding a User" on page 9-3.

Do not store identical user names or group names in more than one user store.

If your external user store contains additional properties for users and groups (for example, **e-mail** and **phone**), accessing those properties involves separate development steps for creating a UUP. See Chapter 6, "Configuring a UUP" for instructions.

Tip: If you make changes to any user store configuration setting in the WebLogic Server Administration Console, restart the server. Restarting the server prevents exceptions in the WebLogic Portal Administration Console.

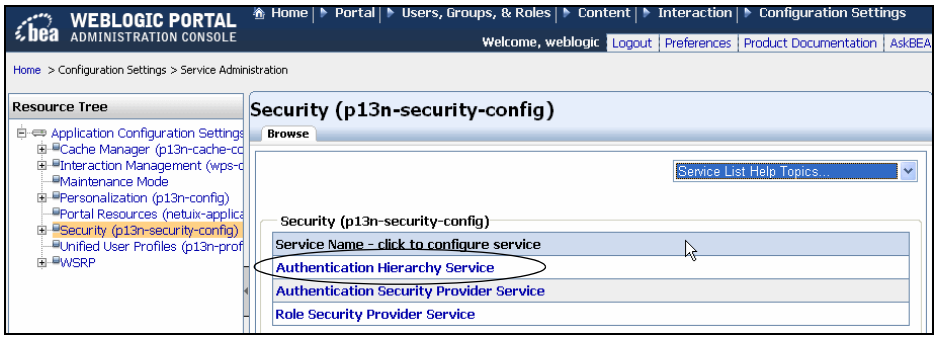
Removing a User Store

If you remove a user store in the WebLogic Server Administration Console, you must also remove the provider from the WebLogic Portal Administration Console.

To remove a user store from the Administration Console:

1. In the Administration Console, choose **Configuration & Monitoring > Service Administration**.
2. In the Resource Tree, select **Security**.
3. In the Browse tab, click **Authentication Hierarchy Service**, as shown in Figure 9-2.

Figure 9-2 Click Authentication Hierarchy Service to Change Its Settings



4. Click **Configuration Settings for: Authentication Hierarchy Service**.
5. In the **Authentication Providers to Build** field, select the check box next to name of the provider you want to remove and click **Remove Selected**.
6. Click **Update**.

Placing Users in Groups

You can add a user to one or more groups. If your user store does not allow write access to users and groups, you will not be able to add users to groups with the Administration Console. You must add users to groups in the user store directly. See Chapter 2, "Planning a User and Group Strategy" for more information on planning users and groups.

This section contains the following topics:

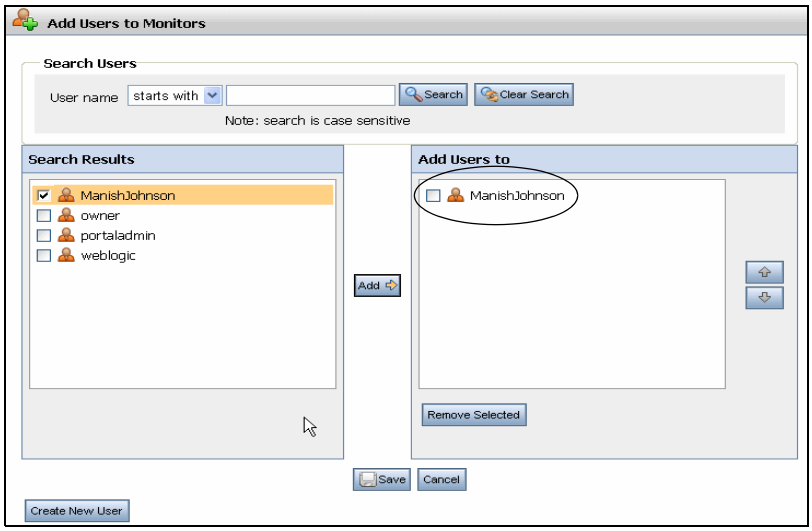
Adding a User to a Group

To add a user to a group:

1. In the Administration Console, choose **Users, Groups, & Roles > Group Management**.

2. Select an authentication provider from the drop-down list above the Group tree. The provider should contain the users you want to add.
3. Select the group to which you want to add the user.
4. Select the **Users In Group** tab.
5. Click **Add Users to Group**.
6. Select the check box next to the user you want to add, and click **Add**. The user you selected now appears in the **Add Users To** list, as shown in Figure 9-3.

Figure 9-3 Select the User You Want to Add to the Group and Click Add



7. Click **Save**. After you add the user to the group, the user will inherit any delegated administration or visitor entitlement rights that the group already has.

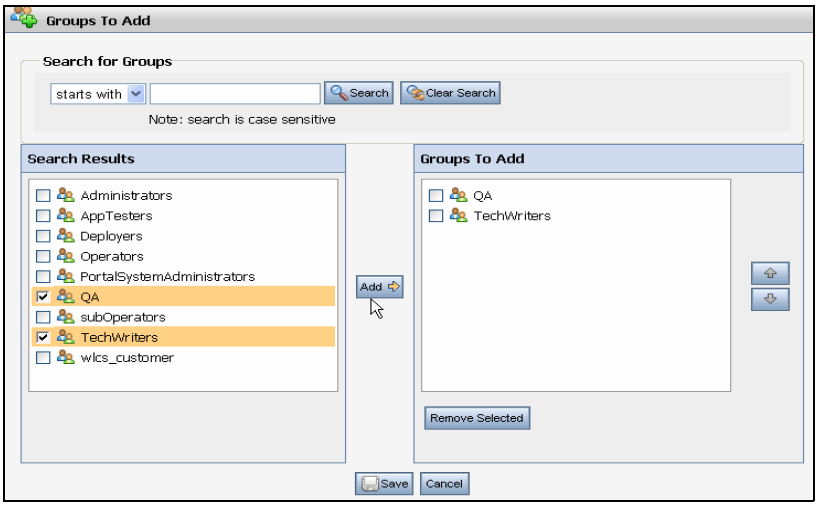
Adding a User to Multiple Groups

To add a user to more than one group:

1. In the Administration Console, choose **Users, Groups, & Roles > User Management**.
2. Select an authentication provider from the drop-down list above the User tree. The provider should contain the user you want to add.
3. Select the user you want to add (see "Finding a Single User" on page 9-12 for instructions).

4. Select the **Group Membership** tab. The groups to which the user belongs are listed.
5. Click **Add Groups**.
6. In the Search Results list, select the check box next to each group to which this user should belong and click **Add**. The groups you chose appear in the **Groups To Add** list. See Figure 9-4.

Figure 9-4 Add a User to More Than One Group



   To remove a group from the Group to Add list, select the check box next to the group and click **Remove Selected**.

7. Click **Save**.

Viewing a User's Group Membership

A user can belong to more than one group. If you are using an RDBMS user store, be aware of case sensitivity when looking up users and groups. For example, *Bob* is different than *bob*.

To see a list of groups to which a user belongs:

1. In the Administration Console, choose **Users, Groups, & Roles > User Management**.
2. Select an authentication provider from the drop-down list above the User tree. The authentication provider's user store should contain the user.
3. Find the user you want to view (see "Finding a Single User" on page 9-12 for instructions).

4. Select the user's name. The groups to which the user belongs are listed.

Note: If a list of groups is not displayed, verify that you built a group hierarchy tree for the user store. If you still do not see a list of groups, the user store probably does not allow read access. See the *Security Guide* for instructions.

Deleting a User From a Group

A group does not own a user, so you can add and remove users from groups without affecting the user's properties. Removing a user from a group removes the user from any delegated administration or visitor entitlement roles based on that group. For example, if you remove a user from the Administrators group, that user might no longer have full administrative access to the Administration Console.

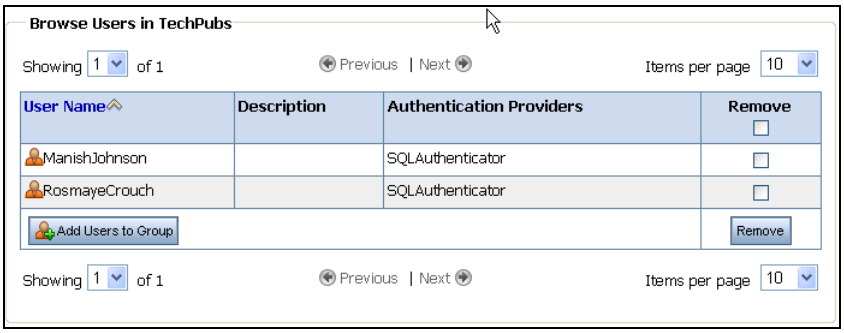
Removing a user from a group does not delete the user from the system or change the user's profile properties. You can remove multiple users from one group, or remove a single user from multiple groups.

Removing Multiple Users From a Single Group

To remove multiple users from a single group:

1. In the Administration Console, choose **Users, Groups, & Roles > Group Management**.
2. Select an authentication provider from the drop-down list above the Group tree. The provider's user store should contain the users you want to manage.
3. Select the group that contains the users you want to remove.
4. Select the **Users In Group** tab.
5. Select the check box next to each user you want to remove from the group, as shown in Figure 9-5. (If you do not see the user listed, type the user's name and click **Search**.)

Figure 9-5 Remove a User From a Group



6. Click **Remove**.

Removing a Single User From Multiple Groups

To remove a single user from multiple groups:

1. In the Administration Console, choose **Users, Groups, & Roles > User Management**.
2. Select an authentication provider from the drop-down list above the User tree. The provider's user store should contain the user you want to manage.
3. Find the user you want to remove from the groups and select that user. See "Finding a Single User" on page 9-12 for instructions. Groups to which the user belongs are listed.
4. Select the **Group Membership** tab.
5. Select the check box for each group to which the user should not belong.
6. Click **Remove** to remove the user from these groups.

Managing Users

You can search for a user or update a user's password in the Administration Console.

This section contains the following topics:

Searching for a User

The Administration Console provides a way for you to locate users that are not already members of a selected group. If you need to perform administrative tasks, such as editing user profiles, removing users from a group, or deleting users from the system, you must first locate those users in the system.

The delegated administration and visitor entitlement features also provide tools for user lookup when adding users to roles.

WebLogic Portal support two ways to locate users by username:

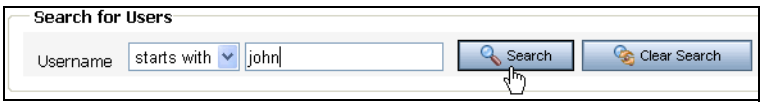
- **WebLogic Portal Administration Console** – If you need to perform administrative tasks, such as editing user profiles, removing users from a group, or deleting users from the system, you must first find those users in the system.
- **BEA Workshop for WebLogic Platform** – Programmatically locate users with JSP tags and controls. See "Searching for Users" on page 4-7 for more information.

Finding a Single User

To search for a user by username:

1. In the Administration Console, choose **Users, Groups, & Roles > User Management**.
2. Select an authentication provider from the drop-down list above the User tree. The provider's user store should contain the user you want to locate. If you know the group that contains the user, select the group. (If you do not see a list of groups, see the Note in Finding Multiple Users.)
3. Select the **Everyone** group.
4. Enter the username and click **Search**, as shown in Figure 9-6.

Figure 9-6 Search for All Usernames that Begin with John



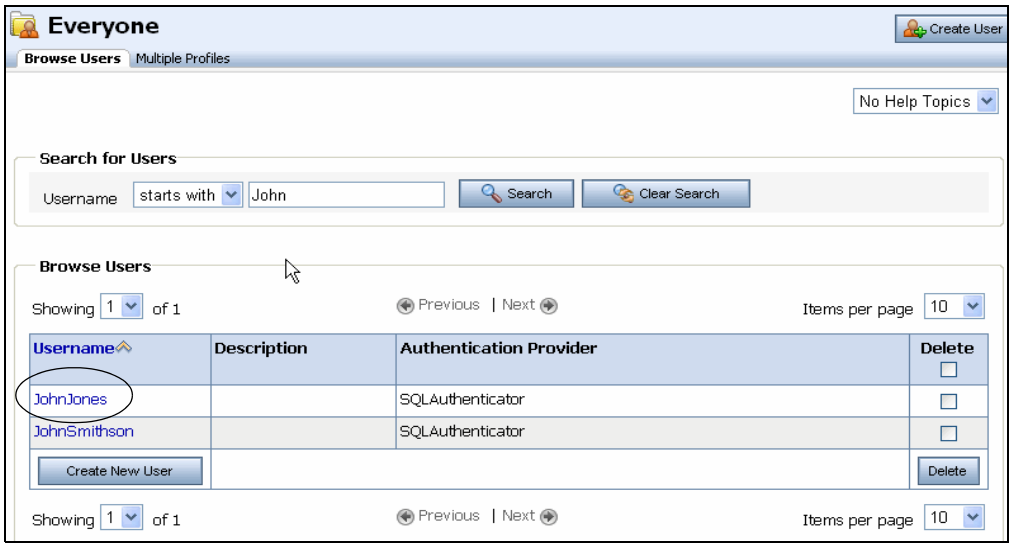
5. The user appears in the **Browse Users** section. If you want to edit the user's properties or user profile, click the user's name.

Finding Multiple Users

To search for users by username:

1. In the Administration Console, choose **Users, Groups, & Roles > User Management**.
2. Select an authentication provider from the drop-down list above the User tree. The provider's user store should contain the users you want to locate. If you know the group that contains the users, select the group.
3. If a list of groups does not appear, verify that you built a group hierarchy tree for the user store. If you still do not see a list of groups, the user store probably does not allow read access.
4. Enter the username and click **Search**.
5. The user appears in the **Browse Users** section. Click the user's name, as shown in Figure 9-7.

Figure 9-7 Click the User's Name to Edit User Profile Properties



Tip: If you are using an RDBMS user store, be aware of case sensitivity when looking up users and groups. For example, *Bob* is different than *bob*.

Changing a User's Password

You might need to reset a password if a user lost or cannot remember a password. If you have the appropriate delegated administration rights, you can change any user's password. See the *Security Guide* for instructions on setting up delegated administration.

Ensure that your user knows the new password, because once the password is changed there is no way to find out what it is. If a user forgets a password, a portal administrator must change it again.

To change a user's password:

1. In the Administration Console, choose **Users, Groups, & Roles > User Management**.
2. Select an authentication provider from the drop-down list above the User tree. The provider's user store should contain the user whose password you want to change.
3. Find the user whose password you want to change. (See "Finding a Single User" on page 9-12 for instructions.)
4. Select the user's name.
5. Click **Change Password**.
6. Enter the new password in the **Password** and the **Confirm Password** fields, and click **Update**.

Removing Users

When you delete a user, you remove the user from the user store. The deleted user is no longer available in any other group or subgroup, and the user will not be able to log into your portal. To get the user back in the system, you must create the user again.

If you want to remove the user from a group without removing the user from the entire system, see "Deleting a User From a Group" on page 9-10.

If you are using an external user store to store users and groups (one that is not the default RDBMS user store built into WebLogic Server), and you want to remove a user from that provider, the provider might be configured to prevent user removal from an outside tool, such as the Administration Console. See the *Security Guide* for instructions.

If the **User Remover** field for the user store is set to `No`, you cannot remove users from that provider with the Administration Console. You must remove users directly from that provider.

To delete a user from WebLogic Portal:

1. In the Administration Console, choose **Users, Groups, & Roles > User Management**.
2. Select an authentication provider from the drop-down list above the User tree. The provider's user store should contain the users you want to remove.
3. In the User tree, select the user you want to delete. (See "Finding a Single User" on page 9-12 for instructions.)
4. Click **Delete**.

Note: You can also delete a user by selecting **Everyone** in the User tree, selecting the check box next to the user's name in the Browse Users tab, and clicking **Delete**.)

If the user is explicitly listed in a delegated administration or visitor entitlement role, remove that user from the role definition on the **Delegated Administration** or **Visitor Entitlement** pages. See the *Security Guide* for more information.

CHAPTER 10

Editing User Profile Property Values

This chapter provides instructions for portal administrators to edit user profile property values in the Administration Console. Developers can use Workshop for WebLogic to programmatically use JSP tags and controls that create and edit user profiles and default values.

A user profile consists of a username and password, as well as any additional attributes you collect and store about a user. Properties can consist of personal data, work-related data, geographic data, or something else that logically categorizes your users.

For example, you could create a property set in Workshop for WebLogic called **human resources** that contains properties such as **gender**, **hire date**, and **email address**. This information can be used to personalize the user's experience in your portal. When users log into a portal, the portal can access the property values and target them with personalized content, e-mails, pre-populated forms, and discounts based on the personalization rules you set up. See the *Interaction Management Guide* for more information on personalization.

Developers use the following tools to programmatically create user profiles and set default property values:

- **JSP tags** – Use the `createProfile` JSP tag in Workshop for WebLogic to create a JSP tag that adds a user profile. Other JSP tags let you retrieve the user profile, and add or edit its properties. See "Editing Properties and Values with JSP Tags" on page 5-7 for more information.
- **Controls** – Use the `createProfile` action in the Profile control in Workshop for WebLogic to add a user profile. Other actions in the User Provider control retrieve the user profile, and add or edit its properties. See "Editing Properties and Values with Controls" on page 5-10 for more information.

- **Java** – Work directly with the `com.bea.p13n.usermgmt.profile.ProfileFactory` Java class to add a user profile. You can change user properties by calling the `ProfileWrapper` object directly. For more information, see the *Javadoc*.

After you use JSP tag and controls to create user profiles and set default property values, you can edit the values in Workshop for WebLogic or in the Administration Console.

The chapter includes the following section:

Editing Property Values

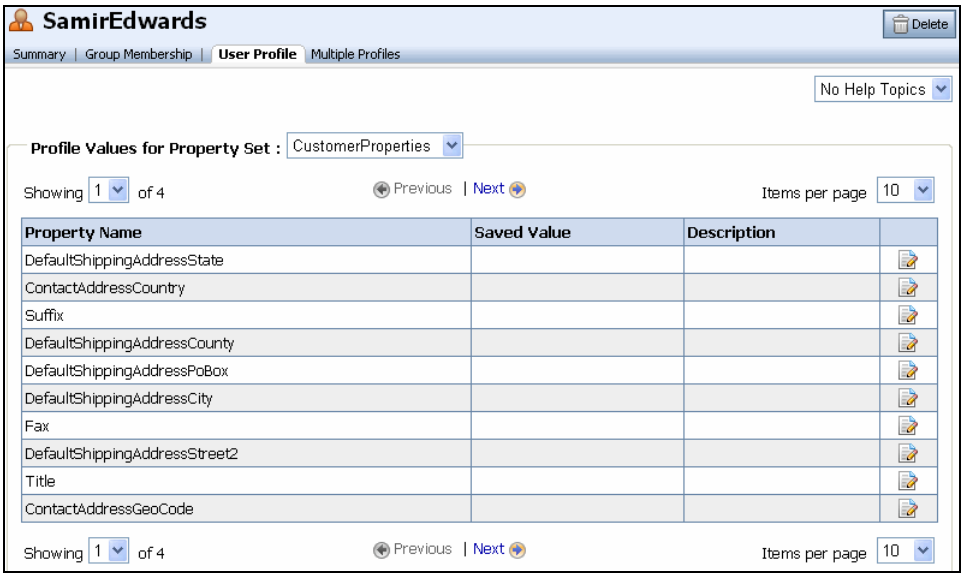
This section describes how to use the Administration Console to edit the property values that are part of each user's profile. You must use the user profile editor in Workshop for WebLogic to create user profile properties and set default values.

User profile properties appear as input fields in the Administration Console when you edit a user profile's values. WebLogic Portal provides a default user profile property set called `CustomerProperties.usr` that contains many common properties.

To modify user profile values in the Administration Console:

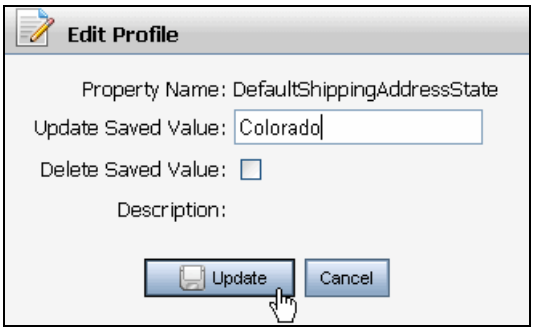
1. In the Administration Console, choose **Users, Groups, & Roles > User Management**.
2. Select an authentication provider from the drop-down list above the User tree.
3. Find the user whose profile you wish to edit. (If you do not see a list of groups, see the Notes below.) See "Finding a Single User" on page 9-12 for instructions.
4. Select the **User Profile** tab.
5. In the **Profile Values for Property Set** field, use the drop-down list to select the property set containing the properties you want to edit.
6. Locate the property name and click **Edit** to change the value, as shown in Figure 10-1.

Figure 10-1 Edit the User Profile Values in the Administration Console



7. Enter a new value in the **Update Saved Value** field and click **Update**, as shown in Figure 10-2. You can delete the value that is currently saved by selecting the **Delete Saved Value** check box. If you delete the saved value, a successor value is returned if one is defined. If a successor was not defined, the default value is returned. If there is no defined successor value or a default value, the property value is `null`.

Figure 10-2 Enter the New Property Value



8. Click **Update**. The Default Shipping Address for the state is now **Colorado**, as shown in Figure 10-3.

Figure 10-3 View Your Changes to the Property Value

Part IV Production

The Production phase is the time that you make small changes to your Production environment. User management tasks might include making small changes to your portal application, such as adding an administrative user, adding a small number of portal users, adding a small number of groups, updating user profiles, and other maintenance tasks.

Some User Management tasks you can do in the Production phase might include the following:

- **Adding a new administrative user** – See "Creating Users" on page 9-2 for instructions
- **Creating a new group to reflect an organizational change** – See "Creating a Group or Subgroup" on page 8-6
- **Updating a User Profile property set to collect different data about users** – See "Adding Properties to a Property Set" on page 5-4
- **Removing old data from Anonymous User Tracking** – See "Setting Up Anonymous User Tracking" on page 7-3

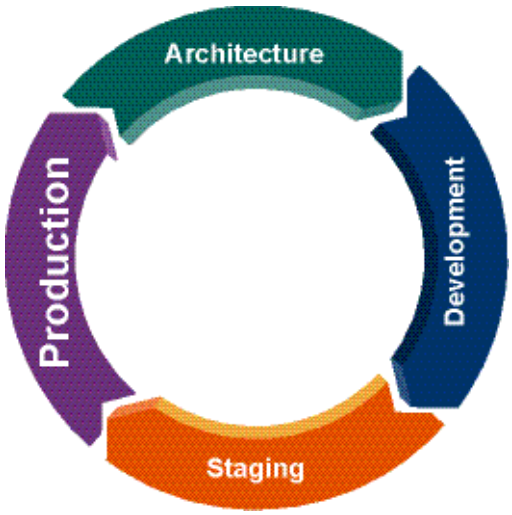
If you add users or groups to your portal application in the Production phase, you must return to the Staging phase to test the functionality you created. This Staging environment simulates a Production environment.

The Production phase can be ongoing; some changes to your portal's functionality might even require you to return to the Development phase and redeploy your portal application. Then you can move to the Staging phase to test your changes and go back to the Production phase.

In the Production phase, you might also modify other features, such as delegated administration and visitor entitlement roles and assignments, interaction management settings and queries, and

desktops. See the *Security Guide* for more information on delegated administration and visitor entitlement. See the *Interaction Management Guide* for instructions on setting up interaction management.

For a description of the Production phase of the portal life cycle, see the *WebLogic Portal Overview*. The portal life cycle is shown in the following graphic:



APPENDIX A

Creating a UUP EJB

WebLogic Portal includes a UUP service that lets you add and manage user properties in a single logical location—even if the user data is stored in external systems, such as an LDAP server. You can add this additional user data to a user's profile.

You can use Workshop for WebLogic to create a UUP EJB, and then use Workshop for WebLogic or the Administration Console to configure the UUP. (In WebLogic Portal 8.1, you could configure a UUP manually, but that method is no longer supported). You must choose one method or the other; the methods are not interchangeable.

The method you choose depends where you are in your portal development process:

- **During the Portal Application Development Phase** – If you are in the portal development phase, use Workshop for WebLogic to create and configure the UUP. The UUP configurations are stored in the META-INF/p13n-profile-config.xml descriptor file and are packaged within the portal application. (A UUP you configure in the Administration Console overwrites Workshop for WebLogic UUP configurations from the `p13n-profile-config.xml` file and stores the settings separately from the portal application.)
- **During the Portal Runtime Phase** – Use the Administration Console to configure the UUP at portal runtime (after the portal is deployed) and overwrite the configuration performed by Workshop for WebLogic. The Administration Console stores application descriptor changes (the `p13n-profile-config.xml` file is one of application descriptors) in the deployment plan, separate from the portal application. See the WebLogic Server documentation for more information about the relationship between the descriptor and the deployment plan.

The Administration Console overwrites UUP configurations from the `p13n-profile-config.xml` file, and stores changes in the deployment plan, separate from the portal application.

WARNING: Configuring a UUP in Workshop for WebLogic or Administration Console takes precedence over existing UUPs that you configured manually in previous versions of WebLogic Portal. Existing UUPs that you created manually can coexist with new UUPs that you configure in the Administration Console or in Workshop for WebLogic.

The chapter includes the following sections:

Creating and Configuring an EntityPropertyManager EJB

You can use Workshop for WebLogic to create and configure the EntityPropertyManager EJB. The next step is to create a UUP in Workshop for WebLogic, and then configure the UUP in Workshop for WebLogic or the Administration Console.

This section contains the following topics:

Creating the EJB

You can use Workshop for WebLogic or another development tool to create a stateless session bean EJB that implements the methods of the EntityPropertyManager interface.

To incorporate data from an external source, you must 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 creates and deletes profile records. By default, `EntityPropertyManager` provides read-only access to external properties.

Tip: To learn how to create a stateless session bean EJB component that is scoped to the application, see the instructions in Workshop for WebLogic. To find this topic, choose **Help > Help Contents**, expand **BEA Workshop for WebLogic Platform Programmer's Guide**, and select **Enterprise Java Beans**.

The stateless session bean should also 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`
- `MyEntityPropertyManagerImpl` extends `javax.ejb.SessionBean`

Using Recommended EJB Guidelines

Use the following guidelines to create your new EJB:

- Your custom `EntityPropertyManager` is not a default `EntityPropertyManager`. A default `EntityPropertyManager` is used to get, set, or remove properties in the portal schema. Your custom `EntityPropertyManager` can throw a `java.lang.UsupportedOperationException` message if it does not support the following methods:
  - The `getDynamicProperties()` method
  - The `getEntityNames()` method
  - The `getHomeName()` method
  - The `getPropertyLocator()` method
  - The `getUniqueId()` method
- If you want to use the portal framework and tools to create and remove users in your external data store, you must support the `createUniqueId()` and `removeEntity()` methods. However, your custom `EntityPropertyManager` is not the default

`EntityPropertyManage`, so your `createUniqueId()` method does not have to return a unique number. The method 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:
  - The `getProperty()` method – 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 the `getProperties()` method.
  - The `setProperty()` method – Use caching.
  - The `removeProperties()` and `removeProperty()` methods – After these methods are called, a call to `getProperty()` should return *null* for the property. You should also remove properties from the cache.
- 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 Javadoc.)
- If the external system contains read-only data, any methods that modify profile data must throw a `java.lang.UnsupportedOperationException` message. Additionally, if the external data source contains users that are created and deleted by something other than your WebLogic Portal `createUniqueId()` and `removeEntity()` methods, you can throw an `UnsupportedOperationException` message.
- To avoid class loader dependency issues, verify 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).

Follow the steps in the next section to configure and deploy your EJB.

Configuring the EJB

You can use Workshop for WebLogic or the Administration Console to configure the new `EntityPropertyManager` EJB.

To configure and deploy the new `EntityPropertyManager` EJB:

1. If you have already deployed the application on a WebLogic Portal instance, stop the server.
2. In the Merged Projects View in Workshop for WebLogic, locate the `p13n-profile-config.xml` file.
3. Right-click the `p13n-profile-config.xml` file and copy it into the `/META-INF` directory in your EAR project folder. Uncomment or edit the applicable lines in the `p13n-profile-config.xml` file and save the file.
4. Deploy the `EntityPropertyManager` EJB.
5. You can also access the Administration Console and add or change lines in the `p13n-profile-config.xml` file. Choose **Configuration & Monitoring > Service Administration**, and select **Unified User Profiles** and your UUP in the Resource Tree. After you make changes to the `p13n-profile-config.xml` file in the Administration Console, redeploy the application.

Retrieving User Profile Data from an LDAP Server

WebLogic Portal ships with a default UUP that retrieves properties from an LDAP server. An LDAP can be any directory server that supports the LDAP protocol, such as Sun Directory Server, Microsoft Active Directory, or OpenLDAP.

WebLogic Portal supports transparent failover, to move from one LDAP server to another LDAP server that contains identical information. You can specify single LDAP PropertyManagers to point to LDAP servers that contain the same user and group information. The order of LDAP servers specified in the configuration determines the order of failover from the previous LDAP server. If the primary server is unavailable, the property retrieval request rolls to the secondary server, and so on. If you do not set up transparent failover and an LDAP server fails, the transaction fails.

See Table A-1 for configuration instructions on transparent failover.

Note: For instructions on connecting to other user stores with UUP, see the *Security Guide*.

The `LdapRealm` Security realm and the `LdapPropertyManager` UUP that retrieve user properties from LDAP are independent of each other. They do not share configuration information, and there is no requirement to use one with the other. A Security realm has nothing to do with a user profile. A Security realm provides user and password data, and user and group associations. A user profile provides user and group properties. A password is not a property.

Configuring an LDAP UUP and Transparent Failover

To implement the LDAP UUP to retrieve properties from your LDAP server or to set up transparent failover:

1. If you have already deployed the application on a WebLogic Portal instance, stop the server.
2. In the Merged Project View in Workshop for WebLogic, deploy the sample `ldap_uup.jar` file as an EJB component of your portal application. To deploy the file, locate the `ldap_uup.jar` file in the <WL-HOME>/wlserver_10.0/common/p13n/lib directory and copy it to your EARContent directory in your EAR Project folder.
3. Locate the `application.xml` file in the `/META-INF` directory in your `EARContent` directory and add the following information to the `<module>` section:

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

   Save the `application.xml` file.
4. Copy the `p13n-profile-config.xml` file from the `p13n-app-lib.ear` file into the `/META-INF` directory of your EAR project, and uncomment the `<property-adapter>` section named `LdapUUPAdapter`.

   Tip: If you already have an existing `p13n-profile-config.xml` file, open it and copy the updated sections into the `p13n-profile-config.xml` file in your `/META-INF` directory in your Portal EAR directory.
5. In your Portal EAR directory, open your `/META-INF/p13n-profile-config.xml` file. (If you have not already copied this file from the library module, copy it now.) The file contains a commented block called `LdapUUPAdapter`. Uncomment and reconfigure this section using the following steps:

a. In the `Ldap Property Manager` block, locate the default settings for your multiple `<adapter-property>` section shown in Table A-1 and replace the default values with your own.

Table A-1 Replace the Default Settings

Default Setting	Adapter Property Name	New Setting
• `ldap://server.company.com:389` • `SunOneLdapServerHost:1234 199.234.1.2 ads.mycompany.com:5678`	• `serverURL` • `serverHosts` If both properties exist, the serverHosts takes over.	• Your LDAP server URL • A space separated LDAP server host:port string list
`uid=admin, ou=Administrators, ou=TopologyManagement, o=NetscapeRoot`	`UserPrincipal`	Your LDAP server's principal
Set in the Administration Console. There is no default.	N/A	The `credentialAlias` that can log into your LDAP server
`ou=People,o=company.com`	`userDN`	Your LDAP server's `UserDN`
`ou=Groups,o=company.com`	`groupDN`	Your LDAP server's `GroupDN`
`uid`	`uid`	Your LDAP server's `usernameAttribute` setting in place of `uid`
`cn=bea.com`	`cn`	Your LDAP server's `groupnameAttribute` setting in place of `cn`
30	`connectTimeout`	Connection timeout (in seconds) to failover to the next configured LDAP server. This setting is optional.

b. If you are configuring transparent failover, use the `serverHosts` instead of the `serverURL` adapter property to configure multiple LDAP servers. The `serverURL` adapter property supports only single LDAP server configuration for backwards compatibility.

c. Save and close the file.

6. Create an LDAP cache. The name of the cache is specified by the adapter parameter `ldapPropertyCacheName` in the `p13n-profile-config.xml` file. The default value for the cache's name is `ldapPropertyCache`. To learn how to create a cache in the Administration Console, see *Cache Reference Guide*.
7. If you want the properties from your LDAP server to appear in the Administration Console (so you can define rules for personalization, delegated administration, and visitor entitlement), create a user profile property set called `newldap.usr`, and create properties in the property set that exactly match the `<property-mapping>` names of the LDAP properties you want to appear. Place the newldap.usr file in your datasync project directory. The datasync project must reside in your portal EAR project.
8. Start the server and redeploy the application. The properties from your LDAP server are now accessible through the WebLogic Portal API, JSP Tags, and Controls.
9. If your LDAP server requires a username and password during connection, set them up by launching the Administration Console and choosing **Configuration & Monitoring > Service Administration.** Select Unified User Profile in the Resource tree and then select your LDAP UUP. Enter the username and password that are mapped to the credential alias adapter property. Re-deploy the application again. Verify the change by logging in as an existing LDAP user.

   To verify this step, go to the Administration Console and choose **Users, Groups, & Roles > User Management** to see if a particular user's LDAP properties are shown in the user's profile.

Tip: A credential alias requires a username and password. If you do not want your LDAP server to require a username and password, remove the `credential-alias` line in the `p13n-profile-config.xml` file.

Enabling Searches for a User and Group

If your users are located in separate branches with a common root in your LDAP server, you must enable the subtree scope feature. The `LdapPropertyManager` EJB in the `ldap_uup.jar` file allows the LDAP schema to be inspected to determine multi-valued versus single-value (or flat) LDAP attributes, allow for multiple `userDN` and `groupDN`, and allow `SUBTREE_SCOPE` searches for users and groups in the LDAP server.

Determining multi-value versus single-value LDAP attributes specifies that the LDAP schema should be used to determine if a property is single- or multi-value. Consult your LDAP Server schema for instructions.

In your portal, you can configure these optional settings in the `<adapter-property>` setting to specify the LDAP schema using one of the following methods:

- **Administration Console** – During portal runtime, you can configure these optional settings according to the instructions in "Configuring a UUP in the Administration Console" on page 6-4.
- **Workshop for WebLogic** – During the development phase, you can configure these optional settings according to the instructions in "Configuring a UUP in Workshop for WebLogic" on page 6-10.

The `p13n-profile-config.xml` file provides sample settings as a commented block.

This feature also implements changes that allow you to use `SUBTREE_SCOPE` searches for users and groups. It also allows you to specify multiple base `userDN` and `groupDN`. You can use the multiple base DN 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 user name or group name.

Do not use `true` for `detectSingleValueFromSchema` in the `p13-profile-config.xml` file unless you plan to write rules that use multi-valued LDAP attributes that have a single value. Using `/detectSingleValueFromSchema = true` adds the overhead of checking the LDAP schema for each attribute instead of the default behavior (`/detectSingleValueFromSchema = false`), which only stores an attribute as multi-valued (in a collection) if it has more than one value.

Perform the following steps to manually enable `SUBTREE-SCOPE` for users and groups:

1. Stop the server.
2. Set the Boolean `objectPropertySubtreeScope` adapter-property-value to `true` in the adapter-property-name element in the `p13n-profile-config.xml` file for the LDAP UUP adapter.
3. Set the `userDN` and `groupDN` adapter-property-values in the `p13n-profile-config.xml` file to be equal to the base DNs where you want your `SUBTREE_SCOPE` searches to begin.

   For example, if you have users in `ou=PeopleA`, `ou=People`, `dc=mycompany`, `dc=com`, and `ou=PeopleB`, `ou=People`, `dc=mycompany`, and `dc=com`, you could set `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 `ou=People` and work its way down the branches (`ou="PeopleA"` and `ou="PeopleB"`).

Do 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 `PeopleA` and your `PeopleB` branches. The `LdapPropertyManager` in the `ldap_uup.jar.jar` file returns property values for the first `userA` that it finds.

Note: Do not enable this change (by setting `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) is to configure multiple base DNs and leave `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 cycles from one base DN to the next until it finds the first matching user or group.

The new `p13n-profile-config.xml` file is fully commented to explain how to set multiple DNs, multiple `usernameAttributes` (or `groupnameAttributes`), and how to set the `objectPropertySubtreeScope` flag.

4. Save and close the file.
5. Start the server and redeploy the application.

Upgrading a UUP

When you upgrade a UUP from WebLogic Portal 8.1, the `p13n_ejb.jar` file is deleted and replaced with a new WebLogic Portal version of this file. The new `p13n_ejb.jar` file is packaged in the library modules that ship with WebLogic Portal 10.0.

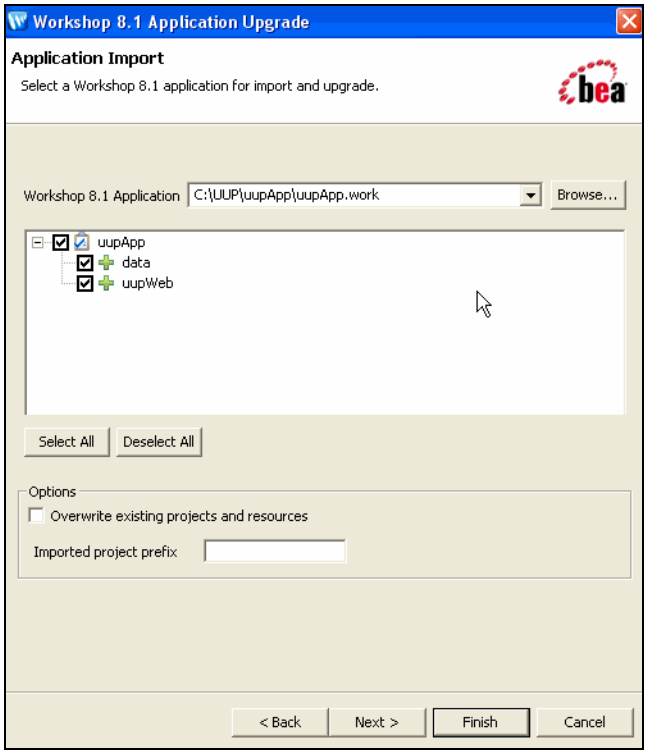
Tip: If you modified settings in your `p13n_ejb.jar` file and you want to preserve them, you must use Workshop for WebLogic to create a `p13n-profile-config.xml` file. Add your existing settings to that XML file and deploy it.

To upgrade a UUP created in WebLogic Portal 8.1 to WebLogic Portal 10.0:

1. Start Workshop for WebLogic and create a new Workspace.
2. Create a new portal domain. Do not create a Portal EAR Project. For instructions on creating a new domain, see the *Portal Development Guide*.
3. Import your Portal 8.1 UUP application into your new environment by choosing **File > Import**.

4. In the Select dialog, click the **Open** folder and select **Workshop 8.1 Application** and click **Next**.
5. In the Application Import dialog, click **Browse** and locate your 8.1 UUP application. Select the `.work` file and click **Open**. Verify that the check boxes for the UUP application are selected and click **Next**, as shown in Figure A-1.

Figure A-1 Locate the 8.1 UUP Application



6. In the Source Upgrade dialog, click **NetUI Project Upgrader** options and select the **Use WebLogic J2EE Shared Libraries** check box. You can also click **JSP File Migrator options** and select the **Replace BEA NetUI tags with Apache Beehive tags** check box (if desired) and click **Finish**.
7. After the upgrade finishes, verify that the following actions occurred:
   – The `p13n-ejb.jar` file was removed from the EARContent directory of the UUP application.

- The UUP EJB JAR file (for example, `UUPExample.jar`) exists in the EARContent directory of the UUP application.
- The UUP EJB JAR file is referenced in a module entry in the `application.xml` file in the <UUPApplication>/EARContent/META-INF/ directory.
- As an example, the cache entry below was added to the p13n-cache-config.xml file in the <UUPApplication>/EARContent/META-INF/ directory:

```
<p13n:cache>
     <p13n:name>UUPExampleCache</p13n:name>
     <p13n:description>Cache for UUP Example</p13n:description>
     <p13n:time-to-live>60000</p13n:time-to-live>
     <p13n:max-entries>100</p13n:max-entries>
</p13n:cache>
```

- Verify that the User Profile file (for example, `UUPExample.usr`) file exists in the data/src/userprofiles/ directory (or where your Datasync folder exists).

8. Associate your UUP application with your WebLogic Server by selecting the server in the Servers tab, right-clicking the server, and choosing **Add and Remove Projects**. Select the UUP application from the **Available Projects** section, click **Add**, and then click **Finish**.
9. Build and publish your application. Verify the application by starting the WebLogic Server Administration Console and clicking **Deployments**. Verify that the UUP application is active. Then open the UUP application by expanding the tree and verifying that the UUP JAR file appears as an EJB.

For more information about upgrading other non-portal applications from WebLogic Portal 8.1, see the *Upgrade Guide*.