BEA WebLogic Workshop™ Help

Version 8.1 SP4
December 2004

Copyright

Table of Contents

Portal Overview

The following topics provide an overview of WebLogic Portal and show you how to get started building portal applications:

Portal Overview

This topic on the BEA E–docs site provides an overview of portals and describes the power, flexibility, and ease of use of WebLogic Portal in portal development and administration.

What is a Portal?

Provides more detail about the portal components you will develop using the WebLogic Workshop Portal Extensions and the WebLogic Portal services you will use to build powerful, feature–rich applications.

Portal Key Concepts and Architecture

Provides a graphic that illustrates the flexibility and extensibility of the WebLogic Portal architecture.

Getting Started

Provides two different paths for getting started: exploring WebLogic Portal with tutorials and samples or beginning portal development.

What is a Portal?

A portal represents a Web site that provides a single point of access to applications and information and may be one of many hosted within a single WebLogic Portal server.

Portals are becoming more and more important to companies, who have an ever–increasing need to provide employees, partners, and customers with an integrated view of applications, information, and business processes. WebLogic Portal meets these needs, allowing companies to build portals that combine functionality and resources into a single interface while enforcing business policies, processes, and security requirements, and providing personalized views of information to end users.

From an end user perspective, a portal is a Web site with pages that are organized by tabs or some other form of navigation. Each page contains a nesting of sub–pages, or one or more portlets individual windows that display anything from static HTML content to complex Web services. A page can contain multiple portlets, giving users access to different information and tools in a single place. Users can also customize their view of a portal by adding their own pages, adding the portlets they want to it, and changing the look and feel of the interface.

The business problem that portals solve is illustrated in the following example. A company has the need for several types of Web presence: an Intranet for its employees, a secure site for interactions with partners, and a public Web site. WebLogic Portal s flexible portal network architecture supports multiple implementation choices which allow re–use of resources across portals.

Portals in Java 2 Enterprise Edition (J2EE) Terms

From a J2EE standpoint, WebLogic Portal is an enterprise application consisting of Enterprise Java Bean (EJB) components and a set of Web applications. The enterprise application contains the core application programming interfaces (API), and the Web applications contain servlets, Java Server Pages (JSPs), JSP tag libraries, and supporting Java classes.

The WebLogic Portal architecture allows multiple portals per Portal Web application, letting you flexibly build and leverage Web application resources and security in multiple portal deployments.

Technically speaking, a portal is a container of resources and functionality that can be made available to end–users. These portal views, which are called Desktops in WebLogic Portal, provide the uniform resource location (URL) that users access.

Components that Make up a Portal

In WebLogic Portal, a portal definition is a single XML file. The XML file is created automatically as you build a portal with the Portal Designer that is provided as part of the WebLogic Workshop Portal Extensions. The portal file contains all the components that make up that particular instance of the portal, such as books, pages, portlets, and look and feel components.

Many components have a hierarchical relationship to each other. For example, a book contains pages and pages contain portlets. The Document Structure illustration shows the relationship between components in a portal file. Following are descriptions of the components that make up a portal interface.
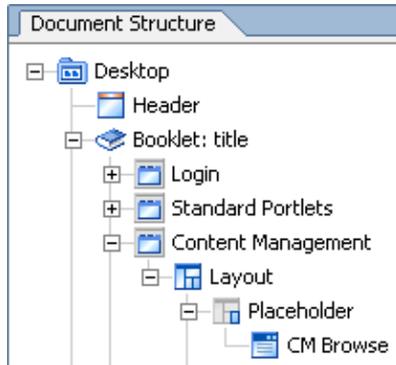
Desktop – A desktop is is an audience specific view of portal components. It contains the portal header, footer, and body. The body contains the bulk of the portal content: books, pages, portlets, and look and feel elements. A portal can support one or more desktops. After a portal administrator sets entitlements on the desktop and makes it ready for public consumption, the desktop is the view of the portal accessed by end users. From there, users may configure their own views through customization of the desktop. Think of a desktop as a user view of a Web site or a portal, exposing a different set of information or tools based on user context. For example, each department in an organization (Human Resources, Accounting, Legal, Sales) can define a portal desktop containing its own portlets, navigation, and look and feel, yet these desktops are all supported by a single portal definition.

Shell – A desktop's header and footer, controlled by a portal shell (.shell file), are the areas that are typically above and below the main body. These areas usually display such things as personalized content, banner graphics, legal notices, and related links.

Book – A Book is a component that provides high–level content organization and navigation. Books contains pages or other books, providing a mechanism for hierarchical nesting of pages and content. Books are identified by a control such as a tab set.

Page – Pages contain the portlets that display the actual portal content. Pages can also contain books and other pages. Pages are identified by a control such as a tab set.

Layout and ***Placeholder*** – A layout is an HTML table definition used by a page to determine the physical locations of portlets on the page. Administrators and users can choose different available layouts for pages. Placeholders are the individual cells in a layout in which portlets are placed.

Portlet – Portlets are the windows that surface your applications, information, and business processes. They can contain anything from static HTML content to Java Controls to complex Web services and process–heavy applications. Portlets can communicate with each other and take part in Java Page Flows that use events to determine a user's path through an application. You can have multiple portlets on a page. You can also have multiple instances of a single Portlet. For example, you can put instances of a portlet on multiple pages; so that if users don't have rights to view one page with that portlet on it, they can see the portlet on a page they do have rights to view. Portlets can have different modes, such as minimize, maximize, edit, delete, configure, and help, selectable from their title bars.

Portal Rendering and Look and Feel Components (not shown in the illustration) – A desktop's appearance is determined by the Look and Feel. A look and feel definition contains two major elements: skins and skeletons.

> ***Skins*** – Skins provide the overall colors, graphics, and styles used by all components in a desktop interface. Skins are collections of graphics and cascading style sheets (CSS) that allow changes to be made to the look and feel of a portal without modifying the portal components directly. References to images and styles are made in the skin rather than being hard coded into the portal definition. The look and feel file provides a path to the skin directory to be used. ***Themes*** are subsets of skins that you can apply to books, pages, and portlets, providing a way of using a different set of styles for individual desktop components.

> ***Skeletons*** – The look and feel file also provides a path to the skeleton directory to be used. Every type of component, from a desktop to a portlet's title bar, has an associated JSP file,

called a skeleton file, that renders it. Some skeleton files are simple, others are more complex. For example, each desktop uses a skeleton file called shell.jsp that simply provides the opening and closing <HTML> tags to render the desktop. A portlet title bar, on the other hand, has a skeleton file called titlebar.jsp that is more complex. It contains Java calls to various windowing methods in the API, references the button graphics to use on the title bar, and determines the placement of title bar elements with an HTML table definition.

After the logic in the skeleton servlets executes for a look and feel, components are rendered hierarchically into a single HTML instance that is the user view of the desktop.

Administrators and users can select from a list of available look and feel definitions, which changes the appearance, and sometimes the behavior, of a user's view of a desktop. For example, the header of one look and feel can contain a static graphic, while the header of another look and feel can contain a campaign that targets a user with personalized content from a content repository.

Portal Development Tools and Services

WebLogic Portal includes many powerful tools and services that make portal development fast and easy.

Designers and Samples – The WebLogic Workshop Portal Extensions include a Portal Designer and a Portlet Designer that provide graphical drag and drop functionality and grid–based property setting, letting you create a sophisticated portal in minutes. For example, you can create a portal file (that appears with a default page), drag one of the sample JSPs from the file tree onto the default page, and a portlet is created for you automatically. You can then save the file and view the portal desktop in a browser.

Java Controls – WebLogic Workshop Enterprise Edition includes many powerful Java Controls that insulate developers from lower–level coding, reducing the lines of code they must write (and therefore reducing the number of bugs). WebLogic Portal provides several Java Controls to assist development of personalized applications, such as a User Profile Control and a Display Content Control.

Portal Java Controls increase developer productivity for application development. Instead of writing code to access an API or J2EE resource directly or even using JSP tags, a developer can insert the User Profile Control into a JSP or Java Page Flow and select the appropriate methods for retrieving and updating user profile information, determining whether or not a user exists, or retrieving a list of users based on search parameters.

JSP Tags – The WebLogic Workshop Portal Extensions include a library of JSP tags that let you perform useful tasks in JSPs with minimal coding.

Content Management – WebLogic Portal includes powerful content management functionality, letting you integrate and manage multiple content management systems in a single virtual content repository. As a developer you can query the repository and retrieve and display personalized content in your portal applications. The Virtual Content Repository is set up and managed in the WebLogic Administration Portal and supports the full development lifecycle.

Unified User Profile – WebLogic Portal includes a unified user profile service that lets you add, access, and manage users and their properties in a single logical location even if the base set of user data is stored on an external system such as an LDAP server. The unified user profile is a key element in such things as triggering personalization, setting role–based administration, and creating end user entitlements to portal resources.

Personalization, Interaction Management, and Behavior Tracking – The Portal Resources Designer lets you define the properties, rules, and actions that display personalized Web content to users, send automatic e−mails, or provide automatic discounts. The designer lets you create campaigns, content selectors, placeholders, and user profile and other properties. The WebLogic Workshop Portal Extensions also provide a set of events for tracking user behavior in portals, and the designer lets you register any custom events you develop.

APIs – WebLogic Portal includes an extensive set of APIs you can use directly for custom application development. For example, the Expressions Package provides a set of expressions that let you construct complex content queries.

Related Topics

Portal Key Concepts and Architecture

Getting Started

Portal Key Concepts and Architecture

The following figure illustrates the flexibility and extensibility of the WebLogic Portal architecture. Whether you are building portal resources and templates in WebLogic Workshop or creating and administering portals with the WebLogic Administration Portal, you can work with individual components that are then unified by the portal framework.

In the figure, (1...n) means one or more, and (0...n) means zero or more. For example, a portal can contain one or more desktops. For resources that occur only once, like Look and Feel and Shell, you can still develop multiple versions even though only one at a time is allowed.
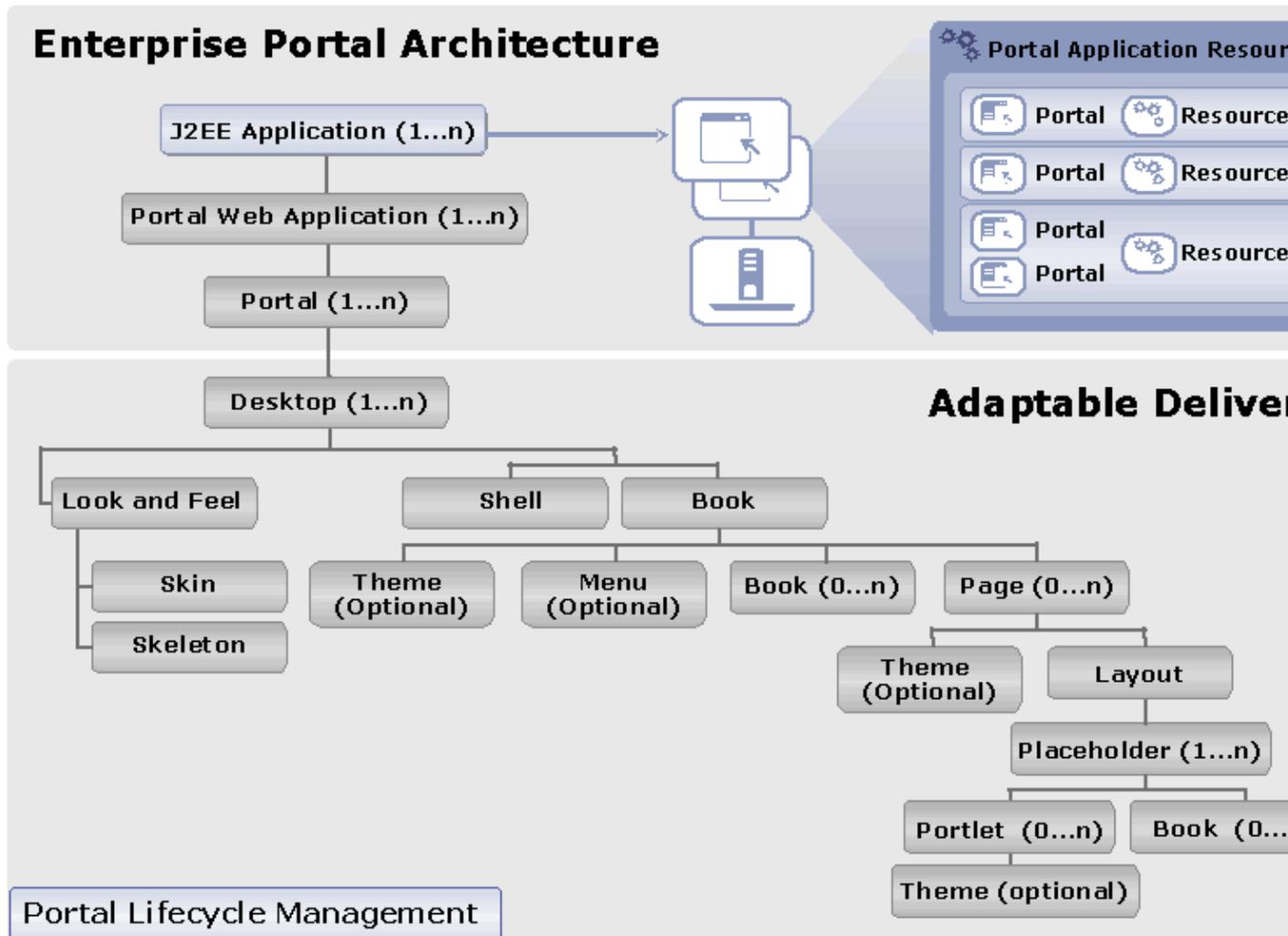
For descriptions of the portal components, see What is a Portal?.

What's Next

Getting Started

Getting Started with WebLogic Portal

You can get started using WebLogic Portal by exploring with samples and tutorials or by diving in and creating a portal.

Exploring WebLogic Portal

Provides links to tutorials that lead you through portal development tasks and to samples that illustrate fully functional portals.

Creating a Portal

Outlines different possibilities for portal development, whether you want to surface an existing application in a portal interface or create a new portal application.

Related Topics

Portal Overview

Exploring WebLogic Portal

Before you begin portal development, you can explore the sample portals provided by BEA and follow the portal development tutorials.

Portal Samples

WebLogic Portal provides many samples to help you become more familiar with tools, features, power, and flexibility of the portal framework.

Portal Tutorials

The Portal Tutorials guide you step by step through different portal development scenarios to help you become more familiar with WebLogic Portal's tools and capabilities.

What's Next

Creating a Portal

Creating a Portal

Creating a portal involves using the WebLogic Portal framework and tools to surface applications in a portal user interface. It also involves adding personalization, campaigns, and behavior tracking to your applications.

You can quickly and easily integrate your own applications into WebLogic Workshop and apply WebLogic Portal's framework, tools, and services to them, or you can create new portal applications in WebLogic Workshop.

The following sections guide you through the process of creating a portal:

Developing Portal Applications

Provides instructions on adding portal functionality to existing applications, developing new portal applications, and building portlets. This section includes information on integrating Web applications, struts applications, and Web services; building Web, JSP, Java Page Flow, and commerce applications; and creating portlets.

Developing Personalized Applications

Provides instructions for adding personalization and campaigns to your portal applications.

Developing Portal User Interfaces

Describes the WebLogic Portal user interface framework and provides instructions for creating a portal user interface and developing user interface resources. This section also provides guidance on building user interfaces to address accessibility guidelines.

Assembling Portal Applications

Provides instructions for assembling portals with existing portal resources, such as creating a portal file, adding books and pages to desktops, changing page layouts, changing the look & feel, and changing the shell.