BEA Workshop for WebLogic Platform Programmer's Guide

Workshop for WebLogic is an IDE built on the Eclipse platform that simplifies the process of developing web-based, service-oriented (SOA) and J2EE applications for deployment on BEA WebLogic Server.

Current Release Information:

- What's New in 9.2
- Upgrading from 8.1

Useful Links:

- Tutorials
- Tips and Tricks

Other Resources:

- Online Docs
- Dev2Dev
- Discussion Forums
- Development Blogs

Workshop for WebLogic provides tools that automate and simplify development:

Web Services

Workshop for WebLogic provides a simple object model for web service development that allows you to define the web service operations as methods. Workshop for WebLogic then automatically handles the staging/sending/receiving/unstaging for SOAP messages that are required to implement a standard web service.

- Learn about these technologies
- Tutorial for developing a web service
- Developing web services in Workshop for WebLogic

For ease of accessing web services, Workshop for WebLogic provides a system control (see below) that allow you to access operations on a remote web service through a simple method call.

Web Applications

When developing **web applications**, Workshop for WebLogic uses the Apache Beehive NetUI framework which integrates a Java controller that maintains state and performs resource access and business logic with JSP files that provide the user interface.

- Learn about these technologies
- Tutorial for developing a web application
- Developing web applications in Workshop for WebLogic

Simplified Resource Access: System Controls

For simplified access to **J2EE** resources and web services, **system controls** encapsulate complex interactions, allowing you to access resources with simple method calls.

- Learn about controls
- Tutorial for working with controls
- Building controls in Workshop for WebLogic

Encapsulating Logic or Resource Access: Custom Controls

For encapsulating your business logic and customizing resource access, **custom controls** provide a simple object model, so that components can be easily used by intermediate programmers without specialized knowledge.

- Learn about these technologies
- Building custom controls in Workshop for WebLogic

Enterprise Java Beans Development Tools

If you need to develop **Enterprise JavaBeans (EJBs)**, Workshop for WebLogic provides tools that simplify the process through EJBGen.

- Learn about these technologies
- Tutorial for developing an EJB
- Creating EJBs in Workshop for WebLogic

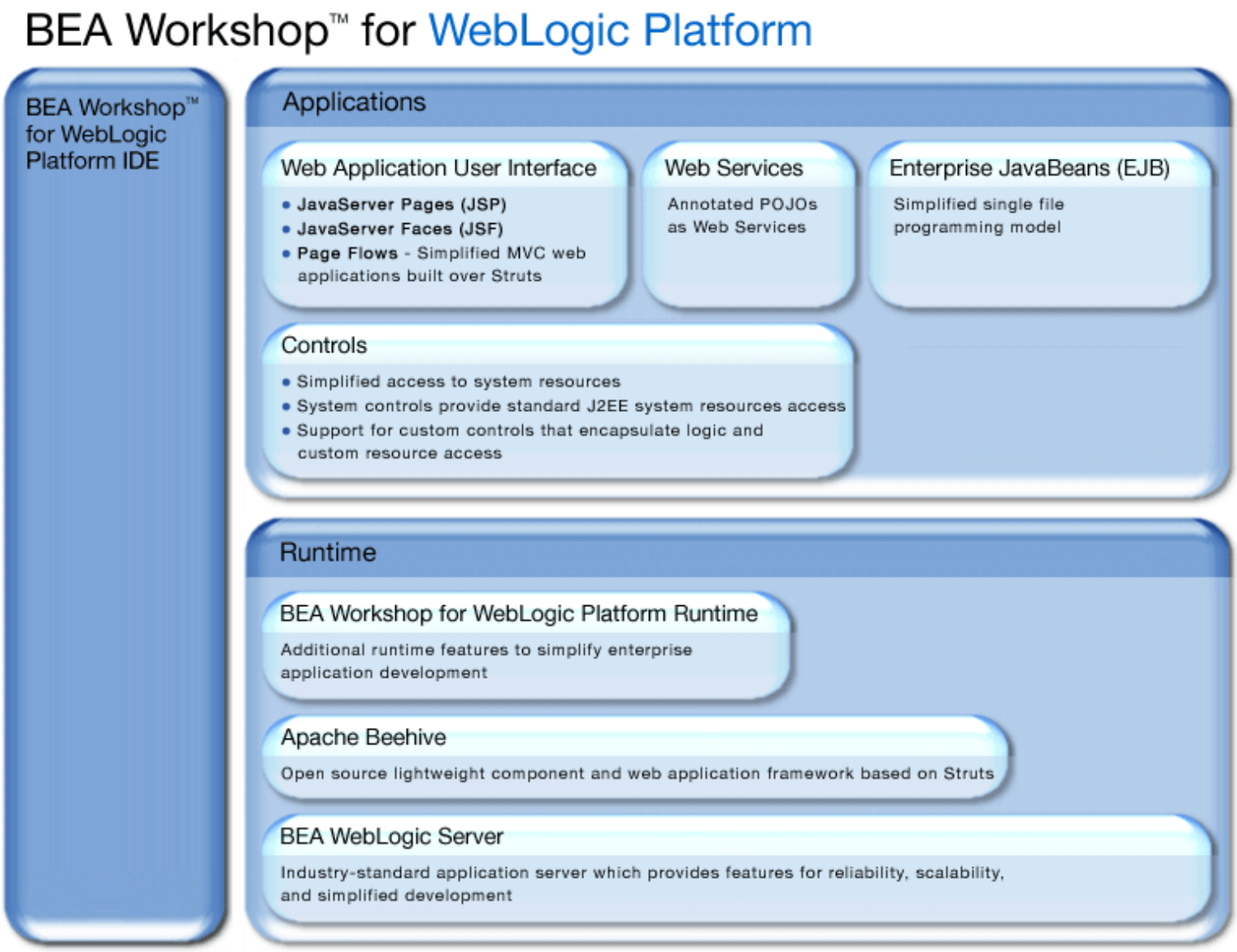
Support for Fundamental Frameworks and Tools

Workshop for WebLogic integrates and extends **open source** tools including:

- J2EE application framework
- Standard web services architecture
- Java 5 annotations
- Eclipse
- Apache Beehive

Overview: BEA Workshop for WebLogic Platform

Workshop for WebLogic helps you develop enterprise-level applications. The diagram below shows the types of applications you can build with Workshop for WebLogic, along with the technologies that support them. Click on the component blocks to see more detailed information for each area.



There are two major components exposed through the IDE:

- **Runtime** Workshop for WebLogic includes extensive runtime libraries that implement support for the application components. Applications developed in Workshop for WebLogic run servers such as BEA WebLogic Server and include standard WebLogic Server libraries, Apache Beehive, and other runtime support implemented within Workshop for WebLogic.
- **Applications** Workshop for WebLogic provides tooling that works with the runtime components to simplify development of a variety of web and enterprise applications which include components built from JSP, JSF, page flows, web services, EJBs and controls.

This documentation set describes the application development features and tooling provided by Workshop for WebLogic to implement and deploy enterprise applications.

One of the best places to start learning about Workshop for WebLogic is with the tutorials provided in this documentation. See Tutorials in Workshop for WebLogic for descriptions of the tutorials included in this documentation.

You can also start by taking a look at sample code. Workshop for WebLogic Samples provides an introduction to the samples included in this release.

What's New in Workshop for WebLogic?

BEA Workshop for WebLogic Platform version 9.2.1

Workshop for WebLogic version 9.2.1 provides new tools for developing web services, web applications, and for testing applications.

Updates available through the Eclipse feature update mechanism. Version 9.2.1 is distributed using an Eclipse update site. For information on downloading the version 9.2.1 update see http://edocs.bea.com/workshop/docs92/index.html on the edocs website and Updating to the Latest Version of Workshop for WebLogic.

Web Service Design View. The web service Design View lets you design all of the major features of a web service using a graphical development environment. The following features are included:

- Creation of web methods, callbacks, and event handlers
- Direct configuration of conversational and buffering properties
- Inline method signature editing, with syntax highlighting and code completion
- Simultaneous editing of separate interface and implementation files
- Improved error detection and messaging

For details see Using Design View to Create Web Services.

Page Flow Overview. The Page Flow Overview gives a graphical overview of an entire page flow, including its JSP pages, actions, and navigational relationships. This high-level view of a page flow gives the developer a total picture of how a page flow is structured. For details see Page Flow Overview.

- Continuous automatic graph layout
- Improved layout algorithm
- Selection is synchronized with the Page Flow Editor
- Graph Tracing mode makes reading complex graphs easier

- Print graphs or export them as images

Improved Test Client functionality. The Test Client now supports directly testing services that have callbacks as well as services that have only JMS as a transport. The Test Client is downloaded along with the Workshop for WebLogic 9.2.1 update, but it must be installed manually. For details on installing and using the Test Client to test web services, see Testing Web Services with the Test Client.

Cheat Sheets. A variety of cheat sheets are now available to get you started with Workshop for WebLogic. The following cheat sheets are available:

- NetUI Web Application: Hello World
- Web Service Application: Hello World
- Using XMLBeans

For details on opening a cheat sheet, see Workshop for WebLogic Cheat Sheets.

BEA Workshop for WebLogic Platform version 9.2

One of the key aspects of this release is its support of open sources technologies. As described below, Workshop for WebLogic integrates these technologies into a development experience whose focus is easing the creation of complex applications.

An IDE built on Eclipse 3.1. This version of the IDE is built on the Eclipse Platform, an open source framework that is now widely used for Java development. Workshop for WebLogic builds on the framework by adding support for the iterative development model introduced in version 8.1, as well as for easing the use of the open source technologies it incorporates. Instead of the proprietary IDE framework used in previous releases, Workshop for WebLogic extends Eclipse 3.1.2 and the Web Tools Platform 1.0.2.

Apache Beehive technologies. Version 9.2 incorporates Apache Beehive, the open source framework for web applications. Beehive includes:

- Beehive NetUI: a web application framework based on Apache Struts. Introduced in 8.1 as Page Flow technology, Beehive NetUI has evolved into an open source web application framework. Version 9.2 includes advanced tooling features to help developers build Beehive NetUI applications, including tools for JSP and Page Flow controller authoring.
- Beehive Controls: a lightweight component framework based upon annotated JavaBeans. Beehive controls provide an easy to use client model for accessing a variety of resource types. The built-in controls introduced in 8.1 (database, EJB, JMS and web service controls) are available as J2EE system controls in 9.2. You can also create your own custom controls using a new, transparent, POJO architecture.

Support for Java Server Faces. Version 9.2 supports integration of Java Server Faces technology with Beehive NetUI, including:

- Support for JSF pages in NetUI Beehive applications
- Tools support for JSF backing files and event handlers
- Integration of JSF with the Beehive NetUI lifecycle

Support for standards-based Java web services. The same groundbreaking web service support in version 8.1 has been carried forward in version 9.2, now built on JSR-181. Web service support includes:

- Asynchronous callbacks
- State management with conversations
- Easy iterative development with the test browser
- Reliable messaging with JMS and WS-Reliability
- Message level security with WS-Security/WS-Policy

Support for Java 5 annotations. As with version 8.1, version 9.2 supports the use of annotations to simplify the development of complex components. While most of the functionality of the version 8.1 annotations carries forward into this version, this version uses Java 5 annotations (based on the JSR-175 standard). Workshop for WebLogic continues to provide tool support to keep the use of annotations simple, including a property editor for intuitive annotation editing.

Tools for upgrading applications from version 8.1. Version 9.2 makes it easy to upgrade your 8.1 applications. Upgrade support includes:

- Pre-upgrade reporting that doesn't affect the original source code
- Automatic handling of application and project structure upgrade
- Automatic file extension renaming
- Wizard and command line versions of the upgrade tool

Documentation Roadmap

By utilizing standardized, open source frameworks such as Eclipse and Beehive, BEA Workshop for WebLogic integrates a standard robust basic IDE platform with BEA's proprietary web development productivity features. However this provides a challenge for documentation delivery, since each component provides its own documentation and the Workshop for WebLogic documentation does not duplicate information located elsewhere.

While documentation is available on the Internet for various components used by Workshop for WebLogic, we recommend that you first consult the documentation shipped with the product. The document set shipped with Workshop for WebLogic is specific to the versions of the software components that are integrated into your version of the product.

Documentation within the IDE (Eclipse and Workshop for WebLogic)

The online documentation is available by clicking **Help > Help Contents** from the Workshop for WebLogic window.

Note: See below for information on launching help in standalone mode.

Top Level Help Heading	Contents
Workbench User Guide	Basics of Eclipse including: • concepts: perspectives, views • display window components and indicators • navigation • importing/exporting files • creating projects CVS integration Using ANT and other external tools
Java Development User Guide	Using the Java code editor Refactoring Using the debugger within applications
Web Application Development Guide	Creating web applications Working with servers J2EE enterprise applications
BEA Workshop for WebLogic Platform Programmer's Guide	Advanced features provided by Workshop for WebLogic, including • Developing web applications with page flows • Creating and using web services • Developing EJBs • Working with controls • Building, debugging and deploying enterprise applications to WebLogic Server This help documentation also provides extensive tutorials and sample code demonstrating how to develop sophisticated J2EE enterprise applications.

The documentation for Workshop for WebLogic is also available online at http://edocs.bea.com. Different versions of documentation are provided, so be sure to check your software version before consulting the web site.

If you are just getting started with Workshop for WebLogic, we recommend that you start with workbench documentation. Then review the BEA Workshop for WebLogic Programmer's Guide, especially the tutorials.

The **BEA Workshop for WebLogic Platform Programmer's Guide** is divided into the following major sections:

IDE User Guide

This section of the manual describes how to get around the IDE:

- Navigation, screen layout
- Building applications, running/testing and debugging
- Defining, using and managing the test server

Upgrading Applications from WebLogic Workshop 8.1

Documentation for the upgrade wizard that converts applications from WebLogic Workshop 8.1 to BEA Workshop for WebLogic Platform 9.2. Since some upgrade requirements cannot be automated, this section of the manual also provides a detailed description of deprecated features and techniques for updating applications.

Web Applications

This section of the manual describes how to develop web applications using page flows. Page flows allow you to separate presentation from business logic by integrating JSP files with a Java controller that maintains state information as the user navigates through JSPs.

Web Services

This section of the manual describes how to develop web services. Web services facilitate Service-Oriented Architecture (SOA) by automating the process of service implementation.

Controls

This section of the manual describes how to develop controls. Controls provide an object model for access to web services and resources. Controls are implemented as annotated classes that provide for standardized resource access or encapsulation of business logic.

Enterprise Java Beans

This section of the manual describes how to develop Enterprise Java Beans (EJBs) with Workshop for WebLogic.

Designing Asynchronous Interfaces

This section of the manual details building web services that are not synchronous ("conversational") and other asynchronous applications.

Beehive Documentation for Java 5 Annotations

Beehive provides extensive PDF and HTML documentation for the controls and annotations used within Workshop for WebLogic to build web applications, web services and EJBs.

The Beehive documents are available locally.

This documentation is also available on the Internet at http://beehive.apache.org.

Java, J2EE and EJB Documentation

The Sun website (http://sun.com) has extensive documentation on:

- Java (http://java.sun.com)

- Enterprise Java Beans (EJBs) (http://java.sun.com/ejb)
- J2EE (http://java.sun.com/javaee)
- Java 5 Annotations (http://java.sun.com/docs/books/tutorial/java/javaOO/annotations.html, http://java.sun.com/j2se/1.5.0/docs/guide/language/annotations.html)

and a great deal more.

Using Help in a Standalone Mode

You might find it useful to launch Workshop for WebLogic help in a standalone mode. For example, due to a characteristic of Eclipse (on which Workshop for WebLogic is built), restarting the IDE by default causes links in an already open help browser to become unavailable after restart. You can have help functionality that's uninterrupted by IDE restarts by launching help in a mode that uses a standalone help server installed with the IDE.

To start standalone help

1. At the command line execute the following. (Execute this command on a single line; it is broken into multiple lines here for readability.)

   ```
   BEA_HOME\jdk150_04\bin\java -classpath BEA_HOME\workshop92\eclipse\plugins\org.eclipse.help.base_3.1.0.jar
       org.eclipse.help.standalone.Infocenter
       -command start
       -eclipsehome BEA_HOME\workshop92\eclipse
       -port 7034
       -noexec
   ```

2. Open a web browser and go to the following URL:

   http://localhost:7034/help/index.jsp

To shut down standalone help

- At the command line execute the following. (Again, execute this command on a single line.)

  ```
  BEA_HOME\jdk150_04\bin\java -classpath BEA_HOME\workshop92\eclipse\plugins\org.eclipse.help.base_3.1.0.jar
      org.eclipse.help.standalone.Infocenter
      -command shutdown
      -eclipsehome BEA_HOME\workshop92\eclipse
      -port 7034
      -noexec
  ```

  Note that after you shut down the help server, links in the help browser will be unavailable until you restart the server.

Workshop for WebLogic Cheat Sheets

The following cheats sheets provide step-by-step instructions for common IDE tasks.

To open the cheat sheets listed below, select **Help > Cheat Sheets**. In the **Cheat Sheet Selection** dialog open the node **BEA Workshop for WebLogic Platform**. Select a cheat sheet from the list provided and click **Ok**.

NetUI Web Application: Hello World

This cheat sheet guides you through the creation of a simple NetUI web application.

To open this cheat sheet select **Help > Cheat Sheets > BEA Workshop for WebLogic Platform> NetUI Web Application: Hello World > Ok**.

Web Service Application: Hello World

This cheat sheet guides you through the creation of a simple NetUI web application.

To open this cheat sheet select **Help > Cheat Sheets > BEA Workshop for WebLogic Platform> Web Service Application: Hello World > Ok**.

Using XMLBeans

This tutorial shows how you can enable automatic generation of Java types from XML schema by using the XMLBeans Builder facet.

To open this cheat sheet select **Help > Cheat Sheets > BEA Workshop for WebLogic Platform> Using XMLBeans > Ok**.

Related Topics

Tutorial: Accessing a Database from a Web Application

Tutorial: Web Service

Using XMLBeans in the IDE

Tutorials in Workshop for WebLogic

In the Workshop for WebLogic documentation you'll find several tutorials to help you get acquainted with the IDE and the applications you can build with it.

In your installation you'll also find sample workspaces containing the completed code for each of these tutorials. For more information, see Workshop for WebLogic Samples.

Getting Started with BEA Workshop for WebLogic Platform

Get an introduction to the basics while creating a simple "Hello, world!" application.

Accessing a Database from a Web Application

Learn how to build a web application capable of accessing a database. This is a general introduction to the web application and control technologies used by the IDE.

Testing a Control with JUnit

Learn how to create a simple control and test it using JUnit.

Create a Simple Timer

Create and test a web service with a timer control. Before you get started, you should know the basics of Workshop for WebLogic, including how to create workspaces, projects and packages and how to run a web service and test operations with the Test Client.

Java Server Faces Integration

Learn how to enable and use Java Server Faces in a web application.

Web Service

Learn how to build a web service and add a simple custom control that calls methods of existing controls.

Advanced Web Services

Find out how to create a web service, insert code to access a control, generate a web service control from a WSDL, and access that control from another control.

Building Enterprise JavaBeans

Get to the basics of building Enterprise JavaBeans by building a very simple application that

includes a session bean, an entity bean, and a Java Page Flow.

Related Topics

Workshop for WebLogic Samples

Updating to the Latest Version of Workshop for WebLogic

The latest version of Workshop for WebLogic is available through an Eclipse update site.

To download and install the latest Workshop for WebLogic features:

1. Select **Help > Software Updates > Manage Configuration**.
2. In the **Product Configuration** dialog, click **Scan for Updates**.
3. In the **Updates** site, open up the nodes **BEA Workshop Update Site > BEA Workshop for WebLogic Platform**. Select the node **BEA Workshop for Weblogic Platform 9.2.x** and click **Next**.
4. Accept the license agreement, click **Next**, and click **Finish**.
5. Wait for the updates to download.
6. In the **Verification** dialog, click **Install All**.
7. After the update is downloaded and installed, the workbench must be restarted for the changes to take effect.

Accessibility Features

Eclipse Accessibility Features

Workshop for WebLogic is built on the Eclipse IDE framework, which provides accessibility features to help people with a physical disability, such as restricted mobility or limited vision, or those with special needs to use software products successfully. The major accessibility features in Eclipse are listed below. Note that these features apply only to Windows. The accessibility features of Eclipse are described in detail in the **Workbench User Guide**.

- Eclipse uses Microsoft Active Accessibility (MSAA) APIs to render user interface elements accessible to assistive technology.
- You can operate all features using the keyboard instead of the mouse. Search for **Navigating the user interface using the keyboard**.
- You can use screen-reader software such as Freedom Scientific's JAWS ™ and a digital speech synthesizer to hear what is displayed on the screen. You can also use voice recognition software, such as IBM ViaVoice ™ to enter data and to navigate the user interface.
- You can magnify what is displayed on your screen in the graphical views.
- Any fonts or colors defined by Eclipse can be set using the **Window > Preferences** dialog**.** Search for **Font and color settings in Eclipse**.
- The function of the keyboard can be extensively customized in Eclipse. Within Eclipse, key strokes and key sequences are assigned to invoke particular commands. Search for **customizing key bindings**.

Workshop for WebLogic Accessibility Features

In addition to the accessibility features of Eclipse, Workshop for WebLogic provides:

- All help documentation is provided online, in a frameless format that is more easily navigated without a mouse
- **Page Flow Editor** provides the keyboard shortcut Shift-F10 to open the context menu of the selected object
- **Page Flow Explorer** adds the following keyboard shortcuts:

  Shift-F10 gives you context menus for the selected object.

  Up Arrow, Down Arrow to move up and down the tree

  Right Arrow/Left Arrow to expand/collapse folders

- **Page Flow Explorer** can be used to manage all the objects (create/rename/open/delete). The only additional features supplied by the **Page Flow Editor** are the ability to create/edit

forwards and invoke actions from pages via anchors or forms

To create/edit forwards via the **Annotations** view, select a corresponding action in the **Page Flow Explorer**, or source editor, then use the property sheet to edit the forward annotations. Use the **Page Flow Explorer** (or source editor) to populate the Annotations view with the relevant annotations, then do the editing there.

Forms and anchors can be created in the JSP editor, and configured through the property sheet.

Key Differences for WebLogic Workshop 8.1 Users

As you've probably already noticed, Workshop for WebLogic 9.2 is quite different from WebLogic Workshop version 8.1. This topic lists a few of the key changes that you should be aware of if you've been using version 8.1.

Many Project Types are Now Represented Through Project Facets

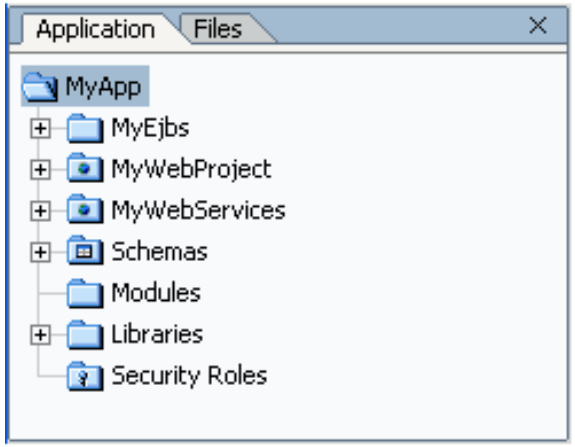
Version 8.1 provided several project types you won't see in version 9.2. The functionality for many of these is provided through project facets. You add facets to add support for the functionality that your application will need. Through facets, Workshop for WebLogic projects can support more functionality than was supported in the version 8.1 web services, web, control, and Java projects.

Contents of an EAR File are Determined by Reference Rather than Containment

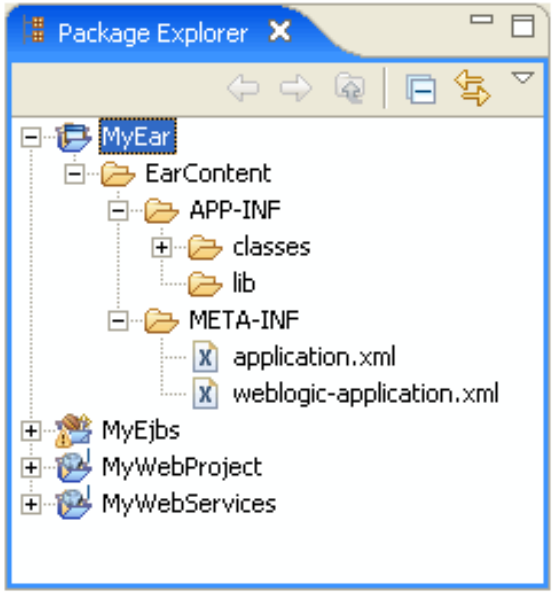
In version 8.1 you created an application that contained projects; projects were in the EAR because they were in the application, and the application represented the resulting EAR file. In version 9.2, you group projects in a workspace, but the workspace doesn't represent the EAR file. Instead, the EAR file is represented by an EAR project in the workspace, and other projects are included in the EAR file when the EAR project has a reference to them. See Applications and Projects for more information.

> **Note:** You can have multiple EAR projects in a given workspace, each with its own EAR file result.

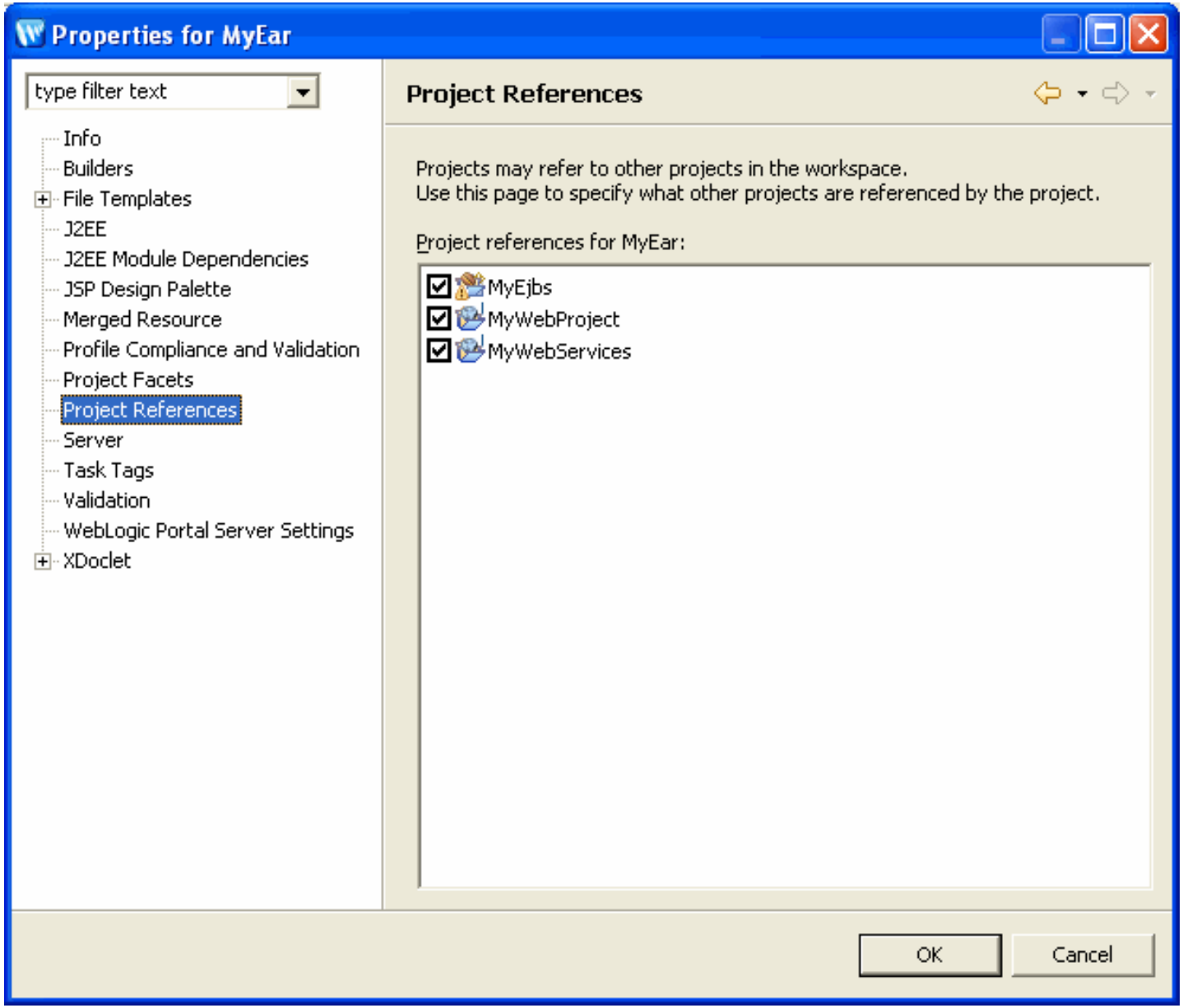
Version 8.1



Version 9.2



The following shows EAR project properties referencing other projects to be included in the EAR file.



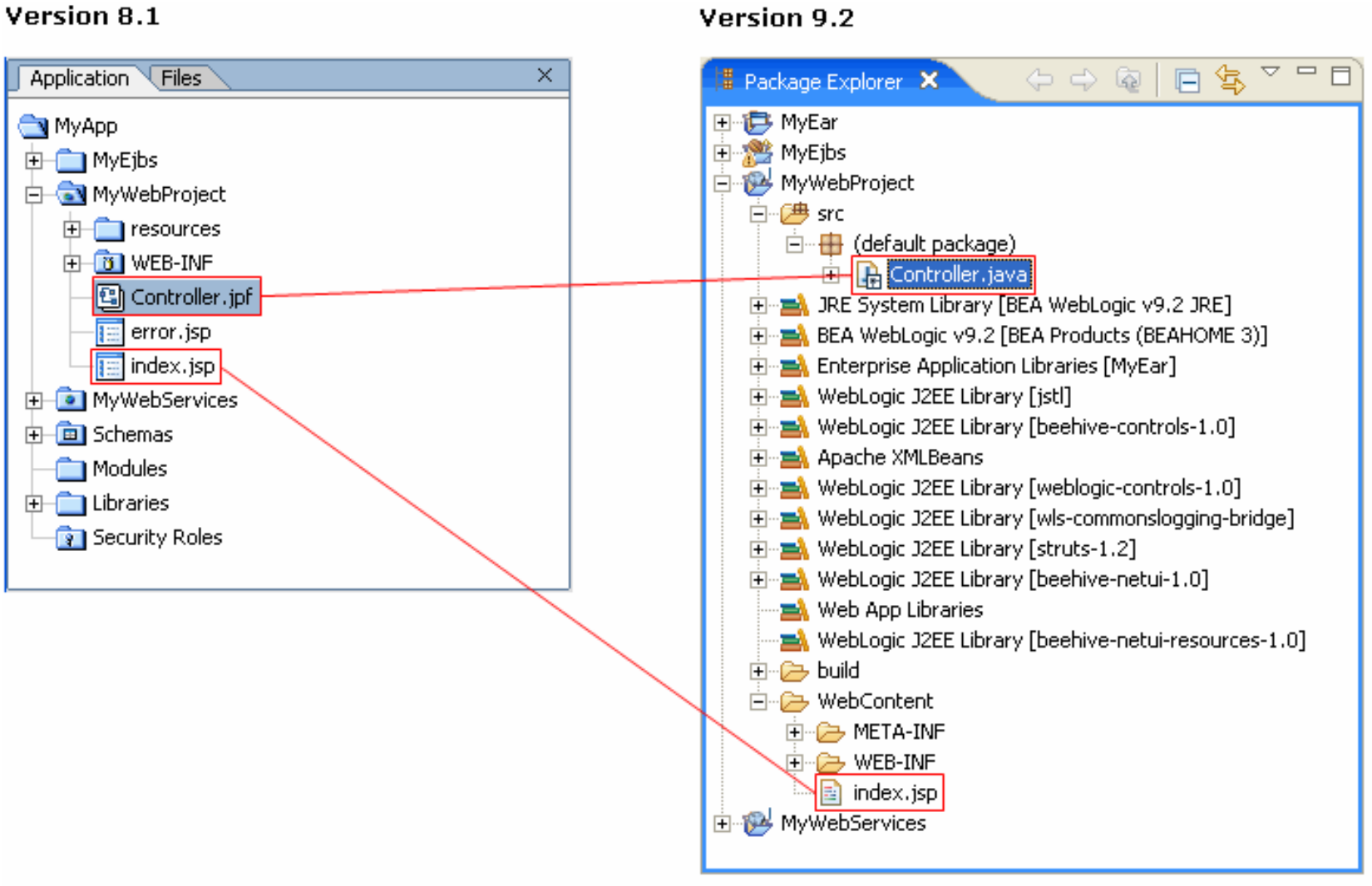
Java Files Now have .java Extensions Regardless of File Type

In version 8.1 web service, page flow controller, and other files had different extensions — such as .jws and .jpf. In the current version these files all have .java extensions.

Page Flow Java and JSP Source Artifacts Now Reside in Separate Parts of a Project

In version 8.1, a page flow's Java and JSP files were kept in the same project directory; in version 9.2 JSP files must be in a corresponding directory in the WebContent area. See Components of the Beehive NetUI Programming Model for more information.

The following illustrates:



Generated Code and Build Output Now Reside in the Project Structure

In version 8.1, generated code and build output was in a parallel directory structure that lived outside user projects. In version 9.2, directories for these resources are within the project structure. These include the .apt_src and build directories, as well as the .xbean_bin and .xbean_src directories (which the XMLBeans Builder is enabled). These changes could impact:

- Source control system configuration. You might need to explicitly exclude these directories (for exampe, the .cvsignore directory).
- Design of external build scripts. As you migrate exported scripts from version 8.1 to version 9.2, you'll want to keep in mind the location of directories that contain generated code and build output.

A Web Project Need Not be Developed as Part of an EAR

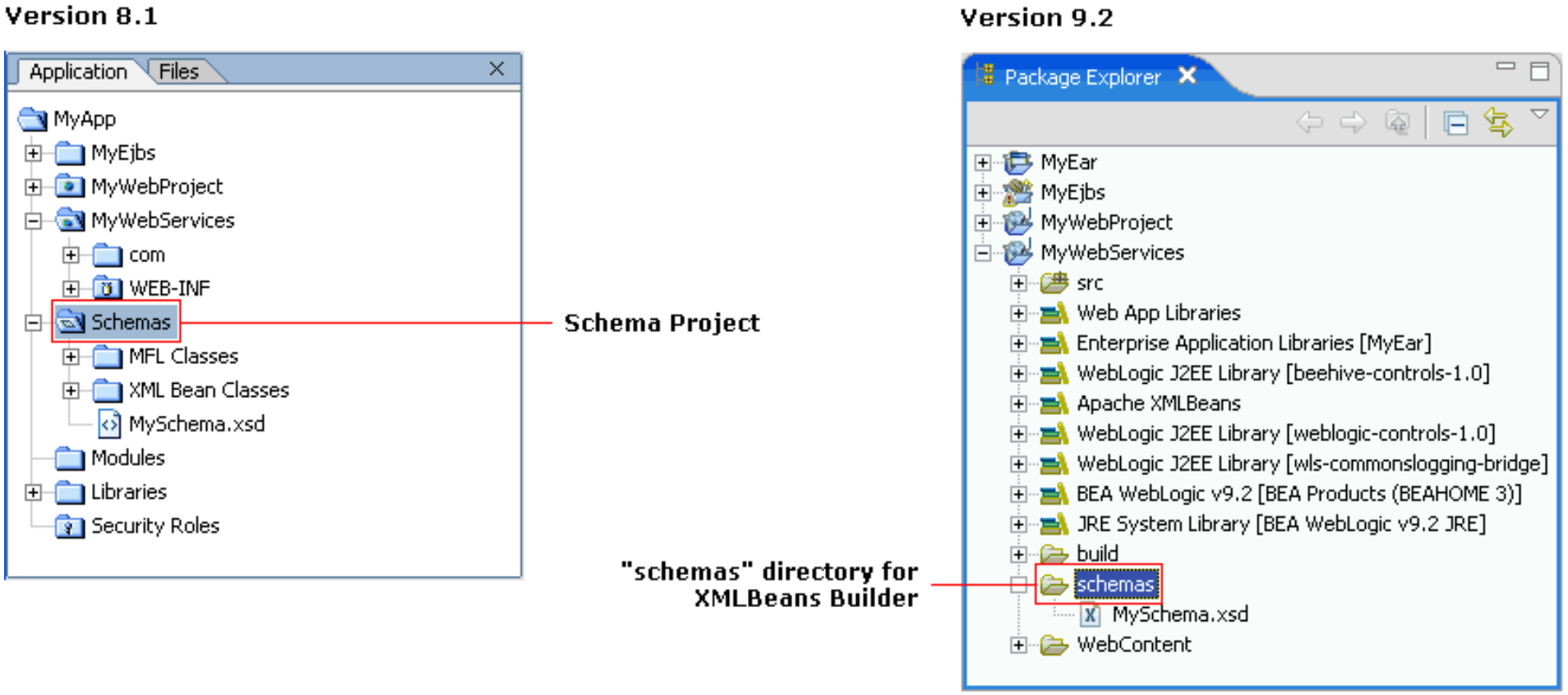
In version 8.1 the built output of an application was always an EAR file. This meant that all projects in the application were deployed in an EAR. A version 9.2 workspace, on the other hand, need not have an EAR project; you can develop and deploy web projects outside the context of an EAR file.

The application.xml and weblogic-application.xml Files are No Longer Generated

In version 8.1 these files were generated from the WORK file, the file that represented an application. In version 9.2, you can edit these files. Note that the Eclipse Web Tools Platform (WTP) components of the IDE manage the representation of projects in these files.

Schema Projects have been Replaced by the XMLBeans Builder Facet

This version doesn't have a schema project type. Instead, you can enable automatic compilation of schemas and WSDLs by adding the XMLBeans Builder facet to the project that will contain them. Note that in this version creating a JAR file with generated Java types is a separate technique for using those types; the JAR file is not automatically generated by the XMLBeans Builder.



Shared Controls and XMLBeans Types are Kept in a Utility Project

In version 8.1 you could use the control and control deliverable projects to develop control code that could be used across your application. Likewise, with a schema project for Java types generated from schema. In version 9.2 you put your control code or schemas into a utility project to share them across projects in the workspace. (Of course, you'd enable the XMLBeans Builder facet on the utility project for schema compilation. See Schema Projects have been Replaced by the XMLBeans Builder Facet for more.)

Related Topics

None.