BEA WebLogic Server™

Deploying WebLogic Server Applications

Release 8.1 Beta
Document Date: December 2002
Revised: December 9, 2002

Copyright

Restricted Rights Legend

Trademarks or Service Marks

Deploying WebLogic Server Applications

Part Number	**Date**	**Software Version**
N/A	December 9, 2002	BEA WebLogic Server Version 8.1 Beta

Contents

About This Document

This document describes how to deploy and redeploy Applications and Modules on WebLogic Server in a production environment.

The document is organized as follows:

- Chapter 1, "Overview of WebLogic Server Deployment," provides an overview of the types of Applications and Modules you can deploy to WebLogic Server.
- Chapter 2, "Quickstart Guide to Deploying Modules," describes how to quickly deploy a new Application or Module to WebLogic Server.
- Chapter 3, "Advanced Deployment Topics," provides detailed information about how WebLogic Server deploys and redeploys application modules.
- Chapter 4, "Performing Common Deployment Tasks," describes how to perform different deployment tasks using both the Administration Console and the `weblogic.Deployer` command-line utility.
- Chapter 5, "Deployment Tools Reference," provides a complete reference to the `weblogic.Deployer` command-line utility and describes other tools used to deploy application modules.

Audience

This document is written for Administrators who want to deploy Java 2 Platform, Enterprise Edition (J2EE) applications or application modules to WebLogic Server. This document assumes that you are working in a production environment, which is generally characterized by multiple WebLogic Server instances or clusters running on multiple machines. It also assumes that you have one or more application module archive files that have been tested and are ready to deploy on a production server.

e-docs Web Site

BEA product documentation is available on the BEA corporate Web site. From the BEA Home page, click on Product Documentation.

How to Print the Document

You can print a copy of this document from a Web browser, one main topic at a time, by using the File→Print option on your Web browser.

A PDF version of this document is available on the WebLogic Server documentation Home page on the e-docs Web site (and also on the documentation CD). You can open the PDF in Adobe Acrobat Reader and print the entire document (or a portion of it) in book format. To access the PDFs, open the WebLogic Server documentation Home page, click Download Documentation, and select the document you want to print.

Adobe Acrobat Reader is available at no charge from the Adobe Web site at http://www.adobe.com.

Related Information

The BEA corporate Web site provides all documentation for WebLogic Server. [[Note to writers: references to other WLS docs that cover or elaborate on the subject matter of your doc, related Sun docs, etc. go here. If you have more than a couple of references to related docs, make it into a bulleted list.]]

Contact Us!

Your feedback on BEA documentation is important to us. Send us e-mail at docsupport@bea.com if you have questions or comments. Your comments will be reviewed directly by the BEA professionals who create and update the documentation.

In your e-mail message, please indicate the software name and version you are using, as well as the title and document date of your documentation. If you have any questions about this version of BEA WebLogic Server, or if you have problems installing and running BEA WebLogic Server, contact BEA Customer Support through BEA WebSupport at http://www.bea.com. You can also contact Customer Support by using the contact information provided on the Customer Support Card, which is included in the product package.

When contacting Customer Support, be prepared to provide the following information:

- Your name, e-mail address, phone number, and fax number
- Your company name and company address
- Your machine type and authorization codes
- The name and version of the product you are using
- A description of the problem and the content of pertinent error messages

Documentation Conventions

The following documentation conventions are used throughout this document.

Convention	**Usage**
Ctrl+Tab	Keys you press simultaneously.
italics	Emphasis and book titles.

Convention	Usage

`monospace text`	Code samples, commands and their options, Java classes, data types, directories, and file names and their extensions. Monospace text also indicates text that the user is told to enter from the keyboard.
	Examples:
	`import java.util.Enumeration;`
	`chmod u+w *`
	`config/examples/applications`
	`.java`
	`config.xml`
	`float`

`monospace italic text`	Placeholders. *Example*: `String` *`CustomerName`*`;`

`UPPERCASE MONOSPACE TEXT`	Device names, environment variables, and logical operators.
	Examples:
	`LPT1`
	`BEA_HOME`
	`OR`

`{ }`	A set of choices in a syntax line.

`[ ]`	Optional items in a syntax line. *Example*:
	`java utils.MulticastTest -n` *`name`* `-a` *`address`*
	     `[-p` *`portnumber`*`] [-t` *`timeout`*`] [-s` *`send`*`]`

`|`	Separates mutually exclusive choices in a syntax line. *Example*:
	`java weblogic.deploy [list|deploy|undeploy|update]`
	     `password {application} {source}`

`...`	Indicates one of the following in a command line: ■ An argument can be repeated several times in the command line. ■ The statement omits additional optional arguments. ■ You can enter additional parameters, values, or other information

Convention	Usage
. . .	Indicates the omission of items from a code example or from a syntax line.

CHAPTER

1 Overview of WebLogic Server Deployment

The following sections provide a basic overview of key BEA WebLogic Server™ deployment topics:

Deployable Modules

A *deployable module* refers to a J2EE application or J2EE module that has been organized according to the J2EE specification. For each module type, the J2EE specification defines both the required files and their location in the directory structure of the module. Modules may include Java classes for EJBs and servlets, resource adapters, Web pages and supporting files, XML-formatted deployment descriptors, and JAR files containing other modules. J2EE does not specify *how* an application is deployed on the target server—only how a standard module is organized.

Module Types

J2EE provides specifications for creating the following module types: Enterprise Applications, Web applications, Web Services, EJB modules, and resource adapters. For more information on a particular module type, refer to the J2EE 1.4 specification at: http://java.sun.com/j2ee/download.html#platformspec.

Enterprise Application

An Enterprise Application consists of one or more of the following components:

- Web Application components—HTML pages, servlets, Java Server Pages, and related files
- Enterprise Java Beans (EJB) components—entity beans, session beans, and message-driven beans
- Connector components—resource adapters
- An `application.xml` deployment descriptor.

Enterprise Applications are packaged as a JAR file with an `.ear` extension. An EAR file contains all of the JAR, WAR, and RAR component archive files for an application as well as the XML descriptor that describes the bundled components.

Web Application

A Web Application always includes the following files:

- At least one servlet or JSP page, along with any helper classes.
- A `web.xml` deployment descriptor, a J2EE standard XML document that describes the contents of a WAR file.

Web Applications may also contain JSP tag libraries, static .html and image files, and a weblogic.xml deployment descriptor, which describes WebLogic Server-specific elements for Web Applications.

Enterprise JavaBean

Enterprise JavaBeans (EJBs) are reusable Java components that implement business logic and enable you to develop component-based distributed business applications. EJB module are packaged as archive files having a `.jar` extension. The archive file or exploded archive directory for an EJB contains the compiled EJB classes, container classes, and XML deployment descriptors for the EJB. See Programming WebLogic Server Enterprise JavaBeans for more information on the different types of EJBs.

Resource Adaptor

A Resource Adaptor (also referred to as a connector) adds Enterprise Information System (EIS) integration to the J2EE platform. Resource Adaptors are packaged as .rar archive files, and `ra.xml` deployment descriptor. Connectors deployed on WebLogic Server may also include a `weblogic-ra.xml` deployment descriptor that specifies WebLogic Server features. See Programming WebLogic Server J2EE Connectors for more information.

XML Deployment Descriptors

A key file that defines each J2EE module type is its XML deployment descriptor. The deployment descriptor an XML document that defines certain runtime characteristics for the module. By editing the XML deployment descriptor, you modify runtime behavior for the module without having to recompile code or reassemble to module itself; the new behavior is read from the descriptor file and implemented when you deploy the module. For example, the deployment descriptors for an EJB module allow you to specify transactional behavior for the EJB when you actually deploy the EJB to a server.

J2EE defines the organization and content of required XML deployment descriptors for each module type. In addition, you can specify optional WebLogic Server XML deployment descriptors to configure deployment behavior in WebLogic Server. These deployment descriptors enable you to maintain the portability of the original J2EE module while utilizing features only available in WebLogic Server.

Table 1-1 lists the types of modules and their associated J2EE-standard and WebLogic-specific deployment descriptors.

Table 1-1 J2EE and WebLogic Deployment Descriptors

Component or Application	Scope	Deployment Descriptors
Web Application	J2EE	`web.xml`
	WebLogic	`weblogic.xml`
Enterprise Bean	J2EE	`ejb-jar.xml`
	WebLogic	`weblogic-ejb-jar.xml` `weblogic-cmp-rdbms-jar.xml`
Resource Adapter	J2EE	`ra.xml`
	WebLogic	`weblogic-ra.xml`
Enterprise Application	J2EE	`application.xml`
	WebLogic	`weblogic-application.xml`
Client Application	J2EE	`application-client.xml`
	WebLogic	`client-application.runtime.xml`

Deployment descriptors are either created manually, or are automatically generated using WebLogic Server Java-based utilities. When you receive a J2EE-compliant JAR file from a developer, it should already contain the J2EE-defined and WebLogic Server deployment descriptors. The Administration Console allows you to modify key deployment descriptor elements in a production environment, without editing the XML by hand.

Deployment Files

WebLogic Server allows you to deploy modules either as a single archive file, or as a directory that contains the same contents of the archive file.

Archive Files

In most production environments, you will receive a deployable module as an archive file. An archive file is a single file that contains all of a J2EE module's classes, static files, directories, and deployment descriptor files. Archive files are created by using the `jar` utility to package the top-level directory of a J2EE module.

Modules that are packaged using the `jar` utility have specific file extension depending on the module type:

- EJBs are packaged as .JAR files.
- Web Applications are packaged as .WAR files.
- Resource Adapters are packaged as .RAR files.
- Enterprise Applications are packaged as .EAR files.

In most cases, you will deploy the archive file itself with no additional preparation.

Exploded Archive Directories

An exploded archive directory contains the same files and directories as a `jar` archive. However, the files and directories reside directly in your file system and are not packaged into a single archive file using the `jar` utility.

You may need to deploy a module as an exploded archive directory, rather than a single archive file, in the following circumstances:

- You are deploying an EJB, Web Application, or Enterprise Application that performs direct file system I/O. In this case, the components that perform the I/O operations should have a physical filesystem directory in which to work.
- You are deploying a Web Application or Enterprise Application that contains static files that you will periodically update. In this case, it is more convenient to deploy the module as an exploded directory, because you can update and refresh the static files without editing the archive.

If you choose to deploy an exploded archive directory, use the `jar` utility to unpack the archive file. For example:

```
jar xvf myejb.jar
```

If you are unpacking an Enterprise Application that contains other archive files (.EJB or .WAR files), you will need to expand each archive file contained in the .EAR file as well.

Location of Files

To deploy a new archive or exploded archive directory, the file(s) must be accessible by the Administration Server for your domain. This means they must reside on the Administration Server machine, or they must be available via a remote, network-mounted directory.

When using the Administration Console to deploy new modules, you have the option to upload files to the Administration Server machine if they are not otherwise available.

Deployment Targets

Deployment Targets are the server instances and clusters to which you deploy a module. During the deployment process, you select the list of targets from the available servers and clusters configured in your domain. You can also change the target list at any time after you have deployed a module.

If you are deploying to a cluster of WebLogic Server instances, by default the deployment targets all server instances in the cluster. This corresponds to homogenous module deployment, recommended in most clusters. If you want to deploy a module only to a subset of servers in the cluster (if you want to "pin" a module to one or more servers), you can also select individual server names. This type of deployment is less common, and should be used only in special circumstances where pinned services are required. See XREF for more information.

Deployment Names

When you deploy a new module to one or more WebLogic Server instances, you specify a deployment name to describe the deployment files, target servers, and other configuration options you selected. You can later redeploy or stop the module on all target servers by simply using the deployment name. The deployment name saves you the trouble of re-identifying the deployment files and target servers when you want to work with the module across servers in a domain.

Advanced Deployment Topics

In addition to the basic deployment concepts described in this section, you can perform more advanced tasks such as updating portions of a deployment, or utilizing special deployment modes. Advanced deployment tasks and concepts are described in "Advanced Deployment Topics" on page 1-7.

CHAPTER

2 Quickstart Guide to Deploying Modules

The following sections how to quickly deploy a module in a BEA WebLogic Server™ domain:

Step 1: Unpack the Archive File (if Necessary)

Deploy modules using an available archive file (.EAR, .WAR, .JAR, or .RAR extension) unless:

- You are deploying a module that performs file I/O and must have a fixed directory structure.
- You are deploying a Web Application and you would like to update static files after the application is deployed.

If either of the above conditions apply, deploy the module from an exploded archive directory. To create an exploded directory, use the `jar` utility to unpack the archive file. Start by creating an empty directory in which you will store the files and moving to the new directory. Name the directory according the application or module you are deploying. For example:

```
mkdir mywebapp
cd mywebapp
```

Use the `jar` utility to unpack the archive:

```
jar xf c:\production\mywebapp.war
```

If you are unpacking an .EAR archive, the archive may contain additional archive files (.JAR and .WAR extensions). In this case, also unpack the embedded archives in the exploded directory:

```
jar xf mymodule1.war
jar xf mymodule2.jar
```

Verify that you have unpacked all archive files in the exploded directory.

Step 2: Start the Deployment Assistant

Access the Administration Console for the domain by loading its URL in your browser (for example, http://myhost:7001/console) and entering the administrator username and password.

Expand the Deployments Node in the left pane of the console, and select the type of module you want to deploy. The available module types are:

- Applications—Enterprise Applications or Web Services packaged as `.ear` files or directories
- EJB Modules—Enterprise JavaBean modules
- Web Application Modules—Web Applications or Web Services packaged as `.war` files or directories
- Connector Modules—Resource adaptors

The right pane of the Administration Console displays currently-deployed modules of the selected type. Click the Deploy a New *module* link in the right pane to start the Deployment Assistant for the Module.

Step 3: Select the Deployment Files

In the first page of the Deployment Assistant, use the links in the Location field to browse directories on the Administration Server machine and locate the Enterprise Application or Web Service to deploy. (If the application does not reside on the Administration Server machine, first use the upload link to upload the application.)

When the assistant detects a valid archive file or exploded archive directory in the current directory, it lists the archive or directory name as a selection beneath the Location field. Select the name of the archive or directory you want to configure for deployment.

If you are deploying an Enterprise Application and your domain contains multiple WebLogic Server instances, you have the option to deploy all modules in the application to a single server, or target different modules of the application to different server instances. Click the Target Application or Target Each Module button, respectively. For other module types, click Continue to select target servers.

Step 4: Select the Target Servers

In the second page of the deployment assistant, use the check boxes to select target servers on which you will deploy the module. To deploy to individual servers, select one or more server instances from the Independent Servers list and click Continue.

To deploy to a cluster of servers, select the name of the cluster from the Clusters list. By default, the assistant deploys an application or module to all server instances in the cluster (the All servers in the cluster option). If you want to deploy to only a subset of the servers in a cluster, select Parts of the cluster and then select the individual server instances to which you want to deploy the application or module.

If you are targeting individual modules of an Enterprise Application, your selections apply only to the module name displayed in the header of the console page (for example, Step 2 - Select targets for module “*module_name*”). If the application has additional modules to deploy, the console re-displays the Select Targets page after you click Continue, allowing you to target the next module to different server instances.

Click Continue when you have finished selecting target servers for the module.

Step 5: Deploy the Files

On the final page of the deployment assistant, you review your selection of target servers, choose a deployment staging mode, and define a deployment name for the module. Review the entries under the Deployment Targets heading. If you need to change a target, click your browser’s Back button.

The Source accessibility header displays the selected staging mode for deploying the archive file or exploded archive directory:

- **Copy this application onto every target for me**—This option is selected by default if you targeted the module to a cluster or to multiple server instances. This corresponds to “stage” mode where the Administration Server copies the application files to each targeted server; the targeted servers then deploy the application using their copy of the source files.
- **I will make the application accessible from the following location**—This option is selected by default if you targeted the module to a single server instance. This corresponds to “nostage” mode where the server deploys a module from a single directory; all targeted servers must be able to access the directory to deploy the application. Select this option if you are deploying to a cluster that resides on a single physical machine.

In the Identity header, the Name field specifies a unique name to refer to this deployment in the Administration Console. Accept the default name or enter a new name to describe the application or module.

Click Deploy to accept the values on this page and deploy the module to the listed server instances.

3 Advanced Deployment Topics

This chapter describes more advanced deployment topics that might not be used in every WebLogic Server installation. It includes the following sections:

Deployment Staging Modes

The deployment staging mode determines how a module's archive files are made available to target servers that must deploy the module. WebLogic Server provides three different options for staging archive files: stage mode, nostage mode, and external_stage mode. You can set the staging mode either at the WebLogic Server level or at the application level, which overrides the server setting.

Stage Mode

Stage mode indicates that the Administration Server copies the deployment files from their original location to the staging directories of each targeted server. For example, if you deploy a J2EE Application to three servers in a cluster, the Administration Server copies the deployment files to directories on each of the three server machines. Each server then deploys the J2EE Application using its local copy of the archive files.

Stage mode is the default (and preferred) mode when deploying to more than one WebLogic Server instance.

Nostage Mode

Nostage mode indicates that the Administration Server does not copy the archive files from their source location. Instead, each targeted server must access the archive files from a single source directory for deployment. For example, if you deploy a J2EE Application to three servers in a cluster, each server must be able to access the same application archive files (from a shared or network-mounted directory) to deploy the application.

In nostage mode, the web application container automatically detects changes to JSPs and servlets.

Nostage mode is the default mode when deploying only to the Administration Server (for example, in a single-server domain). You can also select nostage mode if you run a cluster of server instances on the same machine.

External_stage mode

External_stage mode is similar to stage mode, in that the deployment files must reside locally to each targeted server. However, the Administration Server does not automatically copy the deployment files to targeted servers in external_stage mode; instead, you must ensure that the files are copied to the staging directory of each targeted server.

External_stage mode is the least common deployment staging mode. It is generally used only in environments that are managed by third-party tools that automate the required copying of files.

Summary Of Staging Mode Behavior

The following table describes how staging attribute and path settings affect an application's deployment:

Table 3-1 Application Deployment

Staging Mode	Path	Admin Server	Managed Server
stage	Relative	Application files are copied to a directory named after the application in the staging directory, and activated from there (e.g., `server/stage/myapp/app.ear).`	Files are distributed to the managed server staging area and distributed from there.
nostage	Relative	Path is relative to the staging directory. That is, if the staging directory is `/stage`, and the relative path is xdir/myapp, then files are assumed to reside at `/stage/xdir/myapp`. No files are copied.	Same as for Admin Server.
external_ stage	Absolute, Relative	Files are assumed to reside in the server's staging area and are loaded from there. No files are copied. The deployment should be copied in a directory with the same name as the application name under each target server's staging directory (e.g., `/server/stage/myapp/web.war`)	Same as for Admin Server.
stage	Absolute	Same as if path were relative. Application files are copied to a directory named after the application in the staging directory, and activated from there (e.g., `server/stage/myapp/app.ear).`	Same as if the path were relative. Application files are distributed to the staging directory and activated from there.
nostage	Absolute	Files are not copied and are activated from the absolute path.	Same as for Admin Server.

Best Practices for Choosing a Deployment Staging Mode

Note: The Administration Console automatically recommends the optimal staging mode based on your target server selections. In most cases, you should accept the default staging mode that the Administration Console provides.

If you are deploying to a standalone server (Administration Server), or if all target servers reside on the same machine, select nostage mode. In either case, all of the target servers can access the same set of files for deployment.

If you are deploying to multiple, remote servers in a production environment, use stage mode. Stage mode ensures that each server has a local copy of the deployment files on hand, even if a network outage makes the Administration Server unreachable. If you do not want the Administration Server to copy the files for you, use external_stage mode instead and ensure that the files are copied before deployment.

Server vs. Application Staging Modes

When you deploy an application or module using the Administration Console, the staging mode is set at the application level. The application staging mode always overrides any deployment mode specified for the target server itself.

The server staging mode specifies the default deployment mode for a server if none is specified at deployment time. For example, the server staging mode is used if you deploy an application or module using `weblogic.Deployer` and you do not specify a staging mode.

Deployment Order

By default, WebLogic Server deploys server-level resources (first JDBC and then JMS) before deploying application modules. Modules are then deployed in order starting with connector modules, followed by EJB modules, and finally Web Applications.

The actual deployment order of modules is determined by their Load Order attribute. By default, new applications and modules are configured with a Load Order value of 100. Modules with a lower Load Order value are deployed before those with a higher value during startup. Modules with the same Load Order value are deployed in alphabetical order using the deployment name.

You can change the deployment order for modules by setting `ApplicationMBean LoadOrder` attribute in the Administration Console.

Ordering Components Within an Application

If the application is an EAR, the individual components are loaded in the order in which they are declared in the `application.xml` deployment descriptor. See Editing Enterprise Application Deployment Descriptors. As with other modules, you can specify the load order of modules within an EAR by changing the `LoadOrder` attribute.

Stopping and Redeploying Modules

When you modify a component (for instance, a servlet, JSP, or HTML page) of a Web Application on the Administration Server, you must take additional steps to refresh the modified component so that it is also deployed on any targeted Managed Servers. One way to refresh a component is to redeploy the entire Web Application. Redeploying the Web Application means that the entire Web Application (not just the modified component) is re-sent over the network to all of the Managed Servers targeted by that Web Application.

Note the following regarding re-deployment of Web Applications:

- Depending on your environment, there may be performance implications due to increased network traffic when a Web Application is re-sent to the Managed Servers.
- If the Web Application is currently in production and in use, redeploying the Web Application causes WebLogic Server to lose all active HTTP sessions for current users of the Web Application.

- If you have updated any Java class files, you must redeploy the entire Web Application to refresh the class.
- If you change the deployment descriptors, you must redeploy the Web Application.

Classloading Considerations for Redeployment

Section T.B.D.

Deploying With an Alternate Deployment Descriptor

WebLogic Server enables you to change the run-time deployment configuration of an application without having to modify and repackage the contents of the archive itself. You accomplish this by specifying an alternate deployment descriptor file to use when deploying a module.

An alternate deployment descriptor file can reside anywhere outside a packaged archive file or exploded archive directory. (You cannot store alternate Deployment Descriptor files within an archive or archive directory.) You identify the external file at deployment time, and WebLogic Server uses the alternate file in place of the module's existing (packaged) deployment descriptor. If the module is deployed using stage mode, alternate deployment descriptor files are copied to the top level of the module's subdirectory in each target server's staging directory (for example, `*domain_directory/servername/*stage/myapp/.`).

You can specify an alternate deployment descriptor file to use in place of either a standard J2EE deployment descriptor (such as `application.xml`) or a WebLogic Server deployment descriptor (such as `weblogic-application.xml`).

Common Uses for Alternate Deployment Descriptors

Alternate deployment descriptors are generally used wth archived Enterprise Applications (`.EAR` files), because they enable you to change deployment parameters without repackaging the application itself. You can also specify alternate descriptors to use with exploded archive directories.

Common applications include:

- Changing deployment parameters for an encrypted `.EAR` file that may be more cumbersome to repackage or explode.
- Changing the subset of modules that you choose to deploy from the `.EAR`. For example, although an `.EAR` may contain multiple, logical applications, you can use an alternate deployment descriptor to deploy only a subset of available applications to a particular WebLogic Server instance.
- Making simple changes to an application's resource usage, such as changing the JDBC DataSource it uses in a production environment.

Limitations for Alternate Deployment Descriptors

If you deploy an Enterprise Application using an alternate descriptor, you cannot change the alternate descriptor file when you redeploy the application. For example, you cannot use the `weblogic.Deployer -redeploy` and specify a different filename for the alternate deployment descriptor. WebLogic Server always uses the same alternate descriptor file during a redeployment.

In addition, you cannot specify an alternate deployment descriptor for either the `weblogic.xml` or `weblogic-ejb-jar.xml` files in this release.

Command-Line Options for Specifying Descriptors

To use an alternate deployment descriptor, you simply use one or both of the following options to `weblogic.Deployer`:

- `-altappdd`—specifies the name of an alternate J2EE deployment descriptor (`application.xml`).

- `-altwlsappdd`—specifies the name of an alternate WebLogic Server deployment descriptor (`weblogic-application.xml`).

See “Deploying an Enterprise Application with an Alternate Deployment Descriptor” on page 4-9 for an example of using alternate descriptor files.

Two-Phase Deployment Protocol

The new two-phase deployment protocol helps to maintain domain consistency. In previous versions of WebLogic Server, when you deployed an application, the administration server sent a copy of the application file(s) to all the targeted servers, which then loaded the application. If deployment to any of those servers failed or partially failed, the entire deployment’s state across its target servers became inconsistent.

In the current release of WebLogic Server, deployment first prepares the application across all target servers and then activates the application in a separate phase. If a deployment of an application fails in either the preparation or activation phase, then the cluster deployment is failed.

For information about using the earlier WebLogic Server deployment protocol, see WebLogic Server 6.x Deployment Protocol.

The new deployment protocol supports the following new features for deployed applications:

- **Consistent deployment states for clusters**. If an application targeted to a cluster fails on any of the cluster members in the prepare phase and then in the activate phase, the application is not activated on any of the cluster members. This helps to ensure that the cluster is kept homogeneous.
- **Application ordering**. At server startup, you set the order of application activations. See Deployment Order.
- **Application-scoped configuration**. Certain resources can be configured and scoped for an application. These include connection pools, security realms and XML related resources. See Overview of Application Scoping.
- **Improved redeployment**. You do not need to undeploy before redeploying. See Redeploying or Stopping a Module.

- **Improved API**. A simple API separates configuration from the actual deployment operations. When a deployment is requested, this API creates the necessary configuration (MBeans) for you. Also, the deployment operations are not on the MBeans themselves, so you can change the configuration (such as the target lists) without affecting the deployed application, until a deployment request is initiated. See Deployment Management API, and see also the API documentation at `weblogic.management.deploy.`
- **Deployment status**. It is now easier to track the progress of a deployment especially when it has multiple targets. See Example Uses of the weblogic.Deployer Utility, and WebLogic Administration Console help on Tasks.

Prepare Phase and Activate Phase

The two-phase model makes inconsistent deployment states in clusters less likely by confirming the success of the prepare phase before deploying the application on any targeted servers. A deployment that fails during the prepare phase will not enter the activation phase.

Prepare Phase

The prepare phase of deployment, the first phase, distributes or copies files and prepares the application and its components for activation, validating them and performing error checks on them. The purpose of the prepare phase is to ensure that the application and its components are in a state in which they can be reliably deployed.

Activate Phase

The second phase, the activate phase, is the actual deployment, or activation, of the application and its component with the relevant server subsystem. After the activate phase, the application is made available to clients.

Enforcing Cluster Constraints for Deployment

When you deploy a module to a WebLogic Server cluster, the default two-phase deployment behavior ensures that the deployment succeeds only on clustered server instances that are reachable by the Administration Server. Servers that are unreachable (for example, due to a network failure between the Administration Server and a Managed Server), receive the deployment only after the Administration Server can reach the server. This can potentially lead to a situation where some servers in the cluster use a new version of a deployed module, while other, unreachable servers use an older version of the module.

The `ClusterConstraintsEnabled` option enforces a strict two-phase deployment policy for *all* members of the cluster. `ClusterConstraintsEnabled` ensures that a deployment to a cluster succeeds only if all members of the cluster are reachable and can deploy the specified module. You set the `ClusterConstraintsEnabled` option at the domain level by supplying the startup argument to the domain’s Administration Server:

- `-DClusterConstraintsEnabled=true` enforces strict cluster deployment.
- `-DClusterConstraintsEnabled=false` uses the default two-phase deployment behavior; deployment is ensured only for clustered servers that are reachable by the Administration Server.

WebLogic Server 6.x Deployment Protocol

By default, the two-phase deployment protocol is used for deploying new applications by all available deployment tools. The current administration server still supports the WebLogic Server 6.x deployment protocol, and this protocol is used when:

- Configured applications do not specify the two-phase deployment protocol (`ApplicationMBean.TwoPhase=true`).
- The application contains multiple modules and is not an EAR.

To configure an application that uses 6.x protocol to start using the two-phase protocol, remove the application from the domain—removing its configuration—and then re-activate the application, as follows:

1. Remove the application using `weblogic.Deployer`. Enter a command in the following form:

```
java weblogic.Deployer -adminurl http://admin:7001 -name app
-targets server -remove
```

2. Reactivate the application using `weblogic.Deployer`. Enter a command in the following form:

```
java weblogic.Deployer -activate -name ArchivedEarJar -source
C:/MyApps/JarEar.ear –target server1
```

The application will redeploy using the new protocol.

Deployment Topics for Developers

The following topics apply only to deploying applications to development servers. See the associated links for more information.

Auto-Deployment

Auto-deployment is a method for quickly deploying an application on the administration server. It is recommended that this method be used only in a single-server development environment for testing an application. Use of auto-deployment in a production environment or for deployment of components on managed servers is not recommended.

If auto-deployment is enabled, when an application is copied into the `\applications` directory of the administration server, the administration server detects the presence of the new application and deploys it automatically (if the administration server is running). If WebLogic Server is not running when you copy the application to the `\applications` directory, the application is deployed the next time the WebLogic Server is started. Auto-deployment deploys only to the administration server.

See Auto-Deployment for Development Environments for more information.

Deployment from a Development Directory

WebLogic Server provides a special deployment mode to help you quickly update and redeploy application modules during the development phase. This deployment mode allows you to deploy a module directly from a development directory structure, without having to package the module using either the `jar` utility or by creating an exploded application directory. See Programming WebLogic Server Applications for more information on this deployment mode.

4 Performing Common Deployment Tasks

This section describes how to perform common deployment tasks using the Administration Console and `weblogic.Deployer` utility. It includes the following sections:

Uploading Deployment Files to the Administration Server

In order to deploy a module to servers in a domain, the deployment file(s) must be assessable to the domain’s Administration Server. If the files do not reside on the Administration Server machine or are available to the Administration Server machine via a network mounted directory, use the instructions below to upload files.

Note: The upload functionality helps you upload a single archive file to the Administration Server machine for deployment. If you are deploying an exploded archive directory, use a network file copy utility to copy the exploded directory to the Administration Server Machine.

When you upload files to the Administration Server machine, the archive file is automatically placed in the server’s upload directory. You can configure the path of this directory using the instructions in “Changing a Server Staging Mode or Staging Directory” on page 4-15.

Administration Console Tasks

To upload an archive file to the Administration Server machine using the Administration Console:

1. Start the Administration Console for your domain.
2. Expand the Deployments node in the left pane to display the different deployment types.
3. In the left pane, select the type of application or module that you want to deploy. The available deployment types are:
   - Applications—Enterprise Applications or Web Services packaged as `.ear` files or directories
   - EJB Modules—Enterprise JavaBeans
   - Web Application Modules—Web Applications or Web Services packaged as `.war` files or directories

- Connector Modules—Resource adaptors
- Startup & Shutdown—Startup classes and Shutdown Classes

4. In the right pane, select the Deploy a new *module* link, where *module* is the type of application or module you want to deploy. This initiates the Deployment Assistant for the module.
5. The right pane provides a link to upload the deployment files if necessary. Click the link to display the upload page.
6. Use the Browse button to locate the archive file you want to upload. Select the file and click Open.
7. Click the Upload tab to upload the selected file to the Administration Server's upload directory.

Weblogic.Deployer Tasks

When using weblogic.Deployer to deploy a new module, add the `-upload` option to upload the archive file before deployment. For example:

```
java weblogic.Deployer -adminurl http://localhost:7001 -user
weblogic -password weblogic -name mymodule -targets myserver
-upload -deploy c:\localfiles\myapp.ear
```

Deploying a New Application Module to Server(s) on One Machine

When you deploy a new module to one or more servers that reside on the same machine, you can use the nostage deployment mode. With nostage mode, the administration server and all target servers deploy using the same deployment files (deployment files are not copied to servers' stage directories).

Administration Console Tasks

The Administration Console automatically defaults to nostage mode when you deploy a module to the Administration Server, or to a standalone server. If you target more than one server and the servers reside on the same machine, you must manually specify nostage mode.

To deploy a module to servers on one machine:

1. Start the Administration Console for your domain.
2. Expand the Deployments node in the left pane to display the different deployment types.
3. In the left pane, select the type of application or module that you want to deploy. The available deployment types are:
   - Applications—Enterprise Applications or Web Services packaged as `.ear` files or directories
   - EJB Modules—Enterprise JavaBeans
   - Web Application Modules—Web Applications or Web Services packaged as `.war` files or directories
   - Connector Modules—Resource adaptors
   - Startup & Shutdown—Startup classes and Shutdown Classes
4. In the right pane, select the Deploy a new *module* link, where *module* is the type of application or module you want to deploy. This initiates the Deployment Assistant for the module.
5. In the first step of the Deployment Assistant, select the archive file or exploded archive directory that you want to deploy. Use the online help for the Deployment Assistant if you need further instructions.
6. In the second step of the Deployment Assistant, select the servers and cluster to which you want to deploy the application.
7. In the third step of the Deployment Assistant, review your choices. In the Source Accessibility header, make sure that the following option is selected:
   - I will make the application accessible from the following location

This option corresponds to nostage mode, where all targets deploy the module from the specified location.

8. Click Deploy to deploy the module.

Weblogic.Deployer Tasks

When using the `weblogic.Deployertool` utility, specify the nostage mode explicitly when you deploy to servers on the same machine. For example:

```
java weblogic.Deployer -adminurl http://localhost:7001 -user
weblogic -password weblogic -name mymodule -targets
myserver1,myserver2,myserver3 -nostage -deploy
c:\localfiles\myapp.ear
```

Deploying a New Application Module to Servers on Multiple Machines

When deploying a module to servers on different machine, you generally use the stage deployment mode, which copies deployment files from the Administration Server to each target server before deploying. In rare circumstances you may also use external_stage mode, in which you must ensure that deployment files are copied to each servers staging directory before deployment. See "Deployment Staging Modes" on page 3-1 for more information.

Administration Console Tasks

The Administration Console allows you to select between nostage and stage mode when deploying a new module. If you need to deploy in external_stage mode, use the `weblogic.Deployer` instructions in the next section.

To deploy a module to multiple servers on different machines:

1. Start the Administration Console for your domain.

2. Expand the Deployments node in the left pane to display the different deployment types.
3. In the left pane, select the type of application or module that you want to deploy. The available deployment types are:
   - Applications—Enterprise Applications or Web Services packaged as `.ear` files or directories
   - EJB Modules—Enterprise JavaBeans
   - Web Application Modules—Web Applications or Web Services packaged as `.war` files or directories
   - Connector Modules—Resource adaptors
   - Startup & Shutdown—Startup classes and Shutdown Classes
4. In the right pane, select the Deploy a new *module* link, where *module* is the type of application or module you want to deploy. This initiates the Deployment Assistant for the module.
5. In the first step of the Deployment Assistant, select the archive file or exploded archive directory that you want to deploy. Use the online help for the Deployment Assistant if you need further instructions.
6. In the second step of the Deployment Assistant, select the servers and cluster to which you want to deploy the application.
7. In the third step of the Deployment Assistant, review your choices. In the Source Accessibility header, make sure that the following option is selected:
   - Copy this application onto every target for me

   This option corresponds to stage mode, where the administration server copies the deployment files to each target server before the servers deploy the module.
8. Click Deploy to deploy the module.

Weblogic.Deployer Tasks

When deploying modules to multiple machines using the `weblogic.Deployer` utility, specify the stage or external_stage mode explicitly at the command line. For example:

```
java weblogic.Deployer -adminurl http://localhost:7001 -user
weblogic -password weblogic -name mymodule -targets
myserver1,myserver2,myserver3 -stage -deploy
c:\localfiles\myapp.ear
```

Deploying Enterprise Application Modules to Different WebLogic Server instances

An Enterprise Application module (.EAR file) differs from other deployable modules in the .EAR file can contain other module types (.WAR and .JAR archives). When you deploy an Enterprise Application using the Administration Console, you have the opportunity to target all of the archive's modules together, or distribute the application by targeting different modules to different servers in the domain.

Note: If you want to distribute an .EAR archive's modules over multiple WebLogic Server instances, use the Administration Console for deployment. The `weblogic.Deployer` utility deploys all .EAR modules to the same target servers.

Administration Console Tasks

1. Start the Administration Console for your domain.
2. Expand the Deployments node in the left pane to display the different deployment types.
3. In the left pane, select the Applications node to display the currently-deployed Enterprise Application in your domain.
4. In the right pane, select the Deploy a new Application link. This initiates the Enterprise Application Deployment Assistant.
5. In the first step of the Deployment Assistant, select the .EAR file or exploded Enterprise Application directory that you want to deploy. Use the online help for the Deployment Assistant if you need further instructions on this step.

After you have selected the module, click the Target Each Module button to begin targeting the individual .JAR and .WAR modules included in the Enterprise Application.

6. In the second step of the Deployment Assistant, select the servers and clusters to which you want to deploy the current module. Note that your selection of target servers applies only to the module name displayed at the top of the page.

   Click continue to select target servers for the next module in the Enterprise Application.

7. Repeat step 6 for each module included in the Enterprise Application. When you have finished targeting all available modules, continue with the next step.

8. After you have finished targeting each module, the Deployment Assistant provides the opportunity to review your choices. The Deployment Targets section of the page displays each module in the Enterprise Application and its associated targets. If you need to change the target servers for one or more modules, click your browser's back button and reselect the targets.

9. Select one of the following options in the Source accessibility section to set the deployment staging mode for the application:

   - Copy this application onto every target for me—This option corresponds to stage mode, where the Administration Server copies the deployment files to each target server before the servers deploy the module.
   - I will make the application accessible from the following location—This option corresponds to nostage mode, where all targets deploy the module from the specified location.

10. Accept the default name for the Enterprise Application, or enter a new name to use to refer to this application in the Administration Console.

11. Click Deploy to deploy all modules.

Deploying an Enterprise Application with an Alternate Deployment Descriptor

To specify alternate deployment descriptors (external to the deployed `.EAR` file) when deploying an Enterprise Application, specify the file names with the `weblogic.Deployer` command. Note that you cannot specify alternate descriptor files when deploying with the Administration Console.

Administration Console Tasks

To specify an external `application.xml` and `weblogic-application.xml` file when deploying an Enterprise Application, use the command:

```
java weblogic.Deployer -adminurl http://localhost:7001 -user
weblogic -password weblogic -name myapplication -targets
myserver1,myserver2,myserver3 -stage -deploy -altappdd
c:\myfiles\myextapplication.xml -altwlsappdd
c:\myfiles\myextwlsapplication.xml c:\localfiles\myapp.ear
```

Changing the Order of Deployment for a Module

You can change the deployment order for modules by setting `ApplicationMBean` `LoadOrder` attribute in the Administration Console. The `LoadOrder` attribute controls the load order of deployed modules relative to one another—modules with lower `LoadOrder` values deploy before those with higher values. In all cases, modules are deployed after the WebLogic Server instances has initialized dependent subsystems.

Notes: You cannot change the load order of deployed modules using the `weblogic.Deployer` utility.

Modules deployed prior to WebLogic Server 7.0 specify the load order value in their deployment descriptor files; you cannot change this load order using the Administration Console.

Administration Console Tasks

Follow these steps to view or change the deployment order of modules deployed to the WebLogic Server domain:

1. Select the Deployments node in the left pane. The right pane displays all modules configured for deployment in the domain, listed in their current deployment order.
2. Select the Change button next to a module name to display the Change Deployment Order page.
3. Enter a new value in the Load Order field, and click Apply to apply your changes. The again displays the complete list of modules configured for deployment in the domain.

Redeploying or Stopping a Module

To redeploy or stop a module that is already deployed in the domain, you reference the module's deployment name rather than the actual archive file or exploded directory. When redeploying a module, target servers first stop the module and then deploy it using the available deployment files (local copies of the files, if the module was deployed in stage mode, or the original deployment files for nostage mode).

If you stop a deployed module, you can later redeploy it using the available deployment files and deployment name; you do not need to reselect the deployment files, as they remain associated with the deployment name in the domain.

Administration Console Tasks

To redeploy or stop a module using the Administration Console:

1. Start the Administration Console for your domain.
2. Expand the Deployments node in the left pane to display the different deployment types.
3. In the left pane, select the type of application or module that you want to deploy. The available deployment types are:
   - Applications—Enterprise Applications or Web Services packaged as `.ear` files or directories
   - EJB Modules—Enterprise JavaBeans
   - Web Application Modules—Web Applications or Web Services packaged as `.war` files or directories
   - Connector Modules—Resource adaptors
   - Startup & Shutdown—Startup classes and Shutdown Classes
4. In the right pane, select the name of the module you want to redeploy or stop. This displays a table with the deployment status of the module on all target servers. If you selected an Enterprise Application, the pane displays a deployment status table for each module that makes up the application.
5. To redeploy or stop a module on a single target server, click the Redeploy or Stop button to the right of the target server name. To redeploy or stop a module on all target servers, click the Redeploy All or Stop All button at the bottom of the deployment status table.

Weblogic.Deployer Tasks

To redeploy a module using the `weblogic.Deployer` utility, use the redeploy command and specify the module’s deployment name. The utility uses a different command form if you want to redeploy individual modules of an Enterprise Application.

For example, to redeploy a single module on all available target servers:

```
java weblogic.Deployer -adminurl http://localhost:7001 -user
weblogic -password weblogic -name mymodule -redeploy
```

To redeploy a single module on a subset of the target servers, specify the target server list:

```
java weblogic.Deployer -adminurl http://localhost:7001 -user
weblogic -password weblogic -name mymodule -redeploy -targets
myserver1,myserver2
```

To redeploy a subset of the modules of an Enterprise Application, specify *modulename@servername* in the target server list. For example:

```
java weblogic.Deployer -adminurl http://localhost:7001 -user
weblogic -password weblogic -name mymodule -redeploy -targets
mymodule1@myserver1,mymodule2@myserver2
```

Redeploying Static Files in a Web Application

In a production environment, you may occasionally need to refresh the static content of a Web Application—HTML files, Image files, and so forth—without redeploying the entire application. You can use the weblogic.Deployer utility to notify the server that static files have changed.

weblogic.Deployer Tasks

To redeploy static files associated with a deployed module, specify the file names at the end of the redeploy command. For example:

```
java weblogic.Deployer -adminurl http://localhost:7001 -user
weblogic -password weblogic -name mywebapp -redeploy
mywebapp/copyright.html
```

Always specify the pathname to updated files relative by starting at the top level of the exploded archive directory. In the above example, the Web Application resides in an exploded archive directory named `mywebapp`.

jspRefresh Tasks

While `weblogic.Deployer` can refresh static files in your deployed applications, the command-line syntax for invoking earlier refresh tools remains viable. If you have scripts that invoke the `WebAppComponentRefreshTool` or weblogic.jspRefresh, they will now invoke weblogic.Deployer's refresh capability.

Use `jspRefresh` to refresh deployed static files such as:

- JSPs
- HTML files
- Image files such as gif and jpg
- Text files

You cannot use this utility to refresh Java class files.

To use `jspRefresh`, *you must deploy the Web Application in exploded directory format*. The utility does not work for components archived in WAR files.

To refresh a static file:

1. Set up your development environment so that WebLogic Server classes are in your system `CLASSPATH` and the JDK is available. You can use the `setEnv` script located in the `config/`*`mydomain`* directory to set your environment.

2. Enter the following command:

   ```
   % java weblogic.deploy -url adminServerURL -username
   AdminUserName -jspRefreshFiles fileList
   -jspRefreshComponentName component refresh password application
   ```

   Where:

   - *`url`* is the URL of your WebLogic Administration Server.
   - *`AdminUserName`* is the username for system administration.
   - *`fileList`* is a comma-separated list of files to be refreshed. Wildcard characters (`*.jsp`, for example) are not supported.
   - *`component`* is the name of the Web Application being refreshed.
   - *`password`* is your system administration password.

- *`application`* is the name of an Enterprise Application that contains the Web Application being refreshed. If your Web Application is not part of an Enterprise Application, enter the name of the Web Application.

For example, the following command refreshes the files `HelloWorld.jsp` and `ball.gif` in the `myWebApp` Web Application:

```
java weblogic.deploy -url t3://localhost:7001
 -username myUsername -jspRefreshFiles HelloWorld.jsp,ball.gif
 -jspRefreshComponentName myWebApp refresh myPassword myWebApp
```

Note: Even though the syntax of the command says `-jspRefreshFiles` and `-jspRefreshComponentName`, you can refresh any static file using this command, not just JSP files.

Changing the Target List for a Deployed Module

After you have deployed a module in a WebLogic Server domain, you can change the module's target server list to add new WebLogic Server instances or to remove existing server instances. If you remove a target server, the module is immediately stopped and removed from the server. If you add a new target server, you must explicitly deploy the module on the new server before it is active.

Administration Console Tasks

1. Start the Administration Console for your domain.
2. Expand the Deployments node in the left pane to display the different deployment types.
3. In the left pane, select the type of application or module that you want to deploy. The available deployment types are:
   - Applications—Enterprise Applications or Web Services packaged as `.ear` files or directories

- EJB Modules—Enterprise JavaBeans
- Web Application Modules—Web Applications or Web Services packaged as `.war` files or directories
- Connector Modules—Resource adaptors
- Startup & Shutdown—Startup classes and Shutdown Classes

4. In the right pane, select the name of the module you want to edit. This displays a table with the deployment status of the module on all target servers.
5. Click the Targets tab in the right pane to show the current target servers for the module. If you selected an Enterprise Application, the pane displays a target list for each module that makes up the application.
6. Use the checkboxes next to each server name to add or remove servers from a module's target list. Click Apply to apply your changes.

Weblogic.Deployer Tasks

To add a new server to the target list using `weblogic.Deployer`, simply add the new server name to the target command-line option. For example:

```
java weblogic.Deployer -adminurl http://localhost:7001 -user
weblogic -password weblogic -name mymodule -deploy -targets
newserver
```

Changing a Server Staging Mode or Staging Directory

The server staging mode specifies the default deployment mode for a server if none is specified at deployment time. For example, the server staging mode is used if you deploy an application or module using `weblogic.Deployer` and you do not specify a staging mode. See "Deployment Staging Modes" on page 3-1 for help on when to use staging modes.

Notes: You can only change the server staging mode by using the Administration Console or by directly changing the `ServerMBean` via JMX.

You cannot change the application staging mode without first deleting the application deployment and redeploying with the new mode.

Administration Console Tasks

To set the server staging mode:

1. Expand the Servers node in the left pane.
2. Select the name of the server instance that you want to configure.
3. Select the Configuration->Deployment tab in the right pane to display the current staging mode.
4. Select stage, nostage, or external_stage from the Staging Mode menu. These modes correspond to the staging modes described in "Deployment Staging Modes" on page 3-1, and apply only to the selected server instance.
5. Enter a path in the Staging Directory Name attribute to store staged deployment files. The path is relative to the root directory of the selected server.
6. If you are configuring the staging mode for the Administration Server, also specify an Upload Directory Name, relative to the server's root directory. This is the directory where the Administration Server stores uploaded files for deployment to servers and clusters in the domain.
7. Click Apply to change the staging mode and directory.

Removing Files from a Deployment

If you deploy a Web Application using an exploded archive directory, you can update static contents of the Web Application either by refreshing the files (see “Redeploying Static Files in a Web Application” on page 4-12), or by deleting files from the deployment. To delete files, you must use the `weblogic.Deployer` utility with the `delete_files` option. For example:

```
java weblogic.Deployer -adminurl http://localhost:7001 -user
weblogic -password weblogic -name mywebapp -delete_files
mywebapp/copyright.html
```

Always specify the pathname to updated files relative by starting at the top level of the exploded archive directory. In the above example, the Web Application resides in an exploded archive directory named `mywebapp`.

Deleting a Deployment from the Domain

After you deploy a new module to servers in a domain, the deployment name remains associated with the deployment files you selected. Even after stopping the deployment on all servers, the files remain available for redeployment using either the Administration Console or `weblogic.Deployer` utility.

If you want to remove a deployment name and its associated deployment files from the domain, you must explicitly delete the deployment from the domain. If you need to redeploy a module after deleting it, you must identify the deployment files, staging mode, and module name using the instructions in “Deploying a New Application Module to Server(s) on One Machine” on page 4-3 or “Deploying a New Application Module to Servers on Multiple Machines” on page 4-5.

Administration Console Tasks

To delete a module and its deployment name from the domain:

1. Expand the Deployments node in the left pane to display the different deployment types.
2. In the left pane, select the type of application or module that you want to deploy, redeploy, or stop.
3. In the left pane, right-click the name of the deployment you want to remove, and select Delete *module_name*... from the menu.
4. In the right pane, select Yes to remove the application or module.

Weblogic.Deployer Tasks

To delete a module and its deployment name from the domain using `weblogic.Deployer`, specify the `undeploy` option. For example:

```
java weblogic.Deployer -adminurl http://localhost:7001 -user
weblogic -password weblogic -name mymodule -undeploy
```

5 Deployment Tools Reference

This chapter provides a reference to different tools used in deploying modules to WebLogic Server. It includes the following sections:

- "weblogic.Deployer Utility" on page 5-1
- "WebLogic Builder" on page 5-7
- "Deployment Management API" on page 5-8

weblogic.Deployer Utility

The `weblogic.Deployer` utility is new in WebLogic Server 7.0 and replaces the earlier `weblogic.deploy` utility, which has been deprecated. The `weblogic.Deployer` utility is a Java-based deployment tool that provides a command-line interface to the WebLogic Server deployment API. This utility was developed for administrators and developers who need to initiate deployment from the command line, a shell script, or any automated environment other than Java.

This section describes how to use the `weblogic.Deployer` utility to perform the following tasks:

- Deploying a New Application
- Redeploying an Entire Application
- Deploying a Module Newly Added to an EAR

Deploying Using weblogic.Deployer Utility

To deploy an application or its components using the `weblogic.Deployer` utility:

1. Set up your local environment so that WebLogic Server classes are in your system `CLASSPATH` and the JDK is available. You can use the `setenv` script located in your server's `/bin` directory to set the `CLASSPATH`.
2. Use the following command syntax:

```
% java weblogic.Deployer [options]
[-activate|-deactivate|-remove|-cancel|-list] [files]
```

You can also list the specific `-files` in the archive that are to be deployed (or redeployed, or undeployed, or unprepared, or deactivated, or removed). The file list can include file names and directories relative to the root of the application. If you specify a directory, its entire subtree is deployed or redeployed.

weblogic.Deployer Actions and Options

Table 5-1 weblogic.Deployer Actions

Action	Description
activate	Deploys or redeploys the application specified by `-name` to the servers specified by `-targets`.
cancel	Attempts to cancel the task identified by `-id` if it is not yet completed.
deactivate	Deactivates the application on the target servers. Deactivation suspends the deployed components, leaving staged data in place in anticipation of subsequent reactivation. This command only works in the two-phase deployment protocol.
delete_files	Removes files specified in the file list and leaves the application activated. This is valid only for unarchived applications. You must specify target servers.
deploy	A convenient alias for `-activate`.
examples	Displays example usages of the tool.
help	Prints a help message.
list	Lists the status of the task identified by `-id`.
remove	Physically removes the application and any staged data from the target servers. The components are deactivated and the targets are removed from the applications configuration. If you remove the application entirely, the associated MBeans are also deleted from the system configuration. This command only works with the two-phase deployment model.
undeploy	A convenient alias for `-unprepare`.
unprepare	Deactivates and unloads classes for the application identified by `-name` on the target servers, leaving the staged application files in a state where they may be edited or quickly reloaded.
upload	Transfers the specified source file(s) to the administration server. Use this option when you are on a remote system and want to deploy an application that resides on the remote system. The application files are uploaded to the WebLogic Server administration server prior to distribution to named target servers.

Action	Description
version	Prints version information.

`weblogic.Deployer` options include:

Table 5-2 weblogic.Deployer Options

Option	Description
adminurl	`https://<server>:<port>` is the URL of the administration server. Default is `http://localhost:7001`.
debug	Turns on debug messages in the output log.
external_stage	Indicates that user wants to copy the application to the servers' staging area externally on their own, or using a third-party tool. When specified, WLS looks for the application under `StagingDirectoryName (of target server)/applicationName`.
id	The task identifier `-id` is a unique identifier for the deployment task. You can specify an `-id` with the `-activate`, `-deactivate`, or `-remove` commands, and use it later as an argument to `-cancel` or `-list`. Make sure the `-id` is unique from all other existing deployment tasks. The system generates an `-id` if you do not specify one.
name	The application `-name` specifies the name of the application being deployed. This can be the name of an existing, configured application or the name to use when creating a new configuration.
nostage	Does not stage the application; instead, deploys it from its current location which you specify using the `-source` option. Defaults: `nostage` for admin server and `stage` for managed server targets.
nowait	Once the action is initiated, the tool prints the task id and exits. This is used to initiate multiple tasks and then monitor them later using the `-list` action.

Option	Description
password	Specifies the password on the command line. If you do not provide a password, you will be prompted for one.
remote	Signals that `weblogic.Deployer` is not running on the same machine as the administration server and that the source path should be passed through unchanged because it represents the path on the remote server.
source	Specifies the location of the archive, file or directory to be deployed. Use this option to set the application Path. The `source` option should reference the root directory or archive being deployed. If you are using it with the `upload` command, the source path is relative to the current directory. Otherwise, it is relative to the administration server root directory—the directory where the `config.xml` file resides.
stage	Indicates that application needs to be copied into the target servers staging area before deployment. Defaults: `nostage` for admin server, and `stage` for managed server targets. Sets the `stagingMethod` attribute on the application when it is created so that the application will always be staged. This value overrides the `stagingMethod` attribute on any targeted servers.
targets	Displays a comma-separated list of the targeted server and/or cluster names (`<server 1>,...<component>@<server N>`). Each target may be qualified with a J2EE component name. This enables different components of the archive to deployed on different servers. Default: For an application which is currently deployed, the default is all current targets. For a new application, it is deployed to the administration server, by default.
timeout	Seconds. Specifies the maximum time in seconds to wait for the completion of the deployment task. When the time expires, `weblogic.Deployer` prints out the current status of the deployment and exits.
user	User name.
verbose	Displays additional progress messages.

Example Uses of the weblogic.Deployer Utility

Below are example usages of the `weblogic.Deployer` utility.

Deploying a New Application

```
java weblogic.Deployer -adminurl http://admin:7001 -name app
-source /myapp/app.ear -targets server1,server2 -activate
```

Redeploying an Entire Application

```
java weblogic.Deployer -source /myapp/app.ear -adminurl
http://admin:7001 -name app -activate
```

Note: If you don't provide `-source`, or a list of updated files, this operation will have no effect because the system will assume that nothing has been changed in the application.

Deploying a Module Newly Added to an EAR

If you have added the module `newmodule.war` to the deployed application `myapp.ear` and updated the module in the `application.xml` file, you can deploy `newmodule.war` in `myapp.ear` using the following:

```
java weblogic.Deployer -username myname -password mypassword
-name myapp.ear -activate -targets newmodule.war@myserver
-source /myapp/myapp.ear
```

Note that this command will deploy the new module without redeploying the other modules in the application.

Redeploying Part of an Exploded Web Application (Refresh)

```
java weblogic.Deployer -adminurl http://admin:7001 -name app
-activate jsps/login.jsp
```

where `jsps` is a directory in the top level of the exploded web application. Note that partial redeployment is only supported on exploded `WAR` files. The path is relative to the root of the application as originally deployed.

Deactivating an Application on All Active Targets, Making It Unavailable

```
java weblogic.Deployer -adminurl http://admin:7001 -name app
-deactivate
```

Reactivating a Deactivated Application

```
 java weblogic.Deployer -adminurl http://7001 -name app
-activate
```

Removing an Application from All Targeted Servers

```
java weblogic.Deployer -adminurl http://admin:7001 -name app
-targets server -remove
```

Cancelling a Deployment Task

```
java weblogic.Deployer -adminurl http://admin:7001 -cancel -id
tag
```

Listing All Deployment Tasks

```
java weblogic.Deployer -adminurl http://admin:7001 -list
```

Deploying or Redeploying an Application to a Single Server

```
java weblogic.Deployer -activate -name ArchivedEarJar -source
C:/MyApps/JarEar.ear –target server1
```

Deploying an Application to an Additional Server

```
java weblogic.Deployer -activate -name ArchivedEarJar –target
server2
```

WebLogic Builder

WebLogic Builder is a WebLogic Server tool for generating and editing deployment descriptors for J2EE applications. It can also deploy applications to single servers.

See WebLogic Builder.

Deployment Management API

A deployment task is initiated through a `DeployerRuntimeMBean`—a singleton (an object for which only one instance exists) that resides on a WebLogic Administration Server. `DeployerRuntimeMBean` provides methods for activating, deactivating, and removing an application. These methods return a `DeploymentTaskRuntimeMBean` that encapsulates the request and provides the means for tracking its progress. `DeploymentTaskRuntimeMBean` provides ongoing status of the request through TargetStatus objects, one per target.

The WebLogic Server deployment management API is defined by the following WebLogic Server MBeans:

- `DeployerRuntimeMBean`—programmatic interface to deployment requests. Deployment requests provided through the `DeployerRuntimeMBean` manifest the configuration state into the application and appropriate component configuration MBeans. These MBeans persist the deployment state of applications in the WebLogic Server domain.
- `DeploymentTaskRuntimeMBean`—interface for encompassing deployment tasks.

The deployment management API is asynchronous. The client must poll the status or utilize `ApplicationMBean` notifications to determine when the task is complete.

For more information about WebLogic Server deployment management APIs, see the `weblogic.management.deploy` Javadoc.