BEA WebLogic Server™

BEA WebLogic Server Frequently Asked Questions

Version 8.1
June 28, 2006

Copyright

Restricted Rights Legend

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

Trademarks or Service Marks

Contents

About This Document

About This Document

This document introduces BEA WebLogic Server™ Frequently Asked Questions.

Note: Some of the information in this document may be inconsistent with what you observe in the WebLogic Server 7.0 Beta release.

The document is organized as follows:

- Chapter 1, "FAQs: Administration and Configuration," contains questions and answers about administering and configuring WebLogic Server.
- Chapter 2, "FAQs: Applets," contains questions and answers about applets.
- Chapter 3, "FAQs: Clustering," contains questions and answers about clustering.
- Chapter 4, "FAQs: Code Examples," contains questions and answers about code examples.
- Chapter 6, "FAQs: EJB," contains questions and answers about EJB.
- Chapter 7, "FAQs: Installation," contains questions and answers about installing WebLogic Server.
- Chapter 9, "FAQs: Java," contains questions and answers about Java.
- Chapter 10, "FAQs: J2EE Connector Architecture," contains questions and answers about J2EE Connector Architecture.
- Chapter 11, "FAQs: WebLogic JDBC," contains questions and answers about JDBC.

- Chapter 12, “FAQs: WebLogic jDriver for MSSQL Server,” contains questions and answers about WebLogic jDriver for MSSQL Server.
- Chapter 13, “FAQs: WebLogic jDriver for Oracle,” contains questions and answers about WebLogic jDriver for Oracle.
- Chapter 14, “FAQs: JMS,” contains questions and answers about JMS.
- Chapter 16, “FAQs: JTA,” contains questions and answers about JTA.
- Chapter 18, “FAQs: Server-Related Questions,” contains general questions and answers about WebLogic Server.
- Chapter 19, “FAQs: Server-Side Java (Servlets),” contains questions and answers about servlets.
- Chapter 20, “FAQs: Security,” contains questions and answers about security.
- Chapter 23, “FAQs: Wireless-Related Questions,” contains questions and answers about wireless technology.
- Chapter 24, “FAQs: XML,” contains questions and answers about XML.

Audience

This document is written for application developers who want to build e-commerce applications using the Java 2 Platform, Enterprise Edition (J2EE) from Sun Microsystems. It is assumed that readers know Web technologies, object-oriented programming techniques, and the Java programming language.

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.

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.

`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 you enter from the keyboard.
	Examples:
	`import java.util.Enumeration;`
	`chmod u+w *`
	`config/examples/applications`
	`.java`
	`config.xml`
	`float`

`monospace italic text`	Variables in code. *Example*: `String` *`CustomerName`*`;`
UPPERCASE TEXT	Device names, environment variables, and logical operators. *Example*s: 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}`

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

CHAPTER 1

FAQs: Administration and Configuration

- How do I provide user credentials for starting a server?
- Can I start a Managed Server if the Administration Server is unavailable?
- What is the easiest way to set the classpath?
- How do I edit the `config.xml` file?
- Is there a quick way to create and start a remote Managed Server?
- The Tree View pane of the WebLogic Console is not visible in my browser. How do I enable it?

Q. How do I provide user credentials for starting a server?

A. When you create a domain, the Configuration Wizard prompts you to provide the username and password for an initial administrative user. If you create the domain in development mode, the wizard saves the username and encrypted password in a boot identity file. A WebLogic Server instance can refer to a boot identity file during its startup process. If a server instance does not find such a file, it prompts you to enter credentials.

If you create a domain in production mode, or if you want to change user credentials in an existing boot identity file, you can create a new boot identity file. WebLogic Server does not support copying a boot identity file from one server root directory to another. For information on creating and using boot identity files, see "Boot Identity Files" in *Administration Console Online Help*.

Q. Can I start a Managed Server if the Administration Server is unavailable?

A. By default, if a Managed Server is unable to connect to the specified Administration Server during startup, it can retrieve its configuration by reading a configuration file and other files directly. You cannot change the server's configuration until the Administration Server is available. A Managed Server that starts in this way is running in Managed Server Independence mode. For more information, see "Starting a Managed Server When the Administration Server Is Not Accessible" in *Configuring and Managing WebLogic Server*.

Q. What is the easiest way to set the classpath?

WebLogic Server installs the following script that you can use to set the classpath that a server requires:

`WL_HOME\server\bin\setWLSEnv.cmd` (on Windows)

`WL_HOME/server/bin/setWLSEnv.sh` (on UNIX)

where `WL_HOME` is the directory in which you installed WebLogic Server. For more information, see "Setting the Classpath" in the *WebLogic Server Command Reference*.

Q. How do I edit the `config.xml` file?

A. The persistent configuration for a domain of WebLogic Servers and clusters is stored in an XML configuration file (`config.xml`). You can modify this file in the following ways:

- Use the Administration Console. See "Using the Administration Console" in the *Administration Console Online Help*.
- If you want to create scripts that automate domain management, use the `weblogic.Admin` utility. See "weblogic.Admin Command-Line Reference" in the *WebLogic Server Command Reference* .
- If you want to create Java-based management applications, use the Java Management Extensions (JMX) Application Programming Interface (API). See the Programming WebLogic Management Services with JMX guide.
- If you want to edit the `config.xml` file directly (not recommended), see the BEA WebLogic Server Configuration Reference.

Q. Is there a quick way to create and start a remote Managed Server?

A. The recommended approach is to use the Domain Configuration Wizard, as described in "Setting Up and Starting Managed Servers on a Remote Machine" in *Creating WebLogic Configurations Using the Configuration Wizard* at `http://e-docs.bea.com/platform/docs81/confgwiz/multi.html`.

For a streamlined approach, follow the instructions at “Starting Managed Servers From a WebLogic Server Script” in the *Administration Console Online Help*.

Q. The Tree View pane of the WebLogic Console is not visible in my browser. How do I enable it?

A. Enable the Sun Java Plug-In from the control panel.

FAQs: Applets

Q. What are the alternatives to using applets?

A. BEA supports the use of server-side applications with HTTP servlets and Java Server Pages (JSPs) as part of the J2EE platform. We recommend that before you develop new applications, consider using either servlets or JSPs. A well-designed series of interactive Web pages using servlets and Java Server Pages (JSPs) generally yield a faster and more reliable Web site.If you are currently using applets, you may find that most can be converted to Java applications using Java Web Start and you can continue using WebLogic Server. For information, go to Sun's Java Web Start site.

For more information, see Using Applets With WebLogic Server at http://e-docs.bea.com/wls/docs81/applets/usingapplets.html.

Q. Can I use a "native" two-tier driver for a browser applet?

A. No. Within an unsigned applet, you cannot load native libraries over the wire, access the local file system, or connect to any host except the host from which you loaded the applet. The applet security manager enforces these restrictions on applets as protection against applets being able to do unsavory things to unsuspecting users.

If you are trying to use jDriver for Oracle from an applet, then you are violating the first restriction. Your applet will fail when it attempts to load the native (non-Java layer) library that allows jDriver for Oracle to make calls into the non-Java Oracle client libraries. If you look at the exception that is generated, you will see that your applet fails in `java.lang.System.loadLibrary`, because the security manager determined that you were attempting to load a local library and halted the applet.

You can, however, use the WebLogic JTS or Pool driver for JDBC connectivity in applets. When you use one of these WebLogic multitier JDBC drivers, you need one copy of WebLogic jDriver for Oracle (or any other two-tier JDBC driver) for the connection between the WebLogic Server and the DBMS.

Q. Why doesn't my browser applet connect to the database?

Problem: I'm using a WebLogic multitier driver in an applet as an interface to a DBMS. If I run the class using the Sun Appletviewer on my local machine, I have no problems. But when I try to run the applet in a Netscape browser, it will not connect.

A. If Appletviewer works and Netscape does not, it is an indication that you are violating a Netscape security restriction. In this case, the violation is that an applet cannot open a socket to a machine other than the one from which it loaded the applet. To solve this problem, you will have to serve your applet code from the same machine that hosts the DBMS.

In addition, the IP naming format you use in the applet `CODEBASE` and the constructor for the T3Client must match. That is, you can't use dot-notation in one place and a domain name in the other.

Q. Why does my applet work with Appletviewer, but not with a browser?

Problem: I tried to run two of the applets in the examples directory of the distribution. I installed the WebLogic classes on my local machine (NT server) and on another machine (a Windows 95 client). I am not using any browsers, just trying to run the applets with Appletviewer. The applets work fine when I run Appletviewer from the NT server, but do not work at all from the Windows 95 client.

A. There are two possible problems: Either the `CODEBASE` tag is not properly set in the applet HTML file, or the class files are not properly loaded on the HTTP server.

The applet works on the NT server because you installed the WebLogic distribution on your NT server. Even if the applet cannot successfully load the necessary classes from the HTTP server, it *does* find them in your local CLASSPATH. But when you try to run it from the Windows 95 client, the applet must load the classes over the wire from the HTTP server, and if you haven't installed them correctly, it will fail.

Q. What's causing ClassFormatErrors with my applet?

Problem: I downloaded your distribution and copied the classes to my HTTP server DocumentRoot. I created an applet that I ran successfully from my Netscape server. I placed it in the server directory /webz/ns-home/classes/applets/myapp.class and called it with the following:

```
<APPLET
  CODEBASE=http://myserver.com/webz/ns-home/classes
  CODE=applets.myapp.class>
```

Then I set my attributes in the Administration Console to listen on port 7001, and I started the WebLogic Server on the HTTP machine so I could use my applet with WebLogic JDBC, like this:

```
<APPLET
        CODEBASE=t3://myserver.com:7001/webz/ns-home/classes
        CODE=applets.myapp.class>
```

When I changed the CODEBASE tag to point to the WebLogic Server, I started getting ClassFormatErrors.

A. There are several problems with your setup. The most obvious have to do with your CODEBASE:

1. The CODEBASE tag in your applet should point to your HTTP server, not to WebLogic Server.
2. The directory path referenced in your CODEBASE tag is not an absolute directory path on the HTTP server; it is a configured path that originates from the HTTP Document Root. You are using the absolute path in your CODEBASE tag. If your class "myapp" is in the "applets" package, then the correct CODEBASE for your setup would be:

```
<APPLET
         CODEBASE=http://myserver.com/classes
         CODE=applets.myapp.class>
```

In addition, if you are getting a ClassFormatError, it signals a problem with your HTTP server configuration. It could be that you haven't loaded the WebLogic or applet classes in the correct directory on the HTTP server, or you are specifying the CODEBASE or the CODE incorrectly in your APPLET tag.

Remember, too, that if you installed the WebLogic distribution on the machine from which you are running the applet, your applet will first look for the WebLogic classes in your local `CLASSPATH`, which may obscure the fact that you haven’t properly installed the classes for serving from the HTTP server. To test your HTTP configuration properly, you need to temporarily rename the WebLogic classes in your local `CLASSPATH` or try your applet from another machine.

CHAPTER 3

FAQs: Clustering

- How do stubs work in a WebLogic Server cluster?
- What happens when a failure occurs and the stub cannot connect to a WebLogic Server instance?
- How does a server know when another server is unavailable?
- How are notifications made when a server is added to a cluster?
- How do clients learn about new WebLogic Server instances?
- How do clients handle DNS requests to failed servers?
- How many WebLogic Servers can I have on a multi-cpu machine?
- Should I use a separate network for multicast in a cluster?
- What should I do if my cluster “hangs” or “freezes”?

Q. How do stubs work in a WebLogic Server cluster?

A. Clients that connect to a WebLogic Server cluster and look up a clustered object obtain a replica-aware stub for the object. This stub contains the list of available server instances that host implementations of the object. The stub also contains the load balancing logic for distributing the load among its host servers.

Q. What happens when a failure occurs and the stub cannot connect to a WebLogic Server instance?

A. When the failure occurs, the stub removes the failed server instance from its list. If there are no servers left in its list, the stubb uses DNS again to find a running server and obtain a current list of running instances. Also, the stub periodically refreshes its list of available server instances in the cluster; this allows the stub to take advantage of new servers as they are added to the cluster.

Q. How does a server know when another server is unavailable?

A. WebLogic Server uses two mechanisms to determine if a given server instance is unavailable.

Each WebLogic Server instance in a cluster uses multicast to broadcast regular "heartbeat" messages that advertise its availability. By monitoring heartbeat messages, server instances in a cluster determine when a server instance has failed. The other server instances will drop a server instance from the cluster, if they do not receive three consecutive heartbeats from that server instance

WebLogic Server also monitors socket errors to determine the availability of a server instance. For example, if server instance A has an open socket to server instance B, and the socket unexpectedly closes, server A assumes that server B is offline.

Q. How are notifications made when a server is added to a cluster?

A. The WebLogic Server cluster broadcasts the availability of a new server instance each time a new instance joins the cluster. Cluster-aware stubs also periodically update their list of available server instances.

Q. How do clients learn about new WebLogic Server instances?

A. Once a client has done a JNDI lookup and begins using an object reference, it finds out about new server instances only after the cluster-aware stub has updated its list of available servers.

Q. How do clients handle DNS requests to failed servers?

A. If a server fails and DNS continues to send requests to the unavailable machine, this can waste bandwidth. For a Java client application, this problem occurs only during startup. WebLogic Server caches the DNS entries and removes the unavailable ones, to prevent the client from accessing a failed server twice.

Failed servers can be more of a problem for browser-based clients, because they always use DNS. To avoid unnecessary DNS requests with browser-based clients, use a third-party load-balancer such as Resonate, BigIP, Alteon, and LocalDirector. These products mask multiple DNS addresses as a single address. They also provide more sophisticated load-balancing options than round-robin, and they keep track of failed servers to avoid routing unnecessary requests.

Q. How many WebLogic Servers can I have on a multi-cpu machine?

A. There are many possible configurations and each has its own advantages and disadvantages. BEA WebLogic Server has no built-in limit for the number of server instances that can reside in a cluster. Large, multi-processor servers such as Sun Microsystems, Inc. Sun Enterprise 10000, therefore, can host very large clusters or multiple clusters.

In most cases, WebLogic Server clusters scale best when deployed with one WebLogic Server instance for every two CPUs. However, as with all capacity planning, you should test the actual deployment with your target web applications to determine the optimal number and distribution of server instances. See Performance Considerations for Multi-CPU Machines for additional information.

Q. Should I use a separate network for multicast in a cluster?

A. No. Multicast traffic is not heavy enough to require a separate network.

Q. What should I do if my cluster "hangs" or "freezes"?

A. If your WebLogic Server cluster "freezes," you will need to collect certain diagnostic information, including thread dumps and Java garbage collection metrics, before contacting BEA Technical Support. For details, see Generate a Log File.

FAQs: Code Examples

- Where are the examples located?
- Why won't the examples work?
- Are the build.cmd and the build.sh scripts still being used?
- How do I use ANT?
- Why do I have trouble targeting a PointBase connection pool to a remote server?
- Why do I get an error if I attempt to start the MedRec Server on a system that uses a proxy to connect to the Internet?
- Can I use the PointBase DBMS included with WebLogic Server for development or production?

Q. Where are the examples located?

A. Examples, if installed, are located in the *`BEA_HOME`*`\samples\server\examples\src\examples` directory, where *`BEA_HOME`* refers to the main WebLogic Platform directory. To invoke a Web page that describes the examples, see the Start menu on Windows.

Q. Why won't the examples work?

A. Each example comes with detailed instructions for building the example class files, configuring the server, and running the example. Make sure that you have completed each of the example instructions.

Usually problems with examples are related to your environment. Here are some troubleshooting hints:

1. If you are using a database, make sure you have run the utility `utils.dbping` to verify that your JDBC driver is correctly installed and configured.
2. Run the setEnv script to make sure your CLASSPATH is correctly set in the shell or DOS window in which you are running the examples.
3. Check the instructions for the examples to make sure you have changed any user-specific variables in the code before compiling.
4. Verify that you are compiling with the -d option to direct the class files into the proper directory, as defined in the example instructions.

If the example is an applet, check the CODE and CODEBASE, and make sure WebLogic Server is running.

For more information, see the WebLogic Server Examples Guide located at *`BEA_HOME`*`\samples\server\examples\src\examples\examples.html`, where *`BEA_HOME`* refers to the main WebLogic Platform directory.

Q. Are the build.cmd and the build.sh scripts still being used?

A. No. They have been replaced by ANT.

Q. How do I use ANT?

A. Set up the Examples domain environment by running `setExamplesEnv.cmd` in Windows or `setExamplesEnv.sh` in UNIX. Go into your examples directory and type ANT to build the `build.xml` file. If you want to build your own build script, you can pass in the name of the build script by typing `ANT -f` *`myBuild`*`.xml` where *`myBuild`* is the name of your build script. For more information, see the Apache website.

Q. Why do I have trouble targeting a PointBase connection pool to a remote server?

A. Open the `commEnv.cmd` or `.sh` file in *`WL_HOME`*`\common\bin` and edit the POINTBASE_CLASSPATH entry by removing the bolded section of the entry shown below:

```
set
POINTBASE_CLASSPATH=%POINTBASE_HOME%\lib\pbserver44.jar;%POINTBASE_HOME%\l
ib\pbclient44.jar
```

Q. Why do I get an error if I attempt to start the MedRec Server on a system that uses a proxy to connect to the Internet?

A. The MedRec Server uses the Struts framework and its attempts to access `jakarta.apache.org` on startup result in error messages. The server starts up successfully, but with error messages. Adding the following options to MedRec Server's start script resolves the issue:

```
-Dhttp.proxyHost=<Your Proxy Host>

-Dhttp.proxyPort=<Proxy Port>

-Dhttp.nonProxyHosts=<Your Subnet>|localhost
```

Q. Can I use the PointBase DBMS included with WebLogic Server for development or production?

A. PointBase Server is an all-Java DBMS product included in the WebLogic Server distribution soley in support of WebLogic Server evaluation, either in the form of custom trial applications or through packaged sample applications provided with WebLogic Server. Non-evaluation development and/or production use of the PointBase Server requires a separate license be obtained by the end user directly from PointBase.

FAQs: Deployment

Q. How can I set deployment order for applications?

A. WebLogic Server 8.1 allows you to select the load order for applications. See the ApplicationMBean LoadOrder attribute in Application. WebLogic Server deploys server-level resources (first JDBC and then JMS) before deploying applications. Applications are deployed in this order: connectors, then EJBs, then Web Applications. 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.

Q. Can I refresh static components of a deployed application without having to redeploy the entire application?

A. Yes. You can use `weblogic.Deployer` to specify a component and target a server, using the following syntax:

```
java weblogic.Deployer -adminurl http://admin:7001 -name appname -targets
server1,server2 -deploy jsps/*.jsp
```

Q. When should I use the `-nostage` option?

A. Set the staging mode to `-nostage` (using `weblogic.Deployer` or the Administration Console) if you don't want to copy deployment files but want to deploy an application from its present location. All target servers must be able to access the same set of deployment files.

Q. When should I use the `external_stage` option?

A. Set `-external_stage` using `weblogic.Deployer` if you want to stage the application yourself, and prefer to copy it to its target by your own means.

Q. Can I generate deployment descriptor files automatically?

A. Yes, WebLogic Builder automatically generates deployment descriptor files for your J2EE applications. See WebLogic Builder Online Help.

Q. Can I set the deployment order for application modules? For standalone modules?

A. The Load Order attribute controls the deployment order of standalone modules and applications relative to other modules and applications of the same type. For example, standalone EJBs with smaller Load Order values are deployed before those with higher values.

Modules that are deployed as part of an Enterprise Application (EAR file or directory) are deployed in the order in which they are specified in the `application.xml` deployment descriptor.

Q. What is the difference between the *`WL_HOME`*`/config/examples/applications` folder and the *`WL_HOME`*`/config/examples/stage` folder?

A. The `applications` folder is intended for applications that are not yet ready for a production environment. WebLogic Server dynamically deploys the contents of the `applications` folder. The `stage` folder (or a folder that you create for the same purpose) is for storing copies of deployment files that are ready for deployment in a production environment (deployments that use the `stage` or `external_stage` deployment modes).

Q. How do I turn the auto-deployment feature off?

A. The auto-deployment feature checks the applications folder every three seconds to determine whether there are any new applications or any changes to existing applications and then dynamically deploys these changes.

The auto-deployment feature is enabled for servers that run in development mode. To disable auto-deployment feature, use one of the following methods to place servers in production mode:

- In the Administration Console, click the name of the domain in the left pane, then select the Production Mode checkbox in the right pane.
- At the command line, include the following argument when starting the domain's Administration Server:

  ```
  -Dweblogic.ProductionModeEnabled=true
  ```

Production mode is set for all WebLogic Server instances in a given domain.

Q. Where can I learn more about WebLogic 8.1 deployment?

A. Here: Deployment at http://e-docs.bea.com/wls/docs81/deployment.html.

FAQs: EJB

General

- Why is there no polymorphic-type response from a create() or find() method?
- Must EJBs be homogeneously deployed across a cluster? Why?
- What is the free pool?
- Where can I get a copy of the EJB specification?
- Which versions of the EJB specification are supported by WebLogic Server?

Stateless Session Beans

- How should I set `max-beans-in-free-pool` for stateless beans?
- How should I set `initial-beans-in-free-pool` for stateless beans?
- When are stateless EJBs passivated?
- Can I call remove() on a stateless session bean?
- When are ejbCreate and ejbRemove called on stateless EJBs?

Stateful Session Beans

- Can you explain passivation / activation?
- Why did I get a LockTimedOutException?
- What is the difference between the NRU and LRU cache?

- When should I use a Stateful session bean and when should I use a servlet session?
- How big should I make the cache for a stateful session bean?

General Entity Bean Questions

- Can an entity bean be a listener for JMS messages?

CMP Entity Beans

- When are CMP fields loaded? Is it always determined by the `finders-load-bean` setting? What is the default behavior?
- What's the purpose of the `delay-database-insert-until` deployment descriptor element?
- Can I map an entity bean to more than one table?
- Can I use a join or intermediate table to implement a One-Many relationship?
- Why can't I hold on to a cmr-field (Container Managed Relationship) collection and use it after the transaction commits?
- Can a foreign key column in the database be mapped to both a cmp-field and a cmr-field?
- Why can't I call the setXXX method for a cmr-field during ejbCreate?
- How can I avoid violating NOT NULL constraints on foreign keys that are mapped to cmr-fields?
- Does WebLogic support auto generating primary keys for entity beans?

Message Driven Beans

- Which security principal does an MDB use to connect to JMS?

EJBs and Transactions

- How should I obtain my JDBC connection so that it participates in the EJB container's transaction?
- My transaction aborted, but my database changes did not rollback.
- Why did my JDBC code throw a rollback SQLException?

Q. Why is there no polymorphic-type response from a create() or find() method?

A. The EJB Specification prohibits this behavior, and the weblogic.appc compiler checks for this behavior and prohibits any polymorphic type of response from a `create()` or `find()` method.

The reason the `create()` and `find()` methods are not polymorphic is similar to the reason constructors are not polymorphic in Java. The derived classes generally do not know or cannot initialize the base class properly.

Q. Must EJBs be homogeneously deployed across a cluster? Why?

A. Yes. In WebLogic Server 6.0 and later, EJBs must be homogeneously deployed across a cluster for the following reasons:

- To keep clustering EJBs simple
- To improve performance by avoiding cross-server calls. If EJBs are not deployed on all servers, cross-server calls are more likely.
- To ensure that every EJB is available locall.y
- To ensure that all classes are loaded in an undeployable way. Every server must have access to each EJB's classes so that it can be bound into the local JNDI tree. If only a subset of the servers deploys the bean, the other servers will have to load the bean's classes in their respective system classpaths which makes it impossible to undeploy the beans.

Q. What is the free pool?

A. The free pool is a data structure the EJB container uses to cache anonymous instances of a given bean type. The free pool improves performance by reusing objects and skipping container callbacks when it can.

Q. Where can I get a copy of the EJB specification?

A. You can download it from Sun's EJB site. Point your browser to: http://java.sun.com/products/ejb/index.html.

Q. Which versions of the EJB specification are supported by WebLogic Server?

A. The following table summarizeds EJB specification support by WebLogic Server version:

WebLogic Server Version	EJB Version
4.5.x	1.0
5.1	1.1
6.0	1.1 and 2.0 (PFDI)
6.01	1.1 and 2.0
7.0	1.1 and 2.0
8.1	1.1 and 2.0

Q. How should I set `max-beans-in-free-pool` for stateless beans?

A. This is explained in "Pooling for Stateless Session EJBs" in *Programming WebLogic Enterprise JavaBeans*.

Q. How should I set `initial-beans-in-free-pool` for stateless beans?

A. This is explained in "Pooling for Stateless Session EJBs" in *Programming WebLogic Enterprise JavaBeans*.

Q. When are stateless EJBs passivated?

A. Stateless ejbs are never passivated. Since stateless ejbs do not have state, there is no need to passivate them. They are put back into the free pool after each method call so they will be available to service other requests.

Q. Can I call remove() on a stateless session bean?

A. Yes. Currently, calling remove() on a stateless session bean is a noop.

Q. When are ejbCreate and ejbRemove called on stateless EJBs?

A. When the stateless beans are created and removed by the EJB container. Note that this does not correspond to calls to create and remove on the stateless home. Depending on your `initial-beans-in-free-pool` setting, beans may be created by the container during deployment at which point they are placed in the free pool. Aside from during deployment, beans will only be created to service requests when all of the beans in the free pool are in use and the max-beans-in-free-pool limit has not been met. Stateless beans are removed by the EJB container during undeployment.

Q. Can you explain passivation / activation?

A. Passivation and activation are a standard part of a stateful session bean's lifecycle. For details, see *Stateful Session EJB Life Cycle* in *Programming WebLogic Enterprise JavaBeans.*

Q. Why did I get a LockTimedOutException?

A. When you get a LockTimedOutException while invoking a stateful session EJB, one of two things has occurred:

- You have <allow-concurrent-calls> set to true in your weblogic-ejb-jar.xml descriptor and your call timed out while waiting to be processed. The timeout used in this case is the value <trans-timeout-seconds> element of the weblogic-ejb-jar.xml descriptor or its default value of 30 seconds.
- You do not have <allow-concurrent-calls> set to true and you attempt to invoke a stateful session bean that is already busy processing another request. In this case, the second method call will not block and a LockTimedOutException will be thrown immediately.

Q. What is the difference between the NRU and LRU cache?

A. NRU cache works by avoiding passivation as much as possible. Stateful session instances are only passivated when there is memory pressure (your # of beans in the cache approaches the max-beans-in-cache size). This is the 'NRU' option in the weblogic-ejb-jar.xml and the default behavior. The LRU cache works by passivating a bean after it reached its idle-timeout-seconds. So if your max-beans-in-cache was 1000 and you only had 10 beans in memory, it would still write the 10 beans to disk after their timeout period expired. This is the 'LRU' option in weblogic-ejb-jar.xml. This was added to 5.1 and 6.x because there were some customers that wrote applications depending on the timeout behavior. The was also the default behavior in 3.1-4.5.

Q. When should I use a Stateful session bean and when should I use a servlet session?

A. The answer to this question is very application-specific and there are situations in which either approach will work. A stateful session bean provides declaritive transaction and security checking as well as failover to a secondary in a cluster.

Q. How big should I make the cache for a stateful session bean?

A. The cache for a stateful session bean should usually be equal to the maximum number of concurrent clients of the bean. This is generally a much larger value than the number of execute threads in the server, therefore, stateful session beans will use more server resources.

Q. Can an entity bean be a listener for JMS messages?

A. No. Message driven beans should be used to consume JMS messages.

Q. When are CMP fields loaded? Is it always determined by the `finders-load-bean` setting? What is the default behavior?

A. Finders-load-bean is true by default. A bean is retrieved via a finder explicitly if you call a findXXX() method to get the bean and implicitly if you retrieve the bean via a cmr-field getXXX method. Either of these will result in eager loading of the bean's fields if finders-load-bean is true for the bean.

If you aren't calling a finder, but just accessing the bean via a reference you obtained in a different transaction, then fields are always loaded lazily when using 2.0 CMP, i.e. they aren't read from the DBMS during ejbLoad, but rather when a getXXX() method is called. By default there is a single field group that contains all the fields if you don't define any field groups. So, calling a cmp-field getXXX() method loads all of the persistent state of the bean by default.

Q. What's the purpose of the `delay-database-insert-until` deployment descriptor element?

A. This setting allows you to specify at what point the database insert occurs when creating a CMP bean. Non-null foreign key constaints in your database can make creating CMP beans with relationships tricky. We offer this setting to give developers more flexibility to satisfy those constraints. Also, this setting is necessary to enable the bulk insert feature that was added in WLS 7.0. To enable the bulk insert feature and to provide yourself the maximum amount of flexibilty to satisfy any non-null foreign key constraints, we recommend setting the delay-database-insert-until option to commit. For more information, see Delaying Database Inserts in *Programming WebLogic Enterprise JavaBeans.*

Q. Can I map an entity bean to more than one table?

A. In WLS 8.1 it is possible to map an entity bean to multiple tables, however, there are some restrictions. Namely, each table must contain the same primary key columns. The columns can have different names in different tables, but they must contain the same values. When a new bean is created a row will be inserted in each table, and when a bean is removed a row is removed from each table. The rows in each table are related via their primary key values. The rows mapped to a particular entity bean will always have the same primary key value. See "Multiple Table Mapping" in *Programming WebLogic Enterprise JavaBeans*.

Q. Can I use a join or intermediate table to implement a One-Many relationship?

A. This is currently not supported. One-Many relationships require that the bean on the Many side of the relationship contain a foreign-key in one of its tables.

Q. Why can't I hold on to a cmr-field (Container Managed Relationship) collection and use it after the transaction commits?

A. This is prohibited by the EJB 2.0 specification. The reason for disallowing this is that the DBMS data backing a cmr-field collection can change in unpredictable ways once your transaction commits. Tracking these changes is difficult and can result in the contents of the cmr-field collection changing when the application doesn't expect it. The bottom line is that developers must retrieve and access cmr-field collections within a single transaction.

Q. Can a foreign key column in the database be mapped to both a cmp-field and a cmr-field?

A. Yes, this has been supported since WLS 6.0 SP1. Note that when the cmp-field is a primary-key field the cmr-field is read-only. In other words, the setXXX method for the cmr-field cannot be used. The value of the primary-key should be initialized as usual in this case. Conversely, when the cmp-field is not a primary-key field, then the cmp-field is read-only. The underlying column is updated via the cmr-field, and the cmp-field just provides a read-only view of the foreign-key.

Q. Why can't I call the setXXX method for a cmr-field during ejbCreate?

A. This is disallowed by the EJB 2.0 specification because the primary-key of the current bean isn't necessarily known during ejbCreate and it may be needed depending on how the relationship is mapped to the underlying DBMS. Cmr-field setters should be called during ejbPostCreate instead of ejbCreate if relationships need to be setup during creation of a bean.

Q. How can I avoid violating NOT NULL constraints on foreign keys that are mapped to cmr-fields?

A. In WLS 7.0 and later, you can set `delay-database-insert-until` to 'commit' and assign the cmr-field a value before the current transaction commits. You can also set `delay-database-insert-until` to 'ejbPostCreate' and assign the cmr-field a value during ejbPostCreate.

Q. Does WebLogic support auto generating primary keys for entity beans?

A. Yes, this feature was added in WLS 6.1. For more information, see the DTD comments for the <automatic-key-generation> element. Point your browser to:

http://www.bea.com/servers/wls810/dtd/weblogic-ejb-jar.dtd.

Q. Which security principal does an MDB use to connect to JMS?

A. As of WLS 6.1 SP2, an MDB uses the same principle to connect to JMS as it does to process messages. This is either the principle that is mapped to the run-as role specified for the bean, or 'guest' if no run-as role is provided. Prior to WLS 6.1 SP2, this behavior was not well defined.

Q. How should I obtain my JDBC connection so that it participates in the EJB container's transaction?

A. You must get your JDBC connection from a TxDataSource or from the JTS driver. If you get a JDBC connections from a plain DataSource or directly from the JDBC driver, it will NOT participate in the EJB container transaction.

TxDataSources are recommended instead of accessing the JTS driver directly. TxDataSources are the standard mechanism in JDBC 2.0 for connection pooling, and they support 2PC/XA transactions.

Q. My transaction aborted, but my database changes did not rollback.

A. See previous question. You must obtain your JDBC connections from a TxDataSource.

Q. Why did my JDBC code throw a rollback SQLException?

A. Your JDBC code may throw the following exception:

```
"The coordinator has rolled back the transaction.
     No further JDBC access is allowed within this transaction."
```

The WebLogic JTS JDBC driver throws this exception when the current JDBC connection transaction rolls back prior to or during the JDBC call. This exception indicates that the transaction in which the JDBC connection was participating was rolled back at some point prior to or during the JDBC call.

The rollback may have happened in an earlier EJB invoke that was part of the transaction, or the rollback may have occurred because the transaction timed out. In either case, the transaction will be rolled back, the connection returned to the pool and the database resources released. In order to proceed, the JTS JDBC connection must be closed and reopened in a new transaction.

FAQs: Installation

- Which platforms are available for WebLogic Server?
- I downloaded the WebLogic Server installation file, but the installation program will not run. What should I do?
- Do I need to install WebLogic Server as root on Solaris?
- Can I run the Configuration Wizard outside the installer?
- I love the new domain configuration wizard. Can I create my own templates?

Q. Which platforms are available for WebLogic Server?

A. See the BEA WebLogic Server Supported Configurations page at `http://e-docs.bea.com/platform/suppconfigs/index.html`.

Q. I downloaded the WebLogic Server installation file, but the installation program will not run. What should I do?

A. The installation file may have been corrupted during the download. Run a checksum on the installation file and check with technical support for the proper values.

Q. Do I need to install WebLogic Server as root on Solaris?

A. No you don't need to be root, depending on directory permissions.

Q. Can I run the Configuration Wizard outside the installer?

A. Yes. You can start the Configuration Wizard from the Start menu or using a script in the `utils` directory. See Creating Domains and Servers in *Configuring and Managing WebLogic Server*.

Q. I love the new domain configuration wizard. Can I create my own templates?

A. Support for custom templates for the Configuration Wizard is coming soon. Currently, it is an undocumented, unsupported feature.

FAQs: Integrating Remote JMS Providers

The J2EE standards for JMS (messaging), JTA (transaction), and JNDI (naming) work together to provide reliable java-to-java messaging between different host machines and even different vendors. BEA WebLogic Server provides a variety of tools that leverage these APIs to aid integrating remote JMS providers into a local application.

The following sections provide information on how to integrate WebLogic Server with remote JMS providers.

Understanding JMS and JNDI Terminology

- What is a remote JMS provider?
- What is JNDI?
- What is a JMS connection factory?
- What is a JMS connection-id?
- What is the difference between a JMS topic and a JMS queue?
- What is a topic subscription?
- What is a non-durable topic subscriber?
- What is a durable subscriber?

Understanding Transactions

- What is a transaction?

How to Integrate with a Remote JMS Provider

Best Practices when Integrating with Remote Providers

Using Foreign JMS Server Definitions

Using EJB/Servlet JMS Resource References

Using a Messaging Bridge

Using Message Driven Beans

JMS Interoperability Resources

Q. What is a remote JMS provider?

A. A remote JMS provider is a JMS server that is hosted outside a local stand-alone WebLogic server or outside WebLogic server cluster. The remote JMS server may be a WebLogic or a non-WebLogic (foreign) JMS server.

Q. What is JNDI?

A. JNDI (Java Naming and Directory Interface) is a J2EE lookup service that maps names to services and resources. JNDI provides a directory of advertised resources that exist on a particular stand-alone (unclustered) WebLogic server, or within a WebLogic server cluster. Examples of such resources include JMS connection factories, JMS destinations, JDBC (database) data sources, and application EJBs.

A client connecting to any WebLogic server in a WebLogic cluster can transparently reference any JNDI advertised service or resource hosted on any WebLogic server within the cluster. The client doesn't require explicit knowledge of which particular WebLogic server in the cluster hosts a desired resource.

Q. What is a JMS connection factory?

A. A JMS connection factory is a named entity resource stored in JNDI. Applications, message driven beans (MDBs), and messaging bridges lookup a JMS connection factory in JNDI and use it to create JMS connections. JMS connections are used in turn to create JMS sessions, producers, and consumers that can send or receive messages.

Q. What is a JMS connection-id?

A. JMS connection-ids are used to name JMS client connections. Durable subscribers require named connections, otherwise connections are typically unnamed. Note that within a clustered set of servers or stand-alone server, only one JMS client connection may use a particular named connection at a time. An attempt to create new connection with the same name as an existing connection will fail.

Q. What is the difference between a JMS topic and a JMS queue?

A. JMS queues deliver a message to one consumer, while JMS topics deliver a copy of each message to each consumer.

Q. What is a topic subscription?

A. A topic subscription can be thought of as an internal queue of messages waiting to be delivered to a particular subscriber. This internal queue accumulates copies of each message published to the topic after the subscription was created. Conversely, it does not accumulate messages that were sent before the subscription was created. Subscriptions are not sharable, only one subscriber may subscribe to a particular subscription at a time.

Q. What is a non-durable topic subscriber?

A. A non-durable subscriber creates unnamed subscriptions that exist only for the life of the JMS client. Messages in a non-durable subscription are never persisted—even when the message's publisher specifies a persistent quality of service (QOS). Shutting down a JMS server terminates all non-durable subscriptions.

Q. What is a durable subscriber?

A. A durable subscriber creates named subscriptions that continue to exist even after the durable subscriber exits or the server reboots. A durable subscriber connects to its subscription by specifying topic-name, connection-id, and subscriber-id. Together, the connection-id and subscriber-id uniquely name the subscriber’s subscription within a cluster. A copy of each persistent message published to a topic is persisted to each of the topic's durable subscriptions. In the event of a server crash and restart, durable subscriptions and their unconsumed persistent messages are recovered.

Q. What is a transaction?

A. A transaction is a set of distinct application operations that must be treated as an atomic unit. To maintain consistency, all operations in a transaction must either all succeed or all fail. See “Introducing Transactions” in *Programming WebLogic JTA*.

Q. Why are transactions important for integration?

A. Integration applications often use transactions to assure data consistency. For example, to assure that a message is forwarded exactly-once, a single transaction is often used to encompass the two operations of receiving the message from its source destination and sending to the target destination. Transactions are also often used to ensure atomicity of updating a database and performing a messaging operation.

Q. What is a JTA/XA/global transaction?

A. In J2EE, the terms JTA transaction, XA transaction, user transaction, and global transaction are often used interchangeably to refer to a single global transaction. Such a transaction may include operations on multiple different *XA capable* resources and even different resource types. A JTA transaction is always associated with the current thread, and may be passed from server to server as one application calls another. A common example of an XA transaction is one that includes both a WebLogic JMS operation and a JDBC (database) operation.

Q. What is a local transaction?

A. A JMS local transaction is a transacion in which only a single resource or service may participate. A JMS local transaction is associated with a particular JMS session where the destinations of a single vendor participate. Unlike XA transactions, a database operation can not participate in a JMS local transaction.

Q. How does JMS provide local transactions?

A. Local transactions are enabled by a JMS specific API called `transacted sessions`. For vendors other than WebLogic JMS, the scope of a transacted session is typically limited to a single JMS server. In WebLogic JMS, multiple JMS operations on multiple destinations within an entire cluster can participate in a single transacted session's transaction. In other words, it is scoped to a WebLogic cluster and no remote JMS provider to the JMS session's cluster can participate in a transaction. See the *WebLogic JMS Performance Guide* white-paper available on the JMS topic page.

Q. Are JMS local transactions useful for integration purposes?

A. Local transactions are generally not useful for integration purposes as they are limited in scope to a single resource, typically a messaging or database server.

Q. What is Automatic Transaction Enlistment?

A. Operations on resources such as database servers or messaging servers participate in a J2EE JTA transaction provided that:

- the resource is XA transaction capable
- the resource has been enlisted with the current transaction
- the client library used to access the resource is transaction aware (XA enabled).

Automatic participation of operations on an XA capable resource in a transaction is technically referred to as automatic enlistment.

- WebLogic clients using XA enabled WebLogic APIs automatically enlist operation in the current thread's JTA transaction. Examples of XA enabled WebLogic clients include WebLogic JMS XA enabled (or user transaction enabled) connection factories, and JDBC connection pool data sources that are global transaction enabled.
- Foreign (non-WebLogic) JMS clients do not automatically enlist in the current JTA transaction. Such clients must either go through an extra step of programmatically enlisting in the current transaction, or use WebLogic provided features that wrap the foreign JMS client and automatically enlist when the foreign JMS client is accessed via wrapper APIs.

JMS features that provide automatic enlistment for foreign vendors are:

- Message-Driven EJBs
- JMS resource-reference pools
- Messaging Bridges

To determine if a non-WebLogic vendor's JMS connection factory is XA capable, check the vendor documentation. Remember, support for transacted sessions (local transactions) does not imply support for global/XA transactions.

Q. What does a JMS client do to communicate with a remote JMS provider?

A. To communicate with any JMS provider, a JMS client must perform the following steps:

a. Look up a JMS connection factory object and a JMS destination object using JNDI

b. Create a JMS connection using the connection factory object

c. Create message consumers or producers using the JMS connection and JMS destination objects.

Q. What information do I need to set up communications with a remote JMS provider?

A. You will need the following information to set up communications with a remote JMS provider:

- The destination type—whether the remote JMS destination is a queue or a topic.
- The JNDI name of the remote JMS destination.
- For durable topic subscribers—the connection-id and subscriber-id names that uniquely identify them. Message Driven EJBs provide default values for these values based on the EJB name.
- For non-WebLogic remote JMS providers
  - Initial Context Factory Class Name— the java class name of the remote JMS Provider's JNDI lookup service.
  - The file location of the java jars containing the remote JMS provider's JMS client and JNDI client libraries. Ensure that these jars are specified in the local JVM's classpath.
- The URL of the remote provider's JNDI service. For WebLogic servers, the URL is normally in the form `t3://hostaddress:port`. If you are tunneling over HTTP, begin the URL with `http` rather than `t3`. No URL is required for server application code that accesses a WebLogic JMS server that resides on the same WebLogic server or WebLogic cluster as the application.
- The JNDI name of the remote provider's JMS connection factory. This connection factory must exist on the remote provider, not the local provider.

  If the JMS application requires transactions, the connection factory must be XA capable. WebLogic documentation refers to XA capable factories as user transactions enabled.

  By default, WebLogic servers automatically provide three non-configurable connection factories:

  - `weblogic.jms.ConnectionFactory`—a non-XA capable factory.
  - `weblogic.jms.XAConnectionFactory`—an XA-capable factory
  - `weblogic.jms.MessageDrivenBeanConnectionFactory`—an XA-capable factory for message driven EJBs.

  Additional WebLogic JMS connection factories must be explicitly configured.

Q. What if a foreign JMS provider JNDI service has limited functionality?

A. The preferred method for locating JMS provider connection factories and destinations is to use a standard J2EE JNDI lookup. Occasionally a non-WebLogic JMS provider's JNDI service is hard to use or unreliable. The solution is to create a startup class or load-on-start servlet that runs on a WebLogic server that does the following:

- Uses the foreign provider's proprietary (non-JNDI) APIs to locate connection factories and JMS destinations.
- Registers the JMS destinations and JMS connection factories in WebLogic JNDI.

For sample code, see "Creating Foreign JNDI Objects in a Startup Class" in *Using Foreign JMS Providers with WebLogic Server* white-paper available from the JMS topic page.

Q. How can I pool JMS resources?

A. Remote and local JMS resources, such as client connections and sessions, are often pooled to improve performance. Message driven EJBs automatically pool their internal JMS consumers. JMS consumers and producers accessed through resource-references are also automatically pooled. For more information on resource pooling, including information on writing a custom pool, see the *WebLogic JMS Performance Guide* white-paper available on the JMS topic page.

Q. What version interoperability does WebLogic provide?

A. All WebLogic server releases 6.1 and higher interoperate freely between releases. For example, a WebLogic 8.1 JMS client can send messages directly to a 6.1 JMS server and vice versa. A Messaging Bridge can be used to forward WebLogic 5.1 JMS messages to and from WebLogic server releases 6.1 and higher.

Q. What tools are available for integrating with remote JMS providers?

A. The following table summarizes the tools available for integrating with remote JMS providers:

Method	Automatic Enlistment	JMS Resource Pooling
Direct use of the remote provider’s JMS client	Yes for a WebLogic server provider. Other providers must perform enlistment programmatically.	No. Can be done programmatically.
Messaging Bridge	Yes	N/A

Method	Automatic Enlistment	JMS Resource Pooling
Foreign JMS Server Definition	No. To get automatic enlistment, use in conjuction with a JMS resource reference or MDB.	No. To get resource pooling, use in conjunction with a JMS resource reference or MDB.
JMS Resource Reference	Yes	Yes
Message Driven EJBs	Yes	Yes

Q. How do I receive messages from a remote a JMS provider from within an EJB or Servlet?

A. Use a message driven EJB. Synchronous receives are not recommended because they idle a server side thread while the receiver blocks waiting for a message. See What is a Message Driven EJB (MDB)? and When should I use a MDB?

Q. How do I send messages to a remote JMS provider from within an EJB or Servlet?

A. Use a resource reference. It provides pooling and automatic enlistment. See What are JMS resource references? and What advantages do JMS resource references provide? In limited cases where wrappers are not sufficient, you can write your own pooling code. See the WebLogic JMS Performance Guide white-paper available on the JMS topic page.

If the target destination is remote, consider adding a local destination and messaging bridge to implement a store-and-forward high availability design. See What is a messaging bridge? and When should I use a messaging bridge?.

Another best practice is to use foreign JMS server definitions. Foreign JMS server definitions allow an application's JMS resources to be administratively changed and avoid the problem of hard-coding URLs into application code. In addition, foreign JMS server definitions are required to enable resource references to reference remote JMS providers. See What are Foreign JMS Server Definitions? and When is it best to use a Foreign JMS Server Definition?.

Q. How do I communicate with remote JMS providers from a client?

A. If the destination is not provided by WebLogic Server, and there is a need to include operations on the destination in a global transaction, use a server proxy to encapsulate JMS operations on the foreign vendor in an EJB. Applications running on WebLogic server have facilities to enlist non-WebLogic JMS providers that are transaction (XA) capable with the current transaction. See How do I receive messages from a remote a JMS provider from within an EJB or Servlet? and How do I send messages to a remote JMS provider from within an EJB or Servlet?.

If you need store-and-forward capability, consider sending to local destinations and using messaging bridges to forward the message to the foreign destination. See What is a messaging bridge? and When should I use a messaging bridge?.

Another option is to simply use the remote vendor's JNDI and JMS API directly or configuring foreign JMS providers to avoid hard-coding references to them. You will need to add the foreign provider's class libraries to the client's class-path.

Q. How can I tune WebLogic JMS interoperability features?

A. For information on tuning message driven EJBs and the WebLogic Messaging Bridges, see the *WebLogic JMS Performance Guide* white-paper available on the JMS topic page.

Q. What are Foreign JMS Server Definitions?

A. Foreign JMS server definitions are an administratively configured symbolic link between a JNDI object in a remote JNDI directory, such as a JMS connection factory or destination object, and a JNDI name in the JNDI name space for a stand-alone WebLogic Server or a WebLogic cluster. They can be configured using the Administration console, standard JMX MBean APIs, or programmatically using scripting. See "Simple Access to Remote or Foreign JMS Providers" in the Administration Console Online Help.

Q. When is it best to use a Foreign JMS Server Definition?

A. For this release, a Foreign JMS Server definition conveniently moves JMS JNDI parameters into one central place. You can share one definition between EJBs, servlets, and messaging bridges. You can change a definition without recompiling or changing deployment descriptors. They are especially useful for:

- Any message driven EJB (MDB) where it is desirable to administer standard JMS communication properties via configuration rather than hard code them into the application's EJB deployment descriptors. This applies even if the MDB's source destination isn't remote.
- Any MDB that has a destination remote to the cluster. This simplifies deployment descriptor configuration and enhances administrative control.
- Any EJB or servlet that sends or receives from a remote destination.
- Enabling resource references to refer to remote JMS providers. See What are JMS resource references? and How do I use resource references with non-transactional messaging?.

Q. What are JMS resource references?

A. Resource references are specified by servlet and EJB application developers and packaged with an application. They are easy-to-use and provide a level of indirection that lets applications reference JNDI names defined in an EJB descriptor rather than hard-coding JNDI names directly into application source code.

JMS resource-references provide two additional features:

- Automatic pooling of JMS resources when those resources are closed by the application.
- Automatic enlistment of JMS resources with the current transaction, even for non-WebLogic JMS providers.

Inside an EJB or a servlet application code, use JMS resource references by including resource-ref elements in the deployment descriptors and then use JNDI a context to look them up using the syntax `java:comp/env/jms/<reference name>`.

Resource references provide no functionality outside of application code, and therefore are not useful for configuring a message driven EJB's source destination or a messaging bridge's source or target destinations.

For WebLogic documentation on JMS resource-reference pooling, see "Using JMS with EJBs and Servlets" in *Programming WebLogic JMS.*

Q. What advantages do JMS resource references provide?

A. JMS resource references provide the following advantages:

- They ensure portability of servlet and EJB applications: they can be used to change an application's JMS resource without recompiling the application's source code.
- They provide automatic pooling of JMS Connection, Session, and MessageProducer objects.
- They provide automatic transaction enlistment for non-WebLogic JMS providers. This requires XA support in the JMS provider. If resource references aren't used, then enlisting a non-WebLogic JMS provider with the current transaction requires extra programmatic steps.

Q. How do I use resource references with foreign JMS providers?

A. To enable resource references to reference remote JMS providers, they must be used in conjunction with a foreign JMS definition. This is because resources references do not provide a place to specify a URL or initial context factory. See What are Foreign JMS Server Definitions?.

Q. How do I use resource references with non-transactional messaging?

A. For non-transactional cases, do not use a global transaction (XA) capable connection factory. This will impact messaging performance. If you do, the resource reference will automatically begin and commit an internal transaction for each messaging operation. See What is a transaction?.

Q. What is a messaging bridge?

A. Messaging bridges are administratively configured services that run on a WebLogic server. They automatically forward messages from a configured source JMS destination to a configured target JMS destination. These destinations can be on different servers than the bridge and can even be foreign (non-WebLogic) destinations. Each bridge destination is configured using the four common properties of a remote provider:

- The initial context factory.
- The connection URL.
- The connection factory JNDI name.
- The destination JNDI name.

Messaging bridges can be configured to use transactions to ensure exactly-once message forwarding from any XA capable (global transaction capable) JMS provider to another.

Q. When should I use a messaging bridge?

A. Typically, messaging bridges are used to provide store-and-forward high availability design requirements. A messaging bridge is configured to consume from a sender's local destination and forward it to the sender's actual target remote destination. This provides high availability because the sender is still able to send messages to its local destination even when the target remote destination is unreachable. When a remote destination is not reachable, the local destination automatically begins to store messages until the bridge is able to forward them to the target destination when the target becomes available again.

Q. When should I avoid using a messaging bridge?

A. Other methods are preferred in the following situations:

- Receiving from a remote destination—use a message driven EJB or implement a client consumer directly.
- Sending messages to a local destination—send directly to the local destination.

- Environment with low tolerance for message latency. Messaging Bridges increase latency and may lower throughput. Messaging bridges increase latency for messages as they introduce an extra destination in the message path and may lower throughput because they forward messages using a single thread.

Q. What is a Message Driven EJB (MDB)?

A. Message Driven EJBs are EJB containers that internally use standard JMS APIs to asynchronously receive messages from local, remote, or even foreign JMS destinations and then call application code to process the messages. MDBs have the following characteristics:

- Automatically connects to a source destination and automatically retries connecting if the remote destination is inaccessible.
- Support automatic enlistment of the received messages in container managed transactions, even when the JMS provider is not WebLogic.
- Automatically pool their internal JMS connections, sessions, and consumers.
- A MDB's source destination, URL, and connection factory are configured in the EJB and WebLogic descriptors which are packaged as part of an application.
- The messaging processing application logic is contained in a single method callback `onMessage()`.
- AMDB is a full-fledged EJB that supports transactions, security, JDBC, and other typical EJB actions.

For more information, see “Message-Driven EJBs” in *Programming WebLogic Enterprise JavaBeans*.

Q. When should I use a MDB?

A. MDBs are the preferred mechanism for WebLogic server applications that receive and process JMS messages.

Q. Do I need to use a Messaging Bridge with a MDB?

A. Configure MDBs to directly consume from their source destination rather than insert a messaging bridge between them. MDBs automatically retry connecting to their source destination if the source destination is inaccessible, so there is no need to insert a messaging bridge in the message path to provide higher availability. Introducing a messaging bridge may have a performance impact. See When should I avoid using a messaging bridge?.

Q. What is the best way to configure a MDB?

A. The following section provides tips for configuring a MDB:

- To configure MDB concurrency and thread pools, use the `max-beans-in-free-pool` and `dispatch-policy` descriptor fields. WebLogic may create fewer concurrent instances than `max-beans-in-free-pool` depending on the number of available server threads in the MDB's thread pool.
- Use foreign JMS server definitions when configuring a MDB to consume from a remote JMS provider. Although WebLogic MDB descriptors can be configured to directly refer to remote destinations, this information is packaged with the application and is not dynamically editable. You should configure a foreign JMS server definition and then configure the MDB to reference the foreign definition instead. Please note that some documentation refers to foreign JMS server definitions as wrappers. See What are Foreign JMS Server Definitions?.
- Use care when configuring a MDB for container managed transactions. A MDB supports container managed XA transactions when a MDB's descriptor files have `transaction-type` of `Container` and a `trans-attribute` of `Required` and the JMS connection factory is XA enabled. Failure to follow these steps will result in the MDB being non-transactional. The default WebLogic setting for a MDB connection factory is XA enabled. The MDB automatically begins a transaction and automatically enlists the received message in the transaction.

Q. What additional resources document JMS interoperability?

A. For general information on WebLogic JMS see the JMS topic page.

- The slide presentation "Integrating Foreign JMS Providers with BEA WebLogic Server (from eWorld 2003)" contains sample JMS integration configuration and code.
- The "Using Foreign JMS Providers with WebLogic Server" white-paper provides helpful sections including, "Creating Foreign JNDI Objects in a Startup Class", "Configuring JMS Providers", and "Sample Code".
- The "WebLogic JMS Performance Guide" white-paper.

FAQs: Java

- Can you help me debug my program?
- Where can I get help learning Java?
- Where do I get the JDK?
- How do I set up my CLASSPATH?
- Why won't the examples work?
- Where can I get help with Java error messages?
- Why did a client-server message generate a StackOverflowException?
- Will a JIT make my Java application run faster?
- Can I redistribute the JDK that is bundled with WebLogic Server?

Q. Can you help me debug my program?

A. If the problem you are having is not directly related to our software, we suggest that you use a Java development tool that helps you with debugging, and that you invest in some books or training to help you learn Java. There are many ways to build debugging into your program, and getting good training in Java programming is a good start to understanding how to do this.

Q. Where can I get help learning Java?

A. There are a lot of books and online references to Java. A good starting place is at the JavaSoft website documentation index at http://www.javasoft.com/docs/index.html, which has links to white papers and the Java Tutorial. You can find books on Java at any of the major online book shopping sites.

Q. Where do I get the JDK?

A. WebLogic 8.1 bundles JDK 1.4. See our Platform support page for information about specific JDKs that we have tested and certified for use with WebLogic software.

Once you determine which version of the JDK you will be using, go to the JavaSoft website at http://www.javasoft.com/products/. Many platform vendors provide an optimized version of the JDK for their computers.

Q. How do I set up my CLASSPATH?

A. Setting up your CLASSPATH correctly depends on what you are trying to do. The most common tasks are described below:

- **Starting WebLogic Server.** See Setting the Classpath Option in the "Starting and Stopping WebLogic Servers" section of *Administration Console Online Help*. In addition, your WebLogic distribution includes shell scripts that you can use to start the server. These scripts, which are located in the domain directories under the `config` directory of your WebLogic Server distribution, automatically set up the CLASSPATH variable in the shell before starting the server.
- **Compiling Application Classes or Using WebLogic Server Utilities.** See Setting the Classpath for Compiling in the Developing WebLogic Server Components section of *Developing WebLogic Server Applications*.
- **Working With WebLogic Server Code Examples.** See the WebLogic Server Examples Guide located at `samples/server/examples/src/examples/examples.html` in your WebLogic Server distribution.

Q. Why won't the examples work?

A. Usually problems with examples are related to your environment. Here are some troubleshooting hints:

1. If you are using a database, make sure you have run the utility `utils.dbping` to verify that your JDBC driver is correctly installed and configured.
2. Run the setEnv script to make sure your CLASSPATH is correctly set in the shell or DOS window in which you are running the examples.

3. Check the instructions for the examples to make sure you have changed any user-specific variables in the code before compiling.

4. Verify that you are compiling with the -d option to direct the class files into the proper directory, as defined in the example instructions.

If the example is an applet, check the CODE and CODEBASE, and make sure WebLogic Server is running.

Q. Where can I get help with Java error messages?

A. Many questions we receive at BEA are related to generic Java error messages and are not specific to WebLogic Server. Here are some links that contain helpful information about Java error messages.

Table 9-1

Source	Description
Sun's Java Developer Connection at http://developer.java.sun.com/developer/	This forum includes Questions and Answers on a wide variety of Java topics, including error messages. Use the Search box for fast results. For example, type "classpath error" in the Search box.
Sun's Java APIs at http://java.sun.com/products/	Check the Java API to see if there is an exception description for the class you are using.

Q. Why did a client-server message generate a StackOverflowException?

A. If you are sending a particularly large data structure using java.io.Serialization, you may exceed the per-thread size limit for either the Java or native stack. You can increase the stack size by using the following command line options:

`-ss` *`Stacksize`* to increase the native stack size or
`-oss` *`Stacksize`* to increase the Java stack size,

where *Stacksize* is expressed as an integer followed by “k” or “m” for kbytes or mbytes. For example,

```
$java -ss156k (native)
```

```
$java -oss600k (Java)
```

The default *native* stack size is 128k, with a minimum value of 1000 bytes. The default *java* stack size is 400k, with a minimum value of 1000 bytes.

Q. Will a JIT make my Java application run faster?

A. A Just-In-Time compiler will make some Java applications run faster. A JIT works by storing generated machine code in memory and reusing it when possible. For example, if you execute the same operation 1000 times in a loop, a JIT will improve performance of this operation since the code will only be generated once. Applications with a lot of native methods will not see as much performance improvement as pure-Java applications.

If you use a JIT, you may want to turn off the JIT during debugging to facilitate stacktracing. If you are doing performance testing with a JIT, make sure that you execute the same test several times in the same invocation and then throw away the first result to get an idea of how long the transaction will take when your application is running in a steady state. The first time the code is executed, your test will take longer (the "code generation hit").

Q. Can I redistribute the JDK that is bundled with WebLogic Server?

A. BEA Systems has the non-exclusive right to grant a third party, such as an independent software vendor (ISV), the right to redistribute the JDK that is bundled with WebLogic Server without any modifications of any kind. The following are caveats to this general statement:

- The ISV cannot remove or alter any proprietary legends or notices contained in the JDK. The ISV shall not decompile, disassemble, decrypt, extract, or otherwise reverse engineer or modify the JDK. The JDK may not be leased or assigned in whole or in part.
- The ISV must enter into a signed agreement with its distributors on terms substantially similar to those contained here in this redistribution policy explanation.
- The ISV requires an end user license agreement with the product within which it embeds WebLogic Server.
- The embedding of the JDK in ISV products does not include maintenance and support of the JDK by the JDK Provider. BEA Systems shall be solely responsible for providing maintenance and support for its ISVs and distributors. The ISV shall be solely responsible for providing maintenance and support for the end users of its products.
- If an ISV wants to ship a JDK that is different from the JDK that BEA ships with WebLogic Server, that ISV needs to get those bundling rights directly from Sun or HP. For example:

  Assume that BEA ships WebLogic Server 6.0 with JDK 1.3 only, that BEA ships WebLogic Server 5.1 with JDK 1.1 only, and that an ISV wants to ship JDK 1.1 with

WebLogic Server 6.0 in an integrated product offered by that ISV. Unless BEA, for BEA business reasons, elects to make WebLogic Server 6.0 with JDK 1.1 generally available, the ISV couldn't ship JDK 1.1 with WebLogic Server 6.0 in an integrated product offered by that ISV under BEA's agreement with that ISV or under BEA's agreement with Sun. The ISV could, however, obtain its own binary distribution agreement for the JDK from Sun and under that agreement bundle JDK 1.1 with its value added software solution consisting of the ISV applications and WebLogic Server 6.0 integrated.

FAQs: J2EE Connector Architecture

- Why do I get the following exception when viewing the JNDI tree?
- Is it possible to use the Oracle database rather than PointBase with the current implementation of the WebLogic J2EE Connector Architecture?
- When deploying a resource adapter (.rar) to WebLogic Server, are its classes placed in the WebLogic classpath?
- Why does WebLogic Server invoke the `ManagedConnection.addConnectionEventListener()` function whenever the sample EJB calls `ConnectionFactory.getConnection()` to connect to the EIS?
- Why do I get an exception when I compile my EJB to use a Resource Adapter that supports CCI?
- The BEA com.bea.adapter.dbms.cci.ConnectionImpl does not directly implement javax.resource.cci.Connection. Is there a work-around for this?

Q. Why do I get the following exception when viewing the JNDI tree?

```
isSerializable(class.javax.naming.Binding)
java.io.NotSerializableException:
java.io.PrintWriter at
java.io.ObjectOutputStream.OutputObject
```

A. The Weblogic Server JNDI implementation requires objects to be serializable, not referencable. A PrintWriter cannot be serialized and therefore should be declared transient.

Q. Is it possible to use the Oracle database rather than PointBase with the current implementation of the WebLogic J2EE Connector Architecture?

A. The sample provided contains a resource adapter that represents any database system. By default, configurations are set to use PointBase. In particular, the configuration properties in the `weblogic-ra.xml` file set a PointBase datasource. This can be replaced with Oracle settings.

Also, the resource adapter (in particular the `ManagedConnectionFactory`) must be implemented to support Oracle. The resource adapter provided in this sample makes use of the JDBC and therefore can support any database system which the resource adapter is configured to represent.

Q. When deploying a resource adapter (.rar) to WebLogic Server, are its classes placed in the WebLogic classpath?

For instance, I am deploying an EJB and a resource adapter (.rar), the EJB has no dependencies on the `.rar` because the EJB is writing to the common client interface (CCI). The EJB client application has sends/marshals as parameter classes that are defined in the `.rar`. For some reason the EJB's class loader hierarchy cannot find the definition of this `.rar`-specific class, even though the `.rar` is deploying successfully. I receive the following error on the EJB client:

```
java.rmi.UnmarshalException: error unmarshalling arguments; nested
exception
is:
java.lang.ClassNotFoundException:
com.mycompany.InteractionSpecImpl
```

A. When you pass an instance of `com.myclientcompany.server.eai.InteractionSpecImpl` as an argument to your EJB, the `appServer` needs to de-serialize (unmarshal) the object under the EJB context, and it needs the required class for unmarshalling, inside the `ejb-jar`(`raTester.jar`). So if you include the `interactionspecimpl` class in your `ejb-jar` file, then you do not need to include those classes in your server's classpath.

Q. Why does WebLogic Server invoke the `ManagedConnection.addConnectionEventListener()` function whenever the sample EJB calls `ConnectionFactory.getConnection()` to connect to the EIS?

A. This is a requirement and is part of the contract between the Resource Adapter and the application server.

Q. Why do I get an exception when I compile my EJB to use a Resource Adapter that supports CCI?

I specify javax.resource.cci.ConnectionFactory in the resource-ref but I get the following exception when I attempt to compile my EJB:

```
weblogic.xml.process.SAXValidationException:
ejb-jar.enterprise-beans.session.resource-ref.res-type.
must be one of the values:
javax.sql.DataSource,javax.jms.QueueConnectionFactory,
javax.jms.TopicConnectionFactory,
java.net.URL,
javax.mail.Session
at
weblogic.ejb20.dd.xml.EjbJarLoader_EJB11.__post_84
```

A. Make sure your ejb-jar.xml is referencing the EJB2.0 DTD instead of the EJB1.1 DTD. The ConnectionFactory resource ref is only supported in the EJB 2.0 DTD.

Q. The BEA com.bea.adapter.dbms.cci.ConnectionImpl does not directly implement javax.resource.cci.Connection. Is there a work-around for this?

A. Yes. The BEA com.bea.adapter.dbms.cci.ConnectionImpl extends com.bea.adapter.cci.AbstractConnection; this in turn implements the Connection interface. The Proxy is constructed using the interfaces from the most-derived class (ConnectionImpl). The dumpFamilyTree printout shows that the getInterfaces call on the ConnectionImpl class does NOT include the Connection interface. However, the getInterfaces call on AbstractConnection does include the Connection interface.

The work-around is that the ConnectionImpl class must directly implement the interface class specified in the ra.xml file. (This might be a redundant statement in the code, particularly if it extends a class that has already implemented the class.) Then rebuild the adapter and try your test again.

FAQs: WebLogic JDBC

Configuration

- Are MultiPools supported in a cluster?
- When should I use JTS?
- How can I tell if a database is unavailable?
- How do I configure the driver from Microsoft for SQL Server 2000?
- I have a long running transaction that makes multiple updates to an XA-compliant resource manager and a resource enlistment, prepare, or rollback operation fails with an XAException (`errorCode=XAER_NOTA`). What causes this behavior and how can it be prevented?
- How do testConnsOnReserve and refreshTestMinutes work together?
- How do I use the prepared statement cache?

Programming — General

- Are there recommended programming practices for using JDBC connections?
- Why should I not use `DriverManager.getConnection`?
- Can I use a prepared statement across multiple transactions?
- Why do I get a `java.lang.AbstractMethodError` when calling a method on a driver?
- Why do I get "ResourceException: No Resource Available"?
- How do I ensure that a new database connection is created each time an EJB's container-managed transaction is started (so that I get a new authentication/authorization each time)?
- When you look up a data source via JNDI and access a database connection from an external process, do you get a stub for the Connection instance in the WebLogic process or does it create a new connection pool with separate Connections in the local process?
- If a distributed transaction involves JMS and JDBC, how do I ensure that the JDBC update is available when the JMS message is processed?
- If an application calls `DataSource.getConnection` multiple times in the same thread and transaction, will WebLogic Server handle giving me the same connection and transaction?
- Why do I get a SystemException failure when trying to enlist my XAResource in a client?

Programming — Oracle

- What causes an `OCIW32.dll` error?
- How do I use Unicode codesets with the WebLogic jDriver for Oracle driver?
- What type of object is returned by `ResultSet.getObject()` when using the WebLogic jDriver for Oracle?
- How do I limit the number of Oracle database connections generated by WebLogic Server?
- How do I call Oracle stored procedures that take no parameters?
- How do I bind string values in a PreparedStatement?
- Why do I get unexpected characters from 8-bit character sets in WebLogic jDriver for Oracle?
- How do I learn what codesets are available in Oracle?
- How do I look up an "ORA" SQLException?
- What is error "ORA-6502?"
- Why do I get an error while trying to retrieve the text for ORA-12705?
- Why do I run out of resources during updates with Oracle's database link?
- Why does executing the PreparedStatement class cause a "TRUNC fails: ORA-00932: inconsistent datatypes" error?
- How do I create and update Oracle Blob fields?
- How do I enlist an Oracle XAResource?
- Why do I get "ORA-00600" ?
- Why do I get "ORA-24327" ?
- Why do I get "java.sql.SQLException: getOBJECT is not supported by the WebLogic JDBC Driver"?
- Why do I get an "ORA-01453" when I use SET TRANSACTION?

Q. Can I use the PointBase DBMS included with WebLogic Server for development or production?

A. PointBase Server is an all-Java DBMS product included in the WebLogic Server distribution solely in support of WebLogic Server evaluation, either in the form of custom trial applications or through packaged sample applications provided with WebLogic Server. Non-evaluation development and/or production use of the PointBase Server requires a separate license be obtained by the end user directly from PointBase.

Q. How can I enable Oracle Advanced Security encryption on the JDBC Oracle Thin driver with a WebLogic JDBC Connection Pool?

A. Oracle Advanced Security encryption relies on features available through connection properties in the JDBC driver from Oracle. You can specify connection properties in a WebLogic JDBC connection pool in the Properties attribute. This attribute is available on the JDBC Connection Pool →Configuration →General tab in the Administration Console. When WebLogic Server creates database connections for the connection pool, it passes the properties to the JDBC driver so that connections are created with the specified properties.

For example, to enable Oracle Advanced Security encryption, you may want to specify the following options:

```
Properties:  user=SCOTT
             oracle.net.encryption_client=ACCEPTED
             oracle.net.encryption_types_client=RC4_256
             oracle.net.crypto_checksum_client=ACCEPTED
             protocol=thin
```

Note: See the Oracle documentation for details about required properties for Oracle Advanced Security encryption. Properties listed above are for illustration only.

The resulting entry in the `config.xml` file would look like:

```
<JDBCConnectionPool
    DriverName="oracle.jdbc.driver.OracleDriver"
    Name="oraclePool"
    Password="{3DES}1eNn7kYGZVw="
    Properties="user=SCOTT;
    oracle.net.encryption_client=ACCEPTED;
    oracle.net.encryption_types_client=RC4_256;
    oracle.net.crypto_checksum_client=ACCEPTED;
    protocol=thin"
    URL="jdbc:oracle:thin:@server:port:sid"
/>
```

Note: Line breaks added for readability.

Q. When should I use a TxDataSource instead of a DataSource?

A. See "When to Enable Global Transactions in a Data Source" in the *Administration Console Online Help*. When you select Honor Global Transactions in the Administration Console, you create a JDBCTxDataSource in the `config.xml` file (the default). If you clear the Honor Global Transactions check box, you create a JDBCDataSource in the `config.xml` file.

Q. Which is better: all-java (Type 4) drivers or native drivers (Type 2)?

A. The advantages of Type 4 drivers are the same advantages of Java code, namely portability and safety. Since Java code runs within the Java Virtual Machine, it cannot crash the process.

The traditional advantage of Type 2 native drivers has been performance. However, with the rapid advances in Java performance, this is no longer always true. In general, JDBC driver performance is dependent on many factors, especially the SQL code used issued and the JDBC driver implementation.

Q. How should I set the connection pool initial size?

A. In a production system, set the initial size equal to the maximum size. This will create all necessary JDBC connections at server start-up.

The reason is that if initial number of connections is less than the maximum number of connections, the server has to create additional connections when it becomes loaded. When a server is under heavy load, the goal is to only do useful work with minimal overhead. That is exactly the time that you do *not* want to burden the server with creating additional database connections. For both performance and stability reasons, it is best to have the connection pool make all its connections at startup.

During development, set the initial size to a small value like 1. This helps the server boot faster and allows you to use a database that doesn't support as many concurrent connections.

Q. How should I set the connection pool's maximum size?

A. In production environments, the best maximum pool size depends on the application and the system. A good starting point is to set the maximum pool size equal to the execute thread count. This allows each execute thread to get its own JDBC connection without blocking while waiting for a connection in use by another thread.

Some applications can run with a maximum pool size less than the number of execute threads. This case generally occurs when there is work entering the server that does not ever need a JDBC connection, and the server has lots of threads doing I/O tasks.

Some applications require a maximum pool size greater than the number of execute threads. In this case, each thread may require more than one connection from the same connection pool. A common way this occurs is when an EJB method uses a connection and then calls a `RequiresNew` method on another EJB before committing its transaction. The `RequiresNew` EJB then gets its own connection from the pool.

Theoretically, you need as many connections as you need concurrent users to be served (though this may not be practical and connections may need to be serially reused).

Note that these recommendations do take into account any connection leaks (connections that you reserve but do not close).

During development, set the maximum size to a small value. Unless you are doing load tests against your development machine, you probably do not have very many concurrent users so you don't need many connections. Also, with fewer connections in the connection pool, it will be easier to run into situations where connections are leaking from the connection pool because application code is not closing them. It is better to encounter this during development and fix it before going into production.

If a connection is not explicitly released by an object, the connection is returned back to the connection pool when the object is destroyed or the connection is actually closed. When the references to a connection are gone, the connection is returned to the connection pool during garbage collection. Note that in case of remote references to a connection (when a connection is obtained using a data source), there is an extra delay that may look as though the connection did not get returned. This is because the distributed garbage collection is more time consuming. It may take 6 minutes or more before it is returned to the pool.

Do not rely on the garbage collection to return connections to the connection pool. It is best to track down offending code that doesn't properly return connections to the connection pool after use and fix the code.

Q. Can I enable requests to a JDBC connection pool for a database connection to wait until a connection is available?

A. Yes. You can set two JDBC connection pool properties to enable connection requests to wait for a connection:

- `ConnectionReserveTimeoutSeconds`
- `HighestNumWaiters`

See “Enabling Connection Requests to Wait for a Connection” in *Programming WebLogic JDBC*.

Q. How can I avoid ResourceExceptions when sending more requests for database connections from the connection pool than are currently available?

A. The fundamental problem is too few resources (database connections in the connection pool) for the work load. The correct response is to increase the maximum number of connections in the connection pool. Optimally designed applications only require the server to have one pool connection per execute thread.

You can also enable connection requests to wait for a connection. See "Enabling Connection Requests to Wait for a Connection" in *Programming WebLogic JDBC*.

Note that the proper application response to a resource exception is *not* to retry the request in a tight loop, which would tie up execute threads on the server.

You should design your application to gracefully fail if no connections are available. Try to ensure that you get the connection as late as possible in your application code and return it to the connection pool as early as possible so that you do not see as many NoResource exceptions. It is better to have the connection as a method level variable and close the connection in a finally block as in the following example:

```
try{
  ...
} catch(Exception handleEx) {
  ...
} finally {
  try{ conn.close(); // always return the connection to pool
}catch (Exception ignore){}
}
```

Q. When are JDBC connections returned to the connection pool?

A. It depends on how the application obtains the connection. If the connection is obtained by an EJB or other application code in a way that the connection is part of a WebLogic-managed transaction (using a tx data source, the JTS driver, and so forth), then the connection is returned to the connection pool as soon as the transaction ends, either by commit, rollback, or timeout.

If the connection is obtained by application code outside the context of a WebLogic-managed transaction, such as from a non-tx data source, the connection is returned to the pool as soon as the application code closes it. If the connection is not closed, eventually it will be garbage collected and then returned to the connection pool, but this is slow and indeterminate and can cause failures for applications that need a connection, so it is important to ensure that applications close pool connections when finished using them. Also, in JSPs and servlets, where one class may be running simultaneously in multiple threads, it is important that Connection objects be *method* objects, not class or instance variables, because one thread may overwrite a connection object with a new one. The overwritten connection cannot be closed, and will be unavailable (out of the connection pool) until it is garbage collected.

Q. What happens when my database is restarted or becomes unreachable? Does my connection pool stick around?

A. Yes. The pool is independent of its ability to communicate with to the DBMS. All connections in the connection pool may become defunct, but the connection pool still exists. You can configure the connection pool so that WebLogic Server tests the connections in the pool and replaces bad connections when it can. See "Testing Connection Pools and Database Connections" in *Programming WebLogic JDBC* for more information.

To manually restart the connection pool using the Administration Console after a database failure, you can undeploy the connection pool by removing all of its deployment targets, and then redeploy the connection pool by adding deployment targets. See "Shutting Down a JDBC Connection Pool" in the *Administration Console Help*.

To do this from the command line using "weblogic.Admin", set the "Targets" attribute of the pool to an empty string ("") and then set it to the desired set of targets. See "Using weblogic.Admin Commands to Target Resources" in the *WebLogic Server Command Reference*.

Q. What is the "Connections Total" in the console?

A. The connections total is a cumulative number that indicates how many connections were created during the existence of the pool. It is not the total number of connections currently in the connection pool. Connections get added when the connection pool grows or if a bad connection is replaced.

Q. When should I use MultiPools?

A. You can use MultiPools in one of two ways 1) for high availability in the event a database connection fails, or 2) for load balancing between JDBC connection pools. Because you can choose only one option, you need to determine the primary purpose of your MultiPool.

Note: If you implement Multipools for a JDBC application , do not configure driver-level load balancing or failover for the connection pools used by the MultiPool—the MultiPool provides the same functionality as configuring driver-level load balancing or failover.

For more information, see "Configuring and Using MultiPools" in *Programming WebLogic JDBC*.

Q. Are MultiPools supported in a cluster?

A. Support for MultiPools in a cluster is identical to support for basic connection pools. As far as an application is concerned, MultiPools are indistinguishable from basic connection pools. If you have an application that uses a connection pool, and will be running on all the members of a cluster, then you need to have a like-named connection pool on each member of the cluster. It is identical for MultiPools. The application will ask for a connection from the MultiPool, so there needs to be a like-named MultiPool on each member of the cluster.

Q. When should I use JTS?

A. JTS enables you to use a non-XA JDBC resource (driver) in a global transaction with other XA JDBC resources as participants in the transaction. To configure this XA emulation mode, you configure a JDBC connection pool to use a non-XA JDBC driver to create database connections, then configure a tx data source with the EnableTwoPhaseCommit attribute set to true (in the Administration Console, the attribute is "Emulate Two-Phase Commit for non-XA Driver"), and point tx data source to the JDBC connection pool. Using this mode, database updates are committed or rolled back using a local database transaction. This mode of operation is suited for the following situations:

- When there is a single resource involved in the global transaction. In this case, a one-phase commit optimization is used so heuristic outcomes and recovery failures are not an issue. This will perform better than using an XA JDBC driver.
- When the application can tolerate data inconsistencies.

The limitations of using JTS are as follows.

- Data inconsistencies may result if the transaction is committed but the local transaction fails to commit. This results in a heuristically completed global transaction.
- No recovery is available for a non-XA participant in a transaction. If a transaction is committed but the non-XA participant has not been informed of the decision before the failure, manual intervention is required to commit the transaction.
- All communications for the commit processing are done via connection that is first established with the resource. This may result in additional RMI traffic if multiple servers are involved using the same resource.
- Only one non-XA participant may be involved with any transaction.

See “Emulating Two-Phase Commit” in the *Administration Console Online Help* for more details about this mode.

Q. How can I tell if a database is unavailable?

A. Fundamentally, there is no way to tell if a database has gone down except by trying to make a connection and failing.

Furthermore, a database can become unavailable at any time after you make and use a connection. We recommend that you write your code to handle unexpected failures, which can come in any form depending on what the client is doing when the database goes down.

WebLogic Server does provide the `dbping` utility to test the connection between WebLogic Server and your DBMS using a JDBC driver. See `dbping` in "Using the WebLogic Java Utilities" in the *Command Reference*.

Q. How do I configure the driver from Microsoft for SQL Server 2000?

A. See "Installing and Using the SQL Server 2000 Driver for JDBC from Microsoft" in *Programming WebLogic JDBC*.

Each instance of MS SQL Server must be listening on a different port. So, you can use the port number in the properties that you pass to the `getConnection()` method or, in case of connection pools, you can specify the port property in the following properties:

```
server=machine
Nameport=instancePort
```

To find the port number where each MS SQL Server instance is running, run the server network utility (in the Microsoft SQL Server program group), select the server instance, select TCP/IP, and click the properties button.

The full pathnames for the `msbase.jar`, `msutil.jar`, and `mssqlserver.jar` files must be in the in the `CLASSPATH` for the server startup script (e.g., `STARTWEBLOGIC.CMD`), not just in your `CLASSPATH`.

Also, it may be necessary to append connection properties to the URL for the connection pool in the administration console, separated by semicolons, rather than using the Properties field. For example:

```
jdbc:microsoft:sqlserver://127.0.0.1:1433;user=User;password=Password;data
basename=DBName
```

Note: BEA recommends using the WebLogic Type 4 JDBC Driver for Microsoft SQL Server to create connections to a Microsoft SQL Server database. For information about the WebLogic Type 4 JDBC Driver for Microsoft SQL Server, see *WebLogic Type 4 JDBC Drivers*.

Q. I have a long running transaction that makes multiple updates to an XA-compliant resource manager and a resource enlistment, prepare, or rollback operation fails with an XAException (errorCode=XAER_NOTA). What causes this behavior and how can it be prevented?

A. Some resource managers, such as Oracle, have an internal timeout mechanism that will internally roll back work associated with a transaction branch when the timeout period expires. Afterwards, when the WebLogic transaction manager invokes a XAResource.start/prepare/rollback operation on the resource, the Xid provided no longer exists within the resource manager so the resource manager responds with a XAException XAER_NOTA.

The javax.transaction.xa.XAResource interface provides a method, setTransactionTimeout, which in some driver implementations sets the resource's internal timeout interval. The WebLogic transaction manager can be instructed to invoke this method with a value equal to the global transaction timeout prior to each resource enlistment. For JDBC connection pool configurations, set the attribute "XASetTransactionTimeout" to "true" to enable this feature. Note that this JDBCConnectionPool attribute is only applicable for XA-compliant drivers. Also note that setting this attribute has no affect on XA drivers that do not implement the XAResource.setTransactionTimeout method.

For more details about setting the XASetTransactionTimeout attribute, see "Support for XAResource Transaction Timeout" in the *WebLogic Server Release Notes*.

The Oracle thin driver supports XAResource.setTransactionTimeout. The WebLogic jDriver for Oracle driver does not implement this method. To set the Oracle session timeout when using the WebLogic jDriver for Oracle, set the SesTm in the XA open string. The following is an example of what the XA open string might look like in the JDBCConnectionPool entry in the config.xml file:

```
<:JDBCConnectionPool
DriverName= "weblogic.jdbc.oci.xa.XADataSource"
Name="XAPool"
Properties="user=system;password=manager;
openString=Oracle_XA+Acc=p/system/manager+SesTm=177+db=SUPPORT+Threads=tru
e+Sqlnet=LINUX+logDir=.+dbgFl=0x15;dataSourceName=XAPool"
Targets="myserver" />
```

XAER_NOTA errors are also seen during recovery. These XAER_NOTA errors are thrown for transactions that have been committed before the server restart but still exist in the transaction log at the time the server was booted. During recovery processing, for each transaction record in the transaction log, the transaction manager will inform the participating resources of the commit decision. If the resource commit directive succeeded before the restart, the resource manager will respond to a subsequent commit with XAER_NOTA because it no longer has knowledge of the Xid.

The transaction manager ignores this error assuming that the commit succeeded before the crash. The reason why there are transaction log records that exist for transactions that have already completed is because the transaction manager only removes entries during checkpoint operations. A checkpoint occurs every five minutes by default and deletes transaction log files for which all records have been released. The checkpoint interval can be configured via the `JTAMBean.CheckpointIntervalSeconds` attribute. You can set the Checkpoint Interval in the administration console on the Domain →Configuration →JTA tab. See "Configuring Transactions" in the *Administration Console Online Help*.

Q. How do testConnsOnReserve and refreshTestMinutes work together?

A. They are almost independent. BEA recommends testing connections at reserve time, and *not* using the refresh testing option. Refresh testing may cause temporary exceptions when an application wants a connection and all the free connections are being tested by the refresh test. One value for using refresh testing is when there is a firewall or DBMS that kills connections if they stay idle too long. The refresh tests keep the connections all looking busy enough.

For more information about connection testing, see "Connection Testing Options" in the *Administration Console Online Help*.

Q. How do I use the prepared statement cache?

A. See "Increasing Performance with the Statement Cache" in the *Administration Console Online Help*.

There is also an article on using prepared statements at http://www.theserverside.com/resources/article.jsp?l=Prepared-Statements.

Q. Are there recommended programming practices for using JDBC connections?

A. The general rule is to get the connection from the connection pool as late as possible and give it back to the connection pool as soon as possible. Ensure that the connection is a method variable, and get and release the connection within the same method as where you use it (each thread should have its own connection). The cost of getting the connection is small, the prepared statement cache will reduce the preparation time, the set statements are small, the execute needs to be done no matter what the usage, and the close is small. It is not recommended to create the connection at ejbCreate/activate and close it on ejbRemove/pasivate.

Q. Why should I not use `DriverManager.getConnection`?

A. `DriverManager.getConnection` can cause a deadlock. In the server, all `DriverManager` calls are class-synchronized including many frequent calls that all drivers make, and JDBC drivers do a lot of synchronization internally. One long-waiting call can stop all JDBC work in the whole JVM and cause deadlocks. Also, you should not reregister the driver repeatedly. Regardless of the DBMS state, the one driver that is initially loaded at startup will always work.

Q. Can I use a prepared statement across multiple transactions?

A. Yes. Every transaction uses a dedicated JDBC connection, and all database interaction needs to use this connection object in order to participate in the transaction. So a prepared statement is tied to a particular connection and can't be shared with other connections. But a prepared statement *can* span transactions.

Q. Why do I get a `java.lang.AbstractMethodError` when calling a method on a driver?

A. This usually indicates that the driver has not implemented the method. For instance, you might be calling a JDBC 3.0 method on a driver that has only implemented the JDBC 2.0 methods.

Q. Why do I get "ResourceException: No Resource Available"?

A. One common reason is that you have too many consumers (connection users) for the number of configured JDBC connections in the connection pool or execute threads on the server.

Another reason may be that the refresh testing process has reserved one or more connections for testing so these connections are briefly unavailable.

Q. How do I ensure that a new database connection is created each time an EJB's container-managed transaction is started (so that I get a new authentication/authorization each time)?

A. The EJB should be tx-requires, which means it will start a transaction when called if one is not underway already, or will join the transaction in progress if there is one. Your code will use the standard JTS/JTA API to obtain and start a UserTransaction. Then you should obtain your JDBC connection from a tx data source, and it will be included in the transaction. To get a new connection each time, you could use the dynamic pool API to make a one-connection pool. We suggest configuring the server to have a one-connection pool and a tx data source for it at startup. Then when you want to do a transaction in an external client, you would destroy the initial pool and recreate it with the DBMS user you want. This will allow you to use the tx data source to get a connection, which if obtained in the context of a running UserTransaction, will get automatically included in the tx.

Q. When you look up a data source via JNDI and access a database connection from an external process, do you get a stub for the Connection instance in the WebLogic process or does it create a new connection pool with separate Connections in the local process?

A. If it is a WebLogic DataSource, then you get a stub for the Connection instance, not a connection pool in the local process.

Q. If a distributed transaction involves JMS and JDBC, how do I ensure that the JDBC update is available when the JMS message is processed?

A. The problem is that an application can receive the JMS message from the destination before the associated JDBC data is in the database.

Distributed transactions guarantee all involved changes will either succeed or fail as a unit, but cannot guarantee that they will happen exactly simultaneously (the transaction manager instructs all resource managers to commit but cannot control the timing of the completion of that operation).

For the WebLogic transaction manager, if the JDBC connection pool and the JMS server are both on the same server, and the transaction starts on the same server, the changes are committed in the order in which they were asked for by the transaction. This is not supported behavior, it just happens to be the current behavior. So if you can co-locate JMS and the JDBC connection pool, then you may have a chance.

You could send the JMS message with a delayed birth-time, and hope that this is good enough.

If the receiver fails to find the associated JDBC record, it could rollback/recover the message. You could use the WebLogic JMS redelivery delay feature to prevent the message from being redelivered instantly.

Q. If an application calls `DataSource.getConnection` multiple times in the same thread and transaction, will WebLogic Server handle giving me the same connection and transaction?

A. A common scenario might be to have multiple methods that are called within a transaction (begin/commit) that do something like the following:

```
Context ctx = new InitialContext();
DataSource ds = (javax.sql.DataSource) ctx.lookup("connpoll");
// work using Connection
```

In this case, all of the work will be done within the transaction and the same underlying JDBC connection will be used as long as the DataSource `ds` is a tx data source.

Q. Why do I get a SystemException failure when trying to enlist my XAResource in a client?

A. WebLogic Server does not allow you to register or enlist an XA resource on a client. The reason for this restriction is that a client is deemed to be less reliable than a server in terms of availability. This is also why a client is not allowed to act as a transaction coordinator and register Synchronization objects.

Your client could invoke a remote object on a server that accesses the resource within a transaction. If it's a JDBC resource, then you can configure a JDBCConnectionPool and JDBCTxDataSource using an Oracle XA driver (Oracle thin or WebLogic Type 4 driver for Oracle) and obtain a connection from the data source. Or the client could look up the data source using JNDI and retrieve and manipulate a connection in a transaction. Transaction enlistment is performed automatically.

Q. What causes an `OCIW32.dll` error?

A. You may see the following error message when using your JDBC driver for Oracle: "The ordinal 40 could not be loaded in the dynamic link library OCIW32.dll." This problem is caused by an out-of-date version of OCIW32.DLL in your system directory. Some programs install this file in the system directory in order to run. If you remove this file from the system directory you should no longer receive this error.

Q. How do I use Unicode codesets with the WebLogic jDriver for Oracle driver?

A. To use Unicode codesets:

1. Install the appropriate codeset when you install Oracle. If you did not do this in the original installation, you will need to re-run the Oracle installer and install the proper codeset.
2. Define the NLS_LANG variable in the environment where the JDBC driver is running. Do this by assigning the proper codeset to NLS_LANG in the shell from where you start the WebLogic Server.

The Oracle Developers Guide has more information about internationalization support. For general information about Unicode, see the Unicode Web site. For a list of Unicode language abbreviations, see the JavaSoft Web site.

Q. What type of object is returned by `ResultSet.getObject()` when using the WebLogic jDriver for Oracle?

A. WebLogic jDriver for Oracle always returns a Java object that preserves the precision of the data retrieved. It returns the following from the `getObject()` method:

- For columns of types NUMBER(n) and NUMBER(m,n): a Double is returned if the defined precision of the column can be represented by a Double; otherwise BigDecimal is returned.

- For columns of type NUMBER: Because there is no explicit precision, the Java type to return is determined based on the actual value in each row, and this may vary from row to row. An Integer is returned if the value has a zero-valued fractional component and the value can be represented by an integer.

For example, 1.0000 will be an integer. A long is returned for a value such as 123456789123.00000. If a value has a non-zero fractional component, a Double is returned if the precision of the value can be represented by a Double; otherwise a BigDecimal is returned.

Q. How do I limit the number of Oracle database connections generated by WebLogic Server?

A. You can use connection pools to limit the number of Oracle database connections generated by WebLogic Server in response to client requests. Connection pools allow T3 applications to share a fixed number of database connections. For information on how to set up connection pools, see *Programming WebLogic JDBC*.

Q. How do I call Oracle stored procedures that take no parameters?

A. Try this:

```
CallableStatement cstmt = conn.prepareCall("Begin procName;
    END;");
cstmt.execute();
```

where `procName` is the name of an Oracle stored procedure. This is standard Oracle SQL syntax that works with any Oracle DBMS. You can also use the following syntax:

```
CallableStatement cstmt = conn.prepareCall("{call procName};");
cstmt.execute();
```

This code, which conforms to the Java Extended SQL specification, will work with any DBMS, not just Oracle.

Q. How do I bind string values in a PreparedStatement?

A. Suppose you are trying to get the PreparedStatement class to bind Strings in a statement. The `setString()` method doesn't seem to work. Here is how you have set up the PreparedStatement:

```
String pstmt = "select n_name from n_table where n_name LIKE '?%'";
PreparedStatement ps = conn.prepareStatement(pstmt);
ps.setString(1, "SMIT");
ResultSet rs = ps.executeQuery();
```

The preceding code does not work because the complete value needs to be specified in a String (without using embedded quotes) and then bound to an unquoted question mark (?). Here is the corrected code:

```
String matchvalue = "smit%";
String pstmt = "select n_name from n_table where n_name LIKE ?";
PreparedStatement ps = conn.prepareStatement(pstmt);

ps.setString(1, matchvalue);
ResultSet rs = ps.executeQuery();
```

Q. Why do I get unexpected characters from 8-bit character sets in WebLogic jDriver for Oracle?

A. If you are using an Oracle database with an 8-bit character set on Solaris, make sure you set NLS_LANG to the proper value on the client. If NLS_LANG is not set, it defaults to a 7-bit ASCII character set, and tries to map characters greater than ASCII 128 to a reasonable approximation (for example, &aacute;, &agrave;, &acirc; would all map to a). Other characters are mapped to a question mark (?).

Q. How do I learn what codesets are available in Oracle?

A. To find out what codesets you currently have available in Oracle, execute the following SQL query from SQLPlus at the command line:

```
SQL> SELECT value FROM v$nls_valid_values
         WHERE parameter='CHARACTERSET';
```

The response lists all codesets currently installed on your system. This listing will look something like the following shortened list:

```
VALUE
------------------------------------------------------------
US7ASCII
WE8DEC
WE8HP
US8PC437
WE8EBCDIC37
WE8EBCDIC500
WE8EBCDIC285
...
```

If you want to constrain the value in the query to a specific codeset you are searching for, you can use a SQL query like the following:

```
SQL> SELECT value FROM v$nls_valid_values
         WHERE parameter='CHARACTERSET' and VALUE='AL24UTFFSS';
```

This would produce the following response if the codeset is installed:

```
VALUE
-------------------------------------------------------------
AL24UTFFSS
```

You can use Oracle's installation tools to install additional codesets. Contact Oracle for more information.

Q. How do I look up an "ORA" SQLException?

A. If your WebLogic jDriver for Oracle application produces an SQLException, you can look up the Oracle error by using the oerr command. For example, if you see the following SQLException:

```
java.sql.SQLException: ORA-12536: TNS: operation would block
```

You can find the description of error ORA-12536 can be found with the command:

```
> oerr ora 12536
```

Q. What is error "ORA-6502?"

A. The default length of a string bound to an OUTPUT parameter of a CallableStatement is 128 characters. If the value you assign to the bound parameter exceeds that length, you will get this error.

You can adjust the length of the value of the bound parameter by passing an explicit length with the scale argument to the `CallableStatement.registerOutputParameter()` method.

Q. Why do I get an error while trying to retrieve the text for ORA-12705?

A. This error occurs when you have not set the ORACLE_HOME environment variable properly. In order to use WebLogic jDriver for Oracle, the Oracle client software needs to be installed and ORACLE_HOME must be set.

You may also see this error message if you try to use WebLogic jDriver for Oracle's internationalization capabilities with a language/codeset combination that is not installed on your system. If you get the ORA-12705 error with the correct error text, then either you have set NLS_LANG improperly, or you do not have the right codesets installed on your system.

Q. Why do I run out of resources during updates with Oracle's database link?

A. When you use Oracle's database link to update your database, you may get error "maximum number of temporary table locks exceeded" even if you close your result sets and statements when you finish.

The database link is an object in the local database that allows you to access tables, views, and so forth in a remote database. The database link is controlled by the Oracle server, so the driver has no control over its use of resources. The link appears to perform the commit (since other processes could see the records that were being created), but it doesn't free any resources until the connection is closed. The solution is to remove the database link and use the JDBC driver to do your selects, inserts, and updates.

Q. Why does executing the PreparedStatement class cause a "TRUNC fails: ORA-00932: inconsistent datatypes" error?

A. According to Oracle Metalink Bug Database Doc ID: 144784.1, in the absence of explicit data typecasting, OCI assumes that a bind variable is a CHAR data type. If the SQL statement intends to use the bind variable as a DATE data type, but OCI thought it was a CHAR, the SQL parser will have a conflict in data types. The fix is to explicitly use data conversion functions to convert the bind variables in the problem queries. For example, a select string of

```
String st = "select count(*) from simple_table where TRUNC(mydate) =
TRUNC(?)";
```

should be changed to:

```
String st = "select count(*) from simple_table where TRUNC(mydate) =
TRUNC(TO_DATE(?))";
```

Q. How do I create and update Oracle Blob fields?

A. The following code sample shows how to create and update Oracle Blob fields.

```
public void insert() throws SQLException {
  try {

    // Connect to the database using WebLogic JDBC connection pool
    Hashtable ht = new Hashtable();
    ht.put(Context.INITIAL_CONTEXT_FACTORY,
      "weblogic.jndi.WLInitialContextFactory");
    ht.put(Context.PROVIDER_URL, "t3://localhost:7001");
    InitialContext ctx = new InitialContext(ht);
    javax.sql.DataSource ds = (javax.sql.DataSource)
  ctx.lookup("java:comp/env/jdbc/DSName");
Connection conn = ds.getConnection();

// This is necessary in any case where you are doing
// a select for update. Not doing this will result in a ORA-1002
conn.setAutoCommit (false);

BLOB blob = null;

// Create a Statement
Statement stmt = conn.createStatement ();

// Drop the table if it exists
try {
  stmt.execute ("drop table ImageTable");
  System.out.println("Table droped ...");
}
catch (SQLException e) {
  System.out.println("Table does not exist");
}

// Create the table
stmt.execute ("create table ImageTable (column1 varchar2(20),
  image BLOB)");
System.out.println("Table created ...");

// create a blob entry in the table
stmt.execute("insert into ImageTable values ('one', empty_blob())");
stmt.execute("commit");
System.out.println("inserted empty blob");
String cmd = "select * from ImageTable for update";
ResultSet rset = stmt.executeQuery(cmd);
if (rset.next())  {
      blob = ((OracleResultSet)rset).getBLOB(2);
      System.out.println("got blob reference");
}

else System.out.println("no row to get!!!!");

rset.close();

blob = readFromFile();
```

```
    cmd = "update ImageTable set image = ? where column1 = 'one'";
    PreparedStatement pstmt = conn.prepareStatement(cmd);
    pstmt.setBlob(1, blob);
    pstmt.execute();
    stmt.execute("commit");
    System.out.println("blob updated");
    blob = null;
    cmd = "select * from ImageTable for update";
    rset = stmt.executeQuery(cmd);
    if (rset.next())  {
      System.out.println("get blob");
      blob = ((OracleResultSet)rset).getBLOB(2);
      // do something with blob
    }
    else
      System.out.println("no row to get (2)!!!!");
  }
  catch (SQLException sqle) {
    System.out.println("SQL Exception occured: " + sqle.getMessage());
    sqle.printStackTrace();
  }
  catch(FileNotFoundException e) {
    System.out.println("File Not Found");
  }
  catch (IOException ioe) {
    System.out.println("IO Exception" + ioe.getMessage());
  }
  catch (Exception ioe) {
    System.out.println("Exception" + ioe.getMessage());
  }
}
```

Q. How do I enlist an Oracle XAResource?

A. This code will only work on the server side. It cannot be run in a client. Also note that enlistment is generally done transparently for JDBC resources that implement XAResource.

```
// Here is the XAResource for oracle
String URL = "jdbc:oracle:thin:@DbmsHost:DbmsPort:DbmsName";
DriverManager.registerDriver(new OracleDriver());

// Create XA Connection
OracleXADataSource oxds1 = new OracleXADataSource();
oxds1.setURL(URL);
oxds1.setUser("scott");
oxds1.setPassword("tiger");
javax.sql.XAConnection pc1  = oxds1.getXAConnection();
m_oracleResource = pc1.getXAResource ();

m_oracleConnection = pc1.getConnection();

// Here is the source code for getting the TM.
Context ctx = null;
Hashtable env = new Hashtable();
env.put(Context.INITIAL_CONTEXT_FACTORY,
"weblogic.jndi.WLInitialContextFactory");
// Parameters for the WebLogic Server.
// Substitute the correct hostname, port number
// user name, and password for your environment:
env.put(Context.PROVIDER_URL, "t3://localhost:7001");
env.put(Context.SECURITY_PRINCIPAL, "system");
env.put(Context.SECURITY_CREDENTIALS, "managers");
ctx = new InitialContext(env);
m_tManager =
(TransactionManager)ctx.lookup("javax.transaction.TransactionManager");

//Here is the source code for registering the TM.
m_tManager.begin();
m_transaction = m_tManager.getTransaction();
weblogic.transaction.TransactionManager weblogicTManager =
((weblogic.transaction.TransactionManager) m_tManager);
weblogicTManager.registerStaticResource("oracle",m_oracleResource);

// enlist the resources with the transaction
m_transaction.enlistResource(m_oracleResource);
```

Q. Why do I get "ORA-00600" ?

A. This error generally means that version of Oracle server is newer than version of the driver you are using. In case you are using the Oracle thin driver, you will need to download the latest `ojdbc14.jar` from Oracle and put it at the beginning of your CLASSPATH (and possibly update any scripts that start the server, such as `startweblogic.cmd`, since they override the CLASSPATH).

Q. Why do I get "ORA-24327" ?

A. This error generally means that the environment ORACLE_HOME is not set or is set incorrectly or the D_LIBRARY_PATH or PATH does not include the right dynamic link libraries. It can also indicate a mismatch when trying to use `weblogic.jdbc.oci.Driver` with an earlier or later version of the Oracle client software than is supported. In that case, try to use the Oracle Thin driver instead.

Q. Why do I get "java.sql.SQLException: getOBJECT is not supported by the WebLogic JDBC Driver"?

A. When using the WebLogic JDBC connection pool and `weblogic.jdbc.vendor.oracle.OracleResultSet`, the error is returned (where OBJECT is the name of some Oracle object). It implies that this feature is not supported by WebLogic Server JDBC because the object type is not serializable. There are two alternatives.

- You can switch to using the Oracle thin driver directly. That means that you will get a connection directly to the database using the Thin driver instead of getting the connection from a pool of JDBC connections. That means that you lose all advantages of using the WebLogic Server JDBC subsystem, such as transactions, connection pooling, and caching of prepared statements.
- BEA recommends moving your processing to a stored procedure.

Q. Why do I get an "ORA-01453" when I use SET TRANSACTION?

A. When using Oracle, the message "java.sql.SQLException: ORA-01453: SET TRANSACTION must be first statement of transaction" may be logged. This is due to a limitation in the Oracle interfaces, starting in Oracle 8.1.7. WebLogic tries to minimize the problem by not calling SET TRANSACTION if the mode already matches the current state.

FAQs: WebLogic jDriver for MSSQL Server

- Can the Weblogic JDriver for MSSSQL Server connect to the database server using a trusted connection on NT/WIN2K?
- How do I connect to an SQL Server instance that is running on a machine with multiple instances of SQL Server 2000?

Q. Can the Weblogic JDriver for MSSSQL Server connect to the database server using a trusted connection on NT/WIN2K?

A. Our driver doesn't support trusted connections.

Q. How do I connect to an SQL Server instance that is running on a machine with multiple instances of SQL Server 2000?

A. Each instance of MS SQL Server must be listening on a different port. So, you can use the port number in the properties that you pass to the `getConnection()` method or, in case of connection pools, you can specify the port property in the following properties:

```
server=machineName
port=instancePort
```

To find the port number where each MS SQL Server instance is running, run the server network utility (in the Microsoft SQL Server program group), select the server instance, select TCP/IP, and click the properties button.

FAQs: WebLogic jDriver for Oracle

- Why does FOR UPDATE in Oracle 8 cause an ORA-01002 error?
- What causes an OCIW32.dll error?
- What transaction isolation levels does the WebLogic jDriver for Oracle support?
- How do I use Unicode codesets with the WebLogic jDriver for Oracle driver?
- How do I use OS Authentication with WebLogic jDriver for Oracle and Connection Pools?
- What type of object is returned by ResultSet.getObject()?
- How do I limit the number of Oracle database connections generated by WebLogic Server?
- How do I call Oracle stored procedures that take no parameters?
- How do I bind string values in a PreparedStatement?
- Why do I get unexpected characters from 8-bit character sets in WebLogic jDriver for Oracle?
- How do I learn what codesets are available in Oracle?
- How do I look up an "ORA" SQLException?
- What is error “ORA-6502?”
- Why do I get an error while trying to retrieve the text for ORA-12705?

- Why do I get error "ORA-03120" when trying to access multi-byte characters from the CLOB/NCLOB column?
- Why does executing the PreparedStatement class cause a "TRUNC fails: ORA-00932: inconsistent datatypes" error?
- Why do I get an “ORA-01400: Cannot insert NULL into column name” when inserting a blank string?

Q. Why does FOR UPDATE in Oracle 8 cause an ORA-01002 error?

A. The Oracle 8 server generates an `ORA-01002:fetch out of sequence` error message when you use a FOR UPDATE statement with AUTOCOMMIT turned `on` (which is the default state when using JDBC). This is known to happen on Oracle 8.0 and 8.1 on Solaris and on Oracle 8.1 on Windows NT. If you turn AUTOCOMMIT `off`, you will not receive this error. Because this problem is due to a change in the Oracle 8 server, you should contact Oracle support for more information.

Q. What causes an OCIW32.dll error?

A. You may receive the following error message when using your JDBC driver for Oracle: "The ordinal 40 could not be loaded in the dynamic link library OCIW32.dll." This problem is caused by an out-of-date version of OCIW32.DLL in your system directory. Some programs install this file in the system directory in order to run. If you remove this file from the system directory you should no longer receive this error.

Q. What transaction isolation levels does the WebLogic jDriver for Oracle support?

A. Your servlet application may use Oracle Thin Drivers to access a database that includes BLOB fields. If you install and try to use WebLogic jDriver for Oracle and the same code fails and produces an exception similar to the following:

```
com.roguewave.jdbtools.v2_0.LoginFailureException:
TRANSACTION_READ_UNCOMMITTED isolation level not allowed
  The Stack Trace:
com.roguewave.jdbtools.v2_0.LoginFailureException:
TRANSACTION_READ_UNCOMMITTED isolation level not allowed
  at
  com.roguewave.jdbtools.v2_0.jdbc.JDBCServer.createConnection
  (JDBCServer.java :46)
  at com.roguewave.jdbtools.v2_0.ConnectionPool.getConnection_
```

```
(ConnectionPool.jav a:412)
at com.roguewave.jdbtools.v2_0.ConnectionPool.getConnection
(ConnectionPool.java :109)
```

Setting the `Isolation_level` to 1 in the code that calls the RogueWave JDBCServer class works with the Oracle thin driver but fails with WebLogic jDriver for Oracle.

WebLogic jDriver for Oracle supports the following transaction isolation levels:

```
SET TRANSACTION ISOLATION LEVEL READ COMMITTED
SET TRANSACTION ISOLATION LEVEL SERIALIZABLE
```

According to the Oracle documentation, the Oracle DBMS only supports these two isolation levels. Unlike other JDBC drivers, WebLogic's drivers throw an exception if you try to use an isolation level that is unsupported. Some drivers silently ignore attempts to set an unsupported isolation level. WebLogic suggests testing whether the Oracle thin driver is not just ignoring settings for unsupported isolation events.

Q. How do I use Unicode codesets with the WebLogic jDriver for Oracle driver?

A. To use Unicode codesets:

1. Install the appropriate codeset when you install Oracle. If you did not do this in the original installation, you will need to re-run the Oracle installer and install the proper codeset.
2. Define the NLS_LANG variable in the environment where the JDBC driver is running. Do this by assigning the proper codeset to NLS_LANG in the shell from where you start the WebLogic Server.

The Developers Guide has more information about internationalization support. For general information about Unicode see the Unicode Web site at *http://www.unicode.org*. For a list of Unicode language abbreviations, see the JavaSoft Web site at http://java.sun.com/products/jdk/1.2/docs/guide/internat/encoding.doc.html.

Q. How do I use OS Authentication with WebLogic jDriver for Oracle and Connection Pools?

A. Using OS authentication in connection pools essentially means that you are using the UserId of the user who started WebLogic Server. OS authentication is available on Windows and UNIX. This means that database security will rely strictly on the security of WebLogic; that is, if you are allowed to make a client connection to the WebLogic Server and access the pool, then you can get to the database.

You can do this with WebLogic jDriver for Oracle because Oracle uses the process owner to determine who is attempting the connection. In the case of WebLogic JDBC, this is always the user that started the WebLogic Server.

To set up your Oracle instance to use this feature, your DBA needs to follow these basic steps. The full procedure is described in more detail in your Oracle documentation.

1. Add the following line to the INIT[sid].ORA file:

   ```
   OS_AUTHENT_PREFIX = OPS$
   ```

   Note that the string "OPS$" is arbitrary and up to the DBA.

2. Log in to the Oracle server as SYSTEM.

3. Create a user named `OPS$`*`userid`*, where *userid* is some operating system login ID. This user should be granted the standard privileges (for example, CONNECT and RESOURCE).

4. Once the userid is set up, you can connect with WebLogic jDriver for Oracle by specifying "/" as the username property and "/" as the password property. Here is an example for testing this connection with the `dbping` utility:

```
 $ java utils.dbping ORACLE "/" "" myserver
```

Here is a code example for WebLogic jDriver for Oracle:

```
Properties props = new Properties();
props.put("user",       "/");
props.put("password",   "/");
props.put("server",     "myserver");

Class.forName("weblogic.jdbc.oci.Driver").newInstance();
Connection conn = myDriver.connect("jdbc:weblogic:oracle",
   props);
```

5. Use the Administration Console to set the attribute for your connection pool. The following code is an example of a JDBC connection pool configuration using the WebLogic jDriver for Oracle:

```
<JDBCConnectionPool
    Name="myPool"
    Targets="myserver,server1"
    DriverName="weblogic.jdbc.oci.Driver"
    InitialCapacity="1"
```

```
MaxCapacity="10"
CapacityIncrement="2"
Properties="databaseName=myOracleDB"
```

Q. What type of object is returned by ResultSet.getObject()?

A. WebLogic jDriver for Oracle always returns a Java object that preserves the precision of the data retrieved. WebLogic jDriver for Oracle returns the following from the `getObject()` method:

- For columns of types NUMBER(*n*) and NUMBER(*m,n*): a Double is returned if the defined precision of the column can be represented by a Double; otherwise BigDecimal is returned.
- For columns of type NUMBER: Because there is no explicit precision, the Java type to return is determined based on the actual value in each row, and this may vary from row to row. An Integer is returned if the value has a zero-valued fractional component and the value can be represented by an integer.

For example, 1.0000 will be an integer. A long is returned for a value such as 123456789123.00000. If a value has a non-zero fractional component, a Double is returned if the precision of the value can be represented by a Double; otherwise a BigDecimal is returned.

Q. How do I limit the number of Oracle database connections generated by WebLogic Server?

A. You can use connection pools to limit the number of Oracle database connections generated by WebLogic Server in response to client requests. Connection pools allow T3 applications to share a fixed number of database connections. For information on how to set up connection pools, see Programming WebLogic JDBC at http://e-docs.bea.com/wls/docs81/jdbc.html.

Q. How do I call Oracle stored procedures that take no parameters?

A. Here is what we use that works:

```
CallableStatement cstmt = conn.prepareCall("Begin procName;
    END;");
cstmt.execute();
```

where *procName* is the name of an Oracle stored procedure. This is standard Oracle SQL syntax that works with any Oracle DBMS. You might also use the following syntax:

```
CallableStatement cstmt = conn.prepareCall("{call procName};");
cstmt.execute();
```

This code, which conforms to the Java Extended SQL spec, will work with any DBMS, not just Oracle.

Q. How do I bind string values in a PreparedStatement?

A. Suppose you are trying to get the PreparedStatement class to bind Strings in a statement. The `setString()` method doesn't seem to work. Here is how you have set up the PreparedStatement:

```
String pstmt = "select n_name from n_table where n_name LIKE '?%'";
PreparedStatement ps = conn.prepareStatement(pstmt);
ps.setString(1, "SMIT");
ResultSet rs = ps.executeQuery();
```

The preceding code does not work because the complete value needs to be specified in a String (without using embedded quotes) and then bound to an unquoted question-mark (?). Here is the corrected code:

```
String matchvalue = "smit%";
String pstmt = "select n_name from n_table where n_name LIKE ?";
PreparedStatement ps = conn.prepareStatement(pstmt);

ps.setString(1, matchvalue);
ResultSet rs = ps.executeQuery();
```

Q. Why do I get unexpected characters from 8-bit character sets in WebLogic jDriver for Oracle?

A. If you are using an Oracle database with an 8-bit character set on Solaris, make sure you set `NLS_LANG` to the proper value on the client. If `NLS_LANG` is unset, it defaults to a 7-bit ASCII character set, and tries to map characters greater than ASCII 128 to a reasonable approximation (for example, &aacute;, &agrave;, &acirc; would all map to a). Other characters are mapped to a question mark (?).

Q. How do I learn what codesets are available in Oracle?

A. To find out what codesets you currently have available in Oracle, execute the following SQL query from SQLPlus at the command line:

```
SQL> SELECT value FROM v$nls_valid_values
         WHERE parameter='CHARACTERSET';
```

The response lists of all codesets currently installed on your system. This listing will look something like the following shortened list:

```
VALUE
------------------------------------------------------------
US7ASCII
WE8DEC
WE8HP
US8PC437
WE8EBCDIC37
WE8EBCDIC500
WE8EBCDIC285
...
```

If you want to constrain the value in the query to a specific codeset you are searching for, you might use a SQL query like the following:

```
SQL> SELECT value FROM v$nls_valid_values
         WHERE parameter='CHARACTERSET' and VALUE='AL24UTFFSS';
```

This would produce the following response if the codeset is installed:

```
VALUE
--------------------------------------------------------------
AL24UTFFSS
```

You can use Oracle's installation tools to install additional codesets. Contact Oracle at *http://www.oracle.com/* for more information.

Q. How do I look up an "ORA" SQLException?

A. If your WebLogic jDriver for Oracle application produces an SQLException such as:

```
java.sql.SQLException: ORA-12536: TNS: operation would block
```

You can look up an Oracle error by using the `oerr` command. For example, the description of error ORA-12536 can be found with the command:

```
> oerr ora 12536
```

Q. What is error "ORA-6502?"

A. The default length of a string bound to an `OUTPUT` parameter of a CallableStatement is 128 characters. If the value you assign to the bound parameter exceeds that length, you will get this error.

You can adjust the length of the value of the bound parameter by passing an explicit length with the scale argument to the CallableStatement.registerOutputParameter() method.

Q. Why do I get an error while trying to retrieve the text for ORA-12705?

A. This error occurs when you have not set the ORACLE_HOME environment variable properly — a common mistake. In order to use WebLogic jDriver for Oracle, the Oracle client software needs to be installed and ORACLE_HOME must be set.

You may also see this error message if you try to use WebLogic jDriver for Oracle's internationalization capabilities with a language/codeset combination that is not installed on your system. If you get the ORA-12705 error with the correct error text, then either you have set NLS_LANG improperly, or you do not have the right codesets installed on your system.

Q. Why do I get error "ORA-03120" when trying to access multi-byte characters from the CLOB/NCLOB column?

A. Accessing exceeded characters would result in ORA-03120 error when getting the length of the clob from the CLOB/NCLOB column returns a bigger value than the actual length from the OCI layer. Using Oracle 8.1.6.3 solves this problem.

Q. Why does executing the PreparedStatement class cause a "TRUNC fails: ORA-00932: inconsistent datatypes" error?

A. According to Oracle Metalink Bug Database Doc ID: 144784.1, in the absence of explicit data typecasting, OCI assumes that a bind variable is a CHAR data type. If the SQL statement intends to use the bind variable as a DATE data type, but OCI thought it was a CHAR, the SQL parser will have a conflict in data types. The fix is to explicitly use data conversion functions to convert the bind variables in the problem queries. For example, a select string of

```
String st = "select count(*) from simple_table where TRUNC(mydate) =
TRUNC(?)";
```

should be changed to:

```
String st = "select count(*) from simple_table where TRUNC(mydate) =
TRUNC(TO_DATE(?))";
```

Q. Why do I get an "ORA-01400: Cannot insert NULL into *column name*" when inserting a blank string?

A. This is a known Oracle issue. When inserting or updating a value for a varchar2, if you try to insert an empty string (""), Oracle interprets the value as NULL. If there is a NOT NULL restriction on the column in which you are inserting the value, the database throws the ORA-01400 error.

FAQs: JMS

- Which types of JDBC databases does WebLogic JMS support?
- How do I use a third-party JDBC driver with WebLogic JMS?
- What if my JDBC database becomes corrupt?
- How do I use persistence?
- How does a file store compare with a JDBC store?

Administration

- How important is it to keep the system clocks synchronized among server instances hosting distributed destination members and their connection factories?
- Why am I getting “out of memory” errors?
- What is the value of clustering for WebLogic JMS?
- How can I control on which WebLogic Server(s) my application will run?
- How do I perform a manual fail-over?
- Does the WebLogic JMS server find out about closed or lost connections, crashes, and other problems and does it recover from them?
- How does an application know if an application server goes down?
- Do I need to use the WLS T3 protocol?
- How do I use HTTP tunneling?
- Does WebLogic JMS support SSL?
- How do I integrate non-WebLogic JMS providers with WLS?

Transaction Support

- How do two-phase or global transactions relate to WebLogic JMS?
- Why is my WebLogic JMS work not part of a user transaction (that is, called within a transaction but not rolled back appropriately)? How do I track down transaction problems?
- When do WebLogic JMS operations take place as part of a transaction context?
- How can an application do a JMS operation and have it succeed, independent of the result of the transaction?

- What happens if acknowledge() is called within a transaction?
- Why am I getting JDBC XA errors when using JMS in conjunction with JDBC calls?
- Can I use a one-phase commit if my WebLogic JMS JDBC store is on the same database for which I am doing other database work?
- How do I integrate another vendor’s XAResource with WLS to get JTA transactions with another resource manager?
- Why do I get an exception when I start up WebLogic JMS using an XA driver or with a TX data source?
- Is WL JMS XAResource compliant?
- Why can’t I receive a response to a message that I send within a transaction?
- What happens to a message that is rolled back or recovered?

JMS Programming Practices

- Is it possible to set aside a message and acknowledge it later?
- How should I use sorted queues or topics?
- How do I deal with a listener that doesn't keep up with messages being sent?
- How do I get a thread dump to help track down a problem?
- Do client identifiers need to be unique?
- How do I manage a queue to view and delete specific messages?
- In what order are messages delivered to a consumer?
- How do I ensure message ordering even in the event of rollbacks and recoveries?
- Is it possible to have multiple queue receivers listening on the same queue?
- Is there a way to make a queue such that if one application has one object as listener on that queue, no other application can listen to the messages on that queue?
- Why doesn't setting values work using javax.jms.Message.setJMSPriority, DeliveryMode, Destination, TimeStamp or Expiration?
- What care must be taken when multi-threading WebLogic JMS clients?
- How should an application be set up to subscribe to multiple topics?

- How should I use blocking and asynchronous receive() calls?
- What precautions should I take when I use blocking receive() calls?
- What is the NO_ACKNOWLEDGE acknowledge mode used for?
- When should I use multicast subscribers?
- When should I use server session pools and connection consumers?
- How do I issue the close() method within an onMessage() method call and what are the semantics of the close() method?
- How do I publish an XML message?
- How do I use WebLogic JMS in an applet?
- How do I use a startup class to initialize and later reference WebLogic JMS objects?
- Is it possible to send or receive a message from within a message listener?
- How do I create a producer pool?
- What are pending messages in the console?
- How do I use a less than or greater than on a message selector in ejb-jar.xml?
- Can I use another vendor's destination with a WebLogic JMS API?
- What is the standard way to create threads, do initialization, etc. within the application server?
- Why do I get a JNDI problem when I name a Topic A.B and a second Topic A.B.C?
- What should an XPATH selector look like?
- How do I handle request/response using WebLogic JMS?
- Is it okay to add new sessions and subscribers to a Queue or Topic Connection once it has been started?
- What can I do when I get java.lang.OutOfMemoryError because producers are faster than consumers?
- How should connections and sessions be allocated?
- Is there a way to dynamically change an existing selector for a TopicConsumer using the setMessageSelecter(String)?

- How can I avoid asynchronous message deadlocks?

Message-Driven Beans

- What are the advantages of message-driven beans?
- How does concurrency work for message-driven beans?
- Can an MDB be a message producer or both a producer and consumer?
- If an MDB uses a durable subscription, will messages be accumulated if the MDB is not deployed?
- How do I use non-WebLogic JMS provider destinations to drive MDBs?
- Can you use a foreign JMS provider to drive an MDB transactionally?
- How do I roll back a transaction within an MDB?
- How do server session pools and message driven beans compare?

Q. What makes WebLogic JMS unique?

A. There are numerous features that make WebLogic JMS unique. For a complete listing, see "Introduction to WebLogic JMS" in *Programming WebLogic JMS*.

Q. Where can I learn more about WebLogic JMS?

A. The following links provide more information about WebLogic JMS:

- The WebLogic JMS index page
- *Programming WebLogic JMS*
- "Configuring JMS" and "Tuning JMS" in the *Administration Console Online Help*
- "The WebLogic Messaging Bridge" in the *Administration Console Online Help*
- BEA's dev2dev web site
- Ask BEA
- The WebLogic JMS "weblogic.developer.interest.jms" newsgroup available on the BEA Newsgroup server.

Q. Is there a C/C++ interface to WebLogic JMS?

A. Yes, there is a JMS C client available on the dev2dev Utility and Tools page, which has a downloadable `jmscapi.zip` file that includes all the necessary files, as well as documentation and samples. This is *not* a supported product of BEA. However, if you have questions about this API you can post them to WebLogic JMS "weblogic.developer.interest.jms" newsgroup available on the BEA Newsgroup server.

Q. Is there a smaller version of the `weblogic.jar` file for supporting clients?

A. Yes. WebLogic Server 8.1 provides a true J2EE application client. The WebLogic Server application client is provided as a standard client and a JMS client, packaged as two separate jar files—`wlclient.jar` and `wljmsclient.jar`—in the `/server/lib` subdirectory of the WebLogic Server installation directory. Each jar is about 400 KB.

- For instructions on developing a thin client, see "Developing a J2EE Application Client (Thin Client)" in *Programming WebLogic RMI over IIOP*.
- For information about the JMS jar, see "WebLogic JMS Thin Client" in *Programming WebLogic JMS*.
- For an overview of client options, see "Using RMI over IIOP Programming Models to Develop Applications" in *Programming WebLogic RMI over IIOP*.

Q. How do I start WebLogic Server and configure JMS?

A. Refer to "Starting WebLogic Server and Configuring JMS" in the *Programming WebLogic JMS* for detailed instructions on starting WebLogic Server, accessing the Administration Console, and configuring a basic Weblogic JMS implementation.

Q. How do I configure WebLogic JMS security?

A. A security policy is created when you define an association between a WebLogic resource and a user, group, or role. A WebLogic resource has no protection until you assign it a security policy. You can assign a security policy to any WebLogic JMS destination using the administration console.

Using the navigation tree, access your JMS destinations, which are under Services →JMS → Servers →*<server name>* →Destinations. Right-click a destination, and then select *Define policy* from the pop-up menu. By default, the console screen sets a policy for all operations on each destination. You may also set separate policies for the `send()`, `receive()`, and `browse()` operations on the destination using the list box labeled *Methods*.

For instructions on how to set up security for all WebLogic Server resources, see "Securing WebLogic Resources".

Q. Can I still use the default connection factories supported in WebLogic JMS 5.1?

A. Yes. For detailed information about using 5.1 connection factories in later versions of WebLogic JMS, see "Porting WebLogic JMS Applications" in *Programming WebLogic JMS*.

Q. Why does JMSSession.createTopic or JMSSession.createQueue fail to create a destination in WebLogic JMS 8.1? (It worked in version 5.1?)

A. For a detailed explanation of this issue, refer to the "JMS FAQ" in the version 6.1 *Frequently Asked Questions*.

Q. How do I programmatically get a list of queues or topics?

A. There are JMS Helper methods that allow you to locate JMS runtime and configuration JMX MBeans. There are also methods for dynamically creating and deleting JMS queue and topic destinations, as described in the JMS Helper Method Javadoc.

Q. How do I use a temporary destination?

A. You must create a template on every `JMSServer` where you want to be able to create temporary destinations. You can specify multiple JMSServer entries to support a Temporary Template and the system will load balance among those JMSServers to set up the temporary destination. See "How do I start WebLogic Server and configure JMS?" on page 14-6 for a description about how to configure JMS. The resulting template definition looks something like the following:

```
<JMSTemplate  Name="MyTemplate"/>
```

The `JMSServer` is defined something like:

```
<JMSServer Name="MyJMSServer" TemporaryTemplate="MyTemplate"
Targets="MyServer" >
```

After the template name, you can set any queue/topic attribute you want in the template (not including a JNDI name or topic multicast settings). The template is at the outer most level; that is, it should not be nested in your `<JMSServer>`.

Temporary destinations can only be consumed by the creating connection. Using topics, you create your temporary topic and subscribe to that temporary topic. If you want someone to publish to that temporary topic, you need to tell that someone what your topic is. You can send them a message and include your temporary topic in the JMSReplyTo field. The creator of the TemporaryTopic and the subscriber must be one in the same.

```
import javax.jms.TopicSession;
TemporaryTopic myTopic = mySession.createTemporaryTopic();
TopicSubscriber = mySession.createSubscriber(myTopic);
```

Temporary topics do not get names and cannot be subscribed to by other connections. When you create a temporary topic, the JMS provider returns a `javax.jms.Topic`. You then need to advertise that topic to other parties (those who want to publish to the topic), putting it in your `JMSReplyTo` field so that they can respond. In general, no one else can subscribe to the topic. You advertise the topic any way you want. Topics are Serializable (or in our case, Externalizable), which allows you to pass them around in RMI calls, through a file, binding it to a name in JNDI, etc. In short, create the topic at the subscriber side and advertise so that others can publish. You can get multiple subscribers on the same connection and get concurrent processing using multiple sessions.

For more information about using temporary destinations, see "Using Temporary Destinations" in *Programming WebLogic JMS*.

Q. How do I use MBeans to print runtime statistics?

A. BEA's dev2dev Web site contains a "JMS Statistics View" program to print JMS statistics based on run-time MBeans. Also, there are JMS Helper methods that allow you to access run-time statistics for JMS connection, destination, consumer, and producer MBeans, as described in the JMS Helper Method Javadoc.

Q. Can two JMS servers share the same persistent store?

A. No. Each JMS server must have its own unique persistent store. Two file-based JMS persistent stores may share the same directory, but their messages will be stored in different files. In this case, the filenames will contain different prefixes.

Two JDBC-based JMS persistent stores may share the same database, but they must be configured to use a different Prefix Name which will be prepended to the database tables. For more information on configuring the JDBC Prefix Name, see "Using Prefixes With JMS JDBC Stores" in the *Administration Console Online Help*. If they are configured with the same Prefix Name, persistent messages will be corrupted and/or lost.

Q. Which types of JDBC databases does WebLogic JMS support?

A. The JMS database can be any database that is accessible through a JDBC driver. For a list of drivers that WebLogic JMS detects, see “JMS JDBC Store Tasks” in the *Administration Console Online Help*.

Q. How do I use a third-party JDBC driver with WebLogic JMS?

A. If your JDBC driver is not included in the list of drivers in the question about JDBC databases supported by WebLogic JMS, then the tables required by JMS must be created manually. Follow the procedures in JDBC Database Utility in *Programming WebLogic JMS* to manually create the database tables for the JDBC store.

Note: WebLogic Server only guarantees support for the JDBC drivers listed in “JMS JDBC Stores Tasks” in the *Administration Console Online Help*. Support for any other JDBC driver is not guaranteed.

Another option is to consider using a JMS file store instead of a JMS JDBC store. File stores are easier to configure and may provide significantly better performance.

Q. What if my JDBC database becomes corrupt?

A. The procedures for removing and regenerating the JDBC store tables or creating the database tables manually are described in detail in JDBC Database Utility in *Programming WebLogic JMS*.

Q. How do I use persistence?

A. Use the following guidelines:

1. Make sure the JMSServer you are using has a store configured. The JMSServer configuration entry in the `config.xml` file should contain a line of the form

   ```
   Store="<YOUR-STORE-NAME>"
   ```

   Note that if JMS boots without a store configured, it is assumed the customer did not want one, and persistent messages are silently downgraded to non-persistent (as specified for JMS 1.0.2b).

2. Make sure you are not using "Message.setJMSDeliveryMode". This is overwritten, as it is a vendor-only method.

3. Make sure you are calling either:

   ```
   QueueSender.send(msg, deliveryMode, ...)
   ```

   -- or --

`QueueSender.setDeliveryMode(deliveryMode)`

-- or --

set `DefaultDeliveryMode` mode on connection factory in the `config.xml` file to persistent (the `QueueSender.setDeliver/send` overrides this value). Similarly, for topics, you would set this via the `TopicPublisher`.

4. Make sure you don't have "DeliveryModeOverride" set to Non-Persistent on the Destination in the `config.xml` file.
5. If you are using `pub/sub`, only durable subscriptions persist messages. Non-durable subscriptions have no need to persist messages, as by definition they only exist for the life of the server.

See the question, "How do I start WebLogic Server and configure JMS?" on page 14-6 for a description of how to configure JMS.

Q. How does a file store compare with a JDBC store?

A. The are a number of similarities and differences between file stores and JDBC stores. For a complete listing, see "JMS Stores Tasks" in the *Administration Console Online Help*.

Q. How important is it to keep the system clocks synchronized among server instances hosting distributed destination members and their connection factories?

A. It is very important when using distributed topics with non-durable subscribers. This is because if the clocks between the servers become too far askew, there is a possibility that messages will not be delivered. Here's how this could happen: both the connection factory for the distributed topic and the distributed topic members will use their local system clock to see if a consumer is created after a message is published. Messages published to topics before consumers are created are not visible to consumers. There is always a race in any topic when the message is published before the consumer is created.

Distributed topics widen this race when the system clocks on the server instances hosting the connection factory or the distributed topic members are not in sync. For example, if your application creates short-lived, non-durable topic consumers, and a consumer is listening through a connection factory on *ServerA*, but the message is published to distributed topic member on *ServerB*, and the clocks from *ServerA* and *ServerB* are out of sync (more than the life of the topic consumer you have created), then that consumer will not receive the message sent due to the difference in the system clocks.

Q. Why am I getting "out of memory" errors?

A. The byte and message maximum values are quotas – not flow control. Message quotas prevent a WebLogic JMS server from filling up with messages and possibly running out of memory, causing unexpected results. Unless the "Blocking Sends" feature has been implemented, when you reach your quota, JMS prevents further sends with a `ResourceAllocationException` (rather than blocking). You can set quotas on individual destinations or on a server as a whole. For more information on configuring the "Blocking Sends" feature, see "Avoiding Quota Exceptions by Blocking Message Producers" in the *Administration Console Online Help*.

The thresholds are also not flow control – though they would be better suited to that application than the quotas. The thresholds are simply settings that when exceeded cause a message to be logged to the console to let you know that you are falling behind.

WebLogic JMS also has a flow control feature that enables a JMS server or destination to slow down message producers when it is becoming overloaded. Specifically, when a JMS server/destination exceeds its specified bytes or messages thresholds, it instructs producers to limit their message flow. For more information, see "Controlling the Flow of Messages on JMS Servers and Destinations" in the *Administration Console Online Help*.

Note: The messages maximum setting on a connection factory is not a quota. This specifies the maximum numbers of outstanding messages that can exist after they have been pushed from the server but before an asynchronous consumer has seen them; it defaults to a value of 10.

Q. What is the value of clustering for WebLogic JMS?

A. In version 6.*x*, you could establish cluster-wide, transparent access to destinations from any server in the cluster by configuring multiple connection factories and using *targets* to assign them to WebLogic Servers, as described in "Configuring WebLogic JMS Clustering" in the *Programming WebLogic JMS*. Each connection factory can be deployed on multiple WebLogic Servers, serving as connection concentrators. You could configure multiple JMS servers on the various nodes in the cluster—as long as the servers are uniquely named—and can then assign destinations to the various JMS servers.

For WebLogic JMS 7.0 or later, you can also configure multiple destinations as part of a single distributed destination set within a cluster. Producers and consumers are able to send and receive through a distributed destination. In the event of a single server failure within the cluster, WebLogic JMS then distributes the load across all available physical destinations within the distributed destination. For more information, see "Distributed Destination Tasks" in the *Administration Console Online Help*.

WebLogic JMS also takes advantage of the migration framework implemented in the WebLogic Server core for clustered environments. This allows WebLogic JMS to properly respond to migration requests and bring a JMS server online and offline in an orderly fashion. This includes both scheduled migrations as well as migrations in response to a WebLogic Server failure. For more information, see "Configuring JMS Migratable Targets" in the *Programming WebLogic JMS*.

You can also refer to the "WebLogic JMS Performance Guide" white paper (`WeblogicJMSPerformanceGuide.zip`) on the JMS topic page for more information.

Q. How can I control on which WebLogic Server(s) my application will run?

A. A system administrator can specify on which WebLogic Server(s) applications will run by specifying targets when configuring connection factories. Each connection factory can be deployed on multiple WebLogic servers. For more information on configuring connection factories or using the default connection factories, see "WebLogic JMS Fundamentals" in the *Programming WebLogic JMS*.

Q. How do I perform a manual fail-over?

A. The procedures for recovering from a WebLogic Server failure, and performing a manual failover, including programming considerations, are described in "Recovering From a WebLogic Server Failure" in *Programming WebLogic JMS*.

Q. Does the WebLogic JMS server find out about closed or lost connections, crashes, and other problems and does it recover from them?

A. Yes, but how it does this depends on whether a Java client crashes or WebLogic Server crashes, as follows:

- If a Java client crashes then the JMS server will clean up all the outstanding server-side resource from the crashed client JVM, such as:
  - JMS connection(s) from the crashed client JVM
  - JMS temporary destination(s) created under the above JMS connection(s)
  - JMS session(s) created under the above JMS connection(s)
  - JMS client(s) created under the above JMS session(s) (connection consumer and regular consumer)
  - JMS browser(s) created under the above session(s)
  - JMS producer(s) created under the above session(s)
- If WebLogic Server crashes *and* it is the front-end to the JMS server, then:

- A JMS client will lose *all* the server-side resources listed above.
- The client's `javax.jms.ExceptionListener.onException(...)` will be called (if `javax.jms.JMSConnection.setExceptionListener` is set) with a `LostServerException`, which extends JMSException.

- If WebLogic server crashes *and* it is a back-end to the JMS server, then:
  - A JMS client may partially lose *some* of the server-side resources listed above (only the resource on the crashed server, such as JMS temporary destination(s), JMS client(s) and JMS browser(s).
  - The client's `javax.jms.ExceptionListener.onException(...)` will be called (if `weblogic.jms.extensions.WLSession.setExceptionListener` is set) with a `ConsumerClosedException`, which extends JMSException.

Q. How does an application know if an application server goes down?

A. There are two exception listeners that you can register. Sun Microsystems' JMS specification defines C`onnection.setExceptionListener` that tells you if there is a problem with the connection. That means that all consumers under that connection are also in trouble. The reason you will get the connection exception is because the WebLogic server you connect to on the other side is dead or not responding or someone killed your connection via the Mbean interface.

However, for WebLogic Server JMS, you may have multiple sessions in a connection, with the sessions going to multiple backend servers. WebLogic Server has an extension for this called `WLSession.setExceptionListener` that tells you if there is a problem with a session. For more information, see the JMS WLSession Javadoc.

Q. Do I need to use the WLS T3 protocol?

A. J2EE is all about making the interfaces standard. WebLogic's implementation of the RMI specification uses a proprietary wire-protocol known as T3. Sun's reference implementation of RMI uses a proprietary protocol called JRMP. The fact is that WebLogic developed T3 because they needed a scalable, efficient protocol for building enterprise-class distributed object systems with Java.

While T3 is specific to WebLogic, your application code does not need to know anything about T3 so you should not worry about this. Externalize the "WebLogic-specific strings" (`PROVIDER_URL, INITIAL_CONTEXT_FACTORY`, etc.) to a properties file (or somewhere) and you can make your code completely portable to where you only need change these in the properties file to get your code to run on another J2EE application server.

Note: As of release 8.1, WebLogic JMS also supports the IIOP protocol. In general, this is slower than the T3 protocol.

Q. How do I use HTTP tunneling?

A. If you want to use HTTP tunneling (wrap every message in HTTP to get through a firewall), you need to add `TunnelingEnabled="true"` into your `<Server>` definition in the `config.xml` file or check the appropriate box on the console. Then use a URL like `http://localhost:7001` instead of `t3://localhost:7001` for `Context.PROVIDER_URL` when getting your InitialContext. If you want HTTP tunneling with SSL, use `https://localhost:7002` (where https uses HTTP tunneling with SSL and 7002 is the secure port that you configured). You will pay a performance penalty for doing this, so only use tunneling it if you really need to (i.e., need to go through a firewall).

Q. Does WebLogic JMS support SSL?

A. Yes, SSL is supported in the WebLogic JMS implementation. It is automatically used based on using a URL starting with "t3s:" instead of "t3:" when looking up the initial JNDI context.

Q. How do I integrate non-WebLogic JMS providers with WLS?

A. Refer to "Simple Access to Remote or Foreign JMS Providers" in the *Administration Console Online Help* and the "Using Foreign JMS Providers with WebLogic Server" white paper (`jmsproviders.pdf`) on the JMS topic page, for a discussion on integrating MQ Series, IBus MessageServer, Fiorano, and SonicMQ.

Q. How do two-phase or global transactions relate to WebLogic JMS?

A. A two-phase or global transaction allows multiple resource managers (including EJBs, databases, and JMS servers) to participate in a single transaction.

For example, a client can use a two-phase transaction to send a message from a queue on one JMS server (server A) to a queue on another JMS server (server B). Each server has a unique persistent store. When the transaction is committed, the message is made visible on server B. If the transaction rolls back, the message is put back on the queue on server A.

Note: If both queues happen to be on the same JMS server, then a one-phase transaction is used.

Q. Why is my WebLogic JMS work not part of a user transaction (that is, called within a transaction but not rolled back appropriately)? How do I track down transaction problems?

A. Usually this problem is caused by explicitly using a transacted session which ignores the external, global transaction by design (a JMS specification requirement). A transacted JMS session always has its own inner transaction. It is not affected by any transaction context that the caller may have.

It may also be caused by using a connection factory that is configured with the XAConnectionFactoryEnabled flag set to false.

1. You can check if the current thread is in a transaction by adding these two import lines:

   ```
   import javax.transaction.*
   import weblogic.transaction.*;
   ```

   and adding the following lines (i.e., just after the begin and just before every operation).

   ```
   Transaction tran = TxHelper.getTransaction();
   System.out.println(tran);
   System.out.println(TxHelper.status2String(tran.getStatus()));
   ```

   This should give a clear idea of when new transactions are starting and when infection is occurring.

2. Ensure that the thread sending the JMS message is infected with a transaction. Check that the code is not using a transacted session by setting the first parameter of createQueueSession or createTopicSession to false. Note that creating the connection and/or session is orthogonal to the transaction. You can begin your transaction before or after. You need only start the transaction before you send or receive messages.

3. Check that the XAConnectionFactoryEnabled flag is explicitly set to true for the connection factory in the config.xml file since the default for user-configured connection factories for this value is false. If you are using one of the pre-configured connection factories they are set as follows:

   ```
   weblogic.jms.ConnectionFactory disables user transactions so don't use
   this one for the case where user transactions are desired;

   javax.jms.QueueConnectionFactory and javax.jms.TopicConnectionFactory
   enable user transactions.
   ```

4. You can trace JTA operations by starting the server with this additional property:

   ```
   -Dweblogic.Debug.DebugJMSXA=true
   ```

   You should see trace statements like these in the log:

   ```
   XA ! XA(3163720,487900) <RM-isTransactional() ret=true>
   ```

   This can be used to ensure that JMS is infected with the transaction.

Q. When do WebLogic JMS operations take place as part of a transaction context?

A. When WebLogic JMS is used inside the server, JMS sessions may automatically be enlisted in the JTA transaction depending on the setting of various parameters. Prior to release 8.1, WebLogic JMS sessions would automatically be enlisted in the JTA transaction if either of the following two conditions were met:

- The XAConnectionFactoryEnabled and UserTransactionsEnabled flags were set on the connection factory.
- The XAServerEnabled flag was set on the connection factory.

In WebLogic Server 8.1, it is only necessary to set the XAConnectionFactoryEnabled flag. The old flags are still supported for backward compatibility, however.

When a WebLogic JMS connection factory is registered as a `resource-reference` inside an EJB, servlet, or JSP, and the connection factory is looked up out of the `java:comp/env` JNDI tree, then the EJB container checks to make sure that the appropriate flags are set for transaction enlistment. If the WebLogic JMS connection factory does not support automatic transaction enlistment, then the EJB container will throw an exception if a JMS session is used inside a transaction context. When used without a `resource-reference` however, such as in the case of an EJB that looks up a JMS connection factory directly, without using `java:comp/env`, then no checking takes place. If the JMS session is used outside a JTA transaction, then no enlistment takes place.

The default connection factory, `weblogic.jms.ConnectionFactory`, does not support automatic transaction enlistment. If you desire this behavior, you must use the `weblogic.jms.XAConnectionFactory` factory. (The legacy connection factories `javax.jms.QueueConnectionFactory` and `javax.jms.TopicConnectionFactory` support automatic transaction enlistment as well.)

For more information, see "Using JMS With EJBs and Servlets" in *Programming WebLogic JMS*.

Q. How can an application do a JMS operation and have it succeed, independent of the result of the transaction?

A. In order to do this properly, you must suspend the transaction. How you do this depends on the context in which you are using JMS:

1. Inside an Enterprise Java Bean, there is no way to do this using only standard J2EE APIs. The most *standards compliant* way to do this is by invoking another EJB method (through the EJB container) that has container-managed transactions enabled and a transaction mode of *NotSupported*. This way, the EJB container will suspend the transaction before making the call, and resume it when the call has completed.

2. You may also do this by accessing the WebLogic transaction manager. This method may be used inside an EJB or another server-side component such as a servlet. This requires using a WebLogic-proprietary interface, but you may find it to be more convenient. Here is an example:

```
import javax.transaction.TransactionManager;
TransactionManager tranManager= TxHelper.getTransactionManager();

Transaction saveTx = null;
try {
saveTx = tranManager.suspend();
... do JMS work, it will not participate in transaction
} finally {
// must always resume suspended transactions!
if (saveTx != null) tranManager.resume(saveTx);
}
```

3. Outside an EJB, you have other options. One is to use a transacted session. A transacted JMS session always has its own inner transaction. It is not affected by any transaction context that the caller may have. (However, if you use the deprecated WebLogic Server 5.1 default `javax.jms.QueueConnectionFactory` or `javax.jms.TopicConnectionFactory` factories, or if you define your own factory and set the UserTransactionsEnabled flag to *True*, the JMS session participates in the outer transaction, if one exists and the JMS session is not transacted.)

4. Finally, you may use a WebLogic JMS Connection factory that does not support automatic transaction enlistment. For more information, see the previous question, "When do WebLogic JMS operations take place as part of a transaction context?" on page 14-15.

Q. What happens if acknowledge() is called within a transaction?

A. As per Sun Microsystems' JMS specification, when you are in a transaction, the `acknowledgeMode` is ignored. If `acknowledge()` is called within a transaction, it is ignored.

Q. Why am I getting JDBC XA errors when using JMS in conjunction with JDBC calls?

A. Whenever two resources (such as JMS and a database) participate in a transaction, the transaction becomes two-phase. The database driver you are using is not XA compatible and can't normally participate in a two-phase transaction. The solution is to either use an XA compatible driver, or to configure the `JDBCTxDataSource` value to set `enableTwoPhaseCommit` to true. The caveat for the latter is that this can lead to heuristic errors. If you don't want JMS to participate in the current transaction, see the question "How can an application do a JMS operation and have it succeed, independent of the result of the transaction?" on page 14-16.

Q. Can I use a one-phase commit if my WebLogic JMS JDBC store is on the same database for which I am doing other database work?

A. No. WebLogic JMS is its own resource manager. That is JMS itself implements XAResource and handles the transactions without depending on the database (even when the messages are stored in the database). That means whenever you are using JMS and a database (even if it is the same database as the JMS messages are stored) then it is 2PC.

You may find it will aid performance if you ensure the connection pool used for the database work exists on the same server as the JMS queue—the transaction will still be two-phase, but it will be handled with less network overhead. Another performance boost might be achieved by using JMS file stores rather than JMS JDBC stores.

Q. How do I integrate another vendor's XAResource with WLS to get JTA transactions with another resource manager?

A. In most cases WebLogic JMS will do this for you. For more information, see the "Using Foreign JMS Providers With WebLogic Server" white paper (jmsproviders.pdf) on the JMS topic page.

Q. Why do I get an exception when I start up WebLogic JMS using an XA driver or with a TX data source?

A. You cannot use a TX data source with JMS. JMS must use a JDBC connection pool that uses a non-XA resource driver (you can't use an XA driver or a JTS driver). Do not set the `enableTwoPhaseCommit` option. JMS does the XA support above the JDBC driver.

Q. Is WL JMS XAResource compliant?

A. Yes. WebLogic Server 6.1 or later fully implements the `XAConnection`, `XAConnectionFactory`, `XAQueueConnection`, `XAQueueConnectionFactory`, `XAQueueSession`, `XASession`, `XATopicConnection`, `XATopicConnectionFactory`, and `XATopicSession` methods. These methods are defined as optional in Sun Microsystems' JMS specification and are not part of the `XAResource` interface.

Note: These interfaces are not needed since WebLogic JMS automatically registers itself with the WebLogic transaction monitor.

Q. Why can't I receive a response to a message that I send within a transaction?

A. If you are using container-managed transactions, the original message sent from the EJB will never be sent. Here is what is happening.

1. Container starts transaction.

2. Start method.
3. Generate new message.
4. Send message (message isn't sent – it’s buffered until transaction commit).
5. Do a blocking receive on a queue.
6. End method.
7. Transaction Commit never Reached because original message was never sent because you can't get past blocking receive.

The solution is to either use bean-managed transactions, or to break the send and receive into two separate methods.

Q. What happens to a message that is rolled back or recovered?

A. For more information about what occurs when a message is rolled back or recovered, refer to “Managing Rolled Back, Recovered, Redelivered, or Expired Messages” in *Programming WebLogic JMS*.

Q. Is it possible to set aside a message and acknowledge it later?

A. There are no special primitives for doing this. Here are two possible solutions.

One approach is to use multiple sessions as in the following:

```
while (true) {
  Create a session, subscribe to one message on durable subscription
  Save session reference in memory
  To acknowledge the message, find the session reference and call
  acknowledge() on it.
}
```

Another solution is to use transactions and suspend the work as follows:

```
start transaction
while(true) {
  message = receive();
  if (message is one that I can handle)
  process the message
  commit
} else {
  suspend transaction
  put transaction aside with message
  start transaction
```

```
    }
  }
```

To "acknowledge" the message:

```
resume user transaction
commit
```

To "recover" the message:

```
resume user transaction
rollback
```

Each time you suspend, you need to push the transaction onto a stack or list possibly with the message so you can process it or roll it back later. This solution is high overhead in that there can be a large build up of outstanding transactions. Note that transactions have timeouts and it may rollback on its own, which means you can get the message again (in a different transaction). Note also that there are some practical limits on the number of transactions you should leave outstanding. The default limit is something like 10000. Eventually you want to go back to your stack/list and commit/rollback the transactions. Note that transaction references (`javax.transaction.Transaction`) are not Serializable.

Q. How should I use sorted queues or topics?

A. Destinations are sorted as FIFO (first-in, first-out) by default; therefore, destination keys are used to define an alternate sort order for a specific destination. Destination keys can be message header or property fields. For a list of valid message header and property fields, refer to the "Message" section in *Programming WebLogic JMS*.

Destinations can be sorted in ascending or descending order based on the destination key. A destination is considered to be FIFO if a destination key is defined as *ascending* for the `JMSMessageID` message header field, and LIFO (last-in, first-out) if defined as *descending*. The key defined for the `JMSMessageID` header field, if specified, must be the last key defined in the list of keys. You can define multiple destination keys to sort a destination.

To create a destination key, use the Destination Keys node in the Administration Console. For more information, refer to "Destination Key Tasks" in the *Administration Console Online Help*.

Q. How do I deal with a listener that doesn't keep up with messages being sent?

A. Consider using the asynchronous pipeline for your message listeners to improve performance, as described in the "Asynchronous Message Pipeline" section of *Programming WebLogic JMS*.

Q. How do I get a thread dump to help track down a problem?

A. Ways to get a thread dump:

- Try running this from the command line (after running the `setEnv` script in *WL_HOME*`\server\bin`):

  ```
  java weblogic.Admin -url t3://localhost:7001 THREAD_DUMP
  ```

- On Windows, from the console window, enter `Ctrl+Break`.
- On UNIX, signal the server using `kill -3`.

Q. Do client identifiers need to be unique?

A. Yes, durable subscribers require unique client identifiers. For more information on configuring durable subscribers using the connection factory's Client ID attribute, or by programming your application to set a client ID in its connection (by calling the `setClientID()` connection method), see "Setting Up Durable Subscribers" in *Programming WebLogic JMS*.

Q. How do I manage a queue to view and delete specific messages?

A. Write a program that uses a QueueBrowser. Then delete specific messages by using a QueueReceiver with a selector with the message identifier, as shown in the following example:

```
String selector = "JMSMessageID = '" + message.getMessageID() + "'";
```

Keep in mind that the queue browser is a not a "live" view of the queue. It is a snap-shot.

Q. In what order are messages delivered to a consumer?

A. Order is maintained between any producer and consumer for like delivery mode, sort order, and selector in the absence of a rollback or recover. There are no guarantees of order when multiple producers send to a single consumer or multiple consumers receive from multiple producers.

Order is generally maintained between a producer and a consumer. However, non-persistent messages can get ahead of persistent messages of a higher sort order (i.e., higher priority), can move ahead of each other and a recover or rollback puts messages that were already received back into the queue/topic, which affects order.

Most messaging systems (including WebLogic JMS) maintain order between a producer and a destination and then order between the destination and the consumer. So, once things arrive at the destination, the order does not change.

Finally, the asynchronous pipeline that is supported in WebLogic JMS affects the ordering. By default there can be as many as ten outstanding messages pushed out from the server to an asynchronous client that have not been seen by the client yet. If the asynchronous consumer is "caught" up, these messages will not be sorted. Destination sorting does not occur in the pipeline. If a destination is sorted by priority, and a new message comes in of higher priority than those messages already in the pipeline, it will not leap ahead in the pipeline, it will become first in the destination. The size of the pipeline is configurable; see the MessagesMaximum setting on the connection factory used. If you want real priority sorting, change the maximum number of messages on the factory to one. For more information, see the "Asynchronous Message Pipeline" section of *Programming WebLogic JMS*.

Q. How do I ensure message ordering even in the event of rollbacks and recoveries?

A. In WebLogic JMS 8.1 message ordering can be maintained to single consumers on a queue or topic subscription – even in the event of rollbacks and recoveries, as described in "Ordered Redelivery of Messages" in *Programming WebLogic JMS*.

Q. Is it possible to have multiple queue receivers listening on the same queue?

A. Yes, although the JMS specification does not define the behavior here.

Q. Is there a way to make a queue such that if one application has one object as listener on that queue, no other application can listen to the messages on that queue?

A. No. An alternative is to create a topic with a single durable subscription because a durable subscription may only have one consumer associated with it. The only drawback is that selectors would no longer work the same as they do with queues. Changing the selector on a durable subscription "resets" the subscription as per Sun Microsystems' JMS specification, causing all messages currently in the subscription to be deleted.

Note: If you configure a connection factory that has its Client ID set, this limits the connection factory to one client and may serve the purpose.

Q. Why doesn't setting values work using javax.jms.Message.setJMSPriority, DeliveryMode, Destination, TimeStamp or Expiration?

A. These methods are for vendor use only. The message values are overwritten on each send/publish. You should use the equivalent methods on the `MessageProducer`, `QueueSender`, or `TopicPublisher` to set these values (i.e., `setJMSPriority`, `setDeliveryMode`, `setTimeToLive`). Check to see that these values are not being overridden by the optional template configuration override values.

Q. What care must be taken when multi-threading WebLogic JMS clients?

A. The rules for multi-threading are described in section 2.8 of the JMS specification, with additional language in sections 4.4.6 on session usage, 4.4.9 on using multiple sessions, and 4.4.17 on concurrent message delivery. In a nutshell, it states that JMS sessions are single-threaded. Consequently, if multiple threads simultaneously access a session or one of its consumers or producers the resulting behavior is undefined. In addition, if multiple asynchronous consumers exist on a session, messages will be delivered to them in series and not in parallel.

To take advantage of multiple threads with JMS, use multiple sessions. For example, to allow parallel synchronous receive requests, design the application so that only one consumer may be active per session and use multiple sessions.

Q. How should an application be set up to subscribe to multiple topics?

A. If you want to listen to N topics, using N subscribers and N sessions gives you concurrency up to N simultaneous threads of execution provided you have that many threads to work with. N subscribers and 1 session serializes all subscribers through that one session. If the load is heavy they may not be able to keep up without the extra threads. Also, if you are using `CLIENT_ACKNOWLEDGE`, N sessions gives you N separate message streams that can be individually recovered. Having 1 session crosses the streams giving you less control.

As of version 6.*x* or later, WebLogic JMS on the server side efficiently uses a small, fixed number of threads independent of how many client sessions there are.

Q. How should I use blocking and asynchronous receive() calls?

A. The synchronous `receive()` method blocks until a message is produced, the timeout value, if specified, elapses or the application is closed. We strongly recommend that you avoid using blocking `receive()` calls on the server side because a synchronous `receive()` call consumes resources for the entire duration that the call is blocked.

When methods are received asynchronously, the application is notified using a message listener only when a message has been produced, so no resources are consumed waiting for a message.

Q. What precautions should I take when I use blocking receive() calls?

A. If your application design requires messages to be received synchronously, we recommend using one of the following methods listed in order of preference:

- Pass a timeout value as an argument to the `receive()` method and set it to the minimum value greater than zero, that is allowed by the application to avoid consuming threads that are waiting for a response from the server.

- Use the `receiveNoWait()` method which returns the next message or a null value if no message is currently available. In this case, the call does not block. The servlet should provide a way to return to or reschedule the request, without calling `wait()`.

Note: Use of this option should be minimized, as it may deadlock a busy server.

- Ensure that more threads are configured than the number of possible simultaneous blocking `receive()` calls.

Q. What is the NO_ACKNOWLEDGE acknowledge mode used for?

A. The `NO_ACKNOWLEDGE` acknowledge mode indicates that received messages do not need to be specifically acknowledged which improves performance, but risks that messages are lost. This mode is supported for applications that do not require the quality of service provided by session acknowledge and that do not want to incur the associated overhead.

Messages sent to a `NO_ACKNOWLEDGE` session are immediately deleted from the server. Messages received in this mode are not recovered and, as a result, messages may be lost and/or duplicate message may be delivered if an initial attempt to deliver a message fails.

Note: You should avoid using this mode if your application cannot handle lost or duplicate messages. Duplicate messages may be sent if an initial attempt to deliver a message fails.

In addition, we do not recommend that this acknowledge mode be used with persistent messaging, as it implies a quality of service that may be too low for persistent messaging to be useful.

Q. When should I use multicast subscribers?

A. Multicasting enables the delivery of messages to a select group of hosts that subsequently forwards the messages to multicast subscribers. The benefits of multicasting include:

- Near real-time delivery of messages to host group.
- High scalability due to the reduction in the amount of resources required by the JMS server to deliver messages to multicast subscribers.

Note: Multicasting is only supported for the Pub/sub messaging model.

For an example of when multicasting might be useful, consider a stock ticker. When accessing stock quotes, timely delivery is more important than reliability. When accessing the stock information in real-time, if all, or a portion, of the contents is not delivered, the client can simply request the information be resent. Clients would not want to have the information recovered in this case because by the time it is redelivered it would be out-of-date.

Multicast messages are not guaranteed to be delivered to all members of the host group. For messages requiring reliable delivery and recovery, you should not use multicasting.

Q. When should I use server session pools and connection consumers?

A. WebLogic JMS implements an optional JMS facility for defining a server-managed pool of server sessions. However, session pools are now used rarely, as they are not a required part of the J2EE specification, do not support JTA user transactions, and are largely superseded by message-driven beans (MDBs), which are simpler, easier to manage, and more capable.

For a detailed discussion on this topic, see the "MDBs vs. ServerSessionPools" section in the "WebLogic JMS Performance Guide" white paper (`WeblogicJMSPerformanceGuide.zip`) on the JMS topic page.

Q. How do I issue the close() method within an onMessage() method call and what are the semantics of the close() method?

A. If you wish to issue the `close()` method within an `onMessage()` method call, the system administrator must select the Allow Close In OnMessage check box when configuring the connection factory. For more information, see "JMS Connection Factory Tasks" in the *Administration Console Online Help*. If this check box is not selected and you issue the `close()` method within an `onMessage()` method call, the call will hang.

The session or connection `close()` method performs the following steps to execute an orderly shutdown:

- Terminates the receipt of all pending messages. Applications may return a message or null if a message was not available at the time of the close.
- Waits until all message listeners that are currently processing messages have completed (except for the message listener from which the `close()` method is being called).
- Rolls back in-process transactions on its transacted sessions (unless such transactions are part of an external JTA user transaction).
- Does not force an acknowledge of client-acknowledged sessions. By not forcing an acknowledge, no messages are lost for queues and durable subscriptions that require reliable processing.

When you close a connection, all associated objects are also closed. You can continue to use the message objects created or received via the connection, except the received message's `acknowledge()` method. Closing a closed connection has no effect.

Note: Attempting to acknowledge a received message from a closed connection's session throws an `IllegalStateException`.

When you close a session, all associated producers and consumers are also closed.

For more information about the impact of the `close()` method for each object, see the appropriate `javax.jms` javadoc.

Q. How do I publish an XML message?

A. Follow these steps:

1. Generate XML from the DOM document tree.
2. Serialize the generated DOM document to a `StringWriter`.
3. Call `toString` on the `StringWriter` and pass it into `message.setText`.
4. Publish the message.

Q. How do I use WebLogic JMS in an applet?

A. For detailed instructions and examples on how to accomplish this, see "Using BEA WebLogic JMS with Applets" on BEA's JMS topic page.

Q. How do I use a startup class to initialize and later reference WebLogic JMS objects?

A. This topic is covered in news://newsgroups.bea.com/3ad0d7f3@newsgroups.bea.com. The sample code does not cleanup properly at shutdown. You can use a shutdown class that does something like the following:

```
JMSobject WLSobject = null;
try {
 WLSobject = JMSStartUp.getJMSobject();
 WLSobject.JMSCleanup();
} catch(Exception e) {}
```

Load-on-start servlets can provide a nice solution to provide both initialization and cleanup. For more information, refer to "What is the standard way to create threads, do initialization, etc. within the application server?" on page 14-29.

Q. Is it possible to send or receive a message from within a message listener?

A. Yes. You can send to or receive from any queue or topic from within in a message listener.

Outside of a MDB, you can do this by using the same Connection or Session that the `onMessage()` is part of. When you create your message listener, you pass a session into your constructor. Then you have access to the session in your `onMessage()` method and are able to make synchronous – not asynchronous – calls from within the `onMessage()` method. Do not use another Session that is servicing another `onMessage()`, because that would multi-thread the Session, and Sessions do not support multi-threading.

However, when using this technique outside a MDB, there is no way to guarantee that the receipt of the message by the MessageListener, and the send of the new message, happen as part of the same transaction. So, there can be duplicates or even lost messages. For example:

- If you call acknowledge after the `publish()` and the acknowledge fails for whatever reason (network or server failure), then you will see the message again and will end up publishing twice (possible duplicate semantics). You can try to keep track of sequence numbers to detect duplicates, but this is not easy.

- If you call acknowledge before the `publish()`, you get *at-most-once* transaction semantics. If the `publish()` fails, you don't know if the failure occurred before or after the message reached the server.

If you require *exactly-once* transactional semantics using `onMessage()`, then you must use transactional MDBs. In this case, the `onMessage()` method for a transactional MDB starts the transaction and includes the WebLogic JMS message received within that transaction. Then, you must ensure that the `send` or `publish` of the new message is part of the same transaction as the receipt of the message.

In WebLogic Server 8.1, you can guarantee that this happens by using a connection factory that you get from a `resource-reference` defined for the MDB. For detail instructions on how to accomplish this, see "Using JMS With EJBs and Servlets" in *Programming WebLogic JMS*. By using a `resource-reference`, you also get automatic pooling of the JMS Connection, Session, and MessageProducer objects, and the transaction enlistment will happen automatically regardless of whether you use WebLogic JMS or another JMS provider, as long as the JMS provider supports XA.

In earlier versions of the product, WebLogic JMS would automatically enlist itself with the current transaction if the UserTransactionsEnabled or XAServerEnabled flag was set on the connection factory. However, prior to release 8.1, the server will not pool any JMS objects or automatically enlist a foreign JMS provider in the transaction. In these earlier versions, you may want to cache JMS objects yourself. For more information, see "How do I create a producer pool?" on page 14-28.

Q. How do I create a producer pool?

A. For instructions on how to accomplish this, see "Using JMS With EJBs and Servlets" in *Programming WebLogic JMS*. For a detailed code sample, see the "Appendix A: Producer Pool Example" section in the "WebLogic JMS Performance Guide" white paper (`WeblogicJMSPerformanceGuide.zip`) on the JMS topic page.

Q. What are pending messages in the console?

A. Pending means the message could have been:

- sent in a transaction but not committed.
- received and not acknowledged.
- received and not committed.
- subject to a redelivery delay (as of WebLogic JMS 6.1 or later).
- subject to a delivery time (as of WebLogic JMS 6.1 or later).

A rolled back message remains pending until the transaction actually rolls back. Rolling it back multiple times does not cause double counting, nor does an exception that set a transaction as rollbackOnly followed by an actual rollback.

Current implies messages that are not pending.

Total implies total since server last started. The byte counts only consider the payload of messages which includes the properties and the body but not the header.

Q. How do I use a less than or greater than on a message selector in ejb-jar.xml?

A. Enclose the selector in a CDATA section. That will prevent the XML parser from thinking that less than or greater than is a tag.

```
<jms-message-selector>
<![CDATA[ JMSXAppID <> 'user' ]]>
</jms-message-selector>
```

Q. Can I use another vendor's destination with a WebLogic JMS API?

A. WebLogic Server JMS does not know what to do with foreign destinations that it runs into. This issue has been discussed with Sun and the specification does not clearly define destinations well enough for vendors to interoperate at that level. They agree that it is sufficient not to handle foreign destinations preferably in such a way that sending/receiving still work. For WebLogic JMS, if you do a `setJMSdestination` (you should not because it is only for the provider to set it) with a foreign destination, it gets ignored (set to null). Similarly, if you do a `setJMSReplyTo` for a foreign destination, WebLogic JMS will ignore it (set it to null).

Q. What is the standard way to create threads, do initialization, etc. within the application server?

A. Threads should generally not be created by the user directly is because things may not work correctly. User-created threads do not have some of the thread-local variables pre-set by WebLogic when it creates it's own execute threads, the associated transaction context, or the environment such as the proper class loader. The WebLogic-specific way of doing this is with a startup class or using the WebLogic Time Services. The portable way to do this is to define a load-on-startup servlet, doing the initialization in the `init()` method and the cleanup in the `destroy()` method. The servlet itself does nothing. This approach also allows for undeploy/redeploy of the application without restarting the server, including proper cleanup/initialization each time. It also providers more dynamic management of the dependent classes without restarting the server.

Q. Why do I get a JNDI problem when I name a Topic A.B and a second Topic A.B.C?

A. This is a JNDI implementation issue. JNDI uses the dots to build a directory-like structure. A given element cannot be both a node and a leaf in the tree. In this example, B is used as a leaf off of A, but then is used as a node off of which C is a leaf.

Q. What should an XPATH selector look like?

A. For instructions and samples about using XPATH syntax with WebLogic JMS, see "Defining XML Message Selectors Using the XML Selector Method" in *Programming WebLogic JMS*.

Q. How do I handle request/response using WebLogic JMS?

A. There are several approaches to handling request/response processing with JMS.

- Use a temporary queue for each requestor and have the response go back to that queue.
- Use the QueueRequestor class, which does the temporary queue for you, and wait for the reply, as in the following:

```
// create temporary queue for receiving answer
qrequestor = new QueueRequestor(qsession, queue);
TextMessage msg = qsession.createTextMessage();
TextMessage reply = (TextMessage) qrequestor.request(msg);
```

- Use a dedicated response topic or queue with message selectors.

- For more information on request/response, see the "Using a Request/Response Design" section in the "WebLogic JMS Performance Guide" white paper (`WeblogicJMSPerformanceGuide.zip`) on the JMS topic page.

Q. Is it okay to add new sessions and subscribers to a Queue or Topic Connection once it has been started?

A. Yes, with one caveat. You may not add new subscribers/consumers to a session if it already has active async consumers. Sessions must only be accessed single-threaded as per the JMS Specification. If you feel you need to do this, create a new Session and add it to that one instead.

You can add receivers to a session that is part of a started connection. However, a receiver in itself is not asynchronous. You need a listener to make it asynchronous. The first creation of a receiver is always safe. If you then add a listener for that first receiver, you have to worry for any future receivers in that same session. You can create new sessions and the first receiver for that session with no worries.

Once you want to create a second receiver in a session, if the first receiver has a `MessageListener`, you have to take care to make sure there are no other threads of execution in that session. You can do this by stopping the connection or actually creating your receiver from the `onMessage` routine of the first receiver.

Q. What can I do when I get java.lang.OutOfMemoryError because producers are faster than consumers?

A. Quotas can be used to help this situation. Your sender will then receive `ResourceAllocationExceptions` and the server will stay up. In release 8.1 or later, senders can be configured to block waiting for space rather than receive `ResourceAllocationExceptions`. For more information, see "Avoiding Quota Exceptions by Blocking Message Producers" in the *Administration Console Online Help*.

You can also use the Message Paging feature, which saves memory by swapping messages out from virtual memory to a dedicate paging store when message loads reach a specified threshold. JMS message paging saves memory for both persistent and non-persistent messages, as even persistent messages cache their data in memory. For more information, see "Paging Out Messages To Free Up Memory" in the *Administration Console Online Help*.

Q. How should connections and sessions be allocated?

A. Think of a connection as a single physical connection (a TCP/IP link). A session is a means for producing and consuming an ordered set of messages. Creating a connection is generally expensive. Creating a session is less expensive. Generally people use one connection and share across all the threads with each thread having its own session. If you have thread groups and need to start/stop/close the resources for a given group, one connection per group is good. A group can have exactly one thread.

Q. Is there a way to dynamically change an existing selector for a TopicConsumer using the setMessageSelecter(String)?

A. No. Once you instantiate the consumer the selector is fixed at the time that the consumer is created. Changing the selector is like removing the current consumer, removing all associated messages and then creating a new one.

Q. How can I avoid asynchronous message deadlocks?

A. Due to a limitation in the JMS specification, asynchronous messages can become deadlocked if the `close()` method of a session is inside a user-synchronized block. To resolve this, you must move the `close()` method outside the user-synchronized block. For example:

```
public class CloseTest() {
private void xxx() {
 synchronized (this) {
  create connection/session/consumer
  initialize and set a listener for this consumer;
  wait();
  connection.close();
 }
}

 private void onMessage(Message message) {
 synchronized (this) {
  notify();
 }
 }
}
```

Before the `connection.close()` method is closed, another message can be delivered to the `onMessage` routine by the `JMSProvider`. The `main()` method thread owns the monitor lock for the `CloseTest` method. Before the `onMessage()` method of the `CloseTest` class fires, JMS sets `INLISTENER` as the state for the session in `JMSSession` (the JMS specification says that the `close()` method must wait for the `onMessage` routine), so that the `main()` method thread can wait for the `onMessage` routine to complete.

Now when the `onMessage` routine tries to acquire the monitor lock, it blocks waiting for the `main()` method thread to give up, and the `main()` method thread is waiting for the `onMessage` to be completed.

JMS also blocks when the `close()` method of a consumer is done from an `onMessage` routine and the `allowCloseInOnMessage` attribute is set to false in the `config.xml` file.

Q. What are the advantages of message-driven beans?

A. The message-driven bean is a stateless component that is invoked by the EJB container as a result of receiving messages from a JMS queue or topic. It then performs business logic based on the message contents, effectually freeing you from any JMS configuration and reconnection chores.

The message-driven bean model allows EJB developers to work with a familiar framework and set of tools, and also provides access to the additional support provided by the container. The goal of the message-driven bean model is to assure that developing an EJB that is asynchronously invoked to handle the processing of incoming JMS messages is as easy as developing the same functionality in any other JMS `MessageListener`.

One of the main advantages of using message-driven beans in place of the standard JMS `MessageListener` is that a JTA transaction can be started for you automatically and the received message will be part of that transaction. In this case, other operations can be infected with the same JTA transaction such as database operations. This is the only way to infect a message from an asynchronous consumer and another JTA operation with the same transaction.

For more information on message-driven beans, see "Designing Message-Driven Beans" in *Programming WebLogic Enterprise JavaBeans*.

Q. How does concurrency work for message-driven beans?

A. For a queue, multiple JMS Sessions are created on each server instance where the MDB is deployed. The number of sessions created is never greater than the `max-beans-in-free-pool` setting in the MDB's deployment descriptor. JMS then delivers messages in parallel to the MDB instances as it would for any other kind of message listener. If a MDB is deployed to multiple servers in a cluster, Sessions are created for each MDB instance on each server.

For a topic, however, one topic consumer is used to pass out messages to multiple threads to get concurrency, while producing only a single copy of each message. If multiple MDBs are deployed to listen on the same topic, then each MDB will receive a copy of every message. Therefore, when a MDB is deployed to multiple servers and it listens to a topic, each server will receive its own copy of each message. So, if you want a message to be processed by exactly one MDB, you should use a queue.

Q. Can an MDB be a message producer or both a producer and consumer?

A. Yes. You have no JMS context inside the MDB so you will need to establish a connection, session and producer yourself. One option is to do this every time you come into the `onMessage` routine for the MDB. This is sufficient if the message rate is relatively low. The second option is to establish the necessary objects in `ejbActivate()`. Note that the objects are not serializable so they can't be passivated for a stateful session bean or an entity bean. When the EJB deactivates, you need to close the associated objects. The third option is that you could build up a JMS connection/sender session pool within a startup class complete with your own synchronization and blocking to get a connection. There is an example of this in the question "Is it possible to send or receive a message from within a message listener?" on page 14-26.

For more information, see "Using JMS With EJBs and Servlets" in *Programming WebLogic JMS*.

Q. If an MDB uses a durable subscription, will messages be accumulated if the MDB is not deployed?

A. The durable subscription is created when the MDB is deployed for the first time. The durable subscription is not deleted when the MDB is undeployed or *deleted*. This means that once the MDB has been deployed once, messages will continue to accumulate on the subscription, even if the MDB is undeployed or deleted. So, when an MDB is retired from service, you should delete the durable subscription to prevent a build-up of messages. You can use the administration console to do this, or you can write a standalone program using the Java API that calls `unsubscribe` on the durable subscription.

Q. How do I use non-WebLogic JMS provider destinations to drive MDBs?

A. See the "Using Foreign JMS Providers with WebLogic Server" white paper (`jmsproviders.pdf`) on the JMS topic page.

Q. Can you use a foreign JMS provider to drive an MDB transactionally?

A. Yes. In WebLogic Server 7.0 or later, you can deploy an MDB that supports container-managed transactions against a foreign JMS provider. If the MDB is configured with a "transaction-type" attribute of "Container" and a "trans-attribute" of "Required", then WLS will use XA to automatically enlist the foreign JMS provider in a transaction. (See the next question for an example of an MDB that uses container-managed transactions.)

If the foreign JMS provider does not support XA, then you cannot deploy an MDB that supports container-managed transactions with that provider. Furthermore, if the JMS provider does support XA, you must ensure that the JMS connection factory that you specify in the `weblogic-ejb-jar.xml` file supports XA—each JMS provider has a different way to specify this.

See the "Using Foreign JMS Providers with WebLogic Server" white paper (`jmsproviders.pdf`) on the JMS topic page for an example of how to configure an MDB to use a foreign provider.

Q. How do I roll back a transaction within an MDB?

A. To roll back a transaction, you can either use the Weblogic extension `TXHelper`, or you can use the MDB context as in the following code examples:

```
UserTransaction ut =
   weblogic.transaction.TXHelper.getUserTransaction();
   ut.setRollbackOnly();
```

or

```
private MessageDrivenContext context;

public void setMessageDrivenContext(
  MessageDrivenContext mycontext) {
    context = mycontext;
}
public void onMessage(Message msg) {
try {    // some logic
}
   catch(Exception e) {
   System.out.println("MDB doing rollback");
   context.setRollbackOnly();
}
```

Q. How do server session pools and message driven beans compare?

A. For a detailed discussion on this topic, see the "MDBs vs. ServerSessionPools" section in the "WebLogic JMS Performance Guide" white paper (`WeblogicJMSPerformanceGuide.zip`) on the JMS topic page.

FAQs: WebLogic Message Bridge

- Is there another way to monitor the messaging bridge without using the Administration Console?
- Can the messaging bridge use distributed destinations as source and target destinations?
- Why does the messaging bridge sometimes hang while processing messages?

Q. Why did the messaging bridge fail to connect to the source bridge destination?

A. Either an error occurred when configuring the source bridge destination parameters, or the actual source destination is not running and cannot communicate with the messaging bridge.

- Verify whether the bridge's source destination is correctly configured, by making sure that the following fields on the JMS Bridge Destination →Configuration console page have been properly completed:
  - Connection URL—this must be the URL of the JNDI provider used to look up the connection factory and actual destination.
  - Destination JNDI Name—this must be the JNDI name of the actual destination mapped to the source bridge destination.
  - Connection Factory JNDI Name—this must be the connection factory used to create a connection for the actual destination mapped to the source bridge destination.
  - User Name/Password—make sure that this user ID has permission to access the actual source destination.
- Verify that the actual source queue or topic destination mapped to the source bridge destination is running and healthy, as follows:
  - Is the WebLogic Server instance hosting the source destination running?
  - Is the JMS server hosting the source destination correctly deployed?

Note: This troubleshooting scenario for correcting a source bridge destination connection failure also applies to target bridge destinations.

Q. Can the messaging bridge handle two-phase or global transactions between separate WebLogic Server domains or between different releases?

A. Yes, as long as the communication is between source and target WebLogic domains that are both running release 6.1 SP03 or later, *and* the bridge is configured to use the Exactly-once quality of service.

For more information about using the Exactly-once QOS when interoperating between release 6.1 and release 7.0 domains and higher, see "Using the Messaging Bridge To Access Destinations In a Release 6.1 or Later Domain" in the *Console Help*.

Q. I configured the messaging bridge to use the Exactly-once quality of service for two-phase transactions. So why am I getting a "quality of service is unreachable" error?

A. There are some additional configuration requirements for the messaging bridge to handle transactions between WebLogic domains:

- The supported adapters are located in the `WL_HOME\server\lib` directory. For the Exactly-once QOS, the transaction adapter, `jms-xa-adp.rar`, must be deployed in the release 8.1 domain where the bridge is running, via the select Deployments →Connector node on the console.
- This `jms-xa-adp.rar` adapter must also be identified in the Adapter JNDI Name attribute as `eis.jms.WLSConnectionFactoryJNDIXA` on the JMS Bridge Destination → Configuration tab for both the source and target bridge destinations.
- For WebLogic JMS, verify that you are using the transactional XAConnectionFactory for the queue or topic destinations mapped to both the source and target bridge destinations. To verify this, the following attributes must be set on the JMS →Connection Factory → Configuration →Transactions console tab or in the configuration file (`config.xml`):
- `UserTransactionsEnabled=true`
- `XAConnectionFactory=true`
- For third-party JMS vendors, verify that you are using a transactional connection factory for the destinations mapped to the source and target bridge destinations.

For more information about using the Exactly-once QOS when interoperating between release 6.1 and release 7.0 or later domains, see "Using the Messaging Bridge To Access Destinations In a Release 6.1 or Later Domain" in the *Console Help*.

Q. Can I configure the messaging bridge to automatically downgrade the quality of service if the Exactly-once service isn't available on either the source or target bridge destination?

A. Yes, just make sure to select the QOS Degradation Allowed check box on the Messaging Bridge →Configuration →General administration console page.

Q. Why do I get a security authorization exception when attempting to forward messages from a WebLogic Server 7.0 GA, SP01, or SP02 destination to a release 6.1 destination?

```
java.lang.SecurityException: Invalid Subject: principals=[user1]
```

A. In WebLogic Server 6.1, the trust relationship between two WebLogic Server domains was established if the system password was the same in both domains. For a release 7.0 GA, SP01, and SP02 messaging bridge to communicate with a release 6.1 domain, you must establish a trusted relationship across release the 6.1 and 7.0 domains. A trusted relationship is established when the Credential attribute for one domain matches the Credential attribute for another domain. Therefore, if you want a release 6.1 domain to interoperate with a release 7.0 SP02 or earlier domain, you need to change the Credential attribute in both domains to the password of the "system" user in the release 6.1 domain.

However, for release 7.0 SP03 or later, you do not need to establish a trusted relationship across release 6.1 and 7.0 domains.

- For more information about establishing a trust relationship between release 7.0 and 6.1 domains for the messaging bridge, see "Enabling Security Interoperability for WebLogic Domains" in *Console Help*.
- For more information about release 6.1 domain interoperability security, see "Using Compatibility Security" in *Managing WebLogic Security*.
- For more information about WebLogic Server 7.0 domain interoperability security, see "Enabling Trust Between WebLogic Domains" in *Managing WebLogic Security*.

Q. I deployed the transactional `jms-xa-adp.rar` resource adapter on the WebLogic 8.1 domain where the message bridge is running, but I still get a "*failed to find bridge adapter*" message?

A. You need to associate *both* the source and target bridge destinations with the appropriate `.rar` adapters in order for the bridge to communicate with them. For the `jms-xa-adp.rar` transaction adapter, it must be identified in the Adapter JNDI Name attribute as `eis.jms.WLSConnectionFactoryJNDIXA` on the JMS Bridge Destination →Configuration tab for both the source and target bridge destinations.

Note: The "failed to find bridge adapter" message does not necessarily indicate a problem if it only occurs once. However, if it occurs repeatedly, you should check the adapter deployment and the adapter JNDI name used in the source and target bridge destinations.

For more information about the bridge resource adapters, see "About the Bridge's Resource Adapters" in the *Console Help*.

Q. When configuring a source or target messaging bridge destination, do I need to set the Adapter Classpath field?

A. Leave the Adapter Classpath field blank when connecting to source and target destinations that are both running on release 8.1. When connecting to either a source or target destination that is running on release 6.0 or earlier, the Adapter Classpath field must indicate the location of the classes for the earlier WebLogic Server release. When connecting to a third-party JMS provider, the bridge destination must supply the provider's CLASSPATH in the WebLogic Server CLASSPATH.

Q. Can the messaging bridge forward durable subscription messages between separate WebLogic Server 6.1 and release 7.0 or later domains?

A. Yes, as long as the domain hosting the bridge is using WebLogic 7.0 Service Pack 1 or later. To enable durable messages across the messaging bridge using the Administration Console, select the Durability Enabled attribute on the Messaging Bridge →Configuration →General tab.

Q. How do I enable debugging for the messaging bridge?

A. You can enable debugging for the messaging bridge using either of the followings methods:

- Add the following lines to your WebLogic start script (before the `weblogic.Server` line):

  ```
  -Dweblogic.Debug.DebugMessagingBridgeStartup=true
  -Dweblogic.Debug.DebugMessagingBridgeRuntime=true
  ```

- Add the following statements to the ServerDebug entry in your configuration file (`config.xml`) for the server that the messaging bridge is running on:

  ```
  DebugMessagingBridgeStartup="true"
  DebugMessagingBridgeRuntime="true"
  ```

Once debugging is enabled for the messaging bridge, the debugging messages are sent to the server log by default. However, if you want them to appear in the Administration Console, add "`DumpToConsole`" to the statements show above. For example:

```
-Dweblogic.Debug.DebugMessagingBridgeStartupDumpToConsole=true
```

Q. What do the messaging bridge monitoring states indicate on the Monitor Messaging Bridge console page?

A. When monitoring a messaging bridge's state, use the following table to determine a course of action, if necessary. For more information, see Managing a Messaging Bridge in the *Console Help*.

Table 15-1

Description	Action
WARN: Failed to find the source adapter	Check if the adapter is deployed or the JNDI name in the source JMSBridgeDestination instance is correct.
WARN: Failed to find the target adapter	Check if the adapter is deployed or the JNDI name in the target JMSBridgeDestination instance is correct.
Found both of the adapters and making connections	No.
WARN: Stopped by the administrator	No.
WARN: Failed to look up the source adapter	Check if the adapter is deployed or the JNDI name in the source JMSBridgeDestination instance is correct.
WARN: Failed to look up the target adapter	Check if the adapter is deployed or the JNDI name in the target JMSBridgeDestination instance is correct.
Found two adapters and about to make connections	No.
WARN: Failed to connect to the source	Check all the parameters configured for the source bridge destination. Check if the source server is running and whether the actual destination is active.
Connected to the source	No.
WARN: Failed to connect to the target	Check all the parameters configured for the target bridge destination. Check if the target server is running and whether the actual destination is active.
Connected to the target	No.
Forwarding messages	No.
WARN: Failed to connect and will reconnect later	Check if the source and target bridge destinations are running and healthy.

Q. Is there another way to monitor the messaging bridge without using the Administration Console?

A. Yes, there is a run-time MBean (`MessagingBridgeRuntimeMBean`) for each bridge instance. WebLogic Server run-time MBeans provide a snapshot of information about domain resources. When a particular resource in the domain (such as a messaging bridge) is instantiated, an MBean instance is created which collects information about that resource.

The `MessagingBridgeRuntimeMBean` has a `getState()` method that currently returns a String ("Active" or "Inactive") and a `getDescription()` method, which returns a String with more detailed information. The name of a bridge runtime MBean consists of the WebLogic Server instance name and the bridge name. If a bridge named *mybridge*, runs on WebLogic Server instance named *myserver*, the bridge runtime MBean will be named *myserver.bridge.mybridge*.

For more information about using run-time Mbean management commands, see "WebLogic Server Command-Line Interface Reference" in the *Administration Guide*. For more information about programming MBean monitoring notifications, see "Using WebLogic Server MBean Notifications and Monitors" in *Programming WebLogic JMX Services*.

Q. Can the messaging bridge use distributed destinations as source and target destinations?

A. Yes, the messaging bridge can send to and receive from distributed destinations. Bea recommends the following configurations:

- If the source is distributed destination, the bridge is pinned to one of the members when it connects to the destination. It stays connected only to that member until it reconnects. This means that the bridge will not receive messages from the other members of the distributed destination. Therefore, the best practice is to configure one bridge for each member of a distributed destinations using the member's JNDIName.
- If the target is a distributed destination, the best practice is to send to the distributed destination using the distributed destination's JNDIName and disable server affinity. This allows the distributed destination to load balance incoming messages.

Q. Why does the messaging bridge sometimes hang while processing messages?

A. By default, the `AllowCloseInOnMessage` attribute in the `JMSConnectionFactory` class is set to `false`. To ensure that the server does not hang, set this value to `true`. For more information, see:
http://edocs.bea.com/wls/docs81/config_xml/JMSConnectionFactory.html#AllowCloseInOnMessage

CHAPTER 16

FAQs: JTA

- How can I incorporate MQSeries as an XA resource for distributed transactions in WebLogic Server?
- Can I use a non-XA driver in distributed transactions?
- Can I use more than one non-XA connection pool in distributed transactions?
- How do XA and non-XA drivers differ in distributed transactions?
- What XA drivers can I use in addition to the WebLogic jDriver for Oracle/XA?
- Can I use the Oracle thin driver as an XA driver in distributed transactions?
- Why do I get SQLException “Result set already closed” message?
- Do I need a 2PC licence when I use JMS with one JDBC non-XA driver?
- Why am I getting an exception when I use JMS with a non-XA driver?
- Can I obtain a JDBC connection before I start a distributed transaction?
- Can I close a JDBC connection after the distributed transaction is committed or rolled back?
- I get the following XAER_RMFAIL XAException when accessing an XAResource: "Internal error: XAResource '<name>' is unavailable". What does that mean? How should I handle it?

Q. How can I incorporate MQSeries as an XA resource for distributed transactions in WebLogic Server?

A. You can download a zip file with instructions, support classes, utilities, and an example from the code samples for weblogic server page on BEA's *dev2dev* site. You can also directly download the package from `ftp://edownload:BUY_ME@ftpna2.bea.com/pub/downloads/wlsmqseries.zip`.

Q. Can I use a non-XA driver in distributed transactions?

A. When the non-XA connection pool is the only resource participating in a transaction distributed across multiple servers, you just need to configure a TxDataSource for the non-XA driver.

However, when more than one resource participates in the distributed transaction, you must also set the TxDataSource property `EnableTwoPhaseCommit=true`. For more information, see Configuring JDBC DataSources in the *Administration Console Online Help*. In both cases, always obtain a connection via the DataSource interface, not through the deprecated DriverManager interface. If you obtain a connection via DriverManager, the interface cannot pick up the `EnableTwoPhaseCommit` setting of the TxDataSource; this may result in unexpected behavior in distributed transactions. Also, when you use the DataSource interface, you do not need to distinguish either the URL or the specific WebLogic multitier driver (JTS, RMI, or pool.) The URL and specific driver are obtained through the config.xml file and JNDI lookup.

Q. Can I use more than one non-XA connection pool in distributed transactions?

A. No. Even if you set `EnableTwoPhaseCommit=true` for both TxDataSources of the connection pools, attempting to use two non-XA connection pools in the same distributed transaction will result in:

"java.sql.SQLException: Connection has already been created in this tx context for pool named <first pool's name>. Illegal attempt to create connection from another pool: <second pool's name>"

when you attempt to get the connection from the second non-XA connection pool.

Q. How do XA and non-XA drivers differ in distributed transactions?

A. The differences between XA and non-XA JDBC drivers are:

- **Atomicity Guarantee**. An XA driver implements the XAResource interface and can participate fully in the 2PC protocol driven by the WLS Transaction Manager. This guarantees atomicity of updates across multiple participating resources.

However, a non-XA driver does not implement the XAResource interface and cannot fully participate in the 2PC protocol. When using a non-XA driver in a distributed transaction, WLS implements the XAResource wrapper on behalf of the non-XA driver. If the data source property `enableTwoPhaseCommit` is set to `true,` then the WLS XAResource wrapper returns XA_OK when the Transaction Manager invokes the `prepare()` method. When the Transaction Manager invokes `commit()` or `rollback()` during the second phase, the WLS XAResource wrapper delegates the `commit()` or `rollback()` call to the non-XA JDBC connection. Any failure during `commit()` or `rollback()` results in heuristic exceptions. Application data may be left in an inconsistent state as a result of heuristic failure.

- **Redirecting Connections.** A non-XA driver can be configured to perform updates in the same distributed transaction from more than one process, as explained in Can I use a non-XA driver in distributed transactions?. WLS internally redirects the JDBC calls made from different processes to the same physical JDBC connection in one process. However, when you use a XA driver, no such redirection will be done. Each process will use its own local XA database connection, and the database ensures that all the distributed updates made in the same distributed transaction from different processes will be committed atomically.

- **Connection Management.** Whether you are using the non-XA driver or XA driver in distributed transactions, WLS implements JDBC wrappers that intercept all the JDBC calls and obtains a physical JDBC connection from the connection pool on demand.
  - When you use a non-XA driver in distributed transactions, in order to ensure that updates made from different processes are committed atomically, WLS associates the same physical JDBC connection with the distributed transaction until it is committed or rolled back. As a result, the number of active distributed transactions using the non-XA connection pool is limited by the maximum capacity of the JDBC connection pool.
  - When you use an XA driver, the connection management is more scalable. WLS does not hold on to the same physical XA connection until the transaction is committed or rolled back. In fact, in most cases, the XA connection as only held for the duration of a method invocation. WLS JDBC wrappers intercept all JDBC calls and enlist the XAResource associated with the XA connection on demand. When the method invocation returns to the caller, or when it makes another call to another server, WLS delists the XAResource associated with the XA connection.
  - WLS also returns the XA connection to the connection pool on delistment if there are no open result sets. Also, during commit processing, any XAResource object can be used to commit any number of distributed transactions in parallel. As a result, neither the number of active distributed transactions using the XA connection pool nor the number of concurrent commit/rollbacks is limited by the maximum capacity of the

connection pool. Only the number of concurrent database access connections is limited by the maximum capacity of the connection pool.

Q. What XA drivers can I use in addition to the WebLogic jDriver for Oracle/XA?

A. Theoretically, you can use any third party XA driver that is compliant with the JDBC 2.0 standard extension specification with WLS. However, an individual vendor's XA driver may have bugs that prevent it from working properly.

Refer to JDBC Configuration guidelines for details about how to configure them at Using Third-Partjy Drivers with WebLogic Server in *Programming WebLogic JDBC*.

Q. Can I use the Oracle thin driver as an XA driver in distributed transactions?

A. Oracle 8.1.7 thin driver has threading problems, so BEA developed the following workaround: We use a dedicated XA connection for the duration of prepare, commit, and rollback operation. This is different from the default XA connection management model in that any XAResource object is used to commit any number of transactions in parallel. This limits the number of concurrent commits to the max capacity of the XA connection pool. Note that this workaround is an Oracle specific workaround and will not affect the usage of other XA drivers.

Q. Why do I get SQLException “Result set already closed” message?

Problem: I am using WebLogic jDriver for Oracle/XA (transaction mode) from the client side. Updating in a distributed transaction works fine. However, when I try to perform a query, I get `SQLException Result set already closed`. How do I work around this?

A. WebLogic jDriver for Oracle has a limitation that closes all open result sets when the method returns to the caller.

Using the driver from the server side, for example, in a bean, does not have this limitation. Using the driver from the server side is also recommended from application architecture and performance perspective. Using the driver from the client side incurs round-trip cost for every JDBC call being made.

This limitation exists because WebLogic jDriver for Oracle XA is implemented using Oracle's OCI API and C XA switch, and there is an Oracle problem when using OCI with XA in multi-threaded mode. Closing an OCI cursor in a thread that is different than the thread in which it is opened may result in server crash or unexpected behavior. As a result, the WebLogic driver implicitly closes all open result sets upon returning a call to the caller.

Q. Do I need a 2PC licence when I use JMS with one JDBC non-XA driver?

A. Yes, you do. JMS is also a XAResource that participates in the distributed transaction. Therefore, there are two resources participating in the distributed transaction, and a 2PC license is needed.

Q. Why am I getting an exception when I use JMS with a non-XA driver?

Problem: I am using JMS with one JDBC non-XA driver. Transaction fails to commit with the following exception `javax.transaction.xa.XAException: JDBC driver does not support XA, hence cannot be a participant in two-phase commit.`

A. As mentioned in the previous question "Do I need a 2PC licence when I use JMS with one JDBC non-XA driver?", JMS is also a XAResource that participates in the distributed transaction. When more than one resource is participating in the distributed transaction, you need to set the data source property EnableTwoPhaseCommit=true as explained in "Can I use a non-XA driver in distributed transactions?"

Q. Can I obtain a JDBC connection before I start a distributed transaction?

A. This depends on whether you are using a non-XA or XA driver.

- When you use a non-XA driver in a distributed transaction, always obtain a JDBC connection after the distributed transaction is begun.
- If you are using an XA driver, you can obtain the connection before or after the distributed transaction begins.

Q. Can I close a JDBC connection after the distributed transaction is committed or rolled back?

A. For both non-XA and XA driver, you can close the connection after the distributed transaction is completed.

Q. I get the following XAER_RMFAIL XAException when accessing an XAResource: "Internal error: XAResource '<name>' is unavailable". What does that mean? How should I handle it?

A. JTA has its own resource health monitoring that works as follows:

A resource is considered active either if there are no pending requests or if we get a result from any of the `XAResource` pending requests that is not an `XAER_RMFAIL`. If an `XAResource` is not active within the two minutes, it is declared dead. Any further requests to the `XAResource` are shunned, and an `XAER_RMFAIL XAException` as above is thrown. The intent is to prevent further loss of threads if the `RM` is dead.

A resource is declared active again, if you re-register the `XAResource` with the WebLogic Server Transaction Manager by calling weblogic.transaction.TransactionManager.unregisterResource followed by registerStaticResource or registerDynamicResource, or after a timeout period of 30 minutes. If you are using WLS JDBC connection pools, you only need to enable the JDBC connection pool refresh feature (by specifying the "RefreshMinutes" property of the connection pool), and, upon a successful connection pool refresh, the corresponding XAResource will be re-registered automatically. If you are registering your own XAResource, either via weblogic.transaction.TransactionManager.registerStaticResource or registerDynamicResource APIs, you will need to re-register the XAResource by calling weblogic.transaction.TransactionManager.unregisterResource followed by registerStaticResource or registerDynamicResource.

In general, a good way to debug potential RM problems is to turn on JTA XA debugging, by specifying -Dweblogic.Debug=weblogic.JTAXA as JVM parameter on WLS startup.

FAQs: Plug-Ins

Q. How does the Apache HTTP Server Plug-In work?

A. For information on how a plug-in works, see Installing and Configuring the Apache HTTP Server Plug-In in *Using Web Server Plugins With WebLogic Server*.

Q. How does the plug-in route the request for a sticky session?

A. If the browser sends a cookie, we look for "JSESSIONID" (configurable by a parameter called "CookieName") in the "Cookie: " header.

If the cookie is disabled and URL re-writing is used, the session id is encoded in the URL. In WebLogic Server 5.1 and earlier, it was encoded in the query string:

```
?WebLogicSession=my_dumy_session
```

In WebLogic Server 6.0 and later, it was encoded in the parameter:

```
;JSESSIONID=my_dummy_session
```

If no session is found in the query string or parameter and if it is small enough to be read into memory, WebLogic Server looks for the session in the postdata.

Q. What is new for debugging a plug-in in WebLogic Server 6.0?

```
"Debug = ON" logs only informational and error messages
HFC : headers from the client, informational, and error messages
HTW : headers sent to wls, informational and error messages
HFW : headers sent from wls, informational and error messages
HTC : headers sent to the client, informational and error messages
ALL : everything
OFF : nothing -- default(should be used in production)
```

The log file is configurable for 6.1. For later versions of WebLogic Server, the `WLLogFile` was introduced to configure the debug filename and location.

Q. What is expected in the wlproxy.log?

Every request looks like the following:

```
"================New Request: [GET / HTTP/1.1] ================="
```

```
PathTrim, DefaultFileName, and PathPrepend will be performed in order. The
final request will be logged as the following: "Fri Jun 22 14:24:40 2001 The
request string is '/index.jsp'"
```

Looking for session information and determining the primary:

```
"Fri Jun 22 14:24:40 2001 Initializing lastIndex=0 for a list of length=1
Fri Jun 22 14:24:40 2001 create a new server node: id='qa89:443'
server_name='mint.beasys.com', port='18080'"
```

Init SSL if SecureProxy is set to ON:

```
"Fri Jun 22 14:24:40 2001 INFO: SSL is configured Fri Jun 22 14:24:40 2001
INFO: Initializing SSL library Fri Jun 22 14:24:40 2001 Loaded 1 trusted
```

```
CA's Fri Jun 22 14:24:40 2001 INFO: Successfully initialized SSL Fri Jun 22
14:24:40 2001 INFO: SSL configured successfully"
```

Initial connection being made:

```
"Fri Jun 22 14:24:40 2001 general list: trying connect to '172.17.9.180'/443
Fri Jun 22 14:24:40 2001 Connected to 172.17.9.180:443"
```

Client headers and the post data (if present) being read:

```
"Fri Jun 22 14:24:40 2001 Hdrs from clnt:[Accept]=[image/gif,
image/x-xbitmap, image/jpeg, image/pjpeg, application/vnd.ms-excel,
application/msword, application/vnd.ms-powerpoint, */*]"
```

Client headers and the post data (if any) being sent:

```
"Fri Jun 22 14:24:40 2001 Hdrs to WLS:[Accept]=[image/gif, image/x-xbitmap,
image/jpeg, image/pjpeg, application/vnd.ms-excel, application/msword,
application/vnd.ms-powerpoint, */*]"
```

Response headers being obtained from WebLogic Server:

```
“Fri Jun 22 14:24:46 2001 Hdrs from
WLS:[Set-Cookie]=[JSESSIONID=OzI19WqYmFnRviHEu5gKLvot42ABeD8NPWnF0jW6cawSG
crp2mru!4038528127411848936!-1408169548!80!443; path=/] Fri Jun 22 14:24:46
2001 parsed all headers OK"
```

Response headers being sent to WebLogic Server and indicating if the connection is close or keep-alive is here:

```
"Fri Jun 22 14:24:46 2001 Hdrs to client:[Date]=[Fri, 22 Jun 2001 21:24:48
GMT] Fri Jun 22 14:24:46 2001 Hdrs to client:[Server]=[WebLogic WebLogic
Server 6.1 beta 06/21/2001 10:44:44 #122398 - internal build by jlee on
client jlee.qa89] Fri Jun 22 14:24:46 2001 canRecycle: conn=1 status=200
isKA=0 clen=2705 isCTE=0 Fri Jun 22 14:24:46 2001 closeConnection in
load_utils: deleting URL* Fri Jun 22 14:24:46 2001 INFO: Closing SSL context
Fri Jun 22 14:24:46 2001 INFO: sysSend 23 Fri Jun 22 14:24:46 2001 INFO:
Error after SSLClose, socket may already have been closed by peer Fri Jun
22 14:24:46 2001 r->status=200 returning 0"
```

Q. What changed in the 6.1 plug-in?

A. The following changed:

- HTTP1.1 support -- chunk-transferred and keep-alive (except for apache1.3.x)

- Session parsing (this breaks the backward compatibility)
- SSL support from the plug-in to WebLogic Server

Q. What is static list, dynamic list, and general list?

A. They are defined as follows:

- Static list: the initial server list defined in the config file
- Dynamic list: the current server list sent by wls upon a successful request
- General list: the current server list (could be static or dynamic) besides the primary and secondary servers associated with the current request

Q. Does the plug-in support two-way SSL?

A. No. But the plug-in can be set-up to require the client certificate and pass it on to WebLogic Server. For example:

```
apache ssl

SSLVerifyClient require

SSLVerifyDepth 10

SSLOptions +FakeBasicAuth +ExportCertData +CompatEnvVars +StrictRequire
```

Q. Sometimes a reponse from WebLogic Server to the plug-in contains the Set-Cookie header. Is this normal?

A. Yes. WebLogic Server will send the `Set-Cookie` header in the response if the `Cookie` header is missing from the request or `X-WebLogic-Force-Cookie` is detected. If the plug-in fails to connect to a bad server, the plug-in will send the header `X-WebLogic-Force-Cookie: true` to the next available server to force the client to update the corresponding cookie which contains the correct session information.

Q. If I install mod_wl_ssl.so with mod_perl to Apache 1.3.19, why does Segmentation Fault (11) occur in mod_wl_ssl.so when I access WebLogic via the plug-in?

A. The server could be any 6.x and the operating system could be any version of Solaris. The environment is WebLogic Server 6.x (Use `mod_wl_ssl.so`), Solaris 2.x.

In order to avoid getting Segmentation Fault (11), add a `VirtualHost` block for HTTP as in the following example:

```
#  General setup for the virtual host
DocumentRoot "/export/home/happy/local/apache_1.3.19/htdocs"
ServerName happy1
ServerAdmin happy@happyville
ErrorLog /export/home/happy/local/apache_1.3.19/logs/error_log
TransferLog /export/home/happy/local/apache_1.3.19/logs/access_log
</VirtualHost>
```

For the SSL port, also use the IP address in:

```
<VirtualHost 206.189.223.111:443>
```

Use any valid DNS name for the ServerName.

FAQs: Server-Related Questions

- What should I do if my server “hangs” or “freezes”?
- How do I configure WebLogic to use a SOCKS proxy?
- How can I speed up connection responses?
- How does WebLogic support CORBA and client communication via IIOP?
- How do I speed up HTTP tunneling?
- Can WebLogic Server start with a UNIX boot?
- What affects servlet performance besides clients and traffic?
- Why do I get “NoClassDefFound”/“Too Many Open files”messages on Solaris?
- How do I increase WebLogic Server memory?
- Java-CORBA integration: RMI-IIOP or Java IDL?
- How do an RMI-IIOP application and an existing CORBA object interoperate?
- What is the function of T3 in WebLogic Server?
- How can I debug the Java code that I have running in WebLogic Server?

Q. What should I do if my server “hangs” or “freezes”?

A. If your WebLogic Server "freezes," you will need to collect certain diagnostic information, including thread dumps and Java garbage collection metrics, before contacting BEA Technical Support. For details, see Generate a Log File and Check Garbage Collection.

Q. How do I configure WebLogic to use a SOCKS proxy?

A. You can configure a java.net socket to use SOCKS by setting a Java system property. For details, see How do I make Java work with a proxy server? at *http://www.sunsite.unc.edu/javafaq/javafaq.html#proxy*. Once the property is set, WebLogic socket connections use the SOCKS proxy.

Q. How can I speed up connection responses?

A. Connection delays are often caused by DNS problems. WebLogic performs a reverse lookup on the hostname from which a new connection is made. If the DNS reverse lookup is not working properly because the connection is coming from a proxy server, it could be responsible for the delay. You might want to work with your system administrator to determine whether DNS and the third-party networking software are working properly. Try writing a simple server program that performs a reverse lookup on any connection made to it. If that lookup is delayed, then you know that the proxy server is the source of the problem.

Q. How does WebLogic support CORBA and client communication via IIOP?

A. "CORBA" support means many things to many people. It often means simply IIOP and ORB support and does not use many CORBA services. WebLogic supports CORBA in multiple ways. The WebLogic Server 8.1 implementation of RMI-IIOP allows you to: connect Java RMI clients to WebLogic Server using the standardized IIOP protocol; connect CORBA/IDL clients, including those written in C++, to WebLogic Server; interoperate between WebLogic Server and Tuxedo clients; connect a variety of clients to EJBs hosted on WebLogic Server. For more information, see Using WebLogic RMI over IIOP at http://e-docs.bea.com/wls/docs81/rmi_iiop/index.html.

Q. How do I speed up HTTP tunneling?

A. Unfortunately, a significant performance hit occurs when you use HTTP tunneling. We have optimized it somewhat, but, because everything is encapsulated in HTTP, HTTP tunneling is slower than direct Java-to-Java TCP/IP connections.

Be sure that you really need to use HTTP tunneling. For example, if your firewall can pass IP packets through port 80, you can use the fast T3 protocol on port 80.

Q. Can WebLogic Server start with a UNIX boot?

A. You can add a startup script to your UNIX rc scripts to run WebLogic Server at UNIX boot time. For information about creating startup scripts, refer to "Starting an Administration Server Using a Script" in *Administration Console Online Help*.

Q. What affects servlet performance besides clients and traffic?

A. Response time for a servlet is about 5 times slower when you are running a screen saver on the machine, particularly for the OpenGL screen savers. Try turning off your screen saver and see if that helps!

Q. Why do I get "NoClassDefFound"/"Too Many Open files"messages on Solaris?

Problem: When I am using WebLogic Server on Solaris and try to run my application, I get a "NoClassDefFound" error, although the class causing the error does exist and is in the right directory. In fact, there are other classes in the same directory that are getting loaded. I also get a "Too many open files" error.

A. We have seen this situation when the user account runs out of file descriptors. On Solaris, each user account has a certain limited number of file descriptors. You can find out how many file descriptors you have with the `limit` command in csh.

You can increase file descriptors if you have enough privileges with the `ulimit` command in the csh. Otherwise, ask your system administrator to increase the file descriptors available to your processes.

Q. How do I increase WebLogic Server memory?

A. Increase the allocation of Java heap memory for WebLogic Server. (Set the minimum and the maximum to the same size.) This example starts the server with a fixed heap size of 200MB:

```
$ java ... -ms200m -mx200m ...
```

This allocates 32 megabytes of Java heap memory to WebLogic Server, which improves performance and allows WebLogic Server to handle more simultaneous connections. You can increase this value if necessary.

Q. Java-CORBA integration: RMI-IIOP or Java IDL?

A. It is important to understand the distinction between these two ways of integrating Java with CORBA.

RMI-IIOP is for Java programmers who want to program to the RMI interfaces but use IIOP as the underlying transport. RMI-IIOP provides interoperability with other CORBA objects implemented in various languages, but only if all the remote interfaces are originally defined as

Java RMI interfaces. It is of particular interest to programmers using Enterprise JavaBeans (EJBs), because the remote object model for EJB is RMI-based. It also allows you to use the standardized IIOP protocol and have a lightweight (minimal weblogic classes) client.

Java IDL is for CORBA programmers who want to program in Java based on interfaces defined in CORBA IDL. This is "business as usual" CORBA programming, supporting Java in exactly the same way as other languages like C++ or COBOL. If you want to integrate C++ (or any other language your orb supports into IDL) you would use the CORBA IDL method of programming.

Q. How do an RMI-IIOP application and an existing CORBA object interoperate?

A. If the existing CORBA object has its remote interfaces defined originally in CORBA IDL, then interoperability is not possible. RMI-IIOP applications can interoperate with other CORBA objects only when their remote interfaces are originally defined as Java RMI interfaces.

For example, to interoperate between an RMI-IIOP client and a C++ object you need to:

1. Define the remote interface of the object in Java as an RMI interface..
2. Run rmic -idl against the interface to produce IDL compatible with the RMI interface.
3. Run a C++ stub compiler against the IDL file to produce the C++ skeleton for your C++ server object.

Q. What is the function of T3 in WebLogic Server?

A. T3 provides a framework for WebLogic Server messages that support for enhancements. These enhancements include abbreviations and features, such as object replacement, that work in the context of WebLogic Server clusters and HTTP and other product tunneling.

T3 predates Java Object Serialization and RMI, while closely tracking and leveraging these specifications. T3 is a superset of Java Object. Serialization or RMI; anything you can do in Java Object Serialization and RMI can be done over T3.

T3 is mandated between WebLogic Servers and between programmatic clients and a WebLogic Server cluster. HTTP and IIOP are optional protocols that can be used to communicate between other processes and WebLogic Server. It depends on what you want to do. For example, when you want to communicate between

- A browser and WebLogic Server-use HTTP
- An ORB and WebLogic Server-IIOP.

Q. How can I debug the Java code that I have running in WebLogic Server?

A. You can use tools such as WebGain, JBuilder, NetBeans and JDB that rely on the Java Platform Debugger Architecture (JPDA) to debug your Java code running in WebLogic Server.

JPDA is integrated in the Java 2 Platform, Standard Edition (J2SE) SDK 1.3 on all platforms and SDK 1.2.2 for Linux. There is a download available from Sun to add JPDA support to the J2SE SDK 1.2.2 on Solaris and Microsoft Window platforms. If you are using J2SE SDK 1.2.2 on these platforms you must first get this download.

To allow a debugger to attach to the virtual machine that WebLogic runs you have to start WebLogic in debug mode. In order to start WebLogic in debug mode using a Sun virtual machine follow these steps (start with step one only if using a Solaris platform):

1. If using a Solaris platform, change the `LD_LIBRARY_PATH` environment variable to prepend `$JAVA_HOME/lib/sparc`:

   ```
   export LD_LIBRARY_PATH=$JAVA_HOME/lib/sparc:$LD_LIBRARY_PATH
   ```

2. Add the following parameters to the java command line (before the "weblogic.Server" string) that launches WebLogic server:

   ```
   -Xdebug
   -Xnoagent
   -Xrunjdwp:transport=dt_socket
   server=y
   address=<port_for_debugger_to_connect>
   suspend=n
   -Djava.compiler=NONE
   ```

Note that with the Hotspot Performance engine the `-Xnoagent` and `-Djava.compiler=NONE` options are no longer required, but are accepted and ignored for compatibility reasons.

If `server=y` and no `address` parameter is supplied, WebLogic Server chooses the transport address and prints it to the standard output stream. So, if a line such as:

```
Listening for transport dt_socket at address: 46666
```

prints in your standard output stream when the server starts, the number 46666 is the port number to be supplied to your tool's remote debugger in order to attach it to WebLogic's virutal machine.

CHAPTER 19

FAQs: Server-Side Java (Servlets)

- How do I call a servlet with parameters in the URL?
- How can I run multiple instances of the same servlet class in the same WebLogic Server instance?
- How do I deserialize an httpsession?

Q. How do I call a servlet with parameters in the URL?

A. The usual format of a servlet parameter is a *name=value* pair that comes after a question-mark (?) at the end of the URL. To access these parameters, call the `getParameter()` method on the `HttpServletRequest` object, then write code to test the strings. For example, if your URL parameters are "`func=topic`," where your URL appears as:

```
http://www.myserver.com/myservlet?func=topic
```

then you could parse the parameter as follows, where "`req`" is the `HttpServletRequest` object:

```
String func = req.getParameter("func");
if (func.equalsIgnoreCase("topic")) {
  . . . do some work
}
```

Q. How can I run multiple instances of the same servlet class in the same WebLogic Server instance?

A. If you want to run multiple instances, your servlet will have to implement the `SingleThreadModel` interface. An instance of a class that implements the

SingleThreadModel interface is guaranteed not to be invoked by multiple threads simultaneously. Multiple instances of a SingleThreadModel interface are used to service simultaneous requests, each running in a single thread.

When designing your servlet, consider how you use shared resources outside of the servlet class such as file and database access. Because there are multiple instances of servlets that are identical, and may use exactly the same resources, there are still synchronization and sharing issues that must be resolved, even if you do implement the SingleThreadModel interface.

Q. How do I deserialize an httpsession?

A. To deserialize an httpsession, construct a utility class that uses the current thread's contextclassloader to load the user defined objects within the application context. then add this utility class to the system CLASSPATH.

CHAPTER 20

FAQs: Security

- Why do I get a certificate configuration error when I start the my Weblogic Server?
- Why can't I establish an outbound SSL connection when using the demonstration certificates?
- Why do I get a configuration error when establishing an SSL connection to WebLogic Server?
- Why doesn’t my Custom security provider show up (that is, it doesn’t appear as a `Configure a new Security_Provider_Type` link) in the Administration Console?
- Why do I get a 401 Unauthorized User error when using CLIENT-CERT in the login-config of my web application?
- Why can’t I use keytool to import a digital certificate into the identity keystore?
- Can I programmatically add users and groups to the WebLogic Authentication provider?
- When using the WebLogic Credential Mapping provider, how do you create mappings for anonymous or unauthenticated users?
- How do I configure multiple Authentication providers to work together in a security realm?
- Can an application use Java security?
- When using Java security, how do I change the default permissions for an application?
- How do I protect access to the embedded LDAP server?
- Does the embedded LDAP server support SSL?
- What is the password for the LDAP Admin Account?

Q. What is the difference between the Compatibility realm and myrealm? Under what circumstances should I use each of these realms?

A. If you have a 6.x `config.xml` file and you boot WebLogic Server, the following realms are created:

- Compatibility realm—Allows you to use an existing 6.x security configuration as is in the management environment provided in this release of WebLogic Server. The Realm Adapter providers allows access to existing stores of users, groups, and access control lists (ACLs).
- myrealm—Is the default security realm in this release of WebLogic Server. By default, the WebLogic Security providers are configured in myrealm.

For more information, see *Managing WebLogic Security*.

Q. What are the default groups users and everyone used for?

A. The `users` and `everyone` groups are convenience groups that allow you to apply global roles and security policies. All WebLogic Server users are members of the `everyone` group. Only WebLogic Servers who are not the `<anonymous>` user are members of the `users` group.

For more information, see *Managing WebLogic Security*.

Q. Is there still a guest user?

A. The `guest` user is no longer supported by default in this release of WebLogic Server. In WebLogic Server 6.x, `guest` was both the name of a user and the name for anonymous logins. The new user name for an anonymous user is `<anonymous>`. You can change this username by booting WebLogic Server with the following command line argument:

```
-Dweblogic.security.anonymousUserName=newAnonymousUserName
```

This argument allows you to make the name of the anonymous user `guest` for the purpose of backward compatibility.

If you want to have a `guest` user in this release of WebLogic Server, create a user with the name `guest` in the Authentication provider in the default security realm and give the user the password of `guest`. If your code depends on the `guest` user, you should consider rewriting it to use the utility methods in `weblogic.security.WLSPrincipals`.

Q. I want to provide additional fields in my Web application for form-based authentication. What application programming interfaces (APIs) should I use?

A. The CallbackHandler implementation in the WebLogic Authentication provider supports only stringified versions of a username and password when using form-based authentication.

If a Web application requires more authentication information, use the `javax.security.auth.TextInputCallback` interface of the JAAS Callback application programming interface (API) in the code for your LoginModule. The implementation of the `javax.security.auth.TextInputCallback` interface needs to use the name of the authentication field as the prompt to the method constructor. For example:

```
Callback[] callbacks=new Callback[1];
callbacks[1]=new TextInputCallback("TextField");

try{
       callbackHandler.handle(callbacks)
       textField1=((TextInputCallback)callbacks[2].getText
} catch (java.io.IOException ioe) {
       throw new LoginException(ioe.toString());
}catch (UnsupportedCallbackException uce) {
       throw new LoginException
       ("Error:"+uce.getCallback().toString() +
       "not available to garner authentication information" +
       "from the user");
}
//"textField1 is not set correctly
```

When the `ServletCallbackHandler` gets a `TextInputCallback`, the callback looks for a field matching the prompt of the `TextInputCallback`. If there is a match, the callback handler sets the value in the callback. If no match is found, an `UnsupportedCallback` exception is raised.

Q. I am using the 6.x security realm APIs in my application. How do I upgrade this functionality to the security APIs in this release of WebLogic Server?

Specifically, I am using the `security.getRealm()` method and then the `getGroup()`, `getGroups()`, and `getUser()` methods on the returned realm.

A. You can use your 6.x application as is in the WebLogic Server environment by using Compatibility security.

The management of WebLogic Server changed in 6.x away from direct calls on the realm to the use of MBeans. This change was the first step in creating a unified administration model for WebLogic Server. While the Security MBeans introduced in 6.x mirrored the capabilities defined for the realm, they were not flexible enough to allow security vendors to integrate their products with WebLogic Server. This release of WebLogic Server provides a set of Security Service Provider Interfaces (SSPI) and Security SPI MBeans that allow you to write custom security products for WebLogic Server. These MBeans can also be used to replicate functionality in an existing application so that the application can be used in the WebLogic Server environment.

If you choose to upgrade your realm, you will need to write an implementation of the MBeans found in the `weblogic.management.security` package. These MBeans allow you to implement management methods that are specific to your 6.x realm. For a description of using the MBeans in the `weblogic.management.security` package, see *Developing Security Providers for WebLogic Server*.

The following are some hints that may help you upgrade applications based on security realms to the security architecture available in this release of WebLogic Server:

- If you are using the security realm to authenticate users, instead use the JAAS API to authenticate users.
- Table 20-1 lists the mappings between the interfaces in the 6.x of the `weblogic.security.acl` package to the interfaces in the `weblogic.managment.security.authentication` package.

Note: The `weblogic.security.acl` package is deprecated in this release of WebLogic Server.

Table 20-1 Interface Mapping

Methods in the 6.x weblogic.security.acl package	Corresponding Method in the weblogic.management.security. authentication package
`newUser()`	`UserEditor.createUser()`
`deleteUser()`	`UserRemover.removeUser()`
`newGroup()`	`GroupEditor.createGroup()`
`deleteGroup()`	`GroupRemover.removeGroup()`
`Group.addMember()`	`GroupEditor.addMemberToGroup`
`Group.removeMember()`	`GroupEditor.removeMemberFromGroup()`
`Group.isMember()`	`GroupMemberLister.listGroupMembers()` or `SubjectUtils.isUserInGroup()`
`Group.members()`	`GroupMemberList.listGroupMembers()`
`userExists()`	`UserReader.isUser()`
`isGroupMember()`	`GroupReader.groupExists()`, `GroupReader.isMember()`

Q. Does WebLogic Server support Diffie-Hellman or DSS/DSA digital certificates?

A. No. The exportable version of WebLogic supports only 512 bit RSA with 40 bit RC4. Additionally, Web browsers do not support these types of certificates, and there are no commercial issuers for DSA certificates.

Q. Can a Weblogic Server deployment have one RSA and one non-RSA certificate?

A. No.

Q. Must we pay RSA licensing costs for non-RSA client code?

A. WebLogic Server has licensed RSA for SSL communications between WebLogic Server and WebLogic clients. When using WebLogic Server, no extra licensing for RSA is necessary, although different rules apply to VARs.

Q. How do I use Netscape security certificates with WebLogic Server?

A. Netscape stores the private and public keys together in a key-pair file which prevents you from separating the public and private keys. Therefore, you must generate another request for a certificate, not using Netscape utilities.

Q. How do I restrict access to servlets and JSPs?

The Java Servlet API Specification v2.3, at http://www.java.sun.com/products/servlet/download.html#specs, allows you to declaratively restrict access to specific Servlets and JSPs using the Web Application deployment descriptor. Section 13.3.2 of the specification has an example deployment descriptor that uses declarative security. For more information, see Programming WebLogic HTTP Servlets. You can also specify roles for EJBs and Web applications through the Administration Console. For more information, see *Securing WebLogic Resources*.

Q. Can I use RSA encryption algorithms and javax.crypto.* API to build applications?

A. No. WebLogic's RSA license does not permit end-users to use RSA classes directly. You must obtain your own license for the encryption libraries from RSA.

Q. Can I use a JNDI Initial Context to pass the security credentials of a WebLogic Server user?

A. The ability to use JNDI to pass security credentials was deprecated in 6.1 of WebLogic Server. You can still use this method in this release WebLogic Server. However, BEA recommends using the Java Authentication and Authorization Service (JAAS) `runAs()` method rather than JNDI to associate a user with a security context. For more information, see *Programming WebLogic Security*.

Q. Are WebLogic Server passwords secure?

A. The `config.xml` file no longer has clear text passwords. In place of clear text passwords, the `config.xml` file has encrypted passwords. You cannot copy encrypted passwords from one domain to another. Instead, you can edit the `config.xml` file and replace the existing encrypted passwords with clear text passwords and then copy the file to the new domain. The Administration Console will encrypt the passwords the next time it writes to the file.

Q. Why do I get a certificate configuration error when I start the my Weblogic Server?

For example: `Alert> <WebLogicServer> <Security> configuration problem with certificate file`

A. It is possible that you did not specify a *`WL_HOME`* relative file name in your SSL configuration files.

For more information, see *Managing WebLogic Security*.

Q. Why can't I establish an outbound SSL connection when using the demonstration certificates?

A. When establishing an SSL connection, the subject DN of the digital certificate must match the host name of the server initiating the SSL connection. Otherwise, the SSL connection is dropped. If you use the demonstration certificates, the host names will not match. To avoid this situation, use the following command-line argument when starting WebLogic Server:

```
-Dweblogic.security.SSL.ignoreHostnameVerification=true
```

This argument disables the Hostname Verifier which compares the subject DNs and host names. This solution is recommended in development environments only. A more secure solution is to obtain a new digital certificate for the server making outbound SSL connections.

Q. Why do I get a configuration error when establishing an SSL connection to WebLogic Server?

For example: `<WebLogic Server> <SSLListenThread listening on port 8802> Failed to connect to t3s://localhost:8802.`

A problem with the configuration of the SSL protocol will also raise the following exception:

```
<java.io.IOException: Write Channel Closed, possible handshaking or trust
failure>
```

A. By default, WebLogic Server contains a Hostname Verifier that compares the subject DNs of digital certificates and host names. When establishing an SSL connection, the subject DN of the digital certificate must match the host name of the server initiating the SSL connection. If you use the demonstration certificates the host names will not match. To avoid this situation, use the following command-line argument when starting WebLogic Server:

```
-Dweblogic.security.SSL.ignoreHostnameVerification=true
```

This argument disables the Hostname Verifier. This solution is recommended in development environments only. A more secure solution is to obtain a new digital certificate for your WebLogic client.

In this release of WebLogic Server, WebLogic clients perform a trusted certificate authority check on the digital certificate for WebLogic Server. The client may reject the digital certificate of WebLogic Server if the certificate was not issued by a certificate authority trusted by the client. Previous versions of WebLogic Server did not perform this check.

Q. Why does my servlet return a `no certificate` message?

A. Unless WebLogic Server is configured to ask a client for its digital certificate during the SSL handshake (referred to as Two-Way SSL), WebLogic Server will not have the digital certificate. You get this error when a WebLogic servlet or JSP tries to perform peer validation on the client. Set the Client Certificate Enforced attribute when configuring SSL to requireWebLogic Server to request client certificates.

Q. Why doesn't my Custom security provider show up (that is, it doesn't appear as a `Configure a new` *`Security_Provider_Type`* link) in the Administration Console?

A. Check to make sure the system administrator put the MBean JAR file (MJF) in the `lib/mbeantype` directory.

Q. Why do I get a 401 Unauthorized User error when using CLIENT-CERT in the login-config of my web application?

A. To use a login-config of CLIENT_CERT, ensure the following:

1. Two-way SSL is configured on the server with the Client Enforced option set.
2. The web application is access via https.
3. A user corresponding to the CN attribute of the digital certificate for the web application is defined as a user in the default security realm and that the security realm has an Identity Assertion provider configured.

CLIENT_CERT also applies when perimeter authentication is used (meaning digital certificates are coming in via http headers or cookies). In this case, two-way SSL and https are not required.

Q. Why can't I use keytool to import a digital certificate into the identity keystore?

A. When using the keytool utility to create a Certificate Signing Request (CSR), a self-signed digital certificate is placed in the identity keystore. On occasion, a problem will occur when using the keytool commands to replace the self-signed digital certificate in the keystore with a signed digital certificate received from a trusted CA. Use the keytool utility to import the trusted CA certificate into the keystore before importing the signed digital certificate into the keystore. The steps are as follows:

1. Obtain the trusted CA certificate.
2. Use the `der2pem` utility to convert the trusted CA certificate to a PEM file.
3. Use the keytool utility to create an identity keystore. For example:

   ```
   keytool -genkey -alias subjectkey -keypass keypassword -keystore
   nameofkeystore -storepass keystorepassphrase
   ```

4. Use keytool `certreq` command to create a CSR and submit the CSR to a certificate authority. For example:

   ```
   keytool -certreq -alias subjectkey -keypass keypassword keystore
   nameofkeystore -storepass keystorepassphrase -file mycertificate.cer
   ```

5. Use the keytool `import` command to import the PEM file for the trusted CA into the keystore.
6. Use the keytool `import` command with the `-trustcacerts` option to import the signed digital certificate from the certificate authority into the keystore.

Q. Can I programmatically add users and groups to the WebLogic Authentication provider?

A. Use the `createUser()` and `createGroup()` interfaces in the `weblogic.management.security.authentication` class.

Q. When using the WebLogic Credential Mapping provider, how do you create mappings for anonymous or unauthenticated users?

A. When using the Resource container, special usernames are established for different cases. A mapping called `wls_ra_anonymous` is used when there is not authenticated WebLogic user (meaning, the Subject has no users or is null). The Resource container also defines special mappings for mappings created during a Resource Adapter's pool startup (`wls_ra_initial`) and when no other configured mapping applies to the given Subject (`wls_ra_default`).

Q. How do I configure multiple Authentication providers to work together in a security realm?

A. The Login Modules for all the different types of supported Authentication providers conform to the JAAS specification. If there are two Authentication providers configured in a security realm and you want valid users from either provider to login into the system, set the JAAS Control Flag on each Authentication provider to `REQUISITE`.

Q. Can an application use Java security?

A. Yes. An application can use Java security as well as JAAS authorization within itself. The application can use `checkPermission()` and all the other Java security calls. The only caveat is there is no guarantee of your identity or code base when the application is being called from the server. At all entry points, your identity and code base must be re-established using `AccessController.doPrivileged()` or `Subject.DoAs()`.

Q. When using Java security, how do I change the default permissions for an application?

A. In the Java security policy file, there are three "fake" codebases for each of the three component types:

`"file:/weblogic/application/defaults/EJB"` for EJBs

`"file:/weblogic/application/defaults/Web` for Web applications

`"file:/weblogic/application/defaults/Connector` for Connector applications

These codebases contain the defaults for these types of applications. When evaluating these defaults, the string `"WEBLOGIC-APPLICATION-ROOT"` will be expanded to the top-level directory when the application was deployed.

Q. How do I protect access to the embedded LDAP server?

A. The `acls.prop` file (located in `WLHOME/server/lib/acls.prop`) controls access to the embedded LDAP server. You can modify the `acls.prop` file to control access to the LDAP server. For more information, see Managing the Embedded LDAP Server.

Q. Does the embedded LDAP server support SSL?

A. Yes. Use the SSL listen port (7002). The WebLogic Server muxer is actually performing all the SSL work so there should be no additional configuration required when using SSL to talk to the embedded LDAP server.

Q. What is the password for the LDAP Admin Account?

A. The password is specified in the Credential field of the EmbeddedLDAP MBean. The password is generated on the first boot of the server and written to the config.xml file. The LDAP Admin account DN is `cn=Admin`. Use the WebLogic Server Administration Console to change the password for the LDAP Admin account. For more information, see Configuring the Embedded LDAP Server.

A.

FAQs: Upgrading

Q. What is Compatibility mode?

A. When you upgrade to WebLogic Server 8.1 from previous versions of WebLogic Server, Compatibility mode allows you to keep your old configuration of users, groups, and ACLs.

Q. How do I run WebLogic Server 8.1 in Compatibility mode when upgrading?

A. WebLogic Server 8.1 recognizes your old config.xml file and automatically runs in Compatibility mode.

Q. What is two-phase deployment?

A. WebLogic Server 8.1 uses two-phase deployment. Previous to WebLogic Server 7.0, when you deployed an application, a copy of the application file(s) was sent to all the targeted servers, which in turn, loaded the application. If any of those servers failed (or partially failed), the deployment was placed in an inconsistent state.

In the current release of WebLogic Server, the application is first prepared across the servers and then activated in a separate phase. In the prepare phase, the application is prepared for deployment without allowing user requests to the application. When the servers are ready, the application is activated everywhere. With this model, it is still possible to have a failure occur during the activation phase, which leads to the inconsistent state, but it is much less likely to occur.

For more information on this deployment model and other 8.1 deployment features, see *WebLogic Server Application Deployment*.

Q. Why do I keep getting a NoSuchMethodError when running an application on WebLogic Server 8.1?

A. Make sure that you are not running the server with a patch for a previous version of WebLogic Server. For exact information of a particular patch, contact *Customer Support*.

Also make sure that you do not have any JAR files in your server `CLASSPATH` that conflict with the version of J2EE that is supported by WebLogic Server 8.1, J2EE version 1.3. The 1.3 version of J2EE is added to the WebLogic Server 8.1 `CLASSPATH` by default. For example, you may have to remove `j2ee12.jar` from your server `CLASSPATH`.

FAQs: Web Services

- I am getting a java.lang.IncompatibleClassChangeError exception when I execute the clientgen or autotype Ant task to generate non-built-in data type components from an XML Schema file. This only started happening after I installed WebLogic Platform or WebLogic Workshop; everything worked just fine when WebLogic Server was installed on its own. What is going on?
- Does WebLogic Server 8.1 support SOAP Messages with Attachments?
- Does WebLogic Server 8.1 support SOAP?

Q. I am getting a java.lang.IncompatibleClassChangeError exception when I execute the clientgen or autotype Ant task to generate non-built-in data type components from an XML Schema file. This only started happening after I installed WebLogic Platform or WebLogic Workshop; everything worked just fine when WebLogic Server was installed on its own. What is going on?

A. This problem happens only when you have the wlxbean.jar and xbean.jar files in your build or deployment environment CLASSPATH variable and you are using the clientgen or autotype Ant tasks to generate non-built-in data type components from an XML Schema that references SOAP encoding data types. The two JAR files, used by the XMLBeans feature, are part of WebLogic Platform and Workshop, and not part of WebLogic Server. The JAR files contain SOAP encoding data type classes which come into conflict with those generated by the Ant tasks.

There are three different ways to work around this problem:

- Use the packageName or packageBase attributes of the autotype Ant task (or the typePackageName or typePackageBase attributes of the clientgen Ant task) to force the Ant-task-generated SOAP encoding data types into a user-specified package so that they do not conflict with the XMLBeans data types.

See Web Service Ant Tasks and Command-Line Utilities at http://e-docs.bea.com/wls/docs81/webserv/anttasks.html for information about the attributes of the autotype and client Ant tasks.

- Remove the `xbean.jar` and `wlxbean.jar` files from your build and deployment environment. These files are installed into the *`WL_HOME`*`\server\lib` directory by WebLogic Platform, where *`WL_HOME`* refers to the main WebLogic Platform installation directory, such as `c:\beahome\weblogic81`. If you need these JAR files (because some of your Web Services use XMLBeans), then put a copy of the two JAR files in the `APP-INF/lib` directory of only those EAR files that contain WebLogic Web Services that use XMLBeans.
- If you are not using WebLogic Workshop to create applications, or do not deploy any applications generated by WebLogic Workshop, then you can remove the conflicting SOAP encoding classes from the `wlxbean.jar` and `xbean.jar` files.

Q. Does WebLogic Server 8.1 support SOAP Messages with Attachments?

A. Yes. As long as the data type of the attachment is on the list of supported JAX-RPC data types, then WebLogic Server will automatically handle parameters to Web services that are SOAP attachments. If you want to do further processing of the attachment, you can use handlers to intercept the request and response SOAP message (which includes the attachment.)

Q. Does WebLogic Server 8.1 support SOAP?

A. Yes. WebLogic Server's implementation of SOAP is included as part of the Web Services subsystem. For detailed information on creating Web Services with WebLogic Server, see Programming WebLogic Web Services at http://e-docs.bea.com/wls/docs81/webServices/index.html.

FAQs: Wireless-Related Questions

Q. What is a wireless (mobile) device?

A. A wireless device in this context is a device that has connectivity to the Internet without being physically plugged into a network with a wire. The most common examples of these are the Internet-enabled cell phone such as a WAP Phone or i-Mode phone, personal digital assistant (PDA) such as Palm VII, Pocket PC such as Wireless iPaq, and pager such as RIM Blackberry.

Q. Can WebLogic Server support wireless devices?

A. Yes. For information on wireless support, see *Using WAP with WebLogic Server*. Wireless examples, if installed, are located in the /samples/examples directory of your WebLogicServer installation and are available from the Start menu.

Q. What should I consider when writing an application for wireless devices?

A. The following is a list of factors to consider when writing applications for wireless clients:

- Content should be personalized based on the location of the device and/or the type of the device.
- Device microbrowsers are usually not HTML-based; some are WML-based, some are cHTML-based, some are HDML-based, some use Web Clipping, etc.
- Some devices have additional features such as Bluetooth, power keys and SMS messaging that can be used to enhance an application.
- Often the devices can be used with voice portals using voice recognition and text-to-speech.
- Most of the devices have smaller screens that may or may not display color or graphical images.
- These screens range in form from a vertically-oriented rectangle to a square to a horizontally-oriented rectangle.
- Many of the devices have a difficult data entry and selection mechanism with numeric keypads or styli.
- Connectivity is often lost when the device is mobile.
- The devices usually have no PKI security capability.
- A few of the devices are limited in the amount of data that can be transmitted to them for a page.

For more information see Wireless, Internet and Email.

Q. What is WAP and what is i-Mode?

A. WAP and i-Mode are the two major over-the-air (OTA) protocols for wireless Internet communication with cell phones and some other devices. WAP stands for Wireless Application Protocol and is found predominantly in Europe and North America. i-Mode is the protocol used in Japan by NTT DoCoMo. A new protocol called WAP-NG (WAP Next Generation) is being considered as a replacement for both of these protocols.

Both WAP and i-Mode are comprised of both the OTA protocol and a markup language understood by their microbrowsers. The WAP markup language is WML (Wireless Markup Language) and cHTML (Compact HTML) is the markup language specified by i-Mode. WML and cHTML may be superseded by XHTML (Basic) in the future.

Note that other wireless carriers and devices use protocols and markup languages other than WAP and i-Mode. For example, the Palm VII uses web clipping over a proprietary protocol.

For more information about WAP, see http://www.wapforum.org/faqs.

Q. Will my solution work on different networks such as CDMA, GPRS, TDMA and PDC-P?

A. Yes. WAP and i-Mode were designed to hide the network details from the application developer. They will both work equally well on any underlying network. Therefore, as carriers upgrade their networks, applications written for either WAP or i-Mode will continue to work without any need for modifications. As the networks are upgraded to higher speeds, the performance of wireless applications written in either WML or cHTML should improve as well.

Q. What changes do I need to consider for 3G wireless networks?

A. None. As described in the previous answer, neither WAP nor i-Mode is dependent on the underlying network. However, a developer may consider enriching an application with streaming media or other content that requires greater bandwidth.

FAQs: XML

- Which XML parser comes with WebLogic Server 8.1?
- Is there an XSLT processor included in WebLogic Server 8.1?
- The built-in XML parser is based on Apache's Xerces 2.1.0. Can I upgrade the built-in XML parser to a later Xerces version?
- The built-in XSLT processor is based on Apache's Xalan 2.2D11. Can I upgrade the built-in XSLT processor to a later Xalan version?
- What version of the JAXP API specification is implemented in WebLogic Server 8.1?
- Can I use the getAttribute() and setAttribute() methods of Version 2.3 of the Java Servlet API to parse XML documents?
- How do I identify the document type of an XML document?
- I get a `java.lang.LinkageError` exception when I use Ant to run a client application that uses the WebLogic JAXP implementation. What is the problem?

Q. Which XML parser comes with WebLogic Server 8.1?

A. We bundle the following two parsers with WebLogic Server 8.1:

– a built-in parser based on Apache's Xerces 2.1.0 parser

– WebLogic FastParser, a high-performance non-validating parser that you can use for small to medium sized XML documents.

The WebLogic XML Registry allows you to configure the parser you want to use for specific document types.

You can also use the WebLogic XML Streaming API to parse your XML documents. The Streaming API is based on the SAX API, but provides a more procedural, stream-based handling of XML documents rather than having to write SAX event handlers, which can get complicated when dealing with complex XML documents.

Q. Is there an XSLT processor included in WebLogic Server 8.1?

A. Yes, the one that ships in the JDK 1.4.1_02: Apache's Xalan 2.2D11. This is the built-in XSLT processor for WebLogic Server 8.1.

Q. The built-in XML parser is based on Apache's Xerces 2.1.0. Can I upgrade the built-in XML parser to a later Xerces version?

A. Yes. To upgrade to version 2.2.0, 2.3.0, or 2.4.0 of Apache's Xerces, follow these steps:

1. Append the appropriate Apache `xercesImpl.jar` file to the end of WebLogic Server's CLASSPATH variable. The CLASSPATH variable is typically set in the scripts used to start WebLogic Server. The `xercesImpl.jar` file contains Apache's implementation of the version of Xerces to which you want to upgrade.

   You can also put the `xercesImpl.jar` file in the `WEB-INF/lib` directory of the WAR file of your application. You do not need to enable the `PreferWebInfClasses` flag for your Web application.

   Warning: Do not use the archives from Apache named `xml-apis.jar` or `xmlParserAPIs.jar`.

2. Configure your XML Registry to use `org.apache.xerces.jaxp` for the `DocumentBuilderFactory` or `SAXParserFactory` factories. For details, see *Administering WebLogic Server XML* at http://e-docs.bea.com/wls/docs81/xml/xml_admin.html.

Q. The built-in XSLT processor is based on Apache's Xalan 2.2D11. Can I upgrade the built-in XSLT processor to a later Xalan version?

A. Yes. For instructions, see Endorsed Standards Override Mechanism at http://java.sun.com/j2se/1.4.2/docs/guide/standards/.

The version of Xalan to which you upgrade will be used from any user application code that gets a transformer through JAXP. WebLogic Server itself uses this transformer when processing Web Service attachments and for the JSP XSLT tag library.

Q. What version of the JAXP API specification is implemented in WebLogic Server 8.1?

A. Version 1.1. This version includes pluggable XML transformation as well as pluggable XML parsing.

Q. Can I use the getAttribute() and setAttribute() methods of Version 2.3 of the Java Servlet API to parse XML documents?

A. Yes. Use the `setAttribute()` method for SAX mode parsing and the `getAttribute()` method for DOM mode parsing. Using these methods in a Servlet, however, is a WebLogic-specific feature. This means that the Servlet may not be fully portable to other Servlet engines, so use the feature with caution.

Q. How do I identify the document type of an XML document?

A. If the XML document has a Public ID, then that is its document type. For example, if an XML document contains the following DOCTYPE declaration:

```
<!DOCTYPE mydoc PUBLIC "My public ID String"
              "http://foo.com/url/to/my/dtd">
```

then its document type is `My public ID String`.

If the DOCTYPE declaration does not contain a Public ID, but specifies a System ID, then the document type is the System ID. For example, in the following DOCTYPE declaration:

```
<!DOCTYPE mydoc SYSTEM "http://foo.com/url/to/my/dtd">
```

the document type is `http://foo.com/url/to/my/dtd`.

Note: The System ID is of the DTD, not of the XML document itself. It can, however, still be used as a way to identify the XML document.

If the XML document does not specify a DOCTYPE declaration, then the document type can be either the root element name or the namespace URI, if it has one.

Q. I get a `java.lang.LinkageError` exception when I use Ant to run a client application that uses the WebLogic JAXP implementation. What is the problem?

A. The problem might have to do with Ant classloading. The workaround is to fork the VM by setting the `fork="true"` attribute of the `java` Ant task while running the application.

The full text of the error is as follows:

```
[java] java.lang.LinkageError: loader constraints violated when linking org
/xml/sax/InputSource class
[java] at
weblogic.xml.jaxp.RegistryDocumentBuilderFactory.newDocumentBuilder(RegistryDo
cumentBuilderFactory.java:128)
```

FAQs: WebLogic Express

- What is WebLogic Express?
- What J2EE technologies are supported by the latest version of WebLogic Express?
- When should I upgrade from WebLogic Express to WebLogic Server?
- Is the upgrade from WebLogic Express to WebLogic Server difficult?
- Does WebLogic Express support Web Services?
- Does WebLogic Express support Clustering?
- How do I install a new WebLogic Express license?
- What is a WebLogic Express Domain?
- Can a domain include a mix of WebLogic Express and WebLogic Server instances?
- What version of the Java Servlet and JSP specifications does WebLogic Express support?
- Do I need to make any changes to my Web application currently deployed in Tomcat in order to move it to WebLogic Express?
- How should I use the existing BEA e-docs if I am a WebLogic Express user?
- Can I build Web services with WebLogic Express? If so, how?
- How does the Console differ between WebLogic Express and WebLogic Server?
- Where can I download WebLogic Express for an evaluation?

Q. What is WebLogic Express?

A. BEA WebLogic Express (WLX) is a production-proven Java servlet engine designed to get you up and running on the BEA WebLogic™ product line in a cost-effective manner. WebLogic Express allows you to quickly develop and launch dynamic Web sites and basic Web applications that do not require the full suite of application server capabilities. WebLogic Express is a good fit for projects that don’t yet use advanced technologies such as EJB, JMS, and JCA, and instead favor Servlets, Java Server Pages (JSP), Java objects, RMI, and JDBC. WebLogic Express notably does not support EJB, JMS, JCA, the two-phase commit protocol for transactions, or some other features targeted at enterprise-level applications.

Q. What J2EE technologies are supported by the latest version of WebLogic Express?

A. WebLogic Express 8.1 supports Java Servlets 2.3, JSP 1.2, JDBC 2.0, JNDI 1.2.1, JTA 1.01a, JMX 1.0, and JavaMail 1.1.3.

Q. When should I upgrade from WebLogic Express to WebLogic Server?

A. WebLogic Express is a good fit for less complex applications built on Servlets, JSPs, Java objects, and simple data access. You can use RMI for client-server and client-client communication, and even build basic Web Services using these technologies.

Reasons to migrate to WebLogic Server from WebLogic Express could include the following:

- Your application needs an upgrade - Perhaps you need to repartition your application layers to create a separate business logic/EJB layer to meet certain business needs or to manage architecture complexity.
- You need to integrate with other applications supported with JCA adapters.
- You need a messaging infrastructure (i.e., JMS) for you application.
- You need some of the more advanced features that come with WebLogic Server such as two-phase commit transactions, message-driven beans, and object tier clustering and caching.

Q. Is the upgrade from WebLogic Express to WebLogic Server difficult?

A. Upgrading from WebLogic Express to WebLogic Server is actually very easy. Since WebLogic Express and WebLogic Server share the same code base, your applications will run as is – without any modifications - on WebLogic Server.

Upgrading is as easy as copying over a new WebLogic Server license file to your existing WebLogic Express installation.

Q. Does WebLogic Express support Web Services?

A. Yes, WebLogic Express is Web Services enabled. WebLogic Express offers presentation-level Web Services with the latest XML, SOAP, and JAX-RPC standards.

Q. Does WebLogic Express support Clustering?

A. Yes, WebLogic Express Premium Edition has full clustering capabilities for your mission-critical Web applications. Clustering support in Premium Edition includes clustered JDBC, in-memory replication of servlet sessions with automatic failover, and load balancing features.

Q. How do I install a new WebLogic Express license?

A. The WebLogic Express license file resides under the BEA_HOME directory of your installation. The license file contains licenses for all products of the BEA WebLogic Platform.

A. If you do not have any other Weblogic products installed, you could simply copy over the new WebLogic Express license file to the existing one. Otherwise -- and preferably -- install the new WebLogic Express license with the following steps:

1. Save the new license file with a name other than license.bea in the target BEA Home directory. For example, save the file as platform_license.bea. Use this file as the license_update_file in step 4 of this procedure.
2. Open a command shell and go to the target BEA Home directory.
3. If it is not already included, add the JDK to your PATH variable by entering the following commands:
   - On a Windows system: set PATH=BEA_HOME\jdk131_03\bin;%PATH%
   - On a UNIX system: PATH=BEA_HOME/jdk131_03/bin:$PATH

   export PATH
4. Merge the license update file into your existing license by entering one of the following commands:
   - On a Windows system: UpdateLicense license_update_file
   - On a UNIX system: sh UpdateLicense.sh license_update_file.

   license_update_file is the name to which you saved the license update file in step 1.
5. Save a copy of your updated license.bea file in a safe place outside of the BEA_Home directory or on a different machine.

Q. What is a WebLogic Express Domain?

A. For ease of administration and management, one or more WebLogic Express instances – and their associated resources - are grouped together in an administrative unit called a domain. A domain can include multiple WebLogic Express clusters and non-clustered WebLogic Express instances. A single Administration Server manages a domain.

A domain could consist of only one WebLogic Express instance. However, that sole server instance would serve as an Administration Server, because each domain must have exactly one Administration Server.

You can define multiple domains based on different system administrators' responsibilities, application boundaries, or geographical locations of servers.

Q. Can a domain include a mix of WebLogic Express and WebLogic Server instances?

A. Yes, you can have WebLogic Express and WebLogic Server instances in the same domain. The domain would be managed by a single Administratiion Server and Administration Console.

Q. What version of the Java Servlet and JSP specifications does WebLogic Express support?

A. WebLogic Express 8.0 supports Java Servlet version 2.3 and JSP version 1.2.

Q. Do I need to make any changes to my Web application currently deployed in Tomcat in order to move it to WebLogic Express?

A. Any Web application deployed in standard .WAR format, as per the Servlet specification, will deploy without code changes on WebLogic Express. You may have to define the resource and resource mappings in the WebLogic Express server if your application depends on any WebLogic Express resources.

Q. How should I use the existing BEA e-docs if I am a WebLogic Express user?

A. BEA has provided a separate eDocs page for WebLogic Express. This page provides links to most of the documentation related to WebLogic Express features, installation, programming, management, and “Getting Started” material. You will find that many of the links in the WebLogic Express documentation pages actually point to WebLogic Server documentation pages. This is because WebLogic Express shares the same code base as WebLogic Server and WebLogic Server documentation for common WebLogic Server/WebLogic Express features can be applied to WebLogic Express.

Q. Can I build Web services with WebLogic Express? If so, how?

A. Yes, you can build and deploy Web Services in WebLogic Express using Java classes. See *Overview of WebLogic Web Services* at http://e-docs.bea.com/wls/docs81/webserv/overview.html for instructions on building Web services for WebLogic Express. Note that in WebLogic Express you can only use Java classes to implement Web services. You cannot use EJBs or JMS.

Q. How does the Console differ between WebLogic Express and WebLogic Server?

A. There is no difference between the WebLogic Express and WebLogic Server console.

Q. Where can I download WebLogic Express for an evaluation?

A. You can download a trial version of WebLogic Express at http://commerce.bea.com/downloads/weblogic_express.jsp.

To help you with your WebLogic Express evaluation, there is a WebLogic Express evaluation guide that will guide you through the process of a WebLogic Express installation, creating and deploying a Web application on WebLogic Express, and reviewing WebLogic Express features.

Q. Can I get JDBC drivers with WebLogic Express?

A. Yes, BEA offers two JDBC drivers that come with WebLogic Express:

- Type 2 native JDBC driver for Oracle.
- Type 4 JDBC driver for Microsoft SQL Server.

In addition, you can use any JDBC-compliant driver with WebLogic Express. For more information on WebLogic Express and JDBC drivers, see *Introduction to WebLogic JDBC* at http://e-docs.bea.com/wls/docs81/jdbc/intro.html.

Q. Can I use WebLogic Express as the Web tier in an n-tiered application architecture?

A. Yes, WebLogic Express will fit your Web-tier needs very well. WebLogic Express can be used as your primary Web server and also host advanced J2EE Web applications built with HTML, XML, Servlets, JSP, applets, etc. WebLogic Express has complete Web server functionality, including a built-in HTTP server.

Q. Can I use WebLogic Express as an admin server in a larger WebLogic Server cluster?

A. Yes, you can use WebLogic Express as the admin server in a larger WebLogic Server Cluster.

Q. Can I do integration with WebLogic Express?

A. WebLogic Express is intended to be use as a Java Servlet/Web application engine. It does not support the integration features that come with WebLogic Server like J2EE Connector Architecture (JCA) and WebLogic Server-Tuxedo connectors.

Q. How does WebLogic Express integrate with my IDE?

A. WebLogic Express can work with various third party modeling, development and testing tools. For information on tools support and integration kits see *Tools and Utilities for WebLogic Server 8.1* at http://e-docs.bea.com/wls/docs81/toolstable/ToolsTable.html.

Q. When should I consider Open Source vs WebLogic Express?

A. Please see our white paper discussing the considerations involved in making a choice between WebLogic Express and Open Source Servlet Engines.

Q. Does WebLogic Express support virtual hosting?

A. WebLogic Express supports virtual hosting that allows a single WebLogic Express instance or WebLogic Express cluster to host multiple Web sites. Each logical Web server has its own host name, but all Web servers are mapped in DNS to the same cluster IP address.

When a client sends an HTTP request to the cluster address, a WebLogic Express instance is selected to serve the request. The Web server name is extracted from the HTTP request headers and is maintained on subsequent exchanges with the client so that the virtual host name remains constant from the client's perspective.

Multiple Web applications can be deployed on a WebLogic Express instance, and each Web application can be mapped to a virtual host.

Q. Does WebLogic Express work with other popular Web servers?

A. WebLogic Express can either act as your primary Web server or can be used as the servlet engine behind a Web server like Microsoft IIS, Apache HTTP Server, or Netscape Enterprise Server.

WebLogic Express includes plug-ins that enable WebLogic Express applications to communicate with these other Web servers. For information on using plugins see *Using Web Server Plug-Ins With WebLogic Server* at http://e-docs.bea.com/wls/docs81/plugins/index.html.

Q. Can I use WebLogic Express as a Web server?

A. Yes, WebLogic Express has complete Web server functionality. This allows a Web browser to request pages from WebLogic Express using standard HTTP protocol. Used as a Web server, WebLogic Express supports several standard features like security, SSL, virtual hosting, Proxy configurations, load balancing, and automatic failover. For more information on WebLogic Express as Web server refer to *WebLogic Server Services* at http://e-docs.bea.com/wls/docs81/intro/chap2.html.

Q. Which platforms are WebLogic Express certified on?

A. For a list of certified platforms see Supported Configurations at http://e-docs.bea.com/platform/suppconfigs/index.html.

Q. Can I use other BEA products with WebLogic Express, like WebLogic Workshop, WebLogic Portal, or WebLogic Integration?

A. WebLogic Express can be a front-end to these other BEA WebLogic products and to the BEA WebLogic Platform.

Q. How does clustering work with WebLogic Express Premium edition?

A. WebLogic Express Premium Edition includes clustering and failover of JSPs, Servlets, RMI objects, and JDBC connections for increased reliability and availability. For more information on WebLogic Express Clustering features see *Introduction to WebLogic Server Clustering* at http://e-docs.bea.com/wls/docs81/cluster/overview.html.

Q. What is dev2dev Support?

A. BEA now offers dev2dev Support for WebLogic Express customers. BEA dev2dev Support provides comprehensive Web-based support for easier and more productive implementations so you can meet your project goals. Developers have access to extensive online technical information, best practices, and automated online help request services – all allowing you to tap into the expertise of BEA Support.

Dev2dev Support is an additional offering to the pre-existing Production Support offering from BEA. BEA Production Support is a more comprehensive offering, ensuring your key business processes are up and running. Combining 24x7 response for faster problem resolution with proactive services to leverage best practices, BEA Production Support improves your IT team's productivity. The BEA Production Support offering allows all members of your team access to BEA expertise through phone or Web, as well as full access to online self-help resources.

Q. Can I access CORBA applications with WebLogic Express?

A. Yes. WebLogic Express contains support for RMI over IIOP, which can be used for connectivity with CORBA applications. Because WebLogic Express does not contain support for EJB, you can only use RMI over IIOP to communicate with plain RMI objects, however.

More information on RMI over IIOP, see *Overview of RMI over IIOP* at http://e-docs.bea.com/wls/docs81/rmi_iiop/rmiiiop1.html.

Q. What kind of applications can I build with WebLogic Express?

A. WebLogic Express is designed for building and deploying simple Web applications that do not require the full application server capabilities. WebLogic Express is a good fit for projects that use Servlets and Java Server Pages (JSPs), and simple Java applications using Java classes, RMI, and JDBC. WebLogic Express does not support EJB, JMS, JCA, advanced Web services, or other features targeted at enterprise-level applications.

Q. Are there any newsgroups for WebLogic Express?

A. You can access the BEA newsgroups for your particular area of interest at http://www.bea.com/support/newsgroup.shtml.

Q. How do I start developing on WebLogic Express?

A. The best place to start is with the "Getting Started" section of WebLogic Express.

There is also a WebLogic Express evaluation guide and toolkit that you can download. The guide helps you to get started quickly with WebLogic Express installation and Web application programming with a self paced tutorial.

Q. Where can I take classes on Java development?

A. Information can be found at http://www.bea.com/education/index.shtml.

Q. Why does a JMS error message appear when I start WebLogic Express?

A. The Domain Wizard adds JMS tags when it creates a new domain, and when this domain is used to start the server, the JMS tags are noted, and the error message is displayed. If you have not specified any JMS resources in your domain, you can ignore this error message and continue to run the server.