BEA WebLogic Server

and BEA WebLogic Express

Release Notes for WebLogic Server 6.0

BEA WebLogic Server 6.0
Document Date: June 4, 2001

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

BEA WebLogic Server 6.0 Release Notes

Part Number	Document Date	Software Version
860-001003-008	June 4, 2001	BEA WebLogic Server Version 6.0

Contents

1. What's New in WebLogic Server 6.0

2. Upgrading WebLogic Server to Version 6.0

3. Migrating WebLogic Server Applications to Version 6.0

4. Notes and Problems

5. Resolved Issues

About This Document

This document introduces the current release of BEA WebLogic Server™ 6.0. It contains important information on new features, known problems, and bug fixes. It also contains information for users who are upgrading from an earlier release of WebLogic Server.

The document is organized as follows:

- Chapter 1, "What's New in WebLogic Server 6.0," is an overview of WebLogic Server 6.0. It contains important information concerning the current release of the WebLogic Server and any accompanying Service Packs.
- Chapter 2, "Upgrading WebLogic Server to Version 6.0," contains information for users who are moving to WebLogic Server 6.0 from an earlier version of WebLogic Server.
- Chapter 3, "Migrating WebLogic Server Applications to Version 6.0," is an overview of moving your application from an older version of WebLogic to WebLogic Server 6.0.
- Chapter 4, "Notes and Problems," is a list of the known issues and problems that .have been found in WebLogic Server 6.0.
- Chapter 5, "Resolved Issues," describes all bug fixes that have been made to WebLogic Server 6.0.

Audience

This document is written for all WebLogic users who are new to WebLogic Server 6.0.

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.

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`*`;`

<table>
<tr><th>Convention</th><th>Usage</th></tr>
<tr><td>UPPERCASE TEXT</td><td>Device names, environment variables, and logical operators.
<em>Examples</em>:
LPT1
BEA_HOME
OR</td></tr>
<tr><td><code>{ }</code></td><td>A set of choices in a syntax line.</td></tr>
<tr><td><code>[ ]</code></td><td>Optional items in a syntax line. <em>Example</em>:
<code>java utils.MulticastTest -n <em>name</em> -a <em>address</em></code>
<code>   [-p <em>portnumber</em>] [-t <em>timeout</em>] [-s <em>send</em>]</code></td></tr>
<tr><td><code>|</code></td><td>Separates mutually exclusive choices in a syntax line. <em>Example</em>:
<code>java weblogic.deploy [list|deploy|undeploy|update]</code>
<code>   password {application} {source}</code></td></tr>
<tr><td><code>...</code></td><td>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</td></tr>
<tr><td>.
.
.</td><td>Indicates the omission of items from a code example or from a syntax line.</td></tr>
</table>

CHAPTER

1 What's New in WebLogic Server 6.0

The following release notes apply to WebLogic Server 6.0 and all subsequent WebLogic Server 6.0 Service Pack releases. The sections below describe changes and features in WebLogic Server 6.0:

- J2EE Standards Improvements
- WebLogic Server Services
- Administration and Management
- Security
- Internationalization
- XML
- WebLogic Server Tour
- BEA Jolt
- Important Service Pack Information
- Miscellaneous

J2EE Standards Improvements

WebLogic Server 6.0 reflects improvements in the J2EE specifications. It supports J2EE version 1.2, EJB 2.0, and JMX features of J2EE version 1.3.

Enterprise Java Beans (EJBs)

WebLogic Server 6.0 includes many enhancements to WebLogic EJBs. For more information, see *Programming WebLogic EJBs* at `http://e-docs.bea.com/wls/docs60/ejb/index.html`.

EJB 1.1 Compliance

WebLogic Server 6.0 is compliant with the JavaSoft EJB 1.1 Specification at `http://java.sun.com/products/ejb/index.html`. The WebLogic Server™ EJB documentation describes key features of the EJB 1.1 Specification that you need to understand in order to use WebLogic Server. However, you should also read the JavaSoft EJB 1.1 Specification to familiarize yourself with basic EJB development practices.

See *Programming WebLogic EJBs* at `http://e-docs.bea.com/wls/docs60/ejb/index.html` for a summary of EJB 1.1 changes that may require updates to your EJB classes, interfaces, and client applications.

Additional EJB Features

Additional new features include:

- Application assembly support in deploying EJBs
- DDConverter upgrade utility
- Full JNDI support for EJB environment
- New resource factory support

- Support for EJB home handles
- Improved free pool management
- Improved manageability for multiple persistence types
- Support for `is-modified-method-name` with bean-managed EJBs
- New Java type support for multi-object finders

EJB 2.0

Download the EJB 2.0.jar file to use the EJB 2.0 features. The download is available at http://commerce.bea.com/downloads/products.jsp.

The EJB 2.0 for BEA WebLogic Server container is based on a nonfinal specification. It provides many new EJB 2.0 features, including:

- Support for message-driven beans
- EJB 2.0 container-managed persistence services, including support for EJB-Query Language
- Support for entity EJB home methods
- EJB 2.0 XML deployment properties

In addition to EJB 2.0 features, the new container provides several new features specific to WebLogic Server:

- In-memory replication for stateful session EJBs in a cluster
- Database locking option for entity EJBs
- Improved EJB deployment and redeployment capabilities
- EJB 2.0 example beans
- `isModified()` no longer required for EJB 2.0 CMP entity beans

Because the container is based on pre-release versions of the EJB 2.0 Specification, certain EJB 2.0 features are not yet available. Known differences between this container and the publicly available EJB 2.0 Specification are acknowledged in the EJB documentation where applicable.

The following EJB 2.0 container and WebLogic Server features will not be supported until a future release of the EJB 2.0 for BEA WebLogic Server container:

- Support for the final EJB 2.0 Specification
- Dependent objects

Hypertext Transfer Protocol (HTTP) 1.1

The following sections refer to HTTP features supported with WebLogic Server 6.0.

Web Server

WebLogic Server 6.0 is a functional Web server that can handle high volume Web sites, serving static HTML (text) files as well as servlets and JavaServer Pages (JSP). WebLogic Server 6.0 can also fully integrate with hardware- and software-based Web load balancing solutions. WebLogic Server 6.0 supports the HTTP 1.1 standard.

Each WebLogic Server 6.0 hosts a default *Web Server* and any number of additional Web Servers that you define. Each of these additional Web Servers is configured to respond to a different DNS name, in a process called Virtual Hosting.

You configure the attributes for Web Servers using the new WebLogic Server Administration Console.

For more information, see *Programming WebLogic HTTP Servlets* at `http://e-docs.bea.com/wls/docs60/servlet/index.html`.

HTTP Plug-ins for Apache, Netscape, and IIS

WebLogic Server 6.0 automatically implements keep-alive connections between the plug-ins and WebLogic Server. As of Service Pack 1, WebLogic Server 6.0 now also supports SSL when using the plug-ins.

JavaMail

WebLogic Server 6.0 includes an implementation of the JavaMail Specification. This is the standard reference implementation of the JavaMail Specification. For more information, see *Developing WebLogic Server Applications* at `http://e-docs.bea.com/wls/docs60/programming/index.html`.

Java Transaction API (JTA)

WebLogic Server 6.0 supports distributed transactions and the two-phase commit protocol with a specification-compliant JTA implementation. This implementation works with any certified XA-compliant resource such as WebLogic JMS and the WebLogic jDriver for Oracle.

For more information, see *Programming WebLogic JTA* at `http://e-docs.bea.com/wls/docs60/jta/index.html`.

WebLogic Server Services

The following services are supported with WebLogic Server 6.0.

Clustered JMS

Using clustered JMS, you can configure multiple JMS servers and assign them to defined WebLogic Servers. Each JMS server can be deployed on a single WebLogic Server and handles requests for a set of destinations. This supports load balancing across multiple servers in a cluster.

You can establish cluster-wide, transparent access to destinations from any server in the cluster by configuring multiple connection factories. Applications use JNDI to look up a connection factory and create a connection to communicate with a JMS server.

For more information, see *Programming WebLogic JMS* at `http://e-docs.bea.com/wls/docs60/jms/index.html`.

Multicasting

WebLogic JMS supports multicasting, allowing for the delivery of messages to a select group of hosts using an IP multicast address. Multicasting reduces the number of messages required to be delivered by a JMS server. The JMS server forwards only a single copy of a message to each host group associated with a multicast IP address, regardless of the number of applications that have subscribed.

XML Message Types

WebLogic JMS supports messages containing Extensible Markup Language (XML). Use of the XMLMessage type facilitates message filtering, which is more complex when performed on XML content shipped in a TextMessage.

Java Database Connectivity (JDBC) MultiPools

JDBC MultiPools create a list of connection pools to be used by a single WebLogic Server. A configurable algorithm determines which connection is returned. MultiPools provide support for load balancing and high availability. MultiPools make it easier for an application to switch to another RDBMS for distributed processing or during a failover situation.

For more information, see *Programming WebLogic JDBC* at `http://e-docs.bea.com/wls/docs60/jdbc/index.html`.

In-Memory Replication of Stateful Session EJBs

WebLogic Server 6.0 extends the existing HTTP session replication model to session EJBs. It has the ability to do in-memory replication of stateful session EJBs. This feature provides performance failover of business logic.

For more information, see *Programming WebLogic EJBs* at `http://e-docs.bea.com/wls/docs60/ejb/index.html`.

New Installation Infrastructure

A completely new installation program makes it easier to install WebLogic Server on both Windows and UNIX systems. The installer unpacks the distribution, performs basic configurations, and sets up shortcuts for using WebLogic Server. In addition, the JDK is included in the package so that the server is ready to run.

For details, see the *WebLogic Server Installation Guide* at `http://e-docs.bea.com/wls/docs60/install/index.html`.

Class Loader and CLASSPATH Improvements

CLASSPATH and the class loader are easier to use. For details see *Resolving Class References Between Components* at `http://e-docs.bea.com/wls/docs60/programming/packaging.html`.

ZAC (Zero-Administration-Client)

WebLogic Zero Administration client (ZAC) is now supported with WebLogic Server 6.0 as of Service Pack 1. For more information see the documentation at *Using the Zero Administration Client* at `http://e-docs.bea.com/wls/docs60/zac/index.html`.

Administration and Management

WebLogic Server 6.0 provides a new management architecture which enables you to make dynamic changes to the configuration of running WebLogic Servers. A new Web-based Administration Console is your window into the WebLogic

Administration Service, an implementation of the Java Management Extension (JMX) standard. Using the Administration Console, you can configure attributes, deploy applications and components, monitor resource usage, view log messages, and perform other management activities. Features include:

- Browser interface
- Centralized user and security management
- Dynamic configuration management
- Administrative domains
- Management of one or more clusters
- A centralized application repository
- Metrics on the status of servers
- Centralized access to log messages for all servers in a domain

For more information, see the *WebLogic Server Administration Guide* at `http://e-docs.bea.com/wls/docs60/adminguide/index.html`.

Security

The security functionality of WebLogic Server is being continuously improved. WebLogic Server 6.0 includes:

- JAAS Login replaces realm authentication
- Enhanced support for generating audit trails
- ACL improvements
- Password-guessing protection
- Denial-of-service protection
- Improved administration of the RDBMS realm

For more information, see *Programming WebLogic Security* at `http://e-docs.bea.com/wls/docs60/security/index.html`.

Default and Custom Login Modules

WebLogic Server uses the default LoginModule (`weblogic.security.internal.ServerLoginModule`) to gather authentication information during server initialization. To replace the default Login module, edit the `server.policy` file and replace the name of the default Login module with the name of a custom Login module.

Optionally, custom Login modules can be specified in the `server.policy` file ahead of the default LoginModule. The JAAS implementation in WebLogic Server uses Login modules in the order in which they are defined in the `server.policy` file. The default Login module checks for existing system user authentication definitions prior to execution and does nothing if they are already defined.

The default Login Module is required to define JVM properties for both the system username and password. These properties are specified as `weblogic.management.username` and `weblogic.management.password` respectively. In order to use a custom Login module, these properties must be set accordingly.

Internationalization

WebLogic Server 6.0 is able to handle content and deliver it in any language, including those languages that require double byte character sets. For information about using the new internationalization API, see the *WebLogic Server Internationalization Guide* at `http://e-docs.bea.com/wls/docs60/i18n/index.html`.

In addition, a Kanji version of WebLogic Server 6.0 will be available. For more information, contact your BEA sales representative.

XML

WebLogic Server 6.0 supports XML as an essential component. JSPs can be used to generate and consume XML between servers or between a server and clients. WebLogic Server 6.0 supports XSL processing tags for JSPs. EJBs use XML to describe deployment properties, which provides data portability. The server provides an XML schema repository for DTDs, managed by the new Administration Console.

For more information, see *Programming WebLogic XML* at `http://e-docs.bea.com/wls/docs60/xml/index.html`.

WebLogic Server Tour

The WebLogic Server Tour is completely revised. It provides an overview of WebLogic Server using the Pet Store application to demonstrate features. The Tour is available from the Start menu.

BEA Jolt

The BEA Jolt® product is no longer packaged with WebLogic Server. For information on upgrading the BEA Jolt client for WebLogic Server 6.0, refer to “Upgrading BEA Jolt” on page 2-8 of these Release Notes.

Important Service Pack Information

BEA has released WebLogic Server 6.0 with Service Pack 2. Service Pack 2 includes many new improvements and fixes to the original 6.0 release; it also includes the fixes that were made in Service Pack 1.

For a list of issues fixed in Service Pack 2, see Chapter 5, "Resolved Issues." For a list of open issues still found in WebLogic 6.0 with Service Pack 2, see "Known Problems." on page 4-5. The following section summarizes changes included in Service Pack 2. Users who have installed WebLogic Server 6.0 with Service Pack 2 should read the following sections carefully.

Administration Console

The Administration Console is updated. The Service Pack installer contains a new version of the `console.war` file and places it in the default domain (in `config/mydomain/applications`). If you have previously created any new domains, you must manually copy the new `console.war` into the `/config/`*`domain name`*`/applications` directory for each domain.

Clustering

Clusters of WebLogic Servers no longer require the use of a shared network drive. Both the WebLogic Server installations and your applications can now reside on local file systems.

EJB

- In order to run on WebLogic Server 6.0 with Service Pack 2, EJB 1.1 CMP applications must be recompiled against the Service Packs class libraries. You need to run `ejbc` once the Service Pack has been installed.

- Message-driven EJBs now support external JMS Providers. This change was tracked as Issue 041284.
- CMP entity beans will now by default delay a database insert after `ejbPostCreate` is called. If you want to do the database insert after invoking `ejbCreate`, set the `delay-database-insert-until` tag to `ejbCreate.` This change was tracked as issue 043272.
- WebLogic Server now supports the `max-elements` tag used in `weblogic-cmp-rdbms-jar.xml`. The `max-elements` tag is used to specify the maximum number of elements that should be returned by a multi-valued query. This option is similar to the `maxRows` feature of JDBC.
- When calling EJBs from a JSP or servlet in a Web Application, you no longer need to include the EJB home and remote interfaces in the `WEB-INF/classes` directory of the Web Application.

Examples and Documentation

- Using an HP-UX platform, the `config/petstore/StartPetStore.sh` script has been modified to use the Hotspot Virtual Machine. This is a workaround for a bug in the HP client VM whereby the service pack installer overwrites the existing `StartPetStore.sh` file. If you have made any changes to `StartPetStore.sh`, your changes will be lost. This change is being tracked as Issue 041463.
- New documentation has been added for the `config.xml` file. *WebLogic Server Configuration Reference*, at `http://e-docs.bea.com/wls/docs60/config_xml/index.html`, describes the WebLogic Server configuration file (`config.xml`).

HTTP Servlets and JSP

- The format for cookies has changed as per issue 041528:
    - Cookies created by WebLogic Server 6.0 are not recognized by WebLogic Server 6.0, Service Pack 2.

  - Cookies no longer include the context path of a Web Application. This change allows for a single authentication to be used for all Web Applications in a domain. If you need a Web Application to use a unique authentication or to generate a cookie that is unique to the Web Application, you can specify a cookie name for each Web Application in the weblogic.xml deployment descriptor. For more information, see *Multiple Web Applications and Cookies* at `http://edocs/wls/docs60/adminguide/config_web_app.html#cookiename`.
  - The delimiter used in the cookie has changed from "/" and "|" to "!". The "/" delimeter caused URL rewriting to fail and the "|" delimiter caused Wireless Application Protocol (WAP) gateways to fail. Both URL rewriting and WAP now function correctly with WebLogic Server.

- Several new options have been added to the WebLogic JSP compiler (`weblogic.jspc`). These new options allow you to compile JSPs that use EJBs in Web Applications. For details, see *JSP Descriptor Element* at `http://e-docs.bea.com/wls/docs60/programming/weblogic_xml.html#jsp-descriptor`.

JDBC

A new column, `wl_max_inactive_interval`, has been added to the `wl_servlet_sessions` table used for JDBC-based persistence. This change was tracked as issue number 041867. For information on the `wl_servlet_sessions` table, see our *Configuring Web-Applications* documentation at `http://www.weblogic.com/docs60/adminguide/config_web_app.html`.

jDriver for Oracle

New binaries have been added for 8.1.7 Oracle client libraries.

Platforms

WebLogic Server 6.0 Service Pack 1 and Service Pack 2 support the HotSpot virtual machine (VM) and the Classic VM for HP-UX, but the default start script uses the Classic VM.

Security

ACLs on MBeans

Access Control Lists (ACLs) have been added to the WebLogic MBeans to enable secure remote access to the MBeans. This new ability **only** applies to remote access of MBeans. By default, only clients authenticated as the `system` user have access to the MBeans.

To allow other users to access an MBean, add following ACL:

```
access.weblogic.admin.mbean.Mbean.instance name=user or group with
access
```

For example, the following ACL enables access to the Server MBean named `myserver2` for users that are members of the `Testing` Group:

```
access.weblogic.admin.mbean.myserver2=Testing
```

Use the following ACL to grant access to all the WebLogic MBeans:

```
access.weblogic.admin.mbean=user or group with access
```

If clients fail in their attempt to access an MBean, a `weblogic.management.NoAccessRuntimeException` is returned. The server log contains the details indicating which user attempted to access which resource.

Using the Java Security Manager with WebLogic Server 6.0 Service Pack 2

When you run WebLogic Server under Java 2 (JDK 1.2 or 1.3), WebLogic Server can use the Java Security Manager in Java 2 to provide additional access control for WebLogic Server resources. The Java Virtual Machine (JVM) has security mechanisms built into it that can be managed through a security policy file. The Java

Security Manager can enforce a set of permissions granted to `CodeSource` or `SignedBy` classes. The permissions allow certain classes running in that instance of the JVM to do or not do certain run-time operations. In many cases, where the threat model does not include malicious code being run on the JVM, the Java Security Manager is unnecessary. When an Application Service Provider uses WebLogic Server and unknown classes are being run, the Java Security Manager is necessary. To use the Java Security Manager with WebLogic Server, specify the `-Djava.security.manager property` when starting WebLogic Server.

Note: In past releases of WebLogic Server, the Java Security Manager was enabled by using the `-Dweblogic.security.manager` property when starting WebLogic Server. Please note the change in the property for WebLogic Server version 6.0 and greater. The property is now named `-Djava.security.manager.`

The Java Security Manager uses a security policy file that defines permissions. The full pathname of the security policy file is specified in the `-Djava.security.policy property` when starting WebLogic Server. If you enable the Java Security Manager but do not specify a security policy file, the Java Security Manager uses the default security policies defined in the `java.security` and `java.policy` files in the `$JAVA_HOME/lib/security` directory.

WebLogic Server includes an example security policy file named `weblogic.policy`. This file contains a set of default permissions. You need to make the following edits to the file in order to use the file with your WebLogic Server deployment.

1. Edit the following lines in the `weblogic.policy` file, replacing the specified location with the location of your WebLogic Server installation:

```
grant codebase "file://BEA/-"{
  permission java.io.FilePermission "D:${/}BEA${/}=", ...
```

Note: In this example it is assumed that your installation directory structure is the same as the one described in “BEA Home Directory” in the BEA WebLogic Server Installation Guide.

2. If you want to run the Administration Console, add the following grant block and permissions to the `weblogic.policy` file:

```
grant {
    permission java.io.FilePermission
"D:{/}BEA${/}wlserver600${/}weblogic${/}management${/}console${/}
-",
"read";
```

```
    permission java.io.FilePermission
"D:{/}BEA${/}wlserver600${/}config${/}mydomain${/}applications${/
}.wl_te
mp_do_not_delete${/}weblogic${/}management${/}console${/}-",
"read";
    permission java.util.PropertyPermission   "user.*",   "read";
   };
```

3. If you have extra directories in your CLASSPATH or if you are deploying applications in extra directories, you need to add specific permissions for those directories to your `weblogic.policy` file.

BEA also recommends taking the following precautions:

- Make a backup copy of the `weblogic.policy` file and put the backup copy in a secure location.
- Set the permissions on the `weblogic.policy` file such that only the adminstrator of the WebLogic Server deployment has write and read privileges.

To use the Java Security Manager and the `weblogic.policy` file with your WebLogic Server deployment, use the following properties when starting WebLogic Server:

```
$java... -Djava.security.manager\
-Djava.security.policy==D:/BEA/wlserver600/lib/weblogic.policy
```

For more information about the Java Security Manager, see the Javadoc shipped with Java 2.

The `RecordingSecurityManager` utility can be used to detect permission problems that occur when starting and running WebLogic Server. The utility outputs permissions that can be added to your security policy file to resolve the permission problems that the utility finds. The `RecordingSecurityManager` is available at the BEA Developer's Center.

SSL with Plug-ins for Apache, Netscape, and IIS

With Service Pack 1 or higher, WebLogic Server 6.0 now supports SSL when you use the plug-ins. The *WebLogic Administration Guide* at `http://e-docs.bea.com/wls/docs60/adminguide/index.html` contains information on SSL and each plug-in.

Miscellaneous

If your domain is active, do not edit the `config.xml` file, as explained in the `config.dtd`. It is strongly recommended that changes to your configuration file be made through the Administration Console. Changes made to `config.xml` by any other means than the Administration Console may not be saved if the domain is active.

WebLogic uses an open source product called Beanshell. The initial developer of the original Beanshell code is Pat Niemeyer. Portions created by Pat Niemeyer are Copyright (C) 2000. All Rights Reserved. Beanshell is available under the Sun Public License and the GNU Lesser General Public License. The source code to Beanshell can be found at http://www.beanshell.org.

CHAPTER

2 Upgrading WebLogic Server to Version 6.0

The following sections contain information necessary to upgrade your system from WebLogic Server 5.1 to WebLogic Server 6.0:

- Upgrading WebLogic Server License Files
- Upgrading Your WebLogic Configuration
- WebLogic Server Tools
- JSP Error Handling
- JVM Upgrade Issues
- Platform Support Issues
- LDAP Security Realm

Upgrading WebLogic Server License Files

The Java format license file (`WebLogicLicense.class`) and the XML-format license file (`WebLogicLicense.XML`) are no longer supported. These files were used in conjunction with earlier releases of WebLogic Server and must be converted to a new format. The new license file is called `license.bea`. For instructions on how to convert your licenses to the new format, see *Installing a WebLogic Server License* at `http://e-docs.bea.com/wls/docs60/install/instlic.html#instlic005`.

Upgrading Your WebLogic Configuration

The following sections describe the changes required to upgrade your configuration to WebLogic Server 6.0.

webLogic.properties File Replaced

In earlier versions of WebLogic Server, configuration properties were stored in the `weblogic.properties` file. In WebLogic Server 6.0, configuration attributes for the server are stored in a persistent XML file, `config.xml`. Configuration attributes for applications are stored in `.xml` files specific to that particular application. Depending on your application you may have `web.xml`, `weblogic.xml`, or `application.xml` files associated with your application. The `weblogic.properties` file that handled configuration in earlier releases of WebLogic is not used in WebLogic Server 6.0.

Convert your existing `weblogic.properties` file to the appropriate `.xml` files using the Administration Console. Procedures for converting your `weblogic.properties` file are provided in the *Console Help* documentation at `http://e-docs.bea.com/wls/docs60/ConsoleHelp/conversion.html`.

- Do not edit the `config.xml` file directly. Access to the configuration is provided through the Administration Console, a command line utility, or programmatically through the configuration API. For details on configuring WebLogic Server, refer to the *Administration Guide* at `http://e-docs.bea.com/wls/docs60/adminguide/config.html`.
- Security properties are stored in the `fileRealm.properties` file.
- The `weblogic.common.ConfigServicesDef` API, which provided methods to get properties out of the `weblogic.properties` file, has been removed from this version.
- The `weblogic.jsp.windows.caseSensitive` property is a system property. Specify this in the command line, not in the `weblogic.properties` file. Make sure to set the following in your WebLogic Server start script: `-Dweblogic.jsp.windows.caseSensitive=true`

- The `weblogic.httpd.compileJspEnabled` property has been removed. In its place, you must add the following value to the JSPServlet in your `weblogic.properties` file: `neverStale=true`

WebLogic Classpath Property

Earlier versions of WebLogic Server used the WebLogic classpath property (`weblogic.class.path`) to facilitate dynamic class loading. In WebLogic 6.0, the `weblogic.class.path` is no longer needed. You can now load classes from the Java system classpath. To include the classes that formerly were specified in `weblogic.class.path` in the standard Java system classpath, use the `-classpath` option on the command line, or set the `CLASSPATH` environment variable.

Startup Scripts

If you used WebLogic Server startup scripts with a previous version of the product, modify them to work with 6.0.

- Modify the startup scripts as described in *Setting the Classpath Option* at `http://e-docs.bea.com/wls/docs60/adminguide/startstop.html#settingclasspath`. The WebLogic classpath is no longer used; use the Java system classpath as described in the preceding section, *WebLogic Classpath Property.*
- It is no longer necessary to include the license file in the classpath.
- With the new management system, there is a distinction between an Administration Server and Managed Servers. Consequently, scripts that start servers must be rewritten according to how you plan to administer your servers. For the new commands and their required arguments, see *Starting and Stopping WebLogic Servers* at `http://e-docs.bea.com/wls/docs60/adminguide/startstop.html` as well as *Migrating from Earlier Versions of WebLogic Server* at `http://e-docs.bea.com/wls/docs60/adminguide/startstop.html#MigratingStartScripts`.

JSP Error Handling

The behavior of the JSP include directive has changed between WebLogic Server 5.1 and the current version. In versions through WebLogic Server 5.1, the JSP include directive logged a Warning-level message if it included a non-existent page. In WebLogic Server 6.0 and later, it reports 500 Internal Server Error in that case.You can avert the error by placing an empty file at the referenced location.

Database Issues

BEA, mirroring Oracle's support policy, supports the Oracle releases called out in *Installing and Using WebLogic jDriver for Oracle* at `http://e-docs.bea.com/wls/docs60/oracle/index.html`. BEA no longer supports the following Oracle client versions: 7.3.4, 8.0.4, 8.0.5, and 8.1.5.

If you want to use the Oracle Client Version 7.3.4, you can use the backward compatible oci816_7 shared library. As stated above, BEA no longer supports this configuration.

If you want to upgrade to Oracle Client Version 8.1.6, or read detailed documentation on the WebLogic jDriver and Oracle databases, see *Installing and Using WebLogic jDriver for Oracle* at `http://e-docs.bea.com/wls/docs60/oracle/index.html`.

For supported platforms, as well as DBMS and client libraries, see the BEA *Platform Support Page* at `http://www.weblogic.com/platforms/index.html`. The most current certification information will always be posted on the Platform Support page.

JMS Upgrade

The following table lists the WebLogic JMS configuration attributes that have been added since WebLogic Server 5.1.

For information on migrating an existing application, see *Migrating WebLogic Server Applications* at `http://e-docs.bea.com/wls/docs60/jms/migrat.html`.

Component	**JMS Attribute**	**Description**
JMS Connection Factories	Message Maximum	Maximum number of messages that can exist for an asynchronous session and that have not yet been passed to the message listener. This attribute defaults to 10.
	Overrun Policy	Overrun policy for multicast sessions. When the number of outstanding messages reaches the Messages Maximum attribute value, messages are discarded based on the specified policy. If set to KeepNew, the most recent messages are given priority over the oldest messages, and the oldest messages are discarded, as needed. If set to KeepOld, the oldest messages are given priority over the most recent messages, and the most recent messages are discarded, as needed. This attribute defaults to KeepOld.
	Allow Close In OnMessage	Flag for specifying whether a connection factory creates message consumers that allow a `close()` method to be issued within its `onMessage()` method call. This attribute is disabled by default.
	Transaction Timeout	Timeout value (in milliseconds) for transacted sessions. This attribute defaults to 3600.
	User Transactions Enabled	Flag for specifying whether a connection factory creates sessions that are JTA aware. This attribute is disabled by default.
JMS Servers	JMS Default Connection Factories Enabled	Flag specifying whether the JMS default connection factories are instantiated on the JMS server. This attribute is enabled by default.

Component	JMS Attribute	Description
JMS Destinations	Enable Store	Flag specifying whether the destination uses the backing store specified by the JMS server. If this flag is enabled, but no backing store is defined, then the configuration fails and WebLogic JMS does not boot. If this flag is disabled, then the destination does not support persistent messages. If this flag is set to Default, then the destination uses the backing store if one is defined. This attribute defaults to Default.
	Multicast Address	IP address used for multicasting. This address is used to transmit messages to multicast consumers. This attribute has no default.
	Multicast TTL	Time-to-live value used for multicasting, specifying the number of routers that the message can traverse en route to the consumers. This attribute defaults to 0.
	Multicast Port	IP port used for multicasting. This port is used to transmit messages to multicast consumers. This attribute defaults to 6001.

Windows Convenience Programs Removed

The Windows convenience programs are no longer supported. The following programs have been removed:

- `dbping.exe`
- `install.exe`
- `remove.exe`
- `wlconfig.exe`
- `wlconsole.exe`
- `wlserver.exe`

WebLogic Server Tools

The following sections describe changes to WebLogic Server tools.

Console

The Java-based WebLogic Console and the Administration Servlets have been deprecated and replaced with the Administration Console. Refer to the *Administration Guide* at `http://e-docs.bea.com/wls/docs60/ConsoleHelp/index.html` for details on using the Administration Console.

Command-Line Interface

New command-line interfaces are provided with WebLogic 6.0. In addition, command line parameters have changed. The URL now refers to the Administration Server, and an additional *`servername`* parameter allows you to indicate the server on which a specific command is to run.

For more information, see *Commands for Managing a WebLogic Domain* at `http://e-docs.bea.com/wls/docs60/adminguide/cli.html`.

EJB Deployer

The EJB Deployer tool is no longer provided. Deploy EJBeans in one of the following ways:

- Use the Administration Console to load EJBeans into WebLogic Server.
- Place the `.jar` file or `.jar` directory in the `/config/examples/applications` directory. The EJBeans automatically deploy when WebLogic Server starts.

For details on deploying EJBeans, see *Programming WebLogic Enterprise JavaBeans* at `http://e-docs.bea.com/wls/docs60/ejb/index.html`.

Upgrading BEA Jolt

BEA Jolt is currently packaged with the BEA Tuxedo® product, and is not delivered with WebLogic Server 6.0.

If you currently use WebLogic Server 5.1 and BEA Jolt to connect to BEA Tuxedo applications, you will need to download the pre-GA version of BEA Jolt Release 8.0, which is compatible with the WebLogic Server 6.0 software. The pre-GA version of BEA Jolt Release 8.0 will allow you to maintain your WebLogic Server 5.1 feature functionality, and is available via the *BEA Download Center* at `http://www.bea.com/download.shtml`.

Zero Administration Client (ZAC)

ZAC is supported in WebLogic Server 6.0. If you wish to use ZAC, you must install Service Pack 1 or any subsequent Service Pack for WebLogic Server 6.0.

Utilities

The following Java utility programs are no longer available:

- `utils.eventsnoop`
- `utils.eventviewer`
- `utils.httptest`
- `util.itertest`

WebLogic Tour

The WebLogic Tour is brand new. It uses a version of the J2EE PetStore Demo and is available from the Start menu.

JVM Upgrade Issues

To run WebLogic Server 6.0, you **must** upgrade to JDK 1.3. Only Java client applications can use JDK 1.2 and above. The latest information regarding certified JVMs is available at our *Platform Support Page* at `www.weblogic.com/platforms/index.html`.

Platform Support Issues

WebLogic does rigorous testing on a variety of platforms. Only those platforms that pass our internal tests are certified. For the latest information on supported platforms, plug-ins, and database drivers see our *Platform Support Page* at `www.weblogic.com/platforms/index.html`.

LDAP Security Realm

The LDAP security realm has been rewritten to provide improved performance and configurability. BEA recommends upgrading your LDAP security realm to take advantage of this functionality. The LDAP upgrade can be downloaded from http://commerce.bea.com/downloads/weblogic_server.jsp#wls under "Modules for WebLogic Server 6.0".

When upgrading to the new LDAP security realm, place the `ldaprealm.properties` file in the `wlserver6.0` directory.

CHAPTER

3 Migrating WebLogic Server Applications to Version 6.0

The following sections describe a list of issues you need to consider when migrating your applications from WebLogic Server Version 4.5 or 5.1 to WebLogic Server Version 6.0. For a more detailed procedure, see the *Migration Tutorial* on *BEA dev2dev*. Where indicated, additional details are provided in the feature-specific documents:

- Java2 Enterprise Edition (J2EE)
- Converting weblogic.properties to .xml files
- Setting Up Your New Domain
- Migrating Applications
- Other J2EE Services
- Deprecated APIs and Features
- Removed APIs and Features

Java2 Enterprise Edition (J2EE)

The following sections discuss migration issues for each of the J2EE services. In conformance with the J2EE specification, WebLogic 6.0 deploys applications as one or more of the following types:

- Web Applications that may contain servlets, JSPs, and HTML
- EJB Applications containing Enterprise Java Beans
- Enterprise Applications containing a combination of EJB components and Web Application components
- Client Applications which are Java classes that connect to WebLogic Server using Remote Method Invocation (RMI)

Each of the above application types have their own deployment descriptors and directory format. If you are using legacy applications that were used with WebLogic Server 4.5 or WebLogic Server 5.1, you may need to make changes in order to migrate them successfully to WebLogic Server 6.0. This includes creating XML deployment descriptors and arranging resources according to a prescribed directory format. **There is no longer a weblogic.properties file in WebLogic Server 6.0.** The configuration of your applications is handled through the WebLogic Server Administration Console and configuration information is stored in an XML file.

The following sections contain information on converting your existing `weblogic.properties` files to the new XML format and converting your existing applications to the appropriate J2EE application type.

Converting weblogic.properties to .xml files

Earlier releases of WebLogic Server used a `weblogic.properties` file to configure applications. In WebLogic 6.0, configuration of applications is handled through XML descriptor files and the Administration Console. Converting a `weblogic.properties` file from an earlier version of WebLogic Server creates a new domain for your applications and adds `.xml` files that define how your applications are set up. Convert your `weblogic.properties` file to the appropriate `.xml` files by following these steps:

1. Start the default WebLogic Server 6.0 server and the default WebLogic 6.0 Administration Console. For information on starting WebLogic Server 6.0, see *Post-Installation Tasks* at `http://e-docs.bea.com/wls/docs60/install/instpos.html`.
2. At the home page for the WebLogic Administration Console (for example: `http://localhost:7001/console/index.jsp`) click on the Convert weblogic.properties link under the heading Getting Started.
3. Use the links to navigate the server's file system and find the root of your previous version of WebLogic Server (For example: `C:/weblogic`). When you have found the correct directory, click on the icon next to it to select it.
4. If you have additional per server `weblogic.properties` files or clustering `weblogic.properties` files select them using the provided windows. **If you have chosen the correct root directory, your global weblogic.properties file will be converted regardless of any additional properties files that you select.**
5. Select a name for your new domain. When you have chosen a new name for your domain, click Convert.

Throughout this document, the directory of the new domain you have created is referred to as *`domainName`*. The default domain that is provided with the original installation of WebLogic Server 6.0 is called `mydomain` and is located in the `wlserver6.0/config/` directory. It is necessary to set up your new domain before migrating any applications.

WebLogic Properties Mapping

The `weblogic.properties` file is no longer supported. Earlier releases of WebLogic Server used a `weblogic.properties` file to configure applications. In WebLogic Server 6.1, configuration of applications is handled through XML descriptor files and the Administration Console. If the attribute can be configured in the Administration Console, the table shows the navigation path to the attribute

The table below shows which `config.xml` attribute handles the function formerly performed by which `weblogic.properties` property. weblogic.properties Mapping Table

“weblogic.properties” file Property	**Configuration Attribute**	**Console Navigation**

`weblogic.administrator.email`	`config.xml:`
	`EmailAddress`
	`(Administrator element)`	

`weblogic.administrator.location`	`config.xml:`
	`Notes (freeform, optional)`
	`(Administrator element)999`	

`weblogic.administrator.name`	`config.xml:`
	`Name`
	`(Administrator element)`	

`weblogic.administrator.phone`	`config.xml:`
	`PhoneNumber`
	`(Administrator element)`	

`weblogic.cluster.defaultLoadAlgorithm`	`config.xml:`
	`DefaultLoadAlgorithm`
	`(Cluster element)`
		`Cluster: General: Default Load Algorithm`

`weblogic.cluster.multicastAddress`	`config.xml:`
	`MulticastAddress`
	`(Cluster element)`
		`Cluster: General: Multicast Address`

`weblogic.cluster.multicastTTL`	`config.xml:`
	`MulticastTTL`
	`(Cluster element)`
		`Cluster: Multicast: MulticastTTL`

`weblogic.cluster.name`	`config.xml`
	`ClusterAddress`
	`(Cluster element)`
		`Cluster:General: Name`

“weblogic.properties” file Property	Configuration Attribute	Console Navigation

`weblogic.httpd.authRealmName`	`config.xml: AuthRealmName`
	`(WebAppComponent element)`	

`weblogic.httpd.charsets`	`config.xml: Charsets`
	`(WebServer element)`	

`weblogic.httpd.clustering.enable`	`config.xml: ClusteringEnabled`
	`(WebServer element)`	

`weblogic.httpd.defaultServerName`	`config.xml DefaultServerName`
	`(WebServer element)`	

`weblogic.httpd.defaultWebApp`	`config.xml: DefaultWebApp`
	`(WebServer element)`	

`weblogic.httpd.enable`	`config.xml: HttpdEnabled`
	`(Server element)`	

`weblogic.httpd.enableLogFile`	`config.xml: LoggingEnabled`
	`(WebServer element)`	

`weblogic.httpd.http.keepAliveSecs`	`config.xml: KeepAliveSecs`
	`(WebServer element)`	

`weblogic.httpd.https.keepAliveSecs`	`config.xml: HttpsKeepAliveSecs`
	`(WebServer element)`	

`weblogic.httpd.indexDirectories`	`config.xml: IndexDirectoryEnabled`
	`(WebAppComponent element)`	

“weblogic.properties” file Property	Configuration Attribute	Console Navigation

`weblogic.httpd.keepAlive.enable`	`config.xml:`
	`KeepAliveEnabled`
	`(WebServer element)`	

`weblogic.httpd.logFileBufferKBytes`	`config.xml:`
	`LogFileBufferKBytes`
	`(WebServer element)`	

`weblogic.httpd.logFileFlushSecs`	`config.xml:`
	`LogFileFlushSecs`
	`(WebServer element)`	

`weblogic.httpd.logFileFormat`	`config.xml:`
	`LogFileFormat`
	`(WebServer element)`	

`weblogic.httpd.logFileName`	`config.xml:`
	`LogFileName`
	`(WebServer element)`	

`weblogic.httpd.logRotationPeriodMins`	`config.xml:`
	`LogRotationTimeBegin`
	`(WebServer element)`	

`weblogic.httpd.logRotationPeriodMins`	`config.xml:`
	`LogRotationPeriodMins`
	`(WebServer element)`	

`weblogic.httpd.logRotationType`	`config.xml:`
	`LogRotationType`
	`(WebServer element)`	

`weblogic.httpd.maxLogFileSizeKBytes`	`config.xml:`
	`MaxLogFileSizeKBytes`
	`(WebServer element)`	

`weblogic.httpd.mimeType`	`web.xml:`
	`mime-type`
	`<mime-mapping>`
	`element`	

"weblogic.properties" file Property	Configuration Attribute	Console Navigation

`weblogic.httpd.postTimeoutSecs`	`config.xml: PostTimeoutSecs`
	`(WebServer element)`	

`weblogic.httpd.servlet.extensionCaseSensitive`	`config.xml: ServletExtensionCaseSensitive`
	`(WebAppComponent element)`	

`weblogic.httpd.servlet.reloadCheckSecs`	`config.xml: ServletReloadCheckSecs`
	`(WebAppComponent element)`	

`weblogic.httpd.servlet.SingleThreadedModelPoolSize`	`config.xml: SingleThreadedServletPoolSize`
	`(WebAppComponent element)`	

`weblogic.httpd.session.cacheEntries`	`weblogic.xml: CacheSize`
	`<param-name>/<param-value> element pair`	

`weblogic.httpd.session.cookie.comment`	`weblogic.xml: CookieComment`
	`<param-name>/<param-value> element pair`	

`weblogic.httpd.session.cookie.domain`	`weblogic.xml: CookieDomain`
	`<param-name>/<param-value> element pair`	

`weblogic.httpd.session.cookie.maxAgeSecs`	`weblogic.xml: CookieMaxAgeSecs`
	`<param-name>/<param-value> element pair`	

"weblogic.properties" file Property	Configuration Attribute	Console Navigation
weblogic.httpd.session.cookie.name	weblogic.xml: CookieName <param-name>/<param-value> element pair	
weblogic.httpd.session.cookie.path	weblogic.xml: CookiePath <param-name>/<param-value> element pair	
weblogic.httpd.session.cookies.enable	weblogic.xml: CookiesEnabled <param-name>/<param-value> element pair	
weblogic.httpd.session.debug	weblogic.xml: SessionDebuggable<param-name>/<param-value> element pair	
weblogic.httpd.session.enable	weblogic.xml: SessionTrackingEnabled <param-name>/<param-value> element pair	
weblogic.httpd.session.invalidationintervalSecs	weblogic.xml: InvalidationIntervalSecs <param-name>/<param-value> element pair	
weblogic.httpd.session.jdbc.connTimeoutSecs	weblogic.xml: JDBCConnectionTimeoutSecs <param-name>/<param-value> element pair	
weblogic.httpd.session.persistentStoreDir	weblogic.xml: PersistentStoreDir <param-name>/<param-value> element pair	

"weblogic.properties" file Property	**Configuration Attribute**	**Console Navigation**

`weblogic.httpd.session.persistentStorePool`	`weblogic.xml: PersistentStorePool`
	`<param-name>/<param-value> element pair`	

`weblogic.httpd.session.persistentStoreShared`	`weblogic.xml: SessionPersistentStoreShared`
	`<param-name>/<param-value> element pair`	

`weblogic.httpd.session.persistentStoreType`	`weblogic.xml: PersistentStoreType`
	`<param-name>/<param-value> element pair`	

`weblogic.httpd.session.sessionIDLength`	`weblogic.xml: IDLength`
	`<param-name>/<param-value> element pair`	

`weblogic.httpd.session.swapintervalSecs`	`weblogic.xml: SwapIntervalSecs`
	`<param-name>/<param-value> element pair`	

`weblogic.httpd.session.timeoutSecs`	`weblogic.xml: TimeoutSecs<param-name>/<param-value> element pair`	

`weblogic.httpd.session.URLRewriting.enable`	`weblogic.xml: URLRewritingEnabled`
	`<param-name>/<param-value> element pair`	

`weblogic.httpd.tunneling.clientPingSecs`	`config.xml: TunnelingClientPingSecs`
	`(Server element)`
		`Servers: Tuning: Tunneling Client Ping`

"weblogic.properties" file Property	Configuration Attribute	Console Navigation
weblogic.httpd.tunneling.clientTimeoutSecs	config.xml: TunnelingClientTimeoutSecs (Server element)	Servers: Tuning: Tunneling Client Timeout
weblogic.httpd.tunnelingenabled	config.xml TunnelingEnabled (Server element)	Servers: Tuning: Enable Tuneling
weblogic.httpd.URLResource	config.xml: URLResource (WebServer element)	
weblogic.iiop.password	config.xml: DefaultIIOPPassword (Server element)	Servers: *servername*: Configuration: Protocols: Default IIOPPassword
weblogic.iiop.user	config.xml: DefaultIIOPUser (Server element)	Servers: *servername*: Configuration:Protocols: Default IIOPUser

"weblogic.properties" file Property	Configuration Attribute	Console Navigation
`weblogic.jdbc.connectionPool` - url=*URL for JDBC Driver* - driver=*full package name for JDBC driver* - loginDelaySecs=*seconds between connections* - initialCapacity=*initial number of JDBC connections* - maxCapacity=*maximum number of JDBC connections* - capacityIncrement=*increment interval* - allowShrinking=*true to allow shrinking* - shrinkPeriodMins=*interval before shrinking* - testTable=*name of table for autorefresh test* - refreshTestMinutes=*interval for autorefresh test* - testConnsOnReserve=*true to test connection at reserve* - testConnsOnRelease=*true to test connection at release* - props=*props for JDBC connection*	config.xml JDBCConnectionPool Element - ConnLeakProfilingEnabled - ACLName - URL - DriverName - Properties - LoginDelaySeconds - InitialCapacity - MaxCapacity - CapacityIncrement - CapacityEnabled - ShrinkPeriodMinutes - RefreshMinutes - TestTableName - TestConnectionsOnRelease - SupportsLocalTransaction - KeepLogicalConnOpenOnRelease - Password	
`weblogic.jdbc.enableLogFile`	`config.xml: JDBCLoggingEnabled (Server element)`	
`weblogic.jdbc.logFileName`	`config.xml: JDBCLogFileName (Server element)`	
`weblogic.jms.ConnectionConsumer`	config.xml `JMSConnectionConsumer element` - MessagesMaximum - Selector - Destination	Services: JMS

"weblogic.properties" file Property	**Configuration Attribute**	**Console Navigation**
`weblogic.jms.connectionFactoryArgs.<<factoryName>>` ■ ClientID ■ DeliveryMode ■ TransactionTimeout	`config.xml:` JMSConnectionFactory element ■ ClientID ■ DefaultDeliveryMode ■ TransactionTimeout ■ UserTransactionsEnabled ■ AllowCloseInOnMessage	Services: JMS: Connection Factories
`weblogic.jms.connectionFactoryName`	`config.xml: JMSConnectionFactory element` ■ JNDIName	Services: JMS: Connection Factories

`weblogic.jms.connectionPool`	`ConnectionPool`
	`(JMSJDBCStore element)`
		`Services: JMS: JMSJDBCStores`

`weblogic.jms.queue`	`config.xml: JNDIName`
	`StoreEnabled`
	`(JMSDestination element)`
		`Services: JMS:Connection Factories`

`weblogic.jms.queueSessionPool`	config.xml: ConnectionConsumer ConnectionFactory ListenerClass AcknowledgeMode SessionsMaximum Transacted (JMSSessionPool element)	Services: JMS: Servers: Session Pools
`weblogic.jms.tableNamePrefix`	`config.xml: PrefixName`	

"weblogic.properties" file Property	Configuration Attribute	Console Navigation
weblogic.jms.topic	config.xml JNDIName StoreEnabled (JMSDestination element)	Services: JMS: Servers: JMSServer-0: Destinations: JMS Queus, JMS Topics
weblogic.jms.topicSessionPool	config.xml: ConnectionConsumer ConnectionFactory ListenerClass AcknowledgeMode SessionsMaximum Transacted (JMSSessionPool element)	Services: JMS: Servers: JMSServer-0: JMS Session Pools
weblogic.jndi.transportableObjectFactories	config.xml: JNDITransportableObjectFactoryList (Server element)	
weblogic.login.readTimeoutMillisSSL	config.xml LoginTimeoutMillis (SSL element)	
weblogic.security.audit.provider	config.xml AuditProviderClassName (Security element)	Security: Audit Provider Class
weblogic.security.certificate.authority	config.xml ServerCertificateChainFileName (SSL element)	
weblogic.security.certificate.server	config.xml: ServerCertificateFileName (SSL element)	

“weblogic.properties” file Property	Configuration Attribute	Console Navigation
weblogic.security.certificateCacheSize	config.xml: CertificateCacheSize (SSL element)	
weblogic.security.clientRootCA	config.xml: TrustedCAFileName (SSL element)	
weblogic.security.disableGuest	config.xml: GuestDisabled (Security element)	Security: Guest Disabled
weblogic.security.enforceClientCert	config.xml: ClientCertificateEnforced (SSL element)	
weblogic.security.key.export.lifespan	config.xml: ExportKeyLifespan (SSL element)	
weblogic.security.key.server	config.xml: ServerKeyFileName (SSL element)	
weblogic.security.ldaprealm.authentication	config.xml: AuthProtocol (LDAPRealm element)	
weblogic.security.ldaprealm.credential	config.xml: Credential (LDAPRealm element)	
weblogic.security.ldaprealm.factory	config.xml LdapProvider (LDAPRealm element)	
weblogic.security.ldaprealm.groupDN	config.xml: GroupDN (LDAPRealm element)	

“weblogic.properties” file Property	Configuration Attribute	Console Navigation

`weblogic.security.ldaprealm.groupIsContext`	`config.xml: GroupIsContext`
	`(LDAPRealm element)`	

`weblogic.security.ldaprealm.groupNameAttribute`	`config.xml: GroupNameAttribute`
	`(LDAP Realm element)`	

`weblogic.security.ldaprealm.groupUsernameAttribute`	`config.xml: GroupUsernameAttribute`
	`(LDAPRealm element)`	

`weblogic.security.ldaprealm.principal`	`config.xml: Principal`
	`(LDAPRealm element)`	

`weblogic.security.ldaprealm.ssl`	`config.xml: SSLEnable`
	`(LDAPRealm element)`	

`weblogic.security.ldaprealm.url`	`config.xml: LDAPURL`
	`(LDAPRealm element)`	

`weblogic.security.ldaprealm.userAuthentication`	`config.xml: UserAuthentication`
	`(LDAPRealm element)`	

`weblogic.security.ldaprealm.userDN`	`config.xml: UserDN`
	`(LDAPRealm element)`	

`weblogic.security.ldaprealm.userNameAttribute`	`config.xml: UserNameAttribute`
	`(LDAPRealm element)`	

`weblogic.security.ldaprealm.userPasswordAttribute`	`config.xml: UserPasswordAttribute`
	`(LDAPRealm element)`	

“weblogic.properties” file Property	Configuration Attribute	Console Navigation
weblogic.security.net.connectionFilter	config.xml: ConnectionFilter (Security element)	
weblogic.security.ntrealm.domain	config.xml: PrimaryDomain (NTRealm element)	
weblogic.security.realm.cache.acl.enable	config.xml: ACLCacheEnable (CachingRealm element)	
weblogic.security.realm.cache.acl.size	config.xml: ACLCacheSize (CachingRealm element)	
weblogic.security.realm.cache.acl.ttl.negative	config.xml: ACLCacheTTLNegative (CachingRealm element)	
weblogic.security.realm.cache.acl.ttl.positive	config.xml: ACLCacheTTLPositive (CachingRealm element)	
weblogic.security.realm.cache.auth.enable	config.xml: AuthenticationCacheEnable (CachingRealm element)	
weblogic.security.realm.cache.auth.size	config.xml: AuthenticationCacheSize (CachingRealm element)	

“weblogic.properties” file Property	Configuration Attribute	Console Navigation

`weblogic.security.realm.cache.auth.ttl.negative`	`config.xml: AuthenticationCacheTTLNegative`
	`(CachingRealm element)`	

`weblogic.security.realm.cache.auth.ttl.positive`	`config.xml: AuthenticationCacheTTLPositive`
	`(CachingRealm element)`	

`weblogic.security.realm.cache.caseSensitive`	`config.xml: CacheCaseSensitive(CachingRealm element)`	

`weblogic.security.realm.cache.group.enable`	`config.xml: GroupCacheEnable`
	`(CachingRealm element)`	

`weblogic.security.realm.cache.group.size`	`config.xml: GroupCacheSize`
	`(CachingRealm element)`	

`weblogic.security.realm.cache.group.ttl.negative`	`config.xml: GroupCacheTTLNegative`
	`(CachingRealm element)`	

`weblogic.security.realm.cache.group.ttl.positive`	`config.xml: GroupCacheTTLPositive`
	`(CachingRealm element)`	

`weblogic.security.realm.cache.perm.enable`	`config.xml: PermissionCacheEnable`
	`(CachingRealm element)`	

“weblogic.properties” file Property	Configuration Attribute	Console Navigation
weblogic.security.realm.cache.perm.size	config.xml: PermissionCacheSize (CachingRealm element)	
weblogic.security.realm.cache.perm.ttl.negative	config.xml: PermissionCacheTTLNegative (CachingRealm element)	
weblogic.security.realm.cache.perm.ttl.positive	config.xml: PermissionCacheTTLPositive (CachingRealm element)	
weblogic.security.realm.cache.user.enable	config.xml: UserCacheEnable (CachingRealm element)	
weblogic.security.realm.cache.user.size	config.xml: UserCacheSize (CachingRealm element)	
weblogic.security.realm.cache.user.ttl.negative	config.xml: UserCacheTTLNegative (CachingRealm element)	
weblogic.security.realm.cache.user.ttl.positive	config.xml: UserCacheTTLPositive (CachingRealm element)	
weblogic.security.realm.certAuthenticator	config.xml: CertAuthenticator (SSL element)	

"weblogic.properties" file Property	Configuration Attribute	Console Navigation

`weblogic.security.SSL.ciphersuite`	`config.xml`
	`Ciphersuites`
	`(SSL element)`	

`weblogic.security.ssl.enable`	`config.xml:`
	`Enabled`
	`(SSL element)`	

`weblogic.security.SSL.hostnameVerifier`	`config.xml`
	`HostnameVerifier`
	`(SSL element)`	

`weblogic.security.SSLHandler.enable`	`config.xml:`
	`HandlerEnabled`
	`(SSL element)`	

`weblogic.security.unixrealm.authProgram`	`config.xml:`
	`AuthProgram`
	`(UnixRealm element)`	

`weblogic.system.AdministrationPort`	`config.xml`
	`AdministrationPort`
	`(Server element)`	

`weblogic.system.AdministrationPort`	`config.xml:`
	`AdministrationPort`
	`(Server element)`	

`weblogic.system.bindAddr`	`config.xml:`
	`ListenAddress`
	`(Server element)`	

`weblogic.system.defaultProtocol`	`config.xml:`
	`DefaultProtocol`
	`(Server element)`	

`weblogic.system.defaultSecureProtocol`	`config.xml:`
	`DefaultSecureProtocol`
	`(Server element)`	

`weblogic.system.enableConsole`	`config.xml:`
	`StdoutEnabled`
	`(Kernel element)`	

“weblogic.properties” file Property	Configuration Attribute	Console Navigation

`weblogic.system.enableIIOP`	`config.xml:`
	`IIOPEnabled`
	`(Server element)`	

`weblogic.system.enableReverseDNSLookups`	`config.xml:`
	`ReverseDNSAllowed`
	`(Server element)`	

`weblogic.system.enableSetGID`	`config.xml:`
	`PostBindGIDEnabled`
	`(UnixMachine element)`	

`weblogic.system.enableSetUID`	`config.xml:`
	`PostBindUIDEnabled`
	`(UnixMachine element)`	

`weblogic.system.enableTGIOP`	`config.xml`
	`TGIOPEnabled`
	`(Server element)`	

`weblogic.system.helpPageURL`	`config.xml`
	`HelpPageURL`
	`(Server element)`	

`weblogic.system.home`	`config.xml:`
	`RootDirectory`
	`(Server element)`	

`weblogic.system.ListenPort`	`config.xml`
	`ListenPort`
	`(Server element)`	

`weblogic.system.logFile`	`config.xml:`
	`FileName`
	`(Log element)`	

`weblogic.system.MagicThreadBackToSocket`	`config.xml:`
	`MagicThreadDumpBackToSocket`
	`(ServerDebug element)`	

"weblogic.properties" file Property	Configuration Attribute	Console Navigation

`weblogic.system.MagicThreadDumpFile`	`config.xml: MagicThreadDumpFile`
	`(ServerDebug element)`	

`weblogic.system.MagicThreadDumpHost`	`config.xml: MagicThreadDumpHost`
	`(ServerDebug element)`	

`weblogic.system.magicThreadDumps`	`config.xml: MagicThreadDumpEnabled`
	`(ServerDebug element)`	

`weblogic.system.maxLogFileSize`	`config.xml: FileMinxSize`
	`(Log element)`	

`weblogic.system.nativeIO.enable`	`config.xml: NativeIOEnabled`
	`(Server element)`	

`weblogic.system.nonPrivGroup`	`config.xml PostBindGID`
	`(UnixMachine element)`	

`weblogic.system.nonPrivUser`	`config.xml PostBindUID`
	`(UnixMachine element)`	

`weblogic.system.percentSocketReaders`	`config.xml: ThreadPoolPercentSocketReaders`
	`(Kernel element)`	

`weblogic.system.readTimeoutMillis`	`config.xml LoginTimeoutMillis`
	`(Server element)`	

`weblogic.system.readTimeoutMillis`	`config.xml: LoginTimeoutMillis`
	`(Server element)`	

"weblogic.properties" file Property	**Configuration Attribute**	**Console Navigation**

`weblogic.system.SSL.useJava`	`config.xml:`
	`UseJava`
	`(SSL element)`	

`weblogic.system.SSLListenPort`	`config.xml:`
	`ListenPort`
	`(SSL element)`	

`weblogic.system.startupFailureIsFatal`	`config.xml`
	`FailureIsFatal`
	`(StartupClass element)`	

`weblogic.system.user`	`config.xml:`
	`SystemUser`
	`(Security element)`
		`Security: Users`

`weblogic.system.weight`	`config.xml`
	`ClusterWeight`
	`(Server element)`
		`Server:` *`servername`*`: Cluster: Cluster Weight`

`weblogic.workspace.showUserKeysOnly`	`config.xml:`
	`WorkspaceShowUserKeysOnly`
	`(Server element)`	

`weblogic.zac.enable`	`config.xml:`
	`ZACEnabled`
	`(Server element)`	

`weblogic.zac.publishRootProp`	`config.xml:`
	`ZACPublishRoot`
	`(Server element)`	

Setting Up Your New Domain

A domain is an interrelated collection of WebLogic Server resources and configuration files that occupy a single specific namespace. Only one domain can be active at a time. For more information on domains, see *Domains, the Administration Server and Managed Servers* at `http://e-docs.bea.com/wls/docs60/adminguide/overview.html`. To use your new domain you must start your server with the correct domain and copy the WebLogic Server Administration Console into the appropriate location. The WebLogic Server Administration Console is a sophisticated Web Application and an essential tool in managing your server.

Restarting the Server

In order to use your new domain you must restart the server, specifying before startup your new domain name. To do so, startup scripts have been provided. These scripts, which are generated when a weblogic.property file is converted, are named `start`*`domainName`*`.cmd` (for Windows users) and `start`*`domainName`*`.sh` (for UNIX users) and exist under the `wlserver6.0/config/`*`domainName`* directory. These scripts will start the server in the new domain.

These startup scripts make it easier to start instances of WebLogic Server. You may need to modify them to fit your environment and applications. See *Starting and Stopping the WebLogic Server* at `http://e-docs.bea.com/wls/docs60/adminguide/startstop.html` for more information on scripts and starting servers. More information is also available in the Administrator's Guide, under *Migrating from Earlier Versions of WebLogic Server* at `http://e-docs.bea.com/wls/docs60/adminguide/startstop.html#MigratingStartScripts`.

Copying the Administration Console to Your New Domain

Before restarting your server, it is also necessary to copy the Adminstration Console Web Application into the newly created domain. The Administration Console is a Web Application with its own `.war` file and can be treated as any other Web Application.

To copy it into your new domain, follow these steps:

1. After converting your properties file, shut down the server by closing the window where the server is running.

2. Navigate to `wlserver6.0/config/mydomain/applications`.

3. Copy the `console.war` file.

4. Proceed to `wlserver6.0/config/`*`domainName`*`/applications`

5. Place the copy of `console.war` inside the new `wlserver6.0/config/`*`domainName`*`/applications` directory.

6. Restart your server and refresh the browser window that contains your console.

At the top middle of the main console window as well as the top of the left hand pane, you should see your new domain displayed as the active domain. Now you can begin to migrate your applications.

Migrating Applications

Applications on a J2EE-compliant server such as WebLogic Server 6.0 are created and deployed as one of the following four types: Web Applications, Enterprise JavaBeans, Enterprise Archives, and Client Applications. To migrate your existing components to WebLogic Server 6.0, create the appropriate J2EE deployment units. Web Applications are usually a collection of servlets, JSPs, and HTML files. Enterprise JavaBeans are server-side Java components written according to the EJB specification. Enterprise Archives may contain a combination of EJB components and Web Application components. Client Applications are Java classes that connect to

WebLogic Server using Remote Method Invocation (RMI). Each of the aforementioned J2EE deployment units are discussed in greater detail within the following sections.

Web Applications

A Web Application is created by correctly packaging into a single unit the servlets, JSPs and HTML pages that make up an application.

In order to deploy an application as a Web Application it is necessary to create certain `.xml` files that contain configuration information on that application. If you have converted your properties file, these `.xml` files (`web.xml and weblogic.xml`) have been created for you and placed inside a `wlserver6.0/config/`*`domainName`*`/applications/DefaultWebApp_myserver /WEB-INF` directory. The process of converting your weblogic.properties file also creates the `config.xml` file located in `wlserver6.0/config/`*`domainName`*`/`. This file contains configuration information specific to your server.

In order to migrate applications into a Web Application deployed on WebLogic Server 6.0, the files of your application must be placed within a directory structure that follows a specific pattern. For development, these files can be left in an exploded directory format. However, for production situations, it is highly recommended that you bundle your applications into the appropriate `.war` file as a singular Web Application. For more information on Web Applications see *Understanding WebLogic Server Applications* at `http://e-docs.bea.com/wls/docs60/programming/concepts.html` and *Deploying and Configuring Web Applications* at `http://e-docs.bea.com/wls/docs60/adminguide/config_web_app.html`.

Directory Structure

Web Applications are organized within a specified directory structure so that they can be archived and deployed on WebLogic Server. All servlets, classes, static files, and other resources belonging to a Web Application are organized under a directory hierarchy. The root of this hierarchy defines the document root of your Web Application. All files under this root directory can be served to the client, except for files under the special directories `WEB-INF` and `META-INF` located in the root directory. The root directory should be named with the name of your Web Application and placed inside the `wlserver6.0/config/`*`domainName`*`/applications` directory.

The following diagram illustrates the directory structure of any Web Application.

```
WebApplicationRoot/(Publically available files such as
                    |  .jsp, .html, .jpg, .gif)
                    |
                    +WEB-INF/-+
                              |
                              + classes/(directory containing
                              |         Java classes including
                              |         servlets used by the
                              |         Web Application)
                              |
                              + lib/(directory containing
                              |      jar files used by the
                              |      Web Application)
                              |
                              + web.xml
                              |
                              + weblogic.xml
```

If you have already converted your properties file, the appropriate `web.xml` and `weblogic.xml` files have already been created for you under the directory `wlserver6.0/config/`*`domainName`*`/applications/DefaultWebApp_myserver/WEB-INF`. Follow the above directory structure and place the `.xml` files in the `wlserver6.0/config/`*`domainName`*`/applications/`*`webAppName`*`/WEB-INF` directory that you create. All applications must be placed inside a `wlserver6.0/config/`*`domainName`*`/applications` directory in order to be deployed. For more information see *Developing WebLogic Server Applications* at `http://e-docs.bea.com/wls/docs60/programming/index.html`.

XML Deployment Descriptors

The Web Application Deployment Descriptor (`web.xml`) is a standard J2EE descriptor used to register your servlets, define servlet initialization parameters, register JSP tag libraries, define security constraints, and other Web Application parameters. For detailed instructions on creating the deployment descriptor, see *Writing the Web Application Deployment Descriptor* at `http://e-docs.bea.com/wls/docs60/programming/webappdeployment.html#web-xml`.

There is also a WebLogic-specific Deployment Descriptor (`weblogic.xml`). In this file you define JSP properties, JNDI mappings, security role mappings, and HTTP session parameters. The WebLogic-specific deployment descriptor also defines how named resources in the web.xml file are mapped to resources residing elsewhere in WebLogic Server. For detailed instructions on creating the WebLogic-specific deployment descriptor, see *Writing the WebLogic-specific Deployment Descriptor* at http://e-docs.bea.com/wls/docs60/programming/webappdeployment.html#weblogic-xml. This file may not be required if you have no need for any of the above mentioned properties, mappings, or parameters.

If you have converted your weblogic.properties file, a web.xml and weblogic.xml file have already been created for you. Use the web.xml and weblogic.xml files, in conjunction with the console, to configure your applications. The .xml files can be viewed through any text editor. To edit them, simply make your changes and save the file as web.xml or weblogic.xml with the appropriate path as specified by the prescribed directory structure. See *Configuring your Web Applications* at http://e-docs.bea.com/wls/docs60/adminguide/config_web_app.html for more information. If you do not want to deploy your applications together as a single Web Application, you need to split up the .xml files that have been created for you, creating the appropriate .xml files specific to each Web Application.

WAR Files

A .war file is a Web Application archive. If you have correctly followed the prescribed directory structure of a Web Application and created the appropriate web.xml and weblogic.xml files, it is strongly recommended that in production environments your applications are bundled together in a Web Application deployed as a .war file. Once you have done so, it is important to remove the previously existing directory structure so that WebLogic Server only has one instance of each application.

Use the following command line from the root directory containing your Web Application to create a .war file, replacing '*webAppName*' with the specific name you have chosen for your Web Application:

```
jar cvf webAppName.war *
```

You now have created a .war file that contains all the files and configuration information for your Web Application.

Deploying Web Applications

To deploy your bundled Web Applications properly, place the appropriate .war file in the c:/wlserver6.0/config/*domainName*/applications directory. You can also install the application through the Administration Console. To do so, go to the console home and choose Install Applications under the Getting Started menu. Select the correct `.war` file and it will be installed automatically. Note that it is necessary to have your applications reside in a `c:/wlserver6.0/config/`*`domainName`*`/applications` directory in order for them to work.

Web Applications should be deployed automatically after they have been installed. Check to see that they are deployed under the Deployments node in the left hand pane of the Administration Console.

You can configure certain deployment attributes for your Web Application using the Administration Console. Select the Web Applications node under the Deployments heading. Select your Web Application. Click on the appropriate tab to configure. For more information on setting attributes in the console, see the Web Application section of the *Console Help* at `http://e-docs.bea.com/wls/docs60/ConsoleHelp/webappcomponent.html`.

JavaServer Pages (JSP) and Servlets

This section contains information specific to JSPs and servlets that may be pertinent to your applications.

- Some changes will be necessary in code (both Java and HTML) where the code refers to URLs, which may be different when servlets and JSPs are deployed in a Web Application that is not the default Web Application. See the *Servlet Documentation* at `http://e-docs.bea.com/wls/docs60/servlet/admin.html#servlet-urls` for more information. If relative URLs are used and all components are contained in the same Web Application, these changes are not necessary.

- Only serializable objects may be stored in a session if your application is intended to be distributable.

- You must convert your weblogic.properties to XML attributes in `web.xml` and/or `weblogic.xml`. For additional information on this process, see the conversion section of the *Console Help* at `http://e-docs.bea.com/wls/docs60/ConsoleHelp/conversion.html`.

- ACLs are now defined as security constraints in `web.xml`.

- Server-Side-Includes are not supported. You must use JSP to achieve this functionality.

- This version of WebLogic Server is now fully compliant with the Servlet 2.2 specification.

Example Steps to Migrate a Simple Servlet

The following procedure migrates the simple Hello World Servlet that was provided with WebLogic 5.1 Server to WebLogic Server 6.0.

1. Create the correct directory structure, as described in *Programming HTTP Servlets* at `http://e-docs.bea.com/wls/docs60/servlet/admin.html`. This involves creating a root application directory, such as `C:/hello`, as well as a `C:/hello/WEB-INF` directory and a `C:/hello/WEB-INF/classes` directory. Place the `HelloWorld.Servlet.java` file inside the `C:/hello/WEB-INF/classes` directory.

2. Create a `web.xml` file for this servlet. If you have converted your weblogic.properties file, a `web.xml` file has already been created for you. If you registered HelloWorldServlet in your `weblogic.properties` file, it will be properly configured in your new `web.xml` file. An `.xml` file can be created with any text editor. The following is an example of a basic `web.xml` file that could be used with the HelloWorldServlet.

   ```
   <!DOCTYPE web-app (View Source for full doctype...)>
   - <web-app>
   - <servlet>
   <servlet-name>HelloWorldServlet</servlet-name>
   <servlet-class>examples.servlets.HelloWorldServlet</servlet-cla
   ss>
   </servlet>
   - <servlet-mapping>
   <servlet-name>HelloWorldServlet</servlet-name>
   <url-pattern>/hello/*</url-pattern>
   </servlet-mapping>
   </web-app>
   ```

   For more information on `web.xml` files, see *Writing the Web Application Deployment Descriptor* at `http://e-docs.bea.com/wls/docs60/programming/webappdeployment.html#web-xml`. A `weblogic.xml` file is not necessary with such a simple, stand-alone servlet as HelloWorld.

   For more information on `weblogic.xml` files, see *Writing the WebLogic-Specific Deployment Descriptor* at `http://e-docs.bea.com/wls/docs60/programming/webappdeployment.html#weblogic-xml`. If you registered the HelloWorldServlet in Weblogic 5.1

and have converted your `weblogic.properties` file into `.xml`, simply move the `web.xml` file from `wlserver6.0/config/`*`domainName`*`/applications/DefaultWebApp_myserver/WEB-INF` to `C:/hello/WEB-INF/`. If you did not convert your `weblogic.properties` file, or did not register the HelloWorldServlet in your earlier version of WebLogic, create a `web.xml` file like the above example and place it inside the `C:/hello/WEB-INF/` directory.

3. Set up your development environment (see *Establishing a Development Environment* at `http://e-docs.bea.com/wls/docs60/programming/environment.html#devenv` for more information) and compile the HelloWorldServlet with a command like the following:

   ```
   C:\hello\WEB-INF\classes>javac -d  . HelloWorldServlet.java
   ```

   This should compile the file and create the correct package structure.

4. The servlet can now be bundled into an archive `.war` file with the following command:

   ```
   jar cvf hello.war *
   ```

   This command will create a `hello.war` file and place it inside the `C:/hello` directory.

5. To install this Web Application, start your server and open the Administration Console. Under the Getting Started menu, choose Install Applications. Browse to the newly created `.war` file and click Upload. The servlet should now be deployed and appear under the Web Applications node under Deployments, in the left-hand pane of the console.

6. To call the servlet, type the following in your browser URL window: `http://localhost:7001/hello/hello`. In this case `/hello/` is the context path of the servlet. This is determined by the naming of the `.war` file, in this case `hello.war`. The second `/hello` was mapped in the the servlet mapping tags inside the `web.xml` file.

Enterprise JavaBeans Applications

To migrate a 1.1 EJB from your WebLogic Server 5.1 installation to WebLogic Server 6.0, first open the Administration Console. From the home page click on Install Applications under the Getting Started heading. Locate the `.jar` file you wish to install by clicking the Browse button. Once you have located it, click Open and then Upload. Your bean should now be automatically deployed.

To check your output, run a `setEnv` script on a client window and set your development environment (for more information see *Establishing a Development Environment* at `http://e-docs.bea.com/wls/docs60/programming/environment.html#devenv`). Then compile all the needed client classes. For example, using the Stateless Session Bean sample that was provided with WebLogic Server 5.1, you would use the following command:

```
javac -d %CLIENTCLASSES% Trader.java TraderHome.java
TradeResult.java Client.java
```

Then reference the EJB interfaces in your client's classpath:

```
java -classpath %CLIENTCLASSES%;%CLASSPATH%
examples.ejb.basic.statelessSession.Client
```

To run the client, enter this command:

```
java examples.ejb.basic.statelessSession.Client
```

In order to convert an EJB 1.1 bean to a EJB 2.0 bean, WebLogic provides a DDConverter. It is recommended that you develop EJB 2.0 beans in conjunction with WebLogic server 6.0. However, for 1.1 beans already used in production, it is not necessary to convert them to 2.0 beans. EJB 1.1 beans are deployable with WebLogic Server 6.0. If you do wish to convert 1.1 beans to 2.0 beans, see the *DDConverter* documentation at `http://e-docs.bea.com/wls/docs60/ejb/EJB_utilities.html` for more information.

The basic steps required to convert a simple CMP 1.1 bean to a 2.0 bean are as follows:

1. Make the bean class abstract. EJB 1.1 beans declare CMP fields in the bean. CMP 2.0 beans use abstract getXXX and setXXX methods for each field. For instance, 1.1 Beans will use `public String name`. 2.0 Beans should use `public`

`abstract String getName()` and `public abstract void setName(String n)`. With this modification, the bean class should now read the container-managed fields with the `getName` method and update them with the `setName` method.

2. Any CMP 1.1 finder which used `java.util.Enumeration` should now use `java.util.Collection.` Change your code to reflect this. CMP 2.0 finders cannot return `java.util.Enumeration`.

The following list includes further information that may be pertinent to users migrating their Enterprise JavaBeans to use WebLogic Server 6.0.

- WebLogic Server Version 6.0 supports the Enterprise JavaBeans 1.1 and 2.0 specifications.
- The XML parser is stricter with XML deployment descriptors in WebLogic 6.0. Some errors allowed in earlier versions are no longer permitted. This is described in *Programming WebLogic EJB* at `http://e-docs.bea.com/wls/docs60/ejb/EJB_whatsnew.html`.
- EJB 1.1 beans are deployable in WebLogic Server 6.0. However, if you are developing new beans, it is recommended that you use EJB 2.0. EJB 1.1 beans can be converted to 2.0 using the DDConverter utility. For more information see the *DDConverter* documentation at `http://e-docs.bea.com/wls/docs60/ejb/EJB_utilities.html`.
- EJB 1.0 deployment descriptors can be upgraded to EJB 2.0 using the DDConverter utility also, but first must be upgraded to 1.1 beans. WebLogic Server 5.1 deployment descriptors may be upgraded to 6.0 to take advantage of WebLogic Server 6.0’s new features. Details on the DDConverter utility are provided in *Programming WebLogic EJB* at `http://e-docs.bea.com/wls/docs60/ejb/EJB_utilities.html`.
- The finder expressions feature of EJB 1.1 is no longer supported. This is the only non-supported feature of EJB 1.1.
- The `weblogic.deploy` utility is no longer supported. Deploying beans is described in *Programming WebLogic EJB* at `http://e-docs.bea.com/wls/docs60/ejb/EJB_deployover.html`.
- If `ejbc` has not been run on an EJB, WebLogic Server 6.0 will run `ejbc` automatically when the bean is deployed. You do not need to compile beans with `ejbc` before deploying. If you wish to run `ejbc` during startup, you may do so. See details in *Programming WebLogic EJB* at `http://e-docs.bea.com/wls/docs60/ejb/EJB_deployover.html`.

For more information on Enterprise JavaBeans, see *Enterprise JavaBeans Components* and *Programming WebLogic EJB* at `http://e-docs.bea.com/wls/docs60/ejb/index.html`.

Enterprise Applications

An Enterprise Application consists of assembled components, and is a `.jar` file with an `.ear` extension. An `.ear` file contains all of the `.jar` and `.war` component archive files for an application and an XML descriptor that describes the bundled components. The `META-INF/application.xml` deployment descriptor contains an entry for each Web and EJB module, and additional entries to describe security roles and application resources such as databases.

```
EnterpriseApplicationStagingDirectory/
                                     |
                                     + .jar files
                                     |
                                     + .war files
                                     |
                                     +META-INF/-+
                                                |
                                                + application.xml
```

When you have assembled all of the Web Archive and EJB Archives for your application and placed them under your staging directory, you can bundle them together in an Enterprise Archive (`.ear`) file so that you can deploy all of the dependent components together. Copy the `.war` and EJB `.jar` files into the staging directory and then create a `META-INF/application.xml` deployment descriptor for the application. Follow the directory structure depicted above. The `application.xml` file contains a descriptor for each component in the application, using a DTD supplied by Sun Microsystems. For more information on the application.xml file, see *Client Application Deployment Descriptor Elements* at `http://e-docs.bea.com/wls/docs60/programming/app_xml.html`. Note that if you are using JSPs and want them to compile at run time you must have the home and remote interfaces of the bean included in the classes directory of your `.war` file.

Create the Enterprise Archive by executing a jar command like the following in the staging directory:

```
jar cvf myApp.ear *
```

The .ear file can be installed through the console by clicking on the Install Applications link under the Getting Started heading in the home page of the console. The .ear file needs to be in the c:/wlserver6.0/config/domainName/applications directory. For more information on Enterprise Applications, see *Staging Enterprise Applications* at http://e-docs.bea.com/wls/docs60/programming/packaging.html.

Client Applications

WebLogic Server also supports J2EE client applications, packaged in a jar file with a standard XML deployment descriptor. Client applications (meaning: when the client is not a Web browser) are Java classes that connect to WebLogic Server using Remote Method Invocation (RMI). A Java client can access Enterprise JavaBeans, JDBC connections, JMS messaging, and other services using RMI. Client applications range from simple command line utilities that use standard I/O to highly interactive GUI applications built using the Java Swing/AWT classes.

To execute a WebLogic Server Java client, the client computer needs the weblogic_sp.jar file, the weblogic.jar file, the remote interfaces for any RMI classes and Enterprise Beans that are on WebLogic Server, as well as the client application classes. To simplify maintenance and deployment, it is a good idea to package a client-side application in a jar file that can be added to the client's classpath along with the weblogic.jar and weblogic_sp.jar files. The weblogic.ClientDeployer command line utility is executed on the client computer to run a client application packaged to this specification. For more information about J2EE client applications, see *Staging and Deploying Client Applications* at http://e-docs.bea.com/wls/docs60/programming/packaging.html.

Standalone HTML and JSP PAGES

In the original domain provided with WebLogic Server 6.0, as well as in any domains that have been created using the weblogic.properties file converter, there is a wlserver6.0/config/*domainName*/applications/DefaultWebApp_myserver directory that has been created. This directory can be used to contain files made available by your web server. HTML and JSP files may be placed here and be made available, separate from any applications you install. If necessary, you can create subdirectories within the `DefaultWebApp_myserver` directory to handle relative links, such as image files.

Applications and Managed Servers

By default, applications are deployed on the administration server's `config.xml` block and JVM. However, in most cases, this is not good practice. The administration server should just be used for administrative purposes. Users should define new managed servers and associate the applications with those servers. This is done via the Administration Console. For more information, see *Configuring WebLogic Servers and Clusters* at `http://e-docs.bea.com/wls/docs60/adminguide/config.html` and *Domains, the Administration Server, and Managed Servers* at `http://e-docs.bea.com/wls/docs60/adminguide/overview.html`.

Other J2EE Services

What follows is a list of different components that can be used by applications. Deprecated features, upgrades, and the important changes that have been made in WebLogic Server 6.0 are noted.

Java Transaction API (JTA)

The following changes in JTA have taken place:

- WebLogic Server 6.0 supports the JTA 1.0.1 specification. Updated JTA documentation is provided in *Programming WebLogic JTA* at `http://e-docs.bea.com/wls/docs60/jta/index.html`.
- Based on the inclusion of support for JTA, the JTS JDBC driver (with properties in `weblogic.jts.*` and URL `jdbc:weblogic:jts`:..) has been replaced by a JTA JDBC/XA driver. Existing properties are available for backward

compatibility, but you should change the class name and properties to reflect the JTS to JTA name change.

Java Message Service (JMS)

For specific details on migrating JMS, see *Migrating WebLogic JMS Applications* in *Programming WebLogic JMS* at `http://e-docs.bea.com/wls/docs60/jms/index.html`.

WebLogic Server 6.0 supports the *JavaSoft JMS specification version 1.0.2* at `http://www.javasoft.com/products/jms/docs.html`. In order to use your existing JMS applications, perform the following steps:

1. Upgrade the WebLogic Server environment, as described in the *Installation Guide* at `http://e-docs.bea.com/wls/docs60/install/index.html`.

2. Migrate configuration information using the *Configuration Conversion Facility* at `http://e-docs.bea.com/wls/docs60/ConsoleHelp/conversion.html`. Once your configuration information is converted, the JMS administrator needs to review the resulting configuration to ensure that the conversion meets the needs of the application, and adjust the values, if necessary.

3. Prepare for automatic migration of existing JDBC database stores, as described in *Migrating WebLogic JMS Applications* in *Programming WebLogic JMS* at `http://e-docs.bea.com/wls/docs60/jms/migrat.html`.

   In WebLogic Server 5.1, the WebLogic JMS data for JMS messages and durable subscribers was kept in five database tables that were accessible via JDBC. In WebLogic Server 6.0, JMS queues are defined during configuration, and no longer saved within database tables. Message data and durable subscriptions are stored either in two JDBC tables or via a directory within the file system. The current JDBC database store content format is not compatible with the format in WebLogic Server 6.0 Beta.

4. Update any existing code, as required, to reflect the feature functionality changes described in *Migrating WebLogic JMS Applications* in *Programming WebLogic JMS* at `http://e-docs.bea.com/wls/docs60/jms/index.html`. For example, the `createQueue()` and `createTopic()` methods do not create destinations dynamically, only references to destinations that already exist given the vendor-specific destination name.

5. When you initially boot up the WebLogic Server, the existing JDBC database stores will be migrated automatically. If the automatic migration fails for any reason, the automatic migration will be re-attempted the next time the WebLogic Server boots.

For more information on migrating your WebLogic JMS applications, see *Migrating WebLogic JMS Applications* in *Programming WebLogic JMS* at `http://e-docs.bea.com/wls/docs60/jms/index.html`. Note that WebLogic Events are being deprecated and being replaced by JMS messages with NO_ACKNOWLEDGE or MULTICAST_NO_ACKNOWLEDGE delivery modes. Each of these delivery modes is described in *Programming WebLogic JMS* at `http://e-docs.bea.com/wls/docs60/jms/fund.html`.

Java Database Connectivity (JDBC)

The following changes have been made in JDBC:

- The WebLogic T3 API is deprecated in WebLogic Server 6.0; use the RMI JDBC driver in its place. This also applies to users migrating from WebLogic Server 4.5.x.
- The `weblogic.jdbc20.*` packages are being replaced with `weblogic.jdbc.*` packages. All WebLogic JDBC drivers are now compliant with JDBC 2.0.
- If you have a current connection and are using a `preparedStatement`, and the stored procedure gets dropped in the DBMS, use a new name to create the stored procedure. If you recreate the stored procedure with the same name, the `preparedStatement` will not know how to access the newly created stored procedure—it is essentially a different object with the same name.

RMI

The following tips are for users migrating to WebLogic 6.0 who used RMI in their previous version of WebLogic Server:

- Re-run the WebLogic RMI compiler, `weblogic.rmic`, on any existing code to regenerate the wrapper classes so they are compatible with WebLogic Server 6.0.

- Use `java.rmi.Remote` to tag interfaces as remote. Do not use `weblogic.rmi.Remote`.
- Use `java.rmi.*Exception`. Do not use the `weblogic.rmi` counterparts.
- Use JNDI instead of `*.rmi.Naming`.
- You may use `weblogic.rmic` to generate stubs and skeletons.
- Use `weblogic.rmi.server.UnicastRemoteObject.exportObject()` to get a stub instance.
- The RMI examples have not currently been updated to use `java.rmi.*` and JNDI. The examples will be revised to reflect `java.rmi.*` and JNDI in a future release.

Internationalization (I18N)

Several internationalization and localization changes have been made in this version:

- Changes have been made to the log file format that affect the way that messages are localized. The new message format also has additional things in the first line that were not present in earlier versions of WebLogic Server. New entries are: *begin marker*, *machine name*, *server name*, *thread id*, *user id*, *tran id*, and *message id*.
- There is a new internationalized logging API through which users can log messages in the server and clients.
- Clients log to their own logfiles which are in the same format as the server logfiles, with the exception of the *servername* and *threadid* fields.
- `LogServicesDef` is deprecated. Instead, use the internationalized API or `weblogic.logging.NonCatalogLogger` (when internationalization is not required).

For details on internationalization in this version, see *BEA WebLogic Server Internationalization Guide* at `http://e-docs.bea.com/wls/docs60/i18n/index.html`.

Security

Existing WebLogic Server customers may have private keys that are unprotected in Definite Encoding Rules (DER) format. For those private keys to work with this version of WebLogic Server, you need to convert the private keys to a PKCS#5/PKCS#8 Privacy Enhanced Mail (PEM) format. Use the `protectkey` conversion tool to convert your private keys. The `protectkey` tool takes a private key in DER format and a password and converts them to a PKCS#5/PKCS#8 PEM format. For more information about the `protectkey` tool, see *Managing Security in the Administration Guide* at `http://e-docs.bea.com/wls/docs60/adminguide/cnfgsec.html`.

It is highly recommended that users read our documentation on *Programming WebLogic Security* at `http://e-docs.bea.com/wls/docs60/security/index.html` for the most detailed information on using security with WebLogic Server 6.0. WebLogic users migrating from earlier releases should read the following list of tips and issues:

- A large portion of implementing security in the WebLogic Server environment is configuration. To migrate a current security configuration, convert the `weblogic.properties` file to XML attributes in `config.xml` using the Administration Console. Details on converting `weblogic.properties` files are described in the *Console Help* documentation at `http://e-docs.bea.com/wls/docs60/ConsoleHelp/conversion.html`. For a complete mapping of the security properties to XML attributes, see "Managing Security" in the *Administration Guide* at `http://e-docs.bea.com/wls/docs60/adminguide/cnfgsec.html`.

- The name of the default security realm changed from `WLPropertyRealm` to the File realm. Realm attributes are now stored in the `fileRealm.properties` file instead of the `weblogic.properties` file.

- At some point, redefine your realm and authorization attributes through the Administration Console. The resulting information is stored in the `fileRealm.properties` file. For information about creating a custom realm in WebLogic Server 6.x, see Writing a Custom Security Realm.

- It is highly recommended that at the end of installation, you check all security settings to make sure they are the appropriate ones for their environment.

- ACLs can no longer be used to specify security for stand-alone servlets because stand-alone servlets have been completely replaced by web applications. Web applications can only be secured using the web app's deployment descriptors as defined in the Servlet 2.2 specification.

- The Delegating security realm is no longer supported in WebLogic Server 6.0. If you are using the Delegating security realm, you must use another type of security realm to store Users, Groups, and ACLs. For details, see "http://e-docs.bea.com/wls/docs60/adminguide/cnfgsec.html".

WAP Applications

To run a WAP application on WebLogic Server 6.0, you must now specify the MIME types associated with WAP in the `web.xml` file of the web application. In WebLogic Server 5.1, the default mime-type can be set using `weblogic.httpd.defaultMimeType` in `weblogic.properties` where its default value is "text/plain". WebLogic Server 6.0, WebLogic Server 6.1, and WebLogic Server 7.0 do not have a default `mime-type`. You must explicitly specify `mime-type` for each extension in the `web.xml` file. For information on required MIME types see *Using Wireless Application Protocol (WAP) with WebLogic Server* at `http://e-docs.bea.com/wls/docs60/wap/index.html`. For information on creating and editing a `web.xml` file, see *Writing Web Application Deployment Descriptors* at `http://e-docs.bea.com/wls/docs60/programming/webappdeployment.html`.

An example configuration of the `mime-types` in the `web.xml` file:

```
<web-app>
  <mime-mapping>
    <extension>tiff</extension>
    <mime-type>image/tiff</extension>
  </mime-mapping>
  <mime-mapping>
    <extension>tif</extension>
    <mime-type>image/tiff</extension>
  </mime-mapping>
</web-app>
```

Web Services

The following tips are for users migrating to WebLogic 6.0 who used Web Services in their previous version of WebLogic:

- All of the Web services in WebLogic Server now use Web Applications as the mechanism for defining how WebLogic Server serves up JSPs, servlets, and static HTML pages. In a new installation of WebLogic Server, the server will configure a default Web Application. Customers upgrading to WebLogic 6.0 should not need to perform any registrations because this default Web Application closely approximates the document root, the JSPServlet, and servlet registrations performed using the `weblogic.properties` file contained in earlier versions.
- Convert your existing `weblogic.properties` file to `.xml` files using the Administration Console. See *Console Help* at `http://e-docs.bea.com/wls/docs60/ConsoleHelp/conversion.html` for more details.
- SSI is no longer supported.
- URL ACLs are deprecated. Use Servlet 2.2 features instead.
- Some information has moved from `web.xml` to `weblogic.xml`. This reorganization allows a third-party Web application based strictly on Servlet 2.2 to be deployed without modifications to its J2EE standard deployment descriptor (`web.xml`). WebLogic Server 5.1 style settings made in the `web.xml` file using `<context-param>` elements are supported for backward compatibility, but you should adopt the new way of deploying. The following sets of parameters previously defined in `web.xml` are now defined in `weblogic.xml`:

  JSP Parameters (*`keepgenerated, verbose, packagePrefix, pageCheckSeconds, encoding`*)

  HTTP sessionParameters (*`CookieDomain, CookieComment, CookieMaxAgeSecs, CookieName, CookiePath, CookiesEnabled, InvalidationIntervalSecs, PersistentStoreDir, PersistentStorePool, PersistentStoreType, SwapIntervalSecs, IDLength, CacheSize, TimeoutSecs, JDBConnectionTimeoutSecs, URLRewritingEnabled`*)

- For more information, see *Writing Web Application Deployment Descriptors* at `http://e-docs.bea.com/wls/docs60/programming/webappdeployment.html`.

XML

The XML parser has been updated and is now based on the Apache Xerces parser. The parser implements version 2 of the SAX and DOM interfaces. Users who made use of older parsers that were shipped in the `weblogicaux.jar` (such as Sun's Project X parser) may receive deprecation messages.

Deprecated APIs and Features

The following APIs and features are deprecated in anticipation of future removal from the product:

- `weblogic.security.acl`
- `-Dweblogic.management.host`
- WebLogic Events

  WebLogic Events are deprecated and should be replaced by JMS messages with NO_ACKNOWLEDGE or MULTICAST_NO_ACKNOWLEDGE delivery modes. See *Programming WebLogic JMS* at `http://e-docs.bea.com/wls/docs60/jms/fund.html#nontransacted` for more information.

- WebLogic HTMLKona
- T3Client
- T3 Driver
- Delegating security realm

  The Delegating security realm is no longer supported in WebLogic Server 6.0. If you are using the Delegating security realm, you will have to use another type of security realm to store Users, Groups, and ACLs. For details, see "http://e-docs.bea.com/wls/docs60/adminguide/cnfgsec.html".

Removed APIs and Features

The following APIs and features have been removed:

- The old administrative console GUI.

  Use the new Administration Console.
- The Deployer Tool.
- WebLogic Beans
- WebLogic jHTML
- weblogic.logging.LogOutputStream; use weblogic.logging.NonCatalogLogger
- WebLogic Remote
- WorkSpaces
- WebLogic Tour
- Jview support
- SSI
- Weblogic Bean Bar
- RemoteT3
- Jview support
- Weblogic COM

  This feature relied on the Microsoft JVM (Jview) which is no longer supported.
- WebLogic Tour

CHAPTER

4 Notes and Problems

The following sections describe general issues and levels of support for this release:

- SNMP Support
- Platform Support
- Standards Support
- HTML Pages and Netscape
- Clusters, DNS, and Multihoming on NT
- JDBC Drivers for Oracle
- Known Problems

SNMP Support

The WebLogic SNMP agent is not included in the first general availability release of WebLogic Server 6.0. The agent is being enhanced and is scheduled to be delivered in a future release.

Platform Support

See our *Platforms Support Page* at `http://www.weblogic.com/platforms/index.html` for the most accurate and current information regarding platform support.

Standards Support

Standard	Version
HTTP	1.1
J2EE EJB	2.0
J2EE JDBC	2.0
J2EE JNDI	1.2
J2EE JSP	1.1
J2EE JTA	1.0.1
J2EE JMS	1.0.2
J2EE RMI	1.0
RMI/IIOP	1.0
J2EE Servlet	2.2
LDAP	2
SSL	3
X.509	3

HTML Pages and Netscape

If you already have a Netscape browser running, some HTML pages will not display. For example, if you select the *About WebLogic Server* page from the Start menu, and you have Netscape running and do not have Microsoft Internet Explorer, the screen may flash but the page will not display.

Clusters, DNS, and Multihoming on NT

Be aware of naming issues when using multihoming features on Windows NT in a cluster. For example, there may be naming conflicts if a cluster is running on a multihomed Windows NT machine and one of the servers in the cluster is bound to the same DNS name as the machine name. Attempts to contact that server using the DNS name in a URL may result in Windows NT converting that DNS name to any of the IP addresses of the multihomed Windows NT machine. In this case, the request may go to the wrong address.

JDBC Drivers for Oracle

Performance and functionality issues have been encountered using WebLogic jDriver for Oracle with oci816_8 on WebLogic Server.

The following table provides workarounds to these issues.

If you want to . . .	**Then do this . . .**	**Comments**
Use `Resultset.wasNull` on `LongRaw` NULL datatype	Use the WebLogic jDriver or the Oracle Thin Driver 817. This was fixed for the jDriver with Service Pack 1 for WebLogic Server.	This driver works in preliminary tests, but BEA does not yet fully support Oracle Thin Driver 817.
Change memory usage	Use new environment variable `WL_ORA_CONN_HANDLES`	The WebLogic jDriver for Oracle preallocates fixed number of login data structures when driver loaded; the default is 512. The allocation is static. You can either reduce or increase this number, depending on your memory requirements and number of connections needed.

If you want to . . .	Then do this . . .	Comments
Transactions	Use the WebLogic jDriver for Oracle	The Oracle Thin Driver 816 does not support transactions. Known Oracle bug. XA Resource Manager does not accept a foreign XID implementation. Oracle Thin Driver 817 has XA threading issues. When running multiple transactions in multiple threads in parallel, WebLogic Server may hang.

Known Problems

The following sections describe known problems with this software release. BEA is addressing these issues.

- Documentation Known Problems
- Classloader Known Problems
- EJB Known Problems
- Examples and Pet Store Demo Known Problems
- JavaMail Known Problems
- JDBC and jDrivers Known Problems
- JMS Known Problems
- JSP Known Problems
- JTA Known Problems

- Installer Known Problems
- Operations and Management Known Problems
- Plug-In Known Problems
- Server Known Problems
- Servlet Known Problems
- Utilities Known Problems
- XML Known Problems

Documentation Known Problems

Change Request Number	Description
040177	Additional user information is required when converting `weblogic.properties` files for use with WebLogic Server 6.0. When you choose the root directory for your previous WebLogic server installation, two dialog boxes are displayed. The first box is used to select the servers and the second can be used to select clusters, if any. The conversion proceeds normally if no clusters are selected.

Classloader Known Problems

Change Request Number	Description
CR097716, CR050793	HTTP session failover does not work if you store a serializable object in the HTTP session and place the class file for the object in the `WEB-INF/classes` directory, unless you place the class file in the server's classpath.

EJB Known Problems

Change Request Number	Description
035884	The maximum number of Enterprise JavaBeans (EJBs) that can be deployed on the server is 390. This is due to a Sun bug and is a limitiation caused by the 1.3 JDK; it may be fixed by the 1.3.1 release of the JDK. The Sun Bug ID is: 4390238
037326	`The run-as-specified-identity` tag is not supported.

Change Request Number	Description
039076	The `weblogic.ejbc` utility displays a confusing error message with `<prim-key>/<primkey-field>`: `In EJB AccountBean, the primary key class java.lang.String has a field named CASE_INSENSITIVE_ORDER which is static. No field in the primary key class may be static.` If you have set a `<prim-key-class>` to `java.lang.String` without specifying a `<primkey-field>` you may get this message. This error is technically correct; but is confusing. It means that `weblogic.ejbc` expects that each field in `java.lang.String` corresponds to a `primary-key` field in your bean class.

040409	To deploy EJB 2.0 beans, you must download `ejb20.jar` from BEA Systems and place the `jar` file in the `$WL_HOME/lib` directory. If it is copied to a different location, the `CLASSPATH` used to boot the server must be appended to include the `ejb20.jar` file. If you attempt to deploy an EJB 2.0 bean without `ejb20.jar`, the server throws an EJB 2.0 license exception. The exception is as follows:
	`$$$$$$$$$  License Exception $$$$$$$$$$`
	`Unable to start EJB Service, EJB 2.0 has not been enabled.`
	`$$$$$$$$$  License Exception $$$$$$$$$$`
	If you display EJBs deployed in the Administration Console, the EJB2.0 bean is shown as deployed, but is not actually deployed. Any operations on the EJB2.0 bean may result in exceptions being thrown.

042794	Deploying Message-Driven Beans into a full cluster through the Console’s “Clusters” panel is broken. It is necessary to specify each server individually as a target through the “servers” panel.
042874	In order to run against WebLogic Server 6.0 with Service Pack 2, EJB 1.1 applications must be recompiled against the Service Pack 2 class libraries. You need to run `ejbc` once the Service Pack has been installed.

Examples and Pet Store Demo Known Problems

Change Request Number	Description
041463	Using UNIX, the `config/petstore/StartPetStore.sh` script has been modified to use the Hotspot VM on HP. This is a workaround for a bug in the HP client VM. The Service Pack installer will overwrite the existing `StartPetStore.sh` file. If you have made any changes to this file, your changes will be lost.

043156	The `ejbmanagedclient.jsp` relies on a variable that is set to `t3://localhost:7001` and does not get the port number dynamically.
	So when the page is called, the page gets requested but cannot find the server. The exception is thrown, although the stack trace is not printed out. The page shows up as:
	`--------------UnexpectedError--------------`
	Customers running this example on a different port then 7001, will need to edit the jsp file.
	The variable you must edit is in this section:
	`<!-- Here, we declare a class method -->`
	`<%!`
	`String pagetitle = "JSP example using`
	`EJBean-managed persistence";`
	`String url = "t3://localhost:7001";`
	Edit the URL to point to the correct server/port number that has been installed.

048223	The doumentation included with the WebLogic Server 6.0 WAP examples has not been updated correctly and contains errors. This will be adressed in a future release.
048576	The `setExamplesEnv` script provided with installations of Service Pack upgrades for WebLogic Server 6.0 has errors. JAVA_HOME was set to the wrong directory and as a result, `javac` will not be found when the script is run. Work around this problem by manually setting the PATH and saving the `setExamplesEnv.sh` file. There is also a problem after you run the script. The LD_LIBRARY_PATH is not being set in the environment. This causes an `UnsatisfiedLinkError` when you try to load shared libararies. The work around is to explicitlyexport LD_LIBRARY_PATH=........

Installer Known Problems

Change Request Number	**Description**

039733	This following misleading error message occurs when there is not enough temp space to run the installer:
	`Preparing to install...`
	`The included VM could not be extracted. Please try to download         the installer again and make sure that you download using 'binary' mode.  Please do not attempt to install this currently downloaded copy.`
	By default the installer tries to unpack all the installer files in the `/temp` directory. If there isn't enough space there, you cannot install and will get this message: `You must make room in the /temp directory for the installer files.`

039857	After choosing the Customize option during installation, you are prompted to choose the components to be installed:
	`Please Choose the Components to be installed by this installer.`
	`1- Program Files`
	`2- Samples`
	`ENTER A COMMA-SEPARATED LIST OF NUMBERS FOR THE COMPONENTS TO INSTALL`
	Entering `1, 2` or `"1,2"` results in a partial installation. You must use syntax such as 1,2. Also, problems have been encountered when you choose to install **only** the Samples.

040353	Installation problems can occur on HP-UX and Solaris machines because shared libraries do not have execute permission. File permission changes made by the WebLogic Server installer are reverted by installer-imposed standard attributes. Work around the problem by issuing the `chmod +x` command on all the shared library objects (under `lib/hpux11` and `lib/solaris`).

Change Request Number	Description

043357	If you are getting the following error on Solaris 2.6 or 2.7 UNIX systems:
	`UnsatisfiedLinkError in AWT applications (libmawt.so: open failed: No such file for directory). The policytool will not open. The Java Plug-in Control Panel will not open.`
	Usually you can work around this problem by modifying the LD_LIBRARY_PATH. For example:
	`setenv LD_LIBRARY_PATH`
	`/usr/j2se/jre/lib/i386/motif12:$LD_LIBRARY_PATH`
	In some cases with the `libmawt.so` files, you may not be able to fix this problem and require a patch from Sun: see `http://java.sun.com/j2se/1.3/install-solaris-patches.html` for Sun's patch information.

JavaMail Known Problems

Change Request Number	Description
041450	The default configuration files for JavaMail are not present in the `weblogic.jar`. To work around this problem, download JavaMail 1.2 from *http://java.sun.com/products/javamail* and add the new `mail.jar` to the WebLogic Server classpath before `weblogic.jar`. Unlike JavaMail 1.1.3, which is included in `weblogic.jar`, JavaMail 1.2 includes the POP3 provider from Sun. See the JavaMail 1.2 documentation on the Sun Web site for news about other new features in JavaMail 1.2.

JDBC and jDrivers Known Problems

Change Request Number	Description
033423	There were dbKona problems with `QueryDataSet` and the OCI805_8 driver. The problems have been fixed on Windows NT, but this is still a problem on HPUX and Solaris.
035530	There were problems enabling OS authentication and using Oracle 8.1.6/oci8 with the jDriver. This has been fixed on NT and HP-UX, but still is an issue on Solaris. An error message now reports this problem: `OS level authentication is not currently supported due to a defect in OCI 8 libraries.`
037596	Using `PreparedStatement.setBlob` or `PreparedStatement.setClob` causes a `ClassCastException`.

Change Request Number	Description
040413	Cluster hanging problems have been observed in some instances on Solaris machines. This appears to be caused by a `libthread` deadlock between `SIGALRM` and `SIGLWP`. The deadlock manifests itself when the server is attempting a JDBC connection. To help avoid the problem, set the `InitialCapacity` attribute equal to the `MaxCapacity` attribute in the `JDBCConnectionPool` entry using the Administration Console. A similar problem was previously logged as a Sun Microsystems Bug (ID 4141709 libthread deadlock between SIGALRM and SIGLWP), against Solaris 2.6. There is currently no mention of the problem for Solaris 2.7. For Solaris 2.6, the patch for this problem is 105569-16.
042233	Programmatically defined (not through the Console--through `JdbcServices`) Connection Pools are not working. This issue is being worked on for a future release.
043221	Oracle offers several non-standard (not approved by Sun) extensions to JDBC: ■ `OracleConnection` ■ `OracleStatement` ■ `OracleResultSet` ■ `OraclePreparedStatement` ■ `OracleCallableStatement` If a user attempts to use these classes within an application that gets its connection using WebLogic Server JDBC, they will not work.
043224	Oracle offers the support of its own class of Array (`oracle.sql.Array`) through its ThinDriver. If you attempt to use these classes in an application that gets its connection using WebLogic Server JDBC, they will not work.

JMS Known Problems

Change Request Number	Description
039726	If a pre-6.0 WebLogic Server is shut down when it has active database operations, after upgrading to 6.0 some durable consumers may receive messages that were sent to the topic after the durable subscriptions were created. Problems can occur with unstable topic messages. The state in the database is not updated by the scavenger when the server is killed. The messages are still UNSTABLE when they are supposed to be stable. When the new server starts, all unstable messages are re-published to all durable subscribers that did not acknowledge to the old server. The problem is that durable subscribers do not have a timestamp recorded indicating when they are created. Therefore, there is no way to know which was created first, a particular message or a durable subscription. Before shutting down the pre-6.0 WebLogic Server and upgrading to WebLogic Server 6.0, give the scavenger time to update the database. Shut down the server gracefully and never stop the server when it is in the process of sending or receiving messages.
039809	According to the JMS 1.02 Specification, a received message can not be modified. For `StreamMessage` and `BytesMessage` safeguards exist against writing over received message. However, no protection exists for the other message types. Users need to implement safeguards against overwriting messages of other types.

JSP Known Problems

Change Request Number	Description

033844	The following error message may be received when compiling JSP files on a UNIX machine:
	`failed: java.io.IOException: Not enough space`
	If you receive this message, try any, or all, of the three steps below:
	1. Add more RAM if only you have only 256M.
	2. Raise the file descriptor limit, for example:
	`set rlim_fd_max = 4096`
	`set rlim_fd_cur = 1024`
	3. Use the `-native` flag to use native threads when starting the JVM.

039083	Forwarding a JSP page does not work. For example, the following commands do not work correctly:
	`jsp2 RequestDispatcher rd =`
	`request.getRequestDispatcher("jsp2");`
	`rd.forward(request, response);`
	To work around this problem, forward pages using the JSP tag (`jsp:forward`).

041480	Several new options have been added to the WebLogic JSP compiler (`weblogic.jspc`). These new options allow for compiling JSPs that use EJBs in Web Applications. For details, see JSP Descriptor Element `at http://e-docs.bea.com/wls/docs60/programming/weblogic_xml.html#jsp-descriptor`.
042469	The `precompile` option defined in the `<jsp-descriptor>` element of the `weblogic.xml` deployment descriptor for a Web Application is no longer available.
044350	Setting precompilation of JSPs in the `web.xml` deployment descriptor by setting `weblogic.jsp.precompile` to `true` only precompiles JSPs if the Web Application containing the JSPs is deployed in exploded directory format.

JTA Known Problems

Change Request Number	Description
040378	The `BeforeCompletionIterationLimit` may appear to be reached prematurely. Increase the `BeforeCompletionIterationLimit` attribute to be greater than or equal to the maximum number of servers expected to participate in a single transaction.
042362	According to the JTA spec documentation of `setTransactionTimeout()`: "seconds: The value of the timeout in seconds. If the value is zero, the transaction service restores the default value." If the value is zero, WebLogic Server is not resetting the default value.

Operations and Management Known Problems

Change Request Number	Description
034955	The application directory list used to scan for deployments is not persistent. Multiple application directories are not supported.
034177	If an application's deployment status is set to false on the command line, the console still shows the application as deployed, even when restarting or refreshing the browser the console is running in.
035670	When you reset an MBean name, the configuration is not saved correctly. To work around this problem: 1. In the Administration Console, clone the MBean. 2. Rename the clone. 3. Delete the original MBean.

Change Request Number	**Description**
036201	Occasionally, after making modifications in the Administration Console, WebLogic Server cannot be restarted and a `Fatal initialization exception Throwable: weblogic.server.ServiceFailureException: Initializing JNDI: javax.naming.ServiceUnavailableException [Root exception is java.net.UnknownHostException: Unknown protocol: 'Protocol: 'unknown'']` message is displayed.
032486	JSP exceptions are not correctly propagated to the Administration Console.
036634	In the Administration Console, the Jolt/WLEC Pool | Monitoring link is broken. (Jolt is not supported in 6.0, but WLEC is.)

038589	If Managed Server(s) are communicating via HTTPS with the Administration Server, the discovery of Managed Servers does not work. For example, if the Administration Server is restarted with the `-Dweblogic.management.discover=true` command line argument and the running Managed Servers were booted with the `-Dweblogic.admin.host=https://<adminhost>:<adminport>` argument, then the discovery of Managed Servers does not work.
	Workaround: When Managed Servers are booted with an HTTPS connection to the Administration Server, do not restart the Administration Server with the `-Dweblogic.management.discover=true` argument.
	Instead, do the following:
	1. Restart the Administration Server without the `discover` flag.
	2. Once the Administration Server is booted, force discovery by the Admin Server by running the following command:
	`java weblogic.Admin -url <ADMINHOST>:<ADMINPORT> -username system -password <SYSTEM-PASSWORD>`
	`INVOKE -mbean "<YOURDOMAIN>:Type=Domain,Name=<YOURDOMAIN>" -method discoverManagedServers`

039033	WebLogic Server logs an error when an application that does not exist is undeployed.

Change Request Number	Description
039340	Errors that occur in WebLogic Server before the kernel, administration, and logging are initialized should go to the `*servername*-startup.log` file. A message at the end of the startup file (`close msg`) indicates that post-initialization messages from this server will be in the regular log file. The first message in the regular log file also points to the startup file. However, several confusing things may happen: Sometimes the `close msg` is logged in the beginning of the regular log file, and the next message may point the user to the regular log file for errors at startup time. Thus, two startup log files may be created `-weblogic.log` and `*servername*-startup.log3`. When a component has pre-initialization errors, that component may keep logging in the same startup log file even after the regular log file is being used by the rest of the server.
039253	In the Administration Console under Web Applications, when you click Console sometimes two Console nodes appear. Clicking on the second node does not result in an error.
039311	When you start a managed server that has been cloned, it throws a `ConfigurationException`. The workaround is to create a server instead of cloning it.
039532	An `.ear` file can only be undeployed using the Administration Console once it has been exploded under the applications directory. Also, once the application is undeployed, the only way it can be redeployed is through the Administration Console.

Change Request Number	Description

040034	Attempting to start a server as a Managed Server when it is already running as an Administration Server results in the following misleading error:
	`<Dec 8, 2000 7:44:02 PM PST> <Error> <Configuration Management> <Error connecting to admin server and initializing admin home: admin URL: t3://localhost:7001`
	`java.lang.NullPointerException`
	Workaround: This is incorrect usage. The same server cannot be started as an Administration Server and a Managed Server. Define a new server for your Managed Server, and specify its name in `-Dweblogic.Name` during startup.

042545	Managed Servers in the same domain can hang if started simultaneously. Delay about 30 seconds before starting each Managed Server.

039736	If you deploy an EJB using `weblogic.deploy`, it copies the `.jar` file to the applications directory. The `.jar` file cannot be removed from the applications directory. Attempts to remove the `.jar` file result in the following message:
	`C:\weblogic\dev\src\tools\weblogic\qa\tests\config\mydomain\applications>rm matt.jar`
	`rm: cannot unlink entry "matt.jar": The process cannot access the file because it is being used by another process.`

039995	A server startup error occurs after you create a WLEC Connection pool using the Administration Console. As a workaround for this problem, edit `CLASSPATH` to add `wleorb.jar` and `wlepool.jar`. On Windows NT machines, edit `%WL_HOME%\config\mydomain\startAdminWebLogic.cmd`. Add `.\lib\wleorb.jar;.\lib\wlepool.jar` to `CLASSPATH`. On UNIX machines, edit `$WL_HOME/config/mydomain/startAdminWebLogic.sh`. Add `$WL_HOME/lib/wleorb.jar:$WL_HOME/lib/wlepool.jar` to the `CLASSPATH`.

Change Request Number	**Description**
040124	While the Administration Server is running, copying an existing `.jar` or `.ear` file to an applications directory does not redeploy the application. The following error is displayed in the server log: `javax.naming.NameAlreadyBoundException: Can't rebind anything but a replica-aware stub to a name that is currently bound to a replica-aware stub; remaining name '' <Dec 12, 2000 12:09:37 PM PST> <Error> <J2EE> <Error deploying application <YOUR-EJB-NAME>: Could not deploy: '<YOUR-FILE-NAME>': JNDI name in use` Make sure that either the `guest` user or the `everyone` group is granted lookup permission to `weblogic.jndi`. The `fileRealm.properties` file should contain the following: `acl.lookup.weblogic.jndi=everyone`
040171	The attribute `SingleThreadedServletPoolSize` is not settable through the Administration Console. You can, however, define this attribute in your `config.xml` file, under the `WebAppComponent` element. The default value for this attribute is 5 and sets the initial number of servlet instances that are created on server startup when using the Single Thread Model for HTTP servlets.
040479	When creating topics dynamically using the JMSHelper class methods, as described in Developing a WebLogic JMS Application at `http://e-docs.bea.com/wls/docs60/jms/implement.html`, the configuration information that is generated is invalid.
041290	If the username property is set from the `weblogic.Admin` or `weblogic.Server` command lines, the username must be “`system`”. Note however, that the username syntax is retained to prevent breaking a user's scripts. The username may not be modified in the `config.xml` file. The `weblogic.management.password` property value must be the system password.
041321	Configuring HTTP access log rotation by date or by size does not function. You must rotate HTTP access logs manually.
041543	Using the console, the auto update interval value cannot be changed. If you enter a new number in the ‘Auto Update Interval’ tab, and click "Apply" an error message is displayed and the value is not modified.

Change Request Number	Description
043091	An MBeanHome obtained from the Administration Server at `weblogic.management.MBeanHome.JNDI_NAME+"."+<adminServerName>` returns all MBeans in the entire domain--the home is identical to the administrative mbean home available at `weblogic.management.MBeanHome.ADMIN_JNDI_NAME`. Workaround: To select the run-time or configuration MBeans only relevant to the Administration Server, filter the object names in which `Location=<adminServerName>`. The value of 'Location' can be obtained by invoking: `mbean.getObjectName().getLocation().`

Plug-In Known Problems

Change Request Number	Description
041744	If you are running IIS (with the WebLogic plug-in) on Windows NT 4.0 with Service Pack 5 or 6, crashes may occur when shutting down the IIS server. It is not known yet if the problem is due to the plug-in or NT4.0 SP5/SP6.
049648	The domestic strength plug-ins are not available with the Service Pack upgrade. However, they are available with the full installers of WebLogic Server 6.0 that include the Service Pack. If you wish to have the 128 bit plug-ins, as well as the latest Service Pack you will have to download a full installer (WebLogic Server 6.0 with Service Pack 2) and place it in a BEA home different than your existing BEA home. Then you will have to move your configuration files and applications to the new BEA home. Also, installation of domestic strength plug-ins (128-bit) requires the presence of a license enabled for domestic strength SSL on the target machine. This license must reside in the BEA Home directory prior to installation.

Server Known Problems

Change Request Number	Description
037673	WebLogic Server may hang when used with the HotSpot server VM on Windows NT or Windows 2000. Use the HotSpot client VM on Windows NT or Windows 2000 instead.
038883	Specifying the root directory when starting the server does not work when the root directory is not under a `/config` directory. The server always looks for a config directory in the current directory. To work around, the server should be started from the directory which contains `config/`*`domainName/`*`...`

Change Request Number	**Description**
039575	If your servlet makes an HTTP connection back to the same server there is the possibility of deadlock under load. To work around the problem, use a separate execute queue for the servlet that returns the DTD.
041753	Some inconsistencies exist in the order in which EAR module components are deployed. This problem is being addressed.

042556	The `weblogic.Admin` command now requires the user and password options. For example:
	`java weblogic.Admin -username system -password gumby1234`
	The username “`system`” is required for most functions (for example: VERSION). If you do not specify `-username system`, you will get the following error:
	`Exception in thread "main" java.lang.SecurityException: Authentication for user system denied in realm weblogic`
	`<<no stack trace available>>`

044596	Web-Applications in an open directory structure being redployed after installing Service Pack one may cause an ‘application not found’ error to occur. To solve this problem, make sure your `config.xml` the **absolute path** to your web application; you cannot use a relative path such as `Path="/webapp".`

Servlet Known Problems

Change Request Number	Description
041528	Cookies generated by the first release of WebLogic Server 6.0 are not recognized by any later version of WebLogic Server 6.0.
041687	The CGI-Servlet does not work when being called from a Web-Application deployed as a `.war` file, or when the CGI is registered in a Web-Application and the "cgiDir" parameter points to a relative path.
041689	When using form-based authentication, an HTTP session object is created when an HTTP authentication form is served. Therefore, the `session.isNew()` method always returns `false` after authentication.
041867	Several new columns have been added to the database table used for JDBC session persistence. For more information, see Using a Database for Persistent Storage at `http://e-docs.bea.com/wls/docs60/adminguide/config_web_app.html#jdbc_persistence`.
042473	Web Application authentication is now valid for all Web Applications deployed in a WebLogic Server domain. This behavior, which enables you to use a single sign on for all Web Applications in a domain, is mandated by the Servlet 2.2 specification from Sun Microsystems. Therefore, a user authenticated in one Web Application is authenticated for all Web Applications in the domain. As a consequence of this behavior, you can no longer use the `session.invalidate()` method to log out a user from all Web Applications. A user is logged out of all Web Applications after all sessions in all Web Applications have timed out, when the cookie expires, or if the session.invalidate() method is called in every Web Application that a user has visited.
043089	When using the `HttpClusterServlet` to proxy HTTP requests to a cluster of WebLogic Servers, do not deploy any Web Applications other than the Web Application that contains the `HttpClusterServlet` on the server that is proxying requests to the cluster.

Utilities Known Problems

Change Request Number	Description
039830	The default access control lists (ACLs) resulting from the conversion of `weblogic.properties` files may be incomplete. After converting your `weblogic.properties` file,use the Administration Console to make sure the security settings are the appropriate ones for your environment.

XML Known Problems

Change Request Number	Description
034785	WebLogic Server does not support namespaces in the XML registry.
036428	The `javax.xml.*.parse(File)` method in Apache's implementation of the JAXP classes in Xerces 1.2.0 is broken. The broken method is the bundled Apache jar file, not in the built-in Xerces parser. WebLogic's implementation of this method is correct. To work around this problem, be sure to specify the WebLogic JAXP jar file (`WLS_HOME/lib/xmlx.jar`) in your CLASSPATH before you specify the bundled Xerces jar file.
041540	The Xerces parser serializer bundled with WebLogic Server does not support multi-byte encodings, such as `Shift_JIS` or `EUC-JP`.

CHAPTER

5 Resolved Issues

The following sections contain information on issues that were resolved for the release of WebLogic Server 6.0 with Service Pack 2 (Service Packs are cumulative, so all the changes in earlier Service Packs are included in the latest Service Pack):

- Service Pack 2 Fixes
  - EJB Issues
  - Examples Issues
  - JDBC Issues
  - JMS
  - JSP Issues
  - Miscellaneous Issues
- Service Pack 1 Fixes
  - Console Issues
  - EJB Issues
  - Examples Issues
  - JDBC Issues
  - JMS Issues
  - Internationalization Issues
  - Plug-in Issues
  - Servlets and JSP Issues
  - XML Issues
  - Miscellaneous Issues

Service Pack 2 Fixes

EJB Issues

Change Request Number	**Description**
042956	Fixed a problem with jar files that contained many beans (100+). In these circumstances `ejbc` was failing when running under Windows.
043398	Fixed some problems with a message-driven bean calling a stateless session bean when the two beans are packaged in different jar files. A classcast exception occurred when narrowing on the home of the called bean.
043640	Fixed some problems in EJB in which certain log messages occurring with rolled back transactions were exceedingly verbose.
044179	An unintentional debug statement in the server's internal EJB container has been removed.
044605	Serializing a JMS message that has never been sent resulted in a `NullPointerException`. This happened during the writing of unsent messages to a stream or passing an unsent message as a parameter via an RMI call, including EJB invocations. To fix this issue, WebLogic Server 6.0 with Service Pack 2 or higher must be installed on the JMS client.
044755	Fixed some problems with primary key classes when migrating 1.1 CMP beans. In WebLogic Server 5.1 the primary key class fields were not included within the updated fields, whereas in WebLogic Server 6.0 they **were** updated. Now we have returned to the earlier, correct, behavior--where the primary key class fields are not included as an updated field.

Examples Issues

Change Request Number	Description
042366	Running the Security Audit Example was causing a run-time exception in WebLogic Server. This has been fixed.

JDBC Issues

Change Request Number	Description
038235	Fixed some problems with the jDriver concerning double data types in Oracle.
041793	If a database went down during attempts to access the database, the initial reset of the JDBC connection pool was failing. This has been fixed.
041866	Using `CLOB.getCharacterStream()` is no longer a problem with the Weblogic jdriver.
042134	Fixed some problems with dynamic connection pools.
042279	Multipools now work with a `TxDataSource` with the restriction that all connection pools in the `TxDataSource` must either use the same driver, or they must all use drivers that have the same XA capabilities. (either all the drivers must be XA compliant, or all drivers must not be XA compliant).
043745	When Oracle returned OCI_SUCCESS_WITH_INFO, the WebLogic jDriver returned a warning message without providing the data. Now, both the warning and data are given.
043956	Fixed some problems on HP-UX 11.0 when using the WebLogic jDriver and executing certain SQL commands.
044223	Fixed a problem with connection pools not failing over.

045965	Included some fixes in order to get around XA-Resource issues with the Oracle 8.1.7 thin driver.

JMS

Change Request Number	Description
041031	Fixed problems with asynchronous JMS messages, whereby the last message was not being acknowledged. To fix this issue, WebLogic Server 6.0 with Service Pack 2 or higher must be installed on the JMS client.
043118	Fixed a problem with using a MQSeries JMS Topic driving a message-driven bean. The following exception was being raised: `javax.jms.JMSException: MQJMS1057: Message has no session associated with it`.
043155	Topic message expiration times are no longer ignored for asynchronous subscribers.
043225	During transmission of ObjectMessages, the object stored in the message was serialized at least twice (possibly more for topics). For most situations this has been reduced to once. In addition, calling toString on an ObjectMessage would previously return null if the object was not first unserialized by calling getObject(). This is no longer the case. getObject() is now called from within the toString method of an ObjectMessage.
043519	Fixed a problem in JMS that occurred during session rollback or recover. Asynchronous consumers were refusing to receive messages. To fix this issue, WebLogic Server 6.0 with Service Pack 2 or higher must be applied to both the JMS client and server.
043752	In JMS, recovered expired messages no longer cause incorrect `NullPointerExceptions`.
044285	Fixed a problem in JMS. Multiple simultaneous transacted asynchronous durable subscriptions were possibly causing a server lock-up if not enough threads were configured. Now, no extra threads are needed to service such simultaneous requests.
045963, 045939, 045956	Connection consumers for session pools are behaving properly now.

JSP Issues

Change Request Number	Description
042360	In JSPs, `getCharacterEncoding()` was returning ISO-8859-1 even if `charset` was set to `Shift_JIS` or some other character set. Now the proper character set is returned.
043167	JSP pages that use includes with relative paths (i.e. `<jsp:include page="../bar.html"`) no longer cause parsing errors.
044023	WebLogic's JSP compiler was not case sensitive. If you were using Windows NT and had compiled `HelloWorld.jsp`, the generated file would have been `_helloworld.class`, not `_HelloWorld.class`. The default behavior of the JSP compiler is now case sensitive.
045895	A JSP page that was using a `jsp:useBean` tag and had '`type`' set to '`BeanInterface`' was not working. The following exception was occuring: `weblogic.servlet.jsp.JspException: Could not set property: someValue: java.beans.IntrospectionException: java.lang.Object not superclass of BeanInterface.` This problem has been fixed.
046534	When JSPs are registered in `web.xml`, the '`compileCommand`' entry is no longer ignored.

Miscellaneous Issues

035169	Fixed problems with form authorization in a Web Application. There was a misleading message when users who were already logged in went directly to the `<form-login-page>` specified in the `web.xml` deployment descriptor. The message returned to the browser was:
	`Form based authentication failed. Could not find session.`
	The following error was printed to the log:
	`<E> <WebAppServletContext-test> Form based authentication failed. One of the following reasons could cause it: HTTP sessions are disabled. An old session ID was stored in the browser.`
	This error is incorrect--at that point the user is still logged in, has a valid session, and you can still access the protected resource. Now the server no longer generates these errors and instead, in this case, redirects the user to the parent directory of the login page, which causes the server to look for and serve the Web Application's welcome page. For more information see *Configuring Welcome Pages* at http://e-docs.bea.com/wls/docs60/adminguide/config_web_app.html#welcome_pages.

042661	Fixed a problem in clustering. When a replica-aware stub is called within a transaction, it attempts to route the call to the server hosting the transaction coordinator. The problem was that the stub was sticking to this server for the life of the stub, even if it was called a second time outside of the transaction.
043010	Instances of WebLogic Server 6.0 running as a Windows service no longer shut down automatically and become inaccessible after you log out of the NT session in which the server was running in.
043493	JNDI is no longer accessed prior to it being properly initialized during server startup.
043695	The value returned by `session.getId()` no longer returns the same value as `request.getRequestedSessionId()`. Also note: to determine whether an incoming cookie has a valid session, use `request.getSession(false)`.

043865	Fixed a problem that caused an `EXCEPTION_ACCESS_VIOLATION` when shutting down a WebLogic Server that uses a RDBMS realm with the Weblogic type 2 Oracle JDBC driver for Oracle 8.1.6 on win32.
044105	Fixed problems that occurred when starting the server as an NT service while using the `-hotspot` option.
044282	An unnecessary reverse DNS lookup has been removed.
044450	Removed unnecessary code in JDBC Store that was causing problems with TimesTen.
044556	Fixed a problem with the serialization of HTTP query parameters.
045960	IDL files generated by `weblogic.ejbc` can now be compiled by an IDL compiler.
044987	Fixed problems in the java socket multiplexor with applets reloading/refreshing the browser.
045091	In EJB 2.0 the transaction context is no longer lost if remove() is called on a stateful session bean with a transaction attribute of `NotSupported`.
046058	The WebLogic Plug-ins no longer cause a long delay when they route URL requests with `SecureProxy=ON.`

Servlet Issues

Change Request Number	Description
042281	A servlet no longer continues executing after the browser is closed.
044579	An incorrect `IllegalStateException` is no longer thrown in certain situations when a servlet in a Web Application handles file upload and then has to process the input.
045580	There are no longer problems when mapping a `ServletException` to an error page in `web.xml`.
043541	The servlet engine now properly parses all URLs not ending in '/' or an extension.

Service Pack 1 Fixes

Console Issues

Change Request Number	Description
035788	Fixed some hyperlinks to our documentation that were broken in the Console.
038336	Fixed HTML problems in the Console where some anchors were going to the right help page but not to the right place within that page.
038359	A Multibyte machine name is now displayed correctly when creating a Kanji Machine name.
039629	On the Web App General panel of the Administration Console, there is now a check box indicating that the Web Application is deployed.

040029	A help icon in the Console is no longer displaying an Error 404--Not Found.
040101	Fixed a problem with graphs in Netscape when using monitoring in the console.
040127	On Japanese NT, fixed a problem with the Console and using localized log messages.
040181	When creating a 'Machine' on the Japanese NT, the 'CREATE' button was not making any change in the console. The create page was still virtually the same after clicking the 'CREATE' button. This is now fixed.
040820	Fixed a problem when the ‘Create’ button was selected prematurely while trying to configure a new web application.
040846	Fixed a problem with Netscape and the WebLogic Console. When creating a new conection Factory, at the JNDIName text box, a dialog box said, "This field cannot be blank". This dialog would not disappear until killing the browser.
040865	Fixed a problem with the cluster monitoring tab in the console. It was not showing the correct number of participating servers.
040902	Fixed an unnecessary field in the console when clicking on the garbage can in the security-->group node.
041080	Fixed a problem with the console and security when creating a multi-byte group name and registering a user to the group.
041096	Fixed a problem with Japanese characters in the console.
041112	Fixed a problem with certain domain settings not being persisted and returning to the defaults on restart.
041117	The Console was having problems when ‘refresh’ was being used regularly in the browser. These problems have been fixed.
041141	The Console no longer displays corrupt strings when the navigational tree is disabled.
041268	Fixed a problem in the Console when switching languages from Japanese to another language.

041301	In the Console, the Max Size value under the HTTP tab had a unit of kilobytes. It has been changed to bytes.
041532	Fixed problems with the Console, involving undeploying and redeploying Web Applications.
041606	When entering 'Search text' in the log customize window in multi-byte string, it was displaying the search result. But when displaying the log customize window a second time, the initial value of the string which was entered in the previous search was corrupted. This has been fixed.
041733	The Console was setting the XMLRegistry MBean attributes to "" if they are unset. This confuses the XMLRegistry which thinks the are settings to the value empty string. The XML Registry now considers "" to be the equivalent of an unset field.
041837	There was a problem starting the Console and changing the rotation type to date based in http logging and then restarting the Server.
041875	Added some information to the console for users who delete an ACL.
041943	Fixed a problem in the console when using a Kanji group name.
041944	Fixed a bug in the console when using a Kanji character as an ACL name.
041994	Fixed errors when creating a new domain using the Console.
042058	In the JNDI tree of an active server, nodes of bound objects no longer have incorrect '(access denied)’ labels.
042068	Removed the ability to clone applications from the Console.
042181	The unit in which the configuration parameter "Logging -> Size Threshold" is identified now as 'kilobytes' on both the domain and server logging tabs.
042278	You can now deploy a MultiPool and specify the targets using the Console.
042434	Fixed a problem deleting a multibyte group name.

EJB Issues

Change Request Number	Description
031228	Improved error messaging with `ejbc`. The full stack trace is now printed out when a `NoClassDefFoundError` occurs.
031852	It was possible to set transaction attributes for non-existent methods. A misspelled method name in the deployment descriptor could cause a transaction attribute to not be set. We now validate that all method names specified in the deployment descriptors actually exist in both the remote interface and the bean implementation class.
033538	Fixed a problem in EJB with queries and escape clasuses. Also fixed problems with the Java compiler and the '\' character.
034691	It is incorrect to use EJB 2.0 with WebLogic 6.0 and a WebLogic 5.1 RDBMS XML deployment descriptor. When doing so, a confusing `ClassCastException` was being thrown. Now we inform users that this configuration is incorrect.
035395	Finders were only working if all of the finder method parameters were referenced exactly once by the query. This has been fixed.
036007	Now we throw the appropriate error message when using `ejbc` for a cmp11 type bean with one of the attributes as private.
037622	Removed redundant messages (`setMessageDrivenContext` called, `ejbCreate` called, `setMessageDrivenContext` called) when running the sample server on UNIX.
037957	Now EJB-QL: WHERE clauses are commutative.

039179	There was a confusing and misleading error message in the EJB examples. The message was:
	`log("UNABLE to get a connection from demoPool!");`
	`log("Please make sure that you have setup the connection pool properly").`
	Now this message reads:
	`log("Failed to lookup JDBC Datasource. Please double check that"); log("the JNDI name defined in the resource-description of the ");`
	`log("EJB's weblogic-ejb-jar.xml file is the same as the JNDI name ");`
	`log("for the Datasource defined in your config.xml.")`

039477	Fixed the error message that is displayed if the `ejb20.jar` is not in the server classpath. There are now two distinct errors if EJB 2.0 cannot be started - either it's a `LicenseException`, and the user should contact a sales representative to get a license,or EJB 2.0 has not been enabled and the error message indicates where users can download the EJB 2.0 upgrade to enable it.
039518	Our CMP 2.0 code was not retrieving cmp-fields that were an `EJBObject`. This has been fixed.
039562	Message-Driven EJBs with the `NotSupported` transaction attribute are now destroyed when a System Exception is thrown.
039635	The latest revision of the EJB 2.0 spec requires that CMP and CMR fields start with a lower-case letter. We have added this check to our compliance checker, and made sure our examples do not violate this rule.

039736	If you deployed an EJB using `weblogic.deploy`, it copies the `.jar` file to the applications directory. The `.jar` file cannot be removed from the applications directory. Attempts to remove the `.jar` file result in the following message:
	`C:\weblogic\dev\src\tools\weblogic\qa\tests\config\mydomain\applications>rm matt.jar`
	`rm: cannot unlink entry "matt.jar": The process cannot access the file because it is being used by another process.`
	This has been fixed.

039818	Fixed `AssertionError` when Primary Key fields were Serializable objects.
040052	Removing a bean from a M-N relationship wasn't removing join table entries. This has been fixed.
040066	The setting `max-beans-in-free-pool` now determines the maximum number of concurrent instances that can be created to service messages. It is bound by the number of `ExecuteThreads`.
040108	EJB Compliance Checker now allows `java.lang.Object` as the primary key type.
040303	Having two EJBs with the same `<abstract-schema-name>` creates an inaccurate and misleading error message: `org.xml.sax.SAXParseException: The content of element type "ejb-jar" must match "(description?,display-name?,small-icon?,large-icon?,enterprise-beans,dependents?,relationships?,assembly-descriptor?,ejb-client-jar?)".` We have added a compliance check for this problem.
040461	Set transaction state to committed as soon as the last `ackCommit` message is received.
040469	Previously when you added a message-driven entry to your `ejb-jar.xml` file but neglected to add its counterpart to the `weblogic-ejb-jar.xml` file, ejbc did not give you an error message. This has been fixed.
040470	If a EJB fails to deploy all of the ejbs within the ejb, now the jar file will be undeployed.
040564	Added support to EJB for instances when a Primary-key includes a foreign key column for 1-n and 1-1 relationships.
040586	The EJB compliance checker now ensures that finder return types are the remote interface, `java.util.Collection`, or `java.util.Enumeration`.
040593	The EJB comliance checker now detects when the EJB-QL for a finder method contains a SELECT clause.

040594	Using an abstract base class to declare the public abstract getter and setter methods for the CMP fields of an EntityBean, `ejbc` wasn't able to find the methods on the EntityBean and refusesd to code-generate the container classes. This has been fixed.
040697	Fixed a `ClassCastException` when undeploying Message-Driven EJBs.
040740	Enabled monitoring of managed servers in deadlock and overload situations.
040788	We now allow message driven bean users to specify `NO_ACKNOWLEDGE` or `MULTICAST_NO_ACKNOWLEDGE` for their acknowledgement mode.
040907	Fixed a problem with message driven beans when calling other EJBs in the same ejb-jar.
040924	Now we conform to the latest EJB spec revisions, which don't allow protected CMR get/set methods. These methods must now be public.
041041	Improved performance using EJBs and big CMP 2.0 creations.
041055	Before SQL was generated, an error was not being caught when a CMP field and a CMR field were mapped to the same database column.
041133	The current version of the EJB QL spec states that the 'FROM' clause is required in every query (see EJB2.0 spec sec 10.2). The examples have been updated to reflect this.
041284	Message-Driven EJBs now support external JMS Providers.
041306	During JSP compilation, the EJB interfaces within the same application were not available. This has been fixed.
041369	Fixed `AssertionError` when `EJBMetaData` was used from within the server and the EJB's classes were not in the classpath.
041430	When there was a `NoClassDefFoundError` within ejbc, the stack trace was being omitted. This has been fixed.
041728	When deploying CMP Beans, if the DataSource/Connection was unavailable, users were seeing a misleading message which suggested that the Bean's table didn't exist. This has been fixed.

041801	A `ClassCastException` was occurring when replicated stateful session beans were compiled without an In-Memory replication license and then deployed in a server with the proper license.
041918	Calling a replicated stateful session bean in a non-clustered server no longer causes an Assertion Failure.
041937	If `<primkey-field>` isn't a CMP-field, users were getting a `java.lang.NullPointerException`. This has been fixed.
041986	The EJB compiler was generating invalid SQL queries for the finder methods of 1.1 CMP Beans.
042041	1-N relationships were failing when the beans in the relationship had a compound key, and when the foreign-key used in the relationship was also part of the primary key of one of the beans. The failures manifested when a bean participating in the relationship was removed, and also when ejbc is run in some cases (ejbc fails). This has been fixed.
042132	Any EJB method whose name starts with 'remove' was being deployed as `NotSupported`. This has been fixed.
042217	EJB QL queries that contained a 1-1 or 1-n join weren't working with compound keys. This has been fixed.
042430	The `RemoteException` detail field is now set correctly in EJB.
042483	Fixed a problem when an EJB marks a transaction for rollback and then calls the `RequiredNew` method.
042637	When removing a bean, the server wasn't initializing the bean's CMP variables before putting it into the pool. This caused errors later if the bean was reused. This has been fixed.

Examples Issues

Change Request Number	Description
037669	The WLEC university sample was failing to make a connection when the connection pool initializes. The server couldn't connect because the password specified by the example is greater than 8 characters long. This has been fixed.
039403	In the left hand pane of the console, the audit provider example (found at `/security/audit/package-summary.html`) was referencing an Audit Provider node that didn't exist. This has been fixed.
040411	When using the Administration Console with the Examples server and the default Cloudscape RDBMS realm, you can now shut down the server. Attempts to shut down the server were failing because the user `guest` did not have `shutdown` permission for `weblogic.admin`. The example Cloudscape database did not contain the appropriate ACL to allow the user `guest` to shut down the server.
040429	Fixed a problem in our examples. In the security acl example the documentation has a typo; `java examples.security.acl.Client t3://localhost:7002 joeuser joepass` should have been: `java examples.security.acl.Client t3s://localhost:7002 joeuser joepass`.
040508	Modified all examples to use `java.rmi` and JNDI instead of `weblogic.rmi`.
040633	After undeploying the Pet Store Web Application and then deploying it again to a managed server a `javax.management.MBeanException` was being thrown. This has been fixed.
040682	The JAAS security example was missing an implementation of `AltClient`. `AltClient` is the client side implementation showing how to use mutual authorization with WebLogic.
040954	Added seven new XML examples.

041643	Fixed the cluster connection string example. WebLogic requires that all members of a cluster share the same port.
042136	The `examples.utils.ddl.demo.ddl` file has a mistyped statement; an extra T in DROP: (DROPT table).
042140	A DataSetException was occurring when trying to execute the `samples\examples\dbkona\rowid` example. This has been fixed.
042213	In the date and phoneBook examples, there was an error in the documentation. Under Run this Example, "From the toolkit's General Menu," should have been: "From the Nokia Active Server manager general Menu."
042244	The `setExamplesEnv` script and the `startExamplesServer` script now include the `xmlx.jar` in the CLASSPATH.
042256	The `web.xml` in the examplesWebApp did not have a taglib-uri element for the xslt tag library.
042261	Improved the documentation in `examples/i18n/logging/startup/package-summary.html`
042583	When building the `ejb/simpapp` WLEC sample, the jspc compiler complains that it can't find the `web.xml` file. We added a -webApp flag to the jspc command to fix this. There was a similar problem, also fixed, with `examples/jdbc/pool/build.cmd`.
042585	The instructions for running the university samples say the following: Start a web browser and enter the URL for UniversityForm.html. For example: `http://localhost:7001/examplesWebApp/UniversityForm.html` But the name of the file is actually `universityForm.html`. (The 'U' in university should have been lowercase.)
042586	The WLEC Simpapp Servlet example now catches any `PoolUnavailable` exceptions.

Internationalization Issues

Change Request Number	Description
035036	When WebLogic boots and prompts for the boot account's password, the prompt used is now internationalized.
041021	Fixed a problem failing to generate properties files for some message catalogs after executing l10ngen.
041078	Fixed some duplicated keys in the `english.xml` file.
041264	Fixed some internationalization problems with certain tags not being in `english.xml`.
041266	The time stamp is now internationalized.

JDBC Issues

Change Request Number	Description
033423	There were dbKona problems with `QueryDataSet` and the OCI805_8 driver. This has been fixed on Windows NT, but is still a problem on HPUX and Solaris.
033920	Change There was a problem in JDBC when running tests which access the connection pool.
035530	There were problems enabling OS authentication and using Oracle 8.1.6/oci8 with the jDriver. This has been fixed on NT and HP-UX, but still is an issue on Solaris. We have added an error message that reports this problem: OS level authentication is not currently supported due to a defect in OCI 8 libraries.
037589	Added Oracle thin driver support and fixed some outdated command line help.
037591	Added support for ORACLE_THIN database type. Removed obsolete database types and command line options.

037693	There are no longer problems with the Oracle driver and ROWID selects when running `examples.dbkona.rowid`.
037893	Fixed `getBytes()` to return null instead of a zero-length byte array with Oracle LONG RAW columns.
038143	`JDBCConnectionPoolRuntimeMBean`'s were not working. Added a `RuntimeMBean` implementation to `ConnectionPools`.
038229	The jDriver for Oracle 8.1.6 caused managed servers to run out of memory.
038275	We no longer refer to the `weblogic.classpath` in one of our JDBC error messages. The `weblogic.classpath` does not exist in WebLogic 6.0.
040379	The following error occurred when the `server` and `serverName` properties in a JDBC connection pool were specified with the same value: `<Cannot startup connection pool "jtaXAPool" server and serverName properties must have the same value>` This problem was seen when the XA driver requires the `server` or `serverName` property to be set. It was occurring with the WebLogic jDriver for Oracle, but not the Oracle thin driver, or the Cloudscape driver. It has been fixed.
040509	Fixed a small problem with Solaris 2.6 and the jdriver.
041286	Better error message for misconfigured connection pools
041302	The MS SQLServer driver has been updated.
041514	Added support for JDBC and connection pool monitoring in the Console.

JMS Issues

Change Request Number	Description
038109	The JMS connection is now cleaned up properly when the client terminates without closing.
039249	The managed server no longer has a memory leak when it monitors JMSServer using the Administration Console.
039802	Closing non-durable subscriber messages that were not being pushed to the subscriber at the time, such as messages for the synchronous subscriber, was sometimes causing message statistics to become out of sync. This has been fixed.
039921	A multicast subscriber was not able to receive multicast messages on Solaris machines. This has been fixed.
040088	Problems were occurring when attempting to send messages from a connection consumer's listener class `OnMessage` method using the same session that a session pool uses to send messages to the connection consumer.
040479	Fixed a bug in `JMSHelper` that was causing incorrect xml to be written to the `config.xml` file.
040793	Recovered unacknowledged messages were not being marked as Redelivered. This has been fixed.
041617	The `JMSMessageID` converted using the `JMSHelper` no longer adds an extra zero at the end.
041691	Fixed a bug in JMS that causes Message-Driven-Beans to hang when the trans-attribute flag in the ejb-jar.xml file is set as Required.
042461	When sending a message from WebLogic Server JMS classes through a IBM MQ JMS sender and the correlation ID for the message is null, the IBM code was calling WebLogic with `setJMSCorrelationIDAsBytes()`, which is not supported. We now convert the bytes to a UTF8 string and store it in the correlation id in order to handle `setJMSCorrelationIDAsBytes`.

042484	When trying to create an ObjectMessage with an object that was not in the classpath and within a bean, JMS was throwing `ClassNotFoundException`. This has been fixed.

Plug-in Issues

Change Request Number	Description
038831	The Apache plug-in was failing under heavy load. This has been fixed.
040817	When using IIS to browse and call a regular JSP page that calls a Java Bean and getting a ResultSet reference, there were problems when submitting through the FORM POST method. The HTML stream was getting truncated when it came back to IIS--in other words, the page was only partially painted. This has been fixed.
041332	Using plug-ins, `MatchExpression` can now contain multiple, comma-separtated expressions.
041534	Using the `mod_wl.so`, Apache was not proxying without `Debug` and `DebugConfigInfo` in the `httpd.conf`. This has been fixed.
041580	The Apache plug-in `MatchExpression` now works if an encoded URL is used.
041745	Added a log message when the trusted CA file isn't specified and SSL is used with the Isapi plug-in.
041754	Fixed an infinite loop when ISAPI was configured for SSL but using a non-secure port.
041939	The Apache plug-in wasn't filtering out the header `"transfer-encoding"="chunked"`. So the browser was failing to handle these responses, causing Netscape to hang and IE to display an error page. This has been fixed.
042482	The NSAPI and Apache plugins were having a READ timeout on HPUX-11. These problems have been fixed.

Servlets and JSP Issues

Change Request Number	Description

026488	Fixed a problem with servlets when the parameter to method `println(String)` was null.
029989	The wrong error message was sometimes being sent when an `UnavailableException` was raised within a servlet.

030880	Fixed problems compiling JSPs when debugging was turned on. Now you can specify compiler flags in weblogic.xml. For example:
	`<!-- weblogic.xml: weblogic-specific web app`
	`descriptor --> <weblogic-web-app>`
	`<session-descriptor>`
	`</session-descriptor>`
	`<jsp-descriptor>`
	`<jsp-param>`
	`<param-name>compileCommand</param-name>`
	`<param-value>jikes</param-value>`
	`</jsp-param>`
	`<jsp-param>`
	`<param-name>compileFlags</param-name>`
	`<param-value>+E -nowarn</param-value>`
	`</jsp-param>`

041478	There was a change in session cookie format that also fixed some problems with URL rewriting.
037771	Optimized the checking of JSP’s for reloading.
038739	Improved support for JSP debugging.
039083	The `RequestDispatcher` and forwarding a JSP page now works correctly.
039193	When users forgot to make their servlet class public, a vague and unhelpful error was being thrown. This no longer happens.
039530	Fixed our internal servlet tags to reset their state on release. If you used the same tag more than once on a page the additional tags kept the stale data from the previous tag.
040171	The attribute `SingleThreadedServletPoolSize` was not settable through the Administration Console. This has been fixed.
040338	Some servlets were functioning, but were not showing up in the mydomain>WebApplications>DefaultWebApp_myserver>ActiveServlets window of the console. This has been fixed.

040662	When JSP files had a page tag, the JSP compiler was not generating a class, or java file. This happened only with command line `jspc`. This has been fixed.
040766	Fixed a problem in servlets with using ../ for a relative URL when doing a `response.sendRedirect`.
040814	Added more informative error messages for instances when user errors cause exceptions using servlets.
040843	Fixed problems with deploying `WebAppComponents` using virtual hosts.
040981	Fixed problems associated with WAP and session cookies using the pipe characters.
041234	Fixed a problem in JSP/Servlets using POST and HTTPS.
041605	There was a problem with JSPs where the compiler error page displayed a corrupt error message from javac. This has been fixed.
041684	Fixed a problem in servlets. The XSLT tag threw a SAXParseException while proceesing XML content.
041729	The `weblogic.servlet.jsp.Precompiler` didn't honor `weblogic.xml` parameters, like `workingdir`, `packagePrefix`, etc, for the JSPs that it precompiles. This has been fixed.
041945	When the web.xml had the setting `inputCharset./A*=Shift_JIS` `inputCharset./B*=EUC-JP`, and two clients simultaneously accessed the JSP/Servlet, WebLogic failed to handle these string correctly. It was possible to get the correct result from one client, but not from the other. This has been fixed.
042031	In the webapp data table, when you click the monitoring icon for any webapp and then click on the monitor link in the Servlets column, the correct heading is now displayed.
042293	Servlet URL rewriting was working fine with IE 5.0 but wasn’t working with Netscape 4.7. This has been fixed.
042509	Fixed some problems when forwarding from one web application to another web application and using some form of authentication in both. Some session data was incorrectly getting shared across the applications

XML Issues

Change Request Number	Description
039096	You no longer have to restart WebLogic Server to make it aware of a newly added entry to the XML registry.
039247	Custom parsers are now correctly invoked when an XML document that has been configured to be parsed by the custom parser is parsed using the `setAttribute()` method of the Servlet API.
041734	WebLogic's JAXP implementation no longer requires that the input stream that contains an XML document that is about to be delegated to a particular parser for parsing be resettable.
041778	A `NullPointerException` error is no longer thrown when an XML document that contains an external entity that has been correctly registered in the XML registry is parsed.
041797	After creating a new server, the server was failing to reboot and the `config.xml` had some incorrectly parsed headers. These problems have been fixed.

Miscellaneous Issues

Change Request Number	Description
035039	When WebLogic needs to prompt for the boot password, first it should log (in security.xml) that it is prompting. This would allow an administrator on a remote machine to know why the machine is apparently "hung". This now happens.
036141	A managed server was not starting if `guest` had been disabled (in order to do a JNDI lookup in an ACL). This has been fixed.
036604	The method `ClusterMBean.setDefaultLoadAlgorithm()` no longer accepts illegal values and throws the proper `IllegalArgumentException.`

036986	Using Weblogic's XA driver with CMP was failing at deployment time. This has been fixed.
038883	Specifying the root directory when starting the server was not working when the root directory was not under a `/config` directory.
039066	If you entered an incorrect password when prompted at system startup, the server just tried to start anyway. The startup eventually fails. Now we reprompt for the password when the password is wrong.
039182	When binding two clusterable objects implementing the same interface to the same JNDI name on a single server, an exception was not being thrown and the first clusterable object was being bound. This has been fixed.
039249	A managed server had a memory leak when it monitored JMSServer using the Administration Console. This has been fixed.
039361	If you used the rdbms realm and created a group, you could not delete the group. This has been fixed.
039891	Fixed a problem with the server not setting the context attribute correctly during servlet initialization.
039960	Log Viewer is now working in Non-US Locales.
039995	A server startup error no longer occurs after a WLEC Connection pool was created using the Administration Console.
040168	The logging example for internationalization failed when a dot (`.`) occurred in the path. This has been fixed.
040177	Additional user information was required when converting `weblogic.properties` files for use with WebLogic Server 6.0. This has now been added to the product documentation.
040178	When you were using the conversion utility, if a `weblogic.properties` file included the property `weblogic.httpd.register.`*`Name`*`=weblogic.servlet.FileServlet` and the servlet did not have initial arguments associated with it (it does not have the property `weblogic.httpd.initArgs.file=`*`arg1`*`=`*`value`* defined in the same `weblogic.properties` file), then the conversion utility was throwing a Null pointer exception. This has been fixed.

040213	The listenport of the managed server cannot be the same as the administrative listenport. You cannot, for example, use localhost or 127.0.0.1, with the 7001 host number on both the managed and administrative servers. Added a more descriptive and informative error message when this occurs.
040297	Fixed a bug that was causing attributes set in `application.xml` not to propagate from administration servers to managed servers.
040405	The `config.xml` file is now serialized using the Apache XML serialize classes. This results in the correct XML, including escapes for built-in entities and an XML header with encoding specified. The encoding used is the platform default. This also fixes problems with the "&" character.
040659	When specifying `getParameter()` encoding as iso-8859-1, the POST method is now working properly.
040688	Targeting a mail session to a server that is part of a cluster, the following exception was being thrown since the mailsession was not `serializable: java.io.NotSerializableException: javax.mail.Session`. This has been fixed.
040899	Fixed a problem with a start-up class when the converter utility converted startup `weblogic.properties` into `.xml`.
040950	Fixed a problem in JTA. For XA connections, the Connection Pool test connection logic was attempting to start a distributed transaction before executing the testing SQL. This had the potential to create a `javax.transaction.NotSupported` exception.
041157	The dialog tabs in the Console were not working correctly when the Mbean name contained Kanji characters.
041290	If the username property is set from the `weblogic.Admin` or `weblogic.Server` command lines, the username must be the system password. Note however, that the username syntax is retained to prevent breaking a user's scripts. The `weblogic.management.password` property value must be the system password and the username may not be modified in the `config.xml` file.

041662	To improve JTA, we now advertise resources on JNDI. Resources were dynamically discovered as transactions go along, but now it is also possible to find out which resources are currently being supported at which servers.
041757	Fixed a problem with JTA. Resource advertisenment (JNDIAdvertiser) was incorrectly done under the caller's identity. We now use the system identity while doing this operation.
041845	Updated the NTRealm to generally improve usabilty and functionality.
041854	To help support Java WebStart with WebLogic Server 6.0, we now provide a default Web Application with the `web.xml` configured for this mime type. We have also provided other mime type mappings.
041995	Fixed an Unexpected Error being received in the console when resuming an active server. Also deleted some extraneous informational messages in the console when adding users.
042183	When the `config.dtd` was being generated in the console it was appearing with some errors. This has been fixed.
043388	Clusters of WebLogic Servers no longer require the use of a shared network drive. Both the WebLogic Server installations and your applications can now reside on local file systems.