Oracle® WebLogic Server

Domain Template Reference

10*g* Release 3 (10.1.3)

August 2008

Oracle WebLogic Server Domain Template Reference, 10*g* Release 3 (10.1.3)

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this software or related documentation is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

This software is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications which may create a risk of personal injury. If you use this software in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure the safe use of this software. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software in dangerous applications.

This software and documentation may provide access to or information on content, products and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Contents

Domain Template Reference

Domain Template Reference

This document provides general information about templates. This document discusses the following topics:

- Types of Templates
- Location of Installed Templates
- Template Tools
- Template Summary
- Relationships Between Templates
- Files Typically Included in a Template
- Basic WebLogic Server Domain Template
- WebLogic Advanced Web Services Extension Template
- Avitek Medical Records Sample Domain Template
- Oracle Workshop for WebLogic Extension Template
- WebLogic Server Default Domain Extension Template
- WebLogic Server Examples Extension Template

Types of Templates

The term *template* refers to a Java Archive (JAR) file that contains the files and scripts required to create or extend a domain. The types of template include:

- *Domain template*—defines the full set of resources within a domain, including infrastructure components, applications, services, security options, and general environment and operating system options.

  The product installation includes a predefined Basic WebLogic Server Domain template. This template defines the core set of resources within a domain, including an Administration Server and basic configuration information. For more information on Basic WebLogic Server Domain template, see "Basic WebLogic Server Domain Template" on page 1-9.

  You can also create a custom domain template from an existing domain by using the Domain Template Builder or the `pack` command. By using the Domain Template Builder, you can also create a custom domain template from an existing template.

- *Extension template*—defines the applications and services that you can add to an existing domain, including product component functionality and resources such as JDBC or JMS.

  The product installation includes several predefined extension templates. For a summary of extension templates, see Template Summary.

  You can also create a custom extension template from an existing domain or template using the Domain Template Builder.

- *Managed Server template* – defines the subset of resources within a domain that are required to create a Managed Server domain directory on a remote machine. You can create a custom Managed Server template by using the `pack` command. Complete details are provided in *Creating Templates and Domains Using the pack and unpack Commands*.

Location of Installed Templates

The following table identifies the location of the predefined templates provided with your product installation, where *BEA_HOME* represents the product installation directory.

Table 1 Location of Templates

Type of Template	Directory Location
Domain	`BEA_HOME\common\templates\domains`
Extension	`BEA_HOME\common\templates\applications`

Template Tools

The following table identifies the tools with which you can create templates and the tools with which you can use templates to create or extend a domain.

Table 2 Template Tools

To . . .	Use this tool . . .
Create a new domain	• Configuration Wizard • WLST Offline • `unpack` command
Extend an existing domain	• Configuration Wizard • WLST Offline
Create a new Managed Server domain on a remote machine	`unpack` command
Create a domain template	• Domain Template Builder • `pack` command • WLST Offline
Create an extension template	Domain Template Builder
Create a Managed Server template	`pack` command

Note: All the tools used to create or extend a domain leverage a common underlying infrastructure, referred to as the *Configuration Wizard framework*.

- For information about using the Configuration Wizard, see *Creating WebLogic Domains Using the Configuration Wizard*.
- For information about using the WLST Offline, see *WebLogic Scripting Tool*.
- For information about using the `pack/unpack` commands, see in *Creating Templates and Domains Using the pack and unpack Commands.*
- For information about using the Domain Template Builder, see *Creating Templates Using the Domain Template Builder*.

Template Summary

The following tables summarizes the predefined templates that may be provided in your product installation.

Table 3 Summary of WebLogic Server and Oracle Workshop for WebLogic Platform Templates

Template	Filename	Description
Domain Template		
Basic WebLogic Server Domain Template	`wls.jar`	Creates a base WebLogic Server domain.
Extension Templates		
WebLogic Advanced Web Services Extension Template	`wls_webservice.jar`	Extends an existing WebLogic Server domain to add functionality required for advanced Web Services, including WSRM, Buffering, and JMS Transport.
Avitek Medical Records Sample Domain Template	`medrec.jar`	Extends the Basic WebLogic Server domain to create the Avitek Medical Records sample domain. This domain is a WebLogic Server sample application suite that demonstrates all aspects of the J2EE platform.
Oracle Workshop for WebLogic Extension Template	`workshop_wl.jar`	Extends the Basic WebLogic Server domain to create Oracle Workshop for WebLogic domain.

Table 3 Summary of WebLogic Server and Oracle Workshop for WebLogic Platform Templates (Continued)

Template	Filename	Description
WebLogic Server Default Domain Extension Template	`wls_default.jar`	Extends the Basic WebLogic Server domain with a web application designed to guide new users through an introduction to WebLogic Server. When running the web application, users can review informative content on various topics, including highlights of WebLogic Server functionality. From the web application, users can also run several preconfigured, precompiled examples. Resources from this extension template are required for a WebLogic Server Examples domain.
WebLogic Server Examples Extension Template	`wls_examples.jar`	Extends the WebLogic Server domain containing resources from the base WebLogic Server domain template and the WebLogic Server Default Domain extension template to create a complete WebLogic Server Examples domain. The WebLogic Server Examples domain contains a collection of examples that illustrate best practices for coding individual J2EE and WebLogic Server APIs.

Relationships Between Templates

This section discusses the following topics:

- "WebLogic Server Resources as a Prerequisite" on page 5
- "Relationships Between Templates" on page 6

WebLogic Server Resources as a Prerequisite

WebLogic Server resources must be set up in your domain before you can add resources from an extension template. When you select an extension template, the Configuration Wizard framework checks to make sure the required resources are available for you.

Relationships Between Templates

You can create a base WebLogic domain by using the predefined Basic WebLogic Server domain template, or you can create a Basic WebLogic domain and extend it incrementally using the extension templates. The following table shows the relationships between the templates and the domains created.

Table 4 Relationships Between Templates

This type of domain . . .	Requires resources from these templates . . .
Avitek Medical Records Sample	Basic WebLogic Server Domain template, `wls.jar` + Avitek Medical Records Sample Domain extension template, `medrec.jar`
WebLogic Server (base)	Basic WebLogic Server Domain template, `wls.jar`
WebLogic Server Default	Basic WebLogic Server Domain template, `wls.jar` + WebLogic Server Default Domain extension template, `wls_default.jar`
WebLogic Server Examples	Basic WebLogic Server Domain template, `wls.jar` + WebLogic Server Default Domain extension template, `wls_default.jar` + WebLogic Server Examples extension template, `wls_examples.jar`
WebLogic Advanced Web Services Extension	Basic WebLogic Server Domain, `wls.jar` + WebLogic Advanced Web Services Extension, `wls_webservice.jar`
Oracle Workshop for WebLogic Platform	Basic WebLogic Server Domain, `wls.jar` + Advanced Web Services Extension, `wls_webservice.jar` + Oracle Workshop for WebLogic Extension, `workshop_wl.jar`

Files Typically Included in a Template

The basic files included in any template are `config.xml` and `template-info.xml`. There are additional files in the predefined templates, and a domain is created or extended based on these files. The following table describes the files typically included in a domain or extension template.

Table 5 Files Included in a Template

Filename	Description
product component files	Various files used to complete the domain setup for a specific Oracle product component. Such files may provide information for security and default database settings.
`*-jdbc.xml`	Sets up or extends a domain with JDBC system resources required by a product component. In a template, the `*-jdbc.xml` files must be located in the `config\jdbc` directory.
`*-jms.xml`	Sets up or extends a domain with JMS system resources required by a product component. In a template, the `*-jms.xml` files must be located in the `config\jms` directory.
`clusters.script`	Used to modify the Configuration Wizard framework's default auto-configuration of a cluster. By default, resources are targeted to the cluster. You can `unassign` a resource from the cluster and then `assign` it to another component. To specify a target, you can use the following replacement variables: • `%AManagedServer%` — Any Managed Server • `%AllManagedServers%` — Comma-separated list of all Managed Servers • `%AdminServer%` — Administration Server name • `%Cluster%` — Cluster name • `%ProxyServer%` — Proxy server name • `%HTTPProxyApp%` — http proxy application definition Note the following additional considerations: • You must use the name attribute of an object that is to be replaced. • You can use an asterisk (*) as a wildcard for "All." In a template, the `clusters.script` file must be located in the `script` directory. **Note:** AquaLogic Service Bus overrides this behavior. For more information on the AquaLogic Service Bus Extension template, see the AquaLogic Service Bus Deployment Resources in the *Deployment Guide*.
`config.xml`	Sets up or extends the domain configuration. In a template, the `config.xml` file must be located in the `config` directory.

Table 5 Files Included in a Template (Continued)

Filename	Description
`jdbc.index`	Identifies the locations of SQL scripts used to set up a database. The file lists the scripts in the order in which they must be run. If the scripts are not contained in the template, but are located in the product installation directory, that directory can be represented by a tilde (~) in the pathname for the scripts, as shown in the following example: `~/integration/common/dbscripts/oracle/reporting_runtime.sql` Specifically, the tilde represents the directory path identified by the `$USER_INSTALL_DIR$` variable in the `stringsubs.xml` file. In a template, a `jdbc.index` file must be located in the `_jdbc_\`*`dbtype`*`\`*`dbversion`* directory, where *`dbtype`* is the type of database, such as Oracle, and *`dbversion`* is the database version, such as 9i.
`security.xml`	Used to create user groups and roles that establish identity and access to domain resources. You can create the default Admin user only through the `security.xml` in a *domain* template. However, you can create user groups and roles through the `security.xml` included in either a domain or an extension template.
`startmenu.xml`	Used to create Windows start menu entries.
`startscript.xml`	Used to create the `*.cmd` and `*.sh` files that are placed into a domain's root and `bin` directories.
`stringsubs.xml`	Identifies string substitution values and files that will receive string substitutions during domain creation or extension. The files that will receive string substitutions must already be prepared with replacement variables. During domain creation or extension, the Configuration Wizard framework runs macros to replace variables with the appropriate string substitution, using information from *`BEA_HOME`*`\common\lib\macrorules.xml`, where *`BEA_HOME`* is the product installation directory.
`template-info.xml`	Provides template identification information, such as the template name, software version, type of template (domain or application), author, description, and so on.

Basic WebLogic Server Domain Template

Your product installation provides one predefined Basic WebLogic Server domain template. All other predefined templates are extension templates that you may use to add resources, services, and applications to a Basic WebLogic Server domain. You can easily create or extend a domain by using these predefined templates with the Configuration Wizard or WLST.

Generated Domain Output

The Basic WebLogic Server Domain template allows you to create a simple WebLogic Server domain. By default, when using the Basic WebLogic Server Domain template, you generate a domain that contains only the required components: an Administration Server and a single administrative user. Any required applications must be created and configured within the domain.

The following table defines the default directory structure and files generated by the Basic WebLogic Server Domain template. Unless otherwise specified, by default, the Configuration Wizard framework creates the domain in the `*BEA_HOME*\user_projects\domains\base_domain` directory. If you modify the default configuration settings, the output directory structure may be different from the structure described here.

Table 6 Output Generated from the Basic WebLogic Server Domain Template

Directory	File	Description
`user_projects\applications\base_domain\`		
	n.a.	Directory designated as the repository for any custom application files that you create.
`user_projects\domains\base_domain\`		
	`fileRealm.properties`	File containing ACLs, users, and groups that can be used for the default security realm when Compatibility security is used.
	`startWebLogic.cmd` `startWebLogic.sh`	Scripts used to start the Administration Server on Windows and UNIX systems, respectively.
`autodeploy\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder for automatic deployments.

Table 6 Output Generated from the Basic WebLogic Server Domain Template (Continued)

Directory	File	Description
bin\	setDomainEnv.cmd setDomainEnv.sh	Scripts used to set up the development environment on Windows and UNIX systems, respectively.
	startManagedWebLogic.cmd startManagedWebLogic.sh	Scripts used to start a Managed Server on Windows and UNIX systems, respectively.
	startPointBaseConsole.cmd startPointBaseConsole.sh	Scripts used to start the PointBase console on Windows and UNIX systems, respectively.
	startWebLogic.cmd startWebLogic.sh	Scripts used to start the Administration Server on Windows and UNIX systems, respectively.
	stopManagedWebLogic.cmd stopManagedWebLogic.sh	Scripts used to stop a Managed Server on Windows and UNIX systems, respectively.
	stopWebLogic.cmd stopWebLogic.sh	Scripts used to stop the Administration Server on Windows and UNIX systems, respectively.
config\	config.xml	File containing the configuration information used by the Administration Server. For more information, see Domain Configuration Files in *Understanding Domain Configuration*.
config\deployments\	readme.txt	File providing information about the directory, which initially serves as a placeholder, and is later used for staging an application when the application's staging mode is "staged."
config\diagnostics\	readme.txt	File providing information about the directory, which initially serves as a placeholder, and is later used for storing the system modules associated with instrumentation in the WebLogic Diagnostic Framework (WLDF).

Table 6 Output Generated from the Basic WebLogic Server Domain Template (Continued)

Directory	File	Description
`config\jdbc\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder, and is later used for storing global JDBC modules that can be configured directly from JMX (as opposed to JSR-88).
`config\jms\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder, and is later used for storing global JMS modules that can be configured directly from JMX (as opposed to JSR-88).
`config\lib\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder, and is later used for storing JAR files that are added to the system classpath of the server when the server's Java virtual machine starts.
`config\nodemanager\`	`nm_password.properties`	File containing Node Manager password property values.
`config\security\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder, and is later used for storing system modules for the security framework. The directory contains one security provider configuration extension for each type of security provider in the domain's current realm.
`config\startup\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder, and is later used for storing system modules that contain startup plans. Startup plans are used to generate shell scripts that can be used as part of server startup.
`console-ext\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder for custom extensions to the WebLogic Server Administration Console.

Table 6 Output Generated from the Basic WebLogic Server Domain Template (Continued)

Directory	File	Description
`init-info\`	`domain-info.xml`	File used to identify domain creation and extension information. Such information includes the identity of the components in the domain, the location of the JDK and applications directory used by the domain, and the templates used to create and extend the domain.
	`security.xml`	File used for creating user groups and roles that establish identity and access to domain resources.
	`startscript.xml`	File used to create the `*.cmd` and `*.sh` files that are placed into the domain's root and `bin` directories.
	`tokenValue.properties`	File that contains the actual values to substitute for the tokens specified in the start scripts.
`lib\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder for the domain's libraries. The JAR files in this directory are added dynamically to the end of the server classpath at server startup.

`security\`	`DefaultAuthenticatorInit.ldift`
	`DefaultRoleMapperInit.ldift`
	`XACMLRoleMapperInit.ldift`
		Files used for bootstrapping tasks, including authentication (user and group), authorization, and role mapping. These files contain LDAP-specific information.
		Note: WebLogic domains created with this release use the XACML providers, by default. These XACML security providers are compatible with policies and roles created using the WebLogic Authorization provider (DefaultAuthorizer) and WebLogic Role Mapping provider (DefaultRoleMapper). For more information, see WebLogic Security Providers in *Understanding WebLogic Security*.

	`SerializedSystemIni.dat`	File containing encrypted security information.

Table 6 Output Generated from the Basic WebLogic Server Domain Template (Continued)

Directory	File	Description
`servers\AdminServer\security\`	`boot.properties`	File containing server startup properties, including the user name and password required to boot the server (in encrypted format). It is generated only when you select development startup mode. This file enables you to bypass the prompt for user name and password during a server's startup cycle. For more information, see Provide User Credentials to Start and Stop Servers in *Managing Server Startup and Shutdown*.
`user_staged_config\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder for configuration information optionally staged by an administrator to be copied to managed servers in the domain.

Resources and Services Configured for WebLogic Server Domain Template

The following table identifies the resources and services configured in a domain created with the Basic WebLogic Server Domain template.

Table 7 Resources Configured in a Basic WebLogic Server Domain

Resource Type	Name	Notes

Administration Server	`AdminServer`	When using the Configuration Wizard or WLST Offline to create a new domain, and you want the Administration Server name to be different from the default name, `AdminServer`, you must configure the name manually. You cannot change the name later when applying an extension template.
		For information about customizing the Administration Server name while creating a domain with the Configuration Wizard, see *Creating WebLogic Domains Using the Configuration Wizard*.
		For information about customizing the Administration Server name while creating a domain with WLST Offline, see “Creating and Configuring WebLogic Domains Using WLST Offline” in *WebLogic Scripting Tool*.
		The following sample WLST Offline code snippet shows how to change the default Administration Server name, `AdminServer`, to `MedRecServer`.
		`#-----------------------------------`
		`#Read the Basic WebLogic Server Domain template`
		`readTemplate('d:/bea/wlserver_10.3/common/templates/domains/wls.jar')`
		`#Change the Administration Server name.`
		`cd('Servers/AdminServer')`
		`set('Name', 'MedRecServer')`
		`#-----------------------------------`

Security realm	`myrealm`	n.a.

WebLogic Advanced Web Services Extension Template

By using the Configuration Wizard or WLST, you can easily extend a base WebLogic Server domain to include the resources required for advanced Web services. You accomplish this by adding the resources and services provided in the WebLogic Advanced Web Services extension template to a base WebLogic Server domain.

Generated Domain Output

The following table defines the default directory structure and files generated after applying the WebLogic Advanced Web Services extension template to a base WebLogic Server domain. Unless otherwise specified, by default, the Configuration Wizard creates the domain in the *BEA_HOME*\user_projects\domains\base_domain directory. If you modify the default configuration settings, the output directory structure may be different from the structure described here.

Table 8 Base Domain After Applying the WebLogic Advanced Web Services Extension Template

Directory	File	Description
user_projects\applications\base_domain\		
	n.a.	Directory serving as a placeholder for any custom application files that you create.
user_projects\domains\base_domain\		
	fileRealm.properties	File containing ACLs, users, and groups that can be used for the default security realm when Compatibility security is used.
	pointbase.ini	File containing initialization information for a PointBase JDBC database.
	startWebLogic.cmd startWebLogic.sh	Scripts used to start the Administration Server on Windows and UNIX systems, respectively.
	URLs.dat	File containing the URL for the JDBC database.
autodeploy\	readme.txt	File providing information about the directory, which initially serves as a placeholder for automatic deployments.

Table 8 Base Domain After Applying the WebLogic Advanced Web Services Extension Template (Continued)

Directory	File	Description
`bin\`	`setDomainEnv.cmd` `setDomainEnv.sh`	Scripts used to set up the development environment on Windows and UNIX systems, respectively.
	`startManagedWebLogic.cmd` `startManagedWebLogic.sh`	Scripts used to start a Managed Server on Windows and UNIX systems, respectively.
	`startPointBaseConsole.cmd` `startPointBaseConsole.sh`	Scripts used to start the PointBase console on Windows and UNIX systems, respectively.
	`startWebLogic.cmd` `startWebLogic.sh`	Scripts used to start the Administration Server on Windows and UNIX systems, respectively.
	`stopManagedWebLogic.cmd` `stopManagedWebLogic.sh`	Scripts used to stop a Managed Server on Windows and UNIX systems, respectively.
	`stopWebLogic.cmd` `stopWebLogic.sh`	Scripts used to stop the Administration Server on Windows and UNIX systems, respectively.
`config\`	`config.xml`	File containing the configuration information used by the "Domain Configuration Files" Administration Server. For more information, see in *Understanding Domain Configuration*.
`config\deployments\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder, and is later used for staging an application when the application's staging mode is "staged."
`config\diagnostics\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder, and is later used for storing the system modules associated with instrumentation in the WebLogic Diagnostic Framework (WLDF).

Table 8 Base Domain After Applying the WebLogic Advanced Web Services Extension Template (Continued)

Directory	File	Description
config\jms\	readme.txt	File providing information about the directory, which initially serves as a placeholder, and is later used for storing global JMS modules that can be configured directly from JMX (as opposed to JSR-88).
	wseejmsmodule-jms.xml	Global JMS module for the domain configured for advanced Web Services.
config\lib\	readme.txt	File providing information about the directory, which initially serves as a placeholder, and is later used for storing JAR files that are added to the system classpath of the server when the server's Java virtual machine starts.
config\nodemanager\	nm_password.properties	File containing Node Manager password property values.
config\security\	readme.txt	File providing information about the directory, which initially serves as a placeholder, and is later used for storing system modules for the security framework. The directory contains one security provider configuration extension for each type of security provider in the domain's current realm.
config\startup\	readme.txt	File providing information about the directory, which initially serves as a placeholder, and is later used for storing system modules that contain startup plans. Startup plans are used to generate shell scripts that can be used as part of server startup.
console-ext\	readme.txt	File providing information about the directory, which initially serves as a placeholder for custom extensions to the WebLogic Server Administration Console.

Table 8 Base Domain After Applying the WebLogic Advanced Web Services Extension Template (Continued)

Directory	File	Description
`init-info\`	`domain-info.xml`	File used to identify domain creation and extension information. Such information includes the identity of the components in the domain, the location of the JDK and applications directory used by the domain, and the templates used to create and extend the domain.
	`security.xml`	File used for creating user groups and roles that establish identity and access to domain resources.
	`startscript.xml`	File used to create the `*.cmd` and `*.sh` files that are placed into the domain's root and `bin` directories.
	`tokenValue.properties`	File that contains the actual values to substitute for the tokens specified in the start scripts.
`lib\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder for the domain's libraries. The JAR files in this directory are added dynamically to the end of the server classpath at server startup.

Table 8 Base Domain After Applying the WebLogic Advanced Web Services Extension Template (Continued)

Directory	File	Description

`security\`	`DefaultAuthenticatorInit.ldift`
	`DefaultRoleMapperInit.ldift`
	`XACMLRoleMapperInit.ldift`
		Files used for bootstrapping tasks, including authentication (user and group), authorization, and role mapping. These files contain LDAP-specific information.
		Note: WebLogic domains created with this release use the XACML providers by default. These XACML security providers are compatible with policies and roles created using the WebLogic Authorization provider (DefaultAuthorizer) and WebLogic Role Mapping provider (DefaultRoleMapper). For more information, see WebLogic Security Providers in Understanding WebLogic Security at http://e-docs.bea.com/wls/docs103/secintro/archtect.html#archtect_0111

	`SerializedSystemIni.dat`	File containing encrypted security information.

Table 8 Base Domain After Applying the WebLogic Advanced Web Services Extension Template (Continued)

Directory	File	Description
`servers\AdminServer\security\`	`boot.properties`	File containing server startup properties, including the user name and password required to boot the server (in encrypted format). It is generated only when you select development startup mode. This file enables you to bypass the prompt for user name and password during a server's startup cycle. For more information, see "Provide User Credentials to Start and Stop Servers" in Starting and Stopping Servers in Managing Server Startup and Shutdown at http://e-docs.bea.com/wls/docs103/server_start/overview.html.
`user_staged_config\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder for configuration information optionally staged by an administrator to be copied to managed servers in the domain.

Resources and Services Configured

The following table identifies the resources and services configured in a domain extended with the WebLogic Advanced Web Services extension template.

Table 9 Resources Configured in a WebLogic Advanced Web Services Domain

Resource Type	Name	Extension Result
Administration Server	`AdminServer`	Uses the Administration Server provided in the base WebLogic Server domain. The default name is `AdminServer`, unless changed during domain creation. The Administration Server referenced in the extension template is `cgServer`. For information about naming the Administration Server during domain creation, see "Resources and Services Configured for WebLogic Server Domain Template" on page 14.
JMS Queues	`WseeMessageQueue`	Adds the JMS queue to the JMS server, `WseeJmsServer`.
	`WseeCallbackQueue`	Adds the JMS queue to the JMS server, `WseeJmsServer`.
JMS Server	`WseeJmsServer`	Adds the JMS server as a system resource and targets it to the Administration Server, `AdminServer`.
Security realm	`myrealm`	Uses the security realm provided by the base WebLogic Server domain.

Avitek Medical Records Sample Domain Template

By using the Configuration Wizard or WLST, you can easily extend a base WebLogic Server domain to create an Avitek Medical Records Sample domain. You accomplish this by adding the resources and services provided in the Avitek Medical Records Sample domain extension template to a base WebLogic Server domain.

For more information about the Avitek Medical Records sample application, see Sample Application Examples and Tutorials for WebLogic Server 10.3.

Generated Domain Output

The following table defines the default directory structure and files generated after applying the Avitek Medical Records Sample Domain extension template to a base WebLogic Server domain. Unless otherwise specified, by default, the Configuration Wizard creates the domain in the *BEA_HOME*\user_projects\domains\base_domain directory. If you modify the default configuration settings, the output directory structure may be different from the structure described here.

Table 10 Base Domain After Applying the Avitek Medical Records Sample Extension Template

Directory	File	Description
user_projects\applications\base_domain\		
build\	Various	Includes Avitek Medical Records split directory deployments.
console-extension\	Various	Includes sub-directories containing various files used to demonstrate extending the WebLogic Server Administration Console with a different look and feel.
dist\	Various	Includes sub-directories containing various files of the Avitek Medical Records applications in an exploded (unarchived) directory format.
doc\	Various	Directory and files containing the Avitek Medical Records online documentation.
lib\	Various	Includes sub-directories containing library files supporting the Avitek Medical Records sample.
setup\	build.xml	Ant build file used with corresponding scripts to set up a database for the Avitek Medical Records sample.
setup\db\	medrec_mysql.ddl medrec_mysql_data.sql medrec_oracle.ddl medrec_oracle_data.sql medrec_pointbase.ddl medrec_pointbase_data.sql	SQL scripts used to set up different databases that can be used with the Avitek Medical Records sample.
src\	Various	Includes sub-directories containing Avitek Medical Records source code including various Java, XML, JSP, HTML files, and so on.
user_projects\domains\base_domain\		

Table 10 Base Domain After Applying the Avitek Medical Records Sample Extension Template (Continued)

Directory	File	Description
	`democa.pem`	Provides sample SSL protocol support for servers in the domain.
	`fileRealm.properties`	File containing ACLs, users, and groups that can be used for the default security realm when Compatibility security is used.
	`log4jConfig.xml`	Configures Avitek Medical Records Log4j implementation including the `MedRecApp.log` file.
	`pointbase.ini`	File containing initialization information for a PointBase JDBC database.
	`startWebLogic.cmd` `startWebLogic.sh`	Scripts used to start the Administration Server on Windows and UNIX systems, respectively.
`autodeploy\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder for automatic deployments.

Table 10 Base Domain After Applying the Avitek Medical Records Sample Extension Template (Continued)

Directory	File	Description
`bin\`	`setDomainEnv.cmd` `setDomainEnv.sh`	Scripts used to set up the development environment on Windows and UNIX systems, respectively.
	`startManagedWebLogic.cmd` `startManagedWebLogic.sh`	Scripts used to start a Managed Server on Windows and UNIX systems, respectively.
	`startPointBaseConsole.cmd` `startPointBaseConsole.sh`	Scripts used to start the PointBase console on Windows and UNIX systems, respectively.
	`startWebLogic.cmd` `startWebLogic.sh`	Scripts used to start the Administration Server on Windows and UNIX systems, respectively.
	`stopManagedWebLogic.cmd` `stopManagedWebLogic.sh`	Scripts used to stop a Managed Server on Windows and UNIX systems, respectively.
	`stopWebLogic.cmd` `stopWebLogic.sh`	Scripts used to stop the Administration Server on Windows and UNIX systems, respectively.
`config\`	`config.xml`	File containing the configuration information used by the Administration Server. For more information, see Domain Configuration Files in *Understanding Domain Configuration*.
`config\deployments\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder, and is later used for staging an application when the application's staging mode is "staged."

Table 10 Base Domain After Applying the Avitek Medical Records Sample Extension Template (Continued)

Directory	File	Description
`config\diagnostics\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder, and is later used for storing the system modules associated with instrumentation in the WebLogic Diagnostic Framework (WLDF).
	`MedRecWLDF.xml`	Diagnostic descriptor information for the Avitek Medical Records diagnostics instrumentation.
`config\jdbc\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder, and is later used for storing global JDBC modules that can be configured directly from JMX (as opposed to JSR-88).
	`MedRecGlobalDataSource-jdbc.xml`	Global non-XA JDBC Data Source module for the Avitek Medical Records domain.
	`MedRecGlobalDataSourceXA-jdbc.xml`	Global XA JDBC Data Source module for the Avitek Medical Records domain.
`config\jms\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder, and is later used for storing global JMS modules that can be configured directly from JMX (as opposed to JSR-88).
	`MedRec-jms.xml`	Global JMS module for the Avitek Medical Records domain.
`config\lib\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder, and is later used for storing JAR files that are added to the system classpath of the server when the server’s Java virtual machine starts.
`config\nodemanager\`	`nm_password.properties`	File containing Node Manager password property values.

Table 10 Base Domain After Applying the Avitek Medical Records Sample Extension Template (Continued)

Directory	File	Description
`config\security\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder, and is later used for storing system modules for the security framework. The directory contains one security provider configuration extension for each type of security provider in the domain's current realm.
`config\startup\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder, and is later used for storing system modules that contain startup plans. Startup plans are used to generate shell scripts that can be used as part of server startup.
`console-ext\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder for custom extensions to the WebLogic Server Administration Console.
`incoming\`	`StJohnHospital.xml`	Location where XML files containing fictitious patient names are uploaded by the Administration application of the Avitek Medical Records sample application.
`init-info\`	`domain-info.xml`	File used to identify domain creation and extension information. Such information includes the identity of the components in the domain, the location of the JDK and applications directory used by the domain, and the templates used to create and extend the domain.
	`security.xml`	File used for creating user groups and roles that establish identity and access to domain resources.
	`startscript.xml`	File used to create the `*.cmd` and `*.sh` files that are placed into the domain's root and `bin` directories.
	`tokenValue.properties`	File that contains the actual values to substitute for the tokens specified in the start scripts.

Table 10 Base Domain After Applying the Avitek Medical Records Sample Extension Template (Continued)

Directory	File	Description
`lib\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder for the domain's libraries. The JAR files in this directory are added dynamically to the end of the server classpath at server startup.
	`log4j.jar` `wllog4j.jar`	Libraries used by the Avitek Medical Records domain.
`medrecWseeFileStore\`		File store used by the Avitek Medical Records application.
`physicianFileStore\`		File store used by the Physician application.
`rmfilestore\`		File store used by system resources.

`security\`	`DefaultAuthenticatorInit.ldift`
	`DefaultAuthorizerInit.ldift`
	`DefaultRoleMapperInit.ldift`
	`XACMLAuthorizerInit.ldift`
	`XACMLRoleMapperInit.ldift`
		Files used for bootstrapping tasks, including authentication (user and group), authorization, and role mapping. These files contain LDAP-specific information.
		Note: WebLogic domains created with this release use the XACML providers, by default. These XACML security providers are compatible with policies and roles created using the WebLogic Authorization provider (DefaultAuthorizer) and WebLogic Role Mapping provider (DefaultRoleMapper). For more information, see WebLogic Security Providers in Understanding WebLogic Security at http://e-docs.bea.com/wls/docs103/secintro/archtect.html#archtect_0111.

	`MedRecDBMSPlugin.jar`	A `CustomDBMSAuthenticatorPlugin` used to validate a user/password against a DBMS for the Avitek Medical Records sample application.
	`SerializedSystemIni.dat`	File containing encrypted security information.

Table 10 Base Domain After Applying the Avitek Medical Records Sample Extension Template (Continued)

Directory	File	Description
`servers\AdminServer\security\`	`boot.properties`	File containing server startup properties, including the user name and password required to boot the server (in encrypted format). It is generated only when you select development startup mode. This file enables you to bypass the prompt for user name and password during a server's startup cycle. For more information, see "Provide User Credentials to Start and Stop Servers" in Starting and Stopping Servers in Managing Server Startup and Shutdown at http://e-docs.bea.com/wls/docs103/server_start/overview.html.
`user_staged_config\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder for configuration information optionally staged by an administrator to be copied to managed servers in the domain.

Resources and Services Configured

The following table identifies the resources and services configured in a domain extended with the Avitek Medical Records Sample extension template.

Table 11 Resources Configured in an Avitek Medical Records Domain

Resource Type	Name	Extension Result
Administration Server	`AdminServer`	Uses the Administration Server provided in the base WebLogic Server domain. The default name is `AdminServer`, unless changed during domain creation. The Administration Server referenced in the extension template is `MedRecServer`. For information about naming the Administration Server during domain creation, see "Resources and Services Configured for WebLogic Server Domain Template" on page 14.
Application Deployments	`InitEAR`	Adds the `InitEAR` Web application and targets it to the Administration Server, `AdminServer`.
	`MedRecEAR`	Adds the `MedRecEAR` Web application and targets it to the Administration Server, `AdminServer`.
	`PhysicianEAR`	Adds the `PhysicianEAR` Web application and targets it to the Administration Server, `AdminServer`.
	`StartBrowserEAR`	Adds the `StartBrowserEAR` Web application and targets it to the Administration Server, `AdminServer`.

Table 11 Resources Configured in an Avitek Medical Records Domain (Continued)

Resource Type	Name	Extension Result
File Stores	`FileStore`	Adds the file store and targets the store to the Administration Server, `AdminServer`.
	`MedRecWseeFileStore`	Adds the file store to be used as the persistent store for the JMS server, `MedRecWseeJMSServer`, and targets the store to the Administration Server, `AdminServer`.
	`PhysicianFileStore`	Adds the file store and targets the store to the Administration Server, `AdminServer`.
JDBC Data Sources	`MedRecGlobalDataSource`	Identifies the JDBC data source as a `MedRecGlobalDataSource` system resource.
	`MedRecGlobalDataSourceXA`	Identifies the JDBC data source as a `MedRecGlobalDataSourceXA` system resource.
JDBC Store	`MedRecJMSJDBCStore`	Adds the JDBC store to be used with the JDBC data source, `MedRecGlobalDataSource`, and as the persistent store for the JMS server, `MedRecJMSServer`, and targets the store to the Administration Server, `AdminServer`.
JDBC System Resources	`MedRecGlobalDataSource` `MedRecGlobalDataSourceXA`	Identifies the JDBC data source and connection pool setups to be used for non-XA and XA JDBC system resources, and targets the resources to the Administration Server, `AdminServer`.
JMS Queues	`weblogic.wsee.reliability.wseeMedRecDestinationQueue`	Adds the JMS queue to the JMS server, `MedRecWseeJMSServer`.

Table 11 Resources Configured in an Avitek Medical Records Domain (Continued)

Resource Type	Name	Extension Result
JMS Servers	`MedRecJMSServer`	Adds the JMS server as a `MedRec-jms` system resource and targets it to the Administration Server, `AdminServer`.
	`MedRecWseeJMSServer`	Adds the JMS server as a `MedRec-jms` system resource and targets it to the Administration Server, `AdminServer`.
JMS System Resources	`MedRec-jms`	Adds the JMS servers, connection factories, and queues to be used as JMS system resources, and targets the resources to the Administration Server, `AdminServer`.
Mail Session	`mail/MedRecMailSession`	Adds the mail session.
SAF Agent	`MedRecSAFAgent`	Adds this store-and-forward agent, which uses the file store, `MedRecWseeFileStore`, and targets it to the Administration Server, `AdminServer`.
Security realm	`myrealm`	Uses the security realm provided in the base WebLogic Server domain.
WLDF System Resource	`MedRecWLDF`	Adds the WLDF system resource and defined WLDF instrumentation monitors for dye injection, and targets them to the Administration Server, `AdminServer`.

Oracle Workshop for WebLogic Extension Template

Using the Configuration Wizard or WLST, you can easily extend a base WebLogic Server domain to include the resources required for using Oracle Workshop for WebLogic Platform. You accomplish this by adding the resources and services provided in the Oracle Workshop for WebLogic Platform template to a base WebLogic Server domain.

Note: Using the Configuration Wizard in graphical mode, you can easily create a new Oracle Workshop for WebLogic Platform domain by checking the Oracle Workshop for WebLogic Platform check box in the **Select Domain Source** window. The result is the same as creating a base WebLogic Server domain first and then extending that domain with both the Oracle Workshop for WebLogic Platform extension template. For more information about the templates required to create a Oracle Workshop for WebLogic Platform domain, see "Relationships Between Templates" on page 5.

Generated Domain Output

The following table defines the default directory structure and files generated after applying the Oracle Workshop for WebLogic Platform template to a base WebLogic Server domain. Unless otherwise specified, by default, the Configuration Wizard creates the domain in the *BEA_HOME*\user_projects\domains\base_domain directory. If you modify the default configuration settings, the output directory structure may be different from the structure described here.

Table 12 Base Domain After Applying the Oracle Workshop for WebLogic Platform Template

Directory	File	Description
user_projects\applications\base_domain\		
	n.a.	Directory serving as a placeholder for any custom application files that you create.
user_projects\domains\base_domain\		
	fileRealm.properties	File containing ACLs, users, and groups that can be used for the default security realm when Compatibility security is used.
	pointbase.ini	File containing initialization information for a PointBase JDBC database.
	startWebLogic.cmd startWebLogic.sh	Scripts used to start the Administration Server on Windows and UNIX systems, respectively.
	URLs.dat	File containing the URL for the JDBC database.
autodeploy\\	readme.txt	File providing information about the directory, which initially serves as a placeholder for automatic deployments.

Table 12 Base Domain After Applying the Oracle Workshop for WebLogic Platform Template (Continued)

Directory	File	Description
bin\	setDomainEnv.cmd setDomainEnv.sh	Scripts used to set up the development environment on Windows and UNIX systems, respectively.
	startManagedWebLogic.cmd startManagedWebLogic.sh	Scripts used to start a Managed Server on Windows and UNIX systems, respectively.
	startPointBaseConsole.cmd startPointBaseConsole.sh	Scripts used to start the PointBase console on Windows and UNIX systems, respectively.
	startWebLogic.cmd startWebLogic.sh	Scripts used to start the Administration Server on Windows and UNIX systems, respectively.
	stopManagedWebLogic.cmd stopManagedWebLogic.sh	Scripts used to stop a Managed Server on Windows and UNIX systems, respectively.
	stopWebLogic.cmd stopWebLogic.sh	Scripts used to stop the Administration Server on Windows and UNIX systems, respectively.
config\	config.xml	File containing the configuration information used by the Administration Server. For more information, see Domain Configuration Files in *Understanding Domain Configuration*.
config\deployments\	readme.txt	File providing information about the directory, which initially serves as a placeholder, and is later used for staging an application when the application's staging mode is "staged."
config\diagnostics\	readme.txt	File providing information about the directory, which initially serves as a placeholder, and is later used for storing the system modules associated with instrumentation in the WebLogic Diagnostic Framework (WLDF).

Table 12 Base Domain After Applying the Oracle Workshop for WebLogic Platform Template (Continued)

Directory	File	Description
`config\jdbc\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder, and is later used for storing global JDBC modules that can be configured directly from JMX (as opposed to JSR-88).
	`cgDataSource-jdbc.xml`	Global XA JDBC Data Source module for the domain configured for advanced Web services.
	`cgDataSource-nonXA-jdbc.xml`	Global non-XA JDBC Data Source module for the domain configured for advanced Web services.
`config\lib\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder, and is later used for storing JAR files that are added to the system classpath of the server when the server's Java virtual machine starts.
`config\nodemanager\`	`nm_password.properties`	File containing Node Manager password property values.
`config\security\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder, and is later used for storing system modules for the security framework. The directory contains one security provider configuration extension for each type of security provider in the domain's current realm.
`config\startup\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder, and is later used for storing system modules that contain startup plans. Startup plans are used to generate shell scripts that can be used as part of server startup.
`console-ext\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder for custom extensions to the WebLogic Server Administration Console.

Table 12 Base Domain After Applying the Oracle Workshop for WebLogic Platform Template (Continued)

Directory	File	Description
`init-info\`	`domain-info.xml`	File used to identify domain creation and extension information. Such information includes the identity of the components in the domain, the location of the JDK and applications directory used by the domain, and the templates used to create and extend the domain.
	`security.xml`	File used for creating user groups and roles that establish identity and access to domain resources.
	`startscript.xml`	File used to create the `*.cmd` and `*.sh` files that are placed into the domain's root and `bin` directories.
	`tokenValue.properties`	File that contains the actual values to substitute for the tokens specified in the start scripts.
`lib\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder for the domain's libraries. The JAR files in this directory are added dynamically to the end of the server classpath at server startup.

Table 12 Base Domain After Applying the Oracle Workshop for WebLogic Platform Template (Continued)

Directory	File	Description

`security\`	`DefaultAuthenticatorInit.ldift`
	`DefaultRoleMapperInit.ldift`
	`XACMLRoleMapperInit.ldift`
		Files used for bootstrapping tasks, including authentication (user and group), authorization, and role mapping. These files contain LDAP-specific information.
		Note: WebLogic domains created with this release use the XACML providers, by default. These XACML security providers are compatible with policies and roles created using the WebLogic Authorization provider (DefaultAuthorizer) and WebLogic Role Mapping provider (DefaultRoleMapper). For more information, see WebLogic Security Providers in Understanding WebLogic Security at http://e-docs.bea.com/wls/docs103/secintro/archtect.html#archtect_0111

	`SerializedSystemIni.dat`	File containing encrypted security information.
`servers\AdminServer\security\`	`boot.properties`	File containing server startup properties, including the user name and password required to boot the server (in encrypted format). It is generated only when you select development startup mode. This file enables you to bypass the prompt for user name and password during a server's startup cycle. For more information, see "Provide User Credentials to Start and Stop Servers" in Starting and Stopping Servers in Managing Server Startup and Shutdown at http://edocs.bea.com/wls/docs103/server_start/overview.html.
`user_staged_config\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder for configuration information optionally staged by an administrator to be copied to managed servers in the domain.

Resources and Services Configured

The following table identifies the resources and services configured in a domain extended with the Oracle Workshop for WebLogic Platform template.

Table 13 Resources Configured in a Oracle Workshop for WebLogic Platform Domain

Resource Type	Name	Extension Result
Administration Server	AdminServer	Uses the Administration Server provided in the base WebLogic Server domain. The default name is AdminServer, unless changed during domain creation. The Administration Server referenced in the extension template is cgServer. For information about naming the Administration Server during domain creation, see "Resources and Services Configured for WebLogic Server Domain Template" on page 14.
JDBC Data Source	cgDataSource	Defines an XA JDBC data source including its associated jdbc connection pool. The data source is named cgDataSource.
	cgDataSource-nonXA	Includes the JDBC data source and connection pool setups defined as cgDataSource in the domain and targets them to the correct server(s).
JDBC Store	cgJMSStore	Uses the JDBC store provided by the Oracle Workshop for WebLogic Platform extension template. The JDBC store is to be used with the JDBC data source, cgDataSource-nonXA, and the JMS server, WseeJmsServer, as a persistent store, and is targeted to the Administration Server, AdminServer.
JDBC System Resources	cgDataSource cgDataSource-nonXA	Identifies the JDBC data source and connection pool setups to be used for JDBC system.
JMS Server	WseeJmsServer	Uses the JMS server provided by the Workshop for WebLogic Platform extension template. Identifies the JMS server as a system resource and targets it to the Administration Server, AdminServer.
Security realm	myrealm	Uses the security realm provided by the base WebLogic Server domain.

Table 13 Resources Configured in a Oracle Workshop for WebLogic Platform Domain (Continued)

Resource Type	Name	Extension Result
Commons-Logging Bridge	`wls-commonslogging-bridge#1.0@1.0`	Hooks commons-logging into the WLS logging mechanism.
Libraries Deployed	`beehive-netui-1.0#1.0@1.0`	Adds the Apache Beehive NetUI Version 1.0 libraries. These libraries support pageflow development, and depend upon the libraries contained in `struts-1.1.war` and `weblogic-beehive-1.0.ear`.
	`jstl#1.1@1.1.2`	Adds the Java standard tagging (JSTL) Version 1.1 libraries.
	`jsf-ri#1.1@1.1.1`	Adds the Java Server Faces Reference Implementation libraries.
	`jsf-myfaces#1.1@1.1.1`	Adds the Apache MyFaces libraries.
	`struts-1.1#1.1@1.0`	Adds the Apache Struts Version 1.1 libraries.
	`struts-1.2#1.2@1.0`	Adds the Apache Struts Version 1.2 libraries.
	`weblogic-controls-10.0#10.0@10.0`	Adds the Oracle Workshop for WebLogic controls extensions, including additional system controls (such as service control and timer control) as well as support for adding transactions, security, and message buffering to existing controls. Packaged for EARs.
	`weblogic-controls-10.0-war#10.0@10.0`	Adds the Oracle Workshop for WebLogic Platform controls extensions including additional system controls (such as service control) as well as support for adding transactions, security, and message buffering to existing controls. Excludes those features which require EAR support such as timer control. Packaged for WARs.
	`beehive-controls-1.0#1.0@1.0`	Adds the Apache Beehive Controls 1.0.1 libraries to the domain. This includes the control runtime as well as the Beehive system controls - JdbcControl, JMSControl, and EJBControl.

WebLogic Server Default Domain Extension Template

Using the Configuration Wizard or WLST, you can easily extend a base WebLogic Server domain to include resources required for a default WebLogic Server domain. You accomplish this by adding the resources and services provided in the WebLogic Server Default Domain extension template to a base WebLogic Server domain.

Note: Applying the WebLogic Server Default Domain extension template to a base WebLogic domain is a prerequisite to using the WebLogic Server Examples extension template. For information about the relationship between templates, see "Relationships Between Templates" on page 5.

For more information about the samples that are supported in the WebLogic Server Examples domain, see *Sample Application Examples and Tutorials for WebLogic Server 10.3*.

Generated Domain Output

The following table defines the default directory structure and files generated after applying the WebLogic Server Default Domain extension template to a base WebLogic Server domain. Unless otherwise specified, by default, the Configuration Wizard creates the domain in the `*BEA_HOME*\user_projects\domains\base_domain` directory. If you modify the default configuration settings, the output directory structure may be different from the structure described here.

Table 14 Base Domain After Applying the WebLogic Server Default Domain Extension Template

Directory	File	Description
`user_projects\applications\base_domain\`		
`server\docs\`	Various	Includes sub-directories containing style sheet and graphics files to support the online documentation.
`server\examples\build\`	Various	Includes WebLogic Server examples deployments.
`server\examples\src\`	Various	Includes source code and instructions for WebLogic Server examples.
`user_projects\domains\base_domain\`		

Table 14 Base Domain After Applying the WebLogic Server Default Domain Extension Template (Continued)

Directory	File	Description
	`fileRealm.properties`	File containing ACLs, users, and groups that can be used for the default security realm when Compatibility security is used.
	`pointbase.ini`	File containing initialization information for a PointBase JDBC database.
	`setExamplesEnv.cmd` `setExamplesEnv.sh`	Scripts that set up the environment to use the WebLogic Server Examples on Windows and UNIX systems, respectively.
	`startWebLogic.cmd` `startWebLogic.sh`	Scripts used to start the Administration Server on Windows and UNIX systems, respectively.
	`startWebLogicEx.cmd` `startWebLogicEx.sh`	Scripts used to start the Administration Server for the WebLogic Server Examples domain on Windows and UNIX systems, respectively.
`autodeploy\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder for automatic deployments.

Table 14 Base Domain After Applying the WebLogic Server Default Domain Extension Template (Continued)

Directory	File	Description
`bin\`	`setDomainEnv.cmd` `setDomainEnv.sh`	Scripts used to set up the development environment on Windows and UNIX systems, respectively.
	`startManagedWebLogic.cmd` `startManagedWebLogic.sh`	Scripts used to start a Managed Server on Windows and UNIX systems, respectively.
	`startPointBaseConsole.cmd` `startPointBaseConsole.sh`	Scripts used to start the PointBase console on Windows and UNIX systems, respectively.
	`startWebLogic.cmd` `startWebLogic.sh`	Scripts used to start the Administration Server on Windows and UNIX systems, respectively.
	`stopManagedWebLogic.cmd` `stopManagedWebLogic.sh`	Scripts used to stop a Managed Server on Windows and UNIX systems, respectively.
	`stopWebLogic.cmd` `stopWebLogic.sh`	Scripts used to stop the Administration Server on Windows and UNIX systems, respectively.
`config\`	`config.xml`	File containing the configuration information used by the Administration Server. For more information, see Domain Configuration Files in *Understanding Domain Configuration*.
`config\deployments\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder, and is later used for staging an application when the application's staging mode is "staged."
`config\diagnostics\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder, and is later used for storing the system modules associated with instrumentation in the WebLogic Diagnostic Framework (WLDF).

Table 14 Base Domain After Applying the WebLogic Server Default Domain Extension Template (Continued)

Directory	File	Description
config\jdbc\	readme.txt	File providing information about the directory, which initially serves as a placeholder, and is later used for storing global JDBC modules that can be configured directly from JMX (as opposed to JSR-88).
	examples-demo-jdbc.xml	Global non-XA JDBC Data Source module for the WebLogic Server default domain.
	examples-demoXA-jdbc.xml	Global XA JDBC Data Source module for the WebLogic Server default domain.
config\jms\	readme.txt	File providing information about the directory, which initially serves as a placeholder, and is later used for storing global JMS modules that can be configured directly from JMX (as opposed to JSR-88).
config\lib\	readme.txt	File providing information about the directory, which initially serves as a placeholder, and is later used for storing JAR files that are added to the system classpath of the server when the server's Java virtual machine starts.
config\nodemanager\	nm_password.properties	File containing Node Manager password property values.
config\security\	readme.txt	File providing information about the directory, which initially serves as a placeholder, and is later used for storing system modules for the security framework. The directory contains one security provider configuration extension for each type of security provider in the domain's current realm.
config\startup\	readme.txt	File providing information about the directory, which initially serves as a placeholder, and is later used for storing system modules that contain startup plans. Startup plans are used to generate shell scripts that can be used as part of server startup.

Table 14 Base Domain After Applying the WebLogic Server Default Domain Extension Template (Continued)

Directory	File	Description
`console-ext\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder for custom extensions to the WebLogic Server Administration Console.
`init-info\`	`domain-info.xml`	File used to identify domain creation and extension information. Such information includes the identity of the components in the domain, the location of the JDK and applications directory used by the domain, and the templates used to create and extend the domain.
	`security.xml`	File used for creating user groups and roles that establish identity and access to domain resources.
	`startscript.xml`	File used to create the `*.cmd` and `*.sh` files that are placed into the domain's root and `bin` directories.
	`tokenValue.properties`	File that contains the actual values to substitute for the tokens specified in the start scripts.
`lib\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder for the domain's libraries. The JAR files in this directory are added dynamically to the end of the server classpath at server startup.

Table 14 Base Domain After Applying the WebLogic Server Default Domain Extension Template (Continued)

Directory	File	Description

`security\`	`DefaultAuthenticatorInit.ldift`
	`DefaultAuthorizerInit.ldift`
	`DefaultRoleMapperInit.ldift`
	`XACMLAuthorizerInit.ldift`
	`XACMLRoleMapperInit.ldift`
		Files used for bootstrapping tasks, including authentication (user and group), authorization, and role mapping. These files contain LDAP-specific information.
		Note: WebLogic domains created with this release use the XACML providers by default. These XACML security providers are compatible with policies and roles created using the WebLogic Authorization provider (DefaultAuthorizer) and WebLogic Role Mapping provider (DefaultRoleMapper). For more information, see WebLogic Security Providers in Understanding WebLogic Security at http://e-docs.bea.com/wls/docs103/secintro/archtect.html#archtect_0111

	`SerializedSystemIni.dat`	File containing encrypted security information.
`servers\AdminServer\security\`	`boot.properties`	File containing server startup properties, including the user name and password required to boot the server (in encrypted format). It is generated only when you select development startup mode. This file enables you to bypass the prompt for user name and password during a server's startup cycle. For more information, see "Provide User Credentials to Start and Stop Servers" in Starting and Stopping Servers in Managing Server Startup and Shutdown at http://e-docs.bea.com/wls/docs103/server_start/overview.html.
`user_staged_config\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder for configuration information optionally staged by an administrator to be copied to managed servers in the domain.

Resources and Services Configured

The following table identifies the resources and services configured in a domain extended with the WebLogic Server Default Domain extension template.

Table 15 Resources Configured in a WebLogic Server Default Domain

Resource Type	Name	Extension Result
Administration Server	`AdminServer`	Uses the Administration Server provided in the base WebLogic Server domain. The default name is `AdminServer`, unless changed during domain creation. The Administration Server referenced in the extension template is `examplesServer`. For information about naming the Administration Server during domain creation, see "Resources and Services Configured for WebLogic Server Domain Template" on page 14.
Application Deployments	`ejb20BeanMgedEar`	Adds the application and targets it to the Administration Server, `AdminServer`.
	`examplesWebApp`	Adds the application and targets it to the Administration Server, `AdminServer`.
	`jdbcRowSetsEar`	Adds the application and targets it to the Administration Server, `AdminServer`.
	`jspSimpleTagEar`	Adds the application and targets it to the Administration Server, `AdminServer`.
	`mainWebApp`	Adds the application and targets it to the Administration Server, `AdminServer`.
	`webappCachingEar`	Adds the application and targets it to the Administration Server, `AdminServer`.
	`webservicesJwsSimpleEar`	Adds the application and targets it to the Administration Server, `AdminServer`.
	`xmlBeanEar`	Adds the application and targets it to the Administration Server, `AdminServer`.

Table 15 Resources Configured in a WebLogic Server Default Domain (Continued)

Resource Type	Name	Extension Result
JDBC Data Sources	`examples-demo`	Identifies the JDBC data source as an `examples-demo` system resource.
	`examples-demoXA`	Identifies the JDBC data source as an `examples-demoXA` system resource.
JDBC System Resources	`examples-demo` `examples-demoXA`	Identifies the JDBC data source and connection pool setups to be used for non-XA and XA JDBC system resources and targets them to the Administration Server, `AdminServer`.
Security realm	`myrealm`	Uses the security realm provided by the base WebLogic Server domain.

WebLogic Server Examples Extension Template

Using the Configuration Wizard or WLST, you can easily extend a base WebLogic Server domain to create a WebLogic Server Examples domain. You accomplish this by adding the resources and services provided in both the WebLogic Server Default and WebLogic Server Examples extension templates to a base WebLogic Server domain.

For more information about the samples that are supported in the WebLogic Server Examples domain, see *Sample Application Examples and Tutorials for WebLogic Server 10.3*.

Generated Domain Output

The WebLogic Server Examples domain contains a collection of examples that illustrate best practices for coding individual J2EE APIs, and a set of scripts to run those examples. Once the WebLogic Server Default extension template has been applied to a base domain, applying the WebLogic Server Examples extension template allows you to create the WebLogic Server Examples domain. See “Relationships Between Templates” on page 5 for more details.

Table 16 Base Domain After Applying the WebLogic Server Default and WebLogic Server Examples Extension Templates

Directory	File	Description
`user_projects\applications\base_domain\`		
`server\`	`wls_samples_overview.html`	File that opens the WebLogic Server examples online documentation viewer.
`server\docs\`	Various	Directory and files supporting the WebLogic Server examples online documentation viewer.
`server\examples\build\`	Various	Includes sub-directories containing various Java and XML files used to build and work with WebLogic Server examples.
`server\examples\src\`	Various	Includes sub-directories containing various Java, XML, and HTML files used to work with WebLogic Server examples.
`user_projects\domains\base_domain\`		
	`client2certs.pem` `clientkey.pem`	Demo certificate and keystore files.
	`fileRealm.properties`	File containing ACLs, users, and groups that can be used for the default security realm when Compatibility security is used.
	`pointbase.ini`	File containing initialization information for a PointBase JDBC database.
	`setExamplesEnv.cmd` `setExamplesEnv.sh`	Scripts that set up the environment to use the WebLogic Server Examples on Windows and UNIX systems, respectively.
	`startWebLogic.cmd` `startWebLogic.sh`	Scripts used to start the Administration Server on Windows and UNIX systems, respectively.
	`startWebLogicEx.cmd` `startWebLogicEx.sh`	Scripts used to start the Administration Server for the WebLogic Server Examples domain on Windows and UNIX systems, respectively.

Table 16 Base Domain After Applying the WebLogic Server Default and WebLogic Server Examples Extension Templates (Continued)

Directory	File	Description
`autodeploy\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder for automatic deployments.
`bin\`	`setDomainEnv.cmd` `setDomainEnv.sh`	Scripts used to set up the development environment on Windows and UNIX systems, respectively.
	`startManagedWebLogic.cmd` `startManagedWebLogic.sh`	Scripts used to start a Managed Server on Windows and UNIX systems, respectively.
	`startPointBaseConsole.cmd` `startPointBaseConsole.sh`	Scripts used to start the PointBase console on Windows and UNIX systems, respectively.
	`startWebLogic.cmd` `startWebLogic.sh`	Scripts used to start the Administration Server on Windows and UNIX systems, respectively.
	`stopManagedWebLogic.cmd` `stopManagedWebLogic.sh`	Scripts used to stop a Managed Server on Windows and UNIX systems, respectively.
	`stopWebLogic.cmd` `stopWebLogic.sh`	Scripts used to stop the Administration Server on Windows and UNIX systems, respectively.
`config\`	`config.xml`	File containing the configuration information used by the Administration Server. For more information, see Domain Configuration Files in *Understanding Domain Configuration.*
`config\deployments\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder, and is later used for staging an application when the application's staging mode is "staged."

Table 16 Base Domain After Applying the WebLogic Server Default and WebLogic Server Examples Extension Templates (Continued)

Directory	File	Description
`config\diagnostics\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder, and is later used for storing the system modules associated with instrumentation in the WebLogic Diagnostic Framework (WLDF).
`config\jdbc\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder, and is later used for storing global JDBC modules that can be configured directly from JMX (as opposed to JSR-88).
	`examples-demo-jdbc.xml`	Global non-XA JDBC Data Source module for the WebLogic Server Examples domain.

	`examples-demoXA-2-jdbc.xml`
	`examples-demoXA-jdbc.xml`
	`examples-multiDataSource-demoXAPool-jdbc.xml`
	`examples-oracleXA-jdbc.xml`
		Global XA JDBC Data Source modules for the WebLogic Server Examples domain.

`config\jms\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder, and is later used for storing global JMS modules that can be configured directly from JMX (as opposed to JSR-88).
	`examples-jms.xml`	Global JMS module for the WebLogic Server Examples domain.
`config\lib\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder, and is later used for storing JAR files that are added to the system classpath of the server when the server’s Java Virtual Machine starts.

Table 16 Base Domain After Applying the WebLogic Server Default and WebLogic Server Examples Extension Templates (Continued)

Directory	File	Description
`config\nodemanager\`	`nm_password.properties`	File containing Node Manager password property values.
`config\security\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder, and is later used for storing system modules for the security framework. The directory contains one security provider configuration extension for each type of security provider in the domain's current realm.
`config\startup\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder, and is later used for storing system modules that contain startup plans. Startup plans are used to generate shell scripts that can be used as part of server startup.
`console-ext\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder for custom extensions to the WebLogic Server Administration Console.
`init-info\`	`domain-info.xml`	File used to identify domain creation and extension information. Such information includes the identity of the components in the domain, the location of the JDK and applications directory used by the domain, and the templates used to create and extend the domain.
	`security.xml`	File used for creating user groups and roles that establish identity and access to domain resources.
	`startscript.xml`	File used to create the `*.cmd` and `*.sh` files that are placed into the domain's root and `bin` directories.
	`tokenValue.properties`	File that contains the actual values to substitute for the tokens specified in the start scripts.

Table 16 Base Domain After Applying the WebLogic Server Default and WebLogic Server Examples Extension Templates (Continued)

Directory	File	Description
`lib\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder for the domain's libraries. The JAR files in this directory are added dynamically to the end of the server classpath at server startup.

`security\`	`DefaultAuthenticatorInit.ldift`
	`DefaultAuthorizerInit.ldift`
	`DefaultRoleMapperInit.ldift`
	`XACMLAuthorizerInit.ldift`
	`XACMLRoleMapperInit.ldift`
		Files used for bootstrapping tasks, including authentication (user and group), authorization, and role mapping. These files contain LDAP-specific information.
		Note: WebLogic domains created with this release use the XACML providers, by default. These XACML security providers are compatible with policies and roles created using the WebLogic Authorization provider (DefaultAuthorizer) and WebLogic Role Mapping provider (DefaultRoleMapper). For more information, see WebLogic Security Providers in Understanding WebLogic Security at http://e-docs.bea.com/wls/docs103/secintro/archtect.html#archtect_0111

	`SerializedSystemIni.dat`	File containing encrypted security information.

Table 16 Base Domain After Applying the WebLogic Server Default and WebLogic Server Examples Extension Templates (Continued)

Directory	File	Description
`servers\AdminServer\security\`	`boot.properties`	File containing server startup properties, including the user name and password required to boot the server (in encrypted format). It is generated only when you select development startup mode. This file enables you to bypass the prompt for user name and password during a server's startup cycle. For more information, see "Provide User Credentials to Start and Stop Servers" in Starting and Stopping Servers in Managing Server Startup and Shutdown at http://e-docs.bea.com/wls/docs103/server_start/overview.html.
`user_staged_config\`	`readme.txt`	File providing information about the directory, which initially serves as a placeholder for configuration information optionally staged by an administrator to be copied to managed servers in the domain.
`WseeFileStore\`		Directory to be used for the file store for system resources.

Resources and Services Configured

The following table identifies the resources and services configured in a domain extended with the WebLogic Server Examples extension template.

Table 17 Resources Configured in a WebLogic Server Examples Domain

Resource Type	Name	Extension Result
Administration Server	`AdminServer`	Uses the Administration Server provided in the base WebLogic Server domain. The default name is `AdminServer`, unless changed during domain creation. The Administration Server referenced in the extension template is `examplesServer`. For information about naming the Administration Server during domain creation, see "Resources and Services Configured for WebLogic Server Domain Template" on page 14.

Table 17 Resources Configured in a WebLogic Server Examples Domain (Continued)

Resource Type	Name	Extension Result
Application Deployments	`ejb20BeanMgedEar`	Uses the application provided by the WebLogic Server Default extension template applied to the base WebLogic Server domain.
	`examplesWebApp`	Uses the application provided by the WebLogic Server Default extension template applied to the base WebLogic Server domain.
	`jdbcRowSetsEar`	Uses the application provided by the WebLogic Server Default extension template applied to the base WebLogic Server domain.
	`jspSimpleTagEar`	Uses the application provided by the WebLogic Server Default extension template applied to the base WebLogic Server domain.
	`mainWebApp`	Uses the application provided by the WebLogic Server Default extension template applied to the base WebLogic Server domain.
	`SamplesSearchWebApp`	Adds the application and targets it to the Administration Server, `AdminServer`.
	`webappCachingEar`	Uses the application provided by the WebLogic Server Default extension template applied to the base WebLogic Server domain.
	`webservicesJwsSimpleEar`	Uses the application provided by the WebLogic Server Default extension template applied to the base WebLogic Server domain.
	`xmlBeanEar`	Uses the application provided by the WebLogic Server Default extension template applied to the base WebLogic Server domain.

Table 17 Resources Configured in a WebLogic Server Examples Domain (Continued)

Resource Type	Name	Extension Result
File Store	`WseeFileStore`	Adds the file store to be used as the persistent store for the JMS server, `WseeJMSServer`, and the SAF Agent, `ReliableWseeSAFAgent`, and targets the store to the Administration Server, `AdminServer`.
JDBC Data Sources	`examples-demo` `examples-demoXA`	Uses the non-XA and XA JDBC data sources provided by the WebLogic Server Default extension template applied to the base WebLogic Server domain.
	`examples-oracleXA`	Adds the XA JDBC data source and targets it to the Administration Server, `AdminServer`.
	`examples-demoXA-2`	Adds the XA JDBC data source and targets it to the Administration Server, `AdminServer`.
	`examples-multiDataSource-demoXAPool`	Adds the XA JDBC multi data source and targets it to the Administration Server, `AdminServer`. Maps to `examples-demoXA` and `examples-demoXA-2` data sources.
JDBC Store	`exampleJDBCStore`	Adds the JDBC store to be used as the persistent store for the JDBC data source, `examples-demo`, and the JMS server, `examplesJMSServer`, and targets the store to the Administration Server, `AdminServer`.

Table 17 Resources Configured in a WebLogic Server Examples Domain (Continued)

Resource Type	Name	Extension Result
JDBC System Resources	`examples-demo` `examples-demoXA`	Uses the JDBC data source and connection pool setups provided by the WebLogic Server Default extension template applied to the base WebLogic Server domain.

	`examples-demoXA-2`
	`examples-oracleXA`
	`examples-multiDataSource-demoXAPool`
		Adds the JDBC data source and connection pool setups and targets them to the Administration Server, `AdminServer`.

JMS System Resources	`examples-jms`	Identifies the JMS servers, connection factories, queues, and topics to be used for JMS system resources.

JMS Connection Factories	`exampleTopic`
	`exampleTrader`
	`weblogic.examples.jms.QueueConnectionFactory`
		Adds the JMS connection factories as `examples-jms` system resources and targets them to the Administration Server, `AdminServer`.

JMS Servers	`examplesJMSServer`	Adds the JMS server as an `examples-jms` system resource and targets it to the Administration Server, `AdminServer`.
	`WseeJMSServer`	Adds the JMS server as an `examples-jms` system resource and targets it to the Administration Server, `AdminServer`.
JMS Queues	`exampleQueue`	Adds the JMS queue to the JMS server, `examplesJMSServer`.
	`jms/MULTIDATASOURCE_MDB_QUEUE`	Adds the JMS queue to the JMS server, `examplesJMSServer`.
	`weblogic.wsee.wseeExamplesDestinationQueue`	Adds the JMS queue to the JMS server, `WseeJMSServer`.

Table 17 Resources Configured in a WebLogic Server Examples Domain (Continued)

Resource Type	Name	Extension Result
JMS Topics	`exampleTopic`	Adds the JMS topic to the JMS server, `examplesJMSServer`.
	`quotes`	Adds the JMS topic to the JMS server, `examplesJMSServer`.
SAF Agent	`ReliableWseeSAFAgent`	Adds the SAF agent and targets it to the Administration Server, `AdminServer`.
Security realm	`myrealm`	Uses the security realm provided by the base WebLogic Server domain.