

# Deployit Oracle WebLogic Server (WLS) Plugin Manual

February, 2011

## Contents

<b>Preface</b>	<b>1</b>
<b>Introduction</b>	<b>1</b>
<b>Tomcat Plugin Requirements</b>	<b>1</b>
<b>Supported WLS Versions</b>	<b>1</b>
<b>Supported WLS Features</b>	<b>2</b>
<b>WLS Runbook</b>	<b>2</b>
<b>WLS Configuration Items (CIs)</b>	<b>3</b>
ApacheHttpdWlsPluginConfiguration . . . . .	3
ApacheHttpdWlsPluginConfigurationMapping . . . . .	3
WlsCluster . . . . .	4
WlsClusterResourceMapping . . . . .	4
WlsDataSource . . . . .	4
WlsDeploymentPlan . . . . .	5
WlsDomain . . . . .	5
WlsEarMapping . . . . .	5
WlsEjbJarMapping . . . . .	6
WlsForeignJmsConnectionFactory . . . . .	7
WlsForeignJmsDestination . . . . .	7
WlsForeignJmsServer . . . . .	7
WlsJmsConnectionFactory . . . . .	8
WlsJmsQueue . . . . .	8

WlsJmsServer . . . . .	8
WlsMaximumThreadsConstraint . . . . .	9
WlsServer . . . . .	9
WlsServerResourceMapping . . . . .	10
WlsSharedLibraryJar . . . . .	10
WlsSharedLibraryJarMapping . . . . .	10
WlsWarMapping . . . . .	11
WlsWorkManager . . . . .	11

## Preface

This manual describes the Deployit Tomcat Plugin.

## Introduction

The Oracle WebLogic Server (WLS) Plugin supports the deployment, re-deployment and undeployment of a deployment package to a WLS servlet container.

## Tomcat Plugin Requirements

In addition to the requirements for Deployit, the WLS Plugin has the following additional requirements:

- the user account used to access the WLS server must have permission to perform the following actions:
  - run the WLS, `wlst.sh` script (or `wlst.bat` on Windows)

The target middleware needs to be set up so that the administrative interfaces of the target middleware can be accessed by running it on the machine on which the administrative server of the software is installed. Deployit does not support a setup in which the administrative client is installed on a different machine.

For WebLogic Server this means that Deployit will use SSH to log in to the machine on which the “admin server” is running, upload any Python files and other files needed to a temporary directory and then invoke the `wlst.sh` command. Afterwards any temporary files will be removed.

## Supported WLS Versions

The WLS plugin supports the following versions of WLS:

- **10.3.x**

## Supported WLS Features

The WLS Plugin supports the following features:

Concept	Remarks
EAR/WAR/EJB-Jar	Deploy and undeploy EAR, WAR and EJB-Jar's.
Library.Jar	
Apache Httpd Wls Plugin Configuration	Create and destroy Apache HTTPD Server configuration.
DataSource	Create and destroy datasources.
Foreign Jms Connection Factory	Create and destroy JMS Connection factories.
Foreign Jms Destination	Create and destroy JMS destinations.
Foreign Jms Server	Create and destroy JMS Servers.
Maximum Threads Constraint	Configure maximum thread constraints in the server.
Shared Library Jar	Deploy and undeploy sharedl library jars.
Work Manager	Configure the work manager in the server instances.

## WLS Runbook

When the WLS runbook is triggered, the plugin populates the steplist with steps based on the executed task. First, the WLS runbook determines which servers are affected by the pending task. These are all the WLS servers that are a target of one of the deployed items in the deployment or the WLS server that a deployed application is running on in case of an undeploy.

The WLS runbook adds steps in the following order:

- Undeploy EAR/WAR/EJB-Jar
- Undeploy library Jars
- Activate changes
- Delete queues and topics
- Delete connection factories
- Delete datasources
- Unconfigure work managers
- Unconfigure maximum threads constraint
- Delete configuration files on host
- Copy configuration files to host
- Copy and configure libraries
- Modify servers and clusters
- Modify Queues
- Activate changes
- Copy and execute SQL scripts against databases

- Create queues and topics
- Create datasources
- Configure maximum threads constraint
- Configure work managers
- Deploy EAR/WAR/EJB-Jar
- Start applications

## WLS Configuration Items (CIs)

The WLS Plugin defines configuration items (CIs) needed to deploy to Tomcat middleware. To get more information about these CIs, use Deployit's command line interface (CLI). See the **Deployit Command Line Interface (CLI) Manual** for more information.

### ApacheHttpdWlsPluginConfiguration

An abstraction of Apache Weblogic Plugin Configuration. It is used to generate the configuration file which is included in the Apache main httpd.conf file

*Type:* com.xebialabs.deployit.plugin.wls.ci.ApacheHttpdWlsPluginConfiguration

*Properties:*

- **name**(**STRING**): configuration name
- *errorPage*(*STRING*): Error Page
- *mimeMatchExpressions*(*STRING*): Comma separated list of match expression to proxy requests by MIME type
- *pathExpressions*(*STRING*): Comma separated list of path to be used for proxying requests by path

### ApacheHttpdWlsPluginConfigurationMapping

Description unavailable

*Type:* com.xebialabs.deployit.plugin.wls.ci.ApacheHttpdWlsPluginConfigurationMapping

*Properties:*

- **source**(**java.io.Serializable**): Source
- **target**(**java.io.Serializable**): Target
- *clusters*(*Set<com.xebialabs.deployit.plugin.wls.ci.WlsCluster>*): Clusters
- *sourcePropertyOverrides*(*List<com.xebialabs.deployit.ci.mapping.KeyValuePair>*): Overrides for properties of the mapping's source. The key is the property name (consult the documentation or run 'describe' in the CLI), the value is the value to set. Only string, integer and enumerable properties can be overridden. Example: Key: redeliveryLimits, Value: 2
- *virtualHost*(*STRING*): Virtual Host

## WlsCluster

A WebLogic Cluster, a member of WebLogic Domain. It can have WebLogicServers as it's members

*Type:* com.xebialabs.deployit.plugin.wls.ci.WlsCluster

*Properties:*

- **domain(com.xebialabs.deployit.plugin.wls.ci.WlsDomain):** The domain to which the WebLogic Cluster belongs
- **name(STRING):** Name of the WebLogic Cluster
- **servers(Set<com.xebialabs.deployit.plugin.wls.ci.WlsServer>):** Servers in the WebLogic Cluster

## WlsClusterResourceMapping

A mapping of a WebLogic resource for a WlsCluster to a WlsCluster

*Type:* com.xebialabs.deployit.plugin.wls.ci.WlsClusterResourceMapping

*Properties:*

- **source(java.io.Serializable):** Source
- **target(java.io.Serializable):** Target
- **sourcePropertyOverrides(List<com.xebialabs.deployit.ci.mapping.KeyValuePair>):** Overrides for properties of the mapping's source. The key is the property name (consult the documentation or run 'describe' in the CLI), the value is the value to set. Only string, integer and enumerable properties can be overridden. Example: Key: redeliveryLimits, Value: 2

## WlsDataSource

A datasource to connect to a database.

*Type:* com.xebialabs.deployit.plugin.wls.ci.WlsDataSource

*Properties:*

- **driver(STRING):** The driver of the datasource, the classname, e.g. oracle.jdbc.xa.client.OracleXADataSource
- **init(INTEGER):** The initial connectionpool size.
- **jndiName(STRING):** The JNDI name of the datasource, e.g. jdbc/orderdb
- **max(INTEGER):** The maximum connectionpool size.
- **name(STRING):** The name of the datasource in the WebLogic configuration, e.g. Order DataSource
- **uri(STRING):** The JDBC uri to the database, e.g. jdbc:oracle:thin:@ora-prod:1521:orders
- **password(STRING):** The password credential for the database, e.g. tiger
- **properties(STRING):** A comma separated list of name=value pairs.
- **userName(STRING):** The username credential for the database, e.g. scott

## WlsDeploymentPlan

Deployable Oracle Service bus Customization File artifact.

*Type:* com.xebialabs.deployit.plugin.wls.ci.WlsDeploymentPlan

*Properties:*

- **location(**STRING**)**: Location of the artifact.
- **name(**STRING**)**: The technical name of the artifact as it will be used within application servers.

## WlsDomain

A WebLogic Domain.

*Type:* com.xebialabs.deployit.plugin.wls.ci.WlsDomain

*Properties:*

- **activeHost(**com.xebialabs.deployit.ci.Host**)**: The host that runs the admin server
- **adminServerName(**STRING**)**: The name of the admin server
- **name(**STRING**)**: Name of the WebLogic Domain
- **password(**STRING**)**: Password which is used to login to the WebLogic Domain.
- **port(**INTEGER**)**: Port to be used by the AdminServer for this domain
- **startMode(**ENUM**)**: Tells how a managed server is start and stop, default is NodeManager, others are Script or Windows Service
  - Values: [NodeManager, Script, WindowsService]
- **username(**STRING**)**: Username which is used to login to the WebLogic Domain.
- **wlHome(**STRING**)**: The location of the WebLogic Server installation
- **domainHome(**STRING**)**: The location of the WebLogic domain. Defaults to ‘../user\_projects/domains/’
- **enableWlstShWorkaround(**BOOLEAN**)**: Enable workaround for broken wlst.sh script found in some versions of WLS
- **wlsVersion(**ENUM**)**: Wls Version
  - Values: [WEBLOGIC\_8, WEBLOGIC\_9, WEBLOGIC\_10, WEBLOGIC\_11]

## WlsEarMapping

A mapping of an EAR to a WebLogic target

*Type:* com.xebialabs.deployit.plugin.wls.ci.WlsEarMapping

*Properties:*

- **source(**java.io.Serializable**)**: Source

- **target(java.io.Serializable)**: Target
- *deploymentPlan(com.xebialabs.deployit.plugin.wls.ci.WlsDeploymentPlan)*: Deployment Plan
- *deploymentPlanStagingDirectory(STRING)*: Deployment Plan Staging Directory
- *deploymentStrategy(ENUM)*: Deployment Strategy
  - Values: [CLASSIC, STOP\_START, SIDE\_BY\_SIDE]
- *keyValuePairs(List<com.xebialabs.deployit.ci.mapping.KeyValuePair>)*: Key Value Pairs
- *placeholderFormat(ENUM)*: Placeholder Format
  - Values: [SPRING, WINDOWS\_SHELL, STARS, NONE]
- *stageMode(ENUM)*: Deployment staging mode (default is stage)
  - Values: [Stage, NoStage]
- *stagingDirectory(STRING)*: Remote directory where the archives (ear,jar,war) are copied before deploying
- *virtualHost(STRING)*: Virtual Host

## WlsEjbJarMapping

A mapping of an EjbJar to a WebLogic target

Type: com.xebialabs.deployit.plugin.wls.ci.WlsEjbJarMapping

Properties:

- **source(java.io.Serializable)**: Source
- **target(java.io.Serializable)**: Target
- *deploymentPlan(com.xebialabs.deployit.plugin.wls.ci.WlsDeploymentPlan)*: Deployment Plan
- *deploymentPlanStagingDirectory(STRING)*: Deployment Plan Staging Directory
- *deploymentStrategy(ENUM)*: Deployment Strategy
  - Values: [CLASSIC, STOP\_START, SIDE\_BY\_SIDE]
- *keyValuePairs(List<com.xebialabs.deployit.ci.mapping.KeyValuePair>)*: Key Value Pairs
- *mdbListenerPortJndiNameBindings(List<com.xebialabs.deployit.ci.mapping.MdbListenerPortBinding>)*: Bindings of message driven beans JNDI names to the corresponding listener ports present on the target middleware
- *placeholderFormat(ENUM)*: Placeholder Format
  - Values: [SPRING, WINDOWS\_SHELL, STARS, NONE]
- *stageMode(ENUM)*: Deployment staging mode (default is stage)
  - Values: [Stage, NoStage]
- *stagingDirectory(STRING)*: Remote directory where the archives (ear,jar,war) are copied before deploying

## WlsForeignJmsConnectionFactory

Foreign connection factory represents a connection factory that resides on another server, and which is accessible via JNDI.

*Type:* com.xebialabs.deployit.plugin.wls.ci.WlsForeignJmsConnectionFactory

*Properties:*

- **foreignJmsServer(com.xebialabs.deployit.plugin.wls.ci.WlsForeignJmsServer):** The foreign server in which the foreign connection factory is included.
- **localJndiName(String):** The name that the remote object will be bound to in the local server's JNDI tree.
- **name(String):** The name of the foreign connection factory.
- **remoteJndiName(String):** The name of the remote object that will be looked up in the remote JNDI directory.

## WlsForeignJmsDestination

A foreign destination (topic or queue) is a destination on a remote server. When this destination is looked up on the local server, a look-up will be performed automatically on the remote JNDI directory, and the object will be returned from that directory

*Type:* com.xebialabs.deployit.plugin.wls.ci.WlsForeignJmsDestination

*Properties:*

- **foreignJmsServer(com.xebialabs.deployit.plugin.wls.ci.WlsForeignJmsServer):** The name of the Foreign JMS Server
- **localJndiName(String):** The name that the remote object will be bound to in the local server's JNDI tree
- **name(String):** The name of this foreign destination
- **remoteJndiName(String):** The name of the remote object that will be looked up in the remote JNDI directory

## WlsForeignJmsServer

A WebLogic foreign server representing a JNDI provider that resides outside a WebLogic Server.

*Type:* com.xebialabs.deployit.plugin.wls.ci.WlsForeignJmsServer

*Properties:*

- **connectionURL(String):** The URL that WebLogic Server will use to contact the JNDI provider
- **initContextFactory(String):** The initial context of the foreign connection
- **name(String):** The name of the foreign server



## WlsJmsConnectionFactory

A WebLogic JMS Connection Factory

*Type:* com.xebialabs.deployit.plugin.wls.ci.WlsJmsConnectionFactory

*Properties:*

- **jndiName(**STRING**)**: Jndi Name
- **name(**STRING**)**: The name of the connection factory.

## WlsJmsQueue

A WebLogic JMS Queue

*Type:* com.xebialabs.deployit.plugin.wls.ci.WlsJmsQueue

*Properties:*

- **expirationPolicy(**ENUM**)**: The message Expiration Policy to use when an expired message is encountered on a destination
  - Values: [DISCARD, LOG, REDIRECT]
- **jndiName(**STRING**)**: Jndi Name
- **name(**STRING**)**: The name of the queue
- *errorQueue(com.xebialabs.deployit.plugin.wls.ci.WlsJmsQueue)*: The name of the target error destination for messages that have expired or reached their redelivery limit
- *expirationLoggingFormat(**STRING**)*: The policy that defines what information about the message is logged when the Expiration Policy is set to Log
- *jmsServer(com.xebialabs.deployit.plugin.wls.ci.WlsJmsServer)*: Jms Server
- *redeliveryDelayOverride(**INTEGER**)*: The delay, in milliseconds, before rolled back or recovered messages are redelivered, regardless of the RedeliveryDelay specified by the consumer and/or connection factory
- *redeliveryLimits(**INTEGER**)*: The number of redelivery attempts a message can make before it is moved to the error destination

## WlsJmsServer

WebLogic JMSServer that can run on a WebLogic Server.

*Type:* com.xebialabs.deployit.plugin.wls.ci.WlsJmsServer

*Properties:*

- **fileStorePath(**STRING**)**: Path to the file store that the JMSServer will use to store JMS messages.  
e.g. /var/jms/store1
- **name(**STRING**)**: Name of the JMSServer. e.g. JMSServer1

## WlsMaximumThreadsConstraint

Work Manager that defines a set of request classes and thread constraints that manage work performed by WebLogic Server instances.

*Type:* com.xebialabs.deployit.plugin.wls.ci.WlsMaximumThreadsConstraint

*Properties:*

- **name**(**STRING**): Name of the Work Manager
- **threadCount**(**INTEGER**): Thread Count
- *notes*(*STRING*): Notes

## WlsServer

A standard Weblogic Server

*Type:* com.xebialabs.deployit.plugin.wls.ci.WlsServer

*Properties:*

- **domain**(**com.xebialabs.deployit.plugin.wls.ci.WlsDomain**): WebLogic Domain to which this server belongs
- **host**(**com.xebialabs.deployit.ci.Host**): Host on which this server is running
- **name**(**STRING**): Name of the WebLogic Server
- **port**(**INTEGER**): Port for the WebLogic Server
- *arguments*(*STRING*): The arguments for this server, including initial heapsize (e.g. -Xms64m), max-heap size (-Xms256m), and bootclasspath (-Xbootclasspath/p:/var/lib/addons.jar)
- *classpath*(*STRING*): Classpath entries for this server.
- *enableJVMLogRedirection*(*BOOLEAN*): Enable JVM StdOut to Server Log file
- *logFileLocation*(*STRING*): Absolute path of log file. Example; /opt/bea/user\_projects/domain/managedserver1/ms1.log
- *stageMode*(*ENUM*): Deployment staging mode (default is stage)
  - Values: [Stage, NoStage]
- *startCommand*(*STRING*): Command that should be executed to start the managed server.
- *stopCommand*(*STRING*): Command that should be executed to stop the managed server.

## WlsServerResourceMapping

A mapping of a WebLogic resource for a WlsServer to a WlsServer

*Type:* com.xebialabs.deployit.plugin.wls.ci.WlsServerResourceMapping

*Properties:*

- **source(java.io.Serializable):** Source
- **target(java.io.Serializable):** Target
- *sourcePropertyOverrides(List<com.xebialabs.deployit.ci.mapping.KeyValuePair>):* Overrides for properties of the mapping's source. The key is the property name (consult the documentation or run 'describe' in the CLI), the value is the value to set. Only string, integer and enumerable properties can be overridden. Example: Key: redeliveryLimits, Value: 2

## WlsSharedLibraryJar

Deployable WebLogic library JAR artifact.

*Type:* com.xebialabs.deployit.plugin.wls.ci.WlsSharedLibraryJar

*Properties:*

- **location(STRING):** Location of the artifact.
- **name(STRING):** The technical name of the artifact as it will be used within application servers.

## WlsSharedLibraryJarMapping

A mapping of a JAR library to WebLogic middleware

*Type:* com.xebialabs.deployit.plugin.wls.ci.WlsSharedLibraryJarMapping

*Properties:*

- **source(java.io.Serializable):** Source
- **target(java.io.Serializable):** Target
- *deploymentStrategy(ENUM):* Deployment Strategy
  - Values: [CLASSIC, STOP\_START, SIDE\_BY\_SIDE]
- *keyValuePairs(List<com.xebialabs.deployit.ci.mapping.KeyValuePair>):* Key Value Pairs
- *placeholderFormat(ENUM):* Placeholder Format
  - Values: [SPRING, WINDOWS\_SHELL, STARS, NONE]
- *stageMode(ENUM):* Deployment staging mode (default is stage)
  - Values: [Stage, NoStage]
- *stagingDirectory(STRING):* Remote directory where the archives (ear, jar, war) are copied before deploying

## WlsWarMapping

A mapping of a WAR to WebLogic middleware

*Type:* com.xebialabs.deployit.plugin.wls.ci.WlsWarMapping

*Properties:*

- **source(java.io.Serializable):** Source
- **target(java.io.Serializable):** Target
- *contextRoot(STRING):* Context root to deploy to
- *deploymentPlan(com.xebialabs.deployit.plugin.wls.ci.WlsDeploymentPlan):* Deployment Plan
- *deploymentPlanStagingDirectory(STRING):* Deployment Plan Staging Directory
- *deploymentStrategy(ENUM):* Deployment Strategy
  - Values: [CLASSIC, STOP\_START, SIDE\_BY\_SIDE]
- *ejbReferences(List<com.xebialabs.deployit.ci.mapping.EjbReference>):* Specifies the mapping from ejb reference jndi names and locals used in the web.xml to bean jndi names available in middleware
- *keyValuePairs(List<com.xebialabs.deployit.ci.mapping.KeyValuePair>):* Key Value Pairs
- *placeholderFormat(ENUM):* Placeholder Format
  - Values: [SPRING, WINDOWS\_SHELL, STARS, NONE]
- *resourceEnvironmentEntryReferences(List<com.xebialabs.deployit.ci.mapping.ResourceReference>):* Specifies the mapping from resource environment references jndi names and types used in the web.xml to resource references jndi names available in middleware
- *resourceReferences(List<com.xebialabs.deployit.ci.mapping.ResourceReference>):* Specifies the mapping from resource references jndi names and types used in the web.xml to resource references jndi names available in middleware
- *stageMode(ENUM):* Deployment staging mode (default is stage)
  - Values: [Stage, NoStage]
- *stagingDirectory(STRING):* Remote directory where the archives (ear, jar, war) are copied before deploying
- *virtualHost(STRING):* Virtual host to deploy to

## WlsWorkManager

Work Manager that defines a set of request classes and thread constraints that manage work performed by WebLogic Server instances.

*Type:* com.xebialabs.deployit.plugin.wls.ci.WlsWorkManager

*Properties:*

- **name(STRING):** Name of the Work Manager
- *maximumThreadsConstraint(com.xebialabs.deployit.plugin.wls.ci.WlsMaximumThreadsConstraint):* Maximum Threads Constraint