

Deployit WebSphere Plugin Manual

Februari, 2011

Contents

Preface	1
Introduction	1
WAS Plugin Requirements	1
Supported WAS Versions	1
Supported WAS Features	2
WAS Runbook	2
WAS Configuration Items (CIs)	3
WasCluster	3
WasConfigurationToWasScopeMapping	3
WasDataSource	4
WasDeploymentManager	4
WasEarMapping	5
WasEjbJarMapping	5
WasJndiProperties	6
WasJndiPropertiesToWasScopeMapping	6
WasManagedApacheHttpdServer	6
WasManagedServer	7
WasNodeAgent	8
WasResourceMapping	8
WasSharedLibrary	9
WasSharedLibraryToWasScopeTargetMapping	9

WasUnManagedApacheHttpdServer	9
WasUnmanagedServer	10
WasWarClassLoaderMapping	11
WasWarMapping	12
WasWmqQueue	13
WasWmqQueueConnectionFactory	13
WasWmqTopic	13
WasWmqTopicConnectionFactory	14

Preface

This manual describes the Deployit WebSphere Application Server Plugin.

Introduction

The WebSphere Application Server (WAS) Plugin supports the deployment, re-deployment and undeployment of a deployment package to a WAS Network Deployment and Stand Alone installation. Furthermore, it supports, creation, modification and deletion of Application Servers and Clusters.

WAS Plugin Requirements

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

- the user account used to access the WAS server must have permission to perform the following actions:
 - execute the `wsadmin.sh` command located in the bin directory of your WAS installation

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 WebSphere Application Server this means that Deployit will use SSH to log in to the machine on which the “deployment manager” is running, upload any Python files and other files needed to a temporary directory and then invoke the `wsadmin.sh` command. Afterwards any temporary files will be removed.

Supported WAS Versions

The WAS plugin supports the following versions of WAS Network Deployment and Stand Alone Server:

- **6.1.x**
- **7.0.x**

Supported WAS Features

The WAS Plugin supports the following features:

Concept	Remarks
EAR files	Deploy and undeploy EAR archives to WAS with support for resource references,
WAR files	Deploy and undeploy WAR archives to WAS with support for resource references,
EJB-Jar files	Deploy and undeploy EJB-Jar archives to WAS with support for resource references,
Cluster	Create, Modify, Destroy and start/stop Clusters.
DataSource	Create, Modify, Destroy Oracle and DB2 DataSources.
Was(Un)ManagedApacheHttpdServer	Create, Modify and Destroy WebServers, also allows exposing web applications (v
WasSharedLibrary	Create, Modif and Destroy shared libraries.
WasWmqQueueConnectionFactory	Create, Modif and Destroy WebSphere MQ Queue Connection Factories.
WasWmqTopicConnectionFactory	Create, Modif and Destroy WebSphere MQ Topic Connection Factories.
WasWmqQueue	Create, Modif and Destroy WebSphere MQ Queue Connection Factories.
WasWmqTopic	Create, Modif and Destroy WebSphere MQ Topic Connection Factories.

WAS Runbook

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

The WAS runbook adds steps in the following order:

- Undeploy all EAR/WAR/EJB-JAR files on servers, clusters and web servers.
- Synchronizes all nodes (if running against an ND installation).
- Destroys all Topic/Queues.
- Destroys all Connectiopn Factories.
- Destroys all JNDI properties
- Destroys all Shared Libraries.
- Deletes all Configuration Files.
- Copy configuration files to Host.
- Create/Modify Shared Libraries.
- Synchronize all nodes.
- Modify Clusters.
- Modify Servers.
- Copy and run SQL files against the database.
- Create JNDI properties.

- Create Datasource.
- Create all Connectiopn Factories.
- Create all Topic/Queues.
- Deploy all EAR/WAR/EJB-JAR files on servers, clusters and web servers.
- Start applications.
- Generate and propgate the web server plugin configuration.
- Copy static content to web servers.

WAS Configuration Items (CIs)

The WAS Plugin defines configuration items (CIs) needed to deploy to WAS 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.

WasCluster

A WebSphere cluster managed by a deployment manager (WAS ND)

Type: `com.xebialabs.deployit.plugin.was.ci.WasCluster`

Properties:

- **cell(`com.xebialabs.deployit.plugin.was.ci.WasDeploymentManager`)**: Deployment manager that manages this this cluster
- **name(`STRING`)**: Name of the WebSphere cluster, e.g. cluster1
- **servers(`Set<com.xebialabs.deployit.plugin.was.ci.WasManagedServer>`)**: Servers that are part of this cluster

WasConfigurationToWasScopeMapping

Description unavailable

Type: `com.xebialabs.deployit.plugin.was.ci.WasConfigurationToWasScopeMapping`

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

WasDataSource

Description unavailable

Type: com.xebialabs.deployit.plugin.was.ci.WasDataSource

Properties:

- **dataStoreHelperClass(**STRING**)**: DataStoreHelper implementation class that extends the capabilities of the JDBC driver. E.g. 'com.ibm.websphere.rsadapter.Oracle10gDataStoreHelper'
- **jndiName(**STRING**)**: JNDI name, used by applications to lookup the datasource.
- **name(**STRING**)**: Name of the datasource.
- **password(**STRING**)**: Password used to authenticate the username with the database instance.
- **provider(**STRING**)**: Name of the JDBC provider, used to create the datasource.
- **username(**STRING**)**: Username used to connect to the database instance.
- *databaseName(**STRING**)*: Database name, for non-Oracle datasources. Use together with properties 'databaseServerName', and 'databasePortNumber'.
- *databasePortNumber(**INTEGER**)*: Port number of the database, for non-Oracle datasources. Use together with properties 'databaseName', and 'databaseServerName'.
- *databaseServerName(**STRING**)*: Name or IP address of the server on which the database resides, for non-Oracle datasources. Use together with properties 'databaseName', and 'databasePortNumber'.
- *statementCacheSize(**INTEGER**)*: The statement cache size sets the allocation of procedure cache memory and limits the amount of memory from the procedure cache pool used for cached statements.
- *url(**STRING**)*: DataBase Connection URL for Oracle datasources. E.g. 'jdbc:oracle:thin:@was-61:1521:orcl'. For non-Oracle datasources use the three properties 'databaseName', 'databaseServerName', and 'databasePortNumber'

WasDeploymentManager

A WebSphere Application Server deployment manager (WAS ND)

Type: com.xebialabs.deployit.plugin.was.ci.WasDeploymentManager

Properties:

- **host(com.xebialabs.deployit.ci.Host)**: Host on which the WAS deployment manager runs
- **name(**STRING**)**: Name of the WebSphere cell, e.g. MyCell, WASCell, Cell01
- **version(**ENUM**)**: Version of WebSphere Application Server
 - Values: [WAS_61, WAS_70]
- **wasHome(**STRING**)**: Root path of the WebSphere deployment manager profile. e.g. /opt/ws/6.1/profiles/dmgr
- *password(**STRING**)*: Password which is used to login to the WebSphere deployment manager
- *port(**INTEGER**)*: TCP port which is used to login to the WebSphere deployment manager, default is 8879
- *username(**STRING**)*: Username which is used to login to the WebSphere deployment manager

WasEarMapping

A mapping of an EAR to a WebSphere target

Type: com.xebialabs.deployit.plugin.was.ci.WasEarMapping

Properties:

- **source(java.io.Serializable):** Source
- **target(java.io.Serializable):** Target
- **classLoaderMode(ENUM):** Specifies the Classloader mode
 - Values: [PARENT_FIRST, PARENT_LAST]
- **classLoaderPolicy(ENUM):** Specifies the Classloader policy
 - Values: [SINGLE, MULTIPLE]
- **keyValuePairs(List<com.xebialabs.deployit.ci.mapping.KeyValuePair>):** Key Value Pairs
- **placeholderFormat(ENUM):** Placeholder Format
 - Values: [SPRING, WINDOWS_SHELL, STARS, NONE]
- **securityRoleUserGroupMappings(List<com.xebialabs.deployit.plugin.was.ci.SecurityRoleUserGroupMappings>):** Map Security role to users and groups used by EnterPrise Application
- **sharedLibraries(Set<com.xebialabs.deployit.plugin.was.ci.WasSharedLibrary>):** Set of shared library which will used by the ear
- **startingWeight(INTEGER):** Specifies the order in which applications are started. Lower values start earlier.
- **suffixArtifactNameWithTarget(BOOLEAN):** If true, the artifact name will be suffixed with the name of the target.
- **virtualHost(String):** Virtual Host
- **warClassLoaderMapping(List<com.xebialabs.deployit.plugin.was.ci.WasWarClassLoaderMapping>):** Specifies the Class loader mode to WARs in EAR
- **warsWebserversVirtualHostMapping(List<com.xebialabs.deployit.plugin.was.ci.WarsWebserversVirtualHostMapping>):** Map Wars to Webservers and Virtual hosts in EnterPrise Application
- **webservers(Set<com.xebialabs.deployit.plugin.was.ci.WasManagedApacheHttpdServer>):** Set of web-servers that expose the Eneterprise Application

WasEjbJarMapping

A mapping of an EJB JAR to a WebSphere target

Type: com.xebialabs.deployit.plugin.was.ci.WasEjbJarMapping

Properties:

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

- **target(java.io.Serializable)**: Target
- **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]
- **sharedLibraries(Set<com.xebialabs.deployit.plugin.was.ci.WasSharedLibrary>)**: Set of shared library which will be used by the ejb
- **startingWeight(INTEGER)**: Specifies the order in which applications are started. Lower values start earlier.

WasJndiProperties

Description unavailable

Type: com.xebialabs.deployit.plugin.was.ci.WasJndiProperties

Properties:

- **stringNameSpaceBindings(List<com.xebialabs.deployit.ci.mapping.KeyValuePair>)**: Key/value pairs that are to be stored in the WebSphere JNDI tree

WasJndiPropertiesToWasScopeMapping

Description unavailable

Type: com.xebialabs.deployit.plugin.was.ci.WasJndiPropertiesToWasScopeMapping

Properties:

- **source(java.io.Serializable)**: Source
- **target(java.io.Serializable)**: Target
- **keyValuePairs(List<com.xebialabs.deployit.ci.mapping.KeyValuePair>)**: Values for placeholders in the WasJndiProperties that are to be replaced. The key is the placeholder name. Example: Key: DATABASE_USER, Value: testUser

WasManagedApacheHttpdServer

Description unavailable

Type: com.xebialabs.deployit.plugin.was.ci.WasManagedApacheHttpdServer

Properties:

- **accessLogLocation(String)**: Location where deployit will create a directory where access log will be placed.

- **apachectlPath(String)**: Path of the executable that will restart apache, e.g. /usr/sbin/apachectl
- **configurationLocation(String)**: Location where deployit will generate apache httpd.conf fragment files.
- **errorLogLocation(String)**: Location where deployit will create a directory where error log will be placed.
- **host(com.xebialabs.deployit.ci.Host)**: Host on which the web server runs
- **htdocsLocation(String)**: Location where deployit will create a directory (based on the vhost name) where static content will be placed.
- **name(String)**: Name
- **node(com.xebialabs.deployit.plugin.was.ci.WasNode)**: The WAS node on which Apache is installed.
- **pluginInstallationDirPath(String)**: The directory where the WebSphere plugin for Apache has been installed.
- **port(Integer)**: The port where the Apache HTTPD server is running on. e.g. 80, 443
- **webServerVendorType(Enum)**: The Web server vendor type.
 - Values: [APACHE, IHS]
- **modules(Set<com.xebialabs.deployit.plugin.apache.httpd.ci.ApacheModule>)**: Modules
- **pluginConfigurationPath(String)**: The path of the WebSphere plugin configuration file. Defaults to /config/plugin-cfg.xml

WasManagedServer

A WebSphere server managed by a node that is part of a deployment manager (WAS ND)

Type: com.xebialabs.deployit.plugin.was.ci.WasManagedServer

Properties:

- **name(String)**: Name of the WebSphere server, e.g. server1
- **applicationClassLoaderPolicyAndMode(Enum)**: Server-wide application classloader policy and mode
 - Values: [DEFAULT, MULTIPLE, SINGLE_PARENT_FIRST, SINGLE_PARENT_LAST]
- **bootClasspath(String)**: Boot classpath for this server.
- **classpath(String)**: Classpath for this server.
- **cookieDomain(String)**: Session cookie domain
- **cookieName(String)**: Session cookie name
- **cookiePath(String)**: Session cookie path
- **disableJit(Boolean)**: Disable just-in-time compiler.
- **enableSessionCookies(Boolean)**: Enable session cookies

- *environmentEntries(List<com.xebialabs.deployit.ci.mapping.KeyValuePair>)*: Environment entries
- *initHeapSize(INTEGER)*: Initial heap size to be allocated to the JVM (in megabytes).
- *jvmArguments(STRING)*: Generic JVM arguments.
- *jvmStdErr(STRING)*: Path to the JVM stderr log file. Example; /data/waslogs/jvm_stderr.log
- *jvmStdOut(STRING)*: Path to the JVM stdout log file. Example; /data/waslogs/jvm_stdout.log
- *maxHeapSize(INTEGER)*: Maximum heap size to be allocated to the JVM (in megabytes).
- *maximumSessionsInMemory(INTEGER)*: Maximum # of HTTP sessions in memory
- *node(com.xebialabs.deployit.plugin.was.ci.WasNodeAgent)*: Node on which the server runs
- *servletCaching(BOOLEAN)*: Enable servlet caching
- *sessionTimeout(INTEGER)*: HTTP session timeout in minutes
- *stderr(STRING)*: Path to the stderr log file. Example; /data/waslogs/stderr.log
- *stdout(STRING)*: Path to the stdout log file. Example; /data/waslogs/stdout.log
- *umask(STRING)*: Umask of started process.
- *workingDir(STRING)*: Working directory of started process.

WasNodeAgent

A WebSphere node agent.

Type: com.xebialabs.deployit.plugin.was.ci.WasNodeAgent

Properties:

- **cell(com.xebialabs.deployit.plugin.was.ci.WasDeploymentManager)**: Deployment manager that manages this node agent
- **name(STRING)**: Name of the WebSphere node in the WebSphere cell, e.g. MyNode, Node01

WasResourceMapping

Description unavailable

Type: com.xebialabs.deployit.plugin.was.ci.WasResourceMapping

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

WasSharedLibrary

Description unavailable

Type: com.xebialabs.deployit.plugin.was.ci.WasSharedLibrary

Properties:

- **name(**STRING**)**: Name of the shared library so it can be used by reference in an Application
- **classPath(**STRING**)**: classpath of the shared library, e.g. /var/shared/log4j.jar;/var/shared/lib/ora/ojdbc14.jar
- **configurationFiles(*Set*<com.xebialabs.deployit.ci.artifact.ConfigurationFiles>)**: The set of configuration files that are part of this shared library
- **libraries(*Set*<com.xebialabs.deployit.ci.artifact.Libraries>)**: The set of libraries that are part of this shared library

WasSharedLibraryToWasScopeTargetMapping

Description unavailable

Type: com.xebialabs.deployit.plugin.was.ci.WasSharedLibraryToWasScopeTargetMapping

Properties:

- **source(**java.io.Serializable**)**: Source
- **target(**java.io.Serializable**)**: Target
- **keyValuePairs(*List*<com.xebialabs.deployit.ci.mapping.KeyValuePair>)**: Values for placeholders in the deployable artifact that are to be replaced. The key is the placeholder name. Example: Key: DATABASE_USER, Value: testUser

WasUnManagedApacheHttpdServer

Description unavailable

Type: com.xebialabs.deployit.plugin.was.ci.WasUnManagedApacheHttpdServer

Properties:

- **accessLogLocation(**STRING**)**: Location where deployit will create a directory where access log will be placed.
- **apachectlPath(**STRING**)**: Path of the executable that will restart apache, e.g. /usr/sbin/apachectl
- **configurationLocation(**STRING**)**: Location where deployit will generate apache httpd.conf fragment files.
- **errorLogLocation(**STRING**)**: Location where deployit will create a directory where error log will be placed.
- **host(**com.xebialabs.deployit.ci.Host**)**: Host on which the web server runs

- **htdocsLocation(**STRING**)**: Location where deployit will create a directory (based on the vhost name) where static content will be placed.
- **name(**STRING**)**: Name
- **node(**com.xebialabs.deployit.plugin.was.ci.WasNode**)**: The WAS node on which Apache is installed.
- **pluginInstallationDirPath(**STRING**)**: The directory where the WebSphere plugin for Apache has been installed.
- **port(**INTEGER**)**: The port where the Apache HTTPD server is running on. e.g. 80, 443
- **webServerVendorType(**ENUM**)**: The Web server vendor type.
 - Values: [APACHE, IHS]
- *modules(Set<com.xebialabs.deployit.plugin.apache.httpd.ci.ApacheModule>)*: Modules
- *pluginConfigurationPath(**STRING**)*: The path of the WebSphere plugin configuration file. Defaults to /config//plugin-cfg.xml

WasUnmanagedServer

An unmanaged WebSphere Application Server (WAS Base/SA)

Type: com.xebialabs.deployit.plugin.was.ci.WasUnmanagedServer

Properties:

- **cellName(**STRING**)**: Name of the WebSphere cell, e.g. MyCell, WASCell, Cell01
- **host(**com.xebialabs.deployit.ci.Host**)**: Host on which the unmanaged WAS server runs
- **name(**STRING**)**: Name of the WebSphere server, e.g. server1
- **nodeName(**STRING**)**: Name of the WebSphere node
- **version(**ENUM**)**: Version of WebSphere Application Server.
 - Values: [WAS.61, WAS.70]
- **wasHome(**STRING**)**: Root path of the WebSphere installation path. e.g. /opt/ws/6.1/appserver/profiles/AppSrv01
- *applicationClassLoaderPolicyAndMode(**ENUM**)*: Server-wide application classloader policy and mode
 - Values: [DEFAULT, MULTIPLE, SINGLE_PARENT_FIRST, SINGLE_PARENT_LAST]
- *bootClasspath(**STRING**)*: Boot classpath for this server.
- *classpath(**STRING**)*: Classpath for this server.
- *cookieDomain(**STRING**)*: Session cookie domain
- *cookieName(**STRING**)*: Session cookie name
- *cookiePath(**STRING**)*: Session cookie path
- *disableJit(**BOOLEAN**)*: Disable just-in-time compiler.

- *enableSessionCookies(BOOLEAN)*: Enable session cookies
- *environmentEntries(List<com.xebialabs.deployit.ci.mapping.KeyValuePair>)*: Environment entries
- *initHeapSize(INTEGER)*: Initial heap size to be allocated to the JVM (in megabytes).
- *jvmArguments(STRING)*: Generic JVM arguments.
- *jvmStdErr(STRING)*: Path to the JVM stderr log file. Example; /data/waslogs/jvm_stderr.log
- *jvmStdOut(STRING)*: Path to the JVM stdout log file. Example; /data/waslogs/jvm_stdout.log
- *maxHeapSize(INTEGER)*: Maximum heap size to be allocated to the JVM (in megabytes).
- *maximumSessionsInMemory(INTEGER)*: Maximum # of HTTP sessions in memory
- *password(STRING)*: Password which is used to login to the WebSphere Administration.
- *port(INTEGER)*: TCP port which is used to login to the WebSphere Administration, default is 8880
- *resourceEnvironmentJndiNames(Set)*: Resource environment jndi names
- *servletCaching(BOOLEAN)*: Enable servlet caching
- *sessionTimeout(INTEGER)*: HTTP session timeout in minutes
- *stderr(STRING)*: Path to the stderr log file. Example; /data/waslogs/stderr.log
- *stdout(STRING)*: Path to the stdout log file. Example; /data/waslogs/stdout.log
- *umask(STRING)*: Umask of started process.
- *username(STRING)*: Username which is used to login to the WebSphere Administration.
- *workingDir(STRING)*: Working directory of started process.

WasWarClassLoaderMapping

A mapping of an EAR to a WebSphere target

Type: com.xebialabs.deployit.plugin.was.ci.WasWarClassLoaderMapping

Properties:

- *classLoaderMode(ENUM)*: Class Loader Mode
 - Values: [PARENT_FIRST, PARENT_LAST]
- *warName(STRING)*: War Name

WasWarMapping

A mapping of a WAR to WebSphere middleware

Type: com.xebialabs.deployit.plugin.was.ci.WasWarMapping

Properties:

- **source(java.io.Serializable):** Source
- **target(java.io.Serializable):** Target
- **classLoaderMode(ENUM):** Specifies the Classloader mode
 - Values: [PARENT_FIRST, PARENT_LAST]
- **contextRoot(STRING):** Context root to deploy to
- **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
- **fileServing(ENUM):** Set File Serving enabled to WAR
 - Values: [DO_NOT_OVERRIDE, FALSE, TRUE]
- **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
- **securityRoleUserGroupMappings(List<com.xebialabs.deployit.plugin.was.ci.SecurityRoleUserGroupMappings>):** Map Security role to users and groups used by EnterPrise Application
- **sharedLibraries(Set<com.xebialabs.deployit.plugin.was.ci.WasSharedLibrary>):** Set of shared library which will used by the war
- **startingWeight(INTEGER):** Specifies the order in which applications are started. Lower values start earlier.
- **suffixArtifactNameWithTarget(BOOLEAN):** If true, the artifact name will be suffixed with the name of the target.
- **virtualHost(STRING):** Virtual host to deploy to
- **webservers(Set<com.xebialabs.deployit.plugin.was.ci.WasManagedApacheHttpdServer>):** Set of web-servers that expose the Eneterprise Application

WasWmqQueue

Description unavailable

Type: com.xebialabs.deployit.plugin.was.ci.WasWmqQueue

Properties:

- *baseQueueName(STRING)*: Base Queue Name
- *jndiName(STRING)*: JNDI name
- *name(STRING)*: WebSphere name

WasWmqQueueConnectionFactory

Description unavailable

Type: com.xebialabs.deployit.plugin.was.ci.WasWmqQueueConnectionFactory

Properties:

- *channel(STRING)*: Channel
- *jndiName(STRING)*: JNDI name
- *name(STRING)*: WebSphere name
- *queueManagerHost(com.xebialabs.deployit.ci.Host)*: Queue Manager Host
- *queueManagerName(STRING)*: Queue Manager Name
- *queueManagerPort(INTEGER)*: Queue Manager Port

WasWmqTopic

Description unavailable

Type: com.xebialabs.deployit.plugin.was.ci.WasWmqTopic

Properties:

- *baseTopicName(STRING)*: Base Topic Name
- *jndiName(STRING)*: JNDI name
- *name(STRING)*: WebSphere name

WasWmqTopicConnectionFactory

Description unavailable

Type: com.xebialabs.deployit.plugin.was.ci.WasWmqTopicConnectionFactory

Properties:

- *channel(STRING):* Channel
- *jndiName(STRING):* JNDI name
- *name(STRING):* WebSphere name
- *queueManagerHost(com.xebialabs.deployit.ci.Host):* Queue Manager Host
- *queueManagerName(STRING):* Queue Manager Name
- *queueManagerPort(INTEGER):* Queue Manager Port