

Deployit JBoss Application Server 7+ Plugin Manual

Version 3.8.0

Table of Contents

Table of Contents	2
Preface	3
Overview	3
Features	3
Requirements	3
Usage in Deployment Packages	3
Using the deployables and deployed	4
Deployable vs. Container table	4
Deployed Actions Table	4
Deploying applications	4
Standalone Mode	4
Domain Mode	4
Discovery	4
Extension points	5
Extending the plugin to support JDBC Driver deployment	5
Extending the plugin with custom control task	6
CI Reference	7
Configuration Item Overview	7
Deployables	7
Deployeds	7
Containers	7
Other Configuration Items	7
Configuration Item Details	7
jbossdm.BaseDataSource	7
jbossdm.CliBasedContainer	10
jbossdm.CliManagedDeployed	11
jbossdm.CliManagedDeployedArtifact	12
jbossdm.DataSource	13
jbossdm.DataSourceSpec	15
jbossdm.Domain	17
jbossdm.Ear	17
jbossdm.EarModule	18
jbossdm.JeeDataSource	19
jbossdm.JeeXaDataSource	21
jbossdm.Profile	23
jbossdm.Queue	23
jbossdm.QueueSpec	24
jbossdm.ServerGroup	25
jbossdm.StandaloneServer	25
jbossdm.Topic	26
jbossdm.TopicSpec	26
jbossdm.War	27
jbossdm.WarModule	27
jbossdm.XaDataSource	28
jbossdm.XaDataSourceSpec	30

Preface

This document describes the functionality provided by the JBoss Domain (JBoss AS 7.1+, JBoss EAP 6) plugin.

See the **Deployit Reference Manual** for background information on Deployit and deployment concepts.

Overview

The JBoss Domain plugin is a Deployit plugin that adds the capability to manage deployments and resources on JBoss application server 7.1+ or JBoss EAP 6. The plugin has the capability of managing application artifacts, datasource and other JMS resources via the JBoss Cli, and can easily be extended to support more deployment options or management of new artifacts/resources on JBoss AS.

Features

- Domain and Standalone mode support
- Deployment of application artifacts
 - Enterprise application (EAR)
 - Web application (WAR)
- Deployment of resources
 - Datasource including XA Datasource
 - JMS Queue
 - JMS Topic
- Discovery of Profiles and ServerGroups in Domain

Requirements

- **Deployit requirements**
 - **Deployit:** version 3.7+
 - **Jython:** jython-standalone-2.5.1.jar installed in <DEPLOYIT_SERVER_HOME>/lib
- **Infrastructural requirements**
 - **JBoss AS versions:** 7.1+
 - **JBoss EAP versions:** 6.x
 - **User credentials** for accessing the Host and JBoss Cli.

Usage in Deployment Packages

The plugin works with the standard deployment package of DAR format. Please see the *Packaging Manual* for more details about the DAR format and the ways to compose one.

The following is a sample MANIFEST.MF file that can be used to create a JBoss AS specific deployment package. It contains declarations for an [Ear](#), a [datasource](#) and a couple of JMS resources.

```
Manifest-Version: 1.0
Deployit-Package-Format-Version: 1.3
CI-Application: SampleApp
CI-Version: 1.0

Name: PetClinic-1.0.ear
CI-Name: PetClinic
CI-Type: jee.Ear

Name: testDatasource
CI-Type: jbossm.DatasourceSpec
CI-jndiName: jdbc/sampleDatasource
CI-connectionUrl: jdbc:mysql://localhost/test
CI-driverName: mysql
CI-username: {{DATABASE_USERNAME}}
CI-password: {{DATABASE_PASSWORD}}
```

Name: testQueue
CI-Type: jbossdm.QueueSpec
CI-jndiName: jms/testQueue
Name: testTopic
CI-Type: jbossdm.TopicSpec
CI-jndiName: jms/testTopic

Using the deployables and deployed

The following table describes which deployable/container combinations are possible.

Deployable vs. Container table

Deployable	Container	Generated deployed
jee.Ear jbossdm.Ear	jbossdm.ApplicationContainer	jbossdm.EarModule
jee.War jbossdm.War	jbossdm.ApplicationContainer	jbossdm.WarModule
jee.DataSourceSpec	jbossdm.ResourceContainer	jbossdm.JeeXaDataSource jbossdm.JeeDataSource
jbossdm.XaDataSourceSpec	jbossdm.ResourceContainer	jbossdm.XaDataSource
jbossdm.DataSourceSpec	jbossdm.ResourceContainer	jbossdm.Datasource
jee.QueueSpec jbossdm.QueueSpec	jbossdm.ResourceContainer	jbossdm.Queue
jee.TopicSpec jbossdm.TopicSpec	jbossdm.ResourceContainer	jbossdm.Topic

The following table describes the effect a deployed has on it's container

Deployed Actions Table

Deployed	Actions performed for operations		
	Create	Destroy	Modify
jbossdm.EarModule jbossdm.WarModule	<ul style="list-style-type: none"> upload artifact deploy application 	<ul style="list-style-type: none"> undeploy old application version 	<ul style="list-style-type: none"> undeploy old application version deploy new application version
jbossdm.DataSource jbossdm.XaDataSource jbossdm.JeeDataSource jbossdm.JeeXaDataSource	<ul style="list-style-type: none"> create datasource set connection/datasource properties enable datasource 	<ul style="list-style-type: none"> destroy datasource 	<ul style="list-style-type: none"> destroy datasource create datasource set connection/datasource properties enable datasource
jbossdm.Queue	<ul style="list-style-type: none"> create Queue 	<ul style="list-style-type: none"> destroy Queue 	<ul style="list-style-type: none"> destroy Queue create modified Queue
jbossdm.Topic	<ul style="list-style-type: none"> create Topic 	<ul style="list-style-type: none"> destroy Topic 	<ul style="list-style-type: none"> destroy Topic create modified Topic

Deploying applications

Note that the plugin uses the JBoss Cli to (un)install artifacts and resources. As such, the plugin assumes that the JBoss Domain or Standalone server has already been started. The plugin does not support the starting of the domain or standalone server prior to a deployment.

Standalone Mode

Artifacts (war, ear) and resources (datasources, queues, topics, etc) can be targeted to a [StandaloneServer](#).

Domain Mode

Artifacts (war, ear) can be targeted to either a [Domain](#) or [ServerGroup](#). When targeted to a domain, the artifacts are (un)installed on all server groups defined for the domain. For specific targetting of artifacts to certain server groups, you can define the server groups in your environment.

Resources (datasources, queues, topics, etc) can be targeted to either a [Domain](#) or [Profile](#). When targeted to a domain, the resources are (un)installed in the "default" profile. For specific targetting of resources to certain profiles, you can define the profiles in your environment.

Discovery

The plugin supports the discovery of Profiles and Server Groups in a Domain.

Here is an example CLI script which discovers a sample Domain:

```
host = repository.create(factory.configurationItem('Infrastructure/jboss-host', 'overthere.SshHost',
    {'connectionType': 'SFTP', 'address': 'jboss-7', 'username': 'root', 'password': 'centos', 'os': 'UNIX'}))
jboss = factory.configurationItem('Infrastructure/jboss-host/jboss-domain', 'jbossdm.Domain',
    {'home': '/opt/jboss/7', 'host': 'Infrastructure/jboss-host', 'username': 'jbossAdmin', 'password': 'jboss'})

taskId = deployit.createDiscoveryTask(jboss)
deployit.startTaskAndWait(taskId)
cis = deployit.retrieveDiscoveryResults(taskId)
deployit.print(cis)

#discovery just discovers the topology and keeps the configuration items in memory. Save them in Deployit
repository
repository.create(cis)
```

Few things to note about the above discovery example:

- JBoss Domain has a containment relation with a Host (created under a Host), so the server id has been kept as 'Infrastructure/jboss-host/jboss-domain'

Extension points

The plugin is designed to be extended through Deployit's Plugin API type system and jython. The plugin wraps the JBoss Cli with a jython runtime environment, thus allowing extenders to interact with JBoss and Deployit from the script. Note that the jython script is executed on the Deployit Server itself and has full access to the following Deployit objects :

- **deployed**: The current deployed object on which the operation has been triggered.
- **step**: The step object that the script is being executed from. Exposes an overthere remote connection for file manipulation and a method to execute JBoss Cli commands.
- **container**: The container object to which the deployed is targeted to.
- **delta**: The delta specification that lead to the script being executed.
- **deployedApplication**: The entire deployed application.

The plugin associates **Create**, **Modify**, **Destroy**, **Noop** and **Inspect** operations received from Deployit with jython scripts that need to be executed for the specific operation to be performed.

There also exists an advanced method to extend the plugin, but the implementation of this form of extension needs to be written in the Java programming language and consists of writing so-called `Deployed contributors`, `PlanPreProcessors` and `Contributors`.

Please refer to the *Customization Manual* for a detailed explanation of the type system and advanced methods of customization of plugins. Also refer to the Overthere documentation for working with remote files.

Extending the plugin to support JDBC Driver deployment

In this example we will deploy a jdbc driver jar to a [Domain](#) or [StandaloneServer](#) as a module and register the driver with JBoss datasources subsystem.

Define the deployed and deployable to represent a JDBC Driver

The following synthetic.xml snippet shows the definition of the JDBC Driver deployed. The deployed will be targeted to a [Domain](#) or a [StandaloneServer](#). Please refer to the CI reference section of this document to understand the interfaces and class hierarchy of these types.

```
<type type="jbossdm.JdbcDriverModule" extends="jbossdm.CliManagedDeployedArtifact"
    deployable-type="jbossdm.JdbcDriver" container-type="jbossdm.CliManagingContainer">
  <generate-deployable type="jbossdm.JdbcDriver" extends="udm.BaseDeployableArchiveArtifact">

  <property name="driverName"/>
  <property name="driverModuleName"/>
  <property name="driverXaDatasourceClassName/>

  <!-- hidden properties to specify the jython scripts to execute for an operation -->
  <property name="createScript" default="jboss/dm/ds/create-jdbc-driver.py" hidden="true"/>
</type>
```

Defined the create-jdbc-driver.py

```

from com.xebialabs.overthere.util import OverthereUtils

#create module directory to copy jar and module.xml to
driverModuleName = deployed.getProperty("driverModuleName")
moduleRelPath = driverModuleName.replaceAll("\\.", "/")
moduleAbsolutePath = "%s/modules/%s" % (container.getProperty("home"), moduleRelPath)
moduleDir = step.getRemoteConnection().getFile(moduleAbsolutePath);
moduleDir.mkdirs();
#upload jar
moduleJar = moduleDir.getFile(deployed.file.getName())
deployed.file.copyTo(moduleJar)

moduleXmlContent = """
<?xml version="1.0" encoding="UTF-8"?>
<module xmlns="urn:jboss:module:1.0" name="%s">
  <resources>
    <resource-root path="%s"/>
  </resources>
  <dependencies>
    <module name="javax.api"/>
    <module name="javax.transaction.api"/>
  </dependencies>
</module>
""" % (deployed.getProperty("driverModuleName"), deployed.file.getName())

#create module.xml
moduleXml = moduleDir.getFile("module.xml")
OverthereUtils.write(moduleXmlContent.getBytes(), moduleXml)

#register driver with the datasource subsystem
driverName = deployed.getProperty("driverName")
xaClassName = deployed.getProperty("driverXaDataSourceClassName")
cmd = '/subsystem=datasources/jdbc-driver=%s:add(driver-name="%s",driver-module-name="%s",driver-xa-datasource-
class-name="%s")' % (driverName, driverName, driverModuleName, xaClassName)
cmd = prependProfilePath(cmd) #prefix with profile if deploying to domain
executeCmd(cmd) #used to execute a JBoss Cli command.

```

Extending the plugin with custom control task

The plugin has the capability to add control tasks to [CliManagedDeployed](#) or [CliManagedContainer](#). The control task can be specified as a jython script that will be executed on the Deployit Server or as an OS shell script that will be run on the target host. The OS shell script is first processed with FreeMarker before being executed.

Creating a jython based control task to list jdbc drivers in a StandaloneServer

Synthetic.xml snippet

```

<type-modification type="jbossdm.StandaloneServer">
  <property name="listJdbcDriversPythonTaskScript" hidden="true" default="jboss/dm/container/list-jdbc-
drivers.py"/>
  <!-- Note "PythonTaskScript" is appended to the method name to determine the script to run. -->
  <method name="listJdbcDrivers"/>
</type>

```

list-jdbc-drivers.py snippet

```

drivers = executeCmd("/subsystem=datasources:installed-drivers-list")
logOutput(drivers) #outputs to the step log

```

Start the StandaloneServer

Synthetic.xml snippet

```

<type-modification type="jbossdm.StandaloneServer">
  <property name="startShellTaskScript" hidden="true" default="jboss/dm/container/start-standalone"/>
  <!-- Note "ShellTaskScript" is appended to the method name to determine the script to run. -->
  <method name="start"/>
</type>

```

start-standalone.sh snippet

```

nohup ${container.home}/bin/standalone.sh >>nohup.out 2>&1 &
sleep 2
echo background process to start standalone server executed.

```

CI Reference

Configuration Item Overview

Deployables

CI	Description
jbossdm.DataSourceSpec	DataSource
jbossdm.Ear	A JEE EAR archive
jbossdm.QueueSpec	A Queue
jbossdm.TopicSpec	A Topic
jbossdm.War	A JEE WAR archive
jbossdm.XaDataSourceSpec	XA DataSource

Deployeds

CI	Description
jbossdm.BaseDataSource	Base definition of a DataSource
jbossdm.CliManagedDeployed	Base for all deployed that utilize the JBoss Cli for configuration
jbossdm.CliManagedDeployedArtifact	Base for all deployed artifacts that utilize the JBoss Cli for configuration
jbossdm.DataSource	DataSource
jbossdm.EarModule	Ear with values configured for a deployment
jbossdm.JeeDataSource	Datasource
jbossdm.JeeXaDataSource	XA DataSource
jbossdm.Queue	A Jboss Queue
jbossdm.Topic	A JBoss topic
jbossdm.WarModule	War with values configured for a deployment
jbossdm.XaDataSource	XA DataSource

Containers

CI	Description
jbossdm.CliBasedContainer	JBoss Cli Managed Container
jbossdm.Domain	Description unavailable
jbossdm.Profile	JBoss Profile
jbossdm.ServerGroup	JBoss Server Group
jbossdm.StandaloneServer	JBoss Standalone Server

Other Configuration Items

CI	Description
jbossdm.BaseDataSource	Base definition of a DataSource
jbossdm.CliBasedContainer	JBoss Cli Managed Container
jbossdm.CliManagedDeployed	Base for all deployed that utilize the JBoss Cli for configuration
jbossdm.CliManagedDeployedArtifact	Base for all deployed artifacts that utilize the JBoss Cli for configuration
jbossdm.DataSource	DataSource
jbossdm.DataSourceSpec	DataSource
jbossdm.Domain	Description unavailable
jbossdm.Ear	A JEE EAR archive
jbossdm.EarModule	Ear with values configured for a deployment
jbossdm.JeeDataSource	Datasource
jbossdm.JeeXaDataSource	XA DataSource
jbossdm.Profile	JBoss Profile
jbossdm.Queue	A Jboss Queue
jbossdm.QueueSpec	A Queue
jbossdm.ServerGroup	JBoss Server Group
jbossdm.StandaloneServer	JBoss Standalone Server
jbossdm.Topic	A JBoss topic
jbossdm.TopicSpec	A Topic
jbossdm.War	A JEE WAR archive
jbossdm.WarModule	War with values configured for a deployment
jbossdm.XaDataSource	XA DataSource
jbossdm.XaDataSourceSpec	XA DataSource

Configuration Item Details

jbossdm.BaseDataSource

Virtual Type

Type Hierarchy [jbossdm.CliManagedDeployed](#) >> udm.BaseDeployed >>
udm.BaseConfigurationItem

Interfaces udm.Deployed, udm.ConfigurationItem

Base definition of a DataSource

Public Properties	
* driverName : STRING	Defines the JDBC driver the datasource should use. It is a symbolic name matching the the name of installed driver. In case the driver is deployed as jar, the name is the name of deployment unit.
* jndiName : STRING	Specifies the JNDI name for the datasource
backgroundValidation : BOOLEAN = false	An element to specify that connections should be validated on a background thread versus being validated prior to use. Changing this value can be done only on disabled datasource, requires a server restart otherwise.
checkValidSql : STRING	Specify an SQL statement to check validity of a pool connection. This may be called when managed connection is obtained from the pool
deployable : CI<udm.Deployable>	The deployable that this deployed is derived from.
exceptionSorter : STRING	An org.jboss.jca.adapters.jdbc.ExceptionSorter that provides an isExceptionFatal(SQLException) method to validate if an exception should broadcast an error
maxPoolSize : INTEGER = 0	The max-pool-size element specifies the maximum number of connections for a pool. No more connections will be created in each sub-pool
minPoolSize : INTEGER = 0	The min-pool-size element specifies the minimum number of connections for a pool
password : STRING	Specifies the password used when creating a new connection
prefillEnabled : BOOLEAN = false	Should the pool be prefilled. Changing this value can be done only on disabled datasource, requires a server restart otherwise.
securityDomain : STRING	Specifies the security domain which defines the javax.security.auth.Subject that are used to distinguish connections in the pool
sharePreparedStatements : BOOLEAN	Whether to share prepared statements, i.e. whether asking for same statement twice without closing uses the same underlying prepared statement
staleConnectionChecker : STRING	An org.jboss.jca.adapters.jdbc.StaleConnectionChecker that provides an isStaleConnection(SQLException) method which if it returns true will wrap the exception in an org.jboss.jca.adapters.jdbc.StaleConnectionException
statementCacheSize : INTEGER = -1	The number of prepared statements per connection in an LRU cache
strictMinimum : BOOLEAN = false	Specifies if the min-pool-size should be considered strictly
username : STRING	Specify the user name used when creating a new connection
validConnectionChecker : STRING	An org.jboss.jca.adapters.jdbc.ValidConnectionChecker that provides an isValidConnection(Connection) method to validate a connection. If an exception is returned that means the connection is invalid. This overrides the check-valid-connection-sql element
validateOnMatch : BOOLEAN = false	The validate-on-match element specifies if connection validation should be done when a connection factory attempts to match a managed connection. This is typically exclusive to the use of background validation
validationMillis : INTEGER = -1	The background-validation-millis element specifies the amount of time, in milliseconds, that background validation will run. Changing this value can be done only on disabled datasource, requires a server restart otherwise

Hidden Properties	
* createOrder : INTEGER = 50	The order of the step in the step list for the create operation.
* createScript : STRING	Classpath to the script that is uploaded and executed on the generic container for the create operation.
* createVerb : STRING = Create	Create Verb
* destroyOrder : INTEGER = 40	The order of the step in the step list for the destroy operation.
* destroyVerb : STRING = Destroy	Destroy Verb
* libraries : LIST_OF_STRING = [jboss/dm/ds/datasource-lib.py]	Libraries
* modifyOrder : INTEGER = 50	The order of the step in the step list for the modify operation.
* modifyVerb : STRING = Modify	Modify Verb
* noopOrder : INTEGER = 50	The order of the step in the step list for the noop operation.
* noopVerb : STRING = Modify	Noop Verb
destroyScript : STRING	Classpath to the script that is uploaded and executed on the generic container for the destroy operation.
inspectScript : STRING	Classpath to the script used to inspect the generic container.
modifyScript : STRING	Classpath to the script that is uploaded and executed on the generic container for the modify operation.
noopScript : STRING	Classpath to the script that is uploaded and executed on the generic container for the noop operation.


jbossdm.CliBasedContainer








Virtual Type

Type Hierarchy udm.BaseContainer >> udm.BaseConfigurationItem

Interfaces udm.Taggable, jbossdm.CliManagedContainer, jbossdm.ApplicationContainer, jbossdm.ResourceContainer, udm.ConfigurationItem, jbossdm.CliManagingContainer, udm.Container, overthere.HostContainer

JBoss Cli Managed Container

Parent	
 * host : CI <overthere.Host>	Host

Public Properties	
 *	cliScriptPrefix : STRING = jboss-cli JBoss CLI script prefix. Depending on the host, either an '.sh' or '.bat' will be appended to get the cli script name.
 *	home : STRING JBoss home directory
	adminHostAddress : STRING = localhost Host which is used to login to JBoss Native Administration, default is localhost
	enableDaemon : BOOLEAN = true Connection to CLI is setup using a daemon. Set to false if host connection does not support streaming.
	password : STRING Password which is used to login to JBoss Native Administration.
	port : INTEGER = 9999 TCP port which is used to login to JBoss Native Administration, default is 9999
	tags : SET_OF_STRING If set, only deployables with the same tag will be automatically mapped to this container.
	username : STRING Username which is used to login to JBoss Native Administration.
Hidden Properties	
*	libraries : LIST_OF_STRING = [jboss/dm/library/runtime.py] List of python library scripts that should be automatically loaded when using a JBoss CLI script

jbossdm.CliManagedDeployed

Virtual Type

Type Hierarchy udm.BaseDeployed >> udm.BaseConfigurationItem

Interfaces udm.Deployed, udm.ConfigurationItem

Base for all deployed that utilize the JBoss Cli for configuration

Parent	
*	container : CI <udm.Container> The container on which this deployed runs.
Public Properties	
	deployable : CI <udm.Deployable> The deployable that this deployed is derived from.

Hidden Properties	
* createOrder : INTEGER = 50	The order of the step in the step list for the create operation.
* createScript : STRING	Classpath to the script that is uploaded and executed on the generic container for the create operation.
* createVerb : STRING = Create	Create Verb
* destroyOrder : INTEGER = 40	The order of the step in the step list for the destroy operation.
* destroyVerb : STRING = Destroy	Destroy Verb
* modifyOrder : INTEGER = 50	The order of the step in the step list for the modify operation.
* modifyVerb : STRING = Modify	Modify Verb
* noopOrder : INTEGER = 50	The order of the step in the step list for the noop operation.
* noopVerb : STRING = Modify	Noop Verb
destroyScript : STRING	Classpath to the script that is uploaded and executed on the generic container for the destroy operation.
inspectScript : STRING	Classpath to the script used to inspect the generic container.
libraries : LIST_OF_STRING	List of python library scripts that should be automatically loaded when using a JBoss CLI script.
modifyScript : STRING	Classpath to the script that is uploaded and executed on the generic container for the modify operation.
noopScript : STRING	Classpath to the script that is uploaded and executed on the generic container for the noop operation.

jbosssdm.CliManagedDeployedArtifact

Virtual Type

Type Hierarchy [jbosssdm.CliManagedDeployed](#) >> [udm.BaseDeployed](#) >> [udm.BaseConfigurationItem](#)

Interfaces [udm.Artifact](#), [udm.Deployed](#), [udm.ConfigurationItem](#), [udm.DerivedArtifact](#)

Base for all deployed artifacts that utilize the JBoss Cli for configuration

Parent	
* container : CI < udm.Container >	The container on which this deployed runs.
Public Properties	
deployable : CI < udm.Deployable >	The deployable that this deployed is derived from.
placeholders : MAP_STRING_STRING	A Map containing all the placeholders mapped to their values. Special values are <ignore> or <empty>

Hidden Properties	
* createOrder : INTEGER = 50	The order of the step in the step list for the create operation.
* createScript : STRING	Classpath to the script that is uploaded and executed on the generic container for the create operation.
* createVerb : STRING = Create	Create Verb
* destroyOrder : INTEGER = 40	The order of the step in the step list for the destroy operation.
* destroyVerb : STRING = Destroy	Destroy Verb
* modifyOrder : INTEGER = 50	The order of the step in the step list for the modify operation.
* modifyVerb : STRING = Modify	Modify Verb
* noopOrder : INTEGER = 50	The order of the step in the step list for the noop operation.
* noopVerb : STRING = Modify	Noop Verb
destroyScript : STRING	Classpath to the script that is uploaded and executed on the generic container for the destroy operation.
inspectScript : STRING	Classpath to the script used to inspect the generic container.
libraries : LIST_OF_STRING	List of python library scripts that should be automatically loaded when using a JBoss CLI script.
modifyScript : STRING	Classpath to the script that is uploaded and executed on the generic container for the modify operation.
noopScript : STRING	Classpath to the script that is uploaded and executed on the generic container for the noop operation.

jbosssdm.DataSource

Type Hierarchy [jbosssdm.JeeDataSource](#) >> [jbosssdm.BaseDataSource](#) >> [jbosssdm.CliManagedDeployed](#) >> [udm.BaseDeployed](#) >> [udm.BaseConfigurationItem](#)

Interfaces [udm.Deployed](#), [udm.ConfigurationItem](#)

DataSource. This deployed is used when a [jbosssdm.DataSourceSpec](#) is specified in a package.

Parent	
* container : CI < udm.Container >	The container on which this deployed runs.

Public Properties	
* <code>connectionUrl</code> : <code>STRING</code>	The JDBC driver connection URL
* <code>driverName</code> : <code>STRING</code>	Defines the JDBC driver the datasource should use. It is a symbolic name matching the the name of installed driver. In case the driver is deployed as jar, the name is the name of deployment unit.
* <code>jndiName</code> : <code>STRING</code>	Specifies the JNDI name for the datasource
<code>backgroundValidation</code> : <code>BOOLEAN</code> = <code>false</code>	An element to specify that connections should be validated on a background thread versus being validated prior to use. Changing this value can be done only on disabled datasource, requires a server restart otherwise.
<code>checkValidSql</code> : <code>STRING</code>	Specify an SQL statement to check validity of a pool connection. This may be called when managed connection is obtained from the pool
<code>connectionProperties</code> : <code>MAP_STRING_STRING</code>	JDBC connection properties
<code>deployable</code> : <code>CI<udm.Deployable></code>	The deployable that this deployed is derived from.
<code>exceptionSorter</code> : <code>STRING</code>	An <code>org.jboss.jca.adapters.jdbc.ExceptionSorter</code> that provides an <code>isExceptionFatal(SQLException)</code> method to validate if an exception should broadcast an error
<code>maxPoolSize</code> : <code>INTEGER</code> = <code>0</code>	The max-pool-size element specifies the maximum number of connections for a pool. No more connections will be created in each sub-pool
<code>minPoolSize</code> : <code>INTEGER</code> = <code>0</code>	The min-pool-size element specifies the minimum number of connections for a pool
<code>newConnectionSql</code> : <code>STRING</code>	Specifies an SQL statement to execute whenever a connection is added to the connection pool
<code>password</code> : <code>STRING</code>	Specifies the password used when creating a new connection
<code>prefillEnabled</code> : <code>BOOLEAN</code> = <code>false</code>	Should the pool be prefilled. Changing this value can be done only on disabled datasource, requires a server restart otherwise.
<code>securityDomain</code> : <code>STRING</code>	Specifies the security domain which defines the <code>javax.security.auth.Subject</code> that are used to distinguish connections in the pool
<code>sharePreparedStatements</code> : <code>BOOLEAN</code>	Whether to share prepared statements, i.e. whether asking for same statement twice without closing uses the same underlying prepared statement
<code>staleConnectionChecker</code> : <code>STRING</code>	An <code>org.jboss.jca.adapters.jdbc.StaleConnectionChecker</code> that provides an <code>isStaleConnection(SQLException)</code> method which if it returns true will wrap the exception in an <code>org.jboss.jca.adapters.jdbc.StaleConnectionException</code>
<code>statementCacheSize</code> : <code>INTEGER</code> = <code>-1</code>	The number of prepared statements per connection in an LRU cache
<code>strictMinimum</code> : <code>BOOLEAN</code> = <code>false</code>	Specifies if the min-pool-size should be considered strictly
<code>transactionIsolation</code> : <code>STRING</code>	Set the <code>java.sql.Connection</code> transaction isolation level. Valid values are: <code>TRANSACTION_READ_UNCOMMITTED</code> , <code>TRANSACTION_READ_COMMITTED</code> , <code>TRANSACTION_REPEATABLE_READ</code> , <code>TRANSACTION_SERIALIZABLE</code> and <code>TRANSACTION_NONE</code>
<code>useCcm</code> : <code>BOOLEAN</code> = <code>false</code>	Enable the use of a cached connection manager
<code>useJta</code> : <code>BOOLEAN</code> = <code>false</code>	Enable JTA integration
<code>username</code> : <code>STRING</code>	Specify the user name used when creating a new connection
<code>validConnectionChecker</code> : <code>STRING</code>	An <code>org.jboss.jca.adapters.jdbc.ValidConnectionChecker</code> that provides an <code>isValidConnection(Connection)</code> method to validate a connection. If an exception is returned that means the connection is invalid. This overrides the <code>check-valid-</code>

connection-sql element
validateOnMatch : BOOLEAN = false The validate-on-match element specifies if connection validation should be done when a connection factory attempts to match a managed connection. This is typically exclusive to the use of background validation
validationMillis : INTEGER = -1 The background-validation-millis element specifies the amount of time, in milliseconds, that background validation will run. Changing this value can be done only on disabled datasource, requires a server restart otherwise
Hidden Properties
* createOrder : INTEGER = 50 The order of the step in the step list for the create operation.
* createScript : STRING = jboss/dm/ds/create-datasource.py Create Script
* createVerb : STRING = Create Create Verb
* destroyOrder : INTEGER = 40 The order of the step in the step list for the destroy operation.
* destroyScript : STRING = jboss/dm/ds/destroy-datasource.py Destroy Script
* destroyVerb : STRING = Destroy Destroy Verb
* inspectScript : STRING = jboss/dm/ds/inspect-datasource.py Inspect Script
* libraries : LIST_OF_STRING = [jboss/dm/ds/datasource-lib.py] Libraries
* modifyOrder : INTEGER = 50 The order of the step in the step list for the modify operation.
* modifyVerb : STRING = Modify Modify Verb
* noopOrder : INTEGER = 50 The order of the step in the step list for the noop operation.
* noopVerb : STRING = Modify Noop Verb
modifyScript : STRING Classpath to the script that is uploaded and executed on the generic container for the modify operation.
noopScript : STRING Classpath to the script that is uploaded and executed on the generic container for the noop operation.

jbossdm.DataSourceSpec

Type Hierarchy jee.DataSourceSpec >> jee.JndiResourceSpec >> jee.ResourceSpec >> udm.BaseDeployable >> udm.BaseConfigurationItem

Interfaces udm.Taggable, udm.Deployable, udm.ConfigurationItem

DataSource. This deployed is used when a jbossdm.DataSourceSpec is specified in a package. (deployable)

Public Properties	
backgroundValidation : STRING	An element to specify that connections should be validated on a background thread versus being validated prior to use. Changing this value can be done only on disabled datasource, requires a server restart otherwise. (boolean)
checkValidSql : STRING	Specify an SQL statement to check validity of a pool connection. This may be called when managed connection is obtained from the pool (string)
connectionProperties : MAP_STRING_STRING	JDBC connection properties (map_string_string)
connectionUrl : STRING	The JDBC driver connection URL (string)
driverName : STRING	Defines the JDBC driver the datasource should use. It is a symbolic name matching the the name of installed driver. In case the driver is deployed as jar, the name is the name of deployment unit. (string)
exceptionSorter : STRING	An org.jboss.jca.adapters.jdbc.ExceptionSorter that provides an isExceptionFatal(SQLException) method to validate if an exception should broadcast an error (string)
jndiName : STRING	Specifies the JNDI name for the datasource (string)
maxPoolSize : STRING	The max-pool-size element specifies the maximum number of connections for a pool. No more connections will be created in each sub-pool (integer)
minPoolSize : STRING	The min-pool-size element specifies the minimum number of connections for a pool (integer)
newConnectionSql : STRING	Specifies an SQL statement to execute whenever a connection is added to the connection pool (string)
password : STRING	Specifies the password used when creating a new connection (string)
prefillEnabled : STRING	Should the pool be prefilled. Changing this value can be done only on disabled datasource, requires a server restart otherwise. (boolean)
securityDomain : STRING	Specifies the security domain which defines the javax.security.auth.Subject that are used to distinguish connections in the pool (string)
sharePreparedStatements : STRING	Whether to share prepared statements, i.e. whether asking for same statement twice without closing uses the same underlying prepared statement (boolean)
staleConnectionChecker : STRING	An org.jboss.jca.adapters.jdbc.StaleConnectionChecker that provides an isStaleConnection(SQLException) method which if it returns true will wrap the exception in an org.jboss.jca.adapters.jdbc.StaleConnectionException (string)
statementCacheSize : STRING	The number of prepared statements per connection in an LRU cache (integer)
strictMinimum : STRING	Specifies if the min-pool-size should be considered strictly (boolean)
tags : SET_OF_STRING	If set, this deployable will only be mapped automatically to containers with the same tag.
transactionIsolation : STRING	Set the java.sql.Connection transaction isolation level. Valid values are: TRANSACTION_READ_UNCOMMITTED, TRANSACTION_READ_COMMITTED, TRANSACTION_REPEATABLE_READ, TRANSACTION_SERIALIZABLE and TRANSACTION_NONE (string)
useCcm : STRING	Enable the use of a cached connection manager (boolean)
useJta : STRING	Enable JTA integration (boolean)
username : STRING	Specify the user name used when creating a new connection (string)
validConnectionChecker : STRING	An org.jboss.jca.adapters.jdbc.ValidConnectionChecker that provides an

isValidConnection(Connection) method to validate a connection. If an exception is returned that means the connection is invalid. This overrides the check-valid-connection-sql element (string)

validateOnMatch : **STRING**

The validate-on-match element specifies if connection validation should be done when a connection factory attempts to match a managed connection. This is typically exclusive to the use of background validation (boolean)

validationMillis : **STRING**

The background-validation-millis element specifies the amount of time, in milliseconds, that background validation will run. Changing this value can be done only on disabled datasource, requires a server restart otherwise (integer)

jbosssdm.Domain









Type Hierarchy [jbosssdm.CliBasedContainer](#) >> [udm.BaseContainer](#) >>

[udm.BaseConfigurationItem](#)

Interfaces

[udm.Taggable](#), [jbosssdm.CliManagedContainer](#),
[jbosssdm.ApplicationContainer](#), [jbosssdm.ResourceContainer](#),
[udm.ConfigurationItem](#), [udm.Container](#), [jbosssdm.CliManagingContainer](#),
[overthere.HostContainer](#)

Description unavailable

Parent	
 *	host : CI < overthere.Host > Host
Children	
	profiles : LIST_OF_CI < jbosssdm.Profile > Profiles defined in domain
	serverGroups : LIST_OF_CI < jbosssdm.ServerGroup > Server groups defined in domain
Public Properties	
 *	cliScriptPrefix : STRING = jboss-cli JBoss CLI script prefix. Depending on the host, either an '.sh' or '.bat' will be appended to get the cli script name.
 *	home : STRING JBoss home directory
	adminHostAddress : STRING = localhost Host which is used to login to JBoss Native Administration, default is localhost
	enableDaemon : BOOLEAN = true Connection to CLI is setup using a daemon. Set to false if host connection does not support streaming.
	password : STRING Password which is used to login to JBoss Native Administration.
	port : INTEGER = 9999 TCP port which is used to login to JBoss Native Administration, default is 9999
	tags : SET_OF_STRING If set, only deployables with the same tag will be automatically mapped to this container.
	username : STRING Username which is used to login to JBoss Native Administration.
Hidden Properties	
*	libraries : LIST_OF_STRING = [jboss/dm/library/runtime.py] List of python library scripts that should be automatically loaded when using a JBoss CLI script

jbosssdm.Ear

Type Hierarchy [jee.Ear](#) >> [udm.BaseDeployableArchiveArtifact](#) >>
[udm.BaseDeployableFileArtifact](#) >> [udm.BaseDeployableArtifact](#) >>
[udm.BaseDeployable](#) >> [udm.BaseConfigurationItem](#)

Interfaces

[udm.Taggable](#), [udm.Deployable](#), [udm.SourceArtifact](#), [udm.ArchiveArtifact](#),
[udm.Artifact](#), [udm.DeployableArtifact](#), [udm.ConfigurationItem](#),
[udm.FileArtifact](#)

A JEE EAR archive

Public Properties
excludeFileNamesRegex : STRING Regular expression that matches file names that must be excluded from scanning
placeholders : SET_OF_STRING Placeholders detected in this artifact
scanPlaceholders : BOOLEAN = false Whether to scan this artifact for placeholders when it is imported
tags : SET_OF_STRING If set, this deployable will only be mapped automatically to containers with the same tag.
Hidden Properties
* textFileNamesRegex : STRING = .+\. (cfg conf config ini properties props txt asp aspx htm html jsf jsp xht xhtml sql xml xsd xsl xslt) Regular expression that matches file names of text files

jbosssdm.EarModule

Type Hierarchy `jbosssdm.CliManagedDeployedArtifact >> jbosssdm.CliManagedDeployed >> udm.BaseDeployed >> udm.BaseConfigurationItem`

Interfaces `udm.Artifact, udm.Deployed, udm.ConfigurationItem, udm.DerivedArtifact`

Ear with values configured for a deployment

Parent
* container : CI <udm.Container> The container on which this deployed runs.
Public Properties
deployable : CI <udm.Deployable> The deployable that this deployed is derived from.
placeholders : MAP_STRING_STRING A Map containing all the placeholders mapped to their values. Special values are <ignore> or <empty>
Hidden Properties
* createOrder : INTEGER = 50 The order of the step in the step list for the create operation.
* createScript : STRING = jboss/dm/application/install-ear.py Create Script
* createVerb : STRING = Create Create Verb
* destroyOrder : INTEGER = 40 The order of the step in the step list for the destroy operation.
* destroyScript : STRING = jboss/dm/application/uninstall-ear.py Destroy Script
* destroyVerb : STRING = Destroy Destroy Verb
* inspectScript : STRING = jboss/dm/application/inspect-ear.py Inspect Script
* libraries : LIST_OF_STRING = [jboss/dm/application/application-lib.py] Libraries
* modifyOrder : INTEGER = 50 The order of the step in the step list for the modify operation.
* modifyVerb : STRING = Modify Modify Verb
* noopOrder : INTEGER = 50 The order of the step in the step list for the noop operation.
* noopVerb : STRING = Modify Noop Verb
modifyScript : STRING Classpath to the script that is uploaded and executed on the generic container for the modify operation.
noopScript : STRING Classpath to the script that is uploaded and executed on the generic container for the noop operation.

jbossdm.JeeDataSource

Type Hierarchy [jbossdm.BaseDataSource](#) >> [jbossdm.CliManagedDeployed](#) >>
udm.BaseDeployed >> udm.BaseConfigurationItem

Interfaces udm.Deployed, udm.ConfigurationItem

Datasource. This deployed is used when a jee.DataSourceSpec is specified in a package.

Public Properties	
* <code>connectionUrl</code> : <code>STRING</code>	The JDBC driver connection URL
* <code>driverName</code> : <code>STRING</code>	Defines the JDBC driver the datasource should use. It is a symbolic name matching the the name of installed driver. In case the driver is deployed as jar, the name is the name of deployment unit.
* <code>jndiName</code> : <code>STRING</code>	Specifies the JNDI name for the datasource
<code>backgroundValidation</code> : <code>BOOLEAN</code> = <code>false</code>	An element to specify that connections should be validated on a background thread versus being validated prior to use. Changing this value can be done only on disabled datasource, requires a server restart otherwise.
<code>checkValidSql</code> : <code>STRING</code>	Specify an SQL statement to check validity of a pool connection. This may be called when managed connection is obtained from the pool
<code>connectionProperties</code> : <code>MAP_STRING_STRING</code>	JDBC connection properties
<code>deployable</code> : <code>CI<udm.Deployable></code>	The deployable that this deployed is derived from.
<code>exceptionSorter</code> : <code>STRING</code>	An <code>org.jboss.jca.adapters.jdbc.ExceptionSorter</code> that provides an <code>isExceptionFatal(SQLException)</code> method to validate if an exception should broadcast an error
<code>maxPoolSize</code> : <code>INTEGER</code> = <code>0</code>	The max-pool-size element specifies the maximum number of connections for a pool. No more connections will be created in each sub-pool
<code>minPoolSize</code> : <code>INTEGER</code> = <code>0</code>	The min-pool-size element specifies the minimum number of connections for a pool
<code>newConnectionSql</code> : <code>STRING</code>	Specifies an SQL statement to execute whenever a connection is added to the connection pool
<code>password</code> : <code>STRING</code>	Specifies the password used when creating a new connection
<code>prefillEnabled</code> : <code>BOOLEAN</code> = <code>false</code>	Should the pool be prefilled. Changing this value can be done only on disabled datasource, requires a server restart otherwise.
<code>securityDomain</code> : <code>STRING</code>	Specifies the security domain which defines the <code>javax.security.auth.Subject</code> that are used to distinguish connections in the pool
<code>sharePreparedStatements</code> : <code>BOOLEAN</code>	Whether to share prepared statements, i.e. whether asking for same statement twice without closing uses the same underlying prepared statement
<code>staleConnectionChecker</code> : <code>STRING</code>	An <code>org.jboss.jca.adapters.jdbc.StaleConnectionChecker</code> that provides an <code>isStaleConnection(SQLException)</code> method which if it returns true will wrap the exception in an <code>org.jboss.jca.adapters.jdbc.StaleConnectionException</code>
<code>statementCacheSize</code> : <code>INTEGER</code> = <code>-1</code>	The number of prepared statements per connection in an LRU cache
<code>strictMinimum</code> : <code>BOOLEAN</code> = <code>false</code>	Specifies if the min-pool-size should be considered strictly
<code>transactionIsolation</code> : <code>STRING</code>	Set the <code>java.sql.Connection</code> transaction isolation level. Valid values are: <code>TRANSACTION_READ_UNCOMMITTED</code> , <code>TRANSACTION_READ_COMMITTED</code> , <code>TRANSACTION_REPEATABLE_READ</code> , <code>TRANSACTION_SERIALIZABLE</code> and <code>TRANSACTION_NONE</code>
<code>useCcm</code> : <code>BOOLEAN</code> = <code>false</code>	Enable the use of a cached connection manager
<code>useJta</code> : <code>BOOLEAN</code> = <code>false</code>	Enable JTA integration
<code>username</code> : <code>STRING</code>	Specify the user name used when creating a new connection
<code>validConnectionChecker</code> : <code>STRING</code>	An <code>org.jboss.jca.adapters.jdbc.ValidConnectionChecker</code> that provides an <code>isValidConnection(Connection)</code> method to validate a connection. If an exception is returned that means the connection is invalid. This overrides the <code>check-valid-</code>

connection-sql element
validateOnMatch : BOOLEAN = false The validate-on-match element specifies if connection validation should be done when a connection factory attempts to match a managed connection. This is typically exclusive to the use of background validation
validationMillis : INTEGER = -1 The background-validation-millis element specifies the amount of time, in milliseconds, that background validation will run. Changing this value can be done only on disabled datasource, requires a server restart otherwise
Hidden Properties
* createOrder : INTEGER = 50 The order of the step in the step list for the create operation.
* createScript : STRING = jboss/dm/ds/create-datasource.py Create Script
* createVerb : STRING = Create Create Verb
* destroyOrder : INTEGER = 40 The order of the step in the step list for the destroy operation.
* destroyScript : STRING = jboss/dm/ds/destroy-datasource.py Destroy Script
* destroyVerb : STRING = Destroy Destroy Verb
* inspectScript : STRING = jboss/dm/ds/inspect-datasource.py Inspect Script
* libraries : LIST_OF_STRING = [jboss/dm/ds/datasource-lib.py] Libraries
* modifyOrder : INTEGER = 50 The order of the step in the step list for the modify operation.
* modifyVerb : STRING = Modify Modify Verb
* noopOrder : INTEGER = 50 The order of the step in the step list for the noop operation.
* noopVerb : STRING = Modify Noop Verb
modifyScript : STRING Classpath to the script that is uploaded and executed on the generic container for the modify operation.
noopScript : STRING Classpath to the script that is uploaded and executed on the generic container for the noop operation.

jbosssdm.JeeXaDataSource

Type Hierarchy [jbosssdm.BaseDataSource](#) >> [jbosssdm.CliManagedDeployed](#) >> [udm.BaseDeployed](#) >> [udm.BaseConfigurationItem](#)

Interfaces [udm.Deployed](#), [udm.ConfigurationItem](#)

XA DataSource. This deployed is used when a [jee.DataSourceSpec](#) is specified in a package.

Parent
* container : CI < udm.Container > The container on which this deployed runs.

Public Properties	
* driverName : STRING	Defines the JDBC driver the datasource should use. It is a symbolic name matching the the name of installed driver. In case the driver is deployed as jar, the name is the name of deployment unit.
* jndiName : STRING	Specifies the JNDI name for the datasource
* xaProperties : MAP_STRING_STRING	Properties to assign to the XADataSource implementation class. At least one XA property is required (i.e. url)
backgroundValidation : BOOLEAN = false	An element to specify that connections should be validated on a background thread versus being validated prior to use. Changing this value can be done only on disabled datasource, requires a server restart otherwise.
checkValidSql : STRING	Specify an SQL statement to check validity of a pool connection. This may be called when managed connection is obtained from the pool
deployable : CI<udm.Deployable>	The deployable that this deployed is derived from.
exceptionSorter : STRING	An org.jboss.jca.adapters.jdbc.ExceptionSorter that provides an isExceptionFatal(SQLException) method to validate if an exception should broadcast an error
interleave : BOOLEAN = false	An element to enable interleaving for XA connections
maxPoolSize : INTEGER = 0	The max-pool-size element specifies the maximum number of connections for a pool. No more connections will be created in each sub-pool
minPoolSize : INTEGER = 0	The min-pool-size element specifies the minimum number of connections for a pool
newConnectionSql : STRING	Specifies an SQL statement to execute whenever a connection is added to the connection pool
padXid : BOOLEAN = false	Should the Xid be padded
password : STRING	Specifies the password used when creating a new connection
prefillEnabled : BOOLEAN = false	Should the pool be prefilled. Changing this value can be done only on disabled datasource, requires a server restart otherwise.
sameRmOverride : BOOLEAN = false	The is-same-rm-override element allows one to unconditionally set whether the javax.transaction.xa.XAResource.isSameRM(XAResource) returns true or false
securityDomain : STRING	Specifies the security domain which defines the javax.security.auth.Subject that are used to distinguish connections in the pool
sharePreparedStatements : BOOLEAN	Whether to share prepared statements, i.e. whether asking for same statement twice without closing uses the same underlying prepared statement
staleConnectionChecker : STRING	An org.jboss.jca.adapters.jdbc.StaleConnectionChecker that provides an isStaleConnection(SQLException) method which if it returns true will wrap the exception in an org.jboss.jca.adapters.jdbc.StaleConnectionException
statementCacheSize : INTEGER = -1	The number of prepared statements per connection in an LRU cache
strictMinimum : BOOLEAN = false	Specifies if the min-pool-size should be considered strictly
transactionIsolation : STRING	Set the java.sql.Connection transaction isolation level. Valid values are: TRANSACTION_READ_UNCOMMITTED, TRANSACTION_READ_COMMITTED, TRANSACTION_REPEATABLE_READ, TRANSACTION_SERIALIZABLE and TRANSACTION_NONE
username : STRING	Specify the user name used when creating a new connection
validConnectionChecker : STRING	An org.jboss.jca.adapters.jdbc.ValidConnectionChecker that provides an

isValidConnection(Connection) method to validate a connection. If an exception is returned that means the connection is invalid. This overrides the check-valid-connection-sql element

validateOnMatch : **BOOLEAN** = false

The validate-on-match element specifies if connection validation should be done when a connection factory attempts to match a managed connection. This is typically exclusive to the use of background validation

validationMillis : **INTEGER** = -1

The background-validation-millis element specifies the amount of time, in milliseconds, that background validation will run. Changing this value can be done only on disabled datasource, requires a server restart otherwise

wrapXa : **BOOLEAN** = false

Should the XAResource instances be wrapped in a org.jboss.tm.XAResourceWrapper instance

Hidden Properties

* **createOrder** : **INTEGER** = 50

The order of the step in the step list for the create operation.

* **createScript** : **STRING** = jboss/dm/ds/create-xa-datasource.py

Create Script

* **createVerb** : **STRING** = Create

Create Verb

* **destroyOrder** : **INTEGER** = 40

The order of the step in the step list for the destroy operation.

* **destroyScript** : **STRING** = jboss/dm/ds/destroy-xa-datasource.py

Destroy Script

* **destroyVerb** : **STRING** = Destroy

Destroy Verb

* **inspectScript** : **STRING** = jboss/dm/ds/inspect-xa-datasource.py

Inspect Script

* **libraries** : **LIST_OF_STRING** = [jboss/dm/ds/datasource-lib.py]

Libraries

* **modifyOrder** : **INTEGER** = 50

The order of the step in the step list for the modify operation.

* **modifyVerb** : **STRING** = Modify

Modify Verb

* **noopOrder** : **INTEGER** = 50

The order of the step in the step list for the noop operation.

* **noopVerb** : **STRING** = Modify

Noop Verb

modifyScript : **STRING**

Classpath to the script that is uploaded and executed on the generic container for the modify operation.

noopScript : **STRING**

Classpath to the script that is uploaded and executed on the generic container for the noop operation.

jbosssdm.Profile

Type Hierarchy udm.BaseContainer >> udm.BaseConfigurationItem

Interfaces udm.Taggable, jbosssdm.CliManagedContainer, jbosssdm.ResourceContainer, udm.ConfigurationItem, udm.Container, overthere.HostContainer

JBoss Profile

Parent

* **domain** : **CI** <jbosssdm.Domain >

Domain to which the server group belongs.

Public Properties

tags : **SET_OF_STRING**

If set, only deployables with the same tag will be automatically mapped to this container.

jbosssdm.Queue

Type Hierarchy `jbossdm.CliManagedDeployed >> udm.BaseDeployed >> udm.BaseConfigurationItem`

Interfaces `udm.Deployed, udm.ConfigurationItem`

A Jboss Queue

Parent	
* container : <code>CI<udm.Container></code>	The container on which this deployed runs.
Public Properties	
* jndiName : <code>STRING</code>	(Comma separated list) The jndi names the queue will be bound to.
deployable : <code>CI<udm.Deployable></code>	The deployable that this deployed is derived from.
durable : <code>BOOLEAN = true</code>	Whether the queue is durable or not
selector : <code>STRING</code>	The queue selector
Hidden Properties	
* createOrder : <code>INTEGER = 50</code>	The order of the step in the step list for the create operation.
* createScript : <code>STRING = jboss/dm/jms/create-queue.py</code>	Create Script
* createVerb : <code>STRING = Create</code>	Create Verb
* destroyOrder : <code>INTEGER = 40</code>	The order of the step in the step list for the destroy operation.
* destroyScript : <code>STRING = jboss/dm/jms/destroy-queue.py</code>	Destroy Script
* destroyVerb : <code>STRING = Destroy</code>	Destroy Verb
* inspectScript : <code>STRING = jboss/dm/jms/inspect-queue.py</code>	Inspect Script
* modifyOrder : <code>INTEGER = 50</code>	The order of the step in the step list for the modify operation.
* modifyVerb : <code>STRING = Modify</code>	Modify Verb
* noopOrder : <code>INTEGER = 50</code>	The order of the step in the step list for the noop operation.
* noopVerb : <code>STRING = Modify</code>	Noop Verb
libraries : <code>LIST_OF_STRING</code>	List of python library scripts that should be automatically loaded when using a JBoss CLI script.
modifyScript : <code>STRING</code>	Classpath to the script that is uploaded and executed on the generic container for the modify operation.
noopScript : <code>STRING</code>	Classpath to the script that is uploaded and executed on the generic container for the noop operation.

jbossdm.QueueSpec

Type Hierarchy `jee.QueueSpec >> jee.JndiResourceSpec >> jee.ResourceSpec >> udm.BaseDeployable >> udm.BaseConfigurationItem`

Interfaces `udm.Taggable, udm.Deployable, udm.ConfigurationItem`

A Queue

Public Properties	
durable :	STRING Whether the queue is durable or not (boolean)
jndiName :	STRING (Comma separated list) The jndi names the queue will be bound to. (string)
selector :	STRING The queue selector (string)
tags :	SET_OF_STRING If set, this deployable will only be mapped automatically to containers with the same tag.

jbosssdm.ServerGroup

Type Hierarchy udm.BaseContainer >> udm.BaseConfigurationItem
Interfaces udm.Taggable, jbosssdm.CliManagedContainer, jbosssdm.ApplicationContainer, udm.ConfigurationItem, udm.Container, overthere.HostContainer

JBoss Server Group

Parent	
* domain :	CI <jbosssdm.Domain > Domain to which the server group belongs.
Public Properties	
tags :	SET_OF_STRING If set, only deployables with the same tag will be automatically mapped to this container.

jbosssdm.StandaloneServer

Type Hierarchy jbosssdm.CliBasedContainer >> udm.BaseContainer >> udm.BaseConfigurationItem
Interfaces udm.Taggable, jbosssdm.CliManagedContainer, jbosssdm.ApplicationContainer, jbosssdm.ResourceContainer, udm.ConfigurationItem, udm.Container, jbosssdm.CliManagingContainer, overthere.HostContainer

JBoss Standalone Server

Parent	
* host :	CI <overthere.Host> Host
Public Properties	
* cliScriptPrefix :	STRING = jboss-cli JBoss CLI script prefix. Depending on the host, either an '.sh' or '.bat' will be appended to get the cli script name.
* home :	STRING JBoss home directory
adminHostAddress :	STRING = localhost Host which is used to login to JBoss Native Administration, default is localhost
enableDaemon :	BOOLEAN = true Connection to CLI is setup using a daemon. Set to false if host connection does not support streaming.
password :	STRING Password which is used to login to JBoss Native Administration.
port :	INTEGER = 9999 TCP port which is used to login to JBoss Native Administration, default is 9999
tags :	SET_OF_STRING If set, only deployables with the same tag will be automatically mapped to this container.
username :	STRING Username which is used to login to JBoss Native Administration.
Hidden Properties	
* libraries :	LIST_OF_STRING = [jboss/dm/library/runtime.py] List of python library scripts that should be automatically loaded when using a JBoss CLI script

jbosssdm.Topic

Type Hierarchy `jbosssdm.CliManagedDeployed` >> `udm.BaseDeployed` >>
`udm.BaseConfigurationItem`

Interfaces `udm.Deployed`, `udm.ConfigurationItem`

A JBoss topic

Parent	
* container :	<code>CI<udm.Container></code> The container on which this deployed runs.
Public Properties	
* jndiName :	<code>STRING</code> (Comma separated list) The jndi names the topic will be bound to.
deployable :	<code>CI<udm.Deployable></code> The deployable that this deployed is derived from.
Hidden Properties	
* createOrder :	<code>INTEGER</code> = 50 The order of the step in the step list for the create operation.
* createScript :	<code>STRING</code> = <code>jboss/dm/jms/create-topic.py</code> Create Script
* createVerb :	<code>STRING</code> = <code>Create</code> Create Verb
* destroyOrder :	<code>INTEGER</code> = 40 The order of the step in the step list for the destroy operation.
* destroyScript :	<code>STRING</code> = <code>jboss/dm/jms/destroy-topic.py</code> Destroy Script
* destroyVerb :	<code>STRING</code> = <code>Destroy</code> Destroy Verb
* inspectScript :	<code>STRING</code> = <code>jboss/dm/jms/inspect-topic.py</code> Inspect Script
* modifyOrder :	<code>INTEGER</code> = 50 The order of the step in the step list for the modify operation.
* modifyVerb :	<code>STRING</code> = <code>Modify</code> Modify Verb
* noopOrder :	<code>INTEGER</code> = 50 The order of the step in the step list for the noop operation.
* noopVerb :	<code>STRING</code> = <code>Modify</code> Noop Verb
libraries :	<code>LIST_OF_STRING</code> List of python library scripts that should be automatically loaded when using a JBoss CLI script.
modifyScript :	<code>STRING</code> Classpath to the script that is uploaded and executed on the generic container for the modify operation.
noopScript :	<code>STRING</code> Classpath to the script that is uploaded and executed on the generic container for the noop operation.

jbosssdm.TopicSpec

Type Hierarchy `jee.TopicSpec` >> `jee.JndiResourceSpec` >> `jee.ResourceSpec` >>
`udm.BaseDeployable` >> `udm.BaseConfigurationItem`

Interfaces `udm.Taggable`, `udm.Deployable`, `udm.ConfigurationItem`

A Topic

Public Properties	
jndiName :	<code>STRING</code> (Comma separated list) The jndi names the topic will be bound to. (string)
tags :	<code>SET_OF_STRING</code> If set, this deployable will only be mapped automatically to containers with the same tag.

jbossdm.War

Type Hierarchy `jee.War >> udm.BaseDeployableArchiveArtifact >> udm.BaseDeployableFileArtifact >> udm.BaseDeployableArtifact >> udm.BaseDeployable >> udm.BaseConfigurationItem`

Interfaces `udm.Taggable, udm.Deployable, udm.SourceArtifact, udm.ArchiveArtifact, udm.Artifact, udm.DeployableArtifact, udm.ConfigurationItem, udm.FileArtifact`

A JEE WAR archive

Public Properties	
contextRoot :	STRING Context root for the web application (string)
excludeFileNamesRegex :	STRING Regular expression that matches file names that must be excluded from scanning
placeholders :	SET_OF_STRING Placeholders detected in this artifact
scanPlaceholders :	BOOLEAN = false Whether to scan this artifact for placeholders when it is imported
tags :	SET_OF_STRING If set, this deployable will only be mapped automatically to containers with the same tag.
Hidden Properties	
* textFileNamesRegex :	STRING = .\.(cfg conf config ini properties props txt asp aspx htm html jsf jsp xht xhtml sql xml xsd xsl xslt) Regular expression that matches file names of text files

jbossdm.WarModule

Type Hierarchy `jbossdm.CliManagedDeployedArtifact >> jbossdm.CliManagedDeployed >> udm.BaseDeployed >> udm.BaseConfigurationItem`

Interfaces `udm.Artifact, udm.Deployed, udm.ConfigurationItem, udm.DerivedArtifact`

War with values configured for a deployment

Parent	
* container :	CI<udm.Container> The container on which this deployed runs.
Public Properties	
* contextRoot :	STRING Context root for the web application
deployable :	CI<udm.Deployable> The deployable that this deployed is derived from.
placeholders :	MAP_STRING_STRING A Map containing all the placeholders mapped to their values. Special values are <ignore> or <empty>

Hidden Properties	
* createOrder : INTEGER = 50	The order of the step in the step list for the create operation.
* createScript : STRING = jboss/dm/application/install-war.py	Create Script
* createVerb : STRING = Create	Create Verb
* destroyOrder : INTEGER = 40	The order of the step in the step list for the destroy operation.
* destroyScript : STRING = jboss/dm/application/uninstall-war.py	Destroy Script
* destroyVerb : STRING = Destroy	Destroy Verb
* inspectScript : STRING = jboss/dm/application/inspect-war.py	Inspect Script
* libraries : LIST_OF_STRING = [jboss/dm/application/application-lib.py]	Libraries
* modifyOrder : INTEGER = 50	The order of the step in the step list for the modify operation.
* modifyVerb : STRING = Modify	Modify Verb
* noopOrder : INTEGER = 50	The order of the step in the step list for the noop operation.
* noopVerb : STRING = Modify	Noop Verb
modifyScript : STRING	Classpath to the script that is uploaded and executed on the generic container for the modify operation.
noopScript : STRING	Classpath to the script that is uploaded and executed on the generic container for the noop operation.

jbosssdm.XaDataSource

Type Hierarchy [jbosssdm.JeeXaDataSource](#) >> [jbosssdm.BaseDataSource](#) >> [jbosssdm.CliManagedDeployed](#) >> [udm.BaseDeployed](#) >> [udm.BaseConfigurationItem](#)

Interfaces [udm.Deployed](#), [udm.ConfigurationItem](#)

XA DataSource. This deployed is used when a [jbosssdm.XaDataSourceSpec](#) is specified in a package.

Parent	
* container : CI < udm.Container >	The container on which this deployed runs.

Public Properties	
* driverName : STRING	Defines the JDBC driver the datasource should use. It is a symbolic name matching the the name of installed driver. In case the driver is deployed as jar, the name is the name of deployment unit.
* jndiName : STRING	Specifies the JNDI name for the datasource
* xaProperties : MAP_STRING_STRING	Properties to assign to the XADataSource implementation class. At least one XA property is required (i.e. url)
backgroundValidation : BOOLEAN = false	An element to specify that connections should be validated on a background thread versus being validated prior to use. Changing this value can be done only on disabled datasource, requires a server restart otherwise.
checkValidSql : STRING	Specify an SQL statement to check validity of a pool connection. This may be called when managed connection is obtained from the pool
deployable : CI<udm.Deployable>	The deployable that this deployed is derived from.
exceptionSorter : STRING	An org.jboss.jca.adapters.jdbc.ExceptionSorter that provides an isExceptionFatal(SQLException) method to validate if an exception should broadcast an error
interleave : BOOLEAN = false	An element to enable interleaving for XA connections
maxPoolSize : INTEGER = 0	The max-pool-size element specifies the maximum number of connections for a pool. No more connections will be created in each sub-pool
minPoolSize : INTEGER = 0	The min-pool-size element specifies the minimum number of connections for a pool
newConnectionSql : STRING	Specifies an SQL statement to execute whenever a connection is added to the connection pool
padXid : BOOLEAN = false	Should the Xid be padded
password : STRING	Specifies the password used when creating a new connection
prefillEnabled : BOOLEAN = false	Should the pool be prefilled. Changing this value can be done only on disabled datasource, requires a server restart otherwise.
sameRmOverride : BOOLEAN = false	The is-same-rm-override element allows one to unconditionally set whether the javax.transaction.xa.XAResource.isSameRM(XAResource) returns true or false
securityDomain : STRING	Specifies the security domain which defines the javax.security.auth.Subject that are used to distinguish connections in the pool
sharePreparedStatements : BOOLEAN	Whether to share prepared statements, i.e. whether asking for same statement twice without closing uses the same underlying prepared statement
staleConnectionChecker : STRING	An org.jboss.jca.adapters.jdbc.StaleConnectionChecker that provides an isStaleConnection(SQLException) method which if it returns true will wrap the exception in an org.jboss.jca.adapters.jdbc.StaleConnectionException
statementCacheSize : INTEGER = -1	The number of prepared statements per connection in an LRU cache
strictMinimum : BOOLEAN = false	Specifies if the min-pool-size should be considered strictly
transactionIsolation : STRING	Set the java.sql.Connection transaction isolation level. Valid values are: TRANSACTION_READ_UNCOMMITTED, TRANSACTION_READ_COMMITTED, TRANSACTION_REPEATABLE_READ, TRANSACTION_SERIALIZABLE and TRANSACTION_NONE
username : STRING	Specify the user name used when creating a new connection
validConnectionChecker : STRING	An org.jboss.jca.adapters.jdbc.ValidConnectionChecker that provides an

isValidConnection(Connection) method to validate a connection. If an exception is returned that means the connection is invalid. This overrides the check-valid-connection-sql element

validateOnMatch : **BOOLEAN** = false

The validate-on-match element specifies if connection validation should be done when a connection factory attempts to match a managed connection. This is typically exclusive to the use of background validation

validationMillis : **INTEGER** = -1

The background-validation-millis element specifies the amount of time, in milliseconds, that background validation will run. Changing this value can be done only on disabled datasource, requires a server restart otherwise

wrapXa : **BOOLEAN** = false

Should the XAResource instances be wrapped in a org.jboss.tm.XAResourceWrapper instance

Hidden Properties

* **createOrder** : **INTEGER** = 50

The order of the step in the step list for the create operation.

* **createScript** : **STRING** = jboss/dm/ds/create-xa-datasource.py

Create Script

* **createVerb** : **STRING** = Create

Create Verb

* **destroyOrder** : **INTEGER** = 40

The order of the step in the step list for the destroy operation.

* **destroyScript** : **STRING** = jboss/dm/ds/destroy-xa-datasource.py

Destroy Script

* **destroyVerb** : **STRING** = Destroy

Destroy Verb

* **inspectScript** : **STRING** = jboss/dm/ds/inspect-xa-datasource.py

Inspect Script

* **libraries** : **LIST_OF_STRING** = [jboss/dm/ds/datasource-lib.py]

Libraries

* **modifyOrder** : **INTEGER** = 50

The order of the step in the step list for the modify operation.

* **modifyVerb** : **STRING** = Modify

Modify Verb

* **noopOrder** : **INTEGER** = 50

The order of the step in the step list for the noop operation.

* **noopVerb** : **STRING** = Modify

Noop Verb

modifyScript : **STRING**

Classpath to the script that is uploaded and executed on the generic container for the modify operation.

noopScript : **STRING**

Classpath to the script that is uploaded and executed on the generic container for the noop operation.

jbosssdm.XaDataSourceSpec

Type Hierarchy jee.DataSourceSpec >> jee.JndiResourceSpec >> jee.ResourceSpec >> udm.BaseDeployable >> udm.BaseConfigurationItem

Interfaces udm.Tagable, udm.Deployable, udm.ConfigurationItem

XA DataSource. This deployed is used when a jbosssdm.XaDataSourceSpec is specified in a package. (deployable)

Public Properties	
backgroundValidation : STRING	An element to specify that connections should be validated on a background thread versus being validated prior to use. Changing this value can be done only on disabled datasource, requires a server restart otherwise. (boolean)
checkValidSql : STRING	Specify an SQL statement to check validity of a pool connection. This may be called when managed connection is obtained from the pool (string)
driverName : STRING	Defines the JDBC driver the datasource should use. It is a symbolic name matching the the name of installed driver. In case the driver is deployed as jar, the name is the name of deployment unit. (string)
exceptionSorter : STRING	An org.jboss.jca.adapters.jdbc.ExceptionSorter that provides an isExceptionFatal(SQLException) method to validate if an exception should broadcast an error (string)
interleave : STRING	An element to enable interleaving for XA connections (boolean)
jndiName : STRING	Specifies the JNDI name for the datasource (string)
maxPoolSize : STRING	The max-pool-size element specifies the maximum number of connections for a pool. No more connections will be created in each sub-pool (integer)
minPoolSize : STRING	The min-pool-size element specifies the minimum number of connections for a pool (integer)
newConnectionSql : STRING	Specifies an SQL statement to execute whenever a connection is added to the connection pool (string)
padXid : STRING	Should the Xid be padded (boolean)
password : STRING	Specifies the password used when creating a new connection (string)
prefillEnabled : STRING	Should the pool be prefilled. Changing this value can be done only on disabled datasource, requires a server restart otherwise. (boolean)
sameRmOverride : STRING	The is-same-rm-override element allows one to unconditionally set whether the javax.transaction.xa.XAResource.isSameRM(XAResource) returns true or false (boolean)
securityDomain : STRING	Specifies the security domain which defines the javax.security.auth.Subject that are used to distinguish connections in the pool (string)
sharePreparedStatements : STRING	Whether to share prepared statements, i.e. whether asking for same statement twice without closing uses the same underlying prepared statement (boolean)
staleConnectionChecker : STRING	An org.jboss.jca.adapters.jdbc.StaleConnectionChecker that provides an isStaleConnection(SQLException) method which if it returns true will wrap the exception in an org.jboss.jca.adapters.jdbc.StaleConnectionException (string)
statementCacheSize : STRING	The number of prepared statements per connection in an LRU cache (integer)
strictMinimum : STRING	Specifies if the min-pool-size should be considered strictly (boolean)
tags : SET_OF_STRING	If set, this deployable will only be mapped automatically to containers with the same tag.
transactionIsolation : STRING	Set the java.sql.Connection transaction isolation level. Valid values are: TRANSACTION_READ_UNCOMMITTED, TRANSACTION_READ_COMMITTED, TRANSACTION_REPEATABLE_READ, TRANSACTION_SERIALIZABLE and TRANSACTION_NONE (string)
username : STRING	Specify the user name used when creating a new connection (string)
validConnectionChecker : STRING	An org.jboss.jca.adapters.jdbc.ValidConnectionChecker that provides an isValidConnection(Connection) method to validate a connection. If an exception is returned that means the connection is invalid. This overrides the check-valid-

connection-sql element (string)
validateOnMatch : STRING The validate-on-match element specifies if connection validation should be done when a connection factory attempts to match a managed connection. This is typically exclusive to the use of background validation (boolean)
validationMillis : STRING The background-validation-millis element specifies the amount of time, in milliseconds, that background validation will run. Changing this value can be done only on disabled datasource, requires a server restart otherwise (integer)
wrapXa : STRING Should the XAResource instances be wrapped in a org.jboss.tm.XAResourceWrapper instance (boolean)
xaProperties : MAP_STRING_STRING Properties to assign to the XADatasource implementation class. At least one XA property is required (i.e. url) (map_string_string)