

# **Deployit JBoss Application Server Plugin Manual**

**Version 3.7.0**

# Table of Content

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	5
Extension points	5
Extending the JBoss Plugin	5
Discovery	6
CI Reference	7
Configuration Item Overview	7
Deployable Configuration Items	7
Deployed Configuration Items	7
Topology Configuration Items	7
Virtual Deployed Configuration Items	7
Configuration Item Details	8
jbossas.Artifact	8
jbossas.BaseServer	9
jbossas.Configuration	11
jbossas.ConfigurationFile	13
jbossas.ConfigurationFolder	14
jbossas.Datasource	15
jbossas.DeployedConfigurationFile	17
jbossas.DeployedConfigurationFolder	18
jbossas.DeployedLibrary	20
jbossas.Ear	22
jbossas.EarModule	23
jbossas.Library	25
jbossas.NonTransactionalDatasource	25
jbossas.NonTransactionalDatasourceSpec	28
jbossas.Queue	28
jbossas.QueueSpec	31
jbossas.Resource	31
jbossas.ServerV4	33
jbossas.ServerV5	34
jbossas.ServerV6	36
jbossas.Topic	38
jbossas.TopicSpec	41
jbossas.TransactionalDatasource	41
jbossas.TransactionalDatasourceSpec	44
jbossas.TransactionalXADatasource	44
jbossas.TransactionalXADatasourceSpec	47
jbossas.War	47
jbossas.WarModule	48

# Preface

This document describes the functionality provided by the JBoss application server (AS) plugin.

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

## Overview

The JBoss plugin is a Deployit plugin that adds the capability to manage deployments and resources on JBoss application server. It works out of the box for deploying/ undeploying application artifacts, datasource and other JMS resources (see the *Features* section below) , and can easily be extended to support more deployment options or management of new artifacts/resources on JBoss AS.

## Features

- Application artifacts
  - Enterprise application (EAR)
  - Web application (WAR)
- JBoss-specific artifacts:
  - Service Archive (SAR)
  - Resource Archive (RAR)
  - Hibernate Archive (HAR)
  - Aspect archive (AOP)
- Resources
  - Datasource
  - JMS Queue\*\*
  - JMS Topic\*\*
- Discovery

\*\* While creating JMS resources like Queues and Topics for JBoss AS 6, only the jndi name is used for the creation purpose. Other properties like RedeliveryDelay, MaxDeliveryAttempts and so on are not used even when they have been defined on the CI (in synthetic.xml) and set. These properties are defined by editing the global server configuration, which can be found at

"%JBOSS\_HOME%/server/<configuration/deploy/hornetq/hornetq-jms.xml.

## Requirements

- **Deployit requirements**
  - **Deployit:** version 3.7+
  - **JBoss AS versions:** 4.x, 5.x and 6.x
  - **Other Deployit Plugins:** None
- **Infrastructural requirements**
  - **User credentials** for accessing the Host running the JBoss application Server.

## 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 contain 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: jbossas.TransactionDataSourceSpec	
CI-jndiName: jdbc/sampleDataSource	
CI-connectionUrl: jdbc:mysql://localhost/test	
CI-driverClass: com.mysql.jdbc.Driver	
CI-username: {{DATABASE_USERNAME}}	
CI-password: {{DATABASE_PASSWORD}}	
Name: testQueue	
CI-Type: jbossas.QueueSpec	
CI-jndiName: jms/testQueue	
Name: testTopic	
CI-Type: jbossas.TopicSpec	
CI-jndiName: jms/testTopic	
Name: testConfigFile.txt	
CI-Name: testConfigFiles	
CI-Type: jbossas.ConfigurationFile	
Name: testConfigFolder	
CI-Name: testConfigFolder	
CI-Type: jbossas.ConfigurationFolder	

## Using the deployables and deployments

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

### Deployable vs. Container table

Deployable	Container	Generated deployed
Application artifact: jee.Ear jee.War	Server	jbossas.EarModule jbossas.WarModule
jbossas.TransactionDataSourceSpec jbossas.TransactionXADatasourceSpec jbossas.NonTransactionalDataSourceSpec	Server	jbossas.TransactionDataSource jbossas.TransactionXADatasource jbossas.NonTransactionalDataSource
jbossas.QueueSpec	Server	jbossas.Queue
jbossas.TopicSpec	Server	jbossas.Topic
jbossas.ConfigurationFile	Server	jbossas.DeployedConfigurationFile
jbossas.ConfigurationFolder	Server	jbossas.DeployedConfigurationFolder

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

### Deployed Actions Table

Deployed	Actions performed for operations		
	Create	Destroy	Modify
jbossas.EarModule jbossas.WarModule	<ul style="list-style-type: none"> <li>stop Server</li> <li>deploy application</li> <li>start Server</li> </ul>	<ul style="list-style-type: none"> <li>stop Server</li> <li>undeploy old application version</li> <li>start Server</li> </ul>	<ul style="list-style-type: none"> <li>stop Server</li> <li>undeploy old application version</li> <li>deploy new application version</li> </ul>

			<ul style="list-style-type: none"> <li>• start Server</li> </ul>
jbossas.Transactionaldatasource jbossas.Transactionaldatasource jbossas.NonTransactionalDataSource	<ul style="list-style-type: none"> <li>• create datasource</li> </ul>	<ul style="list-style-type: none"> <li>• destroy datasource</li> </ul>	<ul style="list-style-type: none"> <li>• destroy datasource</li> <li>• create modified datasource</li> </ul>
jbossas.Queue	<ul style="list-style-type: none"> <li>• stop Server</li> <li>• create Queue</li> <li>• start Server</li> </ul>	<ul style="list-style-type: none"> <li>• stop Server</li> <li>• destroy Queue</li> <li>• start Server</li> </ul>	<ul style="list-style-type: none"> <li>• stop Server</li> <li>• destroy Queue</li> <li>• create modified Queue</li> <li>• start Server</li> </ul>
jbossas.Topic	<ul style="list-style-type: none"> <li>• stop Server</li> <li>• create Topic</li> <li>• start Server</li> </ul>	<ul style="list-style-type: none"> <li>• stop Server</li> <li>• destroy Topic</li> <li>• start Server</li> </ul>	<ul style="list-style-type: none"> <li>• stop Server</li> <li>• destroy Topic</li> <li>• create modified Topic</li> <li>• start Server</li> </ul>
jbossas.DeployedConfigurationFile	<ul style="list-style-type: none"> <li>• stop Server</li> <li>• deploying config files</li> <li>• start Server</li> </ul>	<ul style="list-style-type: none"> <li>• stop Server</li> <li>• undeploy config files</li> <li>• start Server</li> </ul>	<ul style="list-style-type: none"> <li>• stop Server</li> <li>• undeploy config files</li> <li>• deploy new config files</li> <li>• start Server</li> </ul>
jbossas.DeployedConfigurationFolder	<ul style="list-style-type: none"> <li>• stop Server</li> <li>• deploying configuration folder</li> <li>• start Server</li> </ul>	<ul style="list-style-type: none"> <li>• stop Server</li> <li>• undeploy configuration folder</li> <li>• start Server</li> </ul>	<ul style="list-style-type: none"> <li>• stop Server</li> <li>• undeploy configuration folder</li> <li>• deploy new configuration folder</li> <li>• start Server</li> </ul>

## Deploying applications

By default, Deployit deploys the application artifacts and resource specifications (datasource, queues, topics etc) to the 'deploy' directory in the server's configuration. So if the server configuration is set to 'default' (which is the default value of server name), the artifact is copied to \${JBoss\_HOME}/server/default/deploy. Also, the server is stopped before copying the artifact and then started again. But all these configurations can be customized to suit more specific scenarios.

## Extension points

The JBoss plugin is designed to be extended through Deployit's Plugin API type system. Also, since the JBoss plugin is built on top of generic-plugin, support for new types can be added using the generic plugin patterns. Refer to *Generic Plugin Manual* for more details. Also, refer to *Customization Manual* for an explanation of the type system.

The next section describes extensibility by examples:

## Extending the JBoss Plugin

### Making existing property hidden/visible or changing the default value

The following synthetic.xml snippet shows how the *restartRequired* property can be made visible and the *targetDirectory* property can be given a default value of '/home/deployer/install-files' for *jbossas.EarModule*.

```
<type-modification type="jbossas.EarModule">
  <!-- make it visible so that I can control whether to restart a Server or not from UI-->
  <property name="restartRequired" kind="boolean" default="true" hidden="false"/>

  <!-- custom deploy directory for my jboss applications -->
  <property name="targetDirectory" default="/home/deployer/install-files" hidden="true"/>
</type-modification>
```

### Adding a new property to a deployed/deployable

The following `synthetic.xml` snippet shows how a new property 'blocking-timeout-millis' can be added to `jbossas.TransactionDataSource`

```
<type-modification type="jbossas.TransactionDataSource">
  <!-- adding new property -->
  <property name="blockingTimeoutMillis" kind="integer" default="3000" description="maximum time in milliseconds to block
    while waiting for a connection before throwing an exception"/>
</type-modification>
```

Note that while adding a new property in JBoss plugin, *the configuration property must be specified in lower camel-case with the hyphens removed from it*. Thus the property 'blocking-timeout-millis' has to be specified as `blockingTimeoutMillis`. Similarly 'idle-timeout-minutes' becomes `idleTimeoutMinutes` in `synthetic.xml`.

### Adding a new type

New types can be added in JBoss plugin using the `Generic Plugin` patterns. For example, the following `synthetic.xml` snippet defines a new type `jbossas.EjbJarModule`

```
<type type="jbossas.EjbJarModule" extends="generic.CopiedArtifact" deployable-type="jee.EjbJar"
      container-type="jbossas.BaseServer">
  <generate-deployable type="jbossas.EjbJar" extends="jee.EjbJar"/>
  <property name="targetDirectory" default="${deployed.container.home}/server/${deployed.container.serverName}/deploy"
    hidden="true"/>
  <property name="targetFile" default="${deployed.deployable.name}.jar" hidden="true"/>
  <property name="createOrder" kind="integer" default="50" hidden="true"/>
  <property name="destroyOrder" kind="integer" default="40" hidden="true"/>
  <property name="restartRequired" kind="boolean" default="true" hidden="true"/>
</type>
```

## Discovery

Once the JBoss server's home location and the host on which JBoss server is running are specified, the following properties can be discovered on a running JBoss server.

- JBoss version
- Control port
- Http port
- Ajp port

Here is an example CLI script which discovers a sample JBoss server:

```
host = repository.create(factory.configurationItem('Infrastructure/jboss-51-host', 'overthere.SshHost',
  {'connectionType':'SFTP','address': 'jboss-51','username': 'root','password':'centos','os':'UNIX'}))
jboss = factory.configurationItem('Infrastructure/jboss-51-host/jboss-51', 'jbossas.ServerV5',
  {'home':'/opt/jboss/5.1.0.GA', 'host':'Infrastructure/jboss-51-host'})
cis = deployit.discover(jboss)
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:

- Hosts are created under the Infrastructure tree, so the host id has been kept as 'Infrastructure/jboss-51-host'
- Host address can be the host IP address or the dns name defined for the host
- JBoss server has a containment relation with a Host (created under a Host), so the server id has been kept as 'Infrastructure/jboss-51-host/jboss-51'

# CI Reference

## Configuration Item Overview

### Deployable Configuration Items

CI	Description
<a href="#">jbossas.ConfigurationFile</a>	File containing application configurations that is deployed to the server conf directory
<a href="#">jbossas.ConfigurationFolder</a>	Folder containing files for application configuration that are deployed to the server conf directory
<a href="#">jbossas.Ear</a>	A JEE EAR archive
<a href="#">jbossas.Library</a>	The Library as deployed on the jboss server
<a href="#">jbossas.NonTransactionalDatasourceSpec</a>	A JBoss No Transaction Datasource (deployable)
<a href="#">jbossas.QueueSpec</a>	Specification for a JBoss queue
<a href="#">jbossas.TopicSpec</a>	Specification for a JBoss topic
<a href="#">jbossas.TransactionalDatasourceSpec</a>	A JBoss Local Transaction Datasource (deployable)
<a href="#">jbossas.TransactionalXADatasourceSpec</a>	A JBoss XA Datasource (deployable)
<a href="#">jbossas.War</a>	A JEE WAR archive

### Deployed Configuration Items

CI	Description
<a href="#">jbossas.DeployedConfigurationFile</a>	Deployed configuration file
<a href="#">jbossas.DeployedConfigurationFolder</a>	Deployed configuration folder
<a href="#">jbossas.DeployedLibrary</a>	The Library as deployed on the jboss server
<a href="#">jbossas.EarModule</a>	Ear with values configured for a deployment
<a href="#">jbossas.NonTransactionalDatasource</a>	A JBoss No Transaction Datasource
<a href="#">jbossas.Queue</a>	A JBoss Queue
<a href="#">jbossas.Topic</a>	A JBoss topic
<a href="#">jbossas.TransactionalDatasource</a>	A JBoss Local Transaction Datasource
<a href="#">jbossas.TransactionalXADatasource</a>	A JBoss XA Datasource
<a href="#">jbossas.WarModule</a>	War with values configured for a deployment

### Topology Configuration Items

CI	Description
<a href="#">jbossas.BaseServer</a>	Base JBoss Application Server
<a href="#">jbossas.ServerV4</a>	Jboss 4 application server
<a href="#">jbossas.ServerV5</a>	Jboss 5 application server
<a href="#">jbossas.ServerV6</a>	Jboss 6 application server

### Virtual Deployed Configuration Items

CI	Description
<a href="#">jbossas.Artifact</a>	Base type for a JBoss artifact
<a href="#">jbossas.Configuration</a>	Base type for a JBoss configuration
<a href="#">jbossas.Datasource</a>	Base type for a JBoss datasource
<a href="#">jbossas.Resource</a>	Description unavailable


## Configuration Item Details

### jbossas.Artifact

**Hierarchy** generic.CopiedArtifact >> generic.AbstractDeployedArtifact >> generic.AbstractDeployed >> udm.BaseDeployed >> udm.BaseConfigurationItem

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

Base type for a JBoss artifact.

Public Properties	
 <b>container</b> : CI<udm.Container>	
The container on which this deployed runs.	
<b>deployable</b> : CI<udm.Deployable>	
The deployable that this deployed is derived from.	
<b>placeholders</b> : MAP_STRING_STRING	
A Map containing all the placeholders mapped to their values. Special values are <ignore> or <empty>	
<b>targetFile</b> : STRING	
Name of the artifact on the generic server.	



## Hidden Properties

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

Create Order

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

Create Verb

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

Destroy Order

**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

**targetDirectory** : **STRING** = *\${deployed.container.home}/server/\${deployed.container.serverName}/deploy*

Target Directory

**createTargetDirectory** : **BOOLEAN** = *false*

Create the target directory on the generic server if it does not exist.

**inspectClasspathResources** : **SET\_OF\_STRING**

Additional classpath resources that should be uploaded to the working directory before executing the inspect script.

**inspectScript** : **STRING**

Classpath to the script used to inspect the generic container.

**inspectTemplateClasspathResources** : **SET\_OF\_STRING**

Additional template classpath resources that should be uploaded to the working directory before executing the inspect script. The template is first rendered and the rendered content copied to a file, with the same name as the template, in the working directory.

**restartRequired** : **BOOLEAN** = *true*

Restart Required

**restartRequiredForNoop** : **BOOLEAN** = *false*

The generic container requires a restart for the NOOP action performed by this deployed.

**targetDirectoryShared** : **BOOLEAN** = *true*

Is the target directory shared by others on the generic server. When true, the target directory is not deleted during a destroy operation; only the artifacts copied to it.

## jbosscas.BaseServer

**Hierarchy** generic.Container >> udm.BaseContainer >> udm.BaseConfigurationItem

**Interfaces** udm.Taggable, udm.ConfigurationItem, generic.GenericContainer, udm.Container, overthere.HostContainer

Base JBoss Application Server

### Public Properties

**ajpPort** : **INTEGER** = *8009*

AJP connector port

**controlPort** : **INTEGER** = *1099*

Connector control port



**home** : **STRING**

JBoss application server installation directory. e.g. /opt/jboss/5.1.0.GA



**host** : **CI**<overthere.Host>

Host upon which the container resides

**httpPort** : **INTEGER** = *8080*

HTTP/1.1 connector port

**serverName** : **STRING** = *default*

Server configuration name e.g. default or minimal

**envVars** : **MAP\_STRING\_STRING**

Environment variables for container

**tags** : **SET\_OF\_STRING**

The tags to map deployables to containers.

## Hidden Properties

**inspectTemplate** : *STRING* = *jboss/container/inspect.sh.ftl*

Script (a freemarker template) used to inspect the server

**jmsResourceFileSuffix** : *STRING* = *service*

Jms Resource File Suffix

**queryTemplate** : *STRING* = *jboss/container/query.sh.ftl*

Script (a freemarker template) used to query a mbean valuf from the server

**restartOrder** : *INTEGER* = *90*

The order of the restart container step in the step list.

**startOrder** : *INTEGER* = *90*

The order of the start container step in the step list.

**startScript** : *STRING* = *jboss/container/start.sh*

Script used to start the server

**startWaitTime** : *INTEGER* = *30*

Duration (in secs) to wait after the start server step has been executed

**stopOrder** : *INTEGER* = *10*

The order of the stop container step in the step list.

**stopScript** : *STRING* = *jboss/container/stop.sh*

Script used to stop the server

**stopWaitTime** : *INTEGER* = *10*

Duration (in secs) to wait after the stop server step has been executed

**inspectClasspathResources** : *SET\_OF\_STRING*

Additional classpath resources that should be uploaded to the working directory before executing the inspect script.

**inspectScript** : *STRING*

Classpath to the script used to inspect the generic container.

**inspectTemplateClasspathResources** : *SET\_OF\_STRING*

Additional template classpath resources that should be uploaded to the working directory before executing the inspect script. The template is first rendered and the rendered content copied to a file, with the same name as the template, in the working directory.

**restartScript** : *STRING*

Classpath to the script used to restart the generic container.

**restartWaitTime** : *INTEGER* = *0*

The time to wait in seconds for a container restart action.

## jbossas.Configuration

**Hierarchy** *jbossas.Artifact* >> *generic.CopiedArtifact* >> *generic.AbstractDeployedArtifact* >> *generic.AbstractDeployed* >> *udm.BaseDeployed* >> *udm.BaseConfigurationItem*

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

Base type for a JBoss configuration.

### Public Properties



**container** : `CI<udm.Container>`

The container on which this deployed runs.

**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>`

**targetFile** : `STRING`

Name of the artifact on the generic server.

## Hidden Properties

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

Create Order

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

Create Verb

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

Destroy Order

**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

**targetDirectory** : **STRING** = *\${deployed.container.home}/server/\${deployed.container.serverName}/conf*

Target Directory

**createTargetDirectory** : **BOOLEAN** = *false*

Create the target directory on the generic server if it does not exist.

**inspectClasspathResources** : **SET\_OF\_STRING**

Additional classpath resources that should be uploaded to the working directory before executing the inspect script.

**inspectScript** : **STRING**

Classpath to the script used to inspect the generic container.

**inspectTemplateClasspathResources** : **SET\_OF\_STRING**

Additional template classpath resources that should be uploaded to the working directory before executing the inspect script. The template is first rendered and the rendered content copied to a file, with the same name as the template, in the working directory.

**restartRequired** : **BOOLEAN** = *true*

Restart Required

**restartRequiredForNoop** : **BOOLEAN** = *false*

The generic container requires a restart for the NOOP action performed by this deployed.

**targetDirectoryShared** : **BOOLEAN** = *true*

Is the target directory shared by others on the generic server. When true, the target directory is not deleted during a destroy operation; only the artifacts copied to it.

## jbosscas.ConfigurationFile

**Hierarchy** generic.File >> udm.BaseDeployableFileArtifact >> udm.BaseDeployableArtifact >> udm.BaseDeployable >> udm.BaseConfigurationItem

**Interfaces** udm.Taggable, udm.Deployable, udm.SourceArtifact, udm.Artifact, udm.DeployableArtifact, udm.ConfigurationItem, udm.FileArtifact

File containing application configurations that is deployed to the server conf directory

#### 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** = *true*

Scan Placeholders

**tags** : **SET\_OF\_STRING**

The tags to map deployables to containers.

**targetFile** : **STRING**

Name of the artifact on the generic server. (string)

#### 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

## jbossas.ConfigurationFolder

**Hierarchy** generic.Folder >> udm.BaseDeployableFolderArtifact >> udm.BaseDeployableArtifact >> udm.BaseDeployable >> udm.BaseConfigurationItem

**Interfaces** udm.Taggable, udm.Deployable, udm.SourceArtifact, udm.Artifact, udm.DeployableArtifact, udm.ConfigurationItem, udm.FolderArtifact

Folder containing files for application configuration that are deployed to the server conf directory

#### 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** = *true*

Scan Placeholders

**tags** : **SET\_OF\_STRING**

The tags to map deployables to containers.

**targetFile** : **STRING**

Name of the artifact on the generic server. (string)

### 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

## jbossas.Datasource

**Hierarchy** `jbossas.Resource` >> `generic.ProcessedTemplate` >> `generic.AbstractDeployedArtifact` >> `generic.AbstractDeployed` >> `udm.BaseDeployed` >> `udm.BaseConfigurationItem`


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

Base type for a JBoss datasource

### Public Properties

**connectionUrl** : `STRING`

Connection Url

 **container** : `CI<udm.Container>`

The container on which this deployed runs.

**driverClass** : `STRING`

Driver Class

**jndiName** : `STRING`

Jndi Name

**maxPoolSize** : `INTEGER = 20`

Max Pool Size

**minPoolSize** : `INTEGER = 0`

Min Pool Size

**password** : `STRING`

Password

**userName** : `STRING`

User Name

**deployable** : `CI<udm.Deployable>`

The deployable that this deployed is derived from.

**useJavaContext** : `BOOLEAN = true`

Use Java Context

## Hidden Properties

**createOrder** : INTEGER = 50

Create Order

**createVerb** : STRING = *Create*

Create Verb

**destroyOrder** : INTEGER = 40

Destroy Order

**destroyVerb** : STRING = *Destroy*

Destroy Verb

**inspectTemplate** : STRING = *jboss/datasource/inspect.sh.ftl*

Inspect Template

**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

**targetDirectory** : STRING = *\${deployed.container.home}/server/\${deployed.container.serverName}/deploy*

Target Directory

**targetFile** : STRING = *\${deployed.deployable.name}-ds.xml*

Target File

**template** : STRING = *jboss/datasource/template.xml.ftl*

Template

**createTargetDirectory** : BOOLEAN = *false*

Create the target directory on the generic server if it does not exist.

**inspectClasspathResources** : SET\_OF\_STRING

Additional classpath resources that should be uploaded to the working directory before executing the inspect script.

**inspectScript** : STRING

Classpath to the script used to inspect the generic container.

**inspectTemplateClasspathResources** : SET\_OF\_STRING

Additional template classpath resources that should be uploaded to the working directory before executing the inspect script. The template is first rendered and the rendered content copied to a file, with the same name as the template, in the working directory.

**restartRequired** : BOOLEAN = *true*

Restart Required

**restartRequiredForNoop** : BOOLEAN = *false*



The generic container requires a restart for the NOOP action performed by this deployed.

**targetDirectoryShared** : **BOOLEAN** = *true*

Is the target directory shared by others on the generic server. When true, the target directory is not deleted during a destroy operation; only the artifacts copied to it.

## jbossas.DeployedConfigurationFile

**Hierarchy** [jbossas.Configuration](#) >> [jbossas.Artifact](#) >> generic.CopiedArtifact >> generic.AbstractDeployedArtifact >> generic.AbstractDeployed >> udm.BaseDeployed >> udm.BaseConfigurationItem

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

Deployed configuration file

### Public Properties

 **container** : **CI**<udm.Container>

The container on which this deployed runs.

**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>

**targetFile** : **STRING**

Name of the artifact on the generic server.

## Hidden Properties

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

Create Order

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

Create Verb

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

Destroy Order

**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

**targetDirectory** : **STRING** = *\${deployed.container.home}/server/\${deployed.container.serverName}/conf*

Target Directory

**createTargetDirectory** : **BOOLEAN** = *false*

Create the target directory on the generic server if it does not exist.

**inspectClasspathResources** : **SET\_OF\_STRING**

Additional classpath resources that should be uploaded to the working directory before executing the inspect script.

**inspectScript** : **STRING**

Classpath to the script used to inspect the generic container.

**inspectTemplateClasspathResources** : **SET\_OF\_STRING**

Additional template classpath resources that should be uploaded to the working directory before executing the inspect script. The template is first rendered and the rendered content copied to a file, with the same name as the template, in the working directory.

**restartRequired** : **BOOLEAN** = *true*

Restart Required

**restartRequiredForNoop** : **BOOLEAN** = *false*

The generic container requires a restart for the NOOP action performed by this deployed.

**targetDirectoryShared** : **BOOLEAN** = *true*

Is the target directory shared by others on the generic server. When true, the target directory is not deleted during a destroy operation; only the artifacts copied to it.

## jbossas.DeployedConfigurationFolder

**Hierarchy** [jbossas.Configuration](#) >> [jbossas.Artifact](#) >> generic.CopiedArtifact >> generic.AbstractDeployedArtifact >> generic.AbstractDeployed >> udm.BaseDeployed >> udm.BaseConfigurationItem

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

Deployed configuration folder

### Public Properties



**container** : [CI<udm.Container>](#)

The container on which this deployed runs.

**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>

**targetFile** : [STRING](#)

Name of the artifact on the generic server.

## Hidden Properties

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

Create Order

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

Create Verb

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

Destroy Order

**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

**targetDirectory** : **STRING** = *\${deployed.container.home}/server/\${deployed.container.serverName}/conf*

Target Directory

**createTargetDirectory** : **BOOLEAN** = *false*

Create the target directory on the generic server if it does not exist.

**inspectClasspathResources** : **SET\_OF\_STRING**

Additional classpath resources that should be uploaded to the working directory before executing the inspect script.

**inspectScript** : **STRING**

Classpath to the script used to inspect the generic container.

**inspectTemplateClasspathResources** : **SET\_OF\_STRING**

Additional template classpath resources that should be uploaded to the working directory before executing the inspect script. The template is first rendered and the rendered content copied to a file, with the same name as the template, in the working directory.

**restartRequired** : **BOOLEAN** = *true*

Restart Required

**restartRequiredForNoop** : **BOOLEAN** = *false*

The generic container requires a restart for the NOOP action performed by this deployed.

**targetDirectoryShared** : **BOOLEAN** = *true*

Is the target directory shared by others on the generic server. When true, the target directory is not deleted during a destroy operation; only the artifacts copied to it.

## jbossas.DeployedLibrary

**Hierarchy** [jbossas.Artifact](#) >> generic.CopiedArtifact >> generic.AbstractDeployedArtifact >> generic.AbstractDeployed >> udm.BaseDeployed >> udm.BaseConfigurationItem

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

The Library as deployed on the jboss server.

### Public Properties



**container** : [CI<udm.Container>](#)

The container on which this deployed runs.

**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>

**targetFile** : [STRING](#)

Name of the artifact on the generic server.

## Hidden Properties

**createOrder** : *INTEGER = 50*

Create Order

**createVerb** : *STRING = Create*

Create Verb

**destroyOrder** : *INTEGER = 40*

Destroy Order

**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

**targetDirectory** : *STRING = \${deployed.container.home}/server/\${deployed.container.serverName}/lib*

Target Directory

**createTargetDirectory** : *BOOLEAN = false*

Create the target directory on the generic server if it does not exist.

**inspectClasspathResources** : *SET\_OF\_STRING*

Additional classpath resources that should be uploaded to the working directory before executing the inspect script.

**inspectScript** : *STRING*

Classpath to the script used to inspect the generic container.

**inspectTemplateClasspathResources** : *SET\_OF\_STRING*

Additional template classpath resources that should be uploaded to the working directory before executing the inspect script. The template is first rendered and the rendered content copied to a file, with the same name as the template, in the working directory.

**restartRequired** : *BOOLEAN = true*

Restart Required

**restartRequiredForNoop** : *BOOLEAN = false*

The generic container requires a restart for the NOOP action performed by this deployed.

**targetDirectoryShared** : *BOOLEAN = true*

Is the target directory shared by others on the generic server. When true, the target directory is not deleted during a destroy operation; only the artifacts copied to it.

## jbossas.Ear

**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 = true`

Scan Placeholders

**tags** : `SET_OF_STRING`

The tags to map deployables to containers.

### 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

## jbossas.EarModule

**Hierarchy** `jbossas.Artifact >> generic.CopiedArtifact >> generic.AbstractDeployedArtifact >> generic.AbstractDeployed >> udm.BaseDeployed >> udm.BaseConfigurationItem`

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

Ear with values configured for a deployment

### Public Properties



**container** : `CI<udm.Container>`

The container on which this deployed runs.

**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

Create Order

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

Create Verb

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

Destroy Order

**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

**targetDirectory** : **STRING** = *\${deployed.container.home}/server/\${deployed.container.serverName}/deploy*

Target Directory

**targetFile** : **STRING** = *\${deployed.deployable.name}.ear*

Target File

**createTargetDirectory** : **BOOLEAN** = *false*

Create the target directory on the generic server if it does not exist.

**inspectClasspathResources** : **SET\_OF\_STRING**

Additional classpath resources that should be uploaded to the working directory before executing the inspect script.

**inspectScript** : **STRING**

Classpath to the script used to inspect the generic container.

**inspectTemplateClasspathResources** : **SET\_OF\_STRING**

Additional template classpath resources that should be uploaded to the working directory before executing the inspect script. The template is first rendered and the rendered content copied to a file, with the same name as the template, in the working directory.

**restartRequired** : **BOOLEAN** = *true*

Restart Required

**restartRequiredForNoop** : **BOOLEAN** = *false*

The generic container requires a restart for the NOOP action performed by this deployed.

**targetDirectoryShared** : **BOOLEAN** = *true*

Is the target directory shared by others on the generic server. When true, the target directory is not deleted during a destroy operation; only the artifacts copied to it.



## jbossas.Library

**Hierarchy** generic.File >> udm.BaseDeployableFileArtifact >> udm.BaseDeployableArtifact >> udm.BaseDeployable >> udm.BaseConfigurationItem

**Interfaces** udm.Taggable, udm.Deployable, udm.SourceArtifact, udm.Artifact, udm.DeployableArtifact, udm.ConfigurationItem, udm.FileArtifact

The Library as deployed on the jboss server. (deployable)

### 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** = *true*

Scan Placeholders

**tags** : **SET\_OF\_STRING**

The tags to map deployables to containers.

**targetFile** : **STRING**

Name of the artifact on the generic server. (string)

### 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

## jbossas.NonTransactionalDatasource

**Hierarchy** jbossas.Datasource >> jbossas.Resource >> generic.ProcessedTemplate >> generic.AbstractDeployedArtifact >> generic.AbstractDeployed >> udm.BaseDeployed >> udm.BaseConfigurationItem

**Interfaces** udm.Deployed, udm.ConfigurationItem

A JBoss No Transaction Datasource

## Public Properties

**connectionUrl** : *STRING*

Connection Url



**container** : *CI<udm.Container>*

The container on which this deployed runs.

**driverClass** : *STRING*

Driver Class

**jndiName** : *STRING*

Jndi Name

**maxPoolSize** : *INTEGER = 20*

Max Pool Size

**minPoolSize** : *INTEGER = 0*

Min Pool Size

**password** : *STRING*

Password

**userName** : *STRING*

User Name

**deployable** : *CI<udm.Deployable>*

The deployable that this deployed is derived from.

**useJavaContext** : *BOOLEAN = true*

Use Java Context

## Hidden Properties

**createOrder** : INTEGER = 50

Create Order

**createVerb** : STRING = *Create*

Create Verb

**destroyOrder** : INTEGER = 40

Destroy Order

**destroyVerb** : STRING = *Destroy*

Destroy Verb

**dsType** : STRING = *no-tx-datasource*

Ds Type

**inspectTemplate** : STRING = *jboss/datasource/inspect.sh.ftl*

Inspect Template

**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

**targetDirectory** : STRING = *\${deployed.container.home}/server/\${deployed.container.serverName}/deploy*

Target Directory

**targetFile** : STRING = *\${deployed.deployable.name}-ds.xml*

Target File

**template** : STRING = *jboss/datasource/template.xml.ftl*

Template

**createTargetDirectory** : BOOLEAN = *false*

Create the target directory on the generic server if it does not exist.

**inspectClasspathResources** : SET\_OF\_STRING

Additional classpath resources that should be uploaded to the working directory before executing the inspect script.

**inspectScript** : STRING

Classpath to the script used to inspect the generic container.

**inspectTemplateClasspathResources** : SET\_OF\_STRING

Additional template classpath resources that should be uploaded to the working directory before executing the inspect script. The template is first rendered and the rendered content copied to a file, with the same name as the template, in the working directory.

**restartRequired** : BOOLEAN = *true*

**Restart Required****restartRequiredForNoop** : **BOOLEAN** = *false*

The generic container requires a restart for the NOOP action performed by this deployed.

**targetDirectoryShared** : **BOOLEAN** = *true*

Is the target directory shared by others on the generic server. When true, the target directory is not deleted during a destroy operation; only the artifacts copied to it.

## jbossas.NonTransactionalDatasourceSpec

**Hierarchy** generic.Resource >> udm.BaseDeployable >> udm.BaseConfigurationItem**Interfaces** udm.Taggable, udm.Deployable, udm.ConfigurationItem

A JBoss No Transaction Datasource (deployable)

**Public Properties****connectionUrl** : **STRING**

Connection Url (string)

**driverClass** : **STRING**

Driver Class (string)

**jndiName** : **STRING**

Jndi Name (string)

**maxPoolSize** : **STRING**

Max Pool Size (integer)

**minPoolSize** : **STRING**

Min Pool Size (integer)

**password** : **STRING**

Password (string)

**tags** : **SET\_OF\_STRING**

The tags to map deployables to containers.

**useJavaContext** : **STRING**

Use Java Context (boolean)

**userName** : **STRING**

User Name (string)

## jbossas.Queue

**Hierarchy** **jbossas.Resource** >> generic.ProcessedTemplate >> generic.AbstractDeployedArtifact >> generic.AbstractDeployed >> udm.BaseDeployed >> udm.BaseConfigurationItem**Interfaces** udm.Deployed, udm.ConfigurationItem

A JBoss Queue

## Public Properties



**container** : `CI<udm.Container>`

The container on which this deployed runs.

**jndiName** : `STRING`

Jndi Name

**maxDepth** : `INTEGER = 0`

Max Depth

**deployable** : `CI<udm.Deployable>`

The deployable that this deployed is derived from.

## Hidden Properties

**createOrder** : INTEGER = 50

Create Order

**createVerb** : STRING = *Create*

Create Verb

**destroyOrder** : INTEGER = 40

Destroy Order

**destroyVerb** : STRING = *Destroy*

Destroy Verb

**inspectTemplate** : STRING = *jboss/queue/inspect.sh.ftl*

Inspect Template

**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

**targetDirectory** : STRING = *\${deployed.container.home}/server/\${deployed.container.serverName}/deploy*

Target Directory

**targetFile** : STRING = *\${deployed.deployable.name}-\${deployed.container.jmsResourceFileSuffix}.xml*

Target File

**template** : STRING = *jboss/queue/template\${deployed.container.version}.xml.ftl*

Template

**createTargetDirectory** : BOOLEAN = *false*

Create the target directory on the generic server if it does not exist.

**inspectClasspathResources** : SET\_OF\_STRING

Additional classpath resources that should be uploaded to the working directory before executing the inspect script.

**inspectScript** : STRING

Classpath to the script used to inspect the generic container.

**inspectTemplateClasspathResources** : SET\_OF\_STRING

Additional template classpath resources that should be uploaded to the working directory before executing the inspect script. The template is first rendered and the rendered content copied to a file, with the same name as the template, in the working directory.

**restartRequired** : BOOLEAN = *true*

Restart Required

**restartRequiredForNoop** : BOOLEAN = *false*

The generic container requires a restart for the NOOP action performed by this deployed.

**targetDirectoryShared** : **BOOLEAN** = *true*

Is the target directory shared by others on the generic server. When true, the target directory is not deleted during a destroy operation; only the artifacts copied to it.

## jbossas.QueueSpec

**Hierarchy** generic.Resource >> udm.BaseDeployable >> udm.BaseConfigurationItem

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

Specification for a JBoss queue

### Public Properties

**jndiName** : **STRING**

Jndi Name (string)

**maxDepth** : **STRING**

Max Depth (integer)

**tags** : **SET\_OF\_STRING**

The tags to map deployables to containers.


## jbossas.Resource

**Hierarchy** generic.ProcessedTemplate >> generic.AbstractDeployedArtifact >> generic.AbstractDeployed >> udm.BaseDeployed >> udm.BaseConfigurationItem

**Interfaces** udm.Deployed, udm.ConfigurationItem

Description unavailable

### Public Properties

 **container** : **CI**<udm.Container>

The container on which this deployed runs.

**deployable** : **CI**<udm.Deployable>

The deployable that this deployed is derived from.

**targetFile** : **STRING**

Name of the artifact on the generic server.

## Hidden Properties

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

Create Order

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

Create Verb

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

Destroy Order

**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

**targetDirectory** : **STRING** = *\${deployed.container.home}/server/\${deployed.container.serverName}/deploy*

Target Directory

**template** : **STRING**

Classpath to the freemarker template used to generate the content of the final text base artifact.

**createTargetDirectory** : **BOOLEAN** = *false*

Create the target directory on the generic server if it does not exist.

**inspectClasspathResources** : **SET\_OF\_STRING**

Additional classpath resources that should be uploaded to the working directory before executing the inspect script.

**inspectScript** : **STRING**

Classpath to the script used to inspect the generic container.

**inspectTemplateClasspathResources** : **SET\_OF\_STRING**

Additional template classpath resources that should be uploaded to the working directory before executing the inspect script. The template is first rendered and the rendered content copied to a file, with the same name as the template, in the working directory.

**restartRequired** : **BOOLEAN** = *true*

Restart Required

**restartRequiredForNoop** : **BOOLEAN** = *false*

The generic container requires a restart for the NOOP action performed by this deployed.

**targetDirectoryShared** : **BOOLEAN** = *true*

Is the target directory shared by others on the generic server. When true, the target directory is not deleted during a destroy operation; only the artifacts copied to it.



## jbossas.ServerV4

**Hierarchy** `jbossas.BaseServer` >> `generic.Container` >> `udm.BaseContainer` >> `udm.BaseConfigurationItem`

**Interfaces** `udm.Taggable`, `udm.ConfigurationItem`, `udm.Container`, `generic.GenericContainer`, `overthere.HostContainer`

Jboss 4 application server

### Public Properties

**ajpPort** : `INTEGER` = `8009`

AJP connector port

**controlPort** : `INTEGER` = `1099`

Connector control port



**home** : `STRING`

JBoss application server installation directory. e.g. `/opt/jboss/5.1.0.GA`



**host** : `CI<overthere.Host>`

Host upon which the container resides

**httpPort** : `INTEGER` = `8080`

HTTP/1.1 connector port

**serverName** : `STRING` = `default`

Server configuration name e.g. `default` or `minimal`

**envVars** : `MAP_STRING_STRING`

Environment variables for container

**tags** : `SET_OF_STRING`

The tags to map deployables to containers.

## Hidden Properties

**inspectTemplate** : *STRING* = *jboss/container/inspect.sh.ftl*

Script (a freemarker template) used to inspect the server

**jmsResourceFileSuffix** : *STRING* = *service*

Jms Resource File Suffix

**queryTemplate** : *STRING* = *jboss/container/query.sh.ftl*

Script (a freemarker template) used to query a mbean valuf from the server

**restartOrder** : *INTEGER* = *90*

The order of the restart container step in the step list.

**startOrder** : *INTEGER* = *90*

The order of the start container step in the step list.

**startScript** : *STRING* = *jboss/container/start.sh*

Script used to start the server

**startWaitTime** : *INTEGER* = *30*

Duration (in secs) to wait after the start server step has been executed

**stopOrder** : *INTEGER* = *10*

The order of the stop container step in the step list.

**stopScript** : *STRING* = *jboss/container/stop.sh*

Script used to stop the server

**stopWaitTime** : *INTEGER* = *10*

Duration (in secs) to wait after the stop server step has been executed

**version** : *INTEGER* = *4*

Version

**inspectClasspathResources** : *SET\_OF\_STRING*

Additional classpath resources that should be uploaded to the working directory before executing the inspect script.

**inspectScript** : *STRING*

Classpath to the script used to inspect the generic container.

**inspectTemplateClasspathResources** : *SET\_OF\_STRING*

Additional template classpath resources that should be uploaded to the working directory before executing the inspect script. The template is first rendered and the rendered content copied to a file, with the same name as the template, in the working directory.

**restartScript** : *STRING*

Classpath to the script used to restart the generic container.

**restartWaitTime** : *INTEGER* = *0*

The time to wait in seconds for a container restart action.

## jbossas.ServerV5

**Hierarchy** *jbossas.BaseServer* >> *generic.Container* >> *udm.BaseContainer* >> *udm.BaseConfigurationItem*

**Interfaces** udm.Taggable, udm.ConfigurationItem, udm.Container, generic.GenericContainer, overthere.HostContainer

Jboss 5 application server

## Public Properties

**ajpPort** : **INTEGER** = *8009*

AJP connector port

**controlPort** : **INTEGER** = *1099*

Connector control port



**home** : **STRING**

JBoss application server installation directory. e.g. /opt/jboss/5.1.0.GA



**host** : **CI**<overthere.Host>

Host upon which the container resides

**httpPort** : **INTEGER** = *8080*

HTTP/1.1 connector port

**serverName** : **STRING** = *default*

Server configuration name e.g. default or minimal

**envVars** : **MAP\_STRING\_STRING**

Environment variables for container

**tags** : **SET\_OF\_STRING**

The tags to map deployables to containers.

## Hidden Properties

**inspectTemplate** : *STRING* = *jboss/container/inspect.sh.ftl*

Script (a freemarker template) used to inspect the server

**jmsResourceFileSuffix** : *STRING* = *service*

Jms Resource File Suffix

**queryTemplate** : *STRING* = *jboss/container/query.sh.ftl*

Script (a freemarker template) used to query a mbean valuf from the server

**restartOrder** : *INTEGER* = *90*

The order of the restart container step in the step list.

**startOrder** : *INTEGER* = *90*

The order of the start container step in the step list.

**startScript** : *STRING* = *jboss/container/start.sh*

Script used to start the server

**startWaitTime** : *INTEGER* = *30*

Duration (in secs) to wait after the start server step has been executed

**stopOrder** : *INTEGER* = *10*

The order of the stop container step in the step list.

**stopScript** : *STRING* = *jboss/container/stop.sh*

Script used to stop the server

**stopWaitTime** : *INTEGER* = *10*

Duration (in secs) to wait after the stop server step has been executed

**version** : *INTEGER* = *5*

Version

**inspectClasspathResources** : *SET\_OF\_STRING*

Additional classpath resources that should be uploaded to the working directory before executing the inspect script.

**inspectScript** : *STRING*

Classpath to the script used to inspect the generic container.

**inspectTemplateClasspathResources** : *SET\_OF\_STRING*

Additional template classpath resources that should be uploaded to the working directory before executing the inspect script. The template is first rendered and the rendered content copied to a file, with the same name as the template, in the working directory.

**restartScript** : *STRING*

Classpath to the script used to restart the generic container.

**restartWaitTime** : *INTEGER* = *0*

The time to wait in seconds for a container restart action.

## jbossas.ServerV6

**Hierarchy** `jbossas.BaseServer` >> `generic.Container` >> `udm.BaseContainer` >> `udm.BaseConfigurationItem`

**Interfaces** `udm.Taggable`, `udm.ConfigurationItem`, `udm.Container`, `generic.GenericContainer`, `overthere.HostContainer`

JBoss 6 application server

## Public Properties

**ajpPort** : `INTEGER` = `8009`

AJP connector port

**controlPort** : `INTEGER` = `1090`

Connector control port

 **home** : `STRING`

JBoss application server installation directory. e.g. `/opt/jboss/5.1.0.GA`

 **host** : `CI<overthere.Host>`

Host upon which the container resides

**httpPort** : `INTEGER` = `8080`

HTTP/1.1 connector port

**serverName** : `STRING` = `default`

Server configuration name e.g. `default` or `minimal`

**envVars** : `MAP_STRING_STRING`

Environment variables for container

**tags** : `SET_OF_STRING`

The tags to map deployables to containers.

## Hidden Properties

**inspectTemplate** : *STRING* = *jboss/container/inspect.sh.ftl*

Script (a freemarker template) used to inspect the server

**jmsResourceFileSuffix** : *STRING* = *hornetq-jms*

Jms Resource File Suffix

**queryTemplate** : *STRING* = *jboss/container/query.sh.ftl*

Script (a freemarker template) used to query a mbean valuf from the server

**restartOrder** : *INTEGER* = *90*

The order of the restart container step in the step list.

**startOrder** : *INTEGER* = *90*

The order of the start container step in the step list.

**startScript** : *STRING* = *jboss/container/start.sh*

Script used to start the server

**startWaitTime** : *INTEGER* = *30*

Duration (in secs) to wait after the start server step has been executed

**stopOrder** : *INTEGER* = *10*

The order of the stop container step in the step list.

**stopScript** : *STRING* = *jboss/container/stop.sh*

Script used to stop the server

**stopWaitTime** : *INTEGER* = *10*

Duration (in secs) to wait after the stop server step has been executed

**version** : *INTEGER* = *6*

Version

**inspectClasspathResources** : *SET\_OF\_STRING*

Additional classpath resources that should be uploaded to the working directory before executing the inspect script.

**inspectScript** : *STRING*

Classpath to the script used to inspect the generic container.

**inspectTemplateClasspathResources** : *SET\_OF\_STRING*

Additional template classpath resources that should be uploaded to the working directory before executing the inspect script. The template is first rendered and the rendered content copied to a file, with the same name as the template, in the working directory.

**restartScript** : *STRING*

Classpath to the script used to restart the generic container.

**restartWaitTime** : *INTEGER* = *0*

The time to wait in seconds for a container restart action.

## jbossas.Topic

**Hierarchy** [jbossas.Resource](#) >> [generic.ProcessedTemplate](#) >> [generic.AbstractDeployedArtifact](#) >>

generic.AbstractDeployed >> udm.BaseDeployed >> udm.BaseConfigurationItem

**Interfaces** udm.Deployed, udm.ConfigurationItem

A JBoss topic

### Public Properties



**container** : CI<udm.Container>

The container on which this deployed runs.

**jndiName** : STRING

Jndi Name

**maxDepth** : INTEGER = 0

Max Depth

**deployable** : CI<udm.Deployable>

The deployable that this deployed is derived from.

## Hidden Properties

**createOrder** : INTEGER = 50

Create Order

**createVerb** : STRING = *Create*

Create Verb

**destroyOrder** : INTEGER = 40

Destroy Order

**destroyVerb** : STRING = *Destroy*

Destroy Verb

**inspectTemplate** : STRING = *jboss/topic/inspect.sh.ftl*

Inspect Template

**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

**targetDirectory** : STRING = *\${deployed.container.home}/server/\${deployed.container.serverName}/deploy*

Target Directory

**targetFile** : STRING = *\${deployed.deployable.name}-\${deployed.container.jmsResourceFileSuffix}.xml*

Target File

**template** : STRING = *jboss/topic/template\${deployed.container.version}.xml.ftl*

Template

**createTargetDirectory** : BOOLEAN = *false*

Create the target directory on the generic server if it does not exist.

**inspectClasspathResources** : SET\_OF\_STRING

Additional classpath resources that should be uploaded to the working directory before executing the inspect script.

**inspectScript** : STRING

Classpath to the script used to inspect the generic container.

**inspectTemplateClasspathResources** : SET\_OF\_STRING

Additional template classpath resources that should be uploaded to the working directory before executing the inspect script. The template is first rendered and the rendered content copied to a file, with the same name as the template, in the working directory.

**restartRequired** : BOOLEAN = *true*

Restart Required

**restartRequiredForNoop** : BOOLEAN = *false*



The generic container requires a restart for the NOOP action performed by this deployed.

**targetDirectoryShared** : **BOOLEAN** = *true*

Is the target directory shared by others on the generic server. When true, the target directory is not deleted during a destroy operation; only the artifacts copied to it.

## jbossas.TopicSpec

**Hierarchy** generic.Resource >> udm.BaseDeployable >> udm.BaseConfigurationItem

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

Specification for a JBoss topic

### Public Properties

**jndiName** : **STRING**

Jndi Name (string)

**maxDepth** : **STRING**

Max Depth (integer)

**tags** : **SET\_OF\_STRING**

The tags to map deployables to containers.

## jbossas.TransactionDataSource

**Hierarchy** [jbossas.Datasource](#) >> [jbossas.Resource](#) >> generic.ProcessedTemplate >>  
generic.AbstractDeployedArtifact >> generic.AbstractDeployed >> udm.BaseDeployed >>  
udm.BaseConfigurationItem

**Interfaces** udm.Deployed, udm.ConfigurationItem

A JBoss Local Transaction Datasource

## Public Properties

**connectionUrl** : *STRING*

Connection Url



**container** : *CI<udm.Container>*

The container on which this deployed runs.

**driverClass** : *STRING*

Driver Class

**jndiName** : *STRING*

Jndi Name

**maxPoolSize** : *INTEGER = 20*

Max Pool Size

**minPoolSize** : *INTEGER = 0*

Min Pool Size

**password** : *STRING*

Password

**userName** : *STRING*

User Name

**deployable** : *CI<udm.Deployable>*

The deployable that this deployed is derived from.

**useJavaContext** : *BOOLEAN = true*

Use Java Context

## Hidden Properties

**createOrder** : INTEGER = 50

Create Order

**createVerb** : STRING = *Create*

Create Verb

**destroyOrder** : INTEGER = 40

Destroy Order

**destroyVerb** : STRING = *Destroy*

Destroy Verb

**dsType** : STRING = *local-tx-datasource*

Ds Type

**inspectTemplate** : STRING = *jboss/datasource/inspect.sh.ftl*

Inspect Template

**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

**targetDirectory** : STRING = *\${deployed.container.home}/server/\${deployed.container.serverName}/deploy*

Target Directory

**targetFile** : STRING = *\${deployed.deployable.name}-ds.xml*

Target File

**template** : STRING = *jboss/datasource/template.xml.ftl*

Template

**createTargetDirectory** : BOOLEAN = *false*

Create the target directory on the generic server if it does not exist.

**inspectClasspathResources** : SET\_OF\_STRING

Additional classpath resources that should be uploaded to the working directory before executing the inspect script.

**inspectScript** : STRING

Classpath to the script used to inspect the generic container.

**inspectTemplateClasspathResources** : SET\_OF\_STRING

Additional template classpath resources that should be uploaded to the working directory before executing the inspect script. The template is first rendered and the rendered content copied to a file, with the same name as the template, in the working directory.

**restartRequired** : BOOLEAN = *true*

**Restart Required****restartRequiredForNoop** : **BOOLEAN** = *false*

The generic container requires a restart for the NOOP action performed by this deployed.

**targetDirectoryShared** : **BOOLEAN** = *true*

Is the target directory shared by others on the generic server. When true, the target directory is not deleted during a destroy operation; only the artifacts copied to it.

## **jbossas.TransactionDataSourceSpec**

**Hierarchy** generic.Resource >> udm.BaseDeployable >> udm.BaseConfigurationItem**Interfaces** udm.Taggable, udm.Deployable, udm.ConfigurationItem

A JBoss Local Transaction Datasource (deployable)

**Public Properties****connectionUrl** : **STRING**

Connection Url (string)

**driverClass** : **STRING**

Driver Class (string)

**jndiName** : **STRING**

Jndi Name (string)

**maxPoolSize** : **STRING**

Max Pool Size (integer)

**minPoolSize** : **STRING**

Min Pool Size (integer)

**password** : **STRING**

Password (string)

**tags** : **SET\_OF\_STRING**

The tags to map deployables to containers.

**useJavaContext** : **STRING**

Use Java Context (boolean)

**userName** : **STRING**

User Name (string)

## **jbossas.TransactionXADatasource**

**Hierarchy** **jbossas.Datasource** >> **jbossas.Resource** >> generic.ProcessedTemplate >>  
 generic.AbstractDeployedArtifact >> generic.AbstractDeployed >> udm.BaseDeployed >>  
 udm.BaseConfigurationItem

**Interfaces** udm.Deployed, udm.ConfigurationItem

A JBoss XA Datasource

## Public Properties

**connectionUrl** : *STRING*

Connection Url



**container** : *CI<udm.Container>*

The container on which this deployed runs.

**driverClass** : *STRING*

Driver Class

**jndiName** : *STRING*

Jndi Name

**maxPoolSize** : *INTEGER = 20*

Max Pool Size

**minPoolSize** : *INTEGER = 0*

Min Pool Size

**password** : *STRING*

Password

**userName** : *STRING*

User Name

**deployable** : *CI<udm.Deployable>*

The deployable that this deployed is derived from.

**useJavaContext** : *BOOLEAN = true*

Use Java Context

## Hidden Properties

**createOrder** : INTEGER = 50

Create Order

**createVerb** : STRING = *Create*

Create Verb

**destroyOrder** : INTEGER = 40

Destroy Order

**destroyVerb** : STRING = *Destroy*

Destroy Verb

**dsType** : STRING = *xa-datasource*

Ds Type

**inspectTemplate** : STRING = *jboss/datasource/inspect.sh.ftl*

Inspect Template

**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

**targetDirectory** : STRING = *\${deployed.container.home}/server/\${deployed.container.serverName}/deploy*

Target Directory

**targetFile** : STRING = *\${deployed.deployable.name}-ds.xml*

Target File

**template** : STRING = *jboss/datasource/template.xml.ftl*

Template

**createTargetDirectory** : BOOLEAN = *false*

Create the target directory on the generic server if it does not exist.

**inspectClasspathResources** : SET\_OF\_STRING

Additional classpath resources that should be uploaded to the working directory before executing the inspect script.

**inspectScript** : STRING

Classpath to the script used to inspect the generic container.

**inspectTemplateClasspathResources** : SET\_OF\_STRING

Additional template classpath resources that should be uploaded to the working directory before executing the inspect script. The template is first rendered and the rendered content copied to a file, with the same name as the template, in the working directory.

**restartRequired** : BOOLEAN = *true*

**Restart Required****restartRequiredForNoop** : **BOOLEAN** = *false*

The generic container requires a restart for the NOOP action performed by this deployed.

**targetDirectoryShared** : **BOOLEAN** = *true*

Is the target directory shared by others on the generic server. When true, the target directory is not deleted during a destroy operation; only the artifacts copied to it.

## jbossas.TransactionXADatasourceSpec

**Hierarchy** generic.Resource >> udm.BaseDeployable >> udm.BaseConfigurationItem**Interfaces** udm.Taggable, udm.Deployable, udm.ConfigurationItem

A JBoss XA Datasource (deployable)

**Public Properties****connectionUrl** : **STRING**

Connection Url (string)

**driverClass** : **STRING**

Driver Class (string)

**jndiName** : **STRING**

Jndi Name (string)

**maxPoolSize** : **STRING**

Max Pool Size (integer)

**minPoolSize** : **STRING**

Min Pool Size (integer)

**password** : **STRING**

Password (string)

**tags** : **SET\_OF\_STRING**

The tags to map deployables to containers.

**useJavaContext** : **STRING**

Use Java Context (boolean)

**userName** : **STRING**

User Name (string)

## jbossas.War

**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

**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* = *true*

Scan Placeholders

**tags** : *SET\_OF\_STRING*

The tags to map deployables to containers.

### 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

## jbossas.WarModule

**Hierarchy** [jbossas.Artifact](#) >> generic.CopiedArtifact >> generic.AbstractDeployedArtifact >> generic.AbstractDeployed >> udm.BaseDeployed >> udm.BaseConfigurationItem

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

War with values configured for a deployment

### Public Properties

 **container** : *CI<udm.Container>*

The container on which this deployed runs.

**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*

Create Order

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

Create Verb

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

Destroy Order

**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

**targetDirectory** : **STRING** = *\${deployed.container.home}/server/\${deployed.container.serverName}/deploy*

Target Directory

**targetFile** : **STRING** = *\${deployed.deployable.name}.war*

Target File

**createTargetDirectory** : **BOOLEAN** = *false*

Create the target directory on the generic server if it does not exist.

**inspectClasspathResources** : **SET\_OF\_STRING**

Additional classpath resources that should be uploaded to the working directory before executing the inspect script.

**inspectScript** : **STRING**

Classpath to the script used to inspect the generic container.

**inspectTemplateClasspathResources** : **SET\_OF\_STRING**

Additional template classpath resources that should be uploaded to the working directory before executing the inspect script. The template is first rendered and the rendered content copied to a file, with the same name as the template, in the working directory.

**restartRequired** : **BOOLEAN** = *true*

Restart Required

**restartRequiredForNoop** : **BOOLEAN** = *false*

The generic container requires a restart for the NOOP action performed by this deployed.

**targetDirectoryShared** : **BOOLEAN** = *true*

Is the target directory shared by others on the generic server. When true, the target directory is not deleted during a destroy operation; only the artifacts copied to it.

