

# Deployit Tomcat Plugin Manual

Version 3.6.0

# Table of Content

Preface	4
Overview	4
Features	4
Requirements	4
Usage in Deployment Packages	4
Tomcat Topology	5
Tomcat Server	5
Tomcat Common Context	5
Tomcat Virtual Host	5
Using the deployables and deployed	5
Deployable vs. Container table	5
Deployed Actions Table	6
Deploying Web Applications	6
Deploying Resources	7
Extension points	7
Adding additional context attributes	7
Adding new context elements	7
CI Reference	8
Configuration Item Overview	8
Deployable Configuration Items	8
Deployed Configuration Items	9
Topology Configuration Items	10
Virtual Deployable Configuration Items	10
Virtual Deployed Configuration Items	11
Virtual Topology Configuration Items	11
Configuration Item Details	11
jee.DataSourceSpec	11
jee.Ear	11
jee.EjbJar	12
jee.JndiResourceSpec	12
jee.MailSessionSpec	13
jee.QueueConnectionFactorySpec	13
jee.QueueSpec	13
jee.ResourceSpec	14
jee.TopicConnectionFactorySpec	14
jee.TopicSpec	14
jee.War	14
tomcat.ActiveMqConnectionFactory	15
tomcat.ActiveMqConnectionFactorySpec	17
tomcat.ActiveMqQueue	17
tomcat.ActiveMqQueueSpec	20
tomcat.ActiveMqTopic	20
tomcat.ActiveMqTopicSpec	23
tomcat.CommonContext	23
tomcat.ContextContainer	24
tomcat.ContextElement	24
tomcat.ContextWarModule	25
tomcat.DataSource	28
tomcat.DataSourceLink	31
tomcat.DataSourceLinkSpec	33
tomcat.DataSourceSpec	33
tomcat.JeeActiveMqConnectionFactory	34
tomcat.JeeActiveMqQueue	37
tomcat.JeeActiveMqTopic	39
tomcat.JeeWebsphereMqQueue	41
tomcat.JeeWebsphereMqQueueConnectionFactory	43
tomcat.JeeWebsphereMqTopic	45

tomcat.JeeWebsphereMqTopicConnectionFactory	47
tomcat.JndiContextElement	49
tomcat.MailSession	51
tomcat.MailSessionSpec	53
tomcat.ResourceLink	53
tomcat.ResourceLinkSpec	56
tomcat.Server	56
tomcat.VirtualHost	58
tomcat.WarModule	58
tomcat.WebsphereMqQueue	61
tomcat.WebsphereMqQueueConnectionFactory	63
tomcat.WebsphereMqQueueConnectionFactorySpec	65
tomcat.WebsphereMqQueueSpec	65
tomcat.WebsphereMqTopic	66
tomcat.WebsphereMqTopicConnectionFactory	68
tomcat.WebsphereMqTopicConnectionFactorySpec	70
tomcat.WebsphereMqTopicSpec	70

# Preface

This document describes the functionality provided by the Tomcat plugin.

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

## Overview

The Tomcat plugin is a Deployit plugin that adds capability for managing deployments and resources to a Tomcat Server. Standard support includes deploying/undeploying web applications, datasources, mail sessions, resource links, ActiveMQ and WebsphereMQFeatures resources. The plugin can easily be extended to support more deployment options or management of new artifacts/resources on Tomcat.

## Features

- Deployment of web applications (WAR) to a Tomcat virtual host.
- Deployment of resources to an application context on a Tomcat virtual host.
- Deployment of resources to the common Tomcat context ie. \$TOMCAT\_HOME/conf/context.xml.
- Deployment of resource links.
- Supported JEE Resources
  - Datasource
  - JMS Queue
  - JMS Queue Connection Factory
  - JMS Topic
  - JMS Topic Connection Factory
  - Mail Session
- Supported Messaging Middleware
  - Active MQ
  - Websphere MQ
- Support for Tomcat Database Connection Pool (DBCP) Configurations
- Stopping and starting a Tomcat Server via control tasks.

## Requirements

- **Deployit requirements**
  - **Deployit:** version 3.6+
  - **Tomcat versions:** 5.5.x, 6.x, 7.x (Unix and Windows)
- **Infrastructural requirements**
  - **User credentials** for accessing the Host running the Tomcat 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 Tomcat specific deployment package. It contain declarations for a [war](#), 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: SampleApp-1.0.war
CI-Name: SampleApp
CI-Type: jee.War
Name: testDataSource
CI-Type: tomcat.DataSourceSpec
CI-jndiName: jdbc/sampleDataSource
CI-url: jdbc:mysql://localhost/test
CI-driverClassName: com.mysql.jdbc.Driver
CI-username: {{DATABASE_USERNAME}}
CI-password: {{DATABASE_PASSWORD}}
Name: sampleQueue
CI-Type: jee.QueueSpec
CI-jndiNames: jms/testQueue
Name: sampleQCf
CI-Type: jee.QueueConnectionFactorySpec
CI-jndiNames: jms/sampleQCf

## Tomcat Topology

The Tomcat plugin allows for the modelling of a Tomcat installation into the Deployit infrastructure as a hierarchical set of containers. [Tomcat Server](#), [Tomcat Common Context](#) and [Tomcat Virtual Hosts](#) containers can then be included into Deployit environment definitions, to which deployables can be deployed to.

### Tomcat Server

Models a Tomcat installation running on a host. This container must be defined under any [Host](#) in the Deployit infrastructure. The container supports OS specific stop and start commands used to control the stopping and starting of the Tomcat Server. In addition, stop and start wait times can be specified. Deployit will wait for the specified amount of time to elapse after the execution of a stop/start command.

### Tomcat Common Context

Models the context.xml file present in the \$TOMCAT\_HOME/conf directory. This container must be defined under a [Tomcat Server](#) container. Only a single instance should be defined. Any [Tomcat Resource](#) can be deployed to this container.

### Tomcat Virtual Host

Models a Tomcat virtual host definition, ie. \$TOMCAT\_HOME/conf/[enginename]/[hostname]. This container must be defined under a [Tomcat Server](#) container. Multiple instances can be defined. [Web Applications](#) and [Tomcat Resource](#) can be deployed to this container.

## Using the deployables and deployed

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

### Deployable vs. Container table

Deployable	Container	Deployed
jee.War	tomcat.VirtualHost	tomcat.WarModule
tomcat.ResourceLinkSpec	tomcat.VirtualHost tomcat.CommonContext	tomcat.ResourceLink
tomcat.DataSourceLinkSpec	tomcat.VirtualHost tomcat.CommonContext	tomcat.DataSourceLink

jee.DataSourceSpec	tomcat.VirtualHost tomcat.CommonContext	tomcat.DataSource
jee.MailSessionSpec	tomcat.VirtualHost tomcat.CommonContext	tomcat.MailSession
jee.QueueConnectionFactorySpec tomcat.ActiveMqConnectionFactorySpec tomcat.WebsphereMqQueueConnectionFactorySpec	tomcat.VirtualHost tomcat.CommonContext	tomcat.ActiveMqConnectionFactory tomcat.WebsphereMqQueueConnectionFactory
jee.TopicConnectionFactorySpec tomcat.WebsphereMqTopicConnectionFactorySpec	tomcat.VirtualHost tomcat.CommonContext	tomcat.WebsphereMqTopicConnectionFactory
jee.QueueSpec tomcat.ActiveMqQueueSpec tomcat.WebsphereMqQueueSpec	tomcat.VirtualHost tomcat.CommonContext	tomcat.ActiveMqQueue tomcat.WebsphereMqQueue
jee.TopicSpec tomcat.ActiveMqTopicSpec tomcat.WebsphereMqTopicSpec	tomcat.VirtualHost tomcat.CommonContext	tomcat.ActiveMqTopic tomcat.WebsphereMqTopic

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

## Deployed Actions Table

Deployed	Actions performed for operations		
	Create	Destroy	Modify
tomcat.WarModule	<ul style="list-style-type: none"> <li>Stop Tomcat</li> <li>Create a context xml using the context root name specified on the war in the virtual host's directory. eg. \$TOMCAT_HOME/conf/Catalina/localhost</li> <li>Upload war to virtual host's application base directory. eg. \$TOMCAT_HOME/webapps</li> <li>Start Tomcat</li> </ul>	<ul style="list-style-type: none"> <li>Stop Tomcat</li> <li>Delete war from virtual host's application base directory.</li> <li>Delete exploded war folder from virtual host's application base directory.</li> <li>Remove attributes, e.g. docBase, from the Context element in the application specific context xml</li> <li>Start Tomcat</li> </ul>	<ul style="list-style-type: none"> <li>Stop Tomcat</li> <li>Delete war from virtual host's application base directory.</li> <li>Delete exploded war folder from virtual host's application base directory.</li> <li>Remove attributes, e.g. docBase, from the Context element in the application specific context xml</li> <li>If war context root changed; rename application specific context xml in the virtual host's directory to the new context root name.</li> <li>Modify attributes, e.g. docBase, from the Context element in the application specific context xml</li> <li>Upload war to virtual host's application base directory. eg. \$TOMCAT_HOME/webapps</li> <li>Start Tomcat</li> </ul>
Any tomcat.JndiContextElement (tomcat.DataSource, tomcat.ActiveMqQueue, tomcat.ResourceLink, etc)	<ul style="list-style-type: none"> <li>Stop Tomcat</li> <li>Create xml fragment, e.g. &lt;Resource name='mail/Session' auth='Container' type='javax.mail.Session' mail.smtp.host='localhost'/&gt;</li> <li>Insert xml fragment into context xml</li> <li>Start Tomcat</li> </ul>	<ul style="list-style-type: none"> <li>Stop Tomcat</li> <li>Find xml fragment in context xml using the name attribute</li> <li>Delete xml fragment from context xml</li> <li>Start Tomcat</li> </ul>	<ul style="list-style-type: none"> <li>Stop Tomcat</li> <li>Find xml fragment in context xml using the name attribute</li> <li>Delete xml fragment from context xml</li> <li>Create xml fragment, e.g. &lt;Resource name='mail/Session' auth='Container' type='javax.mail.Session' mail.smtp.host='localhost'/&gt;</li> <li>Insert xml fragment into context xml</li> <li>Start Tomcat</li> </ul>

## Deploying Web Applications

**Web Applications** can be deployed to a **Tomcat Virtual Host**. The context root must be specified. The context root is

used to create the corresponding application specific context xml in the virtual host.

## Deploying Resources

**Tomcat Resource** can be deployed to a **Tomcat Virtual Host** or **Tomcat Common Context**. Every resource has an optional context property. This property is always set to *context* when the resource is deployed to **Tomcat Common Context**. When the resource is deployed to a **Tomcat Virtual Host**, the user can specify the name of the context to which the resource must be defined. When left blank, the plugin can default the value to that of the war in the current deployment. There must only be a single war in the current deployment for the defaulting to work. Example : deploying a war with context root set to *sample*, and a datasource to a virtual host will automatically result in the *context* property of the datasource been set to *sample*.

## Extension points

The Tomcat plugin is designed to be extended through Deployit's Plugin API type system. Refer to *Customization Manual* for an explanation of the type system.

### Adding additional context attributes

Tomcat context's have several attributes that can be set for a war. By default, the Tomcat Plugin only sets the unpackWAR and docBase attributes. When additional attributes are required, performing a type modification on the corresponding tomcat.WarModule can be done. For example, adding support for the `antiResourceLocking` attribute in `synthetic.xml`:

```
<type-modification type="tomcat.WarModule">
  <property name="antiResourceLocking" default="true" hidden="true"/>
  <property name="contextElementPropertyMapping" kind="map_string_string" hidden="true"
    default="unpackWAR:, docBase:, antiResourceLocking:"
  </type-modification>
```

An additional hidden property `antiResourceLocking` with a default value of "true" is added to the deployed. The property is added to the `contextElementPropertyMapping` property's default value. This property is used to translate properties on the deployed to their xml equivalent in the Tomcat context. The left hand represents the property name in the deployed, while the right hand side represents the attribute name in the xml. When the right hand side is blank, the property name and xml equivalent are the same.

The same technique can be used to add attributes to other Tomcat resources like datasource, queues, etc.

### Adding new context elements

Tomcat context's have several nested components or special features (Context Parameters, Lifecycle Listeners, etc.), that can be defined in the xml.

It is possible to model these features in the Tomcat Plugin and perform subsequence deployments of these resources. For example, adding support for the Context Parameter feature.

```
<Context>
  ...
  <Parameter name="companyName" value="My Company, Incorporated"
    override="false"/>
  ...
</Context>
```

Add to `synthetic.xml`:

<pre>&lt;type type="tomcat.ContextParameter" extends="tomcat.ContextElement"   deployable-type="tomcat.ContextParameterSpec"&gt;   &lt;generate-deployable type="tomcat.ContextParameterSpec" extends="tomcat.ContextElement"/&gt;   &lt;property name="paramName" description="The name of the context parameter"/&gt;   &lt;property name="value"/&gt;   &lt;property name="override" kind="boolean" required="false" default="false"/&gt;    &lt;!--inherited hidden --&gt;   &lt;property name="elementTag" default="Parameter" hidden="true"/&gt;   &lt;property name="elementName" default="paramName" hidden="true"/&gt;   &lt;property name="elementPropertyMapping" kind="map_string_string" hidden="true"     default="paramName:name, value:, override:"/&gt; &lt;/type&gt;</pre>

# CI Reference

## Configuration Item Overview

### Deployable Configuration Items



CI	Description
<a href="#">jee.DataSourceSpec</a>	Deployable Java EE DataSource accessible via JNDI
<a href="#">jee.Ear</a>	Deployable EAR artifact
<a href="#">jee.EjbJar</a>	Deployable EJB JAR artifact
<a href="#">jee.MailSessionSpec</a>	Deployable Java EE Mail Session accessible via JNDI
<a href="#">jee.QueueConnectionFactorySpec</a>	Deployable Java EE JMS QueueConnectionFactory accessible via JNDI
<a href="#">jee.QueueSpec</a>	Deployable Java EE JMS Queue accessible via JNDI
<a href="#">jee.TopicConnectionFactorySpec</a>	Deployable Java EE JMS TopicConnectionFactory accessible via JNDI
<a href="#">jee.TopicSpec</a>	Deployable Java EE JMS Topic accessible via JNDI
<a href="#">jee.War</a>	Deployable WAR artifact
<a href="#">tomcat.ActiveMqConnectionFactorySpec</a>	ActiveMq Connection Factory installed to a Tomcat Virtual Host or the Common Context (deployable)
<a href="#">tomcat.ActiveMqQueueSpec</a>	ActiveMq queue installed to a Tomcat Virtual Host or the Common Context (deployable)
<a href="#">tomcat.ActiveMqTopicSpec</a>	ActiveMq topic installed to a Tomcat Virtual Host or the Common Context (deployable)
<a href="#">tomcat.DataSourceLinkSpec</a>	To create a datasoure link to a global JNDI resource
<a href="#">tomcat.DataSourceSpec</a>	DataSource installed to a Tomcat Virtual Host or the Common Context (deployable)
<a href="#">tomcat.MailSessionSpec</a>	Mail Session installed to a Tomcat Virtual Host or the Common Context (deployable)
<a href="#">tomcat.ResourceLinkSpec</a>	To create a link to a global JNDI resource
<a href="#">tomcat.WebsphereMqQueueConnectionFactorySpec</a>	WebsphereMq Queue Connection Factory installed to a Tomcat Virtual Host or the Common Context (deployable)
<a href="#">tomcat.WebsphereMqQueueSpec</a>	WebsphereMq queue installed to a Tomcat Virtual Host or the Common Context (deployable)
<a href="#">tomcat.WebsphereMqTopicConnectionFactorySpec</a>	WebsphereMq Topic Connection Factory installed to a Tomcat Virtual Host or the Common Context (deployable)
<a href="#">tomcat.WebsphereMqTopicSpec</a>	WebsphereMq topic installed to a Tomcat Virtual Host or the Common Context (deployable)

## Deployed Configuration Items

CI	Description
<a href="#">tomcat.ActiveMqConnectionFactory</a>	ActiveMq Connection Factory installed to a Tomcat Virtual Host or the Common Context
<a href="#">tomcat.ActiveMqQueue</a>	ActiveMq queue installed to a Tomcat Virtual Host or the Common Context
<a href="#">tomcat.ActiveMqTopic</a>	ActiveMq topic installed to a Tomcat Virtual Host or the Common Context
<a href="#">tomcat.DataSource</a>	DataSource installed to a Tomcat Virtual Host or the Common Context
<a href="#">tomcat.DataSourceLink</a>	To create a datasoure link to a global JNDI resource
<a href="#">tomcat.JeeActiveMqConnectionFactory</a>	ActiveMq Connection Factory installed to a Tomcat Virtual Host or the Common Context
<a href="#">tomcat.JeeActiveMqQueue</a>	ActiveMq queue installed to a Tomcat Virtual Host or the Common Context
<a href="#">tomcat.JeeActiveMqTopic</a>	ActiveMq topic installed to a Tomcat Virtual Host or the Common Context
<a href="#">tomcat.JeeWebsphereMqQueue</a>	WebsphereMq queue installed to a Tomcat Virtual Host or the Common Context
<a href="#">tomcat.JeeWebsphereMqQueueConnectionFactory</a>	WebsphereMq Queue Connection Factory installed to a Tomcat Virtual Host or the Common Context
<a href="#">tomcat.JeeWebsphereMqTopic</a>	WebsphereMq topic installed to a Tomcat Virtual Host or the Common Context
<a href="#">tomcat.JeeWebsphereMqTopicConnectionFactory</a>	WebsphereMq Topic Connection Factory installed to a Tomcat Virtual Host or the Common Context
<a href="#">tomcat.MailSession</a>	Mail Session installed to a Tomcat Virtual Host or the Common Context
<a href="#">tomcat.ResourceLink</a>	To create a link to a global JNDI resource
<a href="#">tomcat.WarModule</a>	War installed to a Tomcat Virtual Host
<a href="#">tomcat.WebsphereMqQueue</a>	WebsphereMq queue installed to a Tomcat Virtual Host or the Common Context
<a href="#">tomcat.WebsphereMqQueueConnectionFactory</a>	WebsphereMq Queue Connection Factory installed to a Tomcat Virtual Host or the Common Context
<a href="#">tomcat.WebsphereMqTopic</a>	WebsphereMq topic installed to a Tomcat Virtual Host or the Common Context
<a href="#">tomcat.WebsphereMqTopicConnectionFactory</a>	WebsphereMq Topic Connection Factory installed to a Tomcat Virtual Host or the Common Context

## Topology Configuration Items

CI	Description
<a href="#">tomcat.CommonContext</a>	The Context element information will be loaded by all webapps
<a href="#">tomcat.Server</a>	Tomcat server on host
<a href="#">tomcat.VirtualHost</a>	represents a virtual host, which is an association of a network name for a server (such as 'www

## Virtual Deployable Configuration Items

CI	Description
<a href="#">jee.JndiResourceSpec</a>	Deployable Java EE Resource accessible via JNDI
<a href="#">jee.ResourceSpec</a>	Description unavailable

## Virtual Deployed Configuration Items

CI	Description
<a href="#">tomcat.ContextElement</a>	Base type for all Tomcat resources
<a href="#">tomcat.ContextWarModule</a>	War Module that has an associated context
<a href="#">tomcat.JndiContextElement</a>	Resource installed to a Tomcat Virtual Host or the Common Context

## Virtual Topology Configuration Items

CI	Description
<a href="#">tomcat.ContextContainer</a>	Models a Tomcat context on a Tomcat server in Deployit's Infrastructure

## Configuration Item Details

### [jee.DataSourceSpec](#)

**Hierarchy** [jee.JndiResourceSpec](#) >> [jee.ResourceSpec](#) >> [udm.BaseDeployable](#) >> [udm.BaseConfigurationItem](#)

**Interfaces** [udm.Taggable](#), [udm.Deployable](#), [udm.ConfigurationItem](#)

Deployable Java EE DataSource accessible via JNDI.

Public Properties
<b><a href="#">jndiName</a></b> : <a href="#">STRING</a>
Name used to lookup this resource in JNDI
<b><a href="#">tags</a></b> : <a href="#">SET_OF_STRING</a>
The tags to map deployables to containers.

### [jee.Ear](#)

**Hierarchy** [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](#)

Deployable EAR artifact.

**Public Properties****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

**jee.EjbJar****Hierarchy** 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

Deployable EJB JAR artifact.

**Public Properties****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

**jee.JndiResourceSpec****Hierarchy** [jee.ResourceSpec](#) >> udm.BaseDeployable >> udm.BaseConfigurationItem**Interfaces** udm.Taggable, udm.Deployable, udm.ConfigurationItem

Deployable Java EE Resource accessible via JNDI.

### Public Properties

**jndiName** : `STRING`

Name used to lookup this resource in JNDI

**tags** : `SET_OF_STRING`

The tags to map deployables to containers.

## jee.MailSessionSpec

**Hierarchy** [jee.JndiResourceSpec](#) >> [jee.ResourceSpec](#) >> [udm.BaseDeployable](#) >> [udm.BaseConfigurationItem](#)

**Interfaces** [udm.Taggable](#), [udm.Deployable](#), [udm.ConfigurationItem](#)

Deployable Java EE Mail Session accessible via JNDI.

### Public Properties

**jndiName** : `STRING`

Name used to lookup this resource in JNDI

**tags** : `SET_OF_STRING`

The tags to map deployables to containers.

## jee.QueueConnectionFactorySpec

**Hierarchy** [jee.JndiResourceSpec](#) >> [jee.ResourceSpec](#) >> [udm.BaseDeployable](#) >> [udm.BaseConfigurationItem](#)

**Interfaces** [udm.Taggable](#), [udm.Deployable](#), [udm.ConfigurationItem](#)

Deployable Java EE JMS QueueConnectionFactory accessible via JNDI.

### Public Properties

**jndiName** : `STRING`

Name used to lookup this resource in JNDI

**tags** : `SET_OF_STRING`

The tags to map deployables to containers.

## jee.QueueSpec

**Hierarchy** [jee.JndiResourceSpec](#) >> [jee.ResourceSpec](#) >> [udm.BaseDeployable](#) >> [udm.BaseConfigurationItem](#)

**Interfaces** [udm.Taggable](#), [udm.Deployable](#), [udm.ConfigurationItem](#)

Deployable Java EE JMS Queue accessible via JNDI.

### Public Properties

**jndiName** : `STRING`

Name used to lookup this resource in JNDI

**tags** : `SET_OF_STRING`

The tags to map deployables to containers.

## jee.ResourceSpec

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

Description unavailable

### Public Properties

**tags** : SET\_OF\_STRING

The tags to map deployables to containers.

## jee.TopicConnectionFactorySpec

**Hierarchy** [jee.JndiResourceSpec](#) >> [jee.ResourceSpec](#) >> udm.BaseDeployable >> udm.BaseConfigurationItem  
**Interfaces** udm.Taggable, udm.Deployable, udm.ConfigurationItem

Deployable Java EE JMS TopicConnectionFactory accessible via JNDI.

### Public Properties

**jndiName** : STRING

Name used to lookup this resource in JNDI

**tags** : SET\_OF\_STRING

The tags to map deployables to containers.

## jee.TopicSpec

**Hierarchy** [jee.JndiResourceSpec](#) >> [jee.ResourceSpec](#) >> udm.BaseDeployable >> udm.BaseConfigurationItem  
**Interfaces** udm.Taggable, udm.Deployable, udm.ConfigurationItem

Deployable Java EE JMS Topic accessible via JNDI.

### Public Properties

**jndiName** : STRING

Name used to lookup this resource in JNDI

**tags** : SET\_OF\_STRING

The tags to map deployables to containers.

## jee.War

**Hierarchy** 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

Deployable WAR artifact.

**Public Properties****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

**tomcat.ActiveMqConnectionFactory**

**Hierarchy** `tomcat.JeeActiveMqConnectionFactory` >> `tomcat.JndiContextElement` >> `tomcat.ContextElement` >>  
`generic.ProcessedTemplate` >> `generic.AbstractDeployedArtifact` >> `generic.AbstractDeployed` >>  
`udm.BaseDeployed` >> `udm.BaseConfigurationItem`

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

ActiveMq Connection Factory installed to a Tomcat Virtual Host or the Common Context

**Public Properties****brokerName** : STRING

Broker Name

**brokerUrl** : STRING

Broker Url

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

The container on which this deployed runs.

**description** : STRING = *JMS Connection Factory*

Description

**jndiName** : STRING

Name used to lookup the resource in JNDI.

**context** : STRING

The name of the context that this resource is associated with

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

The deployable that this deployed is derived from.

## Hidden Properties

**auth** : *STRING* = *Container*

Auth

**createOrder** : *INTEGER* = *60*

Create Order

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

Create Verb

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

Destroy Order

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

Destroy Verb

**elementName** : *STRING* = *\${deployed.jndiName}*

Element Name

**elementPropertyMapping** : *MAP\_STRING\_STRING* = *{brokerUrl=brokerURL, description=, brokerName=, jndiName=name, factory=, auth=, resourceType=type}*

Element Property Mapping

**elementTag** : *STRING* = *Resource*

Element Tag

**factory** : *STRING* = *org.apache.activemq.jndi.JNDIReferenceFactory*

Factory

**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

**resourceType** : *STRING* = *org.apache.activemq.ActiveMQConnectionFactory*

Resource Type

**targetDirectory** : *STRING* = *\${deployed.container.contextDirectory}*

Target Directory

**targetFile** : *STRING* = *\${deployed.context}.xml*

Target File

**template** : *STRING* = *tc/context/context-element.ftl*

Template

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

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



**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.

## tomcat.ActiveMqConnectionFactorySpec

**Hierarchy** [jee.QueueConnectionFactorySpec](#) >> [jee.JndiResourceSpec](#) >> [jee.ResourceSpec](#) >>

[udm.BaseDeployable](#) >> [udm.BaseConfigurationItem](#)

**Interfaces** [udm.Taggable](#), [udm.Deployable](#), [udm.ConfigurationItem](#)

ActiveMq Connection Factory installed to a Tomcat Virtual Host or the Common Context (deployable)

### Public Properties

**brokerName** : **STRING**

Broker Name

**brokerUrl** : **STRING**

Broker Url

**context** : **STRING**

The name of the context that this resource is associated with

**description** : **STRING**

Description

**jndiName** : **STRING**

Name used to lookup the resource in JNDI.

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

The tags to map deployables to containers.

## tomcat.ActiveMqQueue

**Hierarchy** [tomcat.JeeActiveMqQueue](#) >> [tomcat.JndiContextElement](#) >> [tomcat.ContextElement](#) >>

[generic.ProcessedTemplate](#) >> [generic.AbstractDeployedArtifact](#) >> [generic.AbstractDeployed](#) >>

[udm.BaseDeployed](#) >> [udm.BaseConfigurationItem](#)

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

ActiveMq queue installed to a Tomcat Virtual Host or the Common Context

## Public Properties



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

The container on which this deployed runs.

**description** : `STRING`

Description

**jndiName** : `STRING`

Name used to lookup the resource in JNDI.

**physicalName** : `STRING`

Physical Name

**context** : `STRING`

The name of the context that this resource is associated with

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

The deployable that this deployed is derived from.

## Hidden Properties

**auth** : *STRING* = *Container*

Auth

**createOrder** : *INTEGER* = *60*

Create Order

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

Create Verb

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

Destroy Order

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

Destroy Verb

**elementName** : *STRING* = *\${deployed.jndiName}*

Element Name

**elementPropertyMapping** : *MAP\_STRING\_STRING* = *{description=, jndiName=name, physicalName=, factory=, auth=, resourceType=type}*

Element Property Mapping

**elementTag** : *STRING* = *Resource*

Element Tag

**factory** : *STRING* = *org.apache.activemq.jndi.JNDIReferenceFactory*

Factory

**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

**resourceType** : *STRING* = *org.apache.activemq.command.ActiveMQQueue*

Resource Type

**targetDirectory** : *STRING* = *\${deployed.container.contextDirectory}*

Target Directory

**targetFile** : *STRING* = *\${deployed.context}.xml*

Target File

**template** : *STRING* = *tc/context/context-element.ftl*

Template

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

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

**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.

## tomcat.ActiveMqQueueSpec

**Hierarchy** [jee.QueueSpec](#) >> [jee.JndiResourceSpec](#) >> [jee.ResourceSpec](#) >> [udm.BaseDeployable](#) >> [udm.BaseConfigurationItem](#)

**Interfaces** [udm.Taggable](#), [udm.Deployable](#), [udm.ConfigurationItem](#)

ActiveMq queue installed to a Tomcat Virtual Host or the Common Context (deployable)

### Public Properties

**context** : **STRING**

The name of the context that this resource is associated with

**description** : **STRING**

Description

**jndiName** : **STRING**

Name used to lookup the resource in JNDI.

**physicalName** : **STRING**

Physical Name

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

The tags to map deployables to containers.

## tomcat.ActiveMqTopic

**Hierarchy** [tomcat.JeeActiveMqTopic](#) >> [tomcat.JndiContextElement](#) >> [tomcat.ContextElement](#) >> [generic.ProcessedTemplate](#) >> [generic.AbstractDeployedArtifact](#) >> [generic.AbstractDeployed](#) >> [udm.BaseDeployed](#) >> [udm.BaseConfigurationItem](#)

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

ActiveMq topic installed to a Tomcat Virtual Host or the Common Context

## Public Properties



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

The container on which this deployed runs.

**description** : `STRING`

Description

**jndiName** : `STRING`

Name used to lookup the resource in JNDI.

**physicalName** : `STRING`

Physical Name

**context** : `STRING`

The name of the context that this resource is associated with

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

The deployable that this deployed is derived from.

## Hidden Properties

**auth** : *STRING = Container*

Auth

**createOrder** : *INTEGER = 60*

Create Order

**createVerb** : *STRING = Create*

Create Verb

**destroyOrder** : *INTEGER = 40*

Destroy Order

**destroyVerb** : *STRING = Destroy*

Destroy Verb

**elementName** : *STRING = \${deployed.jndiName}*

Element Name

**elementPropertyMapping** : *MAP\_STRING\_STRING = {description=, jndiName=name, physicalName=, factory=, auth=, resourceType=type}*

Element Property Mapping

**elementTag** : *STRING = Resource*

Element Tag

**factory** : *STRING = org.apache.activemq.jndi.JNDIReferenceFactory*

Factory

**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

**resourceType** : *STRING = org.apache.activemq.command.ActiveMQTopic*

Resource Type

**targetDirectory** : *STRING = \${deployed.container.contextDirectory}*

Target Directory

**targetFile** : *STRING = \${deployed.context}.xml*

Target File

**template** : *STRING = tc/context/context-element.ftl*

Template

**createTargetDirectory** : *BOOLEAN = false*

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

**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.

## tomcat.ActiveMqTopicSpec

**Hierarchy** [jee.TopicSpec](#) >> [jee.JndiResourceSpec](#) >> [jee.ResourceSpec](#) >> [udm.BaseDeployable](#) >> [udm.BaseConfigurationItem](#)

**Interfaces** [udm.Taggable](#), [udm.Deployable](#), [udm.ConfigurationItem](#)

ActiveMq topic installed to a Tomcat Virtual Host or the Common Context (deployable)

### Public Properties

**context** : **STRING**

The name of the context that this resource is associated with

**description** : **STRING**

Description

**jndiName** : **STRING**

Name used to lookup the resource in JNDI.

**physicalName** : **STRING**

Physical Name

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

The tags to map deployables to containers.

## tomcat.CommonContext

**Hierarchy** [tomcat.ContextContainer](#) >> [generic.NestedContainer](#) >> [udm.BaseContainer](#) >> [udm.BaseConfigurationItem](#)

**Interfaces** [udm.Taggable](#), [udm.ConfigurationItem](#), [generic.GenericContainer](#), [udm.Container](#), [overthere.HostContainer](#)

The Context element information will be loaded by all webapps.

### Public Properties



**server** : **CI**<[tomcat.Server](#)>

Server

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

Environment variables for container

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

The tags to map deployables to containers.

**Hidden Properties****contextDirectory** : *STRING* = *\${container.server.home}/conf*

Context Directory

**tomcat.ContextContainer****Hierarchy** generic.NestedContainer >> udm.BaseContainer >> udm.BaseConfigurationItem**Interfaces** udm.Taggable, udm.ConfigurationItem, generic.GenericContainer, udm.Container, overthere.HostContainer

Models a Tomcat context on a Tomcat server in Deployit's Infrastructure.

**Public Properties****contextDirectory** : *STRING*

Context Directory

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

Environment variables for container

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

The tags to map deployables to containers.

**tomcat.ContextElement****Hierarchy** generic.ProcessedTemplate >> generic.AbstractDeployedArtifact >> generic.AbstractDeployed >>  
udm.BaseDeployed >> udm.BaseConfigurationItem**Interfaces** udm.Deployed, udm.ConfigurationItem

Base type for all Tomcat resources

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

The container on which this deployed runs.

**context** : *STRING*

The name of the context that this resource is associated with

**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*

The order of the step in the step list 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

**targetDirectory** : **STRING**

Path to which artifact must be copied to on the generic server.

**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.

**restartRequired** : **BOOLEAN** = *false*

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

**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.

## tomcat.ContextWarModule

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

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

War Module that has an associated context.xml

## Public Properties



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

The container on which this deployed runs.

**contextRoot** : `STRING = ${deployed.name}`

Context root under which the web application can be access

**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

**contextSubject** : *STRING* = *war settings*

Context Subject

**contextTemplate** : *STRING*

Freemarker template used to create the context xml for the War

**contextXmlTargetDirectory** : *STRING*

Target directory to which the context xml must to copied to on the target system

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

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

**createOrderOfContextXml** : *INTEGER* = *60*

The order of the create context step in the step list 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.

**destroyOrderOfContextXml** : *INTEGER* = *40*

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

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

Destroy Verb

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

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

**modifyOrderOfContextXml** : *INTEGER* = *60*

The order of the rename context root 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*

Path to which artifact must be copied to on the generic server.

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

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

**deleteGeneratedResources** : *SET\_OF\_STRING*

Absolute paths to files on the target system that was generated and must be clean up during a destroy operation

**restartRequired** : *BOOLEAN* = *false*

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

**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.

---

## tomcat.DataSource

**Hierarchy** [tomcat.JndiContextElement](#) >> [tomcat.ContextElement](#) >> generic.ProcessedTemplate >>  
generic.AbstractDeployedArtifact >> generic.AbstractDeployed >> udm.BaseDeployed >>  
udm.BaseConfigurationItem

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

DataSource installed to a Tomcat Virtual Host or the Common Context

## Public Properties

 <b>container</b> : <code>CI&lt;udm.Container&gt;</code>	The container on which this deployed runs.
<b>driverClassName</b> : <code>STRING</code>	The fully qualified Java class name of the JDBC driver to be used.
<b>jndiName</b> : <code>STRING</code>	Name used to lookup the resource in JNDI.
<b>password</b> : <code>STRING</code>	The connection password to be passed to our JDBC driver to establish a connection.
<b>url</b> : <code>STRING</code>	The connection URL to be passed to our JDBC driver to establish a connection.
<b>username</b> : <code>STRING</code>	The connection username to be passed to our JDBC driver to establish a connection.
<b>connectionProperties</b> : <code>MAP_STRING_STRING</code>	The connection properties that will be sent to our JDBC driver when establishing new connections.
<b>context</b> : <code>STRING</code>	The name of the context that this resource is associated with
<b>deployable</b> : <code>CI&lt;udm.Deployable&gt;</code>	The deployable that this deployed is derived from.
<b>initialSize</b> : <code>INTEGER = 0</code>	The initial number of connections that are created when the pool is started.
<b>maxActive</b> : <code>INTEGER = 8</code>	The maximum number of active connections that can be allocated from this pool at the same time, or negative for no limit.
<b>maxIdle</b> : <code>INTEGER = 8</code>	The maximum number of connections that can remain idle in the pool, without extra ones being released, or negative for no limit.
<b>maxWait</b> : <code>INTEGER = -1</code>	The maximum number of milliseconds that the pool will wait (when there are no available connections) for a connection to be returned before throwing an exception, or -1 to wait indefinitely.
<b>minIdle</b> : <code>INTEGER = 0</code>	The minimum number of connections that can remain idle in the pool, without extra ones being created, or zero to create none.

## Hidden Properties

**auth** : *STRING = Container*

Auth

**createOrder** : *INTEGER = 60*

Create Order

**createVerb** : *STRING = Create*

Create Verb

**destroyOrder** : *INTEGER = 40*

Destroy Order

**destroyVerb** : *STRING = Destroy*

Destroy Verb

**elementName** : *STRING = \${deployed.jndiName}*

Element Name

**elementPropertyMapping** : *MAP\_STRING\_STRING = {maxIdle=, connectionProperties=, driverClassName=, password=, url=, resourceType=type, maxActive=, username=, minIdle=, maxWait=, jndiName=name, initialSize=, auth=}*

Element Property Mapping

**elementTag** : *STRING = Resource*

Element Tag

**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

**resourceType** : *STRING = javax.sql.DataSource*

Resource Type

**targetDirectory** : *STRING = \${deployed.container.contextDirectory}*

Target Directory

**targetFile** : *STRING = \${deployed.context}.xml*

Target File

**template** : *STRING = tc/context/context-element.ftl*

Template

**createTargetDirectory** : *BOOLEAN = false*

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

**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.

## tomcat.DataSourceLink

**Hierarchy** [tomcat.ResourceLink](#) >> [tomcat.JndiContextElement](#) >> [tomcat.ContextElement](#) >>  
 generic.ProcessedTemplate >> generic.AbstractDeployedArtifact >> generic.AbstractDeployed >>  
 udm.BaseDeployed >> udm.BaseConfigurationItem

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

To create a datasoure link to a global JNDI resource. Doing a JNDI lookup on the link name will then return the linked global resource. Installed to a Tomcat Virtual Host or the Common Context

### Public Properties



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

The container on which this deployed runs.

**globalName** : **STRING**

The name of the linked global resource in the global JNDI context.

**jndiName** : **STRING**

The name of the resource link to be created, relative to the java:comp/env context.

**context** : **STRING**

The name of the context that this resource is associated with

**deployable** : **CI**<[udm.Deployable](#)>

The deployable that this deployed is derived from.

## Hidden Properties

**createOrder** : INTEGER = 60

Create Order

**createVerb** : STRING = *Create*

Create Verb

**destroyOrder** : INTEGER = 40

Destroy Order

**destroyVerb** : STRING = *Destroy*

Destroy Verb

**elementName** : STRING = *\${deployed.jndiName}*

Element Name

**elementPropertyMapping** : MAP\_STRING\_STRING = *{globalName=, jndiName=name, resourceType=type}*

Element Property Mapping

**elementTag** : STRING = *ResourceLink*

Element Tag

**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

**resourceType** : STRING = *javax.sql.DataSource*

Resource Type

**targetDirectory** : STRING = *\${deployed.container.contextDirectory}*

Target Directory

**targetFile** : STRING = *\${deployed.context}.xml*

Target File

**template** : STRING = *tc/context/context-element.ftl*

Template

**createTargetDirectory** : BOOLEAN = *false*

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

**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.

## tomcat.DataSourceLinkSpec

**Hierarchy** [jee.JndiResourceSpec](#) >> [jee.ResourceSpec](#) >> [udm.BaseDeployable](#) >> [udm.BaseConfigurationItem](#)

**Interfaces** [udm.Taggable](#), [udm.Deployable](#), [udm.ConfigurationItem](#)

To create a datasoure link to a global JNDI resource. Doing a JNDI lookup on the link name will then return the linked global resource. Installed to a Tomcat Virtual Host or the Common Context (deployable)

### Public Properties

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

The name of the context that this resource is associated with

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

The name of the linked global resource in the global JNDI context.

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

The name of the resource link to be created, relative to the java:comp/env context.

**tags** : [SET\\_OF\\_STRING](#)

The tags to map deployables to containers.

## tomcat.DataSourceSpec

**Hierarchy** [jee.DataSourceSpec](#) >> [jee.JndiResourceSpec](#) >> [jee.ResourceSpec](#) >> [udm.BaseDeployable](#) >> [udm.BaseConfigurationItem](#)

**Interfaces** [udm.Taggable](#), [udm.Deployable](#), [udm.ConfigurationItem](#)

DataSource installed to a Tomcat Virtual Host or the Common Context (deployable)

**Public Properties****connectionProperties** : `MAP_STRING_STRING`

The connection properties that will be sent to our JDBC driver when establishing new connections.

**context** : `STRING`

The name of the context that this resource is associated with

**driverClassName** : `STRING`

The fully qualified Java class name of the JDBC driver to be used.

**initialSize** : `STRING`

The initial number of connections that are created when the pool is started.

**jndiName** : `STRING`

Name used to lookup the resource in JNDI.

**maxActive** : `STRING`

The maximum number of active connections that can be allocated from this pool at the same time, or negative for no limit.

**maxIdle** : `STRING`

The maximum number of connections that can remain idle in the pool, without extra ones being released, or negative for no limit.

**maxWait** : `STRING`

The maximum number of milliseconds that the pool will wait (when there are no available connections) for a connection to be returned before throwing an exception, or -1 to wait indefinitely.

**minIdle** : `STRING`

The minimum number of connections that can remain idle in the pool, without extra ones being created, or zero to create none.

**password** : `STRING`

The connection password to be passed to our JDBC driver to establish a connection.

**tags** : `SET_OF_STRING`

The tags to map deployables to containers.

**url** : `STRING`

The connection URL to be passed to our JDBC driver to establish a connection.

**username** : `STRING`

The connection username to be passed to our JDBC driver to establish a connection.

**tomcat.JeeActiveMqConnectionFactory**

**Hierarchy** `tomcat.JndiContextElement` >> `tomcat.ContextElement` >> `generic.ProcessedTemplate` >>  
`generic.AbstractDeployedArtifact` >> `generic.AbstractDeployed` >> `udm.BaseDeployed` >>  
`udm.BaseConfigurationItem`

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

ActiveMq Connection Factory installed to a Tomcat Virtual Host or the Common Context

## Public Properties

**brokerName** : *STRING*

Broker Name

**brokerUrl** : *STRING*

Broker Url



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

The container on which this deployed runs.

**description** : *STRING = JMS Connection Factory*

Description

**jndiName** : *STRING*

Name used to lookup the resource in JNDI.

**context** : *STRING*

The name of the context that this resource is associated with

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

The deployable that this deployed is derived from.

## Hidden Properties

**auth** : *STRING* = *Container*

Auth

**createOrder** : *INTEGER* = *60*

Create Order

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

Create Verb

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

Destroy Order

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

Destroy Verb

**elementName** : *STRING* = *\${deployed.jndiName}*

Element Name

**elementPropertyMapping** : *MAP\_STRING\_STRING* = *{brokerUrl=brokerURL, description=, brokerName=, jndiName=name, factory=, auth=, resourceType=type}*

Element Property Mapping

**elementTag** : *STRING* = *Resource*

Element Tag

**factory** : *STRING* = *org.apache.activemq.jndi.JNDIReferenceFactory*

Factory

**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

**resourceType** : *STRING* = *org.apache.activemq.ActiveMQConnectionFactory*

Resource Type

**targetDirectory** : *STRING* = *\${deployed.container.contextDirectory}*

Target Directory

**targetFile** : *STRING* = *\${deployed.context}.xml*

Target File

**template** : *STRING* = *tc/context/context-element.ftl*

Template

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

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

**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.

## tomcat.JeeActiveMqQueue

**Hierarchy** `tomcat.JndiContextElement` >> `tomcat.ContextElement` >> `generic.ProcessedTemplate` >>  
`generic.AbstractDeployedArtifact` >> `generic.AbstractDeployed` >> `udm.BaseDeployed` >>  
`udm.BaseConfigurationItem`

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

ActiveMq queue installed to a Tomcat Virtual Host or the Common Context

### Public Properties

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

The container on which this deployed runs.

**description** : **STRING**

Description

**jndiName** : **STRING**

Name used to lookup the resource in JNDI.

**physicalName** : **STRING**

Physical Name

**context** : **STRING**

The name of the context that this resource is associated with

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

The deployable that this deployed is derived from.

## Hidden Properties

**auth** : *STRING* = *Container*

Auth

**createOrder** : *INTEGER* = *60*

Create Order

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

Create Verb

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

Destroy Order

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

Destroy Verb

**elementName** : *STRING* = *\${deployed.jndiName}*

Element Name

**elementPropertyMapping** : *MAP\_STRING\_STRING* = *{description=, jndiName=name, physicalName=, factory=, auth=, resourceType=type}*

Element Property Mapping

**elementTag** : *STRING* = *Resource*

Element Tag

**factory** : *STRING* = *org.apache.activemq.jndi.JNDIReferenceFactory*

Factory

**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

**resourceType** : *STRING* = *org.apache.activemq.command.ActiveMQQueue*

Resource Type

**targetDirectory** : *STRING* = *\${deployed.container.contextDirectory}*

Target Directory

**targetFile** : *STRING* = *\${deployed.context}.xml*

Target File

**template** : *STRING* = *tc/context/context-element.ftl*

Template

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

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

**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.

## tomcat.JeeActiveMqTopic

**Hierarchy** `tomcat.JndiContextElement` >> `tomcat.ContextElement` >> `generic.ProcessedTemplate` >>  
`generic.AbstractDeployedArtifact` >> `generic.AbstractDeployed` >> `udm.BaseDeployed` >>  
`udm.BaseConfigurationItem`

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

ActiveMq topic installed to a Tomcat Virtual Host or the Common Context

### Public Properties

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

The container on which this deployed runs.

**description** : **STRING**

Description

**jndiName** : **STRING**

Name used to lookup the resource in JNDI.

**physicalName** : **STRING**

Physical Name

**context** : **STRING**

The name of the context that this resource is associated with

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

The deployable that this deployed is derived from.

## Hidden Properties

**auth** : *STRING = Container*

Auth

**createOrder** : *INTEGER = 60*

Create Order

**createVerb** : *STRING = Create*

Create Verb

**destroyOrder** : *INTEGER = 40*

Destroy Order

**destroyVerb** : *STRING = Destroy*

Destroy Verb

**elementName** : *STRING = \${deployed.jndiName}*

Element Name

**elementPropertyMapping** : *MAP\_STRING\_STRING = {description=, jndiName=name, physicalName=, factory=, auth=, resourceType=type}*

Element Property Mapping

**elementTag** : *STRING = Resource*

Element Tag

**factory** : *STRING = org.apache.activemq.jndi.JNDIReferenceFactory*

Factory

**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

**resourceType** : *STRING = org.apache.activemq.command.ActiveMQTopic*

Resource Type

**targetDirectory** : *STRING = \${deployed.container.contextDirectory}*

Target Directory

**targetFile** : *STRING = \${deployed.context}.xml*

Target File

**template** : *STRING = tc/context/context-element.ftl*

Template

**createTargetDirectory** : *BOOLEAN = false*

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



**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.

## tomcat.JeeWebSphereMqQueue

**Hierarchy** `tomcat.JndiContextElement` >> `tomcat.ContextElement` >> `generic.ProcessedTemplate` >>  
`generic.AbstractDeployedArtifact` >> `generic.AbstractDeployed` >> `udm.BaseDeployed` >>  
`udm.BaseConfigurationItem`

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

WebSphereMq queue installed to a Tomcat Virtual Host or the Common Context

### Public Properties

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

The container on which this deployed runs.

**description** : **STRING**

Description

**jndiName** : **STRING**

Name used to lookup the resource in JNDI.

**physicalName** : **STRING**

Physical Name

**context** : **STRING**

The name of the context that this resource is associated with

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

The deployable that this deployed is derived from.

## Hidden Properties

**auth** : *STRING* = *Container*

Auth

**createOrder** : *INTEGER* = *60*

Create Order

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

Create Verb

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

Destroy Order

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

Destroy Verb

**elementName** : *STRING* = *\${deployed.jndiName}*

Element Name

**elementPropertyMapping** : *MAP\_STRING\_STRING* = *{description=, jndiName=name, physicalName=QU, factory=, auth=, resourceType=type}*

Element Property Mapping

**elementTag** : *STRING* = *Resource*

Element Tag

**factory** : *STRING* = *com.ibm.mq.jms.MQQueueFactory*

Factory

**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

**resourceType** : *STRING* = *com.ibm.mq.jms.MQQueue*

Resource Type

**targetDirectory** : *STRING* = *\${deployed.container.contextDirectory}*

Target Directory

**targetFile** : *STRING* = *\${deployed.context}.xml*

Target File

**template** : *STRING* = *tc/context/context-element.ftl*

Template

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

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

**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.

## tomcat.JeeWebSphereMqQueueConnectionFactory

**Hierarchy** `tomcat.JndiContextElement` >> `tomcat.ContextElement` >> `generic.ProcessedTemplate` >>  
`generic.AbstractDeployedArtifact` >> `generic.AbstractDeployed` >> `udm.BaseDeployed` >>  
`udm.BaseConfigurationItem`

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

WebSphereMq Queue Connection Factory installed to a Tomcat Virtual Host or the Common Context

### Public Properties

**channel** : **STRING**

Channel



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

The container on which this deployed runs.

**description** : **STRING** = *JMS Connection Factory*

Description

**jndiName** : **STRING**

Name used to lookup the resource in JNDI.

**port** : **INTEGER**

Port

**queueManager** : **STRING**

Queue Manager

**server** : **STRING**

Server

**context** : **STRING**

The name of the context that this resource is associated with

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

The deployable that this deployed is derived from.

## Hidden Properties

**auth** : *STRING = Container*

Auth

**createOrder** : *INTEGER = 60*

Create Order

**createVerb** : *STRING = Create*

Create Verb

**destroyOrder** : *INTEGER = 40*

Destroy Order

**destroyVerb** : *STRING = Destroy*

Destroy Verb

**elementName** : *STRING = \${deployed.jndiName}*

Element Name

**elementPropertyMapping** : *MAP\_STRING\_STRING = {port=PORT, transport=TRAN, description=, queueManager=QMGR, jndiName=name, server=HOST, factory=, channel=CHAN, auth=, resourceType=type}*

Element Property Mapping

**elementTag** : *STRING = Resource*

Element Tag

**factory** : *STRING = com.ibm.mq.jms.MQQueueConnectionFactoryFactory*

Factory

**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

**resourceType** : *STRING = com.ibm.mq.jms.MQQueueConnectionFactory*

Resource Type

**targetDirectory** : *STRING = \${deployed.container.contextDirectory}*

Target Directory

**targetFile** : *STRING = \${deployed.context}.xml*

Target File

**template** : *STRING = tc/context/context-element.ftl*

Template

**transport** : *STRING = 1*

Transport

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

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

**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.


## tomcat.JeeWebsphereMqTopic

**Hierarchy** [tomcat.JndiContextElement](#) >> [tomcat.ContextElement](#) >> generic.ProcessedTemplate >>  
generic.AbstractDeployedArtifact >> generic.AbstractDeployed >> udm.BaseDeployed >>  
udm.BaseConfigurationItem

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

WebsphereMq topic installed to a Tomcat Virtual Host or the Common Context

### Public Properties

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

The container on which this deployed runs.

**description** : **STRING**

Description

**jndiName** : **STRING**

Name used to lookup the resource in JNDI.

**physicalName** : **STRING**

Physical Name

**context** : **STRING**

The name of the context that this resource is associated with

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

The deployable that this deployed is derived from.

## Hidden Properties

**auth** : *STRING* = *Container*

Auth

**createOrder** : *INTEGER* = *60*

Create Order

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

Create Verb

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

Destroy Order

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

Destroy Verb

**elementName** : *STRING* = *\${deployed.jndiName}*

Element Name

**elementPropertyMapping** : *MAP\_STRING\_STRING* = *{description=, jndiName=name, physicalName=TOP, factory=, auth=, resourceType=type}*

Element Property Mapping

**elementTag** : *STRING* = *Resource*

Element Tag

**factory** : *STRING* = *com.ibm.mq.jms.MQTopicFactory*

Factory

**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

**resourceType** : *STRING* = *com.ibm.mq.jms.MQTopic*

Resource Type

**targetDirectory** : *STRING* = *\${deployed.container.contextDirectory}*

Target Directory

**targetFile** : *STRING* = *\${deployed.context}.xml*

Target File

**template** : *STRING* = *tc/context/context-element.ftl*

Template

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

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

**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.

## tomcat.JeeWebSphereMqTopicConnectionFactory

**Hierarchy** `tomcat.JndiContextElement` >> `tomcat.ContextElement` >> `generic.ProcessedTemplate` >>  
`generic.AbstractDeployedArtifact` >> `generic.AbstractDeployed` >> `udm.BaseDeployed` >>  
`udm.BaseConfigurationItem`

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

WebSphereMq Topic Connection Factory installed to a Tomcat Virtual Host or the Common Context

### Public Properties

**channel** : **STRING**

Channel



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

The container on which this deployed runs.

**description** : **STRING** = *JMS Connection Factory*

Description

**jndiName** : **STRING**

Name used to lookup the resource in JNDI.

**port** : **INTEGER**

Port

**queueManager** : **STRING**

Queue Manager

**server** : **STRING**

Server

**context** : **STRING**

The name of the context that this resource is associated with

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

The deployable that this deployed is derived from.

## Hidden Properties

**auth** : *STRING = Container*

Auth

**createOrder** : *INTEGER = 60*

Create Order

**createVerb** : *STRING = Create*

Create Verb

**destroyOrder** : *INTEGER = 40*

Destroy Order

**destroyVerb** : *STRING = Destroy*

Destroy Verb

**elementName** : *STRING = \${deployed.jndiName}*

Element Name

**elementPropertyMapping** : *MAP\_STRING\_STRING = {port=PORT, transport=TRAN, description=, queueManager=QMGR, jndiName=name, server=HOST, factory=, channel=CHAN, auth=, resourceType=type}*

Element Property Mapping

**elementTag** : *STRING = Resource*

Element Tag

**factory** : *STRING = com.ibm.mq.jms.MQTopicConnectionFactoryFactory*

Factory

**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

**resourceType** : *STRING = com.ibm.mq.jms.MQTopicConnectionFactory*

Resource Type

**targetDirectory** : *STRING = \${deployed.container.contextDirectory}*

Target Directory

**targetFile** : *STRING = \${deployed.context}.xml*

Target File

**template** : *STRING = tc/context/context-element.ftl*

Template

**transport** : *STRING = 1*

Transport



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

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

**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.

## tomcat.JndiContextElement

**Hierarchy** [tomcat.ContextElement](#) >> generic.ProcessedTemplate >> generic.AbstractDeployedArtifact >> generic.AbstractDeployed >> udm.BaseDeployed >> udm.BaseConfigurationItem

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

Resource installed to a Tomcat Virtual Host or the Common Context

### Public Properties



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

The container on which this deployed runs.

**jndiName** : **STRING**

Name used to lookup the resource in JNDI.

**context** : **STRING**

The name of the context that this resource is associated with

**deployable** : **CI**<[udm.Deployable](#)>

The deployable that this deployed is derived from.

## Hidden Properties

**createOrder** : *INTEGER = 60*

Create Order

**createVerb** : *STRING = Create*

Create Verb

**destroyOrder** : *INTEGER = 40*

Destroy Order

**destroyVerb** : *STRING = Destroy*

Destroy Verb

**elementName** : *STRING = \${deployed.jndiName}*

Element Name

**elementTag** : *STRING = Resource*

Element Tag

**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.contextDirectory}*

Target Directory

**targetFile** : *STRING = \${deployed.context}.xml*

Target File

**template** : *STRING = tc/context/context-element.ftl*

Template

**createTargetDirectory** : *BOOLEAN = false*

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

**elementPropertyMapping** : *MAP\_STRING\_STRING*

Mapping of the CI properties to there factory equivalent. A blank value assumes the same name as the key.

**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.

## tomcat.MailSession

**Hierarchy** [tomcat.JndiContextElement](#) >> [tomcat.ContextElement](#) >> generic.ProcessedTemplate >> generic.AbstractDeployedArtifact >> generic.AbstractDeployed >> udm.BaseDeployed >> udm.BaseConfigurationItem

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

Mail Session installed to a Tomcat Virtual Host or the Common Context

### Public Properties



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

The container on which this deployed runs.

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

Name used to lookup the resource in JNDI.

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

Points at the server that provides SMTP service for your network.

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

The name of the context that this resource is associated with

**deployable** : [CI<udm.Deployable>](#)

The deployable that this deployed is derived from.

## Hidden Properties

**auth** : *STRING* = *Container*

Auth

**createOrder** : *INTEGER* = *60*

Create Order

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

Create Verb

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

Destroy Order

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

Destroy Verb

**elementName** : *STRING* = *\${deployed.jndiName}*

Element Name

**elementPropertyMapping** : *MAP\_STRING\_STRING* = *{smtpHost=mail.smtp.host, jndiName=name, auth=, resourceType=type}*

Element Property Mapping

**elementTag** : *STRING* = *Resource*

Element Tag

**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

**resourceType** : *STRING* = *javax.mail.Session*

Resource Type

**targetDirectory** : *STRING* = *\${deployed.container.contextDirectory}*

Target Directory

**targetFile** : *STRING* = *\${deployed.context}.xml*

Target File

**template** : *STRING* = *tc/context/context-element.ftl*

Template

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

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

**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.

## tomcat.MailSessionSpec

**Hierarchy** `jee.MailSessionSpec` >> `jee.JndiResourceSpec` >> `jee.ResourceSpec` >> `udm.BaseDeployable` >> `udm.BaseConfigurationItem`

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

Mail Session installed to a Tomcat Virtual Host or the Common Context (deployable)

### Public Properties

**context** : **STRING**

The name of the context that this resource is associated with

**jndiName** : **STRING**

Name used to lookup the resource in JNDI.

**smtpHost** : **STRING**

Points at the server that provides SMTP service for your network.

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

The tags to map deployables to containers.

## tomcat.ResourceLink

**Hierarchy** `tomcat.JndiContextElement` >> `tomcat.ContextElement` >> `generic.ProcessedTemplate` >> `generic.AbstractDeployedArtifact` >> `generic.AbstractDeployed` >> `udm.BaseDeployed` >> `udm.BaseConfigurationItem`

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

To create a link to a global JNDI resource. Doing a JNDI lookup on the link name will then return the linked global resource. Installed to a Tomcat Virtual Host or the Common Context

## Public Properties



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

The container on which this deployed runs.

**globalName** : `STRING`

The name of the linked global resource in the global JNDI context.

**jndiName** : `STRING`

The name of the resource link to be created, relative to the `java:comp/env` context.

**resourceType** : `STRING`

The fully qualified Java class name expected by the web application when it performs a lookup for this resource link.

**context** : `STRING`

The name of the context that this resource is associated with

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

The deployable that this deployed is derived from.

## Hidden Properties

**createOrder** : *INTEGER = 60*

Create Order

**createVerb** : *STRING = Create*

Create Verb

**destroyOrder** : *INTEGER = 40*

Destroy Order

**destroyVerb** : *STRING = Destroy*

Destroy Verb

**elementName** : *STRING = \${deployed.jndiName}*

Element Name

**elementPropertyMapping** : *MAP\_STRING\_STRING = {globalName=, jndiName=name, resourceType=type}*

Element Property Mapping

**elementTag** : *STRING = ResourceLink*

Element Tag

**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.contextDirectory}*

Target Directory

**targetFile** : *STRING = \${deployed.context}.xml*

Target File

**template** : *STRING = tc/context/context-element.ftl*

Template

**createTargetDirectory** : *BOOLEAN = false*

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

**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.

## tomcat.ResourceLinkSpec

**Hierarchy** [jee.JndiResourceSpec](#) >> [jee.ResourceSpec](#) >> [udm.BaseDeployable](#) >> [udm.BaseConfigurationItem](#)

**Interfaces** [udm.Taggable](#), [udm.Deployable](#), [udm.ConfigurationItem](#)

To create a link to a global JNDI resource. Doing a JNDI lookup on the link name will then return the linked global resource. Installed to a Tomcat Virtual Host or the Common Context (deployable)

### Public Properties

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

The name of the context that this resource is associated with

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

The name of the linked global resource in the global JNDI context.

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

The name of the resource link to be created, relative to the java:comp/env context.

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

The fully qualified Java class name expected by the web application when it performs a lookup for this resource link.

**tags** : [SET\\_OF\\_STRING](#)

The tags to map deployables to containers.

## tomcat.Server

**Hierarchy** [generic.Container](#) >> [udm.BaseContainer](#) >> [udm.BaseConfigurationItem](#)

**Interfaces** [udm.Taggable](#), [udm.ConfigurationItem](#), [generic.GenericContainer](#), [udm.Container](#), [overthere.HostContainer](#)

Tomcat server on host.



## Public Properties

**home** : *STRING*

Home directory for Tomcat



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

Host upon which the container resides

**startCommand** : *STRING*

OS specific command used to start Tomcat

**startWaitTime** : *INTEGER = 0*

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

**stopCommand** : *STRING*

OS specific command used to stop Tomcat

**stopWaitTime** : *INTEGER = 0*

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

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

Environment variables for container

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

The tags to map deployables to containers.

## Hidden Properties

**restartOrder** : *INTEGER = 90*

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

**serverXml** : *STRING = \${container.home}/conf/server.xml*

Server Xml

**startOrder** : *INTEGER = 90*

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

**startScript** : *STRING = tc/server/start-tc*

Start Script

**stopOrder** : *INTEGER = 10*

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

**stopScript** : *STRING = tc/server/stop-tc*

Stop Script

**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.

## Control Tasks

### start

Start the Tomcat.

### stop

Stop the Tomcat.

## tomcat.VirtualHost

**Hierarchy** `tomcat.ContextContainer` >> `generic.NestedContainer` >> `udm.BaseContainer` >> `udm.BaseConfigurationItem`

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

represents a virtual host, which is an association of a network name for a server (such as 'www.mycompany.com' with the particular server on which Catalina is running.

## Public Properties

**appBase** : `STRING` = *webapps*

The Application Base directory for this virtual host. This is the pathname of a directory that may contain web applications to be deployed on this virtual host. You may specify a pathname that is relative to the Tomcat Server home directory.

**hostName** : `STRING` = *localhost*

Host Name



**server** : `CI<tomcat.Server>`

Server

**envVars** : `MAP_STRING_STRING`

Environment variables for container

**tags** : `SET_OF_STRING`

The tags to map deployables to containers.

## Hidden Properties

**appBaseAbsolutePath** : `STRING` = *\${container.server.home}/\${container.appBase}*

App Base Absolute Path

**contextDirectory** : `STRING` = *\${container.server.home}/conf/Catalina/\${container.hostName}*

Context Directory

## tomcat.WarModule

**Hierarchy** `tomcat.ContextWarModule` >> `generic.CopiedArtifact` >> `generic.AbstractDeployedArtifact` >> `generic.AbstractDeployed` >> `udm.BaseDeployed` >> `udm.BaseConfigurationItem`

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

War installed to a Tomcat Virtual Host

## Public Properties



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

The container on which this deployed runs.

**contextRoot** : `STRING = ${deployed.name}`

Context root under which the web application can be access

**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

**contextElementPropertyMapping** : *MAP\_STRING\_STRING = {docBase=, unpackWAR=}*

Mapping of the Context attributes. A blank value assumes the same name as the key.

**contextSubject** : *STRING = war settings*

Context Subject

**contextTemplate** : *STRING = tc/context/context-attributes.ftl*

Context Template

**contextXmlTargetDirectory** : *STRING = \${deployed.container.contextDirectory}*

Context Xml Target Directory

**createOrder** : *INTEGER = 70*

Create Order

**createOrderOfContextXml** : *INTEGER = 60*

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

**createVerb** : *STRING = Create*

Create Verb

**destroyOrder** : *INTEGER = 30*

Destroy Order

**destroyOrderOfContextXml** : *INTEGER = 40*

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

**destroyVerb** : *STRING = Destroy*

Destroy Verb

**docBase** : *STRING = \${deployed.targetFile}*

Doc Base

**modifyOrder** : *INTEGER = 50*

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

**modifyOrderOfContextXml** : *INTEGER = 60*

The order of the rename context root 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.appBaseAbsolutePath}*

Target Directory

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

Target File

**createTargetDirectory** : *BOOLEAN = false*

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

**deleteGeneratedResources** : `SET_OF_STRING`

Absolute paths to files on the target system that was generated and must be clean up during a destroy operation

**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.

**unpackWAR** : `BOOLEAN = true`

Unpack WAR

## tomcat.WebsphereMqQueue

**Hierarchy** `tomcat.JeeWebsphereMqQueue >> tomcat.JndiContextElement >> tomcat.ContextElement >> generic.ProcessedTemplate >> generic.AbstractDeployedArtifact >> generic.AbstractDeployed >> udm.BaseDeployed >> udm.BaseConfigurationItem`

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

WebsphereMq queue installed to a Tomcat Virtual Host or the Common Context

### Public Properties

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

The container on which this deployed runs.

**description** : `STRING`

Description

**jndiName** : `STRING`

Name used to lookup the resource in JNDI.

**physicalName** : `STRING`

Physical Name

**context** : `STRING`

The name of the context that this resource is associated with

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

The deployable that this deployed is derived from.

## Hidden Properties

**auth** : *STRING = Container*

Auth

**createOrder** : *INTEGER = 60*

Create Order

**createVerb** : *STRING = Create*

Create Verb

**destroyOrder** : *INTEGER = 40*

Destroy Order

**destroyVerb** : *STRING = Destroy*

Destroy Verb

**elementName** : *STRING = \${deployed.jndiName}*

Element Name

**elementPropertyMapping** : *MAP\_STRING\_STRING = {description=, jndiName=name, physicalName=QU, factory=, auth=, resourceType=type}*

Element Property Mapping

**elementTag** : *STRING = Resource*

Element Tag

**factory** : *STRING = com.ibm.mq.jms.MQQueueFactory*

Factory

**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

**resourceType** : *STRING = com.ibm.mq.jms.MQQueue*

Resource Type

**targetDirectory** : *STRING = \${deployed.container.contextDirectory}*

Target Directory

**targetFile** : *STRING = \${deployed.context}.xml*

Target File

**template** : *STRING = tc/context/context-element.ftl*

Template

**createTargetDirectory** : *BOOLEAN = false*

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

**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.

## tomcat.WebsphereMqQueueConnectionFactory

**Hierarchy** [tomcat.JeeWebsphereMqQueueConnectionFactory](#) >> [tomcat.JndiContextElement](#) >> [tomcat.ContextElement](#) >> [generic.ProcessedTemplate](#) >> [generic.AbstractDeployedArtifact](#) >> [generic.AbstractDeployed](#) >> [udm.BaseDeployed](#) >> [udm.BaseConfigurationItem](#)

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

WebsphereMq Queue Connection Factory installed to a Tomcat Virtual Host or the Common Context

### Public Properties

**channel** : **STRING**

Channel



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

The container on which this deployed runs.

**description** : **STRING** = *JMS Connection Factory*

Description

**jndiName** : **STRING**

Name used to lookup the resource in JNDI.

**port** : **INTEGER**

Port

**queueManager** : **STRING**

Queue Manager

**server** : **STRING**

Server

**context** : **STRING**

The name of the context that this resource is associated with

**deployable** : **CI**<[udm.Deployable](#)>

The deployable that this deployed is derived from.

## Hidden Properties

**auth** : *STRING = Container*

Auth

**createOrder** : *INTEGER = 60*

Create Order

**createVerb** : *STRING = Create*

Create Verb

**destroyOrder** : *INTEGER = 40*

Destroy Order

**destroyVerb** : *STRING = Destroy*

Destroy Verb

**elementName** : *STRING = \${deployed.jndiName}*

Element Name

**elementPropertyMapping** : *MAP\_STRING\_STRING = {port=PORT, transport=TRAN, description=, queueManager=QMGR, jndiName=name, server=HOST, factory=, channel=CHAN, auth=, resourceType=type}*

Element Property Mapping

**elementTag** : *STRING = Resource*

Element Tag

**factory** : *STRING = com.ibm.mq.jms.MQQueueConnectionFactoryFactory*

Factory

**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

**resourceType** : *STRING = com.ibm.mq.jms.MQQueueConnectionFactory*

Resource Type

**targetDirectory** : *STRING = \${deployed.container.contextDirectory}*

Target Directory

**targetFile** : *STRING = \${deployed.context}.xml*

Target File

**template** : *STRING = tc/context/context-element.ftl*

Template

**transport** : *STRING = 1*

Transport



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

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

**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.

## tomcat.WebsphereMqQueueConnectionFactorySpec

**Hierarchy** [jee.QueueConnectionFactorySpec](#) >> [jee.JndiResourceSpec](#) >> [jee.ResourceSpec](#) >>  
[udm.BaseDeployable](#) >> [udm.BaseConfigurationItem](#)

**Interfaces** [udm.Taggable](#), [udm.Deployable](#), [udm.ConfigurationItem](#)

WebsphereMq Queue Connection Factory installed to a Tomcat Virtual Host or the Common Context (deployable)

### Public Properties

**channel** : **STRING**

Channel

**context** : **STRING**

The name of the context that this resource is associated with

**description** : **STRING**

Description

**jndiName** : **STRING**

Name used to lookup the resource in JNDI.

**port** : **STRING**

Port

**queueManager** : **STRING**

Queue Manager

**server** : **STRING**

Server

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

The tags to map deployables to containers.

## tomcat.WebsphereMqQueueSpec

**Hierarchy** [jee.QueueSpec](#) >> [jee.JndiResourceSpec](#) >> [jee.ResourceSpec](#) >> [udm.BaseDeployable](#) >>  
[udm.BaseConfigurationItem](#)

**Interfaces** [udm.Taggable](#), [udm.Deployable](#), [udm.ConfigurationItem](#)

## WebsphereMq queue installed to a Tomcat Virtual Host or the Common Context (deployable)

### Public Properties

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

The name of the context that this resource is associated with

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

Description

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

Name used to lookup the resource in JNDI.

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

Physical Name

**tags** : [SET\\_OF\\_STRING](#)

The tags to map deployables to containers.

## tomcat.WebsphereMqTopic

**Hierarchy** [tomcat.JeeWebsphereMqTopic](#) >> [tomcat.JndiContextElement](#) >> [tomcat.ContextElement](#) >> [generic.ProcessedTemplate](#) >> [generic.AbstractDeployedArtifact](#) >> [generic.AbstractDeployed](#) >> [udm.BaseDeployed](#) >> [udm.BaseConfigurationItem](#)

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

WebsphereMq topic installed to a Tomcat Virtual Host or the Common Context

### Public Properties



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

The container on which this deployed runs.

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

Description

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

Name used to lookup the resource in JNDI.

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

Physical Name

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

The name of the context that this resource is associated with

**deployable** : [CI<udm.Deployable>](#)

The deployable that this deployed is derived from.

## Hidden Properties

**auth** : *STRING = Container*

Auth

**createOrder** : *INTEGER = 60*

Create Order

**createVerb** : *STRING = Create*

Create Verb

**destroyOrder** : *INTEGER = 40*

Destroy Order

**destroyVerb** : *STRING = Destroy*

Destroy Verb

**elementName** : *STRING = \${deployed.jndiName}*

Element Name

**elementPropertyMapping** : *MAP\_STRING\_STRING = {description=, jndiName=name, physicalName=TOP, factory=, auth=, resourceType=type}*

Element Property Mapping

**elementTag** : *STRING = Resource*

Element Tag

**factory** : *STRING = com.ibm.mq.jms.MQTopicFactory*

Factory

**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

**resourceType** : *STRING = com.ibm.mq.jms.MQTopic*

Resource Type

**targetDirectory** : *STRING = \${deployed.container.contextDirectory}*

Target Directory

**targetFile** : *STRING = \${deployed.context}.xml*

Target File

**template** : *STRING = tc/context/context-element.ftl*

Template

**createTargetDirectory** : *BOOLEAN = false*

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

**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.

## tomcat.WebsphereMqTopicConnectionFactory

**Hierarchy** `tomcat.JeeWebsphereMqTopicConnectionFactory` >> `tomcat.JndiContextElement` >> `tomcat.ContextElement` >> `generic.ProcessedTemplate` >> `generic.AbstractDeployedArtifact` >> `generic.AbstractDeployed` >> `udm.BaseDeployed` >> `udm.BaseConfigurationItem`

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

WebsphereMq Topic Connection Factory installed to a Tomcat Virtual Host or the Common Context

### Public Properties

**channel** : **STRING**

Channel



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

The container on which this deployed runs.

**description** : **STRING** = *JMS Connection Factory*

Description

**jndiName** : **STRING**

Name used to lookup the resource in JNDI.

**port** : **INTEGER**

Port

**queueManager** : **STRING**

Queue Manager

**server** : **STRING**

Server

**context** : **STRING**

The name of the context that this resource is associated with

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

The deployable that this deployed is derived from.

## Hidden Properties

**auth** : *STRING = Container*

Auth

**createOrder** : *INTEGER = 60*

Create Order

**createVerb** : *STRING = Create*

Create Verb

**destroyOrder** : *INTEGER = 40*

Destroy Order

**destroyVerb** : *STRING = Destroy*

Destroy Verb

**elementName** : *STRING = \${deployed.jndiName}*

Element Name

**elementPropertyMapping** : *MAP\_STRING\_STRING = {port=PORT, transport=TRAN, description=, queueManager=QMGR, jndiName=name, server=HOST, factory=, channel=CHAN, auth=, resourceType=type}*

Element Property Mapping

**elementTag** : *STRING = Resource*

Element Tag

**factory** : *STRING = com.ibm.mq.jms.MQTopicConnectionFactoryFactory*

Factory

**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

**resourceType** : *STRING = com.ibm.mq.jms.MQTopicConnectionFactory*

Resource Type

**targetDirectory** : *STRING = \${deployed.container.contextDirectory}*

Target Directory

**targetFile** : *STRING = \${deployed.context}.xml*

Target File

**template** : *STRING = tc/context/context-element.ftl*

Template

**transport** : *STRING = 1*

Transport

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

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

**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.

## tomcat.WebsphereMqTopicConnectionFactorySpec

**Hierarchy** [jee.QueueConnectionFactorySpec](#) >> [jee.JndiResourceSpec](#) >> [jee.ResourceSpec](#) >>  
udm.BaseDeployable >> udm.BaseConfigurationItem

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

WebsphereMq Topic Connection Factory installed to a Tomcat Virtual Host or the Common Context (deployable)

### Public Properties

**channel** : **STRING**

Channel

**context** : **STRING**

The name of the context that this resource is associated with

**description** : **STRING**

Description

**jndiName** : **STRING**

Name used to lookup the resource in JNDI.

**port** : **STRING**

Port

**queueManager** : **STRING**

Queue Manager

**server** : **STRING**

Server

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

The tags to map deployables to containers.

## tomcat.WebsphereMqTopicSpec

**Hierarchy** [jee.TopicSpec](#) >> [jee.JndiResourceSpec](#) >> [jee.ResourceSpec](#) >> udm.BaseDeployable >>  
udm.BaseConfigurationItem

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

---

**WebsphereMq topic installed to a Tomcat Virtual Host or the Common Context (deployable)****Public Properties****context** : **STRING**

The name of the context that this resource is associated with

**description** : **STRING**

Description

**jndiName** : **STRING**

Name used to lookup the resource in JNDI.

**physicalName** : **STRING**

Physical Name

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

The tags to map deployables to containers.

---