

Deployit IBM WebSphere MQ Plugin Manual

Version 3.8.0

Table of Contents

Table of Contents	2
Preface	3
Overview	3
Features	3
Requirements	3
Usage in Deployment Packages	3
Using the deployables and deployed	3
Deployable vs. Container table	3
Deployed Actions Table	4
Queue manager	4
Extension points	4
Extending the WMQ Plugin	4
CI Reference	5
Configuration Item Overview	5
Deployables	5
Deployeds	5
Containers	5
Other Configuration Items	5
Configuration Item Details	5
wmq.AliasQueue	5
wmq.AliasQueueSpec	7
wmq.LocalQueue	8
wmq.LocalQueueSpec	9
wmq.QueueManager	10
wmq.Resource	11

Preface

This document describes the functionality provided by the Websphere MQ plugin.

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

Overview

The WMQ plugin is a Deployit plugin that adds the capability to manage MQ resources on Websphere MQ environment. It works out of the box for deploying/ undeploying local queues and alias queues on a queue manager and can easily be extended to support management of other possible resources on a Websphere MQ environment.

Features

- Resources
 - Local Queue
 - Alias Queue
- Control Taks
 - Start/Stop queue manager

Requirements

- **Deployit requirements**
 - **Deployit:** version 3.6+
 - **Websphere MQ:** 7.x
 - **Other Deployit Plugins:** None
- **Infrastructural requirements**
 - **User credentials** for accessing the Host running Websphere MQ, and should be having the rights to run WMQ commands.

Usage in Deployment Packages

The plugin works with the standard deployment package 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 WMQ specific deployment package. It contain declarations for a [LocalQueue](#), and an [AliasQueue](#).

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

Name: testLocalQueue
CI-Type: wmq.LocalQueueSpec
CI-maxDepth: 3
CI-boqname: testBackoutQueue
CI-bothresh: 10

Name: testAliasQueue
CI-Type: wmq.AliasQueueSpec
CI-target: testTargetQueue
CI-cluster: QUEUE_CLUSTER
```

Using the deployables and deployed

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

Deployable vs. Container table

Deployable	Container	Generated deployed
wmq.LocalQueueSpec	wmq.QueueManager	wmq.LocalQueue

wmq.AliasQueueSpec	wmq.QueueManager	wmq.AliasQueue
--------------------	------------------	----------------

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

Deployed Actions Table

Deployed	Actions performed for operations		
	Create	Destroy	Modify
wmq.LocalQueue	<ul style="list-style-type: none"> create local queue on queue manager 	<ul style="list-style-type: none"> delete local queue 	<ul style="list-style-type: none"> alter local queue
wmq.AliasQueue	<ul style="list-style-type: none"> create queue alias on queue manager 	<ul style="list-style-type: none"> delete queue alias 	<ul style="list-style-type: none"> alter queue alias

Queue manager

wmq.QueueManager is a container type which represents an existing queue manager running in the MQ environment, and the MQ resources like local queue or alias queue can be targeted to it. It has a containment relationship with Host, which means that it can only be created under a Host. Also, since a `wmq.QueueManager` is meant to represent an existing queue manager, the name of the CI should reflect the existing queue manager name. For example if the existing queue manager running in the MQ environment is called VENUS, the `wmq.QueueManager` CI should be called VENUS.

wmq.QueueManager also supports control tasks for starting and stopping the queue manager. Please refer to the CLI or GUI manual to know more on how to use control tasks.

Extension points

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

The next section describes extensibility by examples:

Extending the WMQ Plugin

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

The following synthetic.xml snippet shows how the `maxDepth` property can be made hidden with a default value set on it in the `wmq.LocalQueue` type:

```
<type-modification type="wmq.LocalQueue">
  <!-- make it hidden, and give a default value if all the local queues are always created with the maxDepth
value of 3-->
  <property name="maxDepth" kind="integer" default="3" hidden="true"/>
</type-modification>
```

Adding a new property to a deployed/deployable

The following synthetic.xml snippet shows how a new property 'DEFPRTY'(for specifying the default priority) can be added to the `wmq.LocalQueue` type:

```
<type-modification type="wmq.LocalQueue">
  <!-- adding new property for setting the default priority-->
  <property name="defprty" kind="integer" default="3" label="default priority"
description="The default priority of messages put on the queue. The value must be in the range zero
(the lowest priority) through to the MAXPRTY queue manager parameter. (MAXPRTY is 9.)"/>
</type-modification>
```

Note that while adding a new property in the WMQ plugin, *the property name should match exactly with the the WMQ command parameter name*. Hence the property has been called `defprty` and not 'defaultProperty' or 'defPriority'. Also, a label can be specified for the property to give it a user-friendly name on the UI. In the example above, the property `defprty` will appear as `default priority` on the UI.

Adding a new type

New types can be added in WMQ plugin using the `Generic Plugin` patterns. For example, the following synthetic.xml snippet defines a new deployed type `wmq.ModelQueue` (and the corresponding deployable type `wmq.ModelQueueSpec`, which will be automatically generated from the deployed definition):

```
<type type="wmq.ModelQueue" extends="wmq.Resource" deployable-type="wmq.ModelQueueSpec"
      container-type="wmq.QueueManager">
  <generate-deployable type="wmq.ModelQueueSpec" extends="generic.Resource"/>
  <property name="createScript" hidden="true" default="wmq/create-qmodel" />
  <property name="modifyScript" hidden="true" default="wmq/modify-qmodel" />
  <property name="destroyScript" hidden="true" default="wmq/destroy-qmodel" />
  <property name="maxDepth" kind="integer" description="The maximum number of messages allowed on the queue"/>
</type>
```

Once this new type has been added in the `synthetic.xml`, the new types `wmq.ModelQueueSpec` and `wmq.ModelQueue` is readily available to the Deployit type system. But to make it usable completely, the corresponding scripts must be added at the specified path. This is how the create script `wmq/create-qmodel.sh` might look:

```
#!/bin/sh
echo "DEFINE QMODEL(${deployed.name}) ${deployed.parameters}"
    | ${deployed.container.executablesDirectory}/runmqsc ${deployed.container.name}
```

Similarly, below is an example destroy script `wmq/destroy-qmodel.sh` for the newly defined type:

```
#!/bin/sh
echo "DELETE QMODEL(${deployed.name})" | ${deployed.container.executablesDirectory}/runmqsc
${deployed.container.name}
```

On similar lines, the modify script can be specified containing the alter command for altering the model queue.

CI Reference

Configuration Item Overview

Deployables

CI	Description
wmq.AliasQueueSpec	Specification of a alias queue
wmq.LocalQueueSpec	Specification of a local queue

Deployeds

CI	Description
wmq.AliasQueue	An alias queue
wmq.LocalQueue	A local queue
wmq.Resource	Description unavailable

Containers

CI	Description
wmq.QueueManager	A queue manager manages the resources associated with it, in particular the queues that it owns

Other Configuration Items

CI	Description
wmq.AliasQueue	An alias queue
wmq.AliasQueueSpec	Specification of a alias queue
wmq.LocalQueue	A local queue
wmq.LocalQueueSpec	Specification of a local queue
wmq.QueueManager	A queue manager manages the resources associated with it, in particular the queues that it owns
wmq.Resource	Description unavailable



Configuration Item Details

wmq.AliasQueue

Type Hierarchy [wmq.Resource](#) >> generic.ExecutedScript >> generic.AbstractDeployed
>> udm.BaseDeployed >> udm.BaseConfigurationItem

Interfaces udm.Deployed, udm.ConfigurationItem

An alias queue

Parent	
 container : CI<udm.Container>	The container on which this deployed runs.
Public Properties	
 target : STRING	The name of the queue or topic object being aliased. The object can be a queue or a topic as defined by TARGTYPE. The maximum length is 48 characters.
cluster : STRING	The name of the cluster to which the queue belongs
deployable : CI<udm.Deployable>	The deployable that this deployed is derived from.

Hidden Properties	
* createOrder : INTEGER = 60	Create Order
* createScript : STRING = wmq/create-qalias	Create Script
* createVerb : STRING = Create	Create Verb
* destroyOrder : INTEGER = 40	Destroy Order
* destroyScript : STRING = wmq/destroy-qalias	Destroy Script
* destroyVerb : STRING = Destroy	Destroy Verb
* modifyOrder : INTEGER = 40	Modify Order
* modifyScript : STRING = wmq/modify-qalias	Modify Script
* 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
* standardPropertiesNotToExpose : STRING = deployable, container, createOrder, createScript, createVerb, modifyOrder, modifyScript, modifyVerb, destroyOrder, destroyScript, destroyVerb, startOrder, startScript, startVerb, stopOrder, stopScript, stopVerb, inspectScript, inspectVerb, securityPermissions, inheritPermissions	Standard properties that are not exposed to any python wsadmin script.
classpathResources : SET_OF_STRING	Additional classpath resources that should be uploaded to the working directory before executing the script.
inspectClasspathResources : SET_OF_STRING	Additional classpath resources that should be uploaded to the working directory before executing the inspect script.
inspectScript : STRING	Classpath to the script used to inspect the generic container.
inspectTemplateClasspathResources : SET_OF_STRING	Additional template classpath resources that should be uploaded to the working directory before executing the inspect script. The template is first rendered and the rendered content copied to a file, with the same name as the template, in the working directory.
noopScript : STRING	Classpath to the script that is uploaded and executed on the generic container for the noop operation.
remoteWorkingDirectoryPath : STRING	Name of working directory on target host. Default is to create a temporary directory which is deleted when connection is closed.
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.
retainRemoteWorkingDirectory : BOOLEAN = false	Retain the specified working directory on target host after completion.
templateClasspathResources : SET_OF_STRING	Additional template classpath resources that should be uploaded to the working directory before executing the script. The template is first rendered and the rendered content copied to a file, with the same name as the template, in the working directory.

wmq.AliasQueueSpec

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

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

Specification of a alias queue

Public Properties	
cluster :	STRING The name of the cluster to which the queue belongs (string)
tags :	SET_OF_STRING If set, this deployable will only be mapped automatically to containers with the same tag.
target :	STRING The name of the queue or topic object being aliased. The object can be a queue or a topic as defined by TARGTYPE. The maximum length is 48 characters. (string)

wmq.LocalQueue

Type Hierarchy [wmq.Resource](#) >> generic.ExecutedScript >> generic.AbstractDeployed
>> udm.BaseDeployed >> udm.BaseConfigurationItem

Interfaces udm.Deployed, udm.ConfigurationItem

A local queue

Parent	
* container :	CI<udm.Container> The container on which this deployed runs.
Public Properties	
* boqname :	STRING The excessive backout requeue name
* bothresh :	INTEGER The backout threshold
* maxDepth :	INTEGER The maximum number of messages allowed on the queue
cluster :	STRING The name of the cluster to which the queue belongs
deployable :	CI<udm.Deployable> The deployable that this deployed is derived from.

Hidden Properties	
* createOrder : INTEGER = 60	Create Order
* createScript : STRING = wmq/create-qlocal	Create Script
* createVerb : STRING = Create	Create Verb
* destroyOrder : INTEGER = 40	Destroy Order
* destroyScript : STRING = wmq/destroy-qlocal	Destroy Script
* destroyVerb : STRING = Destroy	Destroy Verb
* modifyOrder : INTEGER = 40	Modify Order
* modifyScript : STRING = wmq/modify-qlocal	Modify Script
* 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
* standardPropertiesNotToExpose : STRING = deployable, container, createOrder, createScript, createVerb, modifyOrder, modifyScript, modifyVerb, destroyOrder, destroyScript, destroyVerb, startOrder, startScript, startVerb, stopOrder, stopScript, stopVerb, inspectScript, inspectVerb, securityPermissions, inheritPermissions	Standard properties that are not exposed to any python wsadmin script.
classpathResources : SET_OF_STRING	Additional classpath resources that should be uploaded to the working directory before executing the script.
inspectClasspathResources : SET_OF_STRING	Additional classpath resources that should be uploaded to the working directory before executing the inspect script.
inspectScript : STRING	Classpath to the script used to inspect the generic container.
inspectTemplateClasspathResources : SET_OF_STRING	Additional template classpath resources that should be uploaded to the working directory before executing the inspect script. The template is first rendered and the rendered content copied to a file, with the same name as the template, in the working directory.
noopScript : STRING	Classpath to the script that is uploaded and executed on the generic container for the noop operation.
remoteWorkingDirectoryPath : STRING	Name of working directory on target host. Default is to create a temporary directory which is deleted when connection is closed.
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.
retainRemoteWorkingDirectory : BOOLEAN = false	Retain the specified working directory on target host after completion.
templateClasspathResources : SET_OF_STRING	Additional template classpath resources that should be uploaded to the working directory before executing the script. The template is first rendered and the rendered content copied to a file, with the same name as the template, in the working directory.

wmq.LocalQueueSpec

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

Specification of a local queue

Public Properties	
boqname :	STRING The excessive backout requeue name (string)
bothresh :	STRING The backout threshold (integer)
cluster :	STRING The name of the cluster to which the queue belongs (string)
maxDepth :	STRING The maximum number of messages allowed on the queue (integer)
tags :	SET_OF_STRING If set, this deployable will only be mapped automatically to containers with the same tag.

wmq.QueueManager

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

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

A queue manager manages the resources associated with it, in particular the queues that it owns. It provides queuing services to applications for Message Queuing Interface (MQI) calls and commands to create, modify, display, and delete WebSphere MQ objects

Parent	
* host :	CI<overthere.Host> Host upon which the container resides
Public Properties	
* executablesDirectory :	STRING absolute path of the directory containing wmq commands like strmqm,runmqsc etc. e.g. /opt/mqm/bin
envVars :	MAP_STRING_STRING Environment variables for container
tags :	SET_OF_STRING If set, only deployables with the same tag will be automatically mapped to this container.

Hidden Properties		
* restartOrder	: INTEGER = 90	The order of the restart container step in the step list.
* startOrder	: INTEGER = 90	The order of the start container step in the step list.
* startScript	: STRING = wmq/qmgr/start	The command to start queue manager. Arguments containing spaces are not supported
* startWaitTime	: INTEGER = 0	The time to wait in seconds for a container start action.
* stopOrder	: INTEGER = 10	The order of the stop container step in the step list.
* stopScript	: STRING = wmq/qmgr/stop	The command to stop queue manager. Arguments containing spaces are not supported
* stopWaitTime	: INTEGER = 0	The time to wait in seconds for a container stop action.
inspectClasspathResources : SET_OF_STRING		
Additional classpath resources that should be uploaded to the working directory before executing the inspect script.		
inspectScript : STRING		
Classpath to the script used to inspect the generic container.		
inspectTemplateClasspathResources : SET_OF_STRING		
Additional template classpath resources that should be uploaded to the working directory before executing the inspect script. The template is first rendered and the rendered content copied to a file, with the same name as the template, in the working directory.		
restartScript : STRING		
Classpath to the script used to restart the generic container.		
restartWaitTime : INTEGER = 0		
The time to wait in seconds for a container restart action.		
Control Tasks		
	start	Start queue manager
	stop	Stop queue manager

wmq.Resource

Virtual Type

Type Hierarchy generic.ExecutedScript >> generic.AbstractDeployed >> udm.BaseDeployed >> udm.BaseConfigurationItem

Interfaces udm.Deployed, udm.ConfigurationItem

Description unavailable

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

Hidden Properties	
* createOrder : INTEGER = 60	Create Order
* createScript : STRING	Classpath to the script that is uploaded and executed on the generic container for the create operation.
* createVerb : STRING = Create	Create Verb
* destroyOrder : INTEGER = 40	Destroy Order
* destroyVerb : STRING = Destroy	Destroy Verb
* modifyOrder : INTEGER = 40	Modify Order
* 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
* standardPropertiesNotToExpose : STRING = deployable, container, createOrder, createScript, createVerb, modifyOrder, modifyScript, modifyVerb, destroyOrder, destroyScript, destroyVerb, startOrder, startScript, startVerb, stopOrder, stopScript, stopVerb, inspectScript, inspectVerb, securityPermissions, inheritPermissions	Standard properties that are not exposed to any python wsadmin script.
classpathResources : SET_OF_STRING	Additional classpath resources that should be uploaded to the working directory before executing the script.
destroyScript : STRING	Classpath to the script that is uploaded and executed on the generic container for the destroy operation.
inspectClasspathResources : SET_OF_STRING	Additional classpath resources that should be uploaded to the working directory before executing the inspect script.
inspectScript : STRING	Classpath to the script used to inspect the generic container.
inspectTemplateClasspathResources : SET_OF_STRING	Additional template classpath resources that should be uploaded to the working directory before executing the inspect script. The template is first rendered and the rendered content copied to a file, with the same name as the template, in the working directory.
modifyScript : STRING	Classpath to the script that is uploaded and executed on the generic container for the modify operation.
noopScript : STRING	Classpath to the script that is uploaded and executed on the generic container for the noop operation.
remoteWorkingDirectoryPath : STRING	Name of working directory on target host. Default is to create a temporary directory which is deleted when connection is closed.
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.
retainRemoteWorkingDirectory : BOOLEAN = false	Retain the specified working directory on target host after completion.
templateClasspathResources : SET_OF_STRING	Additional template classpath resources that should be uploaded to the working directory before executing the script. The template is first rendered and the rendered content copied to a file, with the same name as the template, in the working directory.