

Deployit Test Plugin Manual

Version 3.5

Table of Content

Preface	3
Overview	3
Features	3
Requirements	3
Usage in Deployment Packages	3
Using the deployables and deployed	4
Deployable vs. Container table	4
Deployed Actions Table	4
Test Station	4
Using a HttpRequestTester	4
CI Reference	4
Configuration Item Overview	4
Deployable Configuration Items	5
Deployed Configuration Items	5
Topology Configuration Items	5
Virtual Deployed Configuration Items	5
Configuration Item Details	5
tests.BaseHttpRequestTester	5
tests.HttpRequestTest	6
tests.HttpRequestTester	7
tests.TestStation	9

Preface

This document describes the functionality provided by the Test plugin.

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

Overview

The tests-plugin is a Deployit plugin that supports execution of post-deployment application tests. These post deployment tests may includes tests such as testing whether the deployed application is accessible or whether the created datasource is functioning properly or not. The tests-plugin is intended to provide such functionality to the Deployit server.

Features

- Ability to check whether the deployed application is accessible through a URL

Requirements

- **Deployit requirements**
 - **Deployit:** version 3.5+
 - **Other Deployit Plugins:** None
- **Infrastructural requirements**
 - **User credentials** for accessing the Host running the JBoss application Server.
 - **User credentials** for accessing the Hosts on which the application accessibility test has to be performed (called the TestStation)
 - **wget** installed on the Hosts (Unix/Windows) on which the application accessibility test has to be performed if it's not same as the Deployit server Host.

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 MANIFEST.MF file snippet that shows how a tester can be included in the deployment package which can be used to run a sanity check to verify that the petclinic application is accessible after the deployment

```
Manifest-Version: 1.0
Deployit-Package-Format-Version: 1.3
CI-Application: PetClinic-ear
CI-Version: 2.0

Name: PetClinic-2.0.ear
CI-Type: jee.Ear
CI-Name: PetClinic

Name: PetclinicTest
CI-Type: test.HttpRequestTest
CI-url: http://jboss-51:8080/petclinic
CI-expectedResponseText: Display all veterinarians
CI-startDelay: 10
CI-noOfRetries: 5
CI-retryWaitInterval: 3
```

Using the deployables and deployed

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

Deployable vs. Container table

Deployable	Container	Generated deployed
test.HttpRequestTest	tests.TestStation	test.HttpRequestTester

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

Deployed Actions Table

Deployed	Actions performed for operations		
	Create	Destroy	Modify
test.HttpRequestTester	<ul style="list-style-type: none"> invoke the application URL to check if application is running 	<ul style="list-style-type: none"> no action 	<ul style="list-style-type: none"> invoke the application URL to check if application is running

Test Station

A test station is a generic.Container type on which a particular test is run. For the httpRequestTest, the type of Host used for creating the test station determines which version of the test should be run. There are two kind of tests possible for http request test.

- **Local test:** If the http test has be performed from the same Host on which the Deployit server is running, use a test station built on overthere.LocalHost. This will use Java to access the URL, means it doesn't require any agent to be installed on the machine.
- **Remote test:** If the http test has to be performed from a remote host (different from Deployit server Host), use a test station built on a Host other than overthere.LocalHost. This expects `wget` to be present on the Host since the script uses `wget` utility to invoke the URL.

Using a HttpRequestTester

Let's say if petclinic application is deployed on the server *production-host*, and the http request test has to be tested from host *test-host* (that has access to the petclinic application), then the following steps can be followed to add this application testing feature as part of the standard deployment:

- Create a host corresponding to test-host in the repository browser (ID: Infrastructure/test-host)
- Create a test station (let's say testStation) under the previously created testHost (ID: Infrastructure/test-host/testStation)
- Add the testStation in the environment to which deployment will be performed
- If the application test (let's say petclinicTest of type test.HttpRequestTest) was not part of the package which was imported for the application version, create it under the application version (so the ID of the test will look something like Applications/PetClinic-ear/1.0/PetclinicTest)
- Deploy the package to the enviromment, and you should see a extra step for testing the application at the bottom of the step list.

CI Reference

Configuration Item Overview

Deployable Configuration Items

CI	Description
tests.HttpRequestTest	The http request test that has to be performed

Deployed Configuration Items

CI	Description
tests.HttpRequestTester	The http request tester

Topology Configuration Items

CI	Description
tests.TestStation	Container from where the http request test is to be performed

Virtual Deployed Configuration Items

CI	Description
tests.BaseHttpRequestTester	Base class for the http request testers

Configuration Item Details

[tests.BaseHttpRequestTester](#)

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

Interfaces udm.Deployed, udm.ConfigurationItem

Base class for the http request testers

Public Properties
container * : CI < udm.Container >
The container on which this deployed runs.
expectedResponseText * : STRING
Text that is expected to be contained in the HTTP response body, if the response code is in the 200 range. A non-2xx response code will cause the test to fail irrespective of the response body
url * : STRING
The URL to test
deployable : CI < udm.Deployable >
The deployable that this deployed is derived from.
showPageInConsole : BOOLEAN
Show the page retrieved from the url

Hidden Properties

createOrder : INTEGER = 50

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

createScript : STRING

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

createVerb : STRING = *Create*

Create Verb

destroyOrder : INTEGER = 40

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

destroyVerb : STRING = *Destroy*

Destroy Verb

modifyOrder : INTEGER = 50

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

modifyVerb : STRING = *Modify*

Modify Verb

noopOrder : INTEGER = 50

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

noopVerb : STRING = *Modify*

Noop Verb

destroyScript : STRING

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

modifyScript : STRING

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

noOfRetries : INTEGER

Number of times to attempt executing the step, incase it fails

noopScript : STRING

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

retryWaitInterval : INTEGER

Time in seconds to wait before next retry

startDelay : INTEGER

Time in seconds to wait before starting the execution of step

tests.HttpRequestTest

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

Interfaces udm.Deployable, udm.ConfigurationItem

The http request test that has to be performed

Public Properties	
expectedResponseText :	STRING
Text that is expected to be contained in the HTTP response body, if the response code is in the 200 range. A non-2xx response code will cause the test to fail irrespective of the response body	
ignoreCertificateWarnings :	STRING
If set, certificate warnings when making a connection will be ignored	
noOfRetries :	STRING
Number of times to attempt executing the step, incase it fails	
retryWaitInterval :	STRING
Time in seconds to wait before next retry	
showPageInConsole :	STRING
Show the page retrieved from the url	
startDelay :	STRING
Time in seconds to wait before starting the execution of step	
url :	STRING
The URL to test	

tests.HttpRequestTester

Hierarchy tests.BaseHttpRequestTester >> generic.ExecutedScript >> generic.AbstractDeployed >> udm.BaseDeployed >> udm.BaseConfigurationItem

Interfaces udm.Deployed, udm.ConfigurationItem

The http request tester

Public Properties

container* : `CI<udm.Container>`

The container on which this deployed runs.

expectedResponseText* : `STRING`

Text that is expected to be contained in the HTTP response body, if the response code is in the 200 range. A non-2xx response code will cause the test to fail irrespective of the response body

noOfRetries* : `INTEGER = 5`

Number of times to attempt executing the step, incase it fails

retryWaitInterval* : `INTEGER = 5`

Time in seconds to wait before next retry

startDelay* : `INTEGER = 5`

Time in seconds to wait before starting the execution of step

url* : `STRING`

The URL to test

deployable : `CI<udm.Deployable>`

The deployable that this deployed is derived from.

ignoreCertificateWarnings : `BOOLEAN`

If set, certificate warnings when making a connection will be ignored

showPageInConsole : `BOOLEAN = false`

Show the page retrieved from the url

Hidden Properties

createOrder : INTEGER = 100

Create Order

createScript : STRING = tests/execute-http-request

Create Script

createVerb : STRING = Run test

Create Verb

destroyOrder : INTEGER = 40

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

destroyScript : STRING = DUMMY_VALUE

Destroy Script

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

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.

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.

tests.TestStation

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

Interfaces udm.ConfigurationItem, udm.Container

Container from where the http request test is to be performed

Public Properties

host : CI<overthere.Host>

Host upon which the container resides

Hidden Properties

restartOrder : INTEGER = 90

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

restartWaitTime : INTEGER = 0

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

startOrder : INTEGER = 90

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

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.

stopWaitTime : INTEGER = 0

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

restartScript : STRING

Classpath to the script used to restart the generic container.

startScript : STRING

Classpath to the script used to start the generic container.

stopScript : STRING

Classpath to the script used to stop the generic container.