

TPTP 4.0

Running a JUnit Test Remotely

TPTP 4.0

Running a JUnit Test Remotely

RUNNING A JUNIT TEST REMOTELY	3
INTRODUCTION	3
<i>Prerequisites</i>	3
OVERVIEW	3
CONFIGURING THE ARTIFACT	3
<i>Overview Tab</i>	4
<i>Test Assets Tab</i>	4
<i>General Properties Tab</i>	5
CONFIGURING THE LOCATION	6
<i>Overview Tab</i>	6
<i>General Properties Tab</i>	7
DEPLOYMENT GROUND RULES	7
<i>General</i>	7
<i>Files to be deployed</i>	7
<i>ROOTDIR Considerations</i>	8
<i>CLASSPATH Considerations</i>	8
<i>Excluded Libraries</i>	8
<i>Agent Controller Considerations</i>	9
<i>OS Considerations</i>	9
File Path Naming Conventions	9
Absolute Paths	9

Running a JUnit Test Remotely

Introduction

This note covers the basic mechanisms used and the ground rules involved in running a JUnit test remotely.

Prerequisites

The note assumes that the reader knows how to create a deployment for a test suite. Unless you make use of the default deployment to run your test locally, you must create a deployment before you can run a test. If you do not know how to create a deployment you should refer to **TPTP Tester Guide** within the TPTP 4.0 online Help, section: *Creating a test deployment*.

In this note we also assume that a deployment (**testDeployment**) has been established which has an associated artifact and location pairing of the **testArtifact** artifact and the **testLocation** location.

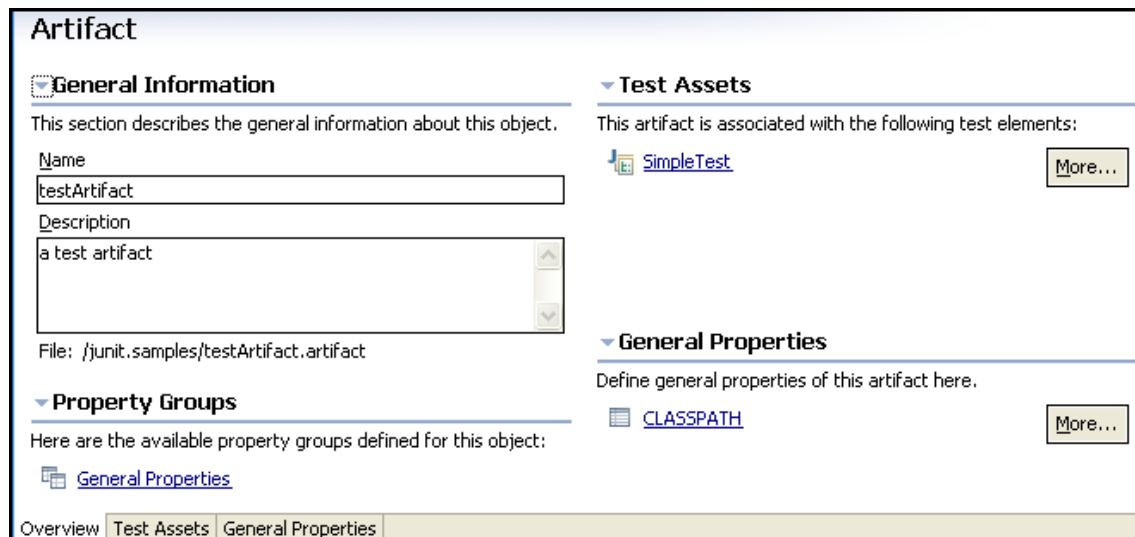
Overview

We look firstly at the mechanics of configuring **artifact** and **location** entities via their respective editors and explain the information and various options available.

We then look in some detail at the rules which apply in relation to these when the JUnit test is deployed remotely.

Configuring the Artifact

If we double-click on an artifact it opens within the Artifact editor:



There are 3 main tabs: **Overview**, **Test Assets**, and **General Properties**. The above screenshot shows the *Overview* tab information.

Overview Tab

The *General Information* section provides the artifact's name (testArtifact), Description ('a test artifact'), and its path within the workspace (/junit.samples/testArtifact.artifact).

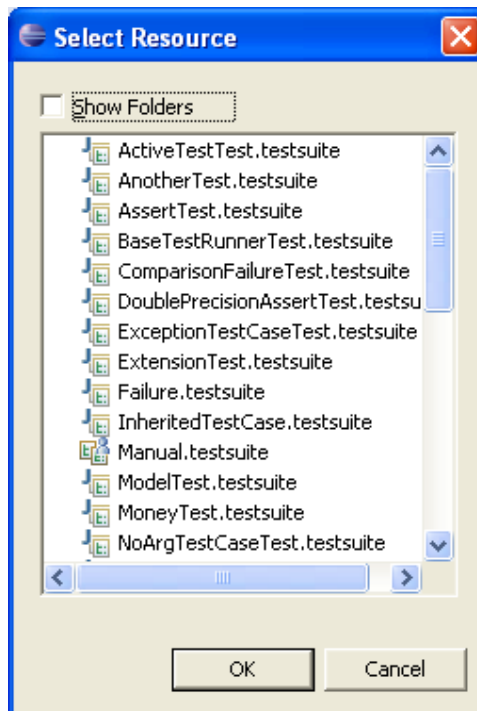
The *Test Assets* section shows the test elements associated with the artifact; in this case **SimpleTest**, which is a JUnit testsuite.

Test Assets Tab

In the Overview tab, clicking on either **SimpleTest**, the **More...** button, or selecting the **Test Assets** tab, results in the Test Assets view being displayed:



You can add or remove test assets by using the **Add...** or **Remove** buttons respectively. Invoking **Add...** brings up the Select Resource dialog:



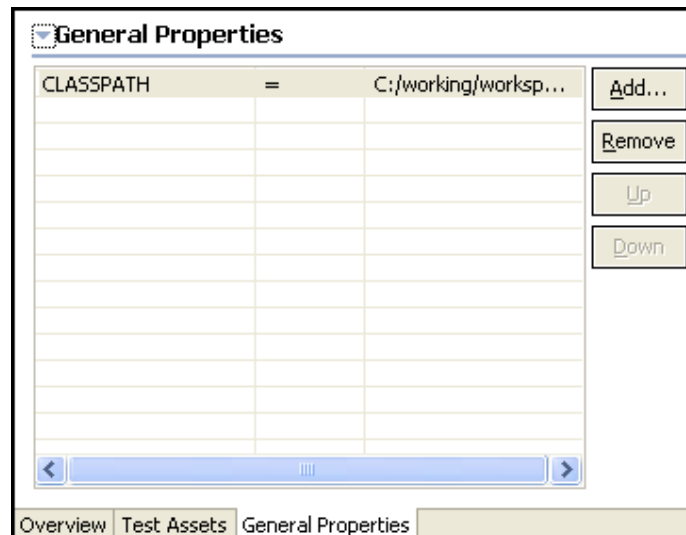
Simply select the resource required and click on **OK**.

To remove an asset, select it within the **Test Assets** tab view then click on **Remove**.

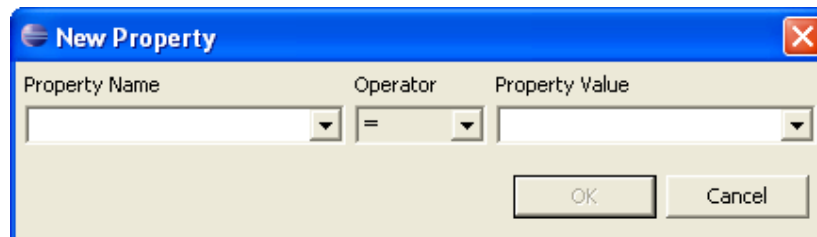
Apart from the JUnit testsuite, the test assets should include any other resources (e.g. datapools) required.

General Properties Tab

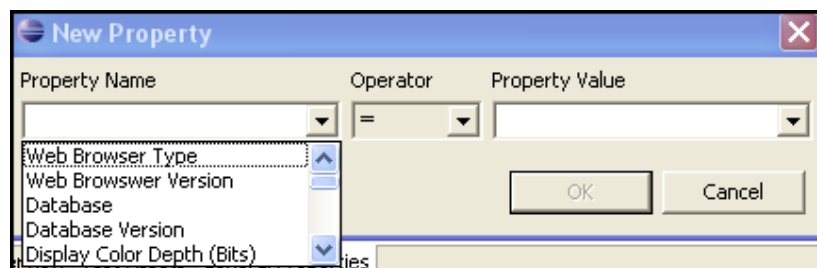
In the Overview tab, clicking on either **Classpath** or the **More...** button in the *General Properties* section of the Overview tab, selecting **GeneralProperties** in the *Property Groups* section, or selecting the **General Properties** tab, all result in the General Properties tab view being displayed:



To *add* a new property, click on the **Add...** button. This brings up the dialog:



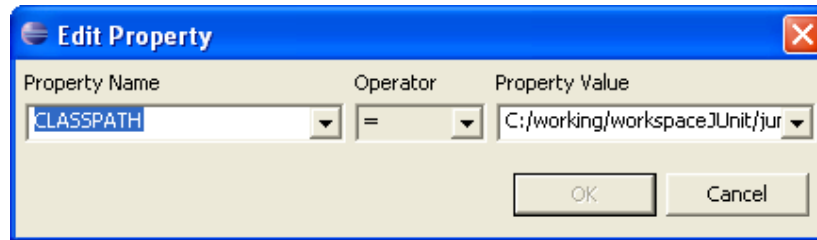
Enter your property name in the **Property Name** field, or select it from the drop-down list:



select the **Operator** from =, !=, >, <, >=, or <=, and insert the required value in the **Property Value** field.

(**Note** that TPTP itself only utilises the **Classpath** property).

To *edit* an existing property, double-click on it to bring up the Edit Property dialog:

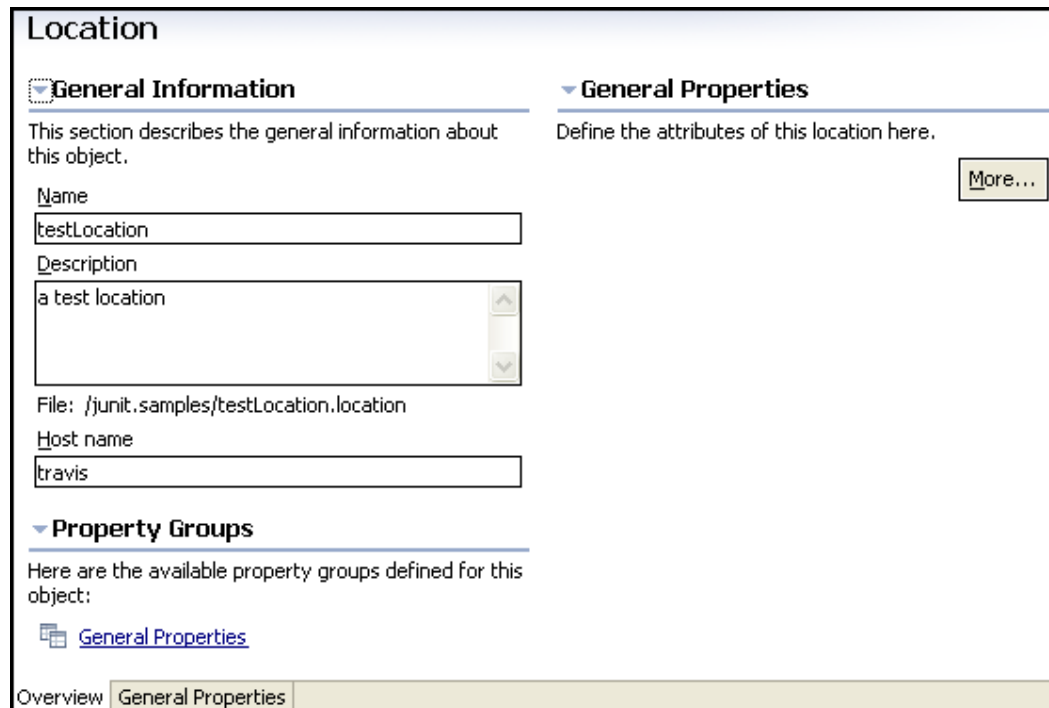


Then change the **Property Name**, **Operator**, or **Property Value** fields as required.

To *remove* an existing property, select it then click on the **Remove** button.

Configuring the Location

If we double-click on a location it opens within the Location editor:



There are 2 main tabs: **Overview** and **General Properties**. The above screenshot shows the *Overview* tab information.

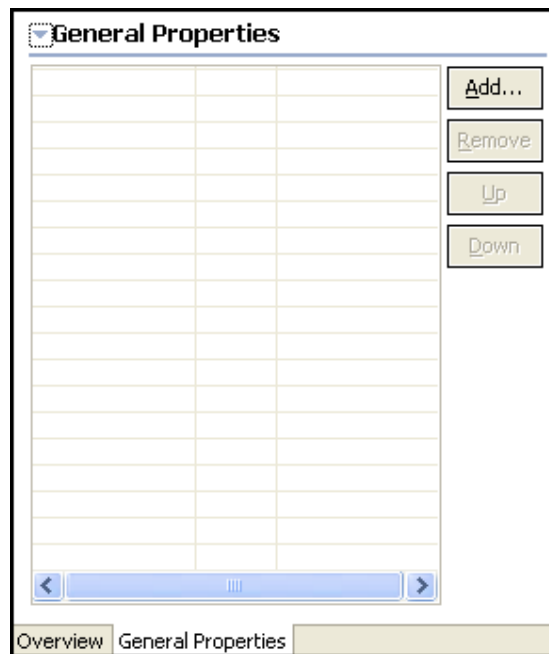
Overview Tab

The *General Information* section provides the location's name (testLocation here), Description ('a test location'), its path within the workspace (/junit.samples/testLocation.location), and the Host name (travis).

The default host name for the location is `localhost`. If the intention is to run on another computer, enter the name of that computer in the **Host name** field; so, in the above example, `travis` is identified as the other computer.

The *General Properties* and *Property Groups* sections both reference the **General Properties** tab. Clicking on the **More...** button in the *General Properties* section or selecting *GeneralProperties* in the *Property Groups* section selects the General Properties tab:

General Properties Tab



The functionality here is identical to that described above for the **General Properties** tab in the artifact editor.

Deployment Ground Rules

General

TPTP only concerns itself with the properties **CLASSPATH** and **ROOTDIR**.

Files to be deployed

The files to be deployed at execution time are:

1. Everything on the classpath of the test project *except* the following file types: **.java**, **.execution**, **.testsuite**, **.deploy**, **.location**, **.artifact**.
2. Any test assets associated with the Artifact being used for test execution
3. Property **Classpath** of the General Properties of the Artifact that is used for test execution.

ROOTDIR Considerations

The following procedure is used to calculate ROOTDIR:

Step 1	TPTP first looks on the remote machine for the ROOTDIR location property (if specified); if this exists, then its value is taken as the output from this step. If the ROOTDIR location property has not been specified, or if it has but does not exist on the remote machine, TPTP derives one using the temporary directory found on the remote machine's environment; it looks for LOCAL_AGENT_TEMP_DIR, TEMP, and TMP, in that order, and, whichever is found first is taken as the output from this step.
Step 2	TPTP looks for the USR_NAME location property on the local machine. If this does not exist it pulls the user.name property from Java (System.getProperty("user.name")).
Step 3	TPTP concatenates the outputs from Steps 1 and 2 and takes this to be the ROOTDIR to use.

CLASSPATH Considerations

To set up JARs that will be deployed and rooted under your ROOTDIR, set up the CLASSPATH parameter for your artifact; to set up JARs that do not need to be deployed (for instance, they already exist on the location/machine to be deployed to) but still need to be included in the classpath, then set up the CLASSPATH parameter in your location. So the distinction between artifact and location CLASSPATH is whether you want the JAR deployed and referenced or just referenced at the deployment location.

Excluded Libraries

Certain libraries are not deployed even if they are listed in the artifact's classpath property. They are:

commons-logging.jar
common-model.jar
common.runner.jar
datapool_api.jar
ecore.jar
ecore.xmi.jar
hcframe.jar
hexcore.jar
hexl.jar
hexr.jar
hexrecr.jar
hgla.jar
hglaconfig.jar
hmodel.jar
hparse.jar
hl14.jar
hlcommons.jar
hlcore.jar
hlevents.jar
hlcbe101.jar
http.hexrecr.jar

http.runner.jar
java.runner.jar
jmxagent.jar
junit.jar
manual.runner.jar
runtime.jar
xercesImpl.jar
xmlParserAPIs.jar

Agent Controller Considerations

Workbench clients before TPTP 3.3 cannot use agent controllers of TPTP 3.3 and beyond (they must continue to use older agent controllers, pre TPTP 3.3 agent controllers). Clients of TPTP 3.3 and beyond can use both old and new agent controllers. In summary, currently, the client/server support in TPTP is the following:

TPTP 3.3+ Clients

Clients \geq TPTP 3.3 and Server \geq TPTP 3.3 [use new file transfer protocol if unsecure AC, old file transfer protocol if secure AC]

Clients \geq TPTP 3.3 and Server $<$ TPTP 3.3 [use old file transfer protocol for unsecure and secure ACs]

TPTP Earlier Clients

Clients $<$ TPTP 3.3 and Server \geq TPTP 3.3 [this combination does not function due to older clients not understanding new server file transfer protocol]

Clients $<$ TPTP 3.3 and Server $<$ TPTP 3.3 [uses old file transfer protocol for unsecure and secure ACs]

There were file transfer service changes in TPTP 3.3 and TPTP 4.0 (to enhance the speed of deployment) that caused a new file transfer service protocol to be used, any clients before TPTP 3.3 do not know about this new protocol and therefore test deployment will fail. Newer clients know about the new and old protocols and therefore deployment will work.

OS Considerations

File Path Naming Conventions

Ensure that any file paths that you set in properties properly reflect the OS naming conventions for the target machine. For example, if you are deploying to a Linux machine from a Windows client, and you have specified ROOTDIR, say, check that its value is a legal path for Linux.

Absolute Paths

All paths *must* be absolute. For example, on Windows, "C:\myRootDir" is appropriate and on Linux "/home/user/myRootDir" is appropriate - all paths must be absolute and not relative.

V2: 09/12/2005