# Integration Testing

**Stale Content**

The content of this page is growing stale and may or may not contain relevant, useful or correct information.

## Objective

We have tests at the module level tests perform basic unit testing of the code in each module. We then have TCK tests that ensure Geronimo's compliance with the J2EE spec.  Now the integration tests introduces a level of testing that falls beyond the scope of the unit tests or the TCK tests. It performs system level tests of features and functionalities that Geronimo provides to the end users. Since Geronimo is an integration platform for the various open source technologies, this integration tests verifies and ensures such a succesful aggregation.

The integration test is designed to be run after the assemblies are built but as a part of the same build process. So a quick run through the integration tests will also ensure and certify a "really" successful build. The integration test can also run standalone against a previously assembled Geronimo archive (for now, *.zip only).

## Getting started

Checkout or update the trunk.
Build it top down (until we have all the 2.0-SNAPSHOT artifacts published)

**Creating an archetype (template) testsuite**

**command**

```
$>cd geronimo/testsuite
$>mvn archetype:create \
    -DarchetypeGroupId=org.apache.geronimo.plugins \
    -DarchetypeArtifactId=geronimo-archetype-testsuite \
    -DarchetypeVersion=2.0-SNAPSHOT \
    -DartifactId=newFunc-testsuite \
    -Dversion=2.0-SNAPSHOT
```

You should now have a template testsuite named *newFunc-testsuite* under your geronimo/testsuite directory.
This maven project is wholly self-sufficient. It starts and stops the server. It does not have any tests to run yet. It even has surefire reports for the server start/stop executions.

**Modifying the template to suit your function tests**

1. Change the *newFunc-testsuite*/*testset-1* directory name to reflect your testset name (say, *basic-testset*).
2. Edit the *newFunc-testsuite*/*basic-testset*/pom.xml and replace the <artifactId> element value to match the new directory name.
3. Populate *newFunc-testsuite*/*basic-testset* project with your tests. Any executions in the *basic-testset*/pom.xml will have to be inside a profile whose id is "child".

**NOTE** Steps 1 & 2 above can be skipped. But there is a possibility that this artifactId will conflict with another artifactId created similarly from archetype. Hence it is recommended.

**Running tests**

**command**

```
$>cd geronimo/testsuite
$>mvn [-DinstallDirectory=/path/to/geronimoHome]
```

Maven will run through all the testsuites. For every testsuite, it will invoke all the testsets under it. By the time it is done, it should have created surefire xmls in every target/surefire-reports directory.

Using the -DinstallDirectory parameter will install Geronimo in one location and the same server instance will be used for each *foo-testsuite*. The server will be started/stopped for each of them too. The default is for every *foo-testsuite* to have it's own installation of the server in it's target directory. All testsets under it will use this same instance of the running server.

You may go to any single testsuite and execute 'mvn' from it to run only it's tests.

**Viewing test results**

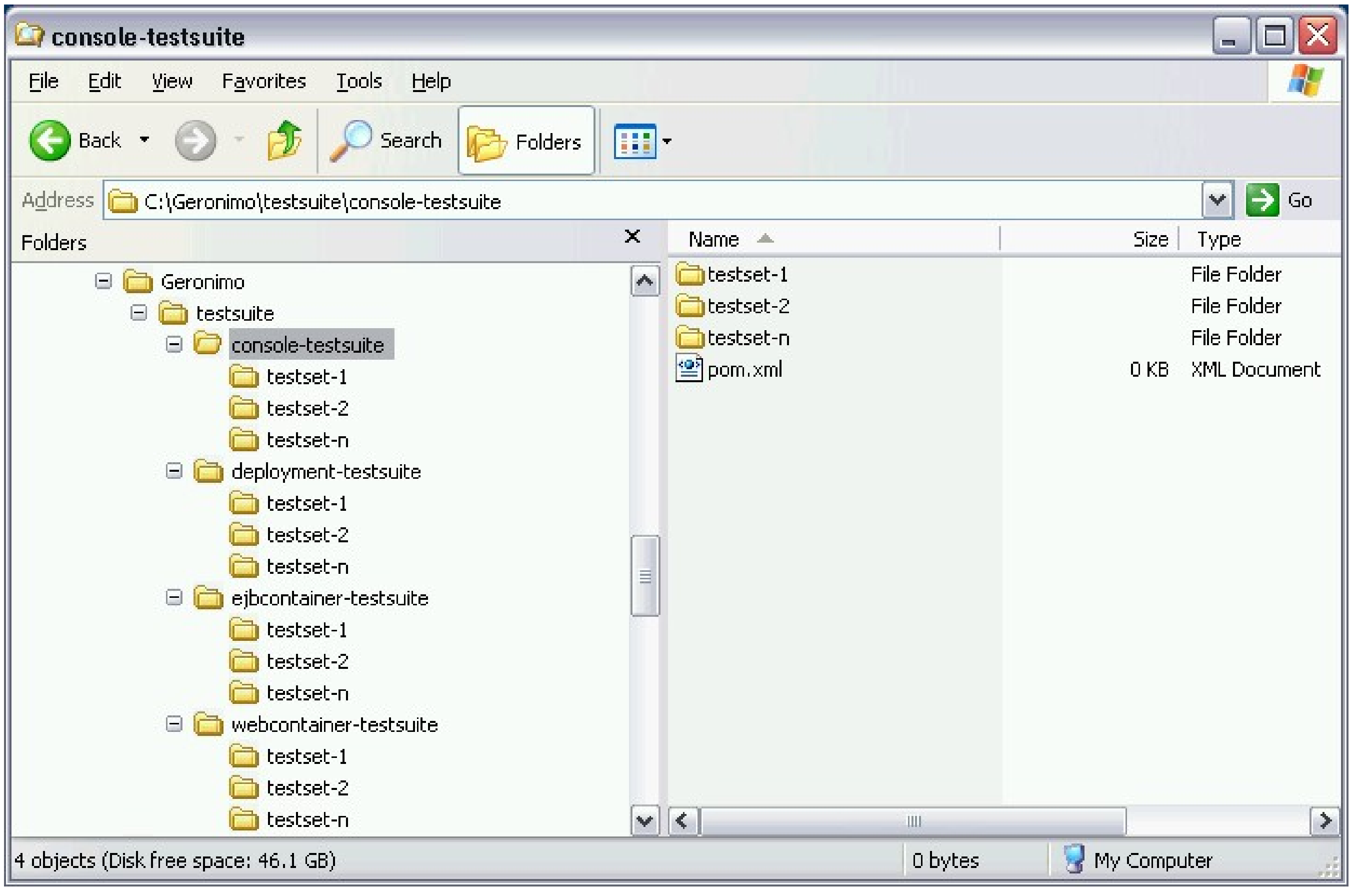
**command**

```
$>cd geronimo/testsuite
$>mvn site-deploy -DbuildNumber=<a timestamp>
```

This will generate a website for the entire testsuite and deploy it to a site under geronimo.apache.org. See the summary of the results here.

## Framework

The integration test is part of the regular geronimo tree and it resides under *geronimo/testsuite*. The child modules under this pom.xml are organized into various functional categories of the server. For eg., console-testsuite will perform tests on the console while the web-testsuites will perform tests against the web container.



C:\Geronimo\testsuite\console-testsuite

```
Geronimo
  testsuite
    console-testsuite
      testset-1
      testset-2
      testset-n
    deployment-testsuite
      testset-1
      testset-2
      testset-n
    ejbcontainer-testsuite
      testset-1
      testset-2
      testset-n
    webcontainer-testsuite
      testset-1
      testset-2
      testset-n
```

| Name | Size | Type |
|---|---|---|
| testset-1 | | File Folder |
| testset-2 | | File Folder |
| testset-n | | File Folder |
| pom.xml | 0 KB | XML Document |

4 objects (Disk free space: 46.1 GB) | 0 bytes | My Computer

Each *foo-testsuite* will contain another set of child modules that will actually perform the tests. Also each *foo-testsuite* will perform the following basic steps

- installing the server.
- starting it.
- invoking the tests under *testset-n.*
- stopping the server.
- creating surefire-reports for the tests in this suite.

# Plugins support.

- #geronimo-maven-plugin: install/start/stop server. deploy/undeploy modules
- #selenium-maven-plugin: start selenium server
- #maven-maven-plugin: invoke *testset-n*/pom.xml and run the tests.
- #testsuite-maven-plugin: generate surefire-report.html for *foo-testsuite*

## geronimo-maven-plugin

http://geronimo.apache.org/maven/server/maven-plugins/geronimo-maven-plugin/index.html

The goals to install/start/stop the server and deploy/undeploy modules are provided by the **geronimo-maven-plugin**. It has configuration option to either install a separate Geronimo server for each *foo-testsuite* or reuse one instance of the server installation. It also has configuration option to log the results for the purpose of generating surefire reports xml. The goals from this plugin are invoked by the pre-integration-test and post-integration-test phase of the Maven lifecycle appropriately.

A set of modules are provided for testing purposes. They are in the *geronimo/testsupport* directory. This pom.xml gets built during the beginning of the build cycle and provides various j2ee archives for use by the testsuite.

See deployment-testsuite for an example

## selenium-maven-plugin

http://geronimo.apache.org/maven/server/maven-plugins/selenium-maven-plugin/index.html

The console-testsuite under geronimo/testsuite tests the console using Selenium (http://openqa.org).  This requires a special server process started which is provided by our **selenium-maven-plugin**.

See console-testsuite for an example

## maven-maven-plugin

http://geronimo.apache.org/maven/genesis/plugins/maven-maven-plugin/index.html

This plugin invokes each of the *testset-n*/pom.xml in the integration-test phase of the maven lifecycle. The testcases are defined in those projects. The tests can be either junit or testng testcases. **maven-surefire-plugin** is configured to skip the tests during the test phase and run in the integration-test phase. surefire-report xmls are generated after the tests.

## testsuite-maven-plugin

http://geronimo.apache.org/maven/server/maven-plugins/testsuite-maven-plugin/index.html

Maven does not generate surefire-reports for pom packagings. *foo-testsuite* is a pom packaging with executions in pre-integration-test and post-integration-test phases. The results of those tests will not be captured by Maven. Also, the surefire reports from the *testset*-n projects should be rolled up into it's parent foo-testsuite. This is done by generate-surefire-xml goal in the testsuite-maven-plugin. Another goal, generate-surefire-report will generate the surefire-report.html from those surefire xmls. Finaly fix-menu goal will fix the menu for this html file.