Running TagUnit Tests from Ant

Like JUnit, TagUnit tests can be executed from the Jakarta Ant build tool so that any custom tag tests can be included as part of the continuous integrate, build and test cycle. To use TagUnit with Ant, simply follow these steps.

Step 1 : Put tagunit.jar in the Ant classpath

For Ant to be able to start TagUnit tests, the tagunit.jar file needs to be in the classpath when Ant is started. There are several ways of doing this:

1. Copy tagunit.jar into the $ANT_HOME/lib directory
2. Add tagunit.jar to your CLASSPATH environment variable.

Step 2 : Use the <tagunit> task in your Ant build file

To execute TagUnit tests from within ant, simply use the `<tagunit>` Ant task within your build file. To do this, you must first tell Ant where to find the `<tagunit>` task with the following (for an alternative method, see the next section):

```
<taskdef name="tagunit" classname="org.tagunit.ant.TagUnitTask"/>
```

Next, simply use the `<tagunit>` task to point Ant to your TagUnit tests with the following, making that you substitute the highlighted values to point to your own tests.

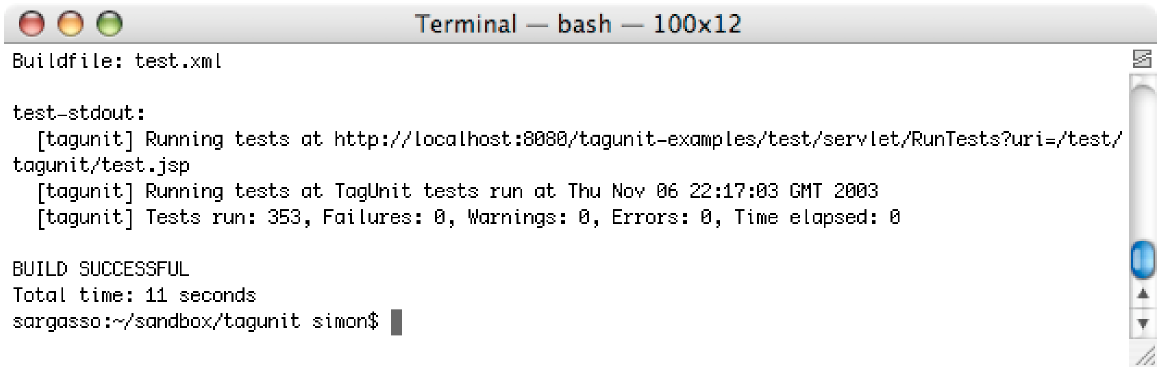
```
<tagunit url="http://localhost:8080/tagunit-
examples/test/servlet/RunTests?uri=/test/tagunit/test.jsp"
ignoreWarnings="true" stopOnFail="false"/>
```

The full set of attributes for this tag are as follows.

Attribute	Description	Required
url	This attribute points to the location of your tests.	Yes
ignoreWarnings	Should warnings be displayed?	No (defaults to true)
stopOnFail	Should failures cause the build to stop?	No (defaults to false)
formatter	The fully qualified class name of the formatter to use (see section below).	No (defaults to the console formatter)
logfile	Used in conjunction with the XML formatter (see below) to specify the name/path of the XML file.	No (defaults to stdout).

Step 3 : Start the server

Before you actually run the tests from Ant, don't forget to ensure that the web application/server hosting the TagUnit tests is running. When the tests are then executed, any errors or warnings will be output to the console in a similar way to JUnit test failures. The screenshot below shows an example of this output:



```
Buildfile: test.xml

test-stdout:
  [tagunit] Running tests at http://localhost:8080/tagunit-examples/test/servlet/RunTests?uri=/test/
tagunit/test.jsp
  [tagunit] Running tests at TagUnit tests run at Thu Nov 06 22:17:03 GMT 2003
  [tagunit] Tests run: 353, Failures: 0, Warnings: 0, Errors: 0, Time elapsed: 0

BUILD SUCCESSFUL
Total time: 11 seconds
sargasso:~/sandbox/tagunit simon$
```

Installing the TagUnit task into Ant

Another way to add the task (more permanently), is to add the task name and implementing class name to the `default.properties` file in the `org.apache.tools.ant.taskdefs` package. Then you can use it as if it were a built-in task, without the `<task-def>` tag.

You should add the following line to the `default.properties` file. This file can be found within the `ant.jar` file. Once you have made the change you can simply jar the files up again and the task will be visible to ant.

```
  tagunit=org.tagunit.ant.TagUnitTask
```

Using Formatters

Like other Ant tasks, the TagUnit task ships with a couple of formatters. One of these (`org.tagunit.ant.ConsoleFormatter`, the default) is used to output test results to stdout, while the other (`org.tagunit.ant.XmlFormatter`) is used to write the result to an XML file. To use a specific formatter, simply specify the fully qualified class name through the `formatter` attribute of the TagUnit task.