Webapp tasks

by Ovidiu Predescu, Jeff Turner

1. Webapp tasks

These tasks relate to Anteater's ability to deploy webapps on an internal Tomcat server.

1.1. servletContainer

Anteater has the ability to listen for incoming HTTP requests, which is useful in testing asynchronous SOAP. Advanced SOAP applications interact by sending asynchronous SOAP requests to each other. After the request is sent, the other party responds later with a SOAP request to the initiating party.

Listening for incoming HTTP requests is achieved by embedding the Tomcat 3.3 servlet container inside Anteater.

servletContainer is used in conjunction with the listener element, which registers *handlers* with the servlet container. The servlet container should be started before any listener task is to be executed. Usually this is achieved by placing the `servletContainer` task inside an *init* target, on which all the other tasks depend upon:

```
<target name="init">
  <servletContainer port="8100, 8101"/>
</target>

<target name="my-test" depends="init">
  <listener path="/foo" ... /> <!-- Register a listener -->
  <httpRequest request="/foo" .. > <!-- Call the listener -->
</target>
```

Attribute name	Type	Default value	Description
port	Integer	`8080`	Defines the ports on which the servlet container should listen on. Multiple ports can be specified by enumerating them with comma or space in between.
action	String	`start`	What action you wish the servlet container to take. If not specified, defaults to starting the container. Possible values are `start` and `stop`.
maxThreads	Integer		Defines the maximum number of threads that can be spawned off by the servlet container when handling incoming HTTP requests.
maxSpareThreads	Integer		Defines the maximum number of spare threads that can be alive at any time.
minSpareThreads	Integer		Defines the minimum number of spare threads that should be available at all times.

Table 1: Attributes

Elements allowed inside `servletContainer`: none

1.2. deploy

Because Anteater embeds a full blown servlet container in it, you can use Anteater to quickly deploy and test your Web application. This testing is not a replacement for testing how your Web application behaves when deployed on your preferred application server however. It is provided just as a quick way for you test the features of your application much faster.

In future, Anteater will add the ability to deploy a Web application on external servlet containers and application servers. And since Anteater is based on Ant, you can write a target which will start your application server right before the tests, run them and then shut it down at the end of the tests.

Attribute name	Type	Default value	Description
path	String		Specifies the *context* path where the new Web application will be accessible in the URL space of the servlet container.
webapp	String		The path to the location on the file system of the Web application. This path should point to either a `.war` file or to the expanded directory of the Web application.

Table 1: Attributes

Elements allowed inside `deploy`: none

Examples

The following example shows how to deploy <u>Apache Cocoon</u> on Anteater's internal servlet container.

```
<target name="deploy" depends="init" description="Run Cocoon">
  <deploy path="/cocoon" webapp="../xml-cocoon2/build/cocoon/webapp"/>
  ...
</target>
```