Match task

by Ovidiu Predescu, Jeff Turner

1. Match task

The result of an action task is an HTTP response or request object. Anteater allows you to test various characteristics of this object using match and test tasks.

A match task groups multiple tests that need to be performed on the result of an action task. All the tests inside the match task need to be successful in order for the match task to be successful.

An action task can have multiple match tasks that can be checked against the result object. These tasks will be executed in order until one of them succeeds, after which the result of the action task is considered to be successful. If none of the tasks succeed, the action task simply fails.

Since the `match` tasks are executed in the order in which they appear inside the action element, it is important to consider the tests that have side-effects, like assigning results to Ant properties.

```
<soapRequest description="Send a SOAP request"
   href="${url}" content="filename">
  <match>
    ...
  </match>

  <match>
    ...
  </match>
  <!-- Other match tasks here -->
  ...
</soapRequest>

<!-- Other Anteater action tasks here -->
```

In the above example, after the SOAP request returns the SOAP result object, the match tasks will be executed in order. If the first match task succeeds, the matching stops here, and the soapRequest task is successful. Otherwise, the next `match` tasks are executed, until one succeeds. If none of them succeeds, the soapRequest task fails. Depending on the value of the `haltonerror` property inherited from task's group, the subsequent action tasks inside it may or may not be executed.

Attribute name	Type	Default value	Description
assign	String		Specifies the name of a property to set if the match task's testers all succeed. In a listener task, this would be the match task responsible for sending the content. This allows script writers to apply conditional logic based on which match succeeded.
value	String		Specifies the value to set the `assign` property if the match task's testers all succeed.

Table 1: Attributes

Elements allowed inside `match`: all Test tasks.

1.1. Conditional logic

As noted in the Anteater_tags section, match tasks can be used as a form of boolean logic. Each `match` block contains a set of AND'ed tests (they must all pass), and as any `match` block's success renders the action task successful, they are effectively OR'ed together. Thus the task:

```
<httpRequest path="/foo.xml">
  <match>
    <A ../>
    <B ../>
  </match>
  <match>
    <C ../>
    <D ../>
    <E ../>
</match>
```

implements the boolean logic (A and B) or (C and D and E).

This can be exploited in various interesting ways. Here is how a listener can deliver different responses based on the value of a parameter:

```
<listener path="/foo">
  <match assign="chose123">
    <method value="GET"/>
    <parameter name="a" value="123"/>
    <sendResponse href="test/responses/good.html" contentType="text/html"/>
  </match>
  <match assign="chose456">
    <method value="GET"/>
    <parameter name="a" value="456"/>
    <sendResponse href="test/responses/bad.html" contentType="text/html"/>
  </match>
</listener>
```

Here is a larger example of how one can determine the type of server being tested, by using the 'assign' attribute on matchers:

```
<httpRequest path="/">
  <match assign="servertype" value="gnu-apache">
    <header name="Server" value="Apache/1.3.24 (Unix) Debian GNU/Linux"/>
  </match>
  <match assign="servertype" value="generic-apache">
    <header name="Server" assign="apache-version" pattern="Apache/([\d.]+)"/>
  </match>
  <match assign="servertype" value="IIS6">
    <header name="Server" value="Microsoft-IIS/6.0"/>
  </match>
  <match assign="servertype" value="generic-IIS">
    <header name="Server" pattern="IIS"/>
  </match>
  <match assign="servertype" value="unknown">
  </match>
</httpRequest>
<echo>servertype is ${servertype}</echo>
```

Note:

This ability to do conditional logic as a byproduct of an operation is reminiscent of how declarative languages like Prolog do it.