Configuring with Struts 1.1

by Don Brown

A guide to stxx

Table of contents

Copyright © 2005 Jeff Pennal and Don Brown. All rights reserved.

1. web.xml

Struts 1.1 introduced a significant number of changes to its architecture. So much, that stxx had to be changed radically in order to co-exists with Struts. Many of these changes to the Struts framework were geared towards making it more extensible. One of these changes, the Struts Plug-in framework, is what stxx uses to make transformations happen. Using stxx with Struts 1.1 requires only a few changes to the web.xml file. Instead of changing the pointer to the main ActionServlet class as you did with Struts 1.0, you can leave all of the Struts default parameters the same, just adding the following parameter, which points to the stxx properties file where further configuration parameters can be set. This file is expected to be loaded from the classpath, so it must reside in `/WEB-INF/classes` to be found. This properties file is used by both versions of stxx.

```
<init-param>
  <param-name>stxxInit</param-name>
  <param-value>/stxx.properties</param-value>
</init-param>
```

Name	Value	Status
stxxInit	`/stxx.properties`	Required

Table 1: All web.xml init-params used by stxx

2. struts-config.xml

In Struts 1.1, the config file is no longer hard-coded to struts-config.xml. The file and it's location are defined by the plugin property 'pipeline-config' for stxx 1.2 or greater and 'transform-config' for stxx 1.1. In this file, you will need to define the plug-in parameters for stxx to be registered in Struts as a plug-in.

Name	Value	Status
pipeline-config	Location of pipeline configuration file located from the root of the web application.	Required for stxx 1.2 unless `transform-config` is set.
transform-config	Location of transform configuration file located from the root of the web application.	Required for stxx 1.1 or earlier, or if `pipeline-config` isn't set.

Table 1: All struts-config.xml plugin properties

The plug-in configuration will look like this:

```
<plug-in className="com.oroad.stxx.plugin.StxxPlugin" >
    <set-property property="pipeline-config"
```


Copyright © 2005 Jeff Pennal and Don Brown. All rights reserved.

```
                    value="/WEB-INF/stxx-transforms.xml" />
</plug-in>
```

This tells Struts to load the class StxxPlugin and to pass the parameter pipeline-config to it. This property tells the plug-in where to find all the transformations to perform.

If you plan to use the localization features of Struts, you also need to configure Struts to use a special factory when loading the message resources. The configuration should look like this:

```
<message-resources parameter="com.oroad.stxx.example.ApplicationResources"
factory="com.oroad.stxx.util.PropertyMessageResourcesFactory"/>
```

Transformations are no longer defined in struts-config.xml. The plug-in is configured to look for any forward with a '.dox' extension. In the case of `pipeline-config`, Struts forwards with '.dox' extensions are matched by a pipeline that contains one or more transform. In the case of `transform-config`, forwards are matched to a <transform> tag in the `transform-config.xml` file.

To make a Struts action map to a transform, simply make the path attribute of the forward tag end with a '.dox', such as:.

```
<forward name="success" path="index.dox"/>
```

The path index.dox will match a pipeline in the `pipeline-config.xml` file.

3. Transform Definitions

As of stxx version 1.2, you can wrap your transform definitions in pipelines that are used to match multiple Struts action forwards. A simple wildcard matching scheme is used to match forward types. Previously, you had to write one transform per Struts action forward which could result in hundreds of lines for a moderately sized application, all of which the needed to be in sync with the Struts configuration files. Now, a moderately sized application (>50 action mappings) requires only a couple of pipelines.

This is an example of a more complex pipeline definition to show its syntax:

```
<pipeline match="browser/*.dox">
    <transform type="html">
        <param name="path" value="/{1}.xsl" />
        <param name="path" value="/style.xsl" />
        <param name="render" value="server" />
    </transform>
    <transform type="html" when="MSIE">
        <param name="path" value="/{1}_IE.xsl" />
        <param name="render" value="client" />
    </transform>
</pipeline>
```


Copyright © 2005 Jeff Pennal and Don Brown. All rights reserved.

A pipeline is used to match multiple action forwards. This pipeline will match "browser/index.dox" or "browser/foo.dox". A "*" matches any text other than "/", and a "**" matches any text _including_ "/". This way you can encode information in the "path" attribute of your "forward" element from your struts-config.xml like using the scheme "PIPELINE_TYPE/NAME.dox". This works exactly like Cocoon 2, in fact the wildcard code is copied from Cocoon. It should make the transition to Cocoon easier.

A transform two attributes: "type" and "when". The type refers to the name of the transformer to use to handle the operation (in this case, an instance of CachedXSLTransformer). The "when" attribute is the same as the "selector" attribute was in the previous schema or the "name" attribute is in the Struts 1.0 version. It basically is the switch to determine which transform gets used. In this case, the transform is selected by determining which browser the client is using. See stxx.properties for more information.

In the transform's params, the "{1}" is used to match the first wildcard in the pipeline's "match" attribute. You can have up to 9 wildcards. Notice there can be more than one instance of a parameter. These values are made available to the Transformer as a List.

Any other parameters, like "render" or "debug", would be used similiarly. For Transformer creators, you can define your own parameters to suite your purposes.

Copyright © 2005 Jeff Pennal and Don Brown. All rights reserved.