LayoutTemplatePattern

Have you ever wondered how to create skins to use in Cocoon? This LayoutTemplate pattern for the sitemap allows you to do exactly that. While there are certainly other ways to do this using xinclude or cinclude, I'm partial to this way of organizing skins myself, as I believe that it makes the switching out of skins that much easier. ILinKuo

The essence of the LayoutTemplate pattern is that it uses an xslt stylesheet to replace selected elements in the layout template with corresponding elements in the source xml. It'll be easier to explain after you've run it within your own Cocoon pipeline to see how it works.

The setup

My directory structure looks like this:

```
cocoon
  - sitemap.xmap
  - xsl
      - insertIntoLayoutTemplate.xsl
  - xml
      - dummyInsert.xml
  - templates
      - dummyTemplate.xml
```

and my sitemap match looks like this:

```
<map:match pattern="test-layoutTemplatePattern">
        <map:generate src="xml/dummyInsert.xml"/>
        <map:transform src="xsl/insertIntoLayoutTemplate.xsl">
                <map:parameter name="relTemplatePath"
                        value="../templates/dummyTemplate.xml"/>
        </map:transform>
        <map:serialize type="xml"/>
</map:match>
```

From the sitemap, you specify a layout template to use (in this case, it's dummyTemplate.xml), and the stylesheet insertIntoLayoutTemplate.xsl does the job of looking up corresponding elements in your template from dummyInsert.xml.

The files

```
<!-- dummyInsert.xml -->
<results>
        <page-title>Test Page for <b>sitemap template insert pattern</b></page-title>
        <leftNav><a href="www.apache.org">Apache Rules</a></leftNav>
        <content>
                <h1>insert your content here!</h1>
        </content>
        <footer>copyright: none</footer>
</results>
```

In the above file, I have the page elements that I want to insert into the layout template

```
<!-- dummyTemplate.xml -->
<html xmlns:c-insert="http://apache.org/cocoon/insert">
<head/>
<body>
<c-insert:page-title>page-title goes here</c-insert:page-title>
<table border="1">
        <tr><td colspan="2"><c-insert:page-banner/></td></tr>
        <tr><td rowspan="2" width="200"><c-insert:leftNav/></td>
                <td width="500"><c-insert:content/></td>
        </tr>
        <tr><td><c-insert:footer/></td></tr>
</table>
</body>
</html>
```

This layout template is a simple four cell table with a top banner, left navigation, central content area, and footer. I've defined a special namespace "c-insert" to mark those elements that I want to have replaced. So, <c-insert:page-banner>, <c-insert:leftNav>, <c-insert:content>, and <c-insert:footer> will get replaced by the corresponding top-level elements <page-banner> <leftNav> <content> and <footer> in my source document dummyInsert.xml. However, since the source document doesn't contain a <page-banner> element, it will simply be removed.

The stylesheet that does all the work is:

```
<!-- insertIntoLayoutTemplate.xsl-->
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
        xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
        xmlns:c-insert="http://apache.org/cocoon/insert"
        xmlns:xhtml="http://www.w3.org/1999/xhtml">

<xsl:param name="relTemplatePath"/>
<xsl:variable name="layoutTemplate" select="document($relTemplatePath)"/>

<!-- Selects the document in the pipeline. The top level elements
        in this document will be inserted into the layout template -->
<xsl:variable name="results" select="/"/>

<xsl:template match="/">
        <xsl:apply-templates select="$layoutTemplate/*"/>
</xsl:template>

<!-- look up content to be inserted using local-name as key -->
<xsl:template match="c-insert:*">
        <xsl:copy-of select="$results/*/*[local-name()=local-name(current())]/node()"/>
</xsl:template>

<!-- identity transform used to copy all other elements as is -->
<xsl:template match="@*|node()" priority="-1">
        <xsl:copy><xsl:apply-templates select="@*|node()"/></xsl:copy>
</xsl:template>

</xsl:stylesheet>
```

Try it out!

Usage

For simplicity, this example shows how to insert static content into the template, but of course it's easy to use dynamically generated content as well, as long as you make sure that by the time the xml gets to the stylesheet in the pipeline, all the top level elements are named appropriately.

Finally, to change your "skin", just change the value of the parameter passed into the stylesheet. In this example, the skin cannot be selected dynamically. but I'll show in a follow-up how this can be done dynamically.

Debugging

A word of warning: Because of Cocoon's caching, if you implement this pattern and then tweak the template document, your changes will not show up immediately. This is because Cocoon caches the calls it makes via the document() function. In order to see your changes, you'll have to either modify or touch the insertIntoLayoutTemplate.xsl file.

Feel free to modify this page and let me know what you think.

Note: A similar approach is described in Style-Free Stylesheets at cocooncenter.
– AndreasHartmann

Note: Thanks for pointing me to the cocooncenter article. I hadn't seen it before. It's essentially the same approach with three small differences:

- I pass in the template path as a parameter
- I use local-name() rather than @name for more robust matching
- The syntax for indicating which elements to be replaced is slightly different, but based on the same principle of using namespaces to distinguish them from regular xhtml elements. --ILinKuo

---