

Sitemap Tutorial

Bernhard Huber, berni_huber@a1.net

1 Sitemap Tutorial

This document describes some features used in the sitemap of the **CssColumnSite**.

2 Sitemap Aggregating

This section explains in detail the reasons for using the `map:aggregate`, and `map:part` sitemap elements. Moreover the attributes `element`, and `strip-root` are explained in more detail.

The sitemap contains following snippet:

```
<!-- generate a page by merging all parts making up a page
-->
<map:resource name="show-page">
  <map:aggregate element="site">
    <map:part src="cocoon:/page-{page}-top-col-1"
      element="top-col-1" strip-root="true"/>
    <map:part src="cocoon:/page-{page}-mid-col-1"
      element="mid-col-1" strip-root="true"/>
    <map:part src="cocoon:/page-{page}-mid-col-2"
      element="mid-col-2" strip-root="true"/>
    <map:part src="cocoon:/page-{page}-mid-col-3"
      element="mid-col-3" strip-root="true"/>
    <map:part src="cocoon:/page-{page}-mid-col-4"
      element="mid-col-4" strip-root="true"/>
    <map:part src="cocoon:/page-{page}-bottom-col-1"
      element="bottom-col-1" strip-root="true"/>
  </map:aggregate>
  <map:transform src="stylesheets/site2html.xsl"/>
  <map:serialize/>
</map:resource>
```

2.1 Reasoning

The reason for using the `map:aggregate`, and `map:part` is simple, we want to merge documents from the section directories to a single xml document. The sitemap

`map:aggregate`, and `map:part` elements provides this functionality.

2.2 map:aggregate

As merged-in documents should be contained inside a common root-element. The attribute `element` of the `map:aggregate` specifies the name of this root-element name.

In the sitemap snippet `<map:aggregate element="site"/>` the root-element is specified to have the element name `site`.

Hence the xml document serialized by this snippet should look like this

```
...
<site>
  <!-- content of part top-col-1 -->
  ...
  <!-- content of part bottom-col-1 -->
</site>
```

2.3 map:part

The sitemap `map:part` defines the parts making up the total aggregation. Each part defines its source in the `src` attribute.

Moreover the attribute `element` is used to mark in the xml document that this xml document subtree has been merged in using this part. This is used later on in the stylesheet `site2html.xsl` to treat each part in a special way.

The attribute `strip-root="yes"` strips off the root element of each part document source. This is done for convenience, later on in the stylesheet you can write a bit more simpler xpath expression.

Enhancing the example above, the serialized xml document looks like this:

```
...
<site>
  <!-- content of part top-col-1 -->
  <top-col-1>
    <!-- content of part top-col-1 w/o it root element -->
  </top-col-1>
  ...
  <!-- content of part bottom-col-1 -->
  <bottom-col-1>
    <!-- content of part bottom-col-1 w/o it root element -->
  </bottom-col-1>
</site>
```

3 Document Existence

This site defaults to a default document per section. If a requested page does not find its xml-document in the section directory, the xml document `index.xml` is used for generating the section content.

For example, the `mid-col-1` section directory does not contain an xml document for the `faq.html` page. Does the `mid-col-1` section for the `faq.html` page uses the document `mid-col-1/index.xml` for providing the content of the `mid-col-1` section of the `faq.html` page.

The following section of the sitemap checks the resolvability of a section xml-document:

```
<map:match pattern="page-*-mid-col-1">
  <map:act type="resource-exists" src="docs/mid-col-1/{1}.xml">
    <map:call resource="load-page">
      <map:parameter name="sect" value="mid-col-1"/>
      <map:parameter name="page" value="{../1}"/>
      <map:parameter name="resource-exists"
        value="{resource-exists}"/>
    </map:call>
  </map:act>
</map:match>
```

The action `resource-exists` checks if its `src` attribute value is resolvable. If it is resolvable it sets the sitemap parameter `resource-exists` to `"true"`, else the sitemap parameter `resource-exists` to `"false"`.

The invoked resource `load-page` evaluates the sitemap parameter `resource-exists`.

4 Sitemap DocumentSelection

The sitemap of this site contains following snippet:

```
<!-- load a page of a section if page is not
  available load index page of this section
-->
<map:resource name="load-page">
  <map:select type="parameter">
    <map:parameter name="parameter-selector-test"
      value="{resource-exists}"/>
    <map:when test="true">
      <map:generate src="docs/{sect}/{page}.xml"/>
    </map:when>
    <map:otherwise>
      <map:generate src="docs/{sect}/index.xml"/>
    </map:otherwise>
  </map:select>
</map:resource>
```

```
</map:select>  
  <map:serialize type="xml"/>  
</map:resource>
```

This sitemap snippet evaluates the parameter `resource-exists`.

The value of this parameter is already defined in the previous `ExtendedResourceAction`. Here we use this value including either the existing section-page xml-document, or including the default section-page xml-document `index.xml`.

The sitemap selector of type `parameter` is used to yield the result of the parameter `resource-exists`.

If the parameter `resource-exists` is equal to `"true"`. A file generator is invoked loading the xml document from the source `docs/{sect}/{page}.xml`.

The calling sequence of this resource passes values for both parameters. The parameter `sect` specifies the top, mid, or bottom section name; the parameter `page` specifies the document in side of this section.