Publication Templating

Table of contents

1 What is Publication Templating?

Imagine you are a company or organisation with some departments which want to edit their content using a Lenya-based CMS. All departments use similar publications, sharing lots of functionality. The layout is based on a common corporate identity, but some departments want to use their own logo or tweak the navigation menu style.

If all departments develop their own publications, it will be very hard to keep them consistent, to add changes to all publications or to update them to a newer Lenya version. This can be simplified using publication templates. You define a base (template) publication which all others are derived from.

```
                     +----------------------+
                     | Template Publication |
                     +----------------------+
                                 |
        +-----------------+------+---------------------+
        |                 |                            |
+--------------+  +--------------+             +--------------+
| Department A |  | Department B |     ...     | Department X |
+--------------+  +--------------+             +--------------+
```

Multiple layers of templates are supported.

```
                 +--------------------------------+
                 | University Website Publication |
                 +--------------------------------+
                                 |
        +-----------------+------+------------------------+
        |                 |                               |
+------------+  +--------------------+     ...     +-------------+
| Uni Zurich |  | Harvard University |     ...     | UC Berkeley |
+------------+  +--------------------+             +-------------+
                          |
           +----------+---+-----------+
           |          |               |
     +----------+  +-----+       +---------+
     | Business |  | Law |  ...  | Medical |
     +----------+  +-----+       +---------+
```

2 The Concept of Publication Templating

- There is a publication *my-pub*.
- It depends on a template which is called *template(my-pub)*.
- When a file is requested using the templating mechanism (`fallback://xslt/page2xhtml.xsl`), it is searched in a certain traversing order:
  1. `context://lenya/pubs/my-pub/xslt/page2xhtml.xsl`
  2. `context://lenya/pubs/template(my-pub)/xslt/page2xhtml.xsl`
  3. `context://lenya/pubs/template(template(my-pub))/xslt/page2xhtml.xsl`
  4. ...
  5. `context://xslt/page2xhtml.xsl`

The publication *my-pub* is called an **instance** of the publication *template(my-pub)*. Note that, in contrast to the fallback mechanism in Lenya 1.2, the prefix `lenya` is not used, but the path is resolved relatively to the `context://` root.

3 Declaration of a Template

The template of a publication is declared in `my-pub/config/publication.xml`:

```
<publication>
  ...
  <templates>
    <template id="my-template"/>
  </templates>
  ...
</publication>
```

4 Usage

To invoke publication templating, it is necessary to use the `fallback://` protocol for all relevant files. For an XSLT stylesheet, the according pipeline looks as follows:

```
<map:transform src="fallback://xslt/doctypes/doctype2xhtml.xsl">
```

5 XSLT Include and Import

To leverage the publication templating concept, it is necessary to apply it to included or imported stylesheets as well. Fortunately, we can make use of the `fallback://` protocol in XSLT stylesheets. At the moment, this only works with Xalan which means you have to use this one as the default transformer.

```
<xsl:include href="fallback://header.xsl"/>
```

To simplify overriding of XSLT stylesheets, it is very useful to import the template stylesheet. The template-fallback source factory skips the current publication when resolving the file, i.e. it resolves the first existing ancestor file:

```
<xsl:import href="template-fallback:mypub://template/xslt/common/header.xsl"/>
```

It is necessary to pass the ID of the publication which contains the importing file after the protocol string because otherwise the template-fallback resolving mechanism will fail. For more information, see bug 40564 (http://issues.apache.org/bugzilla/show_bug.cgi?id=40564) .

6 Sitemaps

If a sitemap is loaded from a template publication, it is very important that the sitemap is completely fallback-enabled. Otherwise, the source resolver will resolve sources relatively to the template sitemap instead of using the overridden ones.

In `lenya/global-sitemap.xmap`, all publication sitemaps are mounted using the fallback module, for instance

```
<!-- Enter the actual publication -->
<map:match pattern="*/**">
```

```
  <map:mount uri-prefix="{1}" src="{fallback:sitemap.xmap}"/>
</map:match>
```

7 Usecases

The usecase framework (../../../../../docs/2_0_x/reference/usecase-framework/index.html) supports publication templating by default.

If you can't (or don't want to) use the usecase framework, you have to implement your own usecase sitemap. The traversing order for usecase sitemaps is

1. `context://lenya/pubs/my-pub/usecase.xmap`
2. `context://lenya/pubs/template(my-pub)/usecase.xmap`
3. `context://lenya/pubs/template(template(my-pub))/usecase.xmap`
4. ...
5. `context://lenya/usecase.xmap`

This behaviour is achieved by the usecase fallback module which is called in `global-sitemap.xmap`:

```
<map:match type="usecase" pattern="*">
  <map:mount src="{usecase-fallback:{1}}" uri-prefix=""/>
</map:match>
```

The decision which `usecase.xmap` to choose is based on the usecase configuration in `publication.xml`. To declare a usecase to be implemented by a publication, add the corresponding entry:

```
<publication>
  ...
  <usecases>
    <usecase name="create"/>
  </usecases>
  ...
</publication>
```

8 Setting Up a Publication To Support Templating

The service `org.apache.lenya.cms.publication.templating.Instantiator` is responsible for creating instances of publications which support templating. If your publication shall support templating, you have to follow these steps:

8.1 Implement an Instantiator Class

```
package org.myproject.lenya;

public class MyInstantiator extends AbstractLogEnabled implements Instantiator {

    public void instantiate(Publication template, String newPublicationId, String name)
            throws Exception {
        ...
    }

}
```

8.2 Add it to cocoon.xconf Using a Patch File

For instance `my-pub/config/cocoon-xconf/instantiator.xconf`:

```
<xconf xpath="/cocoon/template-instantiators"
       unless="/cocoon/template-instantiators/component-instance[@name = 'default']">

    <component-instance name="mypub"
                        logger="myproject.publication"
                        class="org.myproject.lenya.MyInstantiator"/>

</xconf>
```

8.3 Declare the Instantiator in publication.xml

```
<publication>
  ...
  <template-instantiator name="mypub"/>
  ...
</publication>
```