Resource Types

Table of contents

1. Introduction

A resource type defines a certain XML source format, together with processing options. It typically consists of

- an XML structure definition (e.g., Relax NG)
- some presentation pipelines,
- some presentation XSLT stylesheets,
- usecases to manipulate documents.

All of these can be shared between several resource types.

The information describing a resource type is managed by a `ResourceType` (http://lenya.apache.org/apidocs/1.4/org/apache/lenya/cms/publication/ResourceType.html) service. The default implementation is `ResourceTypeImpl` (http://lenya.apache.org/apidocs/1.4/org/apache/lenya/cms/publication/ResourceTypeImpl.html) . It implements `ThreadSafe`, which ensures that only a single instance of every resource type is created. It is not possible to declare multiple resource types with the same name.

2. Choose a Unique Resource Type Name

You should choose a reasonable name for your resource type.

Note:

In the examples, we use the name *profile* (page with information about a person).

3. Adding the Resource Type to a Publication

The resource types used by a publication are declared in `publication.xconf`, including the assignment of a workflow schema to a resource type:

```
<publication>
  ...
  <resource-type name="xhtml" workflow="workflow.xml"/>
  <resource-type name="homepage" workflow="workflow.xml"/>
  <resource-type name="links" workflow="workflow.xml"/>
  <resource-type name="profile" workflow="workflow.xml"/>
  ...
</publication>
```

You can add references to any resource types configured in `<lenya-webapp>/WEB-INF/cocoon.xconf` (see below), containing

- resource types provided by modules,
- resource types of template publications, and
- resource types of the publication itself.

Adding a custom resource type to your publication includes the following steps:

3.1. Providing a Sample XML Document

If you want to enable users to create new resources belonging to your resource type, it is useful to provide a sample XML document. If you want to use the `DefaultBranchCreator` that ships with Lenya, you have to add the sample document because it is used as a template for creating new resources.

The sample document is typically placed in `<publication>/lenya/resources/samples/`. You can choose an arbitrary filename, but it is recommended to use the resource type name (e.g., `profile.xml`).

3.2. Providing an XML Structure Definition

This step is only needed if you want to edit resources with Lenya or validate them after they have been imported or manipulated. The type of the structure definition (XML Schema (http://www.w3.org/XML/Schema) , Relax NG (http://www.relaxng.org/) , ...) depends on the editor or validator you want to use. For instance, the BXE (http://www.bitfluxeditor.org/) WYSIWYG editor requires a Relax NG document.

The structure definition document is typically placed in the directory `<publication>/lenya/resources/schemas/`. The name of the file is arbitrary, but it is recommended to use the resource type name (e.g., `profile.rng`).

3.3. Creating a Workflow Schema

If your resources should have a workflow, you have to add a workflow schema for your resource type as described in Workflow Configuration (../../../docs/1_2_x/components/workflow/configuration.html) . A workflow schema can be shared between multiple resource types. The workflow schema is assigned to a resource type in `publication.xconf` (see section *Adding Resource Types to a Publication*).

3.4. The Resource Type Definition

To declare a custom resource type and assign the creator, schema etc. to it, add the component instance to an XPatch file (e.g., `<publication>/config/cocoon-xconf/resourcetype-profile.xconf`):

```
<xconf xpath="/cocoon/resource-types"
       unless="/cocoon/resource-types/component-instance[@name = 'profile']">

  <component-instance name="profile"
                      logger="lenya.resourcetypes.profile"
                      class="org.apache.lenya.cms.publication.ResourceTypeImpl">
    <schema src="fallback://lenya/resources/schemas/profile.rng"
            language="http://relaxng.org/ns/structure/0.9"/>
    <creator src="org.apache.lenya.cms.authoring.DefaultBranchCreator">
      <sample-name>fallback://lenya/resources/samples/profile.xml</sample-name>
    </creator>
    <link-attribute xpath="//*[namespace-uri() = 'http://foo.bar.org/profile']/@href"/>
    <format name="xhtml" uri="cocoon://modules/profile/profile.xml"/>
  </component-instance>

</xconf>
```

This XPatch file will be used when the publication is deployed (e.g., when you issue a `./build` command). Its contents will be patched into `<lenya-webapp>/WEB-INF/cocoon.xconf`.

3.5. Define a Custom Menubar

If you want to use a custom menubar for your resource type, follow the guidelines on the page The Lenya Menubar (../../../docs/1_2_x/components/layout/lenya-menubar.html) . Typically, a menubar is shared between multiple resource types. Small customizations can be achieved with Java code in the menubar XSP.

To let the user create new resources using the `DefaultBranchCreator`, you have to add the following menu item:

```
  <item uc:usecase="site.create" href="?doctype=profile"><i18n:text>New Profile
Document</i18n:text></item>
```

3.6. Formats

A resource type provides a set of *formats* to provide different ways of presenting content documents. The formats are defined in the resource type declaration:

```
<format name="xhtml" uri="cocoon://modules/profile/profile.xml"/>
```

The `uri` attribute of the `format` element may refer to an arbitrary URL, which is typically a request into the module itself. This URI is matched inside the module sitemap (in our case, `modules/profile/sitemap.xmap`). Typically, an XSLT is applied to the content document to transform it into another format (XHTML, XSL-FO, ...):

```
<!-- apply a format -->
<map:match pattern="*.xml">
  <map:generate src="lenya://lenya/pubs/{page-envelope:publication-id}/ \
    content/{page-envelope:area}/{page-envelope:document-path}"/>
  <map:transform src="fallback://lenya/modules/profile/xslt/profile2xhtml.xsl">
    <map:parameter name="rendertype" value="{request-param:rendertype}"/>
    <map:parameter name="nodeid" value="{page-envelope:document-name}"/>
    <map:parameter name="language" value="{page-envelope:document-language}"/>
  </map:transform>
  <map:serialize type="xml"/>
</map:match>
```

To request a formatted document, use the `format-...` attribute of the `resource-type` input module:

```
<!-- aggregate navigation components and XHTML-formatted content -->
<map:aggregate element="cmsbody">
  <map:part src="cocoon://modules/sitetree/{2}/{3}/breadcrumb/{5}.xml"/>
  <map:part src="cocoon://modules/sitetree/{2}/{3}/tabs/{5}.xml"/>
  <map:part src="cocoon://modules/sitetree/{2}/{3}/menu/{5}.xml"/>
  <map:part src="cocoon://modules/sitetree/{2}/{3}/search/{5}.xml"/>
  <map:part src="{resource-type:format-xhtml}"/>
</map:aggregate>
```

3.7. Presentation

To make your resources available as HTTP pages, you have to add the appropriate pipelines and XSLT stylesheets. In general, there are no restrictions.

If you derive your publication from the default publication, the pipelines have to be placed in `<publication>/doctypes.xmap`. The stylesheets are located in `<publication>/xslt/` and are named `<resource-type>2xhtml.xsl` (e.g., `profile2xhtml.xsl`). The stylesheet is supposed to generate a valid XHTML fragment (in the XHTML namespace) with `<div id="body">` as the document element.