WikiRenderer

This is a small sample Forrest webapp (available for download here), which demonstrates Forrest's support for Wiki content by rendering a directory full of Wiki text files. While mainly done as an exercise and demonstration, having PDF renditions of Wiki content (including aggregations) may prove useful to some.

1. Wiki parsing

All the hard work of parsing is done by the Chaperon parser, using the `wiki.grm` grammar from Cocoon's samples. This allows us to convert Wiki syntax into Forrest's document-v12 format, which can then be rendered to HTML or PDF. Many thanks to Stephan Michels for making this possible!

2. Navigation

A directory, `src/documentation/content/xdocs/wiki/`, contains a selection of Wiki files from the Cocoon Wiki:

```
[wikirenderer ~]$ ls src/documentation/content/xdocs/wiki
ApacheModProxy.cwiki                    DistributingCocoonApplications.cwiki  RhinoWith
AuthWithTomcat.cwiki                    EXistInCocoon.cwiki                   Separatio
BeginnerInstallTomcatUnix.cwiki         ForrestProposal.cwiki                 TextForma
BeginnerInstallTomcatWindows.cwiki      index.cwiki                           WoodySamp
BeginnerSimpleWebappOrganisation.cwiki  Jars2exclude.cwiki                    XMLFormXi
BlocksDefinition.cwiki                  JBossDeployment.cwiki                 XMLFormXi
book.xml                                ModularDatabaseActions.cwiki          XSPSyntax
CommandLine.cwiki                       OlderNews.cwiki
ConfiguringTheLogs.cwiki                OpenOfficeGeneration.cwiki
```

A menu for this directory is auto-generated by overriding navigation.xmap, and defining this matcher:

```
    <map:match pattern="wiki/book.xml">
      <map:generate type="directory" src="content/xdocs/wiki">
        <map:parameter name="dateFormat" value="yyyy-MM-dd hh:mm" />
      </map:generate>
      <map:transform src="resources/stylesheets/directory2book.xsl" />
      <map:serialize type="xml"/>
    </map:match>
```

The 'directory' generator generates an XML directory listing of the `wiki/` directory. The

`directory2book.xsl` stylesheet transforms this into Forrest's `book.xml` menu format. The result is that `wiki/*.html` URLs all have a menu with links to other Wiki pages.

3. Aggregation

Having a single page of all Wiki content is useful for printing or searching. This is achieved by overriding the `forrest.xmap` sitemap, which defines the XML for raw pages, and defining a `wiki/combined.xml` page source:

```
    <map:match pattern="wiki/combined.xml">
      <map:generate type="directory" src="content/xdocs/wiki">
        <map:parameter name="dateFormat" value="yyyy-MM-dd hh:mm" />
      </map:generate>
      <map:transform src="resources/stylesheets/directory2cinclude.xsl" />
      <map:transform type="cinclude"/>
      <map:transform src="resources/stylesheets/docs2document.xsl" />
      <map:transform type="idgen"/>
      <map:serialize type="xml"/>
    </map:match>
```

Once again, we use the DirectoryGenerator to create an XML listing of all files in the `wiki/` directory. This is then transformed by `directory2cinclude.xsl` into food for the CIncludeTransformer, of the form:

```
      <content>
        <cinclude:include src="cocoon:/wiki/ApacheModProxy.xml"/>
        <cinclude:include src="cocoon:/wiki/AuthWithTomcat.xml"/>
        <cinclude:include src="cocoon:/wiki/BeginnerInstallTomcatUnix.xml"/>
        <cinclude:include src="cocoon:/wiki/BeginnerInstallTomcatWindows.xml"/>
        <cinclude:include src="cocoon:/wiki/BeginnerSimpleWebappOrganisation.xml"/>
        <cinclude:include src="cocoon:/wiki/BlocksDefinition.xml"/>
        ...
      </content>
```

The CIncludeTransformer then pulls in the content from each referenced pipeline to form one large XML file of all Wiki content.

The `docs2document.xsl` stylesheet then converts `<document>` tags to `<section>` tags, so the result is valid doc-v12 XML. The `idgen` transformer derives a #fragment-identifier for each section. These will be used in the mini-TOC at the top of the page. Title collisions are resolved to render each #identifier unique within the aggregated page.

Once the `combined.xml` matcher is defined, nothing else is required; the combined.html and combined.pdf pages can now be rendered.

Warning:

The combined PDF is 150 pages (581k), and requires more than the default 64MB allocated to the JVM to render. This can be fixed by running your JVM with a `-Xmx128m` flag, for example in `CATALINA_OPTS`