XfolioRedirectAction

----* Cocoon app design, probably because of the samples, usually control what is served (I want a CSS ? declare it in the sitemap). If Cocoon is used as a more free way server, (ex: an httpd.apache pointing on a folder) lots of nice things could be done (transform !). But if you open this door, it's not always HTML, you may have strange surprises of what users can produce.

For example, which welcome page to choose when there's no equivalent of an index.html ? Redirection accross all a sitemap logic could be expensive and dangerous (infinite loops ! Real life experience). If nothing answer at the end of a sitemap, it could be nice to have something which say, for what is requested, I've nothing exact but this is a not too bad answer.* ----
<<MailTo(frederic DOT glorieux AT xfolio DOT org)>>

Logic

For a first step, what to do when a directory is called (match="" or "**/") ? From Cocoon samples, you have usually things like that

```
<map:match pattern="">
  <map:generate src="welcome.xml"/>
  <map:transform src="welcome.xslt"/>
  <map:serialize type="xhtml"/>
</map:match>
```

Perfect. You know what you have, it's an XML you are waiting for, you know how to transform it.

Imagine now you have authors who are not developpers but real world writers, you can obtain a source folder like this efolder

- 01 first page.xml
- default.xml
- welcome.xml

You can try a Sitemap logic like that

```
<map:match pattern="">
  <map:redirect-to uri="index.html"/>
</map:match>
<map:match pattern="**index.html">
  <map:generate src="{efolder}{1}index.xml"/>
  <!-- all your transformation logic -->
</map:match>
```

And you will have a "Resource Not Found" error, because your authors haven't yet understood that the default page of a folder should be an "index.xml". A work around could be

```
<map:match pattern="**index.html">
  <map:act type="exists" src="{efolder}{1}index.xml">
  <!-- all your transformation logic -->
  </map:act>
  <map:redirect-to uri="default.html"/>
</map:match>
<map:match pattern="**default.html">
  <map:act type="exists" src="{efolder}{1}default.xml">
  <!-- all your transformation logic -->
  </map:act>
</map:match>
```

No Cocoon Exception, but if there no default.xml, still 404.

So I implement something like that

```
<map:match type="regexp" pattern="^$|^(.*/)$">
  <map:act type="xsp" src="actions/redirect.xsp">
    <map:parameter name="folder" value="{context-attr:xfolio.efolder}{1}"/>
    <map:read src="cocoon:/{../1}{radical}.html"/>
  </map:act>
</map:match>
```

"redirect.xsp" is an action, you pass it parameters, it answer parameters, and also, success or failed. From the match, we can guess the folder requested. The regexp do the same as default wildcard (match="" or match="**/"). A piece of code is then able to find in a folder if there is "index.xml", or "default.xml", or if nothing of that, giving the first valid xml file (or whatever navigation logic you have). This for succes, and fail could be, this directory doesn't exist, let sitemap handle 404. The output is a "radical", a not yet precise spec but a filename without extension, probably more convenient to build an URI for your sitemap.

References

- Actions
- XSP
- XSP Actions
- redirections

<<FullSearch>>

Source

```
<xsp:page language="java" xmlns:xsp="http://apache.org/xsp">
  <!--
        This xsp don't use cocoon taglib action.xsl
        (less easier than a simple java line), but learn from it the code to put
-->
  <xsp:structure>
    <xsp:include>org.apache.cocoon.environment.Session</xsp:include>
    <xsp:include>java.util.*</xsp:include>
    <xsp:include>java.io.*</xsp:include>
    <xsp:include>org.apache.avalon.framework.context.ContextException</xsp:include>
    <xsp:include>org.apache.cocoon.generation.XfolioDirectoryGenerator</xsp:include>
    <xsp:include>org.apache.cocoon.environment.Redirector</xsp:include>
    <xsp:include>org.apache.cocoon.acting.ServerPagesAction</xsp:include>
    <xsp:include>java.util.Map</xsp:include>
  </xsp:structure>
  <!-- class variable, outside of the generate() method, begun by the root element <action/> -->
  <xsp:logic>
      private Redirector actionRedirector;
      private Map actionResultMap;
                </xsp:logic>
  <action>
    <xsp:logic>
      <!--
  if you are fed up of &lt; instead of < in your javacode
  remember XML/SGML spec and the <![CDATA[ ]]> sections
--><![CDATA[

  /* from the generated JAVA of the XSPAction logicsheet */
  this.actionRedirector = (Redirector)this.objectModel.get(ServerPagesAction.REDIRECTOR_OBJECT);
  this.actionResultMap = (Map)this.objectModel.get(ServerPagesAction.ACTION_RESULT_OBJECT);

  // output of the action. If it stay like that, action failed
  String radical="";


        String path=parameters.getParameter("folder", null);
        // For sitemap authors
  if (path == null) throw new ProcessingException ("Action, redirect.xsp, folder parameter is empty" );
        File folder=new File (path);
  if (!folder.isDirectory()) {
    //  throwing exception is a bad idea, failed action is a not so bad effect
    //  throw new ProcessingException ("Action, redirect.xsp : " + folder + " is not a directory" );
```

```
  }
  else {
    // when a page not found is provide, for example a not available language or an identifier witout extension
    String search=parameters.getParameter("radical", null);
    File[] files;
    files=folder.listFiles();

    /*
    logic is, scan all files, and keep the best as possible
    some more parameters may be useful

    For now, effect on lang is to get the last for index
    the first for first file
    */

    String candidate="";
    String extension="";
    String htmlisable="html sxw dbx jpg";
    String lang="";
    for (int i=0 ; i < files.length ; i++) {
      candidate=files[i].getName();
      int dotPos=candidate.lastIndexOf(".");
      extension=( dotPos > 0)?candidate.substring(dotPos+1):"";
      candidate=( dotPos > 0)?candidate.substring(0, dotPos):candidate;

      if (files[i].isDirectory()) {}
      else if (candidate.startsWith("_") || candidate.startsWith(".")) {}
      // first file
      else if ("".equals(radical) && htmlisable.indexOf(extension) != -1) {radical=candidate ;}
      // index
      else if (candidate.startsWith("index") && htmlisable.indexOf(extension) != -1) {
        radical=candidate;
      }
      // after index
      else if (radical.startsWith("index")) {break;}
    }

  }

        // give the path of generated file as a sitemap parameter
        this.actionResultMap.put("radical", radical);
        // if no radical found (probably folder doesn't exist), action fails
        try {
                if (!"".equals(radical)) this.objectModel.put(ServerPagesAction.ACTION_SUCCESS_OBJECT, Boolean.
TRUE);
                else this.objectModel.put(ServerPagesAction.ACTION_SUCCESS_OBJECT, Boolean.FALSE);
        }
        catch (Exception e) {this.objectModel.put(ServerPagesAction.ACTION_SUCCESS_OBJECT, Boolean.FALSE);}

        ]]></xsp:logic>
  </action>
</xsp:page>
```

Changes

- 2004-07-14 more detailes explanation --[FredericGlorieux FG]
- 2004-07-13 Creation --[FredericGlorieux FG]