# Annotated core sitemap.xmap

## Table of contents

## 1 sitemap.xmap

<?xml version="1.0"?> <!-- ==========================================
Default Forrest sitemap, defining the whole site. Delegates to the other *.xmap
files. See http://forrest.apache.org/docs/sitemap-ref.html $Revision: 1.12 $
========================================================== -->

### 1.1 Start of Sitemap

<map:sitemap xmlns:map="http://apache.org/cocoon/sitemap/1.0">
<map:components> <map:generators default="file"> <map:generator name="file"
src="org.apache.cocoon.generation.FileGenerator" /> <map:generator name="html"
src="org.apache.cocoon.generation.HTMLGenerator"> <jtidy-config>WEB-INF/
jtidy.properties</jtidy-config> </map:generator> <map:generator name="exception"
src="org.apache.cocoon.generation.ParseExceptionGenerator"/> <!-- <map:generator
name="html" src="org.apache.cocoon.generation.HTMLGenerator"> <jtidy-
config>jtidy.properties</jtidy-config> </map:generator> <map:generator name="directory"
src="org.apache.cocoon.generation.DirectoryGenerator" /> --> <map:generator name="notifier"
src="org.apache.cocoon.sitemap.NotifyingGenerator" /> <map:generator name="text2xml"
logger="sitemap.generator.textgenerator" src="org.apache.cocoon.generation.TextGenerator">
<parameter name="localizable" value="true"/> </map:generator> </map:generators>
<map:transformers default="xslt"> <!-- Add values to skinconf that need
extra processing like the color shades --> <map:transformer name="skinconf"
src="org.apache.forrest.conf.SkinconfTransformer"/> <!-- Generates @id attributes from <title>
strings --> <map:transformer name="idgen" src="org.apache.forrest.util.IdGeneratorTransformer">
<element>//*[local-name() = 'section']</element> <id>title/text()</id> </map:transformer>
<!-- Rewrites links, e.g. transforming href="site:index" to href="../index.html" --
> <!-- See http://forrest.apache.org/docs/sitemap-ref.html#linkrewriting_impl -->
<map:transformer name="linkrewriter" logger="sitemap.transformer.linkrewriter"
src="org.apache.cocoon.transformation.LinkRewriterTransformer"> <link-attrs>href src</
link-attrs> <schemes>site ext</schemes> <input-module name="site"> <input-module
name="linkmap"> <file src="{src}" reloadable="true" /> </input-module> <prefix>/
site//</prefix> <suffix>/@href</suffix> </input-module> <input-module name="ext">
<input-module name="linkmap"> <file src="{src}" reloadable="true" /> </input-module>
<prefix>/site/external-refs//</prefix> <suffix>/@href</suffix> </input-module> </
map:transformer> <map:transformer name="xpath" logger="sitemap.transformer.xpath"
src="org.apache.forrest.util.XPathTransformer" /> <map:transformer name="xslt"
src="org.apache.cocoon.transformation.TraxTransformer" logger="sitemap.transformer.xslt"
pool-max="32" pool-min="8" pool-grow="2"> <use-request-parameters>false</use-request-
parameters> <use-browser-capabilities-db>false</use-browser-capabilities-db> <use-deli>false</
use-deli> <transformer-factory>org.apache.xalan.processor.TransformerFactoryImpl</transformer-
factory> <!--<transformer-factory>net.sf.saxon.TransformerFactoryImpl</transformer-
factory>--> <!--<transformer-factory>com.icl.saxon.TransformerFactoryImpl</transformer-
factory>--> <!--<transformer-factory>org.apache.xalan.xsltc.trax.TransformerFactoryImpl</
transformer-factory>--> </map:transformer> <map:transformer name="xsltc"
src="org.apache.cocoon.transformation.TraxTransformer" logger="sitemap.transformer.xslt" pool-
max="32" pool-min="8" pool-grow="2"> <use-request-parameters>false</use-request-parameters>
<use-browser-capabilities-db>false</use-browser-capabilities-db> <use-deli>false</use-deli>
<transformer-factory>org.apache.xalan.xsltc.trax.TransformerFactoryImpl</transformer-factory>

</map:transformer> <map:transformer name="xslt-saxon" pool-grow="2" pool-max="32" pool-min="8" src="org.apache.cocoon.transformation.TraxTransformer"> <use-request-parameters>false</use-request-parameters> <use-browser-capabilities-db>false</use-browser-capabilities-db> <xslt-processor-role>saxon</xslt-processor-role> </map:transformer> <map:transformer name="xinclude" src="org.apache.cocoon.transformation.XIncludeTransformer" logger="sitemap.transformer.xinclude" pool-grow="2" pool-max="16" pool-min="2" /> <map:transformer name="cinclude" pool-grow="2" pool-max="16" pool-min="2" src="org.apache.cocoon.transformation.CIncludeTransformer" logger="sitemap.transformer.cinclude"/> <map:transformer name="pattern" src="org.apache.cocoon.transformation.PatternTransformer" logger="sitemap.transformer.pattern"> <parameter name="groups" value="true"/> </map:transformer> <map:transformer name="lexer" src="org.apache.cocoon.transformation.LexicalTransformer" logger="sitemap.transformer.lexer"> <parameter name="localizable" value="true"/> </map:transformer> <map:transformer name="parser" src="org.apache.cocoon.transformation.ParserTransformer" logger="sitemap.transformer.parser"> <parameter name="flatten" value="true"/> <parameter name="recovery" value="true"/> <parameter name="localizable" value="true"/> </map:transformer> </map:transformers> <map:readers default="resource"> <map:reader name="resource" src="org.apache.cocoon.reading.ResourceReader"/> </map:readers> <map:serializers default="html"> <map:serializer name="html" mime-type="text/html" src="org.apache.cocoon.serialization.HTMLSerializer"> <doctype-public>-//W3C//DTD HTML 4.01 Transitional//EN</doctype-public> <doctype-system>http://www.w3.org/TR/html4/loose.dtd</doctype-system> <encoding>UTF-8</encoding> </map:serializer> <map:serializer name="xml" mime-type="text/xml" src="org.apache.cocoon.serialization.XMLSerializer"/> <map:serializer name="xml-document" mime-type="text/xml" src="org.apache.cocoon.serialization.XMLSerializer"> <cdata-section-elements>source</cdata-section-elements> <doctype-public>-//APACHE//DTD Documentation V1.3//EN</doctype-public> <doctype-system>document-v13.dtd</doctype-system> </map:serializer> <map:serializer name="links" src="org.apache.cocoon.serialization.LinkSerializer"> <encoding>ISO-8859-1</encoding> </map:serializer> <map:serializer name="svgxml" src="org.apache.cocoon.serialization.XMLSerializer" mime-type="image/svg+xml"> <doctype-public>-//W3C//DTD SVG 1.0//EN</doctype-public> <doctype-system>http://www.w3.org/TR/2001/REC-SVG-20010904/DTD/svg10.dtd</doctype-system> </map:serializer> <map:serializer logger="sitemap.serializer.text" mime-type="text/plain" name="text" src="org.apache.cocoon.serialization.TextSerializer"> <encoding>UTF-8</encoding> </map:serializer> <!-- <map:serializer mime-type="application/x-shockwave-flash" name="swf" src="org.apache.cocoon.serialization.SWFSerializer"/> <map:serializer mime-type="application/msword" name="fo2rtf" src="org.apache.cocoon.serialization.RTFSerializer"/> --> </map:serializers> <map:matchers default="wildcard"> <map:matcher name="wildcard" src="org.apache.cocoon.matching.WildcardURIMatcher"/> <map:matcher name="regexp" src="org.apache.cocoon.matching.RegexpURIMatcher"/> </map:matchers> <map:selectors> <map:selector logger="sitemap.selector.exists" name="exists" src="org.apache.forrest.sourceexists.SourceExistsSelector" /> <map:selector logger="sitemap.selector.exception" name="exception" src="org.apache.cocoon.selection.ExceptionSelector"> <exception name="syntax" class="net.sourceforge.chaperon.process.ParseException"/> <exception class="java.lang.Throwable" unroll="true"/> </map:selector> </map:selectors> <map:pipes default="caching"> <map:pipe name="caching" src="org.apache.cocoon.components.pipeline.impl.CachingProcessingPipeline"/> <!-- <map:pipe name="noncaching" src="org.apache.cocoon.components.pipeline.impl.NonCachingProcessingPipeline"/> <map:pipe name="profile-caching" src="org.apache.cocoon.components.profiler.ProfilingCachingProcessingPipeline"/> <map:pipe

name="profile-noncaching"
src="org.apache.cocoon.components.profiler.ProfilingNonCachingProcessingPipeline"/> -->
</map:pipes> </map:components> <!-- NOTE: the 'links view' is no longer used to discover
a page's links. Instead of filterlinks.xsl, use cli.xconf include/exclude nodes to define which
pages to render. <map:views> <map:view name="links" from-position="last"> <map:transform
src="{forrest:stylesheets}/filterlinks.xsl"> <map:parameter name="ctxbasedir" value="{realpath:.}/"/
> </map:transform> <map:serialize type="links"/> </map:view> </map:views> --> <map:resources>
<map:resource name="skinit"> <map:select type="exists"> <map:when test="{project:skins-
dir}{forrest:skin}/xslt/html/{type}.xsl"> <map:transform src="{project:skins-dir}{forrest:skin}/
xslt/html/{type}.xsl"> <map:parameter name="notoc" value="{notoc}"/> <!-- For backwards-
compat with 0.2 - 0.4 skins --> <map:parameter name="isfaq" value="{notoc}"/> <map:parameter
name="nopdf" value="{nopdf}"/> <map:parameter name="path" value="{path}"/> <map:parameter
name="config-file" value="{project:skinconf}"/> </map:transform> </map:when> <map:otherwise>
<map:transform src="{forrest:context}/skins/{forrest:skin}/xslt/html/{type}.xsl"> <map:parameter
name="notoc" value="{notoc}"/> <!-- For backwards-compat with 0.2 - 0.4 skins --> <map:parameter
name="isfaq" value="{notoc}"/> <map:parameter name="nopdf" value="{nopdf}"/> <map:parameter
name="path" value="{path}"/> <map:parameter name="config-file" value="{project:skinconf}"/
> </map:transform> </map:otherwise> </map:select> <map:serialize/> </map:resource> </
map:resources>

### 1.2 Start of Pipelines

<!-- ========================== Pipelines ================================== -->
<map:pipelines> <map:pipeline type="caching" internal-only="true">

### 1.3 Test for First Pipeline

<map:match pattern="*.xlex"> <map:select type="exists"> <map:when test="resources/chaperon/
grammars/{1}.xlex"> <map:read src="resources/chaperon/grammars/{1}.xlex"/> </map:when>
<map:otherwise> <map:generate type="text2xml" src="{forrest:context}/resources/chaperon/
grammars/{1}.grm"/> <map:transform type="lexer" src="{forrest:context}/resources/chaperon/
grammars/grm.xlex"/> <map:transform type="parser" src="{forrest:context}/resources/chaperon/
grammars/grm.xgrm"/> <map:transform src="{forrest:context}/resources/chaperon/stylesheets/
text4regex.xsl"/> <map:transform type="lexer" src="{forrest:context}/resources/chaperon/grammars/
regex.xlex"/> <map:transform type="parser" src="{forrest:context}/resources/chaperon/grammars/
regex.xgrm"/> <map:transform src="{forrest:context}/resources/chaperon/stylesheets/grm2xlex.xsl"/>
<map:serialize type="xml"/> </map:otherwise> </map:select> </map:match> </map:pipeline>

### 1.4 Insertion Point for Project Sitemap

<!-- This is the user pipeline, that can answer requests instead of the Forrest one, or let requests
pass through. To take over the rendering of a file it must match the file name and path. To
take over the generation of the intermediate format, it must give Forrest the same filename but
ending with xml, and a DTD that Forrest recognizes. --> <map:pipeline internal-only="false">
<map:select type="exists"> <map:when test="{project:sitemap}"> <map:mount uri-prefix=""
src="{project:sitemap}" check-reload="yes" pass-through="true"/> </map:when> </map:select>
</map:pipeline> <map:pipeline internal-only="false"> <map:select type="exists"> <map:when
test="{project:temp-dir}/internal.xmap"> <map:mount uri-prefix="" src="{project:temp-dir}/
internal.xmap" check-reload="yes" pass-through="true"/> </map:when> </map:select> </
map:pipeline> <map:pipeline internal-only="false"> <map:match pattern="skinconf.xml">

<map:generate src="{project:skinconf}" /> <map:transform src="{forrest:stylesheets}/strip-doctype.xsl"/> <map:transform src="{forrest:stylesheets}/upgrade-skinconf.xsl"/> <map:select type="exists"> <map:when test="{project:skins-dir}{forrest:skin}/skinconf.xsl"> <map:transform src="{project:skins-dir}{forrest:skin}/skinconf.xsl"/> </map:when> </map:select> <map:select type="exists"> <map:when test="{forrest:context}/skins/{forrest:skin}/skinconf.xsl"> <map:transform src="{forrest:context}/skins/{forrest:skin}/skinconf.xsl"/> </map:when> </map:select> <map:transform src="{forrest:context}/skins/common/skinconf.xsl"/> <map:transform type="skinconf"/> <map:serialize type="xml" /> </map:match> <!-- Add some build information, which is added to the html head --> <map:match pattern="build-info"> <map:generate src="{project:temp-dir}/build-info.xml"/> <map:serialize type="xml"/> </map:match> </map:pipeline> <map:pipeline internal-only="false"> <!-- ========== --> <!-- SOURCE FORMATS --> <!-- Raw XML sources, typically doc-v12 format --> <!-- ========== --> <!-- http://forrest.apache.org/docs/sitemap-ref.html#source_pipelines -->

### 1.5 First Match for '**.xml'

<!-- Body content --> <map:match pattern="**.xml"> <map:match pattern="linkmap.xml"> <map:mount uri-prefix="" src="linkmap.xmap" check-reload="yes" /> </map:match> <map:match pattern="changes.xml"> <map:mount uri-prefix="" src="status.xmap" check-reload="yes" /> </map:match> <map:match pattern="todo.xml"> <map:mount uri-prefix="" src="status.xmap" check-reload="yes" /> </map:match> <map:match pattern="**dtdx.xml"> <map:mount uri-prefix="" src="dtd.xmap" check-reload="yes" /> </map:match> <map:match pattern="forrest-issues.xml"> <map:mount uri-prefix="" src="issues.xmap" check-reload="yes" /> </map:match> <map:match pattern="**faq.xml"> <map:mount uri-prefix="" src="faq.xmap" check-reload="yes" /> </map:match> <map:match pattern="community/**index.xml"> <map:mount uri-prefix="" src="forrest.xmap" check-reload="yes" /> </map:match> <map:match pattern="community/**.xml"> <map:mount uri-prefix="" src="revisions.xmap" check-reload="yes" /> </map:match> <!-- wholesite is preferred; site is here for compatibility --> <map:match pattern="wholesite.xml"> <map:mount uri-prefix="" src="aggregate.xmap" check-reload="yes" /> </map:match> <map:match pattern="site.xml"> <map:mount uri-prefix="" src="aggregate.xmap" check-reload="yes" /> </map:match> <!-- Lucene index update and search --> <map:match pattern="lucene-*.xml"> <map:mount uri-prefix="" src="search.xmap" check-reload="yes"/> </map:match> <!-- Default source types --> <map:mount uri-prefix="" src="forrest.xmap" check-reload="yes" /> </map:match> <!-- Menu content --> <map:match pattern="abs-menulinks"> <map:mount uri-prefix="" src="menu.xmap" check-reload="yes" /> </map:match> <map:match pattern="**menulinks-*"> <map:mount uri-prefix="" src="menu.xmap" check-reload="yes" /> </map:match> <!-- Link maps --> <map:match pattern="abs-linkmap"> <map:mount uri-prefix="" src="linkmap.xmap"/> </map:match> <map:match pattern="**linkmap-*"> <map:match pattern="linkmap-wholesite.*"> <map:mount uri-prefix="" src="aggregate.xmap" check-reload="yes" /> </map:match> <map:match pattern="linkmap-site.*"> <map:mount uri-prefix="" src="aggregate.xmap" check-reload="yes" /> </map:match> <map:mount uri-prefix="" src="linkmap.xmap" check-reload="yes" /> </map:match> </map:pipeline> <!-- ========== --> <!-- INTERMEDIATE FORMATS --> <!-- Tabs, menus and body HTML. --> <!-- Called from output format pipelines --> <!-- ========== --> <!-- http://forrest.apache.org/docs/sitemap-ref.html#intermediate_pipelines --> <map:pipeline internal-only="false"> <!-- External matches --> <!-- (HTML rendered directly from special formats) --

> <map:match pattern="**body-faq.html"> <map:mount uri-prefix="" src="faq.xmap" check-reload="yes" /> </map:match>

### 1.6 First Match for '**body-*.html'

<map:match pattern="**body-*.html"> <map:select type="exists"> <map:when test="{project:content.xdocs}{1}{2}.ehtml"> <map:generate src="{project:content.xdocs}{1}{2}.ehtml" /> <map:transform src="{forrest:stylesheets}/html2htmlbody.xsl" /> <map:transform type="linkrewriter" src="cocoon:/{1}linkmap-{2}.html"/> <map:transform src="resources/stylesheets/declare-broken-site-links.xsl" /> <map:serialize type="xml" /> </map:when> </map:select> </map:match>

### 1.7 Second Match for '**body-*.html'

<!-- Default matches --> <!-- (HTML rendered from doc-v11 intermediate format --> <map:match pattern="**body-*.html"> <map:generate src="cocoon:/{1}{2}.xml"/>

### 1.8 Returning to the '**body-*.html' Pipeline

<map:transform type="idgen"/> <map:transform type="xinclude"/> <map:transform type="linkrewriter" src="cocoon:/{1}linkmap-{2}.html"/> <map:transform src="resources/stylesheets/declare-broken-site-links.xsl" /> <map:call resource="skinit"> <map:parameter name="type" value="document2html"/> <map:parameter name="path" value="{1}{2}.html"/> <map:parameter name="notoc" value="false"/> </map:call> </map:match> <map:match pattern="**menu-*.html"> <map:generate src="cocoon:/{1}book-{2}.html"/> <map:transform type="linkrewriter" src="cocoon:/{1}linkmap-{2}.html"/> <map:transform src="resources/stylesheets/declare-broken-site-links.xsl" /> <map:call resource="skinit"> <map:parameter name="type" value="book2menu"/> <map:parameter name="path" value="{1}{2}.html"/> </map:call> </map:match> <map:match pattern="**tab-*.html"> <map:mount uri-prefix="" src="tabs.xmap" check-reload="yes" /> </map:match> <map:match pattern="**i18n-*.html"> <map:mount uri-prefix="" src="i18n.xmap" check-reload="yes" /> </map:match> <map:match pattern="**book-*.html"> <map:mount uri-prefix="" src="menu.xmap" check-reload="yes" /> </map:match> </map:pipeline> <!-- ========================================================= --> <!-- OUTPUT FORMATS --> <!-- Serves content directly to the user --> <!-- +=========================================================+ --> <!-- http://forrest.apache.org/docs/sitemap-ref.html#output_pipelines --> <map:pipeline internal-only="false"> <map:select type="exists"> <map:when test="{project:temp-dir}/output.xmap"> <map:mount uri-prefix="" src="{project:temp-dir}/output.xmap" check-reload="yes" pass-through="true"/> </map:when> </map:select> </map:pipeline> <map:pipeline internal-only="false"> <map:match pattern="*.html"> <map:aggregate element="site"> <map:part src="cocoon:/skinconf.xml"/> <map:part src="cocoon:/build-info"/> <map:part src="cocoon:/tab-{0}"/> <map:part src="cocoon:/menu-{0}"/> <map:part src="cocoon:/body-{0}"/> </map:aggregate> <map:call resource="skinit"> <map:parameter name="type" value="site2xhtml"/> <map:parameter name="path" value="{0}"/> </map:call> </map:match>

### 1.9 First Match for "**/*.html"

<map:match pattern="**/*.html"> <map:aggregate element="site"> <map:part src="cocoon:/skinconf.xml"/> <map:part src="cocoon:/build-info"/> <map:part src="cocoon:/{1}/tab-{2}.html"/> <map:part src="cocoon:/{1}/menu-{2}.html"/> <map:part src="cocoon:/{1}/body-{2}.html"/> </map:aggregate> <map:call resource="skinit"> <map:parameter name="type" value="site2xhtml"/>

<map:parameter name="path" value="{0}"/> </map:call> </map:match> <map:match type="regexp" pattern="^.+$"> <map:select type="exists"> <map:when test="{project:content.xdocs}/{0}"> <map:mount uri-prefix="" src="raw.xmap" check-reload="yes" /> </map:when> </map:select> </map:match> <!-- generate faq.fo specially --> <map:match pattern="**faq.fo"> <map:mount uri-prefix="" src="faq.xmap" check-reload="yes" /> </map:match> <!-- generate .fo from .xml --> <map:match type="regexp" pattern="^(.*?)([^/]*).fo$"> <map:select type="exists"> <map:when test="{project:content.xdocs}{1}{2}.fo"> <map:generate src="{project:content.xdocs}{1}{2}.fo"/ > </map:when> <map:otherwise> <map:aggregate element="site"> <map:part src="cocoon:/ skinconf.xml"/> <map:part src="cocoon:/{1}{2}.xml"/> </map:aggregate> <!-- <map:transform type="idgen"/> --> <map:transform type="xinclude"/> <map:transform type="linkrewriter" src="cocoon://{1}linkmap-{2}.fo"/> <map:select type="exists"> <map:when test="{project:skins-dir}{forrest:skin}/xslt/fo/document2fo.xsl"> <map:transform src="{project:skins-dir}{forrest:skin}/ xslt/fo/document2fo.xsl"> <map:parameter name="imagesdir" value="{project:resources.images}/"/ > <map:parameter name="xmlbasedir" value="{project:content.xdocs}{1}"/> </map:transform> </map:when> <map:when test="{forrest:context}/skins/{forrest:skin}/xslt/fo/document2fo.xsl"> <map:transform src="{forrest:context}/skins/{forrest:skin}/xslt/fo/document2fo.xsl"> <map:parameter name="imagesdir" value="{project:resources.images}/"/> <map:parameter name="xmlbasedir" value="{project:content.xdocs}{1}"/> </map:transform> </map:when> <map:otherwise> <map:transform src="{forrest:context}/skins/common/xslt/fo/document2fo.xsl"> <map:parameter name="imagesdir" value="{project:resources.images}/"/> <map:parameter name="xmlbasedir" value="{project:content.xdocs}{1}"/> </map:transform> </map:otherwise> </map:select> </map:otherwise> </map:select> <map:serialize type="xml"/> </map:match> <map:match type="regexp" pattern="^(.*?)([^/]*).svg$"> <map:generate src="cocoon:/{1} {2}.xml"/> <!-- <map:transform type="idgen"/> --> <map:transform type="xinclude"/> <map:transform type="linkrewriter" src="cocoon://{1}linkmap-{2}.svg"/> <map:transform src="resources/stylesheets/declare-broken-site-links.xsl" /> <map:select type="exists"> <map:when test="{project:skins-dir}{forrest:skin}/xslt/svg/document2svg.xsl"> <map:transform src="{project:skins-dir}{forrest:skin}/xslt/svg/document2svg.xsl"> <map:parameter name="imagesdir" value="{project:resources.images}/"/> <map:parameter name="xmlbasedir" value="{project:content.xdocs}{1}"/> </map:transform> </map:when> <map:when test="{forrest:context}/skins/{forrest:skin}/xslt/svg/document2svg.xsl"> <map:transform src="{forrest:context}/skins/{forrest:skin}/xslt/svg/document2svg.xsl"> <map:parameter name="imagesdir" value="{project:resources.images}/"/> <map:parameter name="xmlbasedir" value="{project:content.xdocs}{1}"/> </map:transform> </map:when> <map:otherwise> <map:transform src="{forrest:context}/skins/common/xslt/svg/document2svg.xsl"> <map:parameter name="imagesdir" value="{project:resources.images}/"/> <map:parameter name="xmlbasedir" value="{project:content.xdocs}{1}"/> </map:transform> </map:otherwise> </map:select> <map:serialize type="svgxml"/> </map:match> <map:match pattern="**changes.rss"> <map:mount uri-prefix="" src="status.xmap" check-reload="yes" /> </map:match> <map:match pattern="profiler"> <map:mount uri-prefix="" src="profiler.xmap" check-reload="yes" /> </map:match> <map:match pattern="**.lucene"> <map:mount uri-prefix="" src="search.xmap" check-reload="yes" /> </ map:match> <map:select type="exists"> <map:when test="{project:temp-dir}/resources.xmap"> <map:mount uri-prefix="" src="{project:temp-dir}/resources.xmap" check-reload="yes" pass-through="true"/> </map:when> </map:select> <map:match pattern="**.js"> <map:mount uri-prefix="" src="resources.xmap" check-reload="yes" /> </map:match> <map:match pattern="**.css"> <map:mount uri-prefix="" src="resources.xmap" check-reload="yes" /> </map:match> <map:match pattern="**images**"> <map:mount uri-prefix="" src="resources.xmap" check-reload="yes" /> </map:match> <map:match pattern="**.png"> <map:mount uri-prefix="" src="resources.xmap" check-reload="yes" /> </map:match> <map:match pattern="**.ico"> <map:mount uri-

prefix="" src="resources.xmap" check-reload="yes" /> </map:match> </map:pipeline> <!-- ========== --> <!-- REDIRECTS --> <!-- ========== --> <map:pipeline internal-only="false"> <map:match pattern=""> <map:redirect-to uri="index.html" /> </map:match> <map:match type="regexp" pattern="^.+/$"> <map:redirect-to uri="index.html"/> </map:match> </map:pipeline> </map:pipelines> </map:sitemap>