DecoratorMappers

When a page has been parsed, it then has to be mapped to a decorator. This mapping is performed by a chain of DecoratorMappers (referred to as mappers from here on).

For each request, the first mapper in the chain is asked which decorator should be used. It is passed across a reference to the Page object and HttpServletRequest. It returns either a Decorator object, if it knows which decorator to be used, or null. If null is returned, the next mapper in the chain is queried. This whole process is repeated until there are no more mappers in the chain, or one of the mappers returns a valid decorator. If no mappers return a decorator, the page is not decorated at all and served in its original state.

This way the mappers are chained together and queried is known as the Chain of Responsibility design pattern.

Examples of mappers:

- Determine decorator based on path of requested page.
- Use different decorators based on time, locale or browser.
- Use simplified decorators for search-engine robots.
- Switch decorators based on a URL parameter, request attribute or meta-tag.
- Use custom decorators based on user's saved settings...

The main implementation of DecoratorMapper is ConfigDecoratorMapper which reads the decorators and mappings from /WEB-INF/decorators.xml. The appropriate decorator is then applied depending on the URL pattern.

DecoratorMappers are simple to write and the distribution includes some samples that demonstrate how to write them and how flexible they can be. These are:

DecoratorMapper	Description
AgentDecoratorMapper	Can determine the user-agent (i.e. web-browser) requesting a page, and map to a suitable Decorator.
ConfigDecoratorMapper	Default implementation of DecoratorMapper. Reads decorators and mappings from the config property (default '/WEB-INF/decorators.xml').
CookieDecoratorMapper	Will map a suitable decorator based on a cookie value.
EnvEntryDecoratorMapper	Allows the reference to a web-app environment entry for the decorator name, and falls back to ConfigDecoratorMapper's behavior if no matching environment entry is found.
FileDecoratorMapper	Will treat the name of the decorator as a file-name to use (in the context of the web-app).
FrameSetDecoratorMapper	Will use the specified decorator when the Page is an instance of HTMLPage and isFrameSet() returns true. The name of this decorator should be supplied in the decorator property - if no decorator property is supplied, no decorator is applied to frame based pages.
InlineDecoratorMapper	Used to determine the correct Decorator when using inline decorators.
LanguageDecoratorMapper	Can determine the preferred language set in the browser requesting a page, and map to a suitable Decorator (using the "Accept-Language" HTTP header).
PageDecoratorMapper	The actual Page determines the Decorator to be used. The 'meta.decorator' and 'decorator' properties of the page are accessed and if any of them contain the name of a valid Decorator, that Decorator shall be applied.

ParameterDecoratorMapper	Will choose the decorator based on request parameters. The ParameterDecoratorMapper is configured via three properties. decorator.parameter - the parameter which contains the name of the decorator which will be mapped. The default is "decorator". For example if decorator.parameter is "foobar" then myurl.jsp?foobar=mydecorator will map to the decorator named "mydecorator". You can also supply an optional 'confirmation parameter'. The decorator will only be mapped if the parameter named parameter.name is in the request URI and the value of that parameter is equal to the parameter.value property. For example assuming parameter.name=confirm and parameter.value=true the URI myurl.jsp?decorator=mydecorator&confirm=true will map the decorator mydecorator. where as the URIs myurl.jsp?decorator=mydecorator and myurl.jsp?decorator=mydecorator&confirm=false will not return any decorator.
SessionDecoratorMapper	Will look at a session attribute to find the name of an appropriate decorator to use. If the session attribute is present, the mapper will not do anything and allow the next mapper in the chain to select a decorator. By default, it will look at the 'decorator' session attribute, however this can be overriden by configuring the mapper with a 'decorator.parameter' property.
PrintableDecoratorMapper	Will check to see whether 'printable=true' is supplied as a request parameter and if so, use the specified decorator instead. The name of this decorator should be supplied in the decorator property.
RobotDecoratorMapper	Will use the specified decorator when the requester is identified as a robot (also known as spider, crawler, ferret) of a search engine. The name of this decorator should be supplied in the decorator property.

An example of a custom DecoratorMapper could be one that displays different Decorators based on time (e.g. morning, afternoon, Christmas, etc).

Custom Mapper Configuration

To be able to specify which mappers will be applied to a request, create the file [web-app]/WEB-INF/sitemesh.xml that contains the following:

```
<sitemesh>
    <property name="decorators-file" value="/WEB-INF/decorators.xml" />
    <excludes file="${decorators-file}" />

    <page-parsers>
        <parser content-type="text/html"
            class="com.opensymphony.module.sitemesh.parser.HTMLPageParser" />
        <parser content-type="text/html;charset=ISO-8859-1"
            class="com.opensymphony.module.sitemesh.parser.HTMLPageParser" />
    </page-parsers>

    <decorator-mappers>
        <mapper class="com.opensymphony.module.sitemesh.mapper.ConfigDecoratorMapper">
            <param name="config" value="${decorators-file}" />
        </mapper>
    </decorator-mappers>
</sitemesh>
```

In this example, the only mapper that will be applied is the ConfigDecoratorMapper, and that will only be applied to responses of type text/html or text/html;charset=ISO-8859-1. Responses of any other content type (eg image/gif) will be ignored by Sitemesh. Additionally, any files that match a pattern specified in the excludes file (in this case '/WEB-INF/decorators.xml') will not be touched by Sitemesh.

The excludes file points to an XML file that contains an <excludes /> block similar to the following:

```
<decorators>
    <excludes>
        <pattern>/plainPage.jsp</pattern>
        <pattern>/plain/*.jsp</pattern>
    </excludes>
</decorators>
```

The above example would prevent /plainPage.jsp and any JSP pages in the /plain directory from being decorated. (Note that the pattern matching follows exactly the same rules as the decorator mappings used by the ConfigDecoratorMapper.)

Typically the <excludes /> block is just added at the start of the decorators.xml file, however this is not a requirement and any other XML file can be specified instead by changing the excludes file specified in sitemesh.xml. This might be useful if for example the ConfigDecoratorMapper is not being used in your deployment.

Note that preventing pages from being decorated by adding them to the excludes list superceeds, and is a better approach than, the old method of mapping the pages to a non-existent decorator. This is because when pages were mapped to a non-existent decorator they were still buffered internally by Sitemesh. By using the exclude list Sitemesh will not let the request pass straight through to the servlet container without any buffering.

Default Mapper Configuration

If sitemesh.xml is not found in the WEB-INF dir, the default mapper configuration will be used. The default mapper configuration is defined in sitemesh-default.xml (packaged inside the jar) and consists of the following mappers:

- PageDecoratorMapper
- FrameSetDecoratorMapper
- PrintableDecoratorMapper
- FileDecoratorMapper
- ConfigDecoratorMapper

By default only content of type text/html will be decorated by Sitemesh.