## Sitemesh

Sitemesh是一个网页布局和修饰的框架，利用它可以将网页的内容和页面结构分离，以达到页面结构共享的目的。

通俗的理解就是，sitemesh把网页中变化的和不变化的分离开来，用不变的去装饰各种变化的内容。从而使页面具有统一的布局，而且方便页面的管理。不变的页面称之为装饰页面，内容变化的页面称之为被装饰页面。

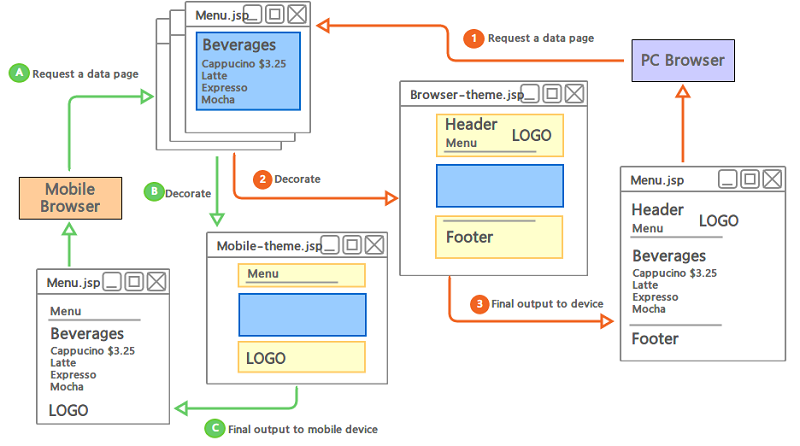
装饰的页面一般包括：页面标签、头部、底部、主体以及公共的css、js。

被装饰页面只需要写它自己需要的内容就可以了。

根据页面需要，装饰页面可以有多个，被装饰页面也可以有不被装饰而保持自己风格的选择，这只需要在配置文件中配置一下的就可以了。

Sitemesh是由一个基于Web页面布局、装饰及与现存Web应用整合的框架。它能帮助我们再由大量页面工程的项目中创建一致的页面布局和外观，如一 致的导航条、一致的banner、一致的版权等。它不仅能处理动态的内容，如JSP、PHP、ASP、CGI等产生的内容，还能处理静态的内容，比如 HTML的内容，使得它的内容也符合你的页面结构的要求。甚至它能像include那样将HTML文件作为一个面板的形式嵌入到别的文件中去。通常我们都是用include标签在每个jsp页面中来不断的包含各种header, stylesheet, scripts and footer，现在，在sitemesh的帮助下，我们可以开心的删掉他

SiteMesh3.0运行环境：servlet、jdk



最新版本：3.0.0-SNAPSHOT

① GitHub 地址：<https://github.com/sitemesh/sitemesh3>

② maven：

1 <dependency>

2 <groupId>org.sitemesh</groupId>

3 <artifactId>sitemesh</artifactId>

4 <version>3.0.0</version>

5 </dependency>

3 . 配置 Sitemesh 3 过滤器

在 web.xml 中添加 Sitemesh Filter：

1 <web-app>

2

3 ...

4

5 <filter>

6 <filter-name>sitemesh</filter-name>

7 <filter-class>org.sitemesh.config.ConfigurableSiteMeshFilter</filter-class>

8 </filter>

9 <filter-mapping>

10 <filter-name>sitemesh</filter-name>

11 <url-pattern>/\*</url-pattern>

12 </filter-mapping>

13

14 </web-app>

4 . 准备两个页面：demo.html 和 decorator.html

① demo.html - “被装饰的页面”，实际要呈现的内容页。

1 <!DOCTYPE html>

2 <html>

3 <head>

4 <title>内容页的标题</title>

5 </head>

6 <body>

7 内容页的body部分

8 </body>

9 </html>

② decorator.html - “装饰页面”，所谓的“母版页”。

1 <!DOCTYPE html>

2 <html>

3 <head>

4 <title>

5 <sitemesh:write property='title' /> - ltcms

6 </title>

7 <sitemesh:write property='head' />

8 </head>

9 <body>

10 <header>header</header>

11 <hr />

12 demo.html的title将被填充到这儿：

13 <sitemesh:write property='title' /><br />

14 demo.html的body将被填充到这儿：

15 <sitemesh:write property='body' />

16 <hr />

17 <footer>footer</footer>

18 </body>

19 </html>

**注意“/\*”其中/表示根目录，\*表示全部文件**

5 . 添加 /WEB-INF/sitemesh3.xml

1 <?xml version="1.0" encoding="UTF-8"?>

2 <sitemesh>

3 <!-- 指明满足“/\*”的页面，将被“/WEB-INF/views/decorators/decorator.html”所装饰 -->

4 <mapping path="/\*" decorator="/WEB-INF/views/decorators/decorator.html" />

5

6 <!-- 指明满足“/exclude.jsp\*”的页面，将被排除，不被装饰 -->

7 <mapping path="/exclude.jsp\*" exclue="true" />

8 </sitemesh>

**（如果有多个不被装饰的那么）这儿可以写为 path1= 被除外的**

**path2=**

**Path3=**

6 . 运行效果

访问 demo.html 页面，实际效果如下：

7 . sitemesh3.xml 配置详解

1 <sitemesh>

2 <!--默认情况下，

3 sitemesh 只对 HTTP 响应头中 Content-Type 为 text/html 的类型进行拦截和装饰，

4 我们可以添加更多的 mime 类型-->

5 <mime-type>text/html</mime-type>

6 <mime-type>application/vnd.wap.xhtml+xml</mime-type>

7 <mime-type>application/xhtml+xml</mime-type>

8 ...

9

10 <!-- 默认装饰器，当下面的路径都不匹配时，启用该装饰器进行装饰 -->

11 <mapping decorator="/default-decorator.html"/>

12

13 <!-- 对不同的路径，启用不同的装饰器 -->

14 <mapping path="/admin/\*" decorator="/another-decorator.html"/>

15 <mapping path="/\*.special.jsp" decorator="/special-decorator.html"/>

16

17 <!-- 对同一路径，启用多个装饰器 -->

18 <mapping>

19 <path>/articles/\*</path>

20 <decorator>/decorators/article.html</decorator>

21 <decorator>/decorators/two-page-layout.html</decorator>

22 <decorator>/decorators/common.html</decorator>

23 </mapping>

24

25 <!-- 排除，不进行装饰的路径 -->

26 <mapping path="/javadoc/\*" exclue="true"/>

27 <mapping path="/brochures/\*" exclue="true"/>

28

29 <!-- 自定义 tag 规则 -->

30 <content-processor>

31 <tag-rule-bundle class="com.something.CssCompressingBundle" />

32 <tag-rule-bundle class="com.something.LinkRewritingBundle"/>

33 </content-processor>

34 ...

35

36 </sitemesh>

8 . 自定义 tag 规则

Sitemesh 3 默认只提供了 body，title，head 等 tag 类型，我们可以通过实现 TagRuleBundle 扩展自定义的 tag 规则：

1 public class MyTagRuleBundle implements TagRuleBundle {

2 @Override

3 public void install(State defaultState, ContentProperty contentProperty,

4 SiteMeshContext siteMeshContext) {

5 defaultState.addRule("myHeader", new ExportTagToContentRule(contentProperty.getChild("myHeader"), false));

6

7 }

8

9 @Override

10 public void cleanUp(State defaultState, ContentProperty contentProperty,

11 SiteMeshContext siteMeshContext) {

12 }

13 }

最后在 sitemesh3.xml 中配置即可：

1 <content-processor>

2 <tag-rule-bundle class="com.lt.common.ext.sitemesh3.MyTagRuleBundle" />

3 </content-processor>