Struts2 Tiles 集成

在本章中，我们会学习到将 Tiles 框架与 Struts2 集成所涉及的步骤。Apache Tiles 是一个模板框架，用于简化 Web 应用程序用户界面的开发。

首先，我们需要从 Apache Tiles 网站下载 tiles jar 文件。你需要将以下 jar 文件添加到项目的类路径。

- tiles-api-x.y.z.jar
- tiles-compat-x.y.z.jar
- tiles-core-x.y.z.jar
- tiles-jsp-x.y.z.jar
- tiles-servlet-x.y.z.jar

除了上面的，我们必须从 **WEB-INF/lib** 中复制以下 jar 文件。

- commons-beanutils-x.y.z.jar
- commons-digester-x.y.z.jar
- struts2-tiles-plugin-x.y.z.jar

现在，让我们设置 Struts-Tiles 集成的 **web.xml**，如下所示。这里有两个要点要注意。首先，我们需要告诉 tile，在哪里可以找到 tiles 的配置文件 **tiles.xml**。在我们的例子中，它将在 **/WEB-INF** 文件夹下。然后，我们需要初始化 Struts2 下载附带的 Tiles 监听器。

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

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

   xmlns="http://java.sun.com/xml/ns/javaee"

   xmlns:web="http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
```

```
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
id="WebApp_ID" version="2.5">
<display-name>Struts2Example15</display-name>

<context-param>
<param-name>
   org.apache.tiles.impl.BasicTilesContainer.DEFINITIONS_CONFIG
</param-name>
<param-value>
   /WEB-INF/tiles.xml
</param-value>
</context-param>

<listener>
<listener-class>
   org.apache.struts2.tiles.StrutsTilesListener
</listener-class>
</listener>

<filter>
<filter-name>struts2</filter-name>
<filter-class>
org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter
</filter-class>
</filter>
```

```
   <filter-mapping>
      <filter-name>struts2</filter-name>
      <url-pattern>/*</url-pattern>
   </filter-mapping>

   <welcome-file-list>
      <welcome-file>index.jsp</welcome-file>
   </welcome-file-list>
</web-app>
```

接下来在 /WEB-INF 文件夹下创建 **tiles.xml**，内容如下：

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

<!DOCTYPE tiles-definitions PUBLIC
   "-//Apache Software Foundation//DTD Tiles Configuration 2.0//EN"
   "http://tiles.apache.org/dtds/tiles-config_2_0.dtd">

<tiles-definitions>

   <definition name="baseLayout" template="/baseLayout.jsp">
      <put-attribute name="title"  value="Template"/>
      <put-attribute name="banner" value="/banner.jsp"/>
      <put-attribute name="menu"   value="/menu.jsp"/>
      <put-attribute name="body"   value="/body.jsp"/>
      <put-attribute name="footer"   value="/footer.jsp"/>
```

```
    </definition>


    <definition name="tiger" extends="baseLayout">
        <put-attribute name="title"  value="Tiger"/>
        <put-attribute name="body"   value="/tiger.jsp"/>
    </definition>


    <definition name="lion" extends="baseLayout">
        <put-attribute name="title"  value="Lion"/>
        <put-attribute name="body"   value="/lion.jsp"/>
    </definition>


</tiles-definitions>
```

接下来，我们在 **baseLayout.jsp** 中定义一个基本的 skeleton 布局。它有五个可重用/可覆盖区域。即 **title**，**banner**，**menu**，**body** 和 **footer**。我们提供 baseLayout 的默认值，然后创建从默认布局扩展的两个自定义。tiger 布局类似于基本布局，除了它使用 **tiger.jsp** 作为其 body 和文本“Tiger”作为 title。类似地，lion 布局也类似于基本布局，除了它使用 **lion.jsp** 作为其 body 和文本“Lion”作为 title。

让我们看看各个 jsp 文件。以下是 **baseLayout.jsp** 文件的内容：

```
<%@ taglib uri="http://tiles.apache.org/tags-tiles" prefix="tiles"%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<html>
```

```
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title><tiles:insertAttribute name="title" ignore="true" />
</title>
</head>

<body>
   <tiles:insertAttribute name="banner" /><br/>
   <hr/>
   <tiles:insertAttribute name="menu" /><br/>
   <hr/>
   <tiles:insertAttribute name="body" /><br/>
   <hr/>
   <tiles:insertAttribute name="footer" /><br/>
</body>
</html>
```

这里我们只是把一个有 tiles 属性的基本的 HTML 页面放在一起。将 tile 属性插入到需要用到它的地方。接下来，让我们创建具有以下内容的 **banner.jsp** 文件：

```
<img src="//atts.w3cschool.cn/attachments/tuploads/struts_2/tp-logo.gif"/>
```

menu.jsp 文件将有以下内容，它们是 TigerMenu.action 和 LionMenu.action 的链接。

```
<%@taglib uri="/struts-tags" prefix="s"%>

<a href="<s:url action="tigerMenu"/>" Tiger</a><br>
```

```
<a href="<s:url action="lionMenu"/>" Lion</a><br>
```

lion.jsp 文件将具有以下内容：

```
<img src="//atts.w3cschool.cn/attachments/tuploads/struts_2/Lion.jpg"/>

The lion
```

tiger.jsp 文件将具有以下内容：

```
<img src="//atts.w3cschool.cn/attachments/tuploads/struts_2/tiger.jpg"/>

The tiger
```

接下来，创建一个 action 类文件 **MenuAction.java**，它包含以下内容：

```
package cn.w3cschool.struts2;


import com.opensymphony.xwork2.ActionSupport;


public class MenuAction extends ActionSupport {

   public String tiger() { return "tiger"; }

   public String lion() { return "lion"; }


}
```

这是一个非常直接的类。我们声明了两种方法 tiger() 和 lion() ，它们分别返回 tiger 和 lion 作为结果。让我们把它们放在 **struts.xml** 文件中：

```
<!DOCTYPE struts PUBLIC

"-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
```

```
"http://struts.apache.org/dtds/struts-2.0.dtd">


<struts>
   <package name="default" extends="struts-default">
      <result-types>
         <result-type name="tiles"
         class="org.apache.struts2.views.tiles.TilesResult" />
      </result-types>

      <action name="*Menu" method="{1}"
         class="cn.w3cschool.struts2.MenuAction">
         <result name="tiger" type="tiles">tiger</result>
         <result name="lion" type="tiles">lion</result>
      </action>

   </package>
</struts>
```

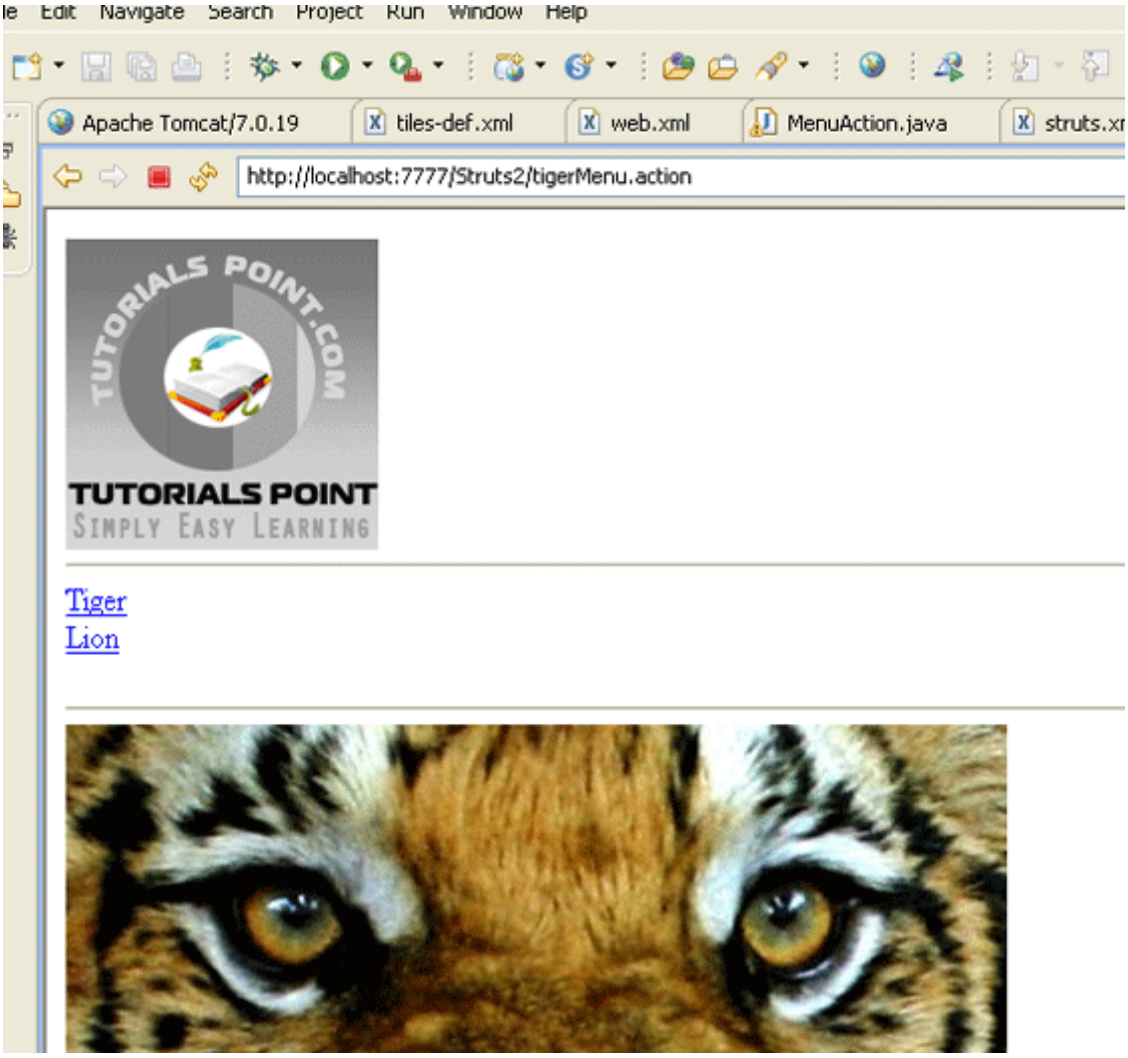
让我们回顾一下我们在上面的文件中做的。首先，我们声明了一种称为“tiles”的新结果类型，因为我们现在使用 tile 而不是普通的 jsp 作为视图技术。Struts2 支持 Tiles View 结果类型，因此我们创建了结果类型“tiles”作为“org.apache.struts2.view.tiles.TilesResult”类的结果类型。

接下来，我们要说的是，如果请求是 /tigerMenu.action，用户跳到 tiger 标题页面，如果请求是 /lionMenu.action，用户跳到 lion 标题页面。

我们使用一些正则表达式来实现这个。在 action 定义中，我们说任何匹配“*Menu”模式的东

西都会被这个 action 处理。匹配方法将在 MenuAction 类中调用。也就是说，tigerMenu.action 将调用 tiger()和 lionMenu.action 将调用 lion()。然后，我们需要将结果映射到适当的标题页面。

现在，右键单击项目名称，然后单击“**Export**”> “**WAR File**”以创建 WAR 文件。然后在 Tomcat 的 webapps 目录中部署 WAR 文件。最后，启动 Tomcat 服务器并尝试访问 URL http://localhost:8080/HelloWorldStruts2/tigerMenu.jsp，将显示以下界面：



同样，如果你转到 lionMenu.action 页面，你会看到 lion 页面使用相同的标题布局。