Struts2常用配置

标签配置解释

- <package>标签：包定义，任何action配置都必须放到包中，和java包没有半毛钱关系
  - name属性：给包起名，这个名称可以用来让其他包继承
  - namespace属性：命名空间名称，这个是访问某个包中Action的url前缀，例如访问本例action就得写成/test/helloWorld.action
  - extends属性：继承的父包名，一般直接继承struts-default，在这个包中实现了Struts2的基本功能，可以查看struts2-core-2.x.x.jar文件中的struts-default.xml的相关设置
- <action>标签：设置Action
  - name属性：Action名称，用来访问Action
  - class属性：Action的完整类型名称
  - method属性：用来指定当访问Action时执行的方法，不设置此属性默认就是execute
- <result>标签：设置转发视图，相当于struts1的forward
  - name属性：设置转发名称，不写默认就是success
  - type属性：此处不做实验，表示result类型，例如转发还是重定向，默认就是请求转发，注意如果是转发到Action需要做单独设置，转发到jsp则默认即可

Action中result的各种转发类型

- result配置类似于struts1中的forward，但struts2中提供了多种结果类型，常用的类型有： dispatcher(默认值)、 redirect 、 redirectAction 、 plainText。

- 在result中还可以使用${属性名}表达式访问action中的属性，表达式里的属性名对应action中的属性。如下：
- <result type="redirect">/view.jsp?id=${id}</result>

- 下面是redirectAction 结果类型的例子，如果重定向的action中同一个包下：
- <result type="redirectAction">helloworld</result>
- 如果重定向的action在别的命名空间下：
- <result type="redirectAction">
- <param name="actionName">helloworld</param>
- <param name="namespace">/test</param>
- </result>
- plaintext:显示原始文件内容，例如：当我们需要原样显示jsp文件源代码 的时候，我们可以使用此类型。
- <result name="source" type="plainText ">
- <param name="location">/xxx.jsp</param>
- <param name="charSet">UTF-8</param><!-- 指定读取文件的编码 -->
- </result>

多个Action共享一个视图--全局result配置

- 当多个action中都使用到了相同视图，这时我们应该把result定义为全局视图。struts1中提供了全局forward，struts2中也提供了相似功能：
- <package ....>
- <global-results>
- <result name="message">/message.jsp</result>
- </global-results>
- </package>

为Action的属性注入值

- Struts2为Action中的属性提供了依赖注入功能，在struts2的配置文件中，我们可以很方便地为Action中的属性注入值。注意：属性必须提供setter方法。

```
package action;
public class SetPropertyValueAction {
	private String siteName;
	public String getSiteName() {
		return siteName;
	}
	public void setSiteName(String siteName) {
		this.siteName = siteName;
	}
	public String execute() {
		System.out.println(this.siteName);
		return null;
	}
}
```

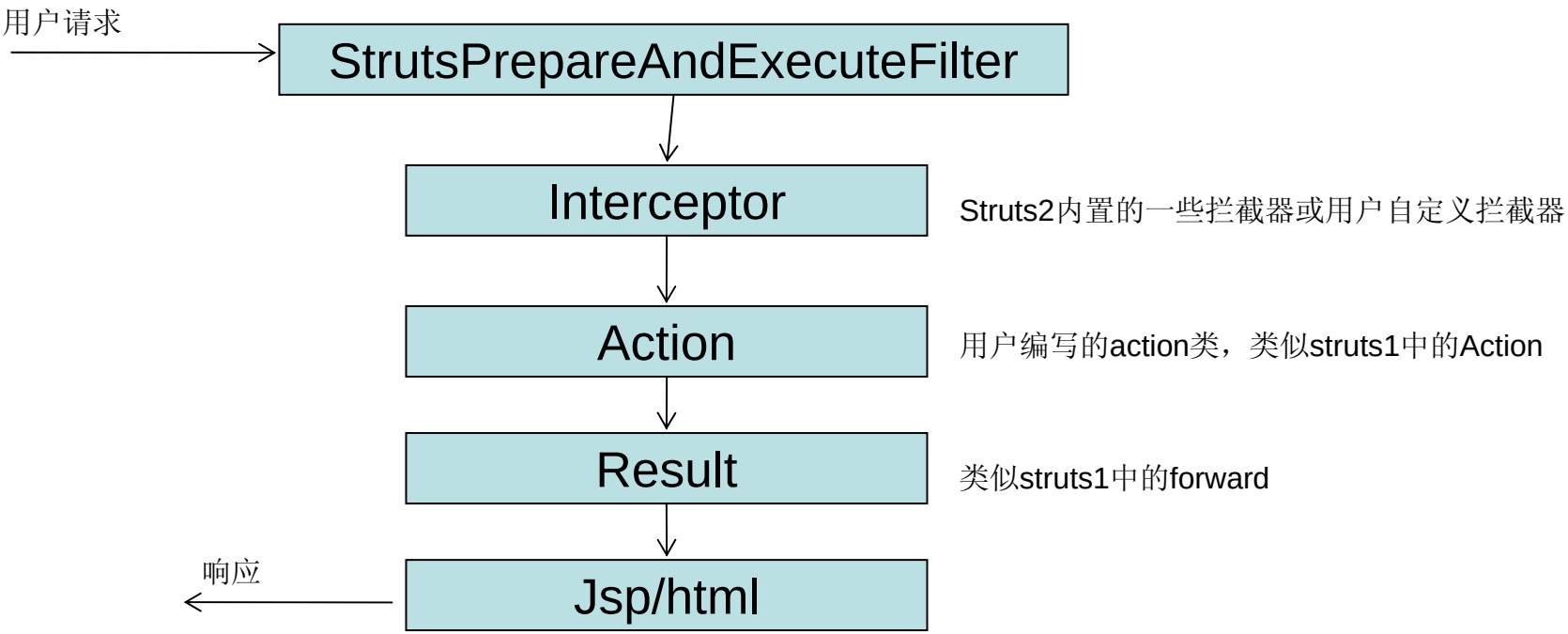
```
<action name="setvalue" class="action.SetPropertyValueAction">
		<param name="siteName">XX企业网</param>
</action>
```

常量定义

- 常量可以在struts.xml或struts.properties中配置，建议在struts.xml中配置
- 常用的常量介绍:

```
<!-- 指定默认编码集,作用于HttpServletRequest的setCharacterEncoding方法  -->
<constant name="struts.i18n.encoding" value="UTF-8"/>
<!-- 设置浏览器是否缓存静态内容,默认值为true(生产环境下使用),开发阶段最好关闭 -->
<constant name="struts.serve.static.browserCache" value="false"/>
<!-- 当struts的配置文件修改后,系统是否自动重新加载该文件,默认值为false(生产环境下使用),开发阶段最好打开 -->
<constant name="struts.configuration.xml.reload" value="true"/>
<!-- 开发模式下使用,这样可以打印出更详细的错误信息 -->
<constant name="struts.devMode" value="true" />
 <!-- 默认的视图主题 -->
<constant name="struts.ui.theme" value="simple" />
<!– 与spring集成时，指定由spring负责action对象的创建 -->
<constant name="struts.objectFactory" value="spring" />
<!–该属性设置Struts 2是否支持动态方法调用，该属性的默认值是true。如果需要关闭动态方法调用，则可设置该属性为false。 -->
<constant name="struts.enable.DynamicMethodInvocation" value="false"/>
<!--上传文件的大小限制-->
<constant name="struts.multipart.maxSize" value=“10701096"/>
```

Struts2的处理流程



StrutsPrepareAndExecuteFilter是Struts 2框架的核心控制器，它负责拦截由<url-pattern>/*</url-pattern>指定的所有用户请求，当用户请求到达时，该Filter会过滤用户的请求。默认情况下，如果用户请求的路径不带后缀或者后缀以.action结尾，这时请求将被转入Struts 2框架处理，否则Struts 2框架将略过该请求的处理。当请求转入Struts 2框架处理时会先经过一系列的拦截器，然后再到Action。与Struts1不同，Struts2对用户的每一次请求都会创建一个Action，所以Struts2中的Action是线程安全的。

应用指定多个struts配置文件

- 在大部分应用里，随着应用规模的增加，系统中Action的数量也会大量增加，导致struts.xml配置文件变得非常臃肿。为了避免struts.xml文件过于庞大、臃肿，提高struts.xml文件的可读性，我们可以将一个struts.xml配置文件分解成多个配置文件，然后在struts.xml文件中包含其他配置文件。下面的struts.xml通过<include>元素指定多个配置文件：

- <?xml version="1.0" encoding="UTF-8"?>
- <!DOCTYPE struts PUBLIC
- "-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
- "http://struts.apache.org/dtds/struts-2.0.dtd">
- <struts>
- <include file="struts-user.xml"/>
- <include file="struts-order.xml"/>
- </struts>

- 通过这种方式，我们就可以将Struts 2的Action按模块添加在多个配置文件中

使用通配符定义action

```
package action;
public class EmployeeAction {
	public String add() {
		System.out.println("增加员工Action被执行...");
		return null;
	}
	public String delete() {
		System.out.println("删除员工Action被执行...");
		return null;
	}
}
```

```
<package name="employee" namespace="/employee" extends="struts-default">
	<action name="list_*" class="action.EmployeeAction" method="{1}">
	</action>
</package>
```

当运行url地址为list_add.action时执行的是add方法，而运行url地址为list_delete.action时执行的是delete方法