JSP JavaBean JSP 表达式语言

JSP 自定义标签

自定义标签是用户定义的JSP语言元素。当JSP页面包含一个自定义标签时将被转化为servlet，标签转化为对被 称为tag handler的对象的操作，即当servlet执行时Web container调用那些操作。

JSP标签扩展可以让你创建新的标签并且可以直接插入到一个JSP页面。 JSP 2.0规范中引入Simple Tag Handlers来编写这些自定义标记。

你可以继承SimpleTagSupport类并重写的doTag()方法来开发一个最简单的自定义标签。

创建"Hello"标签

接下来，我们想创建一个自定义标签叫作<ex:Hello>，标签格式为：

```
<ex:Hello />
```

要创建自定义的JSP标签，你首先必须创建处理标签的Java类。所以，让我们创建一个HelloTag类，如下所示：

```
package com.runoob;

import javax.servlet.jsp.tagext.*;
import javax.servlet.jsp.*;
import java.io.*;

public class HelloTag extends SimpleTagSupport {

  public void doTag() throws JspException, IOException {
    JspWriter out = getJspContext().getOut();
    out.println("Hello Custom Tag!");
  }
}
```

以下代码重写了doTag()方法，方法中使用了getJspContext()方法来获取当前的JspContext对象，并将"Hello Custom Tag!"传递给JspWriter对象。

编译以上类，并将其复制到环境变量CLASSPATH目录中。最后创建如下标签库：<Tomcat安装目录>webapps\ROOT\WEB-INF\custom.tld。

```
<taglib>
  <tlib-version>1.0</tlib-version>
  <jsp-version>2.0</jsp-version>
  <short-name>Example TLD</short-name>
  <tag>
    <name>Hello</name>
```

```
    <tag-class>com.runoob.HelloTag</tag-class>
    <body-content>empty</body-content>
  </tag>
</taglib>
```

接下来，我们就可以在JSP文件中使用Hello标签：

```
<%@ taglib prefix="ex" uri="WEB-INF/custom.tld"%>
<html>
  <head>
    <title>A sample custom tag</title>
  </head>
  <body>
    <ex:Hello/>
  </body>
</html>
```

以上程序输出结果为：

```
Hello Custom Tag!
```

访问标签体

你可以像标准标签库一样在标签中包含消息内容。如我们要在我们自定义的Hello中包含内容，格式如下：

```
<ex:Hello>
   This is message body
</ex:Hello>
```

我们可以修改标签处理类文件，代码如下：

```
package com.runoob;

import javax.servlet.jsp.tagext.*;
import javax.servlet.jsp.*;
import java.io.*;

public class HelloTag extends SimpleTagSupport {

   StringWriter sw = new StringWriter();
   public void doTag()
      throws JspException, IOException
    {
       getJspBody().invoke(sw);
       getJspContext().getOut().println(sw.toString());
```

```
    }

}
```

接下来我们需要修改TLD文件，如下所示：

```
<taglib>
  <tlib-version>1.0</tlib-version>
  <jsp-version>2.0</jsp-version>
  <short-name>Example TLD with Body</short-name>
  <tag>
    <name>Hello</name>
    <tag-class>com.runoob.HelloTag</tag-class>
    <body-content>scriptless</body-content>
  </tag>
</taglib>
```

现在我们可以在JSP使用修改后的标签，如下所示:

```
<%@ taglib prefix="ex" uri="WEB-INF/custom.tld"%>
<html>
  <head>
    <title>A sample custom tag</title>
  </head>
  <body>
    <ex:Hello>
        This is message body
    </ex:Hello>
  </body>
</html>
```

以上程序输出结果如下所示：

```
This is message body
```

自定义标签属性

你可以在自定义标准中设置各种属性，要接收属性，值自定义标签类必须实现setter方法， JavaBean 中的setter方法如下所示：

```
package com.runoob;

import javax.servlet.jsp.tagext.*;
import javax.servlet.jsp.*;
```

```
import java.io.*;

public class HelloTag extends SimpleTagSupport {

   private String message;

   public void setMessage(String msg) {
      this.message = msg;
   }

   StringWriter sw = new StringWriter();

   public void doTag()
      throws JspException, IOException
    {
       if (message != null) {
          /* 从属性中使用消息 */
          JspWriter out = getJspContext().getOut();
          out.println( message );
       }
       else {
          /* 从内容体中使用消息 */
          getJspBody().invoke(sw);
          getJspContext().getOut().println(sw.toString());
       }
   }

}
```

属性的名称是"message"，所以setter方法是的setMessage()。现在让我们在TLD文件中使用的<attribute>元素添加此属性：

```
<taglib>
  <tlib-version>1.0</tlib-version>
  <jsp-version>2.0</jsp-version>
  <short-name>Example TLD with Body</short-name>
  <tag>
    <name>Hello</name>
    <tag-class>com.runoob.HelloTag</tag-class>
    <body-content>scriptless</body-content>
    <attribute>
       <name>message</name>
    </attribute>
  </tag>
</taglib>
```

现在我们就可以在JSP文件中使用message属性了，如下所示：

```
<%@ taglib prefix="ex" uri="WEB-INF/custom.tld"%>
<html>
  <head>
    <title>A sample custom tag</title>
  </head>
  <body>
    <ex:Hello message="This is custom tag" />
  </body>
</html>
```

以上实例数据输出结果为：

```
This is custom tag
```

你还可以包含以下属性：

属性	描述
name	定义属性的名称。每个标签的是属性名称必须是唯一的。
required	指定属性是否是必须的或者可选的,如果设置为false为可选。
rtexprvalue	声明在运行表达式时，标签属性是否有效。
type	定义该属性的Java类类型 。默认指定为 **String**
description	描述信息
fragment	如果声明了该属性,属性值将被视为一个 **JspFragment**。

以下是指定相关的属性实例：

```
.....
    <attribute>
      <name>attribute_name</name>
      <required>false</required>
      <type>java.util.Date</type>
      <fragment>false</fragment>
    </attribute>
.....
```

如果你使用了两个属性，修改TLD文件，如下所示：

```
.....
    <attribute>
      <name>attribute_name1</name>
```

```
      <required>false</required>
      <type>java.util.Boolean</type>
      <fragment>false</fragment>
    </attribute>
    <attribute>
      <name>attribute_name2</name>
      <required>true</required>
      <type>java.util.Date</type>
    </attribute>
.....
```

JSP JavaBean

JSP 表达式语言

1 篇笔记

写笔记

Eclipse 创建 tld 文件：

在 Eclipse 相应工程中右键单击 WEB-INF 目录弹出选项框，依次选：

->New->Other->XML->XML File->next->要取的文件名.tld

->next->Create XML File from an XML schema file

->next->Select XML Catalog entry

选择 http://xmlns.jcp.org/xml/ns/j2ee/web-jsptaglibrary_2_0.xsd 这一项，点击 Finish 即可。

拾肆 2年前 (2017-03-29)