**JSP 语法**

本小节将会简单地介绍一下JSP开发中的基础语法。

**脚本程序**

脚本程序可以包含任意量的Java语句、变量、方法或表达式，只要它们在脚本语言中是有效的。

脚本程序的语法格式：

<% 代码片段 %>

或者，您也可以编写与其等价的XML语句，就像下面这样：

<jsp:scriptlet>

代码片段

</jsp:scriptlet>

任何文本、HTML标签、JSP元素必须写在脚本程序的外面。

下面给出一个示例，同时也是本教程的第一个JSP示例：

<head><title>Hello World</title></head>

<body>

Hello World!<br/>

<%

out.println("Your IP address is " + request.getRemoteAddr());

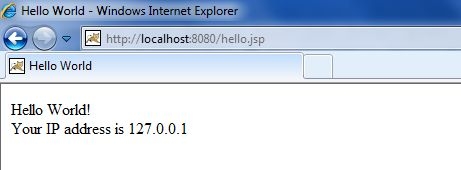
%>

</body>

</html>

**注意：**请确保Apache Tomcat已经安装在C:\apache-tomcat-7.0.2目录下并且运行环境已经正确设置。

将以上代码保存在hello.jsp中，然后将它放置在 C:\apache-tomcat-7.0.2\webapps\ROOT目录下，打开浏览器并在地址栏中输入http://localhost:8080/hello.jsp。运行后得到以下结果：



**JSP声明**

一个声明语句可以声明一个或多个变量、方法，供后面的Java代码使用。在JSP文件中，您必须先声明这些变量和方法然后才能使用它们。

JSP声明的语法格式：

<%! declaration; [ declaration; ]+ ... %>

或者，您也可以编写与其等价的XML语句，就像下面这样：

<jsp:declaration>

代码片段

</jsp:declaration>

程序示例：

<%! int i = 0; %>

<%! int a, b, c; %>

<%! Circle a = new Circle(2.0); %>

通过生成的java文件我们可以看出在servlet类中直接声明的变量和jsp中在<%!..%>块中声明的变量的作用范围是一样的.

在servlet的doPost和doGet方法中声明的变量和jsp中的<%..%>声明的变量的作用范围是一样的.  
  
在<%!…%>中声明的变量相当于一个静态变量, 所有的并发应用程序都使用同一个变量  
而在<%..%>中声明的变量只会在本次操作是有效的, 操作结束变量也就失效啦.  
所以说在<%!…%>中只能声明静态的变量, 一般的变量必须在<%…%>中声明  
同样在servlet中是如此, 在servlet类里面直接声明的变量在操作的时候也都使用同一个值, 一般类型的变量最好声明到doPost或者doGet方法中.

**JSP表达式**

一个JSP表达式中包含的脚本语言表达式，先被转化成String，然后插入到表达式出现的地方。

由于表达式的值会被转化成String，所以您可以在一个文本行中使用表达式而不用去管它是否是HTML标签。

表达式元素中可以包含任何符合Java语言规范的表达式，但是不能使用分号来结束表达式。

JSP表达式的语法格式：

<%= 表达式 %>

同样，您也可以编写与之等价的XML语句：

<jsp:expression>

表达式

</jsp:expression>

程序示例：

<html>

<head><title>A Comment Test</title></head>

<body>

<p>

Today's date: <%= (new java.util.Date()).toLocaleString()%>

</p>

</body>

</html>

运行后得到以下结果：

Today's date: 11-Sep-2013 21:24:25

**JSP注释**

JSP注释主要有两个作用：为代码作注释以及将某段代码注释掉。

JSP注释的语法格式：

<%-- 这里可以填写 JSP 注释 --%>

程序示例：

<html>

<head><title>A Comment Test</title></head>

<body>

<h2>A Test of Comments</h2>

<%-- 该部分注释在网页中不会被显示--%>

</body>

</html>

运行后得到以下结果：

A Test of Comments

不同情况下使用注释的语法规则：

|  |  |
| --- | --- |
| **语法** | **描述** |
| <%-- 注释 --%> | JSP注释，注释内容不会被发送至浏览器甚至不会被编译 |
| <!-- 注释 --> | HTML注释，通过浏览器查看网页源代码时可以看见注释内容 |
| <\% | 代表静态 <%常量 |
| %\> | 代表静态 %> 常量 |
| \' | 在属性中使用的单引号 |
| \" | 在属性中使用的双引号 |

**JSP指令**

JSP指令用来设置与整个JSP页面相关的属性。

JSP指令语法格式：

<%@ directive attribute="value" %>

这里有三种指令标签：

|  |  |
| --- | --- |
| **指令** | **描述** |
| <%@ page ... %> | 定义页面的依赖属性，比如脚本语言、error页面、缓存需求等等 |
| <%@ include ... %> | 包含其他文件 |
| <%@ taglib ... %> | 引入标签库的定义，可以是自定义标签 |

**JSP行为**

JSP行为标签使用XML语法结构来控制servlet引擎。它能够动态插入一个文件，重用JavaBean组件，引导用户去另一个页面，为Java插件产生相关的HTML等等。

行为标签只有一种语法格式，它严格遵守XML标准：

<jsp:action\_name attribute="value" />

行为标签基本上是一些预先就定义好的函数，下表罗列出了一些可用的JSP行为标签：：

|  |  |
| --- | --- |
| **语法** | **描述** |
| jsp:include | 用于在当前页面中包含静态或动态资源 |
| jsp:useBean | 寻找和初始化一个JavaBean组件 |
| jsp:setProperty | 设置 JavaBean组件的值 |
| jsp:getProperty | 将 JavaBean组件的值插入到 output中 |
| jsp:forward | 从一个JSP文件向另一个文件传递一个包含用户请求的request对象 |
| jsp:plugin | 用于在生成的HTML页面中包含Applet和JavaBean对象 |
| jsp:element | 动态创建一个XML元素 |
| jsp:attribute | 定义动态创建的XML元素的属性 |
| jsp:body | 定义动态创建的XML元素的主体 |
| jsp:text | 用于封装模板数据 |

**JSP隐含对象**

JSP支持九个自动定义的变量，江湖人称隐含对象。这九个隐含对象的简介见下表：

|  |  |
| --- | --- |
| **对象** | **描述** |
| request | **HttpServletRequest**类的实例 |
| response | **HttpServletResponse**类的实例 |
| out | **PrintWriter**类的实例，用于把结果输出至网页上 |
| session | **HttpSession**类的实例 |
| application | **ServletContext**类的实例，与应用上下文有关 |
| config | **ServletConfig**类的实例 |
| pageContext | **PageContext**类的实例，提供对JSP页面所有对象以及命名空间的访问 |
| page | 类似于Java类中的this关键字 |
| Exception | **Exception**类的对象，代表发生错误的JSP页面中对应的异常对象 |

**控制流语句**

JSP提供对Java语言的全面支持。您可以在JSP程序中使用Java API甚至建立Java代码块，包括判断语句和循环语句等等。

**判断语句**

If…else块，请看下面这个例子：

<%! int day = 3; %>

<html>

<head><title>IF...ELSE Example</title></head>

<body>

<% if (day == 1 | day == 7) { %>

<p> Today is weekend</p>

<% } else { %>

<p> Today is not weekend</p>

<% } %>

</body>

</html>

运行后得到以下结果：

Today is not weekend

现在来看看switch…case块，与if…else块有很大的不同，它使用out.println()，并且整个都装在脚本程序的标签中，就像下面这样：

<%! int day = 3; %>

<html>

<head><title>SWITCH...CASE Example</title></head>

<body>

<%

switch(day) {

case 0:

out.println("It\'s Sunday.");

break;

case 1:

out.println("It\'s Monday.");

break;

case 2:

out.println("It\'s Tuesday.");

break;

case 3:

out.println("It\'s Wednesday.");

break;

case 4:

out.println("It\'s Thursday.");

break;

case 5:

out.println("It\'s Friday.");

break;

default:

out.println("It's Saturday.");

}

%>

</body>

</html>

运行后得出以下结果：

It's Wednesday.

循环语句

在JSP程序中可以使用Java的三个基本循环类型：for，while，和 do…while。

让我们来看看for循环的例子：

<%! int fontSize; %>

<html>

<head><title>FOR LOOP Example</title></head>

<body>

<%for ( fontSize = 1; fontSize <= 3; fontSize++){ %>

<font color="green" size="<%= fontSize %>">

JSP Tutorial

</font><br />

<%}%>

</body>

</html>

运行后得到以下结果：

JSP Tutorial  
JSP Tutorial  
JSP Tutorial

将上例改用while循环来写：

<%! int fontSize; %>

<html>

<head><title>WHILE LOOP Example</title></head>

<body>

<%while ( fontSize <= 3){ %>

<font color="green" size="<%= fontSize %>">

JSP Tutorial

</font><br />

<%fontSize++;%>

<%}%>

</body>

</html>

运行后得到同样的结果：

JSP Tutorial  
JSP Tutorial  
JSP Tutorial

**JSP运算符**

JSP支持所有Java逻辑和算术运算符。

下表罗列出了JSP常见运算符，优先级从高到底：

|  |  |  |
| --- | --- | --- |
| **类别** | **操作符** | **结合性** |
| 后缀 | () [] . (点运算符) | 左到右 |
| 一元 | ++ - - ! ~ | 右到左 |
| 可乘性 | \* / % | 左到右 |
| 可加性 | + - | 左到右 |
| 移位 | >> >>> << | 左到右 |
| 关系 | > >= < <= | 左到右 |
| 相等/不等 | == != | 左到右 |
| 位与 | & | 左到右 |
| 位异或 | ^ | 左到右 |
| 位或 | | | 左到右 |
| 逻辑与 | && | 左到右 |
| 逻辑或 | || | 左到右 |
| 条件判断 | ?: | 右到左 |
| 赋值 | = += -= \*= /= %= >>= <<= &= ^= |= | 右到左 |
| 逗号 | , | 左到右 |

**JSP常量**

JSP语言定义了以下几个常量：

* Boolean：true and false
* Integer：与Java中的一样
* Floating point：与Java中的一样
* String：以单引号或双引号开始和结束。 " 被转义成 \"，'被转义成 \'， \ 被转义成\\
* Null：null

来源： <<http://www.w3cschool.cc/jsp/jsp-syntax.html>>