

Java EE 企业应用系统设计

JSP

王晓东

wxd2870@163.com

中国海洋大学

December 16, 2012



参考书目

1. 吕海东，张坤编著，Java EE 企业级应用开发实例教程，清华大学出版社，2010 年 8 月



大纲

JSP 概述

JSP 指令

JSP 动作

include 动作

useBean 动作

setProperty 动作

getProperty 动作

forward 动作

param 动作

JSP 脚本

JSP 内置对象



接下来…

JSP 概述

JSP 指令

JSP 动作

include 动作

useBean 动作

setProperty 动作

getProperty 动作

forward 动作

param 动作

JSP 脚本

JSP 内置对象



JSP 基本概念

- ▶ JSP, Java Server Page, 即 Java 服务器页面。
- ▶ JSP 是 Servlet 的扩展。
- ▶ 它将使用 Java 类编写动态 Web 组件转变为使用文本编写, 降低了开发的难度。
- ▶ JSP 提供了一种自然的生成网页的方法。
- ▶ 可以使用 GUI 工具来绘制 JSP 页面, 而不是像 Servlet 写 Java 代码方法。
- ▶ JSP 文件的扩展名必须是.jsp。



JSP 的优点和缺点

❖ 优点

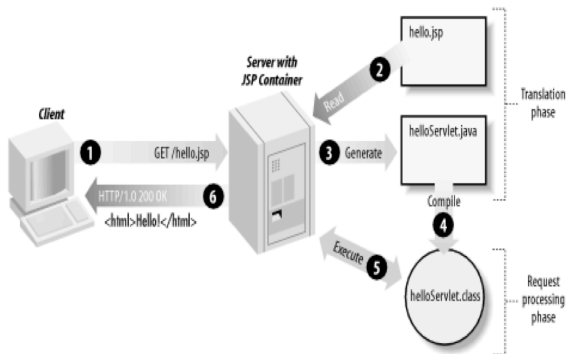
- ▶ 编写动态 Web 网页更加容易；
- ▶ 降低了开发难度；
- ▶ 可以使用工具的拖拉方式生成 JSP 页面；
- ▶ 纯文本文件。

❖ 缺点

- ▶ 非 OO 编程方式；
- ▶ Java 代码嵌入到 HTML 代码中，维护困难；
- ▶ 不适合编写正常的业务处理应用程序。



JSP 的执行过程



JSP 执行过程描述

1. 客户使用浏览器通过 HTTP 请求 JSP 文件的 URL 地址，如：
`http://localhost:8080/web01/w/a.jsp`；
2. Web 服务器接收到请求，如果没有此地址，发出错误响应给浏览器；
3. Web 服务器检查 JSP 文件和对应的 Servlet 版本的时间是否一致，如果一致则执行 servlet 的处理请求方法，类似于 `doGet` 或 `doPost`，发送响应给浏览器；
4. 版本时间不一致，Web 服务器调用转化系统，将 JSP 的文本代码转换为 Servlet 的 Java 代码；
5. 将 Java 代码编译为 class 文件；
6. 调用 Servlet Class 的响应方法。



JSP 页面的组成

一个 JSP 页面由 HTML 标记代码和 JSP 元素组成。HTML 标记用于生成网页的静态部分，JSP 元素用于生成动态内容部分。

JSP 元素

JSP 指令

JSP 动作

JSP 脚本

JSP 内置对象

JSP 扩展标记



接下来...

JSP 概述

JSP 指令

JSP 动作

include 动作

useBean 动作

setProperty 动作

getProperty 动作

forward 动作

param 动作

JSP 脚本

JSP 内置对象



JSP 指令

JSP 指令指示一个 JSP 页面的属性和特征，JSP 指令不会产生任何的输出到当前输出流中。

❖ JSP 指令

- ▶ **page** 指令，用于定义 JSP 页面级的其他元素特征。
- ▶ **include** 指令，用于嵌入另一个文本文件的内容到本页面。
- ▶ **taglib** 指令，用于引入第三方 JSP 扩展标记类库。

❖ JSP 指令的语法

1 `<%@ 指令名属性名="值" 属性名="值" %>`



Page 指令 ①

Page 指令定义应用于整个页面的属性。

语法

```
1 <%@ page 属性名="属性值" %>
```

属性名和值

- ▶ `language="java"`，指定页面语言。
- ▶ `contentType="text/html; charset=gb2312"`，指定页面的内容类型，默认是 `text/html`，可以指定显示的字符集，中文是 `gb2312`。
- ▶ `import="package, package"`，指定 JSP 页面使用的包和类，可以引用多个包，每个包用逗号分隔。



Page 指令 ②

属性名和值

- ▶ `buffer="none | xkb"`，指定输出缓冲区容量，默认为 8KB，`buffer="9kb"`。
- ▶ `errorPage="errorURL"`，指定错误页面地址，当页面出现异常时自动跳转到指定的错误页面。
- ▶ `isErrorPage="true | false"`，指定本页面是否是错误处理页面。
- ▶ `autoFlush="true | false"`，控制输出缓冲区是否自动清空，默认是 `true`。



include 指令

include 指令用于在当前网页中嵌入另一个网页，可以是 JSP、HTML 等。

语法

```
1 <%@ include file="url" %>
```

说明

- ▶ file 属性确定要嵌入的页面。
- ▶ 嵌入页面的源代码被放置在此指令所在的位置。
- ▶ 嵌入的用途是将一个复杂的页面分解为小的页面，然后使用 include 指令将他们再组装在一起。
- ▶ 当修改被 include 的文件时，如果不修改主文件，则嵌入的内容不会改变。因此必须也修改主文件才能反映更新的嵌入文件。



taglib 指令

taglib 指令用于引入扩展标签库，如 JSTL、Struts、自定义标签库。

语法

```
1 <%@ taglib uri="/WEB-INF/tlds/struts-html.tld" prefix="html" %>
2 <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
```



接下来...

JSP 概述

JSP 指令

JSP 动作

include 动作

useBean 动作

setProperty 动作

getProperty 动作

forward 动作

param 动作

JSP 脚本

JSP 内置对象



JSP 动作

JSP 动作使用特定的符合 XML 格式的标记完成特定的任务利用 JSP 动作可以完成动态的插入文件、重用 **JavaBean** 组件、把用户重定向到另外的页面、为 **Java** 插件生成 **HTML** 代码。

❖ JSP 动作的语法

无嵌套封闭格式

```
<jsp:动作名称 属性名="值" 属性名="值" 属性名="值"/>
```

有嵌套封闭格式

```
<jsp:动作名称 属性名="值" 属性名="值" 属性名="值">  
    嵌入的其他动作  
</jsp:动作名称>
```



JSP 动作的类型

`<jsp:include>` 嵌入其他页面输出内容动作；

`<jsp:forward>` 转发动作；

`<jsp:plugin>` 引入插件动作；

`<jsp:param>` 提供参数动作；

`<jsp:useBean>` 使用 JavaBean 动作；

`<jsp:setProperty>` 设置 JavaBean 属性动作；

`<jsp:getProperty>` 取得 JavaBean 属性动作。



include 动作

●○○○○○○○○○○○○○○○○○○

接下来...

JSP 概述

JSP 指令

JSP 动作

include 动作

useBean 动作

setProperty 动作

getProperty 动作

forward 动作

param 动作

JSP 脚本

JSP 内置对象



include 动作

include 动作用于嵌入其他页面的输出内容到此动作所在页面。

```
<jsp:include page="url" flush="true" />
```

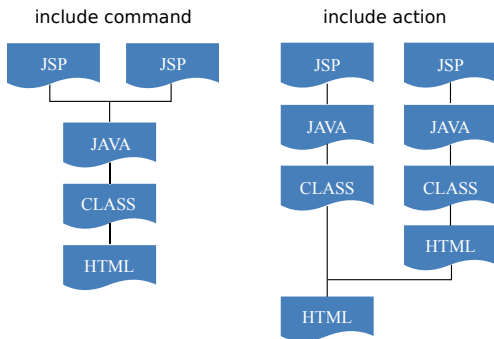
```
<jsp:include page="url" flush="true">  
  <jsp:param name="参数名" value="参数值" />  
</jsp:include>
```

- ▶ page 属性，指定嵌入页面的 URL 地址；
- ▶ flush 属性，指定是否在嵌入页面之前清空响应缓存区，默认为 true；
- ▶ jsp:param，为嵌入的页面传递参数，这些参数可以为动态、也可以为静态。



include 指令和 include 动作的差异

- ▶ 其根本性不同在于他们被调用的时间。**include 指令**在页面解析期间被嵌入；**include 动作**在请求的响应输出时被嵌入。
- ▶ 在实现文件包含上，因该尽可能的使用 **include 动作**。
- ▶ 而 **include 指令**存在的原因是其功能更加强大，执行速度稍快。



useBean 动作

○○●○○○○○○○○○○○○○○

接下来...

JSP 概述

JSP 指令

JSP 动作

include 动作

useBean 动作

setProperty 动作

getProperty 动作

forward 动作

param 动作

JSP 脚本

JSP 内置对象



useBean 动作

- ▶ 引用并调用 **JavaBean** 的方法和属性是 JSP 开发时必须面对的主要任务。
- ▶ JSP 为了简化 **JavaBean**，提供了 **useBean** 动作标记，是的不必使用 Java 脚本代码，而使用标记方式取得 **JavaBean** 对象引用。

❖ useBean 动作语法

```
<jsp:useBean id="name" class="package.ClassName"
  scope="page|request|session|application" />
```

- ▶ **id="name"** 实例化变量名
- ▶ **scope="scope"** 定义实例变量的生存周期范围
 - ▶ **page** 只在本页面中使用，默认值
 - ▶ **request** 在请求范围内有效
 - ▶ **session** 在会话范围内有效
 - ▶ **application** 在整个 Web 启动后有效
- ▶ **class="package.ClassName"** 指定 **JavaBean** 的类



useBean 的执行过程

1. 如果在指定范围内找到指定的对象，则得到此对象引用 (即通过 scope 对象的 `getAttribute()` 方法)。
2. 如果没有找到指定的对象，则实例化一个 class 属性指定的对象 (调用类的无参数构造方法)。
3. 对新建的对象执行嵌入的 `<jsp:setProperty />` 中指定的属性值，即调用 JavaBean 的 `setXXX` 方法，设置类的属性值。
4. 将对象保存到 scope 指定的范围的对象中，即调用内置对象的 `setAttribute(name,Object)` 方法，name 是 id 指定的名称，object 是类的对象。



useBean 动作示例

JavaBean 类 User : User.java

```
1 package com.city.oa.value;
2 import java.io.Serializable;
3 public class User implements Serializable {
4     private String id = null;
5     private String password = null;
6     private String name = null;
7     private age = 0;

9     public String getId() {
10         return id;
11     }
12     public void setId(String id) {
13         this.id = id;
14     }
15     ... .. // (other set/get methods..)
16 }
```

创建/取得指定的 JavaBean 对象，并保存在 Request 对象中

```
1 <jsp:useBean id="user" class="com.city.oa.value.User" scope="request" >
2 <%
3     String name = user.getName();
4     out.println(name);
5 %>
```



使用 useBean 动作 VS. JSP 脚本创建并取得对象引用

```
1 <% User user = new User(); %>
```

VS.

```
1 <jsp:useBean id="user" class="com.city.oa.value.User" scope="request" \>
```

- ▶ 使用 Java 脚本创建的 **JavaBean**，只能在本页面中使用，且每次都是创建新的 **JavaBean** 对象，不会保存在任何范围对象中。
- ▶ **useBean 动作**

```
1 <jsp:useBean id="user" class="com.city.oa.value.User" scope="request" \>
```

相当于如下 Java 脚本

```
1 <%  
2 User user = (User) request.getAttribute("user");  
3 if (user == null) {  
4     user = New User();  
5     request.setAttribute("user", user);  
6 }  
7 % >
```



setProperty 动作

○○○○○○○○●○○○○○○○

接下来...

JSP 概述

JSP 指令

JSP 动作

include 动作

useBean 动作

setProperty 动作

getProperty 动作

forward 动作

param 动作

JSP 脚本

JSP 内置对象



setProperty 动作

用于设定 userBean 动作取得的 Bean 对象的属性。相当于执行 Bean 对象的 setXxx 方法。

```
1 <jsp:setProperty name="beanId" property="*|name" param="参数名" value="value" />
```

```
1 <jsp:useBean id="user" class="com.city.oa.value.User" scope="request" />  
2 <jsp:setProperty name="user" property="name" value="吴明" >
```



getProperty 动作

○○○○○○○○○●○○○○○

接下来...

JSP 概述

JSP 指令

JSP 动作

include 动作

useBean 动作

setProperty 动作

getProperty 动作

forward 动作

param 动作

JSP 脚本

JSP 内置对象



getProperty 动作

用于取得 Bean 对象指定的属性值，转换为 String 类型，并显示在此动作所在的位置。

```
1 <jsp:getProperty name="beanId" property="属性名" />
```

- ▶ name 指定 bean 对象的名称，与 useBean 动作的 id 值对应；
- ▶ property 指定属性名，与 JavaBean 的 getXxx 方法对应。



forward 动作

○○○○○○○○○○●○○○

接下来...

JSP 概述

JSP 指令

JSP 动作

include 动作

useBean 动作

setProperty 动作

getProperty 动作

forward 动作

param 动作

JSP 脚本

JSP 内置对象



forward 动作

forward 动作把请求转到另外的页面。

无嵌入参数的转发动作

```
1 <jsp:forward page="URL" />
```

例如

```
1 <jsp:forward page="main.jsp" />
```

有嵌入参数的转发动作

```
1 <jsp:forward page="URL">
2   <jsp:param name="参数名" value="值" />
3   ...
4 </jsp:forward>
```

例如

```
1 <jsp:forward page="main.jsp">
2   <jsp:param name="id" value="kevin" />
3   <jsp:param name="password" value="1000" />
4 </jsp:forward>
```



param 动作

○○○○○○○○○○○○○○●○

接下来...

JSP 概述

JSP 指令

JSP 动作

include 动作

useBean 动作

setProperty 动作

getProperty 动作

forward 动作

param 动作

JSP 脚本

JSP 内置对象



param 动作

param 动作不能单独使用，需要嵌套在其他动作中，为其他动作提供参数，它可以嵌入到 **include**、**forward** 动作中为目标地址提供参数。例如：

```
1 <jsp:forward page="main.jsp">
2   <jsp:param name="id" value="kevin" />
3   <jsp:param name="password" value="1000" />
4 </jsp:forward>
```



接下来...

JSP 概述

JSP 指令

JSP 动作

include 动作

useBean 动作

setProperty 动作

getProperty 动作

forward 动作

param 动作

JSP 脚本

JSP 内置对象



JSP 脚本类型

代码脚本

```
1 <%  
2     int a=0; // 可以放置任何Java 代码  
3 %>
```

表达式脚本

```
1 <%= a %> 用于输出Java 表达式的值
```

声明脚本

```
1 <%!  
2     int m=0; // 声明JSP 类变量和方法  
3 %>
```

注释脚本

```
1 <%-- 注释-->
```



代码脚本

- ▶ 可以同时嵌入多个脚本代码段，但它们都表示在一个方法内，属于一个代码段，一个脚本内定义的变量，另一个脚本也可以使用。
- ▶ 在代码脚本中定义的变量类似于 Servlet 的 doGet 方法中的局部变量，未初始化使用是非法的。

例如，连接数据库的代码脚本片段如下：

```
1  <%  
2    Connection cn = null;  
3    try {  
4      Class.forName("sun.jdbc.odbc.JdbcOdbc.Driver");  
5      cn = DriverManager.getConnection("jdbc:odbc:cityoa");  
6      String sql = "select * from EMP";  
7      ... ..  
8    }  
9  %>
```



表达式脚本

注意：表达式后不能有分号；= 号与 <% 之间不能有空格，例如：

```
1 <%= rs.getString("NAME") %>
```



声明脚本

用于声明 JSP 页面的类变量和方法。

由于 JSP 在运行时会转换为 Servlet 类，声明的脚本部分会成为类中定义的一部分。例如：

```
1 <%!  
2     int num = 0;  
3     public void addNum() {  
4         num++;  
5     }  
6 %>
```

如下声明脚本有错误：

```
1 <%!  
2     int num = 0;  
3     num++; // 错误，声明脚本中不能直接有非声明代码  
4 %>
```



注释脚本

在 JSP 页面中可以使用 HTML 注释：

```
1 <!-- HTML Comment -->
```

但是，使用 HTML 注释不安全，因为 HTML 注释随着 JSP 生成的 HTML 响应下载到客户端浏览器，客户可以看到。

JSP 注释是服务器端技术，在服务器端处理，不会发送到客户端，比较安全。

```
1 <%-- JSP Comment --%>
```



接下来...

JSP 概述

JSP 指令

JSP 动作

include 动作

useBean 动作

setProperty 动作

getProperty 动作

forward 动作

param 动作

JSP 脚本

JSP 内置对象



JSP 内置对象

JSP 作为 Web 组件，为和 Web 容器以及其他 Web 组件进行通信和协作，提供了内置的与 HTTP 请求和响应相关的对象，这些对象不需要定义和引用，在 JSP 代码脚本和表达式脚本中可以直接使用。

- ▶ request 请求对象
- ▶ response 响应对象
- ▶ session 会话对象
- ▶ application 应用服务器对象
- ▶ page JSP 本身页面类对象
- ▶ pageContext 页面级环境变量，作为页面级容器
- ▶ out 输出对象
- ▶ exception 异常对象
- ▶ config 配置对象，用于读取 web.xml 配置信息



请求对象 request

request 与 Servlet 中传递的请求对象相同，类型为 `javax.servlet.http.HttpServletRequest`。

常见方法

- ▶ `Object getAttribute(String name)` 返回指定属性的属性值
- ▶ `Enumeration getAttributeNames()` 返回所有可用属性名的枚举
- ▶ `String getCharacterEncoding()` 返回字符编码的方式
- ▶ `String getParameter(String name)` 返回指定参数的参数值
- ▶ ...



响应对象 response

JSP 页面使用文本方式实现 HTTP 响应，所以 JSP 内部 response 对象没有像 Servlet 中响应对象使用那么频繁，在 JSP 中实现响应，直接将响应内容写在 JSP 页面就可以，不需要使用响应对象取得 `PrintWriter` 对象，再进行响应输出。

响应对象类型为 `javax.servlet.http.HttpServletResponse`，JSP 使用不是很多。

常见方法

- ▶ `String getCharacterEncoding()` 返回响应用的是何种编码
- ▶ ...



会话对象 session

JSP 的 session 对象对应 Servlet 中的 `javax.servlet.http.HttpSession`。

JSP 页面使用 `page` 指令

```
1 <%@ page session="true|false" %>
```

指示此 JSP 页面是否可以使用 session 内置对象。

常见方法

- ▶ `long getCreationTime()` 返回 session 创建时间
- ▶ `public String getId()` 返回 session 创建是 JSP 引擎为它设置的唯一 ID 号
- ▶ `long getLastAccessdTime()` 返回此 session 中客户端最近一次请求的时间
- ▶ `int getMaxInactiveInterval()` 返回两起请求间隔多长时间此 session 被取消
- ▶ `String[] getValueNames()` 返回一个包含此 session 中所有可用属性的数组
- ▶ `void invalidate()` 取消 session, 使其不可用



服务器环境对象 application

- ▶ application 对应 Servlet 中的 `javax.servlet.ServletContext` 对象实例。
- ▶ 每个 Web 站点只有一个 application 对象。
- ▶ application 对象是一个 Map 类型的容器。
- ▶ application 作为 Web 级的容器，保存整个 Web 应用个 Web 组件之间需要共享的数据，application 可以保存整个请求用户都可以访问的共享信息。
- ▶ 在保存信息到 application 时，要注意多用户同时写的线程同步问题。

常用方法

- ▶ 存入数据方法 `void setAttribute(String name, Object object);`
- ▶ 读取数据的方法 `Object getAttribute(String name);`
- ▶ 删除数据的方法 `void removeAttribute(String name);`



页面对象 page

`page` 对象就是指向当前 JSP 页面本身，有点像类中的 `this` 指针，它是 `java.lang.Object` 的实例，在 JSP 编程中应用较少。

常用方法

- ▶ `int hashCode()` 返回此 `Object` 的 hash 码



页面环境对象 pageContext



输出对象 out

out 内置对象即 JSP 页面向浏览器发出响应流 **PrintWriter** 的实例对象。但 JSP 页面可以直接放入响应文本，因此 JSP 页面基本不使用 **out** 进行文本响应。

一般情况下使用

```
1 <%= %>
```

来代替使用 **out** 对象。



异常对象 exception

JSP 内置对象 `exception` 对应 Java 中的 `java.lang.Throwable` 接口对象。它自动封装 JSP 页面中出现的异常。异常对象只有在错误页面中可以使用，即使用 `page` 指令的属性 `isErrorPage="true"` 声明的页面。

exception01.jsp

```
1  <%@ page language="java" contentType="text/html; charset=utf-8"
2     errorPage="error.jsp" import="java.util.*" pageEncoding="utf-8" %>
3  <html>
4  <head>
5  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
6  <title>Test Exception</title>
7  </head>
8  <body>
9  <%
10 int s = 0;
11 int t = 0;
12 int p = s / t; // 出现异常的语句
13 %>
14 </body>
15 </html>
```



异常对象 exception

error.jsp

```
1 <%@ page language="java" contentType="text/html; charset=utf-8" isErrorPage="true"
2   pageEncoding="utf-8"%>
3 <html>
4 <head>
5 <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
6 <title>Error Page</title>
7 </head>
8 <body>
9 <h1>异常显示信息</h1>错误原因:
10 <%= exception.getMessage() %>
11 </body>
12 </html>
```

使用 **errorPage** 方式实现的页面跳转，使用的是转发方式，而不是重定向模式。



配置对象 config

config 对象提供对 JSP 页面的 `javax.servlet.ServletConfig` 对象的访问。
例如：

```
1 <%= config.getInitParameter("driverName") %>
```





THE END

wxd2870@163.com

