# **JSP动作**

1. **forward动作（request参数可传递，执行的还是下一个页面，结果返回本夜面）：**

**//转发是服务器行为，重定向是客户端行为（两次request）**

1、在servlet中调用转发、重定向的语句如下：

request.getRequestDispatcher("new.jsp").forward(request, response); //转发到new.jsp

response.sendRedirect("new.jsp"); //重定向到new.jsp

在jsp页面中你也会看到通过下面的方式实现转发：

<jsp:forward page="apage.jsp" />

当然也可以在jsp页面中实现重定向：

<%response.sendRedirect("new.jsp"); %> //重定向到new.jsp

1. 解释转发与重定向：

**转发过程：**客户浏览器发送http请求——》web服务器接受此请求——》调用内 部的一个方法在容器内部完成请求处理和转发动作——》将目标资源发送给客 户；在这里，转发的路径必须是同一个web容器下的url，其不能转向到其他 的 web路径上去，中间传递的是自己的容器内的request。在客户浏览器路径栏显示 的仍然是其第一次访问的路径，也就是说客户是感觉不到服务器做了转发的。转发 行为是浏览器只做了一次访问请求。

**重定向过程：**客户浏览器发送http请求——》web服务器接受后发送302状态码 响应及对应新的location给客户浏览器——》客户浏览器发现是302响应，则自动 再发送一个新的http请求，请求url是新的location地址——》服务器根据此请求 寻找资源并发送给客户。在这里location可以重定向到任意URL，既然是浏览器重 新发出了请求，则就没有什么request传递的概念了。在客户浏览器路径栏显示的 是其重定向的路径，客户可以观察到地址的变化的。重定向行为是浏览器做了至少 两次的访问请求的。

1. 转发与重定向的区别：

**重定向：**第一次，客户端request A,服务器响应，并response回来，告诉浏览器， 你应该去B。这个时候IE可以看到地址变了，而且历史的回退按钮也亮了 可以访问自己web应用以外的资源。在重定向的过程中，传输的信息会 被丢失。response.sendRedirect("loginsuccess.jsp");

**请求转发**：是服务器内部把对一个request/response的处理权，移交给另外一个

对于客户端而言，它只知道自己最早请求的那个A，而不知道中间的B， 甚至 C、D。传输的信息不会丢失。

举例说明：

重定向：你先去了A局，A局的人说：“这个事情不归我们管，去B局”，然后，你 就从A退了出来，自己乘车去了B局。

转发：你先去了A局，A局看了以后，知道这个事情其实应该B局来管，但是他没 有把你退回来，而是让你坐一会儿，自己到后面办公室联系了B的人，让他 们办好后，送了过来。

1. Servlet 跳转 redirect与forward跳转的区别：

//在servlet中，一般跳转都发生在doGet, doPost等方法里面。

1. redirect 方式

response.sendRedirect("/a.jsp");

页面的路径是相对路径。sendRedirect可以将页面跳转到任何页面，不一定局限于本web 应用中，如：response.sendRedirect("http://www.ycul.com");

跳转后浏览器地址栏变化。

这种方式要传值出去的话，只能在url中带parameter或者放在session中，无法使用 request.setAttribute来传递。

1. forward方式

RequestDispatcher dispatcher = request.getRequestDispatcher("/a.jsp");

dispatcher .forward(request, response);

页面的路径是相对路径。forward方式只能跳转到本web应用中的页面上。

跳转后浏览器地址栏不会变化。

使用这种方式跳转，传值可以使用三种方法：url中带parameter，session，

request.setAttribute

5.区别：1、forward重定向是在容器内部实现的同一个Web应用程序的重定向，所以 forward方法只能重定向到同一个Web应用程序中的一个资源，重定向后浏览 器地址栏URL不变，而sendRedirect方法可以重定向到任何URL， 因为这种 方法是修改http头来实现的，URL没什么限制，重定向后浏览器地址栏URL 改变。

2、forward重定向将原始的HTTP请求对象（request）从一个servlet实例传递 到另一个实例，而采用sendRedirect方式两者不是同一个application。

**3、基于第二点，参数的传递方式不一样。forward的form参数跟着传递，所 以在第二个实例中可以取得HTTP请求的参数。sendRedirect只能通过链接传 递参数，response.sendRedirect(“login.jsp?param1=a”)。**

4、sendRedirect能够处理相对URL，自动把它们转换成绝对URL，如果地址是 相对的，那么Web container就认为它是相对于当前的请求URI 的。比如，如 果为response.sendRedirect("login.jsp")，则会从当前servlet 的 URL路径下

找login.jsp：http://10.1.18.8:8081/dms/servlet/Servlet 重定向的 URL:http://10.1.18.8:8081/dms/servlet/login.jsp如果为 response.sendRedirect("/login.jsp")则会从当前应用径下查找 url:http://10.1.18.8:8081/login.jsp。而forward不能这样处理相对路径。

1. 使用注意：报错

JSP：1) response.sendRedirect();

和servlet的response.sendRedirect()方式一样。

此语句前不允许有out.flush()，如果有，会有异常：

java.lang.IllegalStateException: Can't sendRedirect() after data has committed to the client.

Atcom.caucho.server.connection.AbstractHttpResponse.sendRedirect(AbstractHttpResponse.java:558)...

跳转后浏览器地址栏变化

如果要跳到不同主机下，跳转后，此语句后面的语句会继续执行，如同新开了线程，但是对response的操作已经无意义了；

如果要跳到相同主机下，此语句后面的语句执行完成后才会跳转；

2) response.setHeader("Location","");

此语句前不允许有out.flush()，如果有，页面不会跳转。

跳转后浏览器地址栏变化

此语句后面的语句执行完成后才会跳转

3) <jsp:forward page="" />

此语句前不允许有out.flush()，如果有，会有异常：

java.lang.IllegalStateException: forward() not allowed after buffer has committed.

at com.caucho.server.webapp.RequestDispatcherImpl.forward(RequestDispatcherImpl.java:134)

at com.caucho.server.webapp.RequestDispatcherImpl.forward(RequestDispatcherImpl.java:101)

at com.caucho.jsp.PageContextImpl.forward(PageContextImpl.java:836)

...

跳转后浏览器地址栏不变，但是只能跳到当前主机下

此语句后面的语句执行完成后才会跳转

二、**include指令：在页面转换期间将文件包含到jsp页面。（最终转换为sevlet时执行包含）**

**Include动作：在标识被执行时，程序会将请求转发。（此时执行包含）【两个sevlet】**

//类似中断，之后还能回来

include指令用于在jsp页面中静态包含一个文件，该文件可以是jsp页面，html页面， 文本文件或一段Java代码。

在主页面“转换为servlet前”将jsp代码插入。被包含的页面发生更改的时候，主页面 也要更新（转译）。

对于文件包含，使用jsp:include。仅在所包含的文件中定义了主页面需要的字段和方法， 或者所包含的文件设置了主页面的响应报头时，才使用include指令。

1，include语法：

**<%@ include file="url"%>**

file的属性值被解释为相对于当前jsp文件的URL.

2，在被包含的文件中不要使用和主页重复的HTML标签。这样会影响原jsp文件中同 样的标签导致错误。

因为原文件和被包含文件可以互相访问彼此定义的变量和方法，所以要避免变量和 方法在命名上的问题。

3，在jsp中，只有3种功能可以将外部内容包含到JSP文档中。

**a**，jsp:include动作//该标识被执行时，程序会将请求转发（不是请求重定向）到被包含

【两个页面】页面，并将执行结果输出到浏览器中，然后返回包含页继续执行后面的代码。

**b**，include指令//最终编译-——>sevlet(一个页面)

**c**，jsp:plugin动作

4，include指令：

服务器将被包含文件内容每字节插入到主页面中，然后将产生的页面作为单个JSP页 面进行处理。

5，jsp:include动作：在“请求”期间将其他页面的“输出”包含进来。这样的行为和 RequestDispatcher类的include方法完全相同。

优点：包含页面被更改，无需对主页面做修改。

缺点：被包含页面不能使用任何有可能影响主页面的jsp构造。

也就是不能将完整的HTML文档作为被包含页面，只能含有适合于出现在文件插入 点的HTML标签。

参数(使用jsp:param动作元素增加请求参数)：

在被包含页面与主页面使用相同的请求对象。可以在被包含页面“看到”主页面的 参数，而在主页面“看不到”被包含页面的参数。

**a**，page属性：指定所包含的页面，应该是指向某种资源的相对URL.

相对URL不以/开头，则表示为相对于主页面的位置。以/开头表示相对于WEB应 用的根目录。

如果由服务器来处理该资源，按相对于当前web应用进行解释；只有浏览器处理 该资源时，按相对于服务器的根目录来解释。

对于/开头的URL，服务器总是相对于当前web应用对它进行解释，浏览器总是相 对于服务器的根目录来对它进行解释。

可以将要包含的页面放在WEB-INF目录下面，对客户端隐藏。

**b，**flush属性：指定将页面包含进来之前是否应该清空主页面的输入流(默认false)。

**6，include指令和jsp:include动作的区别：**

调用的时间：

jsp:include动作在“请求期间”被包含。

include指令在“转换期间”被包含。

包含的内容：

jsp:include动作是包含jsp页面或servlet的“输出”和HTML文本。

include指令是文件的实际内容。

最终转换的servle个数：

jsp:include动作在请求期间，会将被请求页面转译为独立的servlet。和主页面的 servlet对应。

include指令在转换期间将被包含页面的内容插入到主页面中，得到主页面转译成

的servlet。

可否设置影响主页面的响应报头和参数：

jsp:incurred动作不可以，include指令可以。

更新：

jsp:include动作是在请求期间被插入所以他的更改不需要更新主页面。

include指令的更新意味着主页面servlet的更新。

1. <jsp:plugin>、<jsp:params>、<jsp:fallback>动作指令（一般连起来使用）：
2. <jsp:plugin>动作元素用来在JSP中嵌入Java插件；比如Applet。

语法：

1. **<jsp:plugin>**
2. type="bean | applet"
3. name="Applet名称"
4. code="java类名"
5. codebase="Java类所在目录"
6. align="对齐方式"
7. height="高度"
8. width="宽度"
9. hspace="水平间距"
10. vspace="垂直间距"
11. archive="预先加载的类列表"
12. jreversion="JRE版本"
13. iepluginurl="URL"
14. nspluginurl="URL"
15. **</jsp:plugin>**

属性：

属性 属性作用

type 用来指定插件类型，可以是Bean和Applet

name 用来指定Applet或Bean名称

code 用来指定所执行的Java类名，必须以class结尾

codebase 用来指定所执行的Java类所在的目录

align 用来指定Applet或Bean显示时的对齐方式

height 用来指定Applet或Bean显示时的高度

width 用来指定Applet或Bean显示时的宽度

hspace 用来指定Applet或Bean显示时的距离屏幕左右的距离，单位是像素

vspace 用来指定Applet或Bean显示时的距离屏幕上下的距离，单位是像素

archive 用来指定Applet或Bean执行前预先加载的类的列表

iepluginurl 用来指定IE用户能够使用的JRE下载地址

nspluginurl 用来指定Netscape Navigator用户能够使用的JRE下载地址

1. <jsp:params>动作元素用来给Bean或Applet传递参数；

语法：

1. **<jsp:plugin>**
2. **<jsp:params>**
3. **<jsp:param** name="参数名" value="参数值" **/>**
4. **</jsp:params>**
5. **</jsp:plugin>**

属性：

1. <jsp:fallback>动作元素用来指定当浏览器不支持或无法启动的Bean或Applet时，在 页面上打印输出错误提示信息。

语法：

1. **<jsp:plugin>**
2. **<jsp:fallback>**
3. 错误信息
4. **</jsp:fallback>**
5. **</jsp:plugin>**

四、useBean动作(jsp页面使用javaBean的第二种方式):  
**<jsp:useBean id="对象名" class="包名.类名" scope="作用范围(request/page/application/session)"/>**  
 作用域默认为page(本页面有效).

五、getProperty动作(name为useBean动作中的id).  
 从对象中取出属性值：

**<jsp:getProperty name="javaBean对象" property="javaBean对象属性名" />**

六、setProperty动作(name为useBean动作中的id):  
 为对象设置属性值：

<jsp:setProperty name="javaBean对象" property="javaBean对象属性名"value=http://www.hake.cc/kf/201109/"值"/>  
 为对象设置属性值:

<jsp:setProperty property="javaBean对象属性名" name="javaBean对象" param="username"/>  
(param="username" 相当于 value=http://www.hake.cc/kf/201109/<%=request.getParameter("username")%>)  
 