[**Java Web中的相对路径与绝对路径总结**](http://www.cnblogs.com/jichunhu/archive/2013/01/31/2887042.html)

**1、基本概念的理解**

绝对路径：绝对路径就是你的主页上的文件或目录在硬盘上真正的路径，(URL和物理路径)例如：C:\xyz\test.txt 代表了test.txt文件的绝对路径。<http://www.sun.com/index.htm>也代表了一个URL绝对路径。

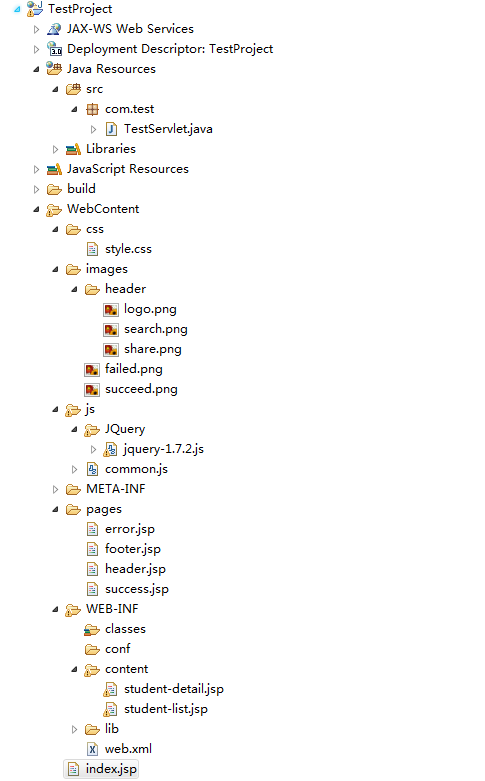
相对路径：相对与某个基准目录的路径。包含Web的相对路径（HTML中的相对目录），

在项目中，"/"代表Web站点的根目录，"./"代表当前目录, "../"代表上级目录， http://www.cnblogs.com/代表上上级目录，以此类推。

例如：C：\test\AC\下面有a.html和c.html, C：\test\B\下面有b.html，则在a.htm中访问b.html绝对路径为C：\test\B\b.html，相对路径为../B/b.html;访问c.html绝对路径为C：\test\AC\c.html，相对路径为./c.html或者c.html。

**2、关于JSP/Servlet中的相对路径和绝对路径。**

2.1 服务器端的地址



假设我们的项目访问地址为：<http://172.0.0.1/TestProject/>

　　服务器端的相对地址指的是相对于你的web应用的地址，这个地址是在服务器端解析的（不同于html和javascript中的相对地址，他们是由客户端浏览器解析的）也就是说这时候在jsp和servlet中的相对地址应该是相对于你的web应用，即相对于<http://172.0.0.1/TestProject/>的。

　　其用到的地方有：   
forward：servlet中的request.getRequestDispatcher(address);这个address是在服务器端解析的，所以，你要forward到error.jsp应该这么写：  
request.getRequestDispatcher(“pages/error.jsp”)，这个/相对于当前的web应用TestProject，其绝对地址就是：<http://172.0.0.1/TestProject/>pages/error.jsp。   
sendRedirect：在jsp中<%response.sendRedirect("/pages/error.jsp ");

在Servlet中，如果要访问WEB-INF下面的jsp页面时，必须加上/WEB-INF这个路径，因为Servlet的路径是相对服务器端地址的，即<http://172.0.0.1/TestProject/>，例如：request.getRequestDispatcher("/WEB-INF/content/student-detail.jsp")

2.2客户端的地址   
  
所有的html页面中的相对地址都是相对于服务器根目录([http://172.0.0.1/](http://192.168.0.1/))的，而不是(跟目录下的该Web应用的目录)<http://172.0.0.1/TestProject/>的。 Html中的form表单的action属性的地址应该是相对于服务器根目录([http://172.0.0.1/](http://192.168.0.1/))的，所以，如果提交到error.jsp为：action="<%=request.getContextPath()%>/pages/error.jsp”；提交到servlet为action＝"<%=request.getContextPath()%>/handleservlet"; Javascript也是在客户端解析的，所以其相对路径和form表单一样。

因此，一般情况下，在JSP/HTML页面等引用的CSS,Javascript.Action等属性前面最好都加上<%=request.getContextPath()%>,以确保所引用的文件都属于Web应用中的目录。另外，应该尽量避免使用类似".","./","http://www.cnblogs.com/"等类似的相对该文件位置的相对路径，因为这样当文件移动时，很容易出问题。

**2.3** 站点根目录和系统目录

站点根目录：在Web项目中“/”是表示站点根目录的相对路径，例如在jsp页面中引入<link rel="stylesheet" type="text/css" href="/css/style.css" />  
<script type="text/javascript" src="/js/common.js"></script>，style.css和common.js的路径分别为[http://172.0.0.1/](http://172.0.0.1/TestProject/)css/style.css和([http://172.0.0.1/](http://172.0.0.1/TestProject/)js/common.js。因为/css/…. 是相对于站点根目录([http://172.0.0.1/](http://172.0.0.1/TestProject/))的相对路径。 这样jsp是无法访问到css或js文件的。

系统目录：众所周知，JSP是Servlet的一种特殊形式，每个JSP页面就是一个Servlet实例——JSP页面由系统编译成Servlet，Servlet再负责响应用户请求。JSP其实也是Servlet的一种简化，使用JSP时，其实还是使用Servlet，因为Web应用中的每个JSP页面都会由Servlet容器生成对应的Servlet。也就说jsp所在的目录还是服务端目录，即JSP地址还是相对系统根目录<http://172.0.0.1/TestProject/>的。

也就是说站点根目录为[http://172.0.0.1/](http://172.0.0.1/TestProject/)，系统根目录为<http://172.0.0.1/TestProject/>，这跟客户端地址和服务端地址是相似的。

**3、在JSP中引用js/css/的两种方式**

第一种：站点根目录下绝对路径的方式

假设我们项目的URL为：<http://172.0.0.1/TestProject/>index.jsp。

为了在以后更改工程名时也能正确访问，我们需要加 <%basePath=request.getContextPath() %>  
如:<script src="<%=request.getContextPath() %> /js/common.js"></script>

写<%=request.getContextPath() %>太麻烦，可以在每一个jsp文件顶部加入以下内容后，

1. <%
2. String path = request.getContextPath();
3. String basePath = request.getScheme()+"://" +request.getServerName()+":"+request.getServerPort()+path+"/" ;
4. %>
5. <base href="<%=basePath%>" >

就可直接使用 <script src=" /js/common.js"></script>了。

第二种：系统根目录下相对路径的访问方式

当在jsp中引入css/js时，如果其相对路径相对于当前jsp文件的，而当前的jsp的相对路径又是相对于系统目录<http://172.0.0.1/TestProject/>的。

所以在上面的引入方式中去掉“/”即可。即<link rel="stylesheet" type="text/css" href="css/style.css" /><script type="text/javascript" src="js/common.js"></script>。在jsp页面引入图片也是如此<img src="images/succeed.png"/>。如果一定要使用站点根目录的话就这样使用<img src="<%=request.getContextPath()%>/images/succeed.png"/>，这两种方式都是可以的。

**4、在Servlet/Action中跳转到相应的Jsp页面的方式**

第一种情况：假如jsp页面是可以直接访问到的，即是放在WEB-INF外面的，假设该jsp页面访问地址为<http://172.0.0.1/TestProject/>pages/test.jsp。则Servlet中通过request.getRequestDispatcher("/pages/test.jsp").forward(request, response)这种方式即可跳转到test.jsp。

第二种情况：假如该jsp页面是不可以直接访问的，是在WEB-INF里面的。假设该jsp页面所在的目录为<http://172.0.0.1/TestProject/>WEB-INF/content/student-list.jsp。则Servlet中通过request.getRequestDispatcher("/WEB-INF/content/student-list.jsp").forward(request, response)跳转。在Struts2中通过配置<result name="success">/content/student-list.jsp</result>即可访问。

**5、Forward、Redirect及Include的区别**

首先,sendRedirect与forward,include的最大区别是request,response. sendRedirect产生新的请求与响应，服务器收到请求后发送一个状态头给客户，客户将再次请求，就有两次网络通行的来往。redirect是服务端根据逻辑,发送一个状态码,告诉浏览器重新去请求那个地址.所以地址栏显示的是新的URL.所以redirect等于客户端向服务器端发出两次request，同时也接受两次response.所以redirect可以重定向到外部的资源,如：response.sendRedriect("http://www.baidu.com"); 当Servlet发送“http://www.baidu.com”到服务器时，服务器再将该地址返回客户端，此时浏览器URL为“http://www.baidu.com”，然后客户端再去访问“http://www.baidu.com”这个地址。

其次，forward与include基本上都是转发到context内部的资源。它们是服务器内部重定向，程序接收到请求后重新定向到另一个程序，而客户端并不知晓。从地址栏显示来说forward与include是服务器请求资源,服务器直接访问目标地址的URL,把那个URL的响应内容读取过来,然后把这些内容

再发给浏览器.浏览器根本不知道服务器发送的内容从哪里来的,所以它的地址栏还是原来的地址. forward与include会将 request state、bean、等信息带到下一个jsp页面；

使用getAttribute（）来取得前一个jsp所放的信息，redirect 是送到客户端后再次request，因此上一个jsp的信息不被保留。

最后，forward与include的区别就是，forward方法是把请求的内容转发到另外的一个servlet。而include是把另一个servlet处理过后的内容拿过来.

使用方式：

**1.** sendRedirect

servlet和jsp里面一样

response.sendRedirect();

**2.** include

1) servlet里面

request.getRequestDispatcher( "jsp2.jsp" ).include(request, response);

2) jsp里面

<jsp:include page= "include.jsp" />

**3.** forword

1) servlet里面

request.getRequestDispatcher( "jsp2.jsp" ).forward(request, response);   
2) jsp里面

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