2018-1-18

下面笔记来自[孤傲苍狼](http://www.cnblogs.com/xdp-gacl/)的博客

**一、什么是JSP？**

　　JSP全称是Java Server Pages，它和servle技术一样，都是SUN公司定义的一种用于开发动态web资源的技术。  
　　JSP这门技术的最大的特点在于，写jsp就像在写html，但它相比html而言，html只能为用户提供静态数据，而Jsp技术允许在页面中嵌套java代码，为用户提供动态数据。

**二、JSP原理**

**2.1、Web服务器是如何调用并执行一个jsp页面的？**

　　浏览器向服务器发请求，不管访问的是什么资源，其实都是在访问Servlet，所以当访问一个jsp页面时，其实也是在访问一个Servlet，服务器在执行jsp的时候，首先把jsp翻译成一个Servlet，所以我们访问jsp时，其实不是在访问jsp，而是在访问jsp翻译过后的那个Servlet，例如下面的代码：

**index.jsp**

<%@ page language="java" import="java.util.\*" pageEncoding="UTF-8"%>

<%

String path = request.getContextPath();

String basePath = request.getScheme()+"://"+request.getServerName()+":"+request.getServerPort()+path+"/";

%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<html>

<head>

<base href="<%=basePath%>">

<title>First Jsp</title>

</head>

<body>

<%

out.print("Hello Jsp");

%>

</body>

</html>

**2.2、Jsp页面中的html排版标签是如何被发送到客户端的？**

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<html>

<head>

<base href="http://localhost:8080/JavaWeb\_Jsp\_Study\_20140603/">

<title>First Jsp</title>

</head>

<body>

Hello Jsp

</body>

</html>

都是在\_jspService方法中使用如下的代码输出给浏览器的：

out.write('\r');

out.write('\n');

String path = request.getContextPath();

String basePath = request.getScheme()+"://"+request.getServerName()+":"+request.getServerPort()+path+"/";

out.write("\r\n");

out.write("\r\n");

out.write("<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.01 Transitional//EN\">\r\n");

out.write("<html>\r\n");

out.write(" <head>\r\n");

out.write(" <base href=\"");

out.print(basePath);

out.write("\">\r\n");

out.write(" \r\n");

out.write(" <title>First Jsp</title>\r\n");

out.write("\t\r\n");

out.write(" </head>\r\n");

out.write(" \r\n");

out.write(" <body>\r\n");

out.write(" ");

out.print("Hello Jsp");

out.write("\r\n");

out.write(" </body>\r\n");

out.write("</html>\r\n");

在jsp中编写的java代码和html代码都会被翻译到\_jspService方法中去，在jsp中编写的java代码会原封不动地翻译成java代码，如<%out.print("Hello Jsp");%>直接翻译成out.print("Hello Jsp");，而HTML代码则会翻译成使用**out.write("<html标签>\r\n");**的形式输出到浏览器。在jsp页面中编写的html排版标签都是以**out.write("<html标签>\r\n");**的形式输出到浏览器，浏览器拿到html代码后才能够解析执行html代码。

**2.3、Jsp页面中的java代码服务器是如何执行的？**

　　在jsp中编写的java代码会被翻译到\_jspService方法中去，当执行\_jspService方法处理请求时，就会执行在jsp编写的java代码了，所以Jsp页面中的java代码服务器是通过调用\_jspService方法处理请求时执行的。

**2.4、Web服务器在调用jsp时，会给jsp提供一些什么java对象？**

　　查看\_jspService方法可以看到，Web服务器在调用jsp时，会给Jsp提供如下的8个java对象

PageContext pageContext;

HttpSession session;

ServletContext application;

ServletConfig config;

JspWriter out;

Object page = this;

HttpServletRequest request,

HttpServletResponse response

其中page对象，request和response已经完成了实例化，而其它5个没有实例化的对象通过下面的方式实例化

1 pageContext = \_jspxFactory.getPageContext(this, request, response,null, true, 8192, true);

2 application = pageContext.getServletContext();

3 config = pageContext.getServletConfig();

4 session = pageContext.getSession();

5 out = pageContext.getOut();

 这8个java对象在Jsp页面中是可以直接使用的，如下所示：

<%

session.setAttribute("name", "session对象");//使用session对象,设置session对象的属性

out.print(session.getAttribute("name")+"<br/>");//获取session对象的属性

pageContext.setAttribute("name", "pageContext对象");//使用pageContext对象,设置pageContext对象的属性

out.print(pageContext.getAttribute("name")+"<br/>");//获取pageContext对象的属性

application.setAttribute("name", "application对象");//使用application对象,设置application对象的属性

out.print(application.getAttribute("name")+"<br/>");//获取application对象的属性

out.print("Hello Jsp"+"<br/>");//使用out对象

out.print("服务器调用index.jsp页面时翻译成的类的名字是："+page.getClass()+"<br/>");//使用page对象

out.print("处理请求的Servlet的名字是："+config.getServletName()+"<br/>");//使用config对象

out.print(response.getContentType()+"<br/>");//使用response对象

out.print(request.getContextPath()+"<br/>");//使用request对象

%>

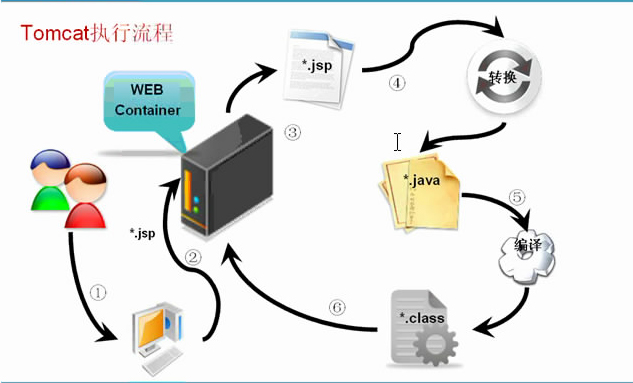
运行结果如下：

**2.5、Jsp最佳实践**

　　Jsp最佳实践就是jsp技术在开发中该怎么去用。

　　不管是JSP还是Servlet，虽然都可以用于开发动态web资源。但由于这2门技术各自的特点，在长期的软件实践中，人们逐渐把servlet作为web应用中的控制器组件来使用，而把JSP技术作为数据显示模板来使用。其原因为，程序的数据通常要美化后再输出：让jsp既用java代码产生动态数据，又做美化会导致页面难以维护。让servlet既产生数据，又在里面嵌套html代码美化数据，同样也会导致程序可读性差，难以维护。因此最好的办法就是根据这两门技术的特点，让它们各自负责各的，servlet只负责响应请求产生数据，并把数据通过转发技术带给jsp，数据的显示jsp来做。

**2.6、Tomcat服务器的执行流程**

****

第一次执行：

1. 客户端通过电脑连接服务器，因为是请求是动态的，所以所有的请求交给WEB容器来处理
2. 在容器中找到需要执行的\*.jsp文件
3. 之后\*.jsp文件通过转换变为\*.java文件
4. \*.java文件经过编译后，形成\*.class文件
5. 最终服务器要执行形成的\*.class文件

第二次执行：

1. 因为已经存在了\*.class文件，所以不在需要转换和编译的过程

修改后执行：

       1.源文件已经被修改过了，所以需要重新转换，重新编译。