JSP小记

1.JSP和HTML之间使用HTML标签的name属性来确定关系

比如HTML中一个输入框<input type="text" name="username">

根据HTTP协议，这部分数据将以http get请求或http post请求的形式传入servlet中，

JSP使用内置对象request的getParameter方法对name进行定位，获取用户input的内容

request.getParameter(“username”)返回String

# 2. request.getRequestDispatcher方法的作用是什么

这一步之前的工作是对提交的request做处理，这一步（这句话）是表示：处理完了，分发到下一个[JSP](https://www.baidu.com/s?wd=JSP&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YdPWRzmWczn1fkmHRsnHcY0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnH0LnW0srHnv)页面或者下一个Action继续处理。  
会有**forward**()和**redirect**()**两种**情况，**forward()是request中的参数继续传递，redirect()则是重新生成request了**。

3. **${pageContext.request.contextPath}**等价于**<%=request.getContextPath()%>** 或者可以说是**<%=request.getContextPath()%**>的EL版 意思就是取出部署的应用程序名或者是当前的项目名称

比如我的项目名称是ajax01 在浏览器中输入为**http://localhost:8080/ajax01/login.jsp**

**我们通过${pageContext.request.contextPath**}或**<%=request.getContextPath()%>**取出来的就是**/ajax01**,而"/"代表的含义就是**http://localhost:8080**

所以我们项目中应该这样写**${pageContext.request.contextPath}/login.jsp**

4.不要忘了**导包**和**修改字符编码**

要么导入包，要么使用完全限定名（和普通java类的规则一样）

如：<%@ page import="java.util.List"%>

pageEncoding是jsp文件本身的编码  
contentType的charset是指服务器发送给客户端时的内容编码

<%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*

pageEncoding=*"UTF-8"*%>

<!-- 在JSP页面使用时必须导包，因为jsp实际上背后是一个servlet类，包要自己导，不然找不到你想使用的类 -->

<%@ page import=*"java.util.List"*%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<!—- 有了上面的字符编码指定，这里还需要同步 -->

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=UTF-8"*>

<title>Insert title here</title>

</head>

<body>

<h1 align=*"center"*>Beer Recommendations JSP</h1>

<p>

<!-- 在JSP中嵌入java代码脚本 -->

<%

List<String> styles = (List<String>)request.getAttribute("styles");

**for** (String brand : styles) {

out.println("<br>Got brands： " + brand);

}

%>

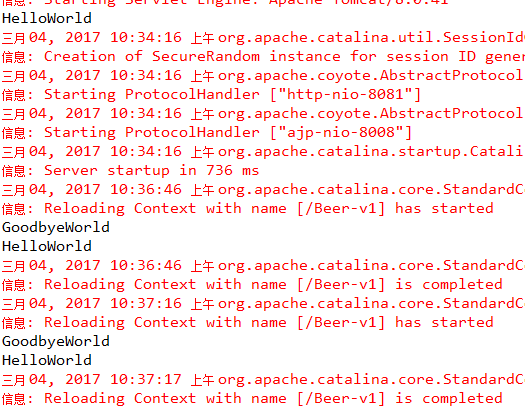
</body>

</html>

5. 开发测试过程中，**只是修改JSP的话，容器能够热刷新**，但是配置文件如web.xml修改了就不能了，要重新部署项目才行。原因就是**servlet只加载一次**

在eclipse+tomcat平台测试时，我们加入了servletListener监听者，使其分别在servlet初始化时和销毁时打印HelloWorld和GoodByeWorld，

结果**，每当我们修改一次JSP，平台都将自动重新编译对应的servlet，并将原来的servlet销毁，重新加载初始化新的这个servlet**。



6.