Java Server Page

JSP程序设计(上)

isszym sysu.edu.cn

2016.11.27

概述

- ## 什么是**JSP**

 JSP(Java Server Pages)是由Sun Microsystems公司开发的一种实现普通静态HTML和动态HTML混合编码的技术。

- ## Servlet和JSP

  Servlet是最早开始使用Java Web编程方式。Servlet编程通过扩展预定义的HttpServlet类响应HTTP的Get和Post请求。JSP建立在Servlet类的基础上。它分离了表现逻辑和业务逻辑。
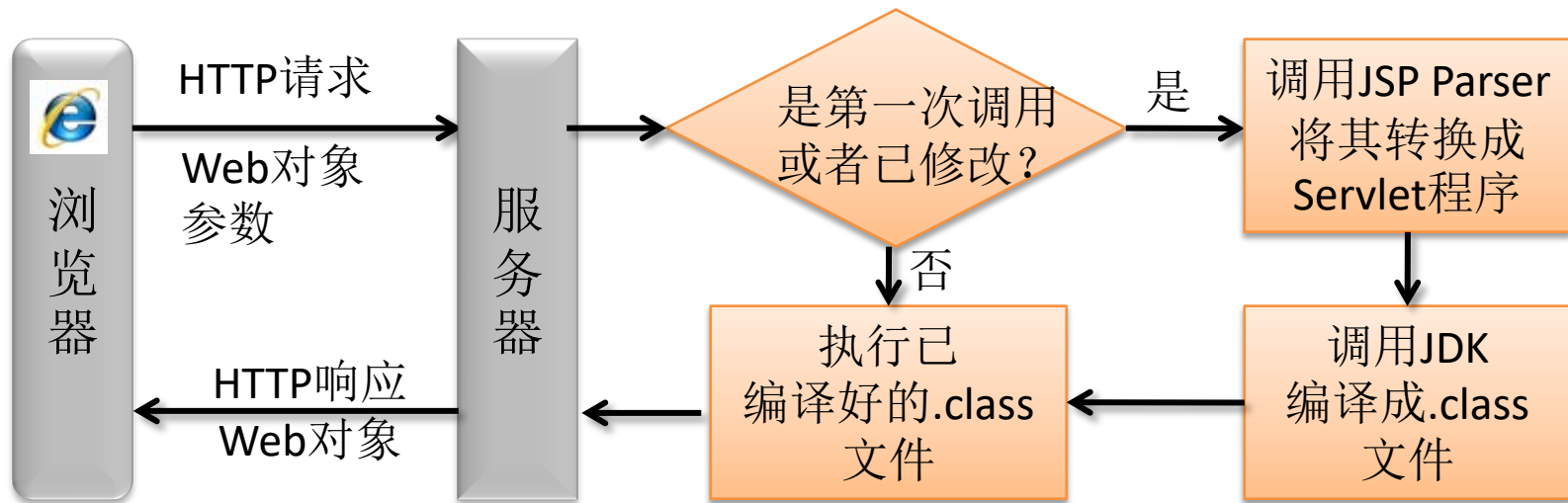
  [JSP教程](#)

JSP Web服务器

　　Web服务器是接收HTTP请求并返回HTTP响应的软件。下面是一种JSP web服务器结构，Apache Tomcat为Web服务器，它是基于Java虚拟机的。这个结构采用了MySQL数据库。如果编程使用Eclipse，注意其内部Tomcat与外部的Tomcat可能会冲突（只能启动一个）。

JSP Web服务器

Apache
Tomcat

Java
虚拟机

mysql
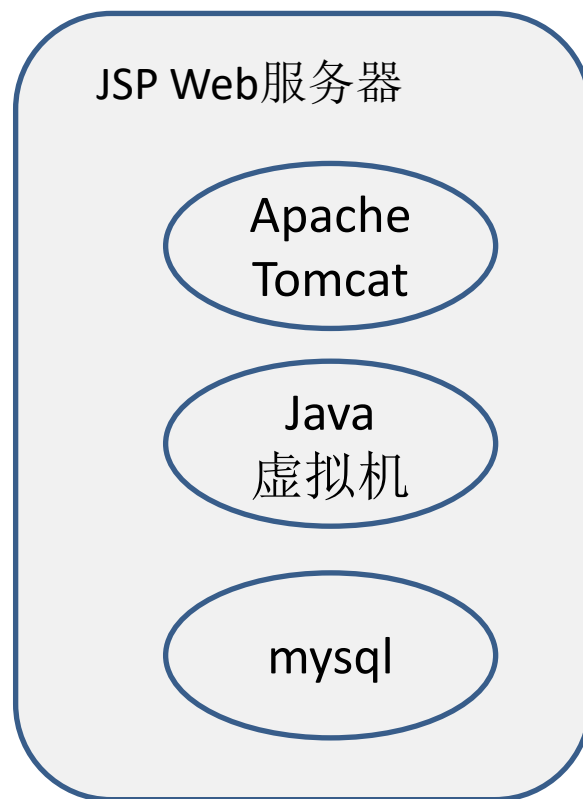
JSP工作原理

（1）当Web服务器收到对JSP文件(JSP页面)的HTTP请求，如果所请求的JSP文件是修改后的第一次访问，则调用JSP Parser将其转换成Servlet程序（.java），然后调用JSDK编译成Servlet字节码（.class）。

（2）Java虚拟机执行Servlet字节码发回HTTP响应。

（3）如果有多个HTTP请求发给同一个JSP文件，则Tomcat服务器会为每一个请求启动一个线程执行该文件对应的Servlet字节码。

第一个Servlet程序

下面的Servlet程序(Hello.java)继承了系统的HttpServlet类.

http://172.18.187.230:8080/jsp/servlet/Hello

```java
package com.group.servlet;
import java.io.*;
import java.util.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class Hello extends HttpServlet {
    @SuppressWarnings("deprecation")
    @Override
    public void doGet(HttpServletRequest req, HttpServletResponse resp)
            throws ServletException, IOException  {
        resp.setContentType("text/html");
        PrintWriter out = resp.getWriter();
        out.println("<html><head><title>First Servlet Program</title></head>");
        out.println("<body>");
        out.println("<h1>Hello World<h1>");
        out.println("<h2>" + new Date().toLocaleString()+ "</h2>");
        out.println("</body>");
        out.println("</html>");
        out.flush();
    }
}
            doPost
```

Hello World

2016-11-6 23:18:21
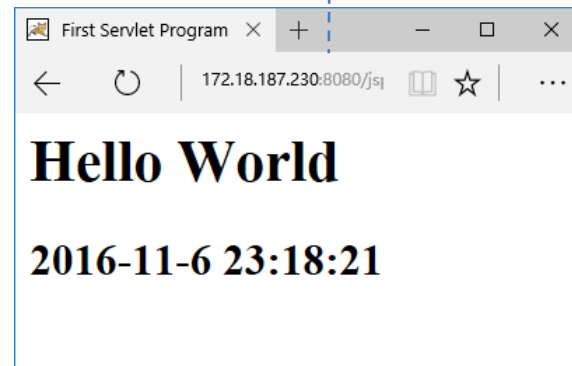
C:\java>javac -classpath .\servlet-api.jar Hello.java
* servlet-api.jar在tomcat的lib目录中

http://172.18.187.230:8080/jsp/servlet/Hello

- ✓ 📁 Program Files
  - ✓ 📁 Apache Software Foundation
    - ✓ 📁 Tomcat 8.5
      - 📁 bin
      - 📁 conf ———————— 放置全局配置文件
      - 📁 lib ———————— 公用库(*.jar)
      - 📁 logs
      - 📁 temp
      - ✓ 📁 webapps
        - 📁 docs
        - 📁 examples
        - 📁 host-manager
        - ✓ 📁 jsp
          - 📁 META-INF
          - ✓ 📁 WEB-INF
            - ✓ 📁 classes
              - ✓ 📁 com
                - ✓ 📁 group
                  - 📁 servlet
              - 📁 lib
        - 📁 manager
        - 📁 ROOT
      - 📁 work

server.xml
web.xml
config.xml

webapps下的子目录都是虚拟目录。要定义其它虚拟目录，需要修改配置文件server.xml。

*jsp不需要配置classpath

存放.jsp文件，可以建子目录

📄 context.xml

📄 web.xml

context.xml文件内容：

`<?xml version="1.0"?>`
`<Context reloadable="true"/>`

*修改后自动重载，不需要重启Tomcat服务器

📄 Hello.class

私用库

- ☕ commons-fileupload-1.2.1.jar
- ☕ commons-io-1.4.jar
- ☕ jstl-1.2.jar
- ☕ mysql-connector-java-5.1.36-bin.jar
- ☕ standard-1.1.2.jar

也可以放在公用库中

*http://172.18.187.230:8080/jsp/**servlet/Hello***

web.xml

```xml
<?xml version="1.0" encoding="ISO-8859-1"?>
<web-app>
    <description>Servletand JSP Examples. </description>
    <display-name>Servlet and JSP Examples</display-name>
    <servlet>                        （3）
        <servlet-name>HelloWorld</servlet-name>
        <servlet-class>com.group.servlet.Hello</servlet-class>
    </servlet>                       （4）
    <servlet-mapping>       （2）
        <servlet-name>HelloWorld</servlet-name>
        <url-pattern>/servlet/Hello</url-pattern>
    </servlet-mapping>
                                （1）URL
    <servlet>
        <servlet-name>Requ</servlet-name>
        <servlet-class>serv.RequestInfo</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>Requ</servlet-name>
        <url-pattern>/servlet/request</url-pattern>
    </servlet-mapping>
</web-app>
```
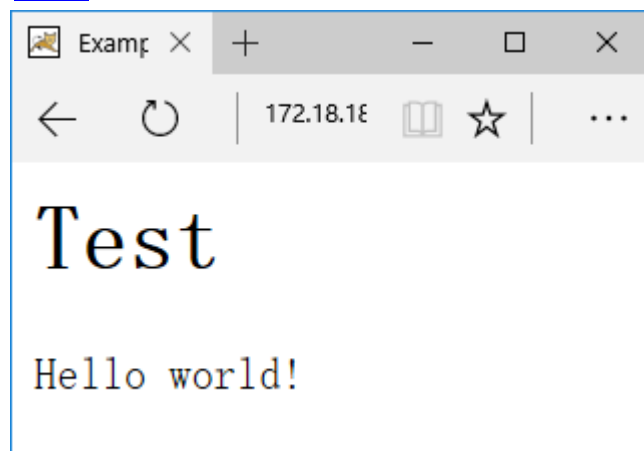
一个servlet

另一个servlet

第一个JSP程序

hello.jsp

指令元素

生成http响应的指示，例如，要使用的包，http响应的头部content-type.

HTML模版

脚本片段
(scriptlet)

HTML模版

```jsp
<%@ page contentType="text/html;charset=gb2312"%>
<html>
<head>
<title>Example</title>
</head>
<body>
<h1> Test </h1>
<p>
<%
    out.print("Hello world!");
%>
</p>
</body>
</html>
```

网址

Examp × + — □ ×

← ○ | 172.18.18 ☆ | …

Test

Hello world!

Java程序放在<% %>之间。out为一个JSP的系统对象。

如果只输出一个表达式而不是语句，可以采用**<%=exp%>**输出，例如：**<%="Hello World!"%>**。

hello.jsp会被**自动转换**为servlet程序hello_jsp.java，放在work子目录下，然后再编译成字节码文件hello_jsp.class，并在Java虚拟机上运行，最后把运行的输出结果用**http响应**传送到客户端。

```java
public final class hello_jsp extends
            org.apache.jasper.runtime.HttpJspBase{
    ...可以在这里声明<%! %>内定义全局变量、函数和类
    public void _jspService(... request,... response)
        定义了jsp内置对象：out、session、application、config
            ...
        out.write("<html>\r\n");
        out.write("<head>\r\n");
        out.write("<title>Hello world</title>\r\n");
        out.write("</head>\r\n");
        out.write("<body>\r\n");
        out.write("<h1>Test</h1>\r\n");
        out.write("<p>\r\n");
        out.print("Hello world!");
        out.write("</p>\r\n");
        out.write("</body>\r\n");
        out.write("</html>\r\n");
        out.write("\r\n");
            ...
    }
}
```

* out.print()只能输出字符，而 out.write还可以输出字节流

* hello_jsp.java的详细内容见附录

为什么for语句里面可以加入html的语句？如何解释？

hellofor.jsp

```jsp
<%@ page contentType="text/html;charset=gb2312"%>
<html>
<head>
<title>Example</title>
</head>
<body>
<h1>Test</h1>
<%for(int i=0;i<10;i++){%>
    <p>Hello!<%=i%></p>
<%}%>
</body>
</html>
```

Test

Hello!0   网址

Hello!1

Hello!2

Hello!3

Hello!4

Hello!5

Hello!6

Hello!7

Hello!8

Hello!9

```
    ...
out.write("\r\n");
out.write("<html>\r\n");
out.write("<head>\r\n");
out.write("<title>Example</title>\r\n");
out.write("</head>\r\n");
out.write("<body>\r\n");
out.write("<h1>Test</h1>\r\n");
for(int i=0;i<10;i++){
    out.write("\r\n");
    out.write("<p>Hello!");
    out.print(i);
    out.write("</p>\r\n");
}
out.write("\r\n");
out.write("</body>\r\n");
out.write("</html>\r\n");
out.write("\r\n");
out.write("\r\n");
out.write("\r\n");
    ...
```

也可以这样编程，先计算出变量值，再把结果代入Web页面，实现Java代码与HTML代码分离：

hellofor2.jsp

```jsp
<%@ page contentType="text/html;charset=gb2312"%>
<% String s="";
   for(int i=0;i<10;i++){
     s=s+"<p>Hello!"+i+"</p>";
   }
 %>
<html>
<head>
<title>Example</title>
</head>
<body>
<h1>Test</h1>
<%=s%>
</body>
</html>
```

网址

测试

声明

　JSP程序的函数和类只能在**声明<%!... %>**中定义，它们会被分别转换为servlet类的方法和内部类。声明中定义的变量会被所有用户线程所共享。

```jsp
<%@ page contentType="text/html;charset=gb2312"%>
<%! int cnt=0;
    int getSum(int n){
      int sum=0;
      for(int i=1;i<=n;i++){
          sum=sum+i;
      }
      return sum;
    }
 %>
<html>
<head>
<title>getSumFunc</title>
</head>
<body>
  <h1>getSumFunc</h1>
  <%=getSum(100)%>
</body>
</html>
```


getSumFunc

5050

[网址](网址)

　在声明中不能使用jsp内置对象，例如：out、session、request、response等。这些对象可以作为参数传入到声明中。这些对象的具体类型见附录的servlet程序。

getSumClass.jsp

```jsp
<%@ page contentType="text/html;charset=gb2312"%>
<%! class Sum{
        int getSum(int n){
            int sum=0;
            for(int i=1;i<=n;i++){
                sum=sum+i;
            }
            return sum;
        }
    }
%>
<html>
<head>
<title>getSumClass</title>
</head>
<body>
    <h1>getSumClass</h1>
    <% Sum sum=new Sum();
        out.print(sum.getSum(100));
    %>
</body>
</html>
```

getSumClass   ×   +   —   □   ×

←   ○   172.18.187.230:8080/j   ☆   ...

getSumClass

5050   网址

```java
//Sum.java
package com.group.jsp;
public class Sum {
    public int getSum(int n){
        int sum=0;
        for(int i=1;i<=n;i++){
            sum=sum+i;
        }
        return sum;
    }
}
```

```jsp
//getSumClass1.jsp
<%@ page import="com.group.jsp.Sum"
contentType="text/html;charset=gb2312"%>
<html>
<head>
<title>getSumClass</title>
</head>
<body>
  <h1>getSumClass</h1>
  <%
    Sum sum=new Sum();
    out.print(sum.getSum(100));
  %>
</body>
</html>
```
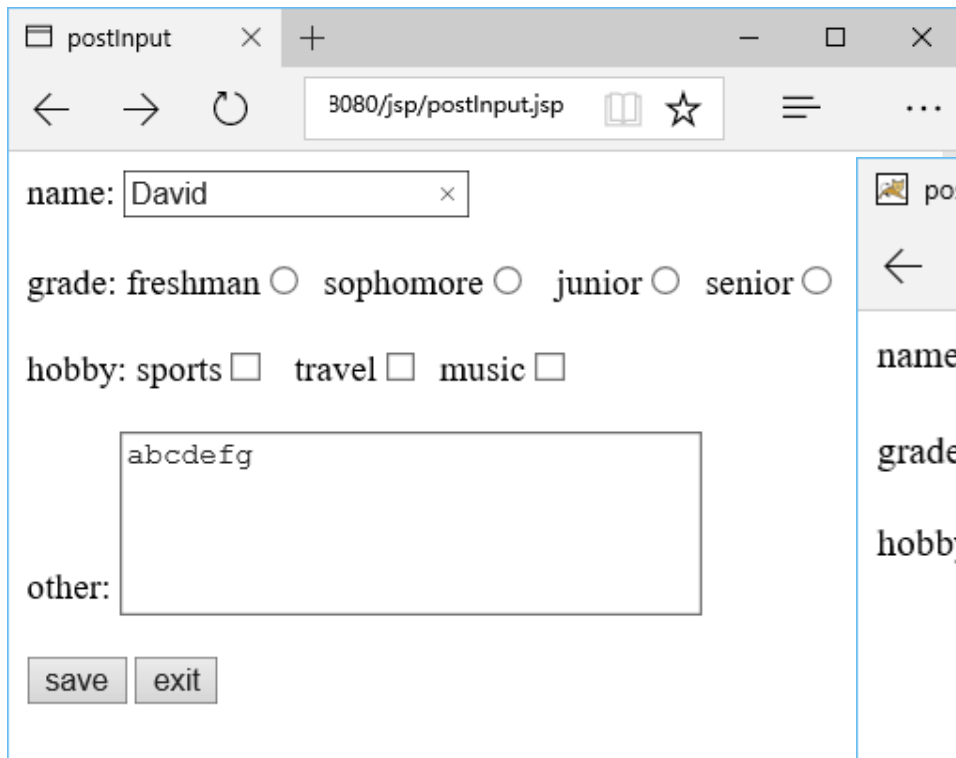
Sum.class文件放在
tomcat\webapps\jsp\WEB-INF\classes\
com\group\jsp目录下

- JSP的类一定要指定包名，不能用默认包
- 一个网页可以出现多个import：
    `<%@ page import="a.b.*" %>`
    `<%@ page import="x.y.*" %>`
- 可以合并，用逗号隔开：
    `<%@ page import="a.b.*,x.y.*" %>`

如何取得提交的数据

- **提交给另一个网页（必须是动态网页）**

初始页面

输入后页面

postInput.jsp

点击 ⟶

*disabled的数据不会被提交

点击后进入的页面：　　　　　postShow.jsp

postInput.jsp

```jsp
<%@ page language="java" import="java.util.*"
         contentType="text/html; charset=utf-8" pageEncoding="utf-8"%>
<!DOCTYPE HTML>
<html>
  <head><title>postInput</title></head>
  <body>
    <form action="postShow.jsp?user=wang" method="post">
        <input type="hidden" name="stuid" value="1400001" />
        name: <input type="text" name="name" value="David"/>
        grade: freshman<input type="radio" name="grade" value="freshman"/>
               sophomore<input type="radio" name="grade" value="sophomore"/>
               junior<input type="radio" name="grade" value="junior"/>
               senior<input type="radio" name="grade" value="senior"/>
        hobby: sports<input type="checkbox" name="hobby1" value="sports"/>
               travel<input type="checkbox" name="hobby2" value="travel"/>
               music<input type="checkbox" name="hobby3" value="music"/>
        other: <textarea rows="5" cols="30" name="other">abcdefg</textarea>
        <input type="submit"  name="submit1" value="save" />
        <input type="submit"  name="submit2" value="exit" />
    </form>
  </body>
</html>
```
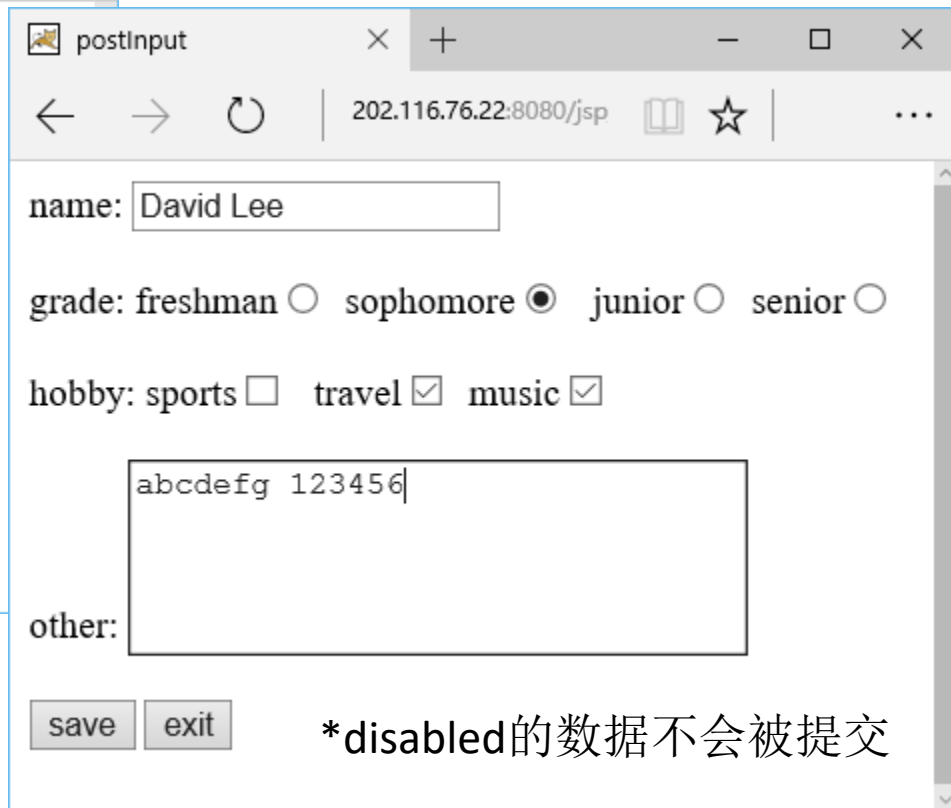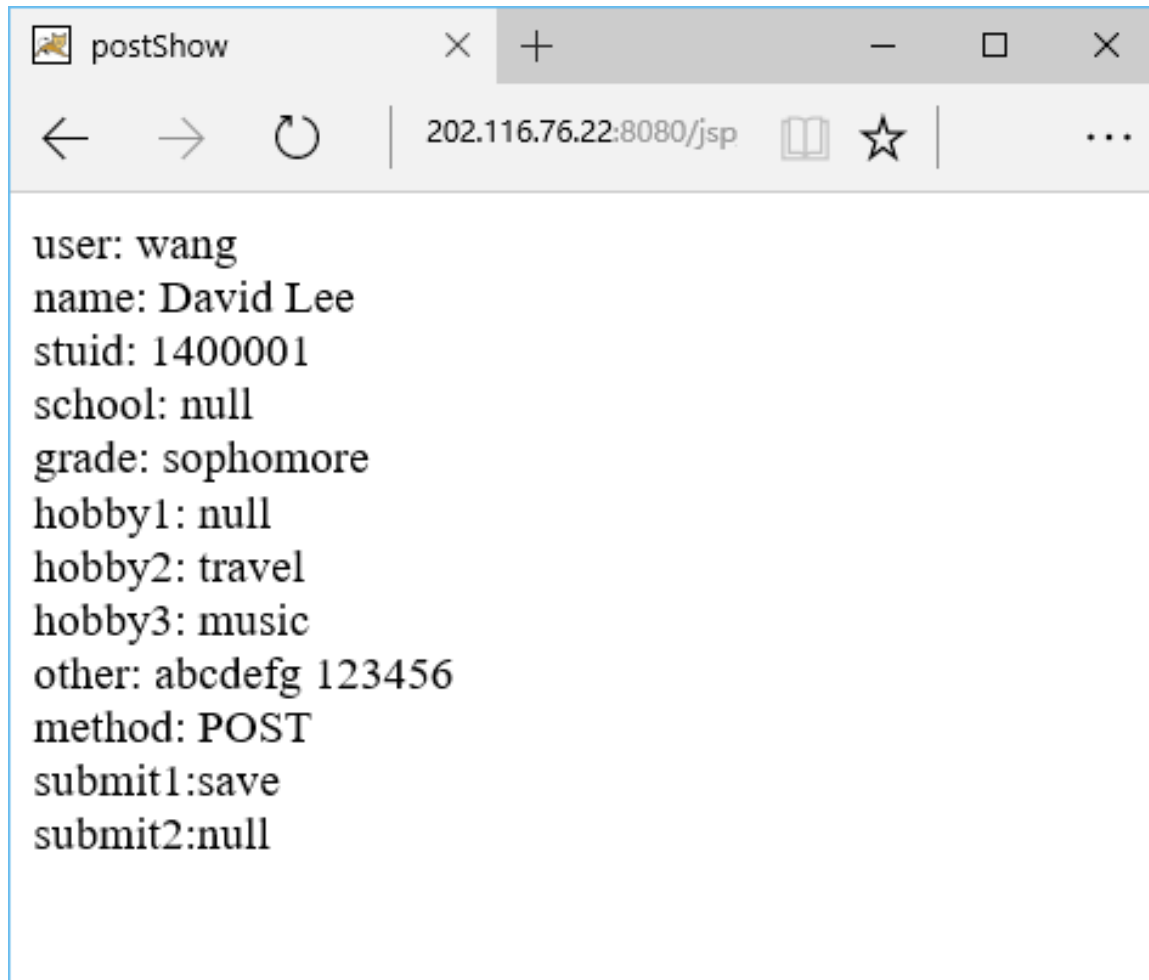
form没有action时，会提交给本网页，如果method为get（默认），则会自动把输入键值对作为url的参数（IE会对汉字进行编码而Chrome则不会）。

postShow.jsp

```jsp
<%@ page language="java" import="java.util.*"
         contentType="text/html; charset=utf-8"%>
<%request.setCharacterEncoding("utf-8");%>
<!DOCTYPE HTML>
<html>
  <head> <title>postShow</title></head>
  <body>user:   <%= request.getParameter("user")    %>
        name:    <%= request.getParameter("name")    %>
        stuid:  <%= request.getParameter("stuid")   %>
        school: <%= request.getParameter("school")  %>
        grade:  <%= request.getParameter("grade")   %>
        hobby1: <%= request.getParameter("hobby1")  %>
        hobby2: <%= request.getParameter("hobby2")  %>
        hobby3: <%= request.getParameter("hobby3")  %>
        other:  <%= request.getParameter("other")   %>
        method: <%= request.getMethod()             %>
        submit1:<%= request.getParameter("submit1") %>
        submit2:<%= request.getParameter("submit2") %>
  </body>
</html>
```

页面编码确定次序：pageEncoding, charset。默认采用ISO-8859-1。ISO-8859-1为单字节编码（欧洲字符集） 参考

　　　　　　＊没有输入值的取值null

```java
String s = request.getParameter("name");
byte[] bs = s.getBytes("iso-8859-1 ");
String s2 = new String(bs,"utf-8");
```

等价

```java
request.setCharacterEncoding("utf-8");
String s = request.getParameter("name");
```

- 提交给自己

postInput.jsp 提交

```
<form action="postShow.jsp?user=wang" method="post">
```

postToSelf.jsp 提交

```
<form action="postToSelf.jsp?user=wang" method="post">
```


输入后 →

← 提交后

点击

参考

- 保持输入

postToSelf.jsp

```jsp
<%@ page language="java" import="java.util.*"
         contentType="text/html; charset=utf-8"%>
<%request.setCharacterEncoding("utf-8");%>
<%
  String submit1 = request.getParameter("submit1");
  String submit2 = request.getParameter("submit2");
  String method = request.getMethod();
  boolean post = method.equalsIgnoreCase("post");  // method:GET 或 POST

  String user = request.getParameter("user");
  if(user==null) user="";
  String name = request.getParameter("name");
  if(name==null) name="";
  String stuid = request.getParameter("stuid");
  if(stuid==null) stuid="";

  String grade = request.getParameter("grade");
  if(grade==null) grade="";
  String grades[]= {"","","",""};
```

```java
if(grade.equals("freshman"))
    grades[0] = "checked";
else if(grade.equals("sophomore"))
    grades[1] = "checked";
else if(grade.equals("junior"))
    grades[2] = "checked";
else if(grade.equals("senior"))
    grades[3] = "checked";

String hobby1 = request.getParameter("ah1");
if(hobby1==null) hobby1="";
if(hobby1.equals("sports")){
    hobby1 = "checked";
}
String hobby2 = request.getParameter("ah2");
if(hobby2==null) hobby2="";
if(hobby2.equals("travel")){
    hobby2 = "checked";
}
String hobby3 = request.getParameter("ah3");
if(hobby3==null) hobby3="";
if(hobby3.equals("music")){
    hobby3 = "checked";
}
String other = request.getParameter("other");
if(other==null) other="";
%>
```

```html
<!DOCTYPE HTML>
<html>
<head>
  <title>postAndKeep</title>
</head>
<body>
<form action="postToSelf.jsp?user=<%=user%>" method="post">
 <input type="hidden" name="stuid" value="<%=stuid%>" />
 name: <input type="text" name="name" value="<%=name%>"/>
 grade:
  freshman<input type="radio" name="grade" value="freshman" <%=grades[0]%>/>
  sophomore<input type="radio" name="grade" value="sophomore" <%=grades[1]%>/>
  junior<input type="radio" name="grade" value="junior" <%=grades[2]%>/>
  senior<input type="radio" name="grade" value="senior" <%=grades[3]%>/>
 hobby:
   sports<input type="checkbox" name="ah1" value="sports" <%=hobby1%>/>
   travel<input type="checkbox" name="ah2" value="travel" <%=hobby2%>/>
   music<input type="checkbox" name="ah3" value="music" <%=hobby3%>/>
   other: <textarea rows="5" cols="30" name="other"><%=other%></textarea>
   <input type="submit"  name="submit1" value="save" />
   <input type="submit"  name="submit2" value="exit" />
 </form>
</body>
</html>
```

select?

- 取到所有提交的名值对　　getParameterNames.jsp

```jsp
<%@ page language="java" import="java.util.*"
        contentType="text/html; charset=utf-8"%>
<%request.setCharacterEncoding("utf-8");%>
<% String saveButton=request.getParameter("submit1"); //判断是否按了保存按钮
    if(saveButton!=null){
      Enumeration<String> enums=request.getParameterNames();
      while(enums.hasMoreElements()){
        String name=(String)enums.nextElement();
        out.print(name+":"+request.getParameter(name)+"<br />");
      }
    }
%>
```

- 取得同名控件提交的值　　getParameterValues.jsp

  如果把前面的checkbox的name都改为hobby，就会出现多个同名的控件。

```jsp
<%@ page language="java" import="java.util.*"
        contentType="text/html; charset=utf-8"%>
<% request.setCharacterEncoding("utf-8");%>
<% String[] values=null;
    if(request.getMethod().equalsIgnoreCase("post")){
        values=request.getParameterValues("hobby");
    }
%>
```

●获得http请求的头部内容

getHeaderNames.jsp

```jsp
<%@ page language="java" import="java.util.*"
         contentType="text/html; charset=utf-8"%>
<%request.setCharacterEncoding("utf-8");%>
<% Enumeration<String> enums=request.getHeaderNames();
    while(enums.hasMoreElements()){
        String name=(String)enums.nextElement();
        out.println("<B>"+name+"</B>: "+request.getHeader(name)+"<br>");
    }
%>
<!DOCTYPE HTML>
<html>
  <head>
    <title>getHeaderNames</title>
  </head>
  <body>
    <form action="getHeaderNames.jsp" method="post">
        <input type="hidden" name="stuid" value="123456" />
        name: <input type="text" name="name" value=""/>
        <input type="submit" name="submit1" value="提交" />
    </form>
  </body>
</html>
```

如何操作数据库

- 浏览学生记录(简单)

```jsp
<%@ page language="java" import="java.util.*,java.sql.*"
        contentType="text/html; charset=utf-8"
%><% request.setCharacterEncoding("utf-8");
    String msg="";
    String table="";
    String connectString = "jdbc:mysql://172.18.187.230:3306/teaching"
                    + "?autoReconnect=true&useUnicode=true"
                    + "&characterEncoding=UTF-8";
        try {
          Class.forName("com.mysql.jdbc.Driver");
          Connection con=DriverManager.getConnection(connectString,
                        "user", "123");
          Statement stmt=con.createStatement();
          ResultSet rs=stmt.executeQuery("select * from stu");
          if(rs.next()) {
              table=rs.getString("name");
          }
          rs.close();  stmt.close();  con.close();
        }
        catch (Exception e){
          msg = e.getMessage();
        }
%>
```

```
<!DOCTYPE HTML>
<html>
<head>
<title>浏览学生名单</title>
</head>
<body>
   <h1>浏览学生名单</h1>
   <%=table%><br><br>
</body>
</html>
```

- 翻页浏览学生记录

```
String sql=String.format("select * from stu limit %d,%d", pgno*pgcnt,pgcnt);
ResultSet rs=stmt.executeQuery(sql);
```
页号   每页行数

```
<%@ page language="java" import="java.util.*,java.sql.*"
        contentType="text/html; charset=utf-8"
%><%  request.setCharacterEncoding("utf-8");
      String msg ="";
      Integer pgno = 0;                        //当前页号
      Integer pgcnt = 4;                       //每页行数
      String param = request.getParameter("pgno");
      if(param != null && !param.isEmpty()){
          pgno = Integer.parseInt(param);
      }
      param = request.getParameter("pgcnt");
      if(param != null && !param.isEmpty()){
          pgcnt = Integer.parseInt(param);
      }
      int pgprev = (pgno>0)?pgno-1:0;
      int pgnext = pgno+1;
      String connectString = "jdbc:mysql://172.18.187.230:3306/teaching"
          + "?autoReconnect=true&useUnicode=true&characterEncoding=UTF-8";
```

```java
String user="user";
String pwd="123";
StringBuilder table = new StringBuilder();
try{
    Class.forName("com.mysql.jdbc.Driver");
    Connection con=DriverManager.getConnection(connectString, user, pwd);
    Statement stmt=con.createStatement();
    String sql=String.format("select * from stu limit %d,%d", pgno*pgcnt,pgcnt);
    ResultSet rs=stmt.executeQuery(sql);
    table.append("<table><tr><th>id</th><th>学号</th><th>姓名</th>"+
                "<th>-</th></tr>");
    while(rs.next()) {
        table.append(String.format(
                "<tr><td>%s</td><td>%s</td><td>%s</td><td>%s %s</td></tr>",
                rs.getString("id"),rs.getString("num"),rs.getString("name"),
                "<a href='updateStu.jsp?pid="+rs.getString("id")+"'>修改</a>",
                "<a href='deleteStu.jsp?pid="+rs.getString("id")+"'>删除</a>"
            )
        );
    }
    table.append("</table>");
    rs.close();  stmt.close();  con.close();
}
catch (Exception e){
    msg = e.getMessage();
}
%>
```

```html
<!DOCTYPE HTML>
<html>
<head>
<title>浏览学生名单</title>
<style>
...   见下页
</style>
</head>
<body>
  <div class="container">
     <h1>浏览学生名单</h1>
     <%=table%>
     <div style="float:left">
       <a href="addStu.jsp">新增</a>
     </div>
     <div style="float:right">
       <a href="browseStu.jsp?pgno=<%=pgprev%>&pgcnt=<%=pgcnt%>">上一页</a>
       <a href="browseStu.jsp?pgno=<%=pgnext%>&pgcnt=<%=pgcnt%>">下一页</a>
     </div>
     <br><br>
     <%=msg%><br><br>
  </div>
</body>
</html>
```

```
<style>
  table{
        border-collapse: collapse;
        border: none;
        width: 500px;
  }
  td,th{
        border: solid grey 1px;
        margin: 0 0 0 0;
        padding: 5px 5px 5px 5px
  }
  a:link,a:visited {
    color:blue
  }

  .container{
    margin:0 auto;
    width:500px;
    text-align:center;
  }
}
</style>
```

- 新增操作

```jsp
<%@ page language="java" import="java.util.*,java.sql.*"
         contentType="text/html; charset=utf-8"%>
<% request.setCharacterEncoding("utf-8");
String msg = "";
String connectString = "jdbc:mysql://172.18.187.11:3306/teaching"
        + "?autoReconnect=true&useUnicode=true&characterEncoding=UTF-8";
String user="user"; String pwd="123";
String num = request.getParameter("num");
String name = request.getParameter("name");
if(request.getMethod().equalsIgnoreCase("post")){
    Class.forName("com.mysql.jdbc.Driver");
    Connection con = DriverManager.getConnection(connectString,user, pwd);
    Statement stmt = con.createStatement();
    try{
        String fmt="insert into stu(num,name) values('%s', '%s')";
        String sql = String.format(fmt,num,name);
        int cnt = stmt.executeUpdate(sql);
        if(cnt>0)msg = "保存成功!";
        stmt.close();
        con.close();
    }catch (Exception e){
        msg = e.getMessage();
    }
}
%>
```

```html
<!DOCTYPE HTML>
<html>
<head>
  <title>新增学生记录</title>
  <style>
      a:link,a:visited {color:blue;}
      .container{
          margin:0 auto;
          width:500px;
          text-align:center;
      }
  </style>
</head>
<body>
  <div class="container">
        <h1>新增学生记录</h1>
        <form action="addStu.jsp" method="post" name="f">
                学号:<input id="num" name="num" type="text" >
                姓名:<input id="name" type="text" name="name" >
                <input type="submit" name="sub" value="保存">
        </form>
        <%=msg%>
        <a href='browseStu.jsp'>返回</a>
  </div>
</body>
</html>
```

附录

YM Zhang sysu.edu.cn

附录1、hello_jsp.java程序

黑色的粗体字为JSP的基本内置对象

```
package org.apache.jsp;
import javax.servlet.*;
import javax.servlet.http.*;
import javax.servlet.jsp.*;
public final class hello_jsp extends org.apache.jasper.runtime.HttpJspBase
    implements org.apache.jasper.runtime.JspSourceDependent {
    private static final javax.servlet.jsp.JspFactory _jspxFactory =
        javax.servlet.jsp.JspFactory.getDefaultFactory();
... //定义了对象：_jspx_dependants; _el_expressionfactory; _jsp_instancemanager;
...//定义了方法： getDependants() {...},_jspInit(){...}, _jspDestroy() {...}
...
public void _jspService(
        final javax.servlet.http.HttpServletRequest request,
        final javax.servlet.http.HttpServletResponse response)
    throws java.io.IOException, javax.servlet.ServletException {
    final javax.servlet.jsp.PageContext pageContext;
    javax.servlet.http.HttpSession session = null;
    final javax.servlet.ServletContext application;
    final javax.servlet.ServletConfig config;
    javax.servlet.jsp.JspWriter out = null;
    final java.lang.Object page = this;
    javax.servlet.jsp.JspWriter _jspx_out = null;
    javax.servlet.jsp.PageContext _jspx_page_context = null;
```

```java
try {
    response.setContentType("text/html;charset=gb2312");
    pageContext = _jspxFactory.getPageContext(this, request, response,
                        null, true, 8192, true);
    _jspx_page_context = pageContext;
    application = pageContext.getServletContext();
    config = pageContext.getServletConfig();
    session = pageContext.getSession();
    out = pageContext.getOut();
    _jspx_out = out;
    out.write("<html>\r\n");
    out.write("<head>\r\n");
    out.write("<title>Hello world</title>\r\n");
    out.write("</head>\r\n");
    out.write("<body>\r\n");
    out.print("Hello world!");
    out.write("</body>\r\n");
    out.write("</html>\r\n");
    out.write("\r\n");
```

```java
      out.write("\r\n");
      out.write("\r\n");
    } catch (java.lang.Throwable t) {
      if (!(t instanceof javax.servlet.jsp.SkipPageException)){
        out = _jspx_out;
        if (out != null && out.getBufferSize() != 0)
          try { out.clearBuffer(); } catch (java.io.IOException e) {}
        if (_jspx_page_context != null)
              _jspx_page_context.handlePageException(t);
      }
    } finally {
      _jspxFactory.releasePageContext(_jspx_page_context);
    }
  }  // _jspService
} // class
```

附录2、JSP声明的源码转换

getSumFuncClass.jsp

```jsp
<%@ page contentType="text/html;charset=gb2312"%>
<%! int cnt = 0;
    int getSum(int n){
      int sum=0;
      for(int i=1;i<=n;i++){
        sum=sum+i;
      }
      return sum;
    }
    class Sum{
      int getSum(int n){
        int sum=0;
        for(int i=1;i<=n;i++){
          sum=sum+i;
        }
        return sum;
      }
    }
 %>
<html>
<head>
<title>getSumFunc</title>
</head>
<body>
  <h1>getSumFunc</h1>
  <p>sum1:<%=getSum(100)%></p>
  <p>sum2:<% Sum sum=new Sum();%><%=sum.getSum(100)%></p>
  count:<%=++cnt%>
</body>
</html>
```

getSumFuncClass_jsp.jsp

```java
package org.apache.jsp;
import javax.servlet.*;
import javax.servlet.http.*;
import javax.servlet.jsp.*;
public final class getSumfuncClass_jsp extends org.apache.jasper.runtime.HttpJspBase
    implements org.apache.jasper.runtime.JspSourceDependent,
                org.apache.jasper.runtime.JspSourceImports {
    int cnt = 0;
    int getSum(int n){
      int sum=0;
      for(int i=1;i<=n;i++){
         sum=sum+i;
      }
      return sum;
    }
    class Sum{
      int getSum(int n){
        int sum=0;
        for(int i=1;i<=n;i++){
          sum=sum+i;
        }
        return sum;
      }
    }

  private static final javax.servlet.jsp.JspFactory _jspxFactory =
          javax.servlet.jsp.JspFactory.getDefaultFactory();
  private static java.util.Map<java.lang.String,java.lang.Long> _jspx_dependants;
  private static final java.util.Set<java.lang.String> _jspx_imports_packages;
  private static final java.util.Set<java.lang.String> _jspx_imports_classes;
  static {
    _jspx_imports_packages = new java.util.HashSet<>();
    _jspx_imports_packages.add("javax.servlet");
    _jspx_imports_packages.add("javax.servlet.http");
    _jspx_imports_packages.add("javax.servlet.jsp");
    _jspx_imports_classes = null;
  }
```

```java
  private volatile javax.el.ExpressionFactory _el_expressionfactory;
    private volatile org.apache.tomcat.InstanceManager _jsp_instancemanager;
    public java.util.Map<java.lang.String,java.lang.Long> getDependants() {
      return _jspx_dependants;
    }
    public java.util.Set<java.lang.String> getPackageImports() {
      return _jspx_imports_packages;
    }
    public java.util.Set<java.lang.String> getClassImports() {
      return _jspx_imports_classes;
    }
    public javax.el.ExpressionFactory _jsp_getExpressionFactory() {
      if (_el_expressionfactory == null) {
        synchronized (this) {
          if (_el_expressionfactory == null) {
            _el_expressionfactory =
                _jspxFactory.getJspApplicationContext(getServletConfig().getServletContext()).getExpressionFactory();
          }
        }
      }
      return _el_expressionfactory;
    }
    public org.apache.tomcat.InstanceManager _jsp_getInstanceManager() {
      if (_jsp_instancemanager == null) {
        synchronized (this) {
          if (_jsp_instancemanager == null) {
            _jsp_instancemanager = org.apache.jasper.runtime.InstanceManagerFactory.getInstanceManager(getServletConfig());
          }
        }
      }
      return _jsp_instancemanager;
    }
    public void _jspInit() {
    }
    public void _jspDestroy() {
    }
    public void _jspService(final javax.servlet.http.HttpServletRequest request,
                            final javax.servlet.http.HttpServletResponse response)
        throws java.io.IOException, javax.servlet.ServletException {
      final java.lang.String _jspx_method = request.getMethod();
      if (!"GET".equals(_jspx_method) && !"POST".equals(_jspx_method) && !"HEAD".equals(_jspx_method)
          && !javax.servlet.DispatcherType.ERROR.equals(request.getDispatcherType())) {
        response.sendError(HttpServletResponse.SC_METHOD_NOT_ALLOWED, "JSPs only permit GET POST or HEAD");
        return;
      }
```

```java
final javax.servlet.jsp.PageContext pageContext;
    javax.servlet.http.HttpSession session = null;
    final javax.servlet.ServletContext application;
    final javax.servlet.ServletConfig config;
    javax.servlet.jsp.JspWriter out = null;
    final java.lang.Object page = this;
    javax.servlet.jsp.JspWriter _jspx_out = null;
    javax.servlet.jsp.PageContext _jspx_page_context = null;
    try {
      response.setContentType("text/html;charset=gb2312");
      pageContext = _jspxFactory.getPageContext(this, request, response,
                                  null, true, 8192, true);
      _jspx_page_context = pageContext;
      application = pageContext.getServletContext();
      config = pageContext.getServletConfig();
      session = pageContext.getSession();
      out = pageContext.getOut();
      _jspx_out = out;
      out.write('\r');
      out.write('\n');
      out.write("\r\n");
      out.write("<html>\r\n");
      out.write("<head>\r\n");
      out.write("<title>getSumFunc</title>\r\n");
      out.write("</head>\r\n");
      out.write("<body>\r\n");
      out.write("  <h1>getSumFunc</h1>\r\n");
      out.write("  <p>sum1:");
      out.print(getSum(100));
      out.write("</p>\r\n");
      out.write("  <p>sum2:");
      Sum sum=new Sum();
      out.print(sum.getSum(100));
      out.write("</p>\r\n");
      out.write("  count:");
      out.print(++cnt);
      out.write("\r\n");
      out.write("</body>\r\n");
      out.write("</html>\r\n");
```

```java
      } catch (java.lang.Throwable t) {
        if (!(t instanceof javax.servlet.jsp.SkipPageException)){
          out = _jspx_out;
          if (out != null && out.getBufferSize() != 0)
            try {
              if (response.isCommitted()) {
                out.flush();
              } else {
                out.clearBuffer();
              }
            } catch (java.io.IOException e) {}
          if (_jspx_page_context != null) _jspx_page_context.handlePageException(t);
          else throw new ServletException(t);
        }
      } finally {
        _jspxFactory.releasePageContext(_jspx_page_context);
      }
    }
}
```

附录3、JSP页面构成元素

- JSP页面主要是在HTML页面中加入JSP的内容。JSP内容均在<%和%>之间定义。

- 生成响应页面时HTML模版直接输出到客户端，JSP内容要执行语句或给出编译指示，只有输出的内容会被发送到客户端。

- JSP页面由五种元素组成：
  (1) HTML模版
  (2) 指令标签
  (3) 动作标签
  (4) 脚本程序：声明片段、脚本片段和注释
  (5) 表达式

附录4、JSP页面元素

<%@ 指令%>	对后面的JSP页面给出编译指示，主要指令有：**page**，**include**，taglib
<% 动作 %>	用来实现特殊的功能，主要标签：<jsp:include> <jsp:forward> <jsp:useBean> <jsp:setProperty> <jsp:getProperty>
<%! 声明%>	声明全局JSP变量和方法。它们被所有线程所共享。
<% 脚本片段 %>	有效的java程序段，其中定义的变量为局部变量。
<%-- 注释--%>	JSP注释
<%= 表达式%>	计算JSP表达式的值并输出到页面。

附录5、JSP指令标签(Directive)

- 概述

    JSP指令标签指示JSP引擎对JSP页面需要做什么。

- 主要指令

    （1）Page指令
    （2）include指令
    （3）taglib指令(后面再讲)

- page指令

  功能

  用来定义整个JSP页面的属性。一个JSP页面可以有多条page指令。除了import属性，其它属性只能定义一次。

  语法(红色为默认值)

<%@ page [language="java"]   定义所用语言(默认为java)

      [extends="package.class"]   定义JSP文件的继承类

      [import="{package.class, ..."]   指明想要引入的java类包

      [session="true|false"]   是否允许使用session对象

      [buffer="none|8kb|xkb"]   输出缓冲的大小

      [autoFlush="true|false"]   当缓冲满时是否自动输出

      [isThreadSafe="true|false"]   采用多线程(默认)还是单线程

[info="text"]   定义JSP页面的文本信息

[contentType="text/html ; charset=ISO-8859-1"]
                                  定义传送给浏览器的内容（或文件）
                                  类型和字符编码。

[errorPage="relativeURL"]   处理意外的页面

[isErrorPage="true|false"]   指定当前页面是否可以处理来自另
                                  一个页面的错误，缺省为"false"。

[pageCoding="ISO-8859-1"]   定义JSP源程序使用的字符编码

%>

（1）**JSP默认已引入以下包**：java.lang.*，javax.servlet.*，
   javax.servlet.jsp.*，javax.servlet.http.*。其它类包必须用指令引入：
     <%@ page import=*"java.util.*" %>
     <%@ page import=*"java.io.*,java.net.*,java.sql.*" %>

（2）**autoFlush**指明每当缓冲区满时是否自动把缓冲区的内容输出给客户端。如果autoFlush为false，当buffer不足时，会抛出错误。当buffer设置为none, autoFlush必须为true。

```
<%@ page buffer="24kb" autoFlush=false %>
```

（3）设置所支持的语言，目前**JSP支持的语言**只有java：

```
<%@ page language="java"  %>
```

（4）用`<%@ page contentType="text/html; charset=utf-8"%>`

设置http响应的内容（或文件）类型和编码。

其它文件类型：*text/plain(文本)，image/gif（gif图像），image/jpeg（jpeg图像），application/xshockwave（flash文件），application /msword（msword文件），…。*

支持的显示字符集(charset): *ISO-8859-1，GBK，GB2312，UTF-8, …。*

也可以采用`<% response.setContentType("text/html; charset=utf-8"); %>`进行设置。

这个指令会影响HTTP响应：

HTTP/1.1 200 OK\r\n

…

Content-Type: text/html; charset=utf-8\r\n

\r\n

正文

（5）用<%@ page errorPage=*"errorPage.jsp"* %>指出出错处理程序为errorPage.jsp。
在这个程序中用命令 **<%@page isErrorPage="true" %>**说明是错误处理程序。用 **<%=exception.toString()%>**显示出错信息。

（6）用<%@page info=*"增加学生信息"* %>定义页面信息
用**<%=getServletInfo()%>**可以显示出该信息。

（7）用<%@page pageEncoding=*"GBK"* %>定义源文件编码，它和contentType如果只有一个定义了编码，则都使用该编码，如果两个都没定义，则默认使用*ISO-8859-1*。

JSP编译：把pageEncoding的JSP源代码转换为unicode的servlet(.java)程序，然后编译为字节码程序。http响应的内容要转换为contentType指定的编码。提交的内容用request.setCharacterCode()指定的编码进行解码。

附录6、JavaBean的scope

getServerTime.jsp

```
<%@ page language="java" import="java.util.*" pageEncoding="utf-8"%>
<jsp:useBean id="getServerTime" class="Java.util.Date" scope="page" />
<%= getServerTime.toString() %>
```
对象名

当scope=page/request时，每次访问该网页(刷新页面)都会创建新的JavaBean对象。
当scope=session时，只有获得新的sessionID时，才会创建新的JavaBean对象。
当scope=application时，只有重启Web服务器才会创建新的JavaBean对象。在不同主机和浏览器访问时都会取到同一个的时间。

page表示将JavaBean实例对象存储在PageContext对象中，作用范围是当前JSP页面有效。
request表示将JavaBean实例对象存储在ServletRequest对象中，即可以被属于同一个请求的所有Servlet和JSP页面访问。
session表示将JavaBean实例对象存储在HttpSession对象中，存储在HttpSession对象中的JavaBean对象可以被属于同一个会话的所有Servlet和JSP页面访问。要求当前JSP页面支持Session，即page指令的session属性设置为true（默认）。
application表示将JavaBean实例对象存储在ServletContext对象中，存储在ServletContext对象中的JavaBean对象可以被同一个Web应用程序中的所有Servlet和JSP页面访问。也就是所有用户都可以使用。

附录7、配置问题

* 如果希望jsp文件被修改后自动编译运行，就需要在子目录META-INF下新建context.xml文件，内容为：<Context reloadable="true"/>

* 所有应用程序都要使用的jar包应该放在子目录tomcat\lib中。jsp不需要配置classpath。

* 下面为增加虚拟主机和虚拟目录的方法。webapps下的子目录都是虚拟目录。要在其它地方定义虚拟目录，需要修改配置文件tamcat/conf/server.xml。为了定义lab和test两个虚拟目录，下面在server.xml增加了两个配置：

```
<Host name="localhost"  appBase="webapps" unpackWARs="true" autoDeploy="true">
  <Context path="lab" docBase="D:\Lab" reloadable="true"></Context>
  <Context path="test" docBase="D:\test" reloadable="true"></Context>
</Host>
<Host name= "www.group.com"  appBase="d:\group" unpackWARs="true"
      autoDeploy="true">
  <Context path= "tech" docBase="D:\Tech" reloadable="true"></Context>
  <Context path="teach" docBase="D:\Teach" reloadable="true"></Context>
</Host>
```