

# 课程回顾

## 1 session基本使用

- 1 servlet中, 通过getSession()获取对象
- 2 setAttribute()完成数据存储
- 3 getAttribute()获取数据

## 2 session底层实现

- 1 session生命周期:
- 2 1.session数据保存服务器上, 服务器保存session的默认时常: 30分钟
- 3 2.session底层基于cookie在浏览器端存储一个JSESSIONID, cookie默认生命周期浏览器关闭, cookie就会丢失, session在浏览器重启之后, 数据也获取不到
- 4 3.延迟session的JSESSIONID存储时常, 通过修改Cookie的setMaxAge()

## 3 验证码

12306买票系统使用验证码

- 1 3-1 生成验证码
- 2 ``
- 3 3-2 看不清, 换一张
- 4 3-3 服务器端 session对象获取正确的验证码

## 4 URL重写 ===== 了解

- 1 解决问题: 浏览器禁用cookie
- 2 URL重写技术实现原理: 保证每次跟服务器交互过程中, url地址带上JSESSIONID
- 3 格式:
- 4 `url;JSESSIONID=session.getId()`
- 5 JAVAX提供实现url重写的方法:
- 6 `response.encodeURL()`
- 7 `response.encodeRedirectURL()`

## 5 文件下载 \*\*

- 1 步骤:
- 2 1.提供文件下载的资源, 资源通常保存位置: WEB-INF下面
- 3 2.jsp页面如何提供下载链接
- 4 `<a href=".....?下载的文件名称"></a>`
- 5 3.Servlet如何实现下载
- 6 `response.setHeader("content-disposition", "attachment;fileName=定义文件名称");`
- 7 IO流输出

# 课程目标

## 1 文件上传 ===== 理解

## 2 四大域对象 ===== 理解

## 3 JSP使用 ===== 理解

# 课程实施

## 1 文件上传

### 1-1 文件上传和下载

日常项目常用的功能

提问：淘宝、京东文件上传的功能？

举例：头像、评价+加图、商家上架图片上传

OA系统：秘书起草文件，上传给领导审核

### 1-2 文件上传步骤

- 1 1. 必须使用表单，而不能是超链接；<input type=file>
- 2 a) 超链接走get请求，get提交数据在浏览器地址栏上面
- 3 2. 表单的method必须是POST，而不能是GET；
- 4 a) GET: get传递数据时候，用查询字符串，在地址栏明码传递，地址栏传递的数据长度有限制，文件过大地址栏数据丢失问题。
- 5 3. 表单的enctype必须是multipart/form-data；
- 6 a) Enctype: application/x-www-mulipat...
- 7 i. 客户端向服务器发送表单数据name=value
- 8 b) multipart/form-data
- 9 c) 常见普通form提交post，请求体 username:值
- 10 4. 在表单中添加file表单字段，即<input type="file".../>

### 1 jsp页面的设计

```
1 <%@ page contentType="text/html;charset=UTF-8" language="java" %>
2 <html>
3 <head>
4   <title>文件上传</title>
5 </head>
6 <body>
7 <!--
8 application/x-www-form-urlencoded: 表单默认数据提交格式
9 --%>
10 <form action="${pageContext.servletContext.contextPath}/UploadServlet"
    method="post"
11   enctype="multipart/form-data">
12   用户名: <input type="text" name="username"/><br/>
13   头像图片: <input type="file" name="face"/><br/>
14   <input type="submit" value="注册"/>
15 </form>
16 </body>
17 </html>
```

## 2 添加文件上传的依赖

```
lib
> commons-fileupload-1.3.3.jar
> commons-io-2.6.jar
```

## 3 UploadServlet实现文件上传

```
1 package cn.kgc.controller;
2 import org.apache.commons.fileupload.FileItem;
3 import org.apache.commons.fileupload.disk.DiskFileItemFactory;
4 import org.apache.commons.fileupload.servlet.ServletFileUpload;
5
6 import javax.servlet.ServletException;
7 import javax.servlet.annotation.WebServlet;
8 import javax.servlet.http.HttpServlet;
9 import javax.servlet.http.HttpServletRequest;
10 import javax.servlet.http.HttpServletResponse;
11 import java.io.File;
12 import java.io.IOException;
13 import java.util.List;
14
15 @WebServlet("/uploadServlet")
16 public class UploadServlet extends HttpServlet {
17     @Override
18     protected void doGet(HttpServletRequest request, HttpServletResponse
19 response) throws ServletException, IOException {
20         //取
21         //request.getParameter()不再适用
22         //ServletOutputStream out = response.getOutputStream();//输出响应体
23         //ServletInputStream in = request.getInputStream();//请求体
24         //读取上传文件的内容?? 肯定可以读
25         //依赖fileupload.jar
26         //1.创建工厂对象
27         DiskFileItemFactory factory=new DiskFileItemFactory();
28         //2.创建Servlet端使用请求解析器
29         ServletFileUpload upload=new ServletFileUpload(factory);
30         //3.通过解析器将request转换为List<FileItem>
31         //3-1 FileItem:form表提交的一个一个表单项的数据
32         try {
33             List<FileItem> fileItems = upload.parseRequest(request);
34             for(FileItem item :fileItems) {
35                 //获取用户提交的用户名 普通表单项 type=text password
36                 if(item.isFormField()){//true:普通的表单项
37                     //item.getFieldName():<input type="" name=""/>获取表单项的
38                     name属性值
39                     //getString():当前这个表单项提交的数据
40                     System.out.println(item.getFieldName()+"提交的数据
41 是: "+item.getString());
42                 }else {
43                     //获取用户提交图片 文件域 type=file
44                     //实现文件上传
45                     /*
46                     * 实现原理: 将浏览器提交的文件, 使用io复制到服务器端
47                     */
48                     //InputStream in = item.getInputStream();
49                     //文件复制后放在服务器的什么地方呢?
```

```

47         //获取服务器指定文件路径，常用代码
48         //getName():获取文件域上次文件实际的名称
49         String savePath=getServletContext().getRealPath("/WEB-INF/upload/"+item.getName()); //获取指定目录在服务器上的绝对路径
50         File os=new File(savePath);
51         //实现实际文件保存
52         item.write(os);
53     }
54 }
55 } catch (Exception e) {
56     e.printStackTrace();
57 }
58
59 }
60
61 @Override
62 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
63     doGet(request, response);
64 }
65 }
66

```

## 1-3 文件上传的实现细节

### 中文乱码问题

```

1 //解决上传文件中文乱码的问题
2 response.setCharacterEncoding("utf-8");
3 //如果以上代码不能处理中文乱码，可以在getString()时再处理
4 fileItem.getString("utf-8")

```

### 文件格式问题

```

1 /*上传图片文件：
2     文件合法格式: jpg jpeg png gif bmp ico
3 */
4 String fileFullName = item.getName(); //提交上传文件的全名
5 //获取文件后缀名
6 String lastName = fileFullName.substring(fileFullName.lastIndexOf("."));
7 //限制文件格式: jpg jpeg png....
8 String[] rightLastName={"jpg","jpeg","gif","bmp"};
9 //文件是否合法
10 boolean isOk=false; //默认不合法
11 for(String name:rightLastName){
12     if(fileFullName.toLowerCase().endsWith(name)){
13         isOk=true;
14     }
15 }
16 if(!isOk){
17     //不合法
18     request.setAttribute("msg","只能上传后缀名是: "+
19 Arrays.toString(rightLastName));
20     request.getRequestDispatcher("/upload.jsp").forward(request, response);
21     return;
22 }

```

## 上传文件大小问题

```
1  /*
2  单个图片限制大小：4m以内 4m=4*1024*1024==>字节
3  file.length()://单位字节*/
4  //1.创建工厂对象
5  DiskFileItemFactory factory=new DiskFileItemFactory();
6
7  //2.创建Servlet端使用请求解析器
8  ServletFileUpload upload=new ServletFileUpload(factory);
9
10 //设置单个文件上传的大小
11 upload.setFileSizeMax(4*1024*1024);//如果单个文件超过4m，抛出500的异常
12
13 //3.通过解析器将request转换为List<FileItem>
14 List<FileItem> fileItems = upload.parseRequest(request);
```

HTTP Status 500 - org.apache.commons.fileupload.FileUploadBase\$FileSizeLimitExceededException: The field face exceeds its maximum permitted size of 4194304 bytes.

type Exception report

message org.apache.commons.fileupload.FileUploadBase\$FileSizeLimitExceededException: The field face exceeds its maximum permitted size of 4194304 bytes.

description The server encountered an internal error that prevented it from fulfilling this request.

exception

java.lang.RuntimeException: org.apache.commons.fileupload.FileUploadBase\$FileSizeLimitExceededException: The field face exceeds its maximum permitted size of 4194304 bytes.  
cn.kgc.controller.UploadServlet.doGet(UploadServlet.java:91)  
cn.kgc.controller.UploadServlet.doPost(UploadServlet.java:98)  
javax.servlet.http.HttpServlet.service(HttpServlet.java:647)  
javax.servlet.http.HttpServlet.service(HttpServlet.java:728)

root cause

org.apache.commons.fileupload.FileUploadBase\$FileSizeLimitExceededException: The field face exceeds its maximum permitted size of 4194304 bytes.  
org.apache.commons.fileupload.FileUploadBase\$FileItemIteratorImpl\$FileItemStreamImpl1.raiseError(FileUploadBase.java:789)  
org.apache.commons.fileupload.util.LimitedInputStream.checkLimit(LimitedInputStream.java:78)  
org.apache.commons.fileupload.util.LimitedInputStream.read(LimitedInputStream.java:137)  
java.io.FilterInputStream.read(FilterInputStream.java:107)  
org.apache.commons.fileupload.util.Streams.copy(Streams.java:100)  
org.apache.commons.fileupload.util.Streams.copy(Streams.java:70)  
org.apache.commons.fileupload.FileUploadBase.parseRequest(FileUploadBase.java:347)  
org.apache.commons.fileupload.servlet.ServletFileUpload.parseRequest(ServletFileUpload.java:115)  
cn.kgc.controller.UploadServlet.doGet(UploadServlet.java:40)  
cn.kgc.controller.UploadServlet.doPost(UploadServlet.java:98)  
javax.servlet.http.HttpServlet.service(HttpServlet.java:647)  
javax.servlet.http.HttpServlet.service(HttpServlet.java:728)

上传文件超出限制的大小4m后的异常

## 文件重名的问题

```
1  //文件名称以上传的时间命名: now().getTime().jpg
2  String fileFullName = item.getName();//提交上传文件的全名
3  //获取文件后缀名
4  String lastName = fileFullName.substring(fileFullName.lastIndexOf("."));
5  //自动生成一个服务器端保存的文件名称
6  String fileFirstName=new Date().getTime()+"";//123432435.jpg
7  //生成新的保存文件的名称
8  String newFileFullName=fileFirstName+lastName;
9  //获取指定目录在服务器上的绝对路径
10 String savePath=getServletContext().getRealPath("/WEB-INF/upload/"+newFileFullName);
11
12 System.out.println(savePath);
13 File os=new File(savePath);
14
15 //实现实际文件保存
16 item.write(os);
```

## 1-4 文件上传解决细节问题的完整代码

```
1 package cn.kgc.controller;
2
3 import org.apache.commons.fileupload.FileItem;
4 import org.apache.commons.fileupload.disk.DiskFileItemFactory;
5 import org.apache.commons.fileupload.servlet.ServletFileUpload;
6
7 import javax.servlet.ServletException;
8 import javax.servlet.annotation.WebServlet;
9 import javax.servlet.http.HttpServlet;
10 import javax.servlet.http.HttpServletRequest;
11 import javax.servlet.http.HttpServletResponse;
12 import java.io.File;
13 import java.io.IOException;
14 import java.util.Arrays;
15 import java.util.Date;
16 import java.util.List;
17
18 @WebServlet("/UploadServlet")
19 public class UploadServlet extends HttpServlet {
20     @Override
21     protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
22         //解决上传文件中文乱码的问题
23         response.setCharacterEncoding("utf-8");
24         //取
25         //request.getParameter()不再适用
26         //ServletOutputStream out = response.getOutputStream();//输出响应体
27         //ServletInputStream in = request.getInputStream();//请求体
28         //读取上传文件的内容?? 肯定可以读
29         //依赖fileupload.jar
30         //1.创建工厂对象
31         DiskFileItemFactory factory=new DiskFileItemFactory();
32
33         //2.创建Servlet端使用请求解析器
34         ServletFileUpload upload=new ServletFileUpload(factory);
35         //设置单个文件上传的大小
36         upload.setFileSizeMax(4*1024*1024);//如果单个文件超过4m, 抛出500的异常
37         //3.通过解析器将request转换为List<FileItem>
38         //3-1 FileItem:form表提交的一个一个表单项的数据
39         try {
40             List<FileItem> fileItems = upload.parseRequest(request);
41             for(FileItem item :fileItems) {
42                 //获取用户提交的用户名 普通表单项 type=text password
43                 if(item.isFormField()){//true:普通的表单项
44                     //item.getFieldName():<input type="" name=""/>获取表单的
name属性值
45                     //getString(编码格式):解决中文乱码当前这个表单项提交的数据
46                     System.out.println(item.getFieldName()+"提交的数据
是: "+item.getString("utf-8"));
47                 }else {
48                     //获取用户提交图片 文件域 type=file
49                     //实现文件上传
50                     /*
51                     * 实现原理: 将浏览器提交的文件, 使用io复制到服务器端
52                     */
```

```

53         //InputStream in = item.getInputStream();
54         //文件复制后放在服务器的什么地方呢?
55         //获取服务器指定文件路径, 常用代码
56         //getName(): 获取文件域上次文件实际的名称
57         //getRealPath("必须以/开始, 且不包含web层")
58         String fileFullName = item.getName();//提交上传文件的全名
59         //获取文件后缀名
60         String lastName =
fileFullName.substring(fileFullName.lastIndexOf("."));
61         //限制文件格式: jpg jpeg png....
62         String[] rightLastName={"jpg","jpeg","gif","bmp"};
63         //文件是否合法
64         boolean isOk=false;//默认不合法
65         for(String name:rightLastName){
66             if(fileFullName.toLowerCase().endsWith(name)){
67                 isOk=true;
68             }
69         }
70         if(!isOk){
71             //不合法
72             request.setAttribute("msg","只能上传后缀名是: "+
Arrays.toString(rightLastName));
73
74             request.getRequestDispatcher("/upload.jsp").forward(request, response);
75             return;
76         }
77         //自动生成一个服务器端保存的文件名称
78         String fileFirstName=new
Date().getTime()+"";//123432435.jpg
79         String newFileFullName=fileFirstName+lastName;
80         String savePath=getServletContext().getRealPath("/WEB-
INF/upload/"+newFileFullName);//获取指定目录在服务器上的绝对路径
81
82         System.out.println(savePath);
83         File os=new File(savePath);
84         //实现实际文件保存
85         item.write(os);
86     }
87     } catch (Exception e) {
88         throw new RuntimeException(e);
89     }
90 }
91
92 @Override
93 protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
94     doGet(request, response);
95 }
96 }
97 }

```

## 2 JSP周边

tomcat服务器上保存jsp解析的目录是work目录, 但是因为idea发布项目的方式是虚拟路径的形式, 所以在idea中发布的项目并没有实际放在tomcat的webapps下面:

backup	2018/8/31 15:10	文件夹	
bin	2020/11/3 9:33	文件夹	
conf	2019/3/15 17:08	文件夹	
lib	2013/7/2 8:59	文件夹	
logs	2022/5/16 11:11	文件夹	
temp	2022/5/25 16:52	文件夹	
webapps	2022/5/16 11:11	文件夹	
work	work保存服务器解析jsp的工作过程	文件夹	
LICENSE	2013/7/2 8:59	文件	57 KB
NOTICE	2013/7/2 8:59	文件	2 KB
RELEASE-NOTES	2013/7/2 8:59	文件	9 KB
RUNNING.txt	2013/7/2 8:59	文本文档	17 KB

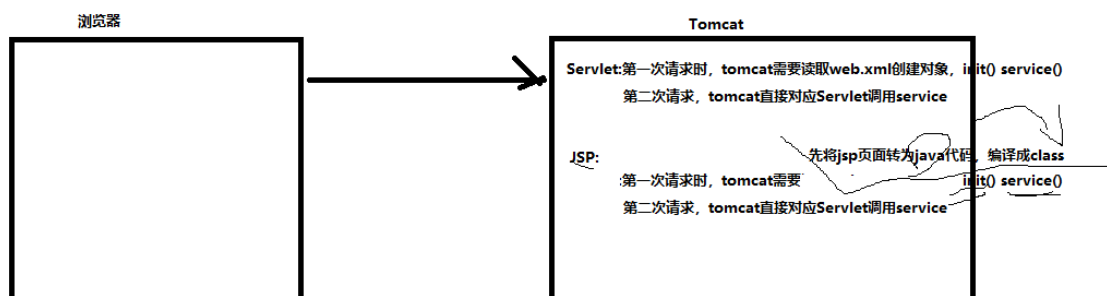
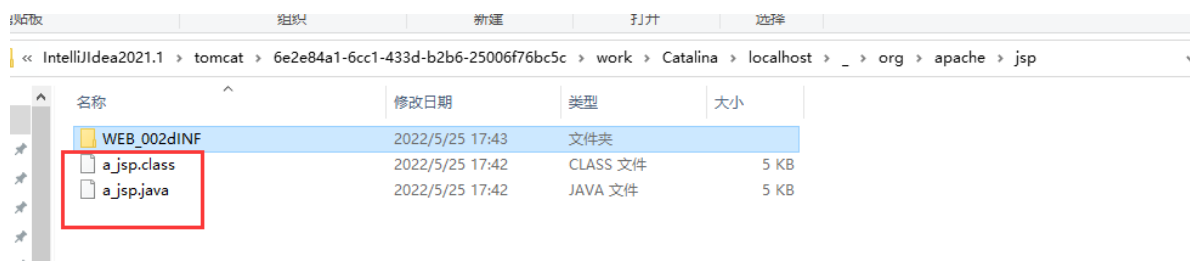
idea发布项目的实际路径是在tomcat日志信息中“Using CATALINA\_BASE”对应的路径上：

```

deployment...
Using CATALINA_BASE: "C:\Users\Administrator\AppData\Local\JetBrains\IntelliJ\Idea2021.1\tomcat\6e2e84a1-6cc1-433d-b2b6-25006f76bc5c"
Using CATALINA_HOME: "D:\apache-tomcat-7.0.42"
Using CATALINA_TMPDIR: "D:\apache-tomcat-7.0.42\temp"
Using JRE_HOME: "C:\Program Files\Java\jdk1.8.0_111"
Using CLASSPATH: "D:\apache-tomcat-7.0.42\bin\bootstrap.jar;D:\apache-tomcat-7.0.42\bin\tomcat-jar.jar"
五月 25, 2022 5:08:37 下午 org.apache.catalina.core.AprLifecycleListener init
信息: The APR based Apache Tomcat Native library which allows optimal performance in production is
not found on the java.library.path: C:\Program Files\Java\jdk1.8.0_111\bin;C:\Windows\Sun\Java\bin;C:\Windows\system32\bin;C:\Windows;C:\Program Files\Python39\Scripts\;C:\Program Files\Python39\bin;C:\Windows\system32\cmd;C:\Windows\System32\Wbem;C:\Program Files\Java\jdk1.8.0_111\bin;C:\Windows\System32\WindowsPowerShell\v1.0\;C:\Windows\System32\OpenSSH\;E:\maven\apache-maven-3.6.3\bin;E:\nodejs\node;E:\nodejs\node_global;C:\Program Files (x86)\Intel\Intel(R) Management Engine Components\bin\

```

拿到项目的发布路径后，进入发布目录，找到work，可以看到JSP在服务器上的解析过程



使用记事本或其他阅读工具，打开a\_jsp.java文件，通过阅读代码会有以下发现

**注意：**HttpJspBase 的父类是 HttpServlet

```

1 package org.apache.jsp;

```



```

2
3 import javax.servlet.*;
4 import javax.servlet.http.*;
5 import javax.servlet.jsp.*;
6
7 public final class a_jsp extends org.apache.jasper.runtime.HttpJspBase
8     implements org.apache.jasper.runtime.JspSourceDependent {
9
10     private static final javax.servlet.jsp.JspFactory _jspxFactory =
11         javax.servlet.jsp.JspFactory.getDefaultFactory();
12
13     private static java.util.Map<java.lang.String,java.lang.Long>
14     _jspx_dependants;
15
16     static {
17         _jspx_dependants = new
18         java.util.HashMap<java.lang.String,java.lang.Long>(2);
19     }
20
21     private javax.el.ExpressionFactory _el_expressionfactory;
22     private org.apache.tomcat.InstanceManager _jsp_instancemanager;
23
24     public java.util.Map<java.lang.String,java.lang.Long> getDependants() {
25         return _jspx_dependants;
26     }
27
28     public void _jspInit() {
29         _el_expressionfactory =
30         _jspxFactory.getJspApplicationContext(getServletConfig().getServletContext())
31         .getExpressionFactory();
32         _jsp_instancemanager =
33         org.apache.jasper.runtime.InstanceManagerFactory.getInstanceManager(getServlet
34         Config());
35     }
36
37     public void _jspDestroy() {
38     }
39
40     public void _jspService(final javax.servlet.http.HttpServletRequest
41     request, final javax.servlet.http.HttpServletResponse response)
42         throws java.io.IOException, javax.servlet.ServletException {
43
44         final javax.servlet.jsp.PageContext pageContext;
45         javax.servlet.http.HttpSession session = null;
46         final javax.servlet.ServletContext application;
47         final javax.servlet.ServletConfig config;
48         javax.servlet.jsp.JspWriter out = null;
49         final java.lang.Object page = this;
50         javax.servlet.jsp.JspWriter _jspx_out = null;
51         javax.servlet.jsp.PageContext _jspx_page_context = null;
52
53         try {
54             response.setContentType("text/html;charset=UTF-8");
55             pageContext = _jspxFactory.getPageContext(this, request, response,
56                 "500.jsp", true, 8192, true);
57             _jspx_page_context = pageContext;
58             application = pageContext.getServletContext();

```

```

53     config = pageContext.getServletConfig();
54     session = pageContext.getSession();
55     out = pageContext.getOut();
56     _jspx_out = out;
57
58     out.write("\r\n");
59     out.write("\r\n");
60     out.write('\r');
61     out.write('\n');
62     out.write("\r\n");
63     out.write("\r\n");
64     out.write("<html>\r\n");
65     out.write("<head>\r\n");
66     out.write("    <title>Title</title>\r\n");
67     out.write("</head>\r\n");
68     out.write("<body>\r\n");
69     out.write("<h1>HelloWorld</h1>\r\n");
70
71     //jsp鎖則ervice()鏄呭彨session漢硅簿鏄岃涸"櫟甯□絳録浹濂戒簡鏄�墜涕 ♡▼蹇忔
鍑 鋳□互鏄�存棦浣跨
72
73     //session.setAttribute("", "");
74
75     //System.out.println(12/0);
76
77     out.write("\r\n");
78     out.write("</body>\r\n");
79     out.write("</html>\r\n");
80     out.write('\r');
81     out.write('\n');
82 } catch (java.lang.Throwable t) {
83     if (!(t instanceof javax.servlet.jsp.SkipPageException)){
84         out = _jspx_out;
85         if (out != null && out.getBufferSize() != 0)
86             try { out.clearBuffer(); } catch (java.io.IOException e) {}
87         if (_jspx_page_context != null)
88             _jspx_page_context.handlePageException(t);
89         else throw new ServletException(t);
90     } finally {
91         _jspxFactory.releasePageContext(_jspx_page_context);
92     }
93 }
94 }

```

## 2-1 jspのpage指令

### 设置errorPage

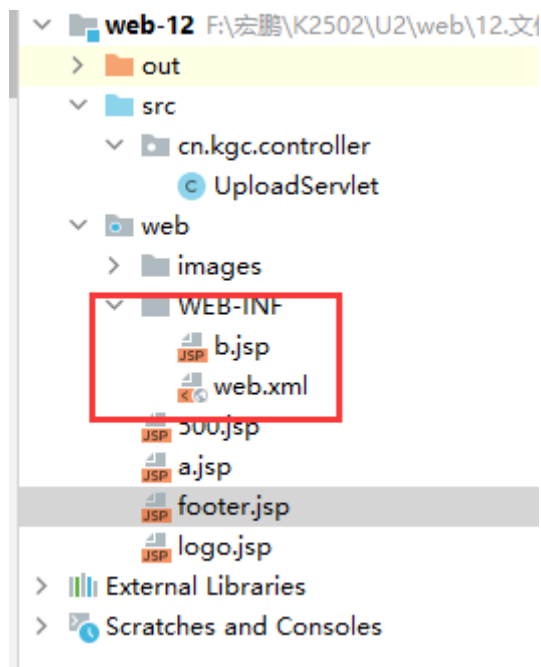
```
1 <%@ page contentType="text/html; charset=UTF-8" language="java"
  isErrorPage="true" %>
2 <html>
3 <head>
4   <title>错误页</title>
5 </head>
6 <body>
7 
8 </body>
9 </html>
10
```

## 设置jsp出错时，进入错误页

```
1 <!--
2   errorPage="指定出错显示页面"
3 --%>
4 <%@ page contentType="text/html; charset=UTF-8" language="java"
  errorPage="500.jsp" %>
5
6 <html>
7 <head>
8   <title>Title</title>
9 </head>
10 <body>
11 <h1>HelloWorld</h1>
12 <%
13     //jsp的service()内置session对象，服务器帮你创建好了，所以程序员可以直接使用
14
15     //session.setAttribute("", "");
16
17     System.out.println(12/0);
18 %>
19 </body>
20 </html>
```

## 2-2 JSPの配置

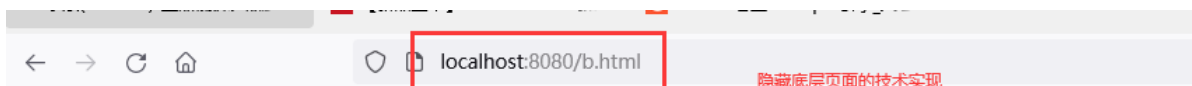
将jsp页面放在WEB-INF下面



## 在web.xml配置b.jsp的访问路径

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <web-app xmlns="http://java.sun.com/xml/ns/javaee"
3         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4         xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
5         http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
6         version="2.5">
7     <!--jsp配置-->
8     <servlet>
9         <servlet-name>bb</servlet-name>
10        <jsp-file>/WEB-INF/b.jsp</jsp-file>
11    </servlet>
12    <servlet-mapping>
13        <servlet-name>bb</servlet-name>
14        <url-pattern>/b.html</url-pattern>
15    </servlet-mapping>
16 </web-app>
17
```

## 浏览器访问的方式:



共同的头部信息 HelloWorld,b.html 友情链接

## 2-3 jsp的include指令

include:主要是实现多个页面的合并响应。

优点: 解决jsp页面重复设计代码、重复的java代码问题

实现方式如下：

## header.jsp

```
1 <%@ page contentType="text/html; charset=UTF-8" language="java" %>
2 <html>
3 <head>
4     <title>Title</title>
5 </head>
6 <body>
7     共同的头部信息
8 </body>
9 </html>
10
```

## footer.jsp

```
1 <%@ page contentType="text/html; charset=UTF-8" language="java" %>
2 <html>
3 <head>
4     <title>Title</title>
5 </head>
6 <body>
7     友情链接
8 </body>
9 </html>
10
```

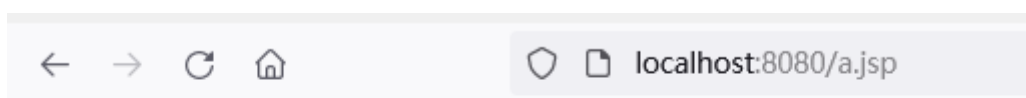
## 设计请求的页面：a.jsp

```
1 <!--
2     errorPage="指定出错显示页面"
3 --%>
4 <%@ page contentType="text/html; charset=UTF-8" language="java"
5     errorPage="500.jsp" %>
6 <!-- 多个jsp合并 --%>
7 <%@include file="logo.jsp"%>
8 <html>
9 <head>
10     <title>Title</title>
11 </head>
12 <body>
13 <h1>Hello world</h1>
14 <%
15     System.out.println(12/2);
16 %>
17 </body>
18 </html>
19 <%@include file="footer.jsp"%>
```

## 设计请求的页面：b.jsp

```
1 <%@ page contentType="text/html;charset=UTF-8" language="java" %>
2 <%@include file="../logo.jsp"%>
3 <html>
4 <head>
5     <title>Title</title>
6 </head>
7 <body>
8     HelloWorld,b.html
9 </body>
10 </html>
11 <%@include file="../footer.jsp"%>
12
```

## 浏览器访问a.jsp效果



共同的头部信息

# HelloWorld

友情链接

## 课程总结

- 1 1 文件上传的基本流程
- 2 2 文件上传的细节实现：
  - 3 2-1 文件重名
  - 4 2-2 文件中文名乱码
  - 5 2-3 文件大小限制
  - 6 2-4 文件格式显示
- 7 3 JSP工作原理
  - 8 3-1 work目录的跟踪
  - 9 3-2 jsp在服务器端的解析过程
  - 10 3-3 jsp常用指令page和include的基本使用案例

## 课程预习

ajax实现：

原生态代码实现机制

jQuery提供ajax(,,,)

前后端分离项目：基于ajax axios

