**文件上传和下载**

# 授课内容

授课科目：JavaEE

授课内容：文件上传和下载

授课类型：讲授+实践

授课时间：3+1学时

主讲教师：孟双英

# 教学目标要求

## 能力目标：

1. 培养学生融会贯通知识的能力；
2. 培养学生了解互联网中文件上传的诸多应用场景。
3. 培养学生分析和解决问题的能力。

## 知识目标：

1. 了解Java开发中各类文件上传组件；
2. 掌握Apache commons-fileupload组件实现文件上传；
3. 掌握文件的下载。

# 课件分析

概 述：回顾Web开发中学过的表单元素，强调最常用的几种表单元素的使用注意事项，引出文件上传的概念和应用场景，介绍Java中实现文件上传常用的第三方插件，重点介绍Apache commons-fileupload插件及其实现文件上传的流程，引申学生思考文件上传中的一些问题；第二部分内容重点介绍文件的下载，以及在Java程序中实现文件下载的流程。

教学重点：Apache commons-fileupload实现文件上传，文件下载。

教学难点：Apache commons-fileupload的API。

# 教学方法

问题教学法、讲授法、演示法等。

# 教学过程

## 课前补充

（5分钟）

文件上传和下载是Web应用中非常常用的功能，常见到的一些文件上传和下载的应用场景，比如：头像的上传，各大运行商的网盘 （即包括文件的上传又包括文件的下载），管理系统中规章制度的上传和下载，QQ空间中照片的上传和分享等等。

## 上节回顾

（0分钟）

## 作业点评

（0分钟）

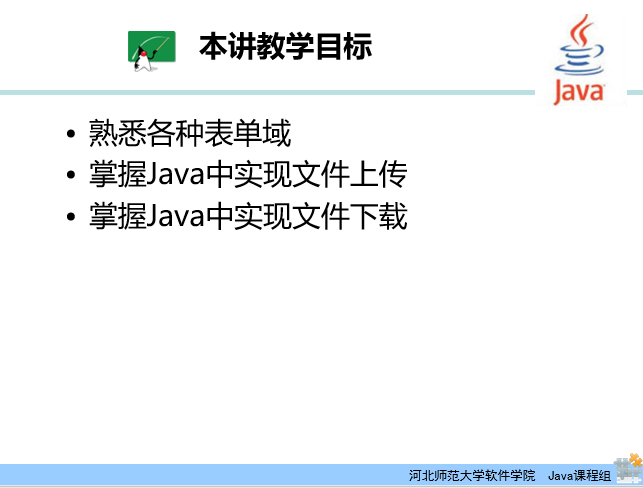
## 导入新课

（2分钟）

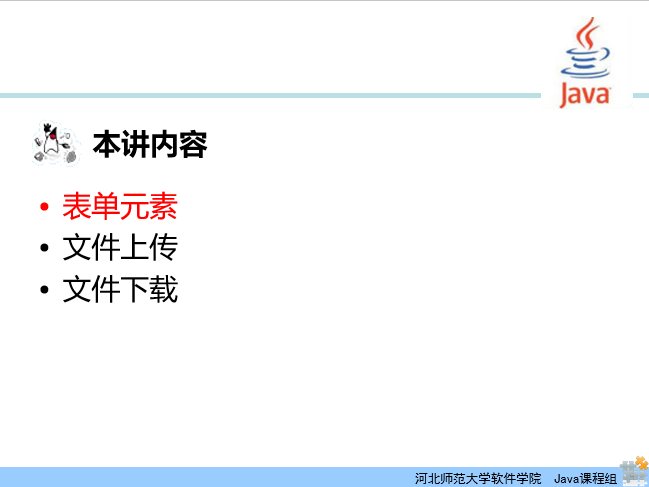
不同的后台开发技术实现文件的上传和下载的具体代码和语法是不同的，但是基本流程都是一致的，并且对于前端的html页面的代码基本是一致的，故今天的课程首先介绍实现文件上传html页面如何来实现，随后再介绍Java程序中如何接收客户端传递过来的文件信息。

## 讲授新课

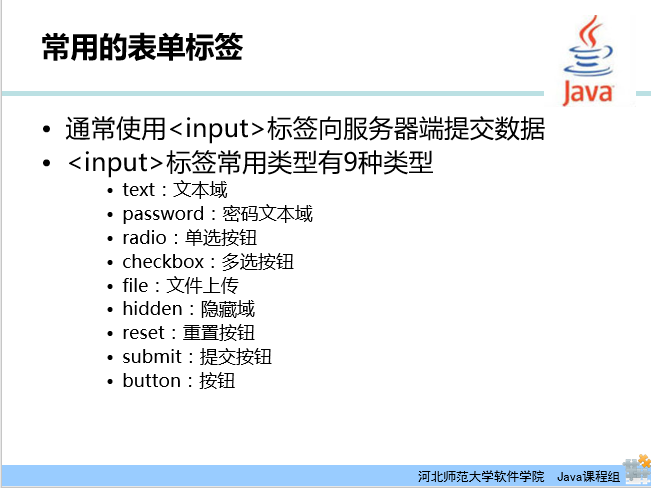
（120分钟）



本章的教学目标：回顾各种表单元素及其使用，掌握Java中实现文件上传的插件和编程流程，掌握Java中实现文件下载的流程。



从客户端到服务器端的信息传递离不开Web开发中学习过的表单元素，一起回顾常用的表单元素。



input标签是表单元素里使用最广泛的，其type属性取值的不同可以代表9类不同的表单元素。

分别是：<input type=”text” />单行文本域

<input type=”password”/>密码域

<input type=”radio”/>单选按钮

<input type=”checkbox”/>多选域

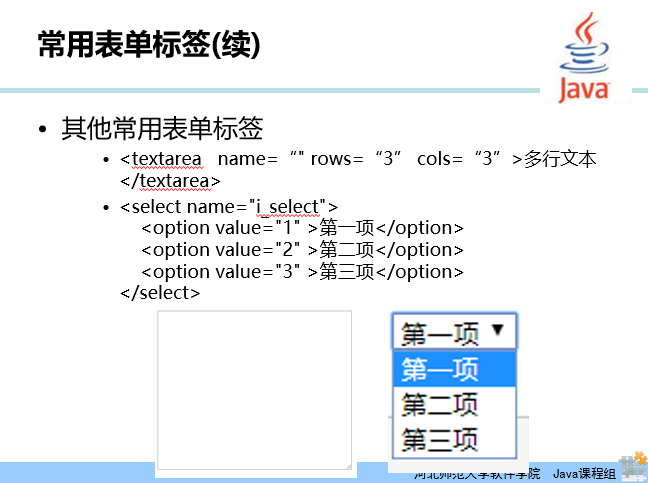
<input type=”hidden”/>隐藏域，在以前的Web开发课程中提及比较少，课堂上演示其使用。

<input type=”file”/>文件域

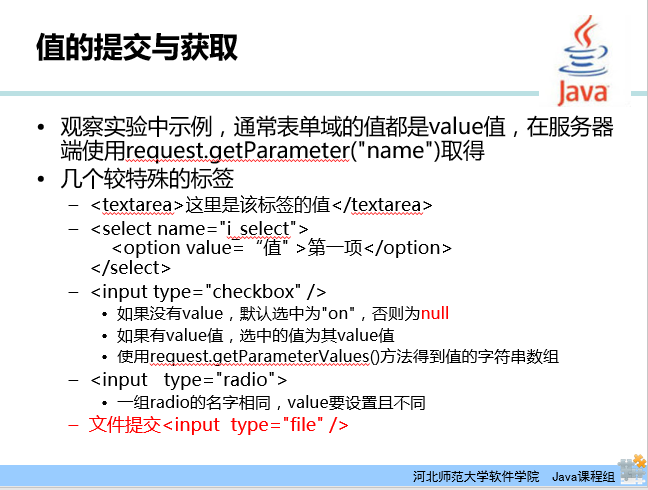
<input type=”reset”/>重置按钮

<input type=”submit”/>提交按钮

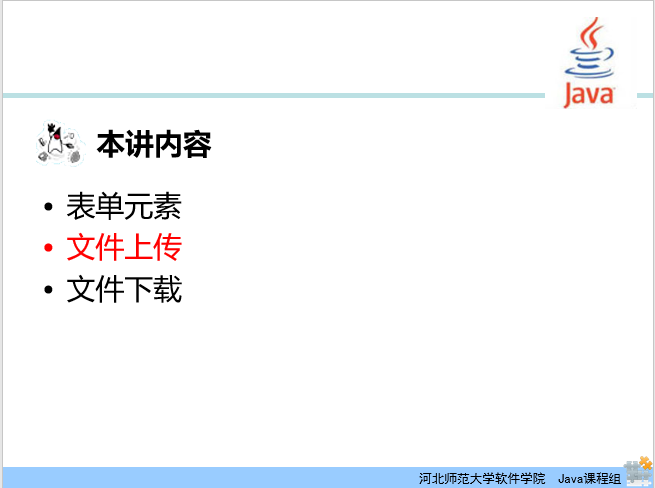
<input type=”button”/>普通按钮



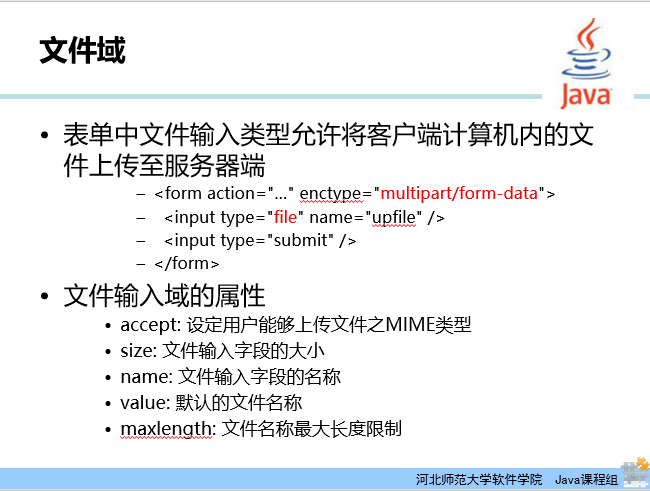
除input标签，表单元素中还有<textarea>标签和<select><option>标签，<textarea>标签代表多行文本框，rows属性用于设置文本框的高度，cols属性用于设置文本框的宽度；<select>代表下拉选择框，选择框中的每一个选项用一个<option>标签表示。



通常的表单元素提交到服务器上的信息是其value属性的值，在服务器端通过request.getParameter("name属性值")来得到信息，而有一些特殊的表单标签提交到服务器端的信息比较特殊，首先第一个是<textarea>标签，其提交到服务器端的信息是<textarea>和</textarea>之间的文本信息；再者<input type=”checkbox”/>标签，多选框在页面上是可以多选的，为了表示一组多选往往设置其name属性值为相同的值，在服务器端通过request.getParameterValues()方法得到一组值，并且得到的也是其value属性的值，但如果标签中未设置value属性则得到的选中的是on，未选中的是null；<input type="radio" />与checkbox是一致的；最后一个特殊的表单元素是<input type="file"/>文件域，专门用于从客户端传递文件到服务器端的表单元素，其提交的也不仅仅是value属性的值。

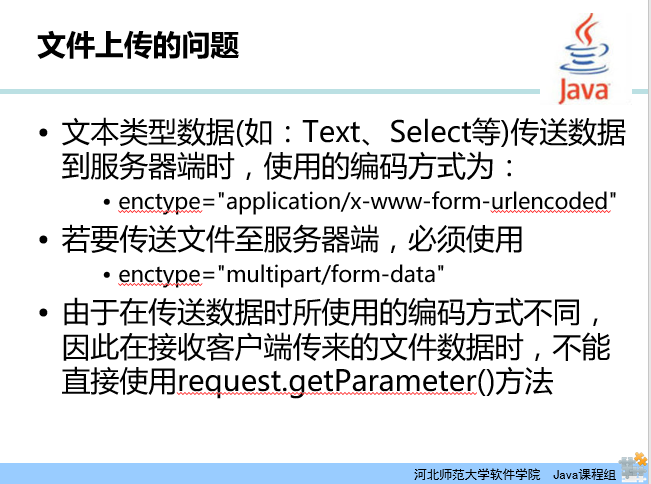


下面就来介绍文件的上传。

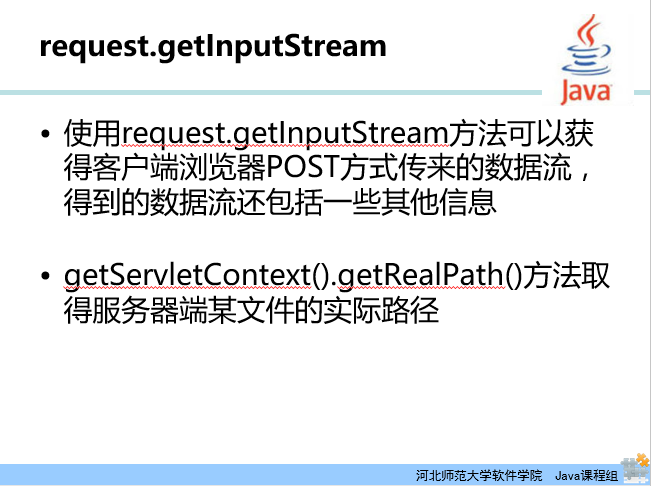


普通的文本信息与文件信息上传至服务器端，form表单的设置是不同的，最主要的区别是enctype属性的设置，enctype 属性规定了在信息发送到服务器之前应该如何对表单数据进行编码。常用的enctype属性的取值有两种，一种是application/x-www-form-urlencoded，默认值，表示信息在发送到服务器之前，所有字符都会进行编码（空格转换为 "+" 加号，特殊符号转换为 ASCII HEX 值）；第二种是multipart/form-data表示不对字符编码，在使用包含文件域上传文件的表单中，必须使用该值。

文件域常用的一些属性包括：accept属性，用于设定上传文件的MIME类型；size属性，用于设置上传文件的大小；name属性，文件域的名称；value属性，文件域的默认文件名称；maxlength，文件名的最大长度限制。这些属性可以对上传文件设置一些条件限制，实际开发中会用到。



由于文本和文件信息在form中已设置了不同的编码类型，故在服务器端接收时，文件的接收和文本的接收是不同的，文本信息的接收通过request.getParameter()方法，而文件信息的接收却不能使用这个方法。

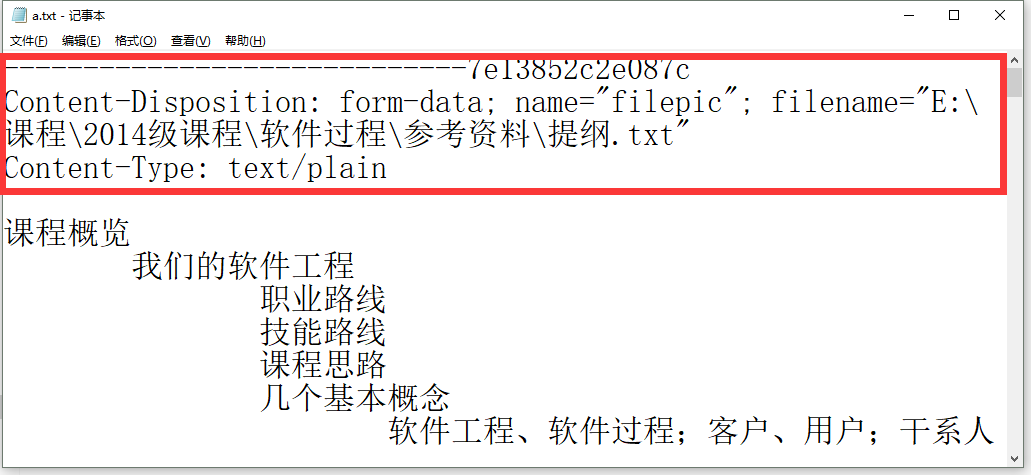


最简单的文件的接收是使用request.getInputStream()方法得到客户端的输入流，以流的方式来获取客户端传递过来的文件信息，核心代码如下：

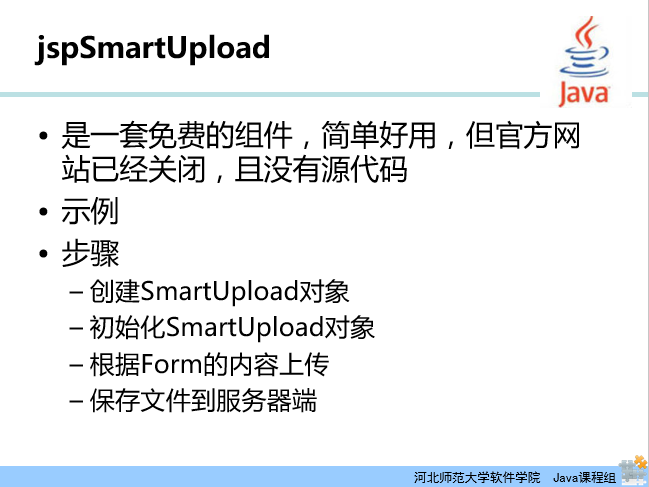
|  |
| --- |
| request.setCharacterEncoding("UTF-8");  response.setCharacterEncoding("UTF-8");  ServletInputStream in = request.getInputStream();  BufferedReader reader=new BufferedReader(new InputStreamReader(in));  File file = new File("d:/a.txt");  FileWriter fw = new FileWriter(file);  BufferedWriter bw = new BufferedWriter(fw);  String s = reader.readLine();  while(s!=null){  bw.write(s);  bw.newLine();  s=reader.readLine();  }  bw.flush();  bw.close();  in.close(); |

分析示例中将上传文件写死在了D：/根路径下，如果服务器端没有D盘（或者服务器根本就不是windows系统）该如何？一种解决方案是根据当前站点的路径找一个相对位置存储文件，可调用getServletContext().getRealPath(“/”)获取站点根路径。

再分析通过流的方式读取客户端传递来的文件信息，不仅得到了文件的信息还包括请求的相关头信息，如下图所示：



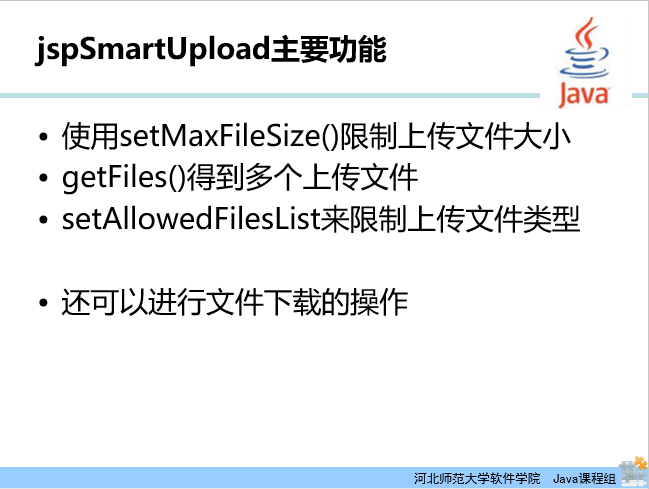
很显然服务器端接收的结果是不合理的，为了得到合理的文件信息，需要再服务器端对流中的数据再做进一步的解析和判断，过程比较繁琐，实际的开发中会借助第三方的上传插件来实现合理的文件上传。本章介绍三个插件，都是针对Java开发。



JSPSmartUpload插件，是一套免费的、简单好用的文件上传的插件，但其目前已经停止更新，故作为了解看看示例程序即可，现在的实际项目中应用也比较少。



jspSmartUpload程序示例。



jspSmartUpload插件中可以用于对上传文件限制的一些方法，简单了解即可。



O’reilly cos 插件，是O’reilly公司提供的一个文件上传的插件，非常简洁，看示例代码即可体会。

|  |
| --- |
| PrintWriter out = response.getWriter();  //文件上传后，保存在D:\\upload  String saveDirectory ="D:\\upload";  //每个文件最大5m,最多3个文件,所以  int maxPostSize =3 \* 5 \* 1024 \* 1024 ;  //response的编码为"gb2312",同时采用缺省的文件名冲突解决策略,实现上传  //就这一句就完成上传了  MultipartRequest multi = new MultipartRequest(request, saveDirectory, maxPostSize,"UTF-8");    //输出反馈信息  Enumeration files = multi.getFileNames();  while (files.hasMoreElements()) {  String name = (String)files.nextElement();  File f = multi.getFile(name);  if(f!=null){  String fileName = multi.getFilesystemName(name);  String lastFileName= saveDirectory+"\\" + fileName;  out.println("上传的文件:"+lastFileName);  out.println("<hr>");  }  } |

核心代码只有红色字体部分。在以上示例代码中是原模原样保存了上传文件，如果想重命名上传文件的名称可以重写插件中的FileRenamePolicy接口中的rename()方法即可，将红色代码替换为以下代码即可：

|  |
| --- |
| RandomFileRenamePolicy rfrp=new RandomFileRenamePolicy();  MultipartRequest multi = new MultipartRequest(request, saveDirectory, maxPostSize,"gb2312",rfrp); |

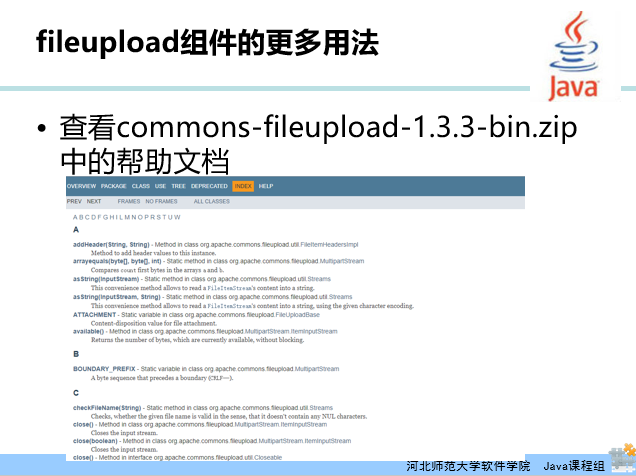


下面介绍Java文件上传中最重要的一个插件，也是最常用的插件，Apache公司的commons-fileupload插件，它本身依赖于Apache公司的io插件，故项目中使用此插件必须同时导入commons-fileupload.jar和commons-io.jar。

fileupload插件最核心的三个类包括：DiskFileUpload、FileItem和FileUploadException。DiskFileUpload类是Apachecommons-fileupload的核心类，应用程序通过这个类来与Apache文件上传组件进行交互；FileItem类用来封装单个表单字段元素数据的，一个表单元素对应一个FileItem对象，通过调用FileItem对象的方法可以获得相关表单元素的数据；FileUploadException类表示在文件上传过程中，可能发生各种各样的异常，例如网络中断、数据丢失等等。

commons-fileupload插件的示例代码：

|  |
| --- |
| DiskFileItemFactory factory = new DiskFileItemFactory();  ServletFileUpload upload = new ServletFileUpload(factory);  upload.setSizeMax(-1);//设置上传文件限制大小,-1无上限  try {  List<FileItem> list = upload.parseRequest(request);  String va = null;  String realPath="D:/upload/";  for(FileItem item : list){  if(item.isFormField()){//判断是否是文件流  //如果是文本框，获取文本框的内容：  va = item.getString();  }else{//是文件流则读取文件，保存文件  String value = item.getName();//会将完整路径名传过来  int start = value.lastIndexOf("\\");  String fileName = value.substring(start+1);  item.write(new File(realPath,fileName));  }  }  } catch (FileUploadException e) {  // TODO Auto-generated catch block  e.printStackTrace();  } |



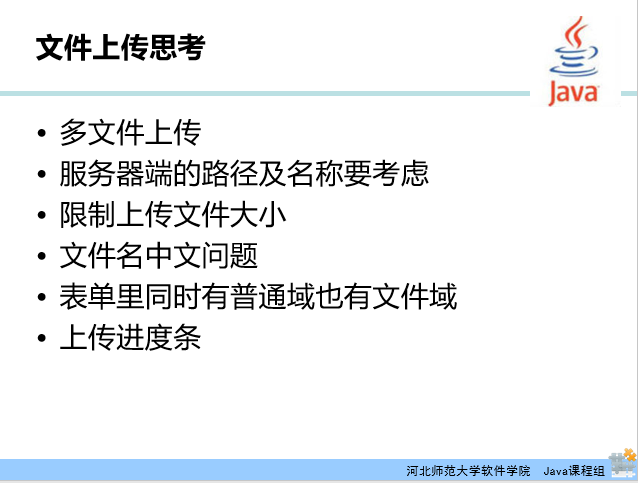
fileupload插件的功能非常强大，比如：重命名上传文件，限制上传文件的大小、类型等。

示例：限制上传文件的类型

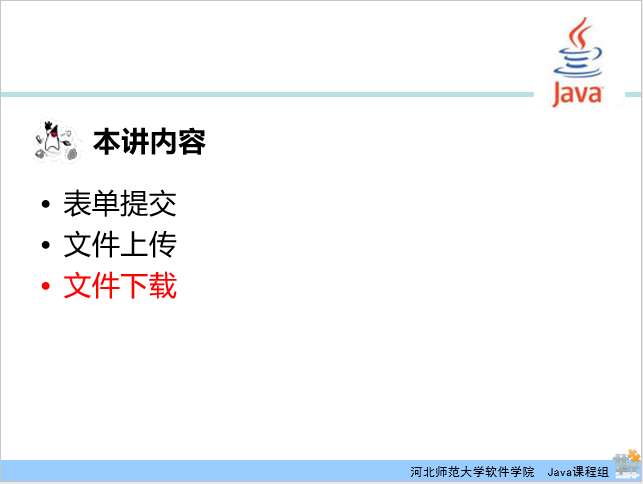
|  |
| --- |
| if (!item.getContentType().startsWith("image")) {  return;  } |

示例：限制上传文件的大小

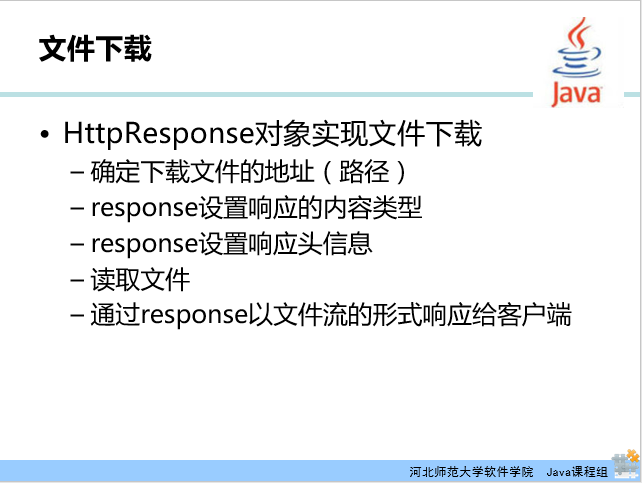
|  |
| --- |
| //限制上传文件大小  upload.setFileSizeMax(1024 \* 196);// 限制单个文件的大小不超过200kb  //upload.setSizeMax(5\*1024\*1024);//限制上传的总的文件大小不超过5M |



在实际的文件上传中，会遇到各种特殊的需求，比如同时上传多个文件，服务器端的存储路径，限制文件的大小和类型，文件名的重命名，以及文件域和文本域混合提交，以及大文件在上传过程中的进度条提示等等，其他的问题课程中或多或少都介绍了，进度条显示进度没涉及，留作课后思考题。



文件上传的问题解决了，与之相对的是文件下载。



与文件上传相比较，文件下载较简单，客户端到服务器端文件的传送是通过流的方式，同样，服务器端到客户端文件的传送也要通过流的方式。

将要下载的文件转换成字节流通过response对象响应给客户端即可，客户端浏览器接收到响应后会根据响应的内容类型选择不同的方式来展示，如果是流形式，则浏览器会启动下载窗口，用户可以选择将文件保存到本地的某一路径下即可。

具体的步骤：

1.确定下载文件的地址；

2.设置响应的内容类型；

3.设置响应的头信息；

4.使用response将文以流的形式传送给客户端。

核心示例代码：

|  |
| --- |
| request.setCharacterEncoding("UTF-8");  response.setCharacterEncoding("UTF-8");  //设置文件MIME类型  String filename="教学大纲C++.xlsx";  response.setContentType(getServletContext().getMimeType(filename));  //设置Content-Disposition  response.setHeader("Content-Disposition", "attachment;filename="+filename);  //读取目标文件，通过response将目标文件写到客户端  //获取目标文件的绝对路径  String fullFileName="D://"+filename;  //System.out.println(fullFileName);  //读取文件  InputStream in = new FileInputStream(fullFileName);  OutputStream out = response.getOutputStream();  //写文件  int b;  while((b=in.read())!= -1)  {  out.write(b);  }  in.close();  out.close(); |

## 课时小结

（10分钟）

1. 复习表单元素及其使用注意事项。
2. Java实现文件上传常用的插件。
   1. O’reilly cos插件；
   2. Apache commons-fileupload插件；
3. Java实现文件下载。

# 布置作业

（5分钟）

完成上传头像的实验手册。

实现多文件的上传。

实现限制上传文件的类型和大小的多文件上传。

课程案例代码在fileuploadtest项目中。