# 文件上传入门

## 实现文件上传条件

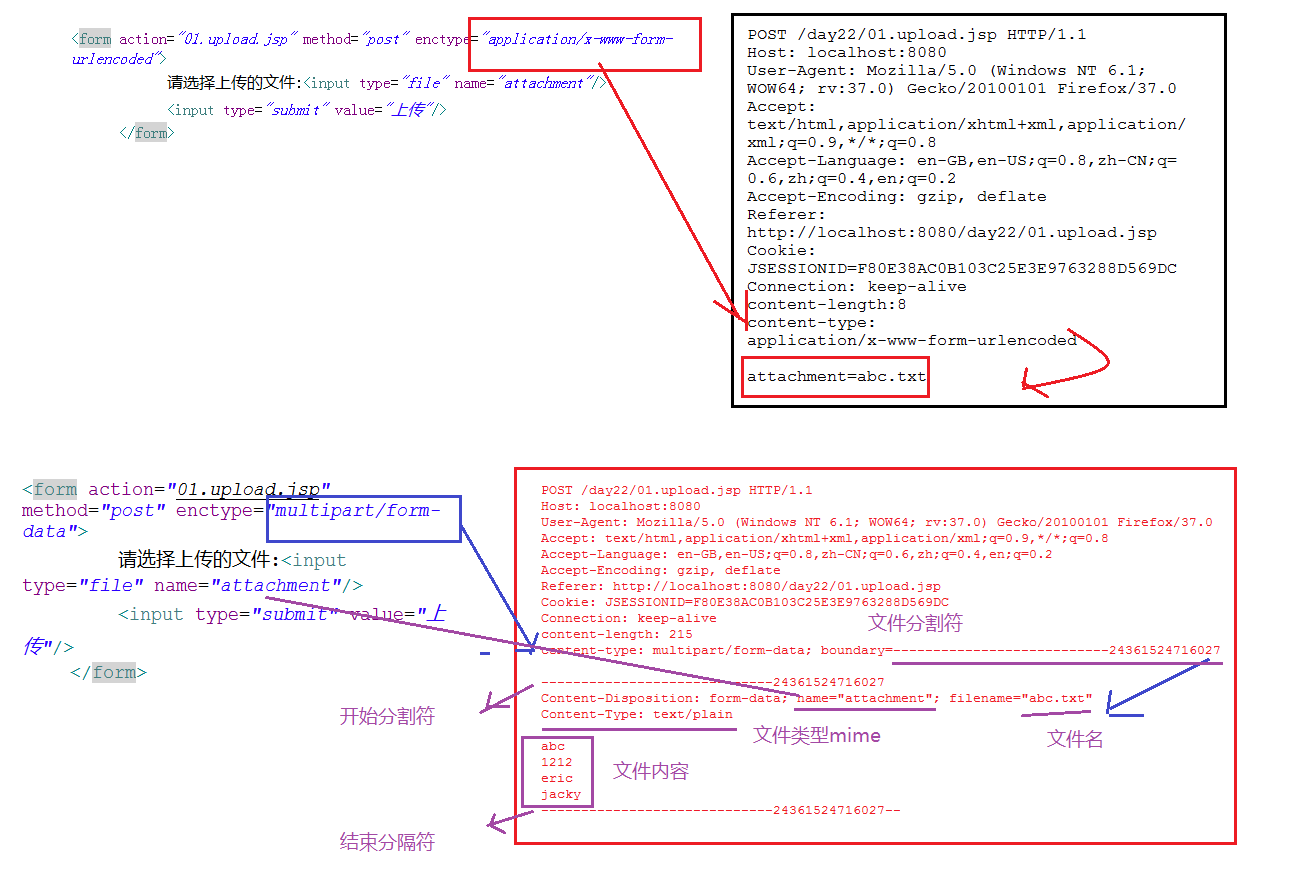
1．表单的提交方式必须是POST方式。（才有content-type属性）

2．有文件上传表单，表单中有<input type="file"/>的选择文件的标签

3．把表单设置为enctype="multipart/form-data"，提交的数据不再是key-value对，而是字节数据

|  |
| --- |
| <form action=*"*${pageContext.request.contextPath }*/UploadDemo1"* **method=*"post"* enctype=*"multipart/form-data"***>  请选择文件： **<input type=*"file"* name=*"img"*/><br/>**  <input type=*"submit"* value=*"上传"* />  </form> |

### 上传原理



## 2．手动解析上传文件

|  |
| --- |
| /\*\*  \* 手动处理上传文件的逻辑  \* **@author** APPle  \*  \*/  **public** **class** UploadDemo1 **extends** HttpServlet {  **public** **void** doGet(HttpServletRequest request, HttpServletResponse response)  **throws** ServletException, IOException {  //得到实体内容数据  InputStream in = request.getInputStream();  BufferedReader br = **new** BufferedReader(**new** InputStreamReader(in));    //读取文件的开始符  String startTag = br.readLine();    //读取文件名： Content-Disposition: form-data; name="img"; filename="news.txt"  String line = br.readLine();  String fileName = line.substring(line.lastIndexOf("filename=\"")+10, line.lastIndexOf("\"") );  System.*out*.println("文件名："+fileName);    //跳过2行  br.readLine();  br.readLine();    //读取文件的实际内容  String str = **null**;  BufferedWriter bw = **new** BufferedWriter(**new** FileWriter("E:/files/"+fileName));  **while**((str=br.readLine())!=**null**){  //读到文件结束符时退出循环  **if**((startTag+"--").equals(str)){  **break**;  }    //把内容写出文件中  bw.write(str);  bw.newLine();  bw.flush();  }  //关闭  bw.close();  br.close();  }  **public** **void** doPost(HttpServletRequest request, HttpServletResponse response)  **throws** ServletException, IOException {  doGet(request, response);  }  } |

# 工具实现文件上传

## 1．commons-fileupload组件

是Apache组织旗下的开源的文件上传的组件。使用非常简单易用。

## 2．核心的API

DiskFileItemFactory类： 用于创建上传对象，设置文件缓存区大小，设置文件缓存目录。

ServletFileUpload类： 用于在Servlet程序中实现文件上传

List<FileItem> list = parseRequest(request): 用于解析请求数据，提取和封装文件信息。

FileItem类： 封装一个文件的所有相关的信息javabean。包含文件名称，文件大小，文件类型， 文件数据内容。

## 3．开发步骤

1．导入commoms-fileuload的jar包

commons-fileupload-1.2.2.jar 核心包

commons-io-2.1.jar 辅助包

2．编写程序

# 实现单文件上传

写在Servlet的doGet中

|  |
| --- |
| //1.创建DiskFileItemFactory类  /\*\*  \* 参数一： 表示文件缓存区的大小。如果上传的文件没有超过缓存区大小，则文件不缓存；否则缓存文件，缓存到临时目录。（byte）  \* 参数二： 表示缓存区的临时目录。  \*/  DiskFileItemFactory factory = **new** DiskFileItemFactory(10\*1024,**new** File("e:/temp/"));    //2.创建ServletFileUpload类  ServletFileUpload upload = **new** ServletFileUpload(factory);    /\*\*  \* 设置文件名的编码  \*/  upload.setHeaderEncoding("utf-8");    //3.解析request数据（把每一个文件封装到FileItem对象中，FileItem放入List中）  **try** {  List<FileItem> list = upload.parseRequest(request);    //取出第一个上传的文件  FileItem file = list.get(0);    //得到文件名(getName())  String fileName = file.getName();  //得到文件大小  **long** fileSize = file.getSize();  //得到内容类型  String contentType = file.getContentType();  //得到文件数据内容  InputStream in = file.getInputStream();    /\*\*  \* 4.把文件数据内容存储到服务器端的硬盘中  \*/  FileUtils.*copyInputStreamToFile*(in, **new** File("e:/files/"+fileName));    /\*\*  \* 5.文件上传完毕，手动清理缓存文件  \*/  file.delete();      System.*out*.println("文件名："+fileName);  System.*out*.println("文件大小："+fileSize);  System.*out*.println("文件类型："+contentType);  System.*out*.println("文件数据内容："+in);    } **catch** (FileUploadException e) {  e.printStackTrace();  } |

# 实现多文件上传

|  |
| --- |
| //1.创建DiskFileItemFactory对象  DiskFileItemFactory factory = **new** DiskFileItemFactory(10\*1024, **new** File("e:/temp/"));  //2.创建ServletFileUpload对象  ServletFileUpload upload = **new** ServletFileUpload(factory);  //3.设置文件编码  upload.setHeaderEncoding("utf-8");  //4.开始解析文件  **try** {  List<FileItem> list = upload.parseRequest(request);  **if**(list!=**null**){  List<UploadFile> ufList = **new** ArrayList<UploadFile>();  //遍历多个文件  **for**(FileItem file: list){  //取出文件相关信息  String fileName = file.getName();  **long** fileSize = file.getSize();  String contentType = file.getContentType();  //封装到javabean中  UploadFile uf = **new** UploadFile();  uf.setFileName(fileName);  uf.setFileSize(fileSize);  uf.setFileType(contentType);  //放入list中  ufList.add(uf);    //把文件保存到服务器端的硬盘  FileUtils.*copyInputStreamToFile*(file.getInputStream(), **new** File("e:/files/"+fileName));  //删除缓存文件  file.delete();  }  request.setAttribute("ufList", ufList);  request.getRequestDispatcher("/success.jsp").forward(request, response);  }  } **catch** (FileUploadException e) {  e.printStackTrace();  } |

# 动态选择多文件上传

|  |
| --- |
| <html>  <head>  <title>使用组件实现动态多文件上传</title>  </head>    <body>  <form action=*"*${pageContext.request.contextPath }*/UploadDemo3"* method=*"post"* enctype=*"multipart/form-data"* name=*"uploadForm"*>  <table border=*"1"* width=*"400px"*>  <tbody>  <tr id=*"1"*>  <td>  请选择文件：  </td>  <td>  <input type=*"file"* name=*"file"*/><input type=*"button"* value=*"删除"* onclick="delItem(1)"/>  </td>  </tr>  </tbody>  <tr>  <td colspan=*"2"*><input type=*"button"* value=*"添加"* onclick="addIten()"/></td>  </tr>  <tr>  <td colspan=*"2"*><input type=*"button"* value=*"上传"* onclick="checkSunbit()"/></td>  </tr>  </table>  </form>    <script type=*"text/javascript"*>  **var** id = 2;  //添加一行  **function** addIten(){  **var** trNode = document.createElement("tr");  trNode.setAttribute("id", id);    **var** tdNode1 = document.createElement("td");  tdNode1.innerHTML = "请选择文件：";    **var** tdNode2 = document.createElement("td");  **var** input1 = document.createElement("input");  input1.setAttribute("type", "file");  input1.setAttribute("name", "file");  **var** input2 = document.createElement("input");  input2.setAttribute("type", "button");  input2.setAttribute("value", "删除");  input2.setAttribute("onclick", "delItem("+id+")");    tdNode2.appendChild(input1);  tdNode2.appendChild(input2);    trNode.appendChild(tdNode1);  trNode.appendChild(tdNode2);    //把trNode放入table的tbodyNode中  **var** tbodyNode = document.getElementsByTagName("tbody")[0];  tbodyNode.appendChild(trNode);    id++;    }    //删除一行（根据tr的id值删除）  **function** delItem(id){  **if**(id>1){  **var** trNode = document.getElementById(id);  **var** tbodyNode = document.getElementsByTagName("tbody")[0];  tbodyNode.removeChild(trNode);    id--;  }  }    //提交并且检查file属性  **function** checkSunbit(){  //检查file属性是否全部填上  **var** fileList = document.getElementsByName("file");  **for**(**var** i=0;i<fileList.length;i++){  //如果为选择file，则其value值为空  **if**(fileList[i].value==**null** || fileList[i].value==""){  alert("请选择第"+(i+1)+"个文件");  **return**;  }  }    //提交表单  **var** form = document.forms['uploadForm'];  form.submit();  }  </script>  </body>  </html> |

# 文件上传的细节

## 1．限制文件类型

|  |
| --- |
| //得到文件类型  String contentType = file.getContentType();  System.*out*.println(contentType);  //如果是图片,才可以上传(image/bmp，jepg，jpg，gif)  **if**(!contentType.toLowerCase().matches("image/[a-z]\*")){  **throw** **new** FileTypeErrorException("文件类型不符合条件！");  } |

## 2．限制上传文件大小

**ServletFileUpload类：**

setFileSizeMax：设置单个文件的最大容量。

setSizeMax ： 设置所有文件的最大容量。

|  |
| --- |
| //4.解析请求  **try** {  List<FileItem> list = upload.parseRequest(request);  **if**(list!=**null**){  **for**(FileItem file : list){  /\*\*  \* 限制文件类型  \*/  //得到文件类型  String contentType = file.getContentType();  System.*out*.println(contentType);  //如果是图片,才可以上传(image/bmp，jepg，jpg，gif)  **if**(!contentType.toLowerCase().matches("image/[a-z]\*")){  **throw** **new** FileTypeErrorException("文件类型不符合条件！");  }    //保存文件  FileUtils.*copyInputStreamToFile*(file.getInputStream(), **new** File("e:/files/"+file.getName()));  //删除缓存文件  file.delete();  }  }  } **catch (FileTypeErrorException e) {**  **//e.printStackTrace();**  **//处理文件类型错误的异常**  **request.setAttribute("message", e.getMessage());**  **request.getRequestDispatcher("/05.upload.jsp").forward(request, response);**    **} catch (FileSizeLimitExceededException e) {**  **//e.printStackTrace();**  **//处理文件超过限制的异常**  **request.setAttribute("message", "单个文件不能超过1M");**  **request.getRequestDispatcher("/05.upload.jsp").forward(request, response);**    **} catch (SizeLimitExceededException e) {**  **//e.printStackTrace();**  **//处理文件超过限制的异常**  **request.setAttribute("message", "所有文件不能超过5M");**  **request.getRequestDispatcher("/05.upload.jsp").forward(request, response);**    } **catch** (FileUploadException e) {  e.printStackTrace();  } |

## 3．查看文件上传的进度

ServletFileUpload类：

setProgressListener(进度监听器): 设置文件上传的监听器

## 4．处理文件名重复问题

|  |
| --- |
| /\*\*  \* 解决文件名重复问题：  \* 1）日期\_时间\_随机数.jpg  \* 2）使用UUID算法（在一台PC都是唯一的）  \*/  String fileName = file.getName();  //得到源文件的后缀名  String supfix = fileName.substring(fileName.lastIndexOf(".")); //.jpg  //使用UUID算法生成随机名称  fileName = UUID.*randomUUID*().toString()+supfix; |

## 5．把同一个目录的文件打散到不同的目录下

假如所有文件都放在同一个目录下，不方便管理。

|  |
| --- |
| /\*\*  \* 注意： 使用UUID算法生成唯一的文件名，避免因为文件名相同覆盖原有的文件  \*/  String uuid = UUID.*randomUUID*().toString();  String fileName = item.getName();  //后缀名  fileName.substring(fileName.lastIndexOf("."));  //新文件名  String newFileName = uuid+fileName;    //新的目录结构  String baseDir = "e:/files/";  String subDir = makeDirectory(newFileName);//使用算法生成的子目录结构  String finalDir = baseDir + subDir;  //生成目录结构  File file = **new** File(finalDir);  **if**(!file.exists()){  //构建目录结构  file.mkdirs();  }    //保存  FileUtils.*copyInputStreamToFile*(item.getInputStream(), **new** File(finalDir+newFileName));    //删除临时文件  item.delete(); |

|  |
| --- |
| **private** String makeDirectory(String fileName){  //1.得到文件名的hashCode值  **int** code = fileName.hashCode();    //2.算出第一层目录的名称  **int** first = code & 0xF;    //3.算出第二层目录的名称  **int** second = code & (0xF>>1);  **return** first+"/"+second+"/";  } |

## 6．得到普通文本控件内容

|  |
| --- |
| //判断该FileItem是否是文件还是普通文本空间  **if**(file.isFormField()){  //普通文本空间（text/password/checkbox/radio/select/texearea）  //得到控件内容    //处理一个普通文本  /\*String info = file.getString("utf-8");  System.out.println("描述："+info);\*/    //处理多个普通文本  String fieldName = file.getFieldName();  **if**("info1".equals(fieldName)){  String info1 = file.getString("utf-8");  System.*out*.println("描述1："+info1);  }  **if**("info2".equals(fieldName)){  String info2 = file.getString("utf-8");  System.*out*.println("描述2："+info2);  } |

# 文件下载

## 1．普通文件下载

使用超链接。缺点：1）暴露文件的路径 2）扩展型和安全性不好

## 2．使用servlet程序下载（推荐）

|  |
| --- |
| //得到需要下载的文件  String path = **this**.getServletContext().getRealPath("/upload/9/1/图片1.png");    File file = **new** File(path);  //读取服务器本地的文件  FileInputStream in = **new** FileInputStream(file);    /\*\*  \* 处理URL编码问题  \*/  String fileName = file.getName();  //对文件名进行URl编码  fileName = URLEncoder.*encode*(fileName, "utf-8");    //判断不同浏览器  String userAgent = request.getHeader("user-agent");  String filefix = **null**;  **if**(userAgent.contains("Trident")){  //IE  filefix = "filename="+fileName;  }**else** **if**(userAgent.contains("Firefox")){  //Firefox  filefix = "filename\*="+fileName;  }**else**{  filefix = "filename="+fileName;  }    //告诉浏览器以下载方式打开资源  response.setHeader("Content-Disposition", "attachment;"+filefix);    //把本地文件发送给浏览器  **byte**[] buf = **new** **byte**[1024];  **int** len = 0;  **while**( (len=in.read(buf))!=-1 ){  response.getOutputStream().write(buf, 0, len);  }  //关闭  in.close(); |

# 案例 -- 文件的上传下载

[D:\教程\Java案例\上传下载+数据库 - 文件的上传下载](file:///D:\教程\Java案例\上传下载+数据库%20-%20文件的上传下载)