# 文件上传

步骤:

1. 提交方式必须为:post,enctype必须为multipart/form-data,默认的是application/x-www-form-urlencoded,不能用于文件上传
2. 表单中存在文件域元素

|  |
| --- |
| <form action="" method="post" enctype="multipart/form-data">  用户名:<input type="text" name="username">  文件:<input type="file" name="file1">  </form> |

注意:

1. method为post,enctype为multipart/form-data时(这两个条件同时满足),任何表单元素通过getParameter(“XXX”)都获取不到数据,只能通过getInputStream()获取或者用FileUpload组件获取数据
2. 当表单元素类型为file时,enctype是application/x-www-form-urlencoded时,通过

getParameter(“XXX”)都获取到的数据是文件名称

jsp文件:

|  |
| --- |
| form method="post" action="${pageContext.request.contextPath}/hello" enctype="multipart/form-data">  <input type="text" name="username" value="众人人哈哈哈"> <br>  <input type="file" name="fil1">  <input type="submit" value="post提交"><br>  </form> |

servlet文件:

|  |
| --- |
| request.setCharacterEncoding("utf-8");  ServletInputStream inputStream = request.getInputStream();  BufferedReader reader = new BufferedReader(new InputStreamReader(inputStream));  String str = null;  while ((str=reader.readLine())!=null){  System.out.println(str);  } |

打印出的内容:

|  |
| --- |
| ------WebKitFormBoundaryoH1JgQvPXTPFDZYE  Content-Disposition: form-data; name="username"  浼椾汉浜哄搱鍝堝搱  ------WebKitFormBoundaryoH1JgQvPXTPFDZYE  Content-Disposition: form-data; name="fil1"; filename="鏂板缓鏂囨湰鏂囨。.txt"  Content-Type: text/plain  无聊的时候能做的事情:  1)一个人骑行  2)read  ------WebKitFormBoundaryoH1JgQvPXTPFDZYE-- |

如果要获取上传文件数据,需要对上面结果进行解析,如果自己解析,比较麻烦,

通常对于文件上传,用别人写好的组件,组件叫FileUpload组件(文件上传组件)

# FileUpload组件

FileUpload组件,也叫文件上传组件,是apache提供的开源项目

使用这个组件需要两个jar包:

1. commons-fileupload.jar
2. commons-io.jar

form:称为表单

form里面的Input标签:称为表单元素

**相关的api:**

1)FileItemFactory 文件上传工厂类(把每一个请求表单项封装为一个FileItem对象)

2) **DiskFileItemFactory**类中的方法setRepository(repository);上传缓存目录(没有设置也会有默认设置),这个类的构造函数还能设置缓存大小(如果上传的文件小于等于,该文件直接被上传到服务器,如果大于缓冲大小,则先被缓冲,最后上传到服务器),和缓存目录,

3) **ServletFileUpload**文件上传的核心类,

a.通过List<FileItem> parseRequest(request);可以获取所有的FileItem对象

b.静态函数boolean isMultipartContent(request); 判断,表单enctype属性的值是否为multipart/form-data

c.void setFileSizeMax(fileSizeMax); //设置单个文件大小的最大值

d.void setSizeMax(sizeMax); //设置所有文件大小的最大值

e.void upload.setHeaderEncoding("utf-8");//设置上传的文件名称编码,其实还可以用request. setCharacterEncoding("utf-8");也是可以设置文件名称的编码

4)**FileItem**封装了表单元素值

a.String getFieldName();获取表单元素的name的值

b.String getString();另一个重载的String getString(“utf-8”)用来设置编码的,如果是普通的表单元素,则是获取value的值,如果是文件表单元素,则返回的是文件的内容

c.String getContentType() 如果是普通表单元素,则为空,如果是文件,则返回的是文件的类型,如:

d.InputStream getInputStream();和getString()很像,如果是普通的表单元素,则返回value的值的输入流对象,如果是文件表单元素,则返回的是文件的内容的输入流

e.String getName();//如果是普通表单元素,返回null,如果是文件表单元素返回的是文件名称，如果是空的文件表单元素(**即是文件表单元素,但是没有选择文件,则name为””**)

f.void write(File file);写入到某个文件中,对任何表单元素有使用

g.item.delete();删除临时文件

代码:

|  |
| --- |
| //1.创建文件上传工厂类  FileItemFactory fac = new DiskFileItemFactory();  //2.创建文件上传核心类对象  ServletFileUpload upload = new ServletFileUpload(fac);  //判断,表单enctype属性的值是否为multipart/form-data  if(ServletFileUpload.isMultipartContent(request)){  try {  //3.把请求数据转化为FileItem对象的集合  List<FileItem> fileItems = upload.parseRequest(request);  //4.遍历得到每一个上床项  for(FileItem item : fileItems){  //判断:是普通表单项,还是文件上传表单项  if(item.isFormField()){  //普通表单  String fieldName = item.getFieldName();//表单元素的name的值  String content = item.getString();//普通表单元素的value值  InputStream inputStream = item.getInputStream(); //表单元素的value值(输入流)  System.out.printf("普通表单");  }else{  //文件上传表单  String fieldName = item.getFieldName();/表单元素的name的值  String contentType = item.getContentType();//文件类型  String name = item.getName();//文件名称  String str1 = item.getString();//上传的文件的内容,而不是表单元素的value的值  InputStream inputStream = item.getInputStream(); //文件内容对应的输入文件流  }  }  } catch (FileUploadException e) {  e.printStackTrace();  }  }else{  System.out.println("当前表单不是文件上传表单,不处理");  } |

# 完整个的文件上传

要求:

1. 设置文件总大小
2. 设置单个文件大小
3. 设置不能上传的文件类型
4. 解决重名问题:使用UUID算法
5. 把文件打散目录里即生成一级目录和二级目录
6. 上传文件的进度条
7. 获取普通表单元素的内容

上传文件的页面:

|  |
| --- |
| <%@ page contentType="text/html;charset=UTF-8" language="java" %>  <html>  <head>  <title>Title</title>  </head>  <body>  <font color="red">${requestScope.msg}</font>  <form action="${pageContext.request.contextPath}/upload" method="post" enctype="multipart/form-data">  请选择上传文件1:<input type="file" name="file1">描述1:<input type="text" name="info1"><br>  请选择上传文件2:<input type="file" name="file2">描述2:<input type="text" name="info2"><br>  <input type="submit" value="提交">  </form>  </body>  </html> |

Upload.java:

|  |
| --- |
| package servlet;  import MyException.FileTypeErrorException;  import org.apache.commons.fileupload.FileItem;  import org.apache.commons.fileupload.FileUploadBase;  import org.apache.commons.fileupload.ProgressListener;  import org.apache.commons.fileupload.disk.DiskFileItemFactory;  import org.apache.commons.fileupload.servlet.ServletFileUpload;  import javax.servlet.ServletException;  import javax.servlet.annotation.WebServlet;  import javax.servlet.http.HttpServlet;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  import java.io.File;  import java.io.IOException;  import java.util.List;  import java.util.UUID;  @WebServlet(name = "Upload")  public class Upload extends HttpServlet {  protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  this.doGet(request, response);  }  protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  DiskFileItemFactory fac = new DiskFileItemFactory(10\*1024,new File("F:/tmp"));  try {  ServletFileUpload upload = new ServletFileUpload(fac);  //防止文件名称乱码  upload.setHeaderEncoding("utf-8");  //设置单个文件上传大小  upload.setFileSizeMax(1024\*30);//30k  //设置总文件大小  upload.setSizeMax(1024\*56);//56k  //注册一个文件上传的监听器  upload.setProgressListener(new MyProgressListener());  //注意:fileItems里封装了所有的表单元素  List<FileItem> fileItems = upload.parseRequest(request);  if(fileItems != null){  for(FileItem item : fileItems) {  /\*\*  \* 注意:限制文件的类型,文件的mime-type  \* 需求:不能上传图片  \*/  if(item.isFormField()){  //普通表单元素  if(item.getFieldName().equals("info1")){//获取表单元素的name的值,即input的name属性的值  String info1 = item.getString("utf-8");  System.out.println("info1:" + info1);  }  if(item.getFieldName().equals("info2")){  String info2 = item.getString("utf-8");  System.out.println("info2:" + info2);  }  }else {  //文件表单元素  //得到文件的类型  String contentType = item.getContentType();  if (contentType.contains("image/")) {  //上传的文件是图片  throw new FileTypeErrorException("文件类型错误,请换一张!");  }  //保存文件,使用UUID算法来生成唯一的文件名  String uuid = UUID.randomUUID().toString();//uuid  String fileName = item.getName();//文件名  if (!fileName.equals("") && fileName != null) {  String newFileName = uuid + fileName.substring(fileName.lastIndexOf("."));  //新的目录结构  String baseDir = "F:/tmp1/";  String subDir = makeDirectory(newFileName);  String finalDir = baseDir + subDir;  //生成目录结构  File dir = new File(finalDir);  if (!dir.exists()) {  //构件目录结构  dir.mkdirs();  }  item.write(new File(finalDir, newFileName));  }  item.delete();//删除临时文件  }  }  }  }catch (FileTypeErrorException e){ //先捕获业务的异常  request.setAttribute("msg",e.getMessage());  request.getRequestDispatcher("/upload.jsp").forward(request,response);  }catch (FileUploadBase.SizeLimitExceededException e){//捕获超过总文件大小的异常  request.setAttribute("msg","超过总文件大小56k");  request.getRequestDispatcher("/upload.jsp").forward(request,response);  }catch (FileUploadBase.FileSizeLimitExceededException e){  request.setAttribute("msg","超过单个文件大小30k");  request.getRequestDispatcher("/upload.jsp").forward(request,response);  }  catch (Exception e){  e.printStackTrace();  }  }  private static String makeDirectory(String fileName){  //1.根据文件名称返回hashCode值  int code = fileName.hashCode();  //2.算出第一层目录的名称  //这个first值的范围在0~15之间  int first = code&0xF;  //3.算出第二层目录的名称  //0xF>>1的结果是7,即second值的范围是0~7  int second = code & (0xF>>1);  return first + "/" + second + "/";  }  }  /\*\*  \* 实现ProgressListener接口,显示文件上传进度  \*  \*/  class MyProgressListener implements ProgressListener{  /\*\*  \* @param pBytesRead 目前上传了多少字节  \* @param pContentLength 总长度  \* @param pItems 目前正在第几个文件  \*/  @Override  public void update(long pBytesRead, long pContentLength, int pItems) {  System.out.println("已经上传了" + pBytesRead + "字节,总字节数为" + pContentLength + "字节,目前正在上传第" + pItems + "个文件");  }  } |

# 下载文件

注意:不要像这样下载,没有逻辑的代码,不能进行很多操作,而是要用servlet代码进行下载

<a href="${pageContext.request.contextPath}/upload/1.jpg">下载文件</a>

InputStream in = this.getServletContext().getResourceAsStream("/upload/" + fileName);

fileName = URLEncoder.encode(fileName,"utf-8");//注意这一行的位置

response.setHeader("content-Disposition","attachment;filename\*=" + fileName);

不同的浏览器对于content-Disposition解析有差异

对于firefox来说: attachment;filename\*=

对于chrome来说:attachement;filename=

down.jsp

|  |
| --- |
| <%@ page contentType="text/html;charset=UTF-8" language="java" %>  <html>  <head>  <title>使用servlet进行下载</title>  </head>  <body>  <a href="${pageContext.request.contextPath}/download">下载文件</a>  </body>  </html> |

DownloadServlet.java:

|  |
| --- |
| package servlet;  import javax.servlet.ServletException;  import javax.servlet.ServletOutputStream;  import javax.servlet.annotation.WebServlet;  import javax.servlet.http.HttpServlet;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  import java.io.\*;  import java.net.URLEncoder;  @WebServlet(name = "DownServlet")  public class DownServlet extends HttpServlet {  protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  this.doGet(request, response);  }  protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  //1)读取服务器的文件  //读取当前web应用的根目录下的资源文件  String fileName = "美女.jpg";  //对中文进行URLEncoder加密  //设置一个响应头:content-Disposition告诉浏览器以下载的方式打开该资源  //不同的浏览器对content-Disposition有差别  InputStream in = this.getServletContext().getResourceAsStream("/upload/" + fileName);  //注意这里要进行URL编码,就像访问一个网站时的URL会被编码一样,只不过这是浏览器自动帮你编码了,但是这里需要手动编码  fileName = URLEncoder.encode(fileName,"utf-8");//注意这一行的位置  String userAgent = request.getHeader("user-agent");  String content = "";  if(userAgent.contains("Firefox")){  //firefox  content = "attachment;filename\*=" + fileName;  }else{  //chrome  content = "attachment;filename=" + fileName;  }  response.setHeader("content-Disposition",content);  //2)把文件发送给浏览器  ServletOutputStream out = response.getOutputStream();  //边读边写  byte[] buf = new byte[1024];  int len = 0;  while((len=in.read(buf))!=-1){  out.write(buf,0,len);  }  out.close();  in.close();  }  } |

# 文件上传和下载结合数据库

思路:

1)上传

文件上传->服务器(1.文件内容保存起来,文件相关信息存储到数据库中)

2)下载

数据库中读取文件信息->拿着文件路径到服务器找文件->使用servlet下载

三层接口开发顺序:

1. 设计数据库
2. 设计实体
3. 编写DAO
4. 编写service
5. 编写servlet + jsp

这个项目是:Demo2