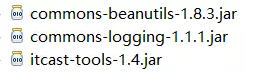
# 文件的上传与下载\_邮件

## CommomUtils

依赖 jar 包：



## 上传

### 项目（重要）

file\_upload\_and\_download

### 上传对表单的限制

* Method=”post”;
* Enctype=”mutipart/form-data”
* 表单中需要添加文件表单项：<input type=”file” name=… />;

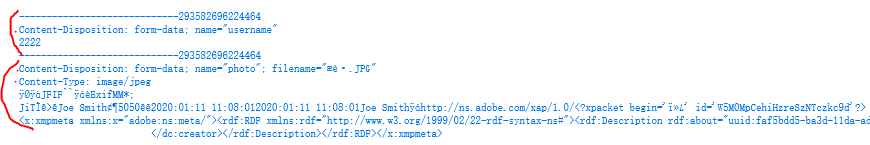
### 多部件表单的体

* 一个表单项一个部件；
* 一个部件中包含请求头和空行，以及请求体；
* 其中，普通表单项：
* 一个头：Content-Disposition,包含 name=”xxx”，即表单项的名称；
* 体就是表单项的值。
* 文件表单项：
* 两个头：

Content-Disposition,包含 name=”xxx”，即表单项的名称；filename=”xxx”,表示上传文件的名称；

Content-Type:他是上传文件的 MIME 类型，例如：image/pjpeg,表示上传的是图片，扩展名为 .jpg.

* 体就是上传文件的内容。
* 实例图片



### 上传对servlet 的限制

* Request.getParameter(“xxx”);这个方法在表单为Enctype=”mutipart/form-data”时，作废，不论表单中的type为何值，永远返回 null;
* 此时应该使用方法：

ServletInputStream request.getInputStream();

返回整个请求的体！

* 相关的 jar 包：

Commons-fileupload:

* Commons-fileupload.jar
* 前者依赖于：commons-io.jar

作用：

帮我们解析 request 中的上传数据，解析后的结果使一个表单项数据封装到一个 FileItem 对象中，我们只需要调用 FileItem 中的方法即可。

## 上传三步

### 相关类

* 工厂：DiskFileItemFactory;
* 解析器：ServletFileUpload;
* 表单项：FileItem

### 步骤

* 创建工厂：

DiskFileItemFactory factory = new DiskFileItemFactory();

* 创建解析器：

ServletFileUpload sfu = new ServletFileUpload(factory);

* 使用解析器来解析 request ,得到 FileItem 集合：

List<FileItem> fileItemList = sfu.parseRequest(request);

## FileItem

* Boolean isFormFiled()

是否为普通表单项，true：普通表单项，false:文件表单项。

* String getFiledName()

返回普通表单项的名称。

* String getString(String charset)

返回普通表单项的值。

* String getName()

返回上传文件的名称。

* Long getSize()

返回上传文件的字节数。

* InputStream getInputStream()

返回上传文件对应的输入流。

* Void write(File destFile)

把上传的文件内容保存到指定文件中。

## 上传的细节

### 文件必须保存到WEB-INF 中

目的是不让浏览器直接访问到。

设想：如果用户上传 index.jsp 将项目中的相关文件覆盖，并在 jsp 文件中执行恶意代码，后果不堪设想。

### 文件名称相关问题

* 有些浏览器上传的文件名是绝对路径，需要切割！c:\files\photo.jpg

String filename = param2.getName();

int index = filename.lastIndexOf("\\");

if(index != -1){

filename = filename.substring(index+1);

}

* 文件名乱码或者普通表单项乱码：

通过Request.setCharacterEncoding(“UTF-8”) 解决；

因为 ServletFileUpload 内部会调用 request.getChracterEncoding();

或者使用

servletFileUpload.setHeaderEncoding(“UTF-8”);//优先级更高

* 文件同名问题：

需要为每个文件添加名称前缀，并且要保证前缀不能重复。Uuid

### 目录打散

不能再一个目录下存放过多文件。

* 首字母打散：

使用文件的首字母作为文件的名称；

但是，中文不太实用。

* 时间打散

使用当前日期作为目录；

* 哈希打散

\*- 通过文件名称得到 int 值，调用 hashCode()；

\*- 将其 int 值转化为十六进制的0~9，A~F；

\*- 获取 16 进制的前两位用来生成目录，目录为两层，例如：

\*-- 1A2B3C4D5E6F.txt,保存结构为 1/a/文件名

### 上传文件的大小限制

* 单个文件大小限制：
* sfu.setFileSizeMax(100\*1024); //此方法的单位为 byte,即 限制单个文件大小为 100 k
* 上面的方法调用，必须在解析开始之前调用；
* 如果上传的文件大小超出限制，会在 parseRequest() 执行时抛出异常！
* 异常类型：

FileUploadBase.FileSizeLimitExceededException

* 整个请求所有文件大小限制：
* sfu.setSizeMax(1024\*1024); //表单中的所有文件 不超过 1M
* 方法的限制同上面的方法
* 异常类型：

FileUploadBase.SizeLimitExceededException

* 两个方法抛出的异常不同！

### 缓存大小与临时目录

* 缓存大小

超出多大，才向硬盘中保存！默认为 10 KB!

* 临时目录

向硬盘的什么位置保存？

* 更改缓存大小与临时目录

new DiskFileItemFactory(20\*1024,new File("D:\\upload\_tmp"));

\*- 参数1 ，缓存大小；

\*- 参数2 ，保存位置。

## 下载

### 原理

下载就是向客户端响应字节数据！

原来响应的都是 html 的字符数据；

而此时需要将文件变成字节数组，使用 response.getOutputStream() 来响应给浏览器。

### 下载的要求

* 两个头一个流
* Content-Type

说明传递给客户端的文件是什么 MIME 类型，例如 image/pjpeg;

\*- 通过文件名称调用ServletContext.getMimeType()方法获取MIME 类型。

* Content-Disposition

默认值是 inline ，表示在浏览器窗口中打开！

Attachment;filename=xxx

\*- xxx 即为显示在下载框中的文件名称。

* 流：要下载的文件数据。

\*- 自己 new 一个输入流即可。

## 下载的细节

* 下载框中显示中文名称时，会出现乱码

通用方案：

Filename = new String(filename.getBytes(“GBK”),”ISO-8859-1”)

//filename 即为下载框中显示的文件名称。

或者针对浏览器分别对待：



## JavaMail

### 导包

Mail.jar

Actvition.jar

### 核心类

* Session

获取到它，表示已经与服务器连接上了，与Connection的作用相似！

* MimeMessage

表示一个邮件对象，可以调用其setForm()方法，设置发件人、收件人、主题、正文等。

* TransPort

只有一个功能，发邮件！