

文件上传

- 工具依赖
 - IO 和 FileUpload 工具包
 - IO 用于处理上传缓存, FileUplad 用来处理上传文件流

```
1  protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws
   IOException {
2      resp.setContentType("text/html; charset=utf-8");
3      req.setCharacterEncoding("UTF-8");
4      if (!FileUpload.isMultipartContent( req )) {
5          resp.getWriter().print("请使用上传表单域");
6          return ;
7      }
8      //创建文件存储工具类
9      DiskFileItemFactory diskFileItemFactory = new DiskFileItemFactory();
10     //创建上传工具类
11     FileUpload fileUpload = new FileUpload(diskFileItemFactory);
12     try {
13         //把request对象交给FileUplad工具类处理 得到的是一个列表 存储的是表单字段
14         List<FileItem> fileItems = fileUpload.parseRequest(req);
15         //我们循环每个表单字段
16         for ( FileItem fileItem : fileItems ) {
17             //我们只处理 文件上传
18             if ( !fileItem.isFormField() ) {
19                 //拼接保存文件的路径
20                 String realPath = req.getServletContext().getRealPath("") + "uploads";
21                 //通过File对象创建路径
22                 File file1 = new File(realPath);
23                 //如果路径不存在则创建
24                 if ( !file1.exists() ) {
25                     //创建文件夹
26                     file1.mkdirs();
27                 }
28                 //拼接文件名
29                 String path = realPath + File.separator + fileItem.getName();
30                 System.out.println(path);
31                 //撞见文件File实例
32                 File file = new File(path);
33                 //保存文件
```

```

34         fileItem.write(file);
35     }
36 }
37     resp.getWriter().print("文件上传成功!!");
38 } catch (FileUploadException e) {
39     e.printStackTrace();
40 } catch (Exception e) {
41     e.printStackTrace();
42 }
43
44 }

```

文件下载

- 文件下载比较简单
- 主要是使用 `IOUtils.copy( fileInputStream, outputStream );` 对本地文件和 `resp` 数据库合并操作
- 需要注意的是 `header` 不能有中文字符, 需要单独处理, 否则浏览器不识别文件名

```

1  protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws
   IOException {
2      /**
3       * 正常的情况下 , 下载是不走Java逻辑代码的
4       * 我们下载文件都交给 Tomcat, 文件是静态资源, Http 服务器 (Tomcat, Nginx ) 都能处理
5       * 我们不用管 文件下载的事情
6       * 特俗情况, 付费下载, 统计文件下载数量, 管控下载流量
7       */
8
9      /**
10     * 一般情况下, 管控文件下载需要 Token !
11     * 我们改需要 文件名
12     * 我们使用 Java 管控文件下载 不一定非要 使用Java IO去处理文件
13     * 301 跳转
14     */
15
16     /**
17     * 一般情况下 我们需要把文件重命名, 或者加随即时间戳
18     * 总之呢 尽可能不让用户猜到文件路径, 方案有很多
19     * 用户使用的是一个映射文件名 "abc.pdf" => "202030517-abc.pdf"
20     * 用户可能上传 攻击脚本
21     */

```

```

22
23     req.setCharacterEncoding("UTF-8");
24     String fileName = req.getParameter("filename");
25
26     if ( fileName == null || fileName.equals("") ) {
27         resp.setContentType("text/html; charset=utf-8");
28         resp.getWriter().print("请携带文件名访问");
29         return ;
30     }
31     //取存储文件的文件夹路径
32     String filePath = req.getServletContext().getRealPath("") + "uploads";
33     //取拼接文件名
34     String file = filePath + File.separator + fileName;
35     //判断文件是否存在
36     File file1 = new File(file);
37     if ( !file1.exists() ) {
38         resp.setContentType("text/html; charset=utf-8");
39         resp.getWriter().print("<h1>文件不存在</h1>");
40         return ;
41     }
42     //文件转文件流
43     FileInputStream fileInputStream = new FileInputStream(file1);
44     //取相应流
45     ServletOutputStream outputStream = resp.getOutputStream();
46     //向浏览器声明下载
47     resp.addHeader("content-type", "application/octet-stream");
48     //声明文件名
49     String downloadFileName = URLEncoder.encode(fileName, "UTF-8").replaceAll("\\\\+",
50 "%20");
51     //header 使用 filename*=UTF-8'' 协议告诉浏览器后台文件编码是 UTF-8 可以解决中文文件名乱
52 码问题
53     resp.addHeader("content-disposition", "attachment; filename*=UTF-8''" +
54 downloadFileName);
55     //将文件的文件流和resp相应流合并输出
56     IOUtils.copy( fileInputStream, outputStream );
57 }

```

项目Jar包依赖

```
1 <dependency>
2   <groupId>javax.servlet</groupId>
3   <artifactId>javax.servlet-api</artifactId>
4   <version>4.0.1</version>
5 </dependency>
6 <dependency>
7   <groupId>commons-io</groupId>
8   <artifactId>commons-io</artifactId>
9   <version>2.11.0</version>
10 </dependency>
11 <dependency>
12   <groupId>commons-fileupload</groupId>
13   <artifactId>commons-fileupload</artifactId>
14   <version>1.4</version>
15 </dependency>
```