文件上传类库 FileUpload 使用文档

1. 准备文件

1. fileupload.css
2. fileupload.js
3. jquery.1.11.0.js
4. FileUploader.jar

2. Web 客户端配置

1. 引入资源文件

```
<link rel="stylesheet" type="text/css" href="fileupload.css">
<script type="text/javascript" src="jquery.1.11.0.js"></script>
<script src="fileupload.js" charset="gb2312"></script>
```

2. HTML 结构

1. 文件表单

```
<!-- fileupload 的 HTML 模板 -->
<!-- 建议把文本字段放在前面，因为为本参数大多数决定文件的存放位置 -->
<form action="uploadFile.up" id="myform" method="post"
       ENCTYPE="multipart/form-data" target="hidden_frame">
    <span class="fade_upload_btn">
    <input type="hidden" name="textfield1" value="">
    <input type="hidden" name="textfield2 value="">
    <input type="file" name="file"  id="upfiles"
         class="_fileupload_fileinput"  multiple="multiple">
    <a href="javascript:;" class="upload_link">上传</a>
    </span>
    <iframe name="hidden_frame" id="hidden_frame" tyle="display:none"
        src=""></iframe>
</form>
```

2. 进度显示容器

```
<ul id="uplist" class="_fileupload_uploadlist">
  <!--  该部分为 js 运行时动态添加
  <p lass="_fileupload_filenames">apache-maven-3.3.3-bin.zip</p>
  <div lass="_fileupload_progressWrapper">
    <div class="progressValue"></div>
```

```
  </div>
  </li>
  <li class="_fileupload_status">
    <p>上传成功!</p>
  </li>
  -->
</ul>
```

3. 注册上传事件

```
//文件按钮
var fileinput = document.getElementById("upfiles");
//文件上传进度列表容器
var progress = document.getElementById("uplist");
//注册事件
fileupload(fileinput, progress, "uploader", stener","sab");
```

3. 服务器端配置

1. 在你的 web 工程下引入 FileUpload 的 jar 包 *FileUploader.jar*
2. 创建文件上传类 *FileUploadDemo* 使之继承

   com.hty.util.fileuploader.FileUpload
3. 在 web.xml 文件下配置如下三个 Servlet：

```
<!-- 文件上传地址 -->
<servlet>
  <servlet-name>uploader</servlet-name>
  <servlet-class>com.demo.servlet.FileUploadDemo</servlet-class>
</servlet>
<!-- 上传状态获取地址 -->
<servlet>
<servlet-name>uploaderlistener</servlet-name>
<servlet-
class>com.hty.util.fileuploader.FetchUploadProgress</servlet-class>
</servlet>
<!-- 创建文件上传会话地址 -->
<servlet>
<servlet-name>sab</servlet-name>
<servlet-
class>com.hty.util.fileuploader.SessionUploadAttributeBuilder</servl
et-class>
</servlet>
```

```
<servlet-mapping>
<servlet-name>sab</servlet-name>
<url-pattern>/sab</url-pattern>
</servlet-mapping>
<servlet-mapping>
<servlet-name>uploaderlistener</servlet-name>
<url-pattern>/uploaderlistener</url-pattern>
</servlet-mapping>
<servlet-mapping>
<servlet-name>uploader</servlet-name>
<url-pattern>/uploader</url-pattern>
</servlet-mapping>
```

★ 特别提醒

如果你使用的是 Struts，请在 Struts 的拦截器之前加上自己的拦截器，这个拦截器可以这样写：

```
@Override
public void doFilter(ServletRequest request, ServletResponse response,
        FilterChain chain) throws IOException, ServletException {
    HttpServletRequest req = (HttpServletRequest) request;
    //拦截文件上传类特定的 URI,使之跳过 Struts 的拦截
    if("/uploader".contains(req.getServletPath())){
        chain.doFilter(request, response);
    }else
        super.doFilter(request, response, chain);
}
```

类 *FileUpload* 参数说明：

参数名	类型	说明
buffersize	int	缓冲区大小
updatePerLength	int	更新一次上传状态读取的字节数
maxUploadSize	int	最大上传文件大小
cpulimit	int	CPU 限频
file	File	当前上传的文件（可选，可自行定义
form_id	String	当前上传表单标识符
regfile	List<File>	所有上传文件列表
paraMap	Map<String, String>	文本参数集合
ops	OutputStream	当前文件输出流
uploadStatusList	Map<String, UploadStatus>	上传会话集合

status	com.hty.util.fileuploader.UploadStatus	当前上传状态载体
request	HttpServletRequest	Http 请求
response	HttpServletResponse	Http 响应

类 *FileUpload* 方法说明：

方法	返回值	说明
initConfig()	void	初始化参数
onTextField(String, String)	void	读取到文本参数
onFileField(String)	void	读取到文件
onFileEnd()	void	结束读取一个文件
writeFile(byte[], int, int)	void	写当前文件字节流
onRequestInputStreamInterrupt()	void	读取中断
onFileSizeExceed()	Boolean	文件超过设定大小
onUploadFinish()	void	上传完成
doPost(HttpServletRequest, HttpServletResponse)	void	--

何天意 *1129353184@qq.com*

2015-09-19