# Uploadify介绍

## Uploadify简介

来自国外的一款优秀jQuery插件，主要功能是批量上传文件，此插件在项目中已被广泛使用。特点是使用ajax实现无刷新的文件上传。

## Uploadify工作原理

通过Ajax发送文件上传请求，服务端接收到文件，并做文件的相应处理，并将文件保存的处理结果返回给客户端浏览器，通过回调函数返回服务端信息，并在页面不刷新的前提下执行其他操作。

注意：当Uploadify调用时，文件已经发生了上传工作，如果在上传完成后想要取消上传或替换文件，需要编写额外的方法删除后台服务端文件。

## 一个简单的Uploadify例子

官方demo地址：<http://www.uploadify.com/demos/>

客户端页面代码：

定义一个页面input的file标签

<input type="file" name="file\_upload" id="file\_upload\_1" />

$("#file\_upload\_1").uploadify({

height : 30,

swf : '/uploadify/uploadify.swf',

uploader : '/uploadify/uploadify.php',

width : 120

});

后台配套代码：

**SpringMVC**使用MultipartHttpServletRequest类来获取带有文件流的Http请求，获取临时文件MultipartFile类并通过transferTo函数将文件保存到指定的位置。

MultipartHttpServletRequest multipartRequest = (MultipartHttpServletRequest) request;

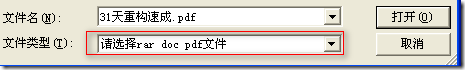
**Struts2**使用MultiPartRequestWrapper类来获取带有文件流的Http请求，并通过multiWrapper.getFiles(inputName);来获取临时文件

MultiPartRequestWrapper multiWrapper = (MultiPartRequestWrapper) request;

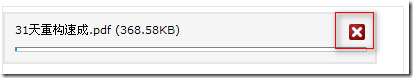
## Uploadify参数配置

Uploadify作为一个jquery插件，需要进行参数配置。

**uploader** ： uploadify.swf 文件的相对路径，该swf文件是一个带有文字BROWSE的按钮，点击后淡出打开文件对话框，默认值：uploadify.swf。   
**script** ： 后台处理程序的相对路径。默认值：uploadify.php   
**checkScript** ：用来判断上传选择的文件在服务器是否存在的后台处理程序的相对路径   
**fileDataName** ：设置一个名字，在服务器处理程序中根据该名字来取上传文件的数据。默认为Filedata   
**method** ： 提交方式Post 或Get 默认为Post   
**scriptAccess ：**flash脚本文件的访问模式，如果在本地测试设置为always，默认值：sameDomain   
**folder ：** 上传文件存放的目录 。   
**queueID ：** 文件队列的ID，该ID与存放文件队列的div的ID一致。   
**queueSizeLimit ：** 当允许多文件生成时，设置选择文件的个数，默认值：999 。   
**multi ：** 设置为true时可以上传多个文件。   
**auto ：** 设置为true当选择文件后就直接上传了，为false需要点击上传按钮才上传 。   
**fileDesc ：** 这个属性值必须设置fileExt属性后才有效，用来设置选择文件对话框中的提示文本，如设置fileDesc为“请选择rar doc pdf文件”，打开文件选择框效果如下图：

[](http://images.cnblogs.com/cnblogs_com/oec2003/WindowsLiveWriter/JQueryUploadify_13EFC/2010-01-05_220920_2.png)

**fileExt ：** 设置可以选择的文件的类型，格式如：'\*.doc;\*.pdf;\*.rar' 。   
**sizeLimit ：** 上传文件的大小限制 。   
**simUploadLimit ：** 允许同时上传的个数 默认值：1 。   
**buttonText ：** 浏览按钮的文本，默认值：BROWSE 。   
**buttonImg ：** 浏览按钮的图片的路径 。   
**hideButton ：** 设置为true则隐藏浏览按钮的图片 。   
**rollover ：** 值为true和false，设置为true时当鼠标移到浏览按钮上时有反转效果。   
**width ：** 设置浏览按钮的宽度 ，默认值：110。   
**height ：** 设置浏览按钮的高度 ，默认值：30。   
**wmode ：** 设置该项为transparent 可以使浏览按钮的flash背景文件透明，并且flash文件会被置为页面的最高层。 默认值：opaque 。   
**cancelImg** ：选择文件到文件队列中后的每一个文件上的关闭按钮图标，如下图：

[](http://images.cnblogs.com/cnblogs_com/oec2003/WindowsLiveWriter/JQueryUploadify_13EFC/2010-01-05_220626_2.png)

## Uploadify回调函数

**onSelectOnce** ：在单文件或多文件上传时，选择文件时触发。该函数有两个参数event，data，data对象有以下几个属性：

* fileCount：选择文件的总数。
* filesSelected：同时选择文件的个数，如果一次选择了3个文件该属性值为3。
* filesReplaced：如果文件队列中已经存在A和B两个文件，再次选择文件时又选择了A和B，该属性值为2。
* allBytesTotal：所有选择的文件的总大小。

**onCancel :** 当点击文件队列中文件的关闭按钮或点击取消上传时触发。该函数有event、queueId、fileObj、data四个参数，前三个参数同**onSelect** 中的三个参数，data对象有两个属性fileCount和allBytesTotal。

* fileCount：取消一个文件后，文件队列中剩余文件的个数。
* allBytesTotal：取消一个文件后，文件队列中剩余文件的大小。

**onClearQueue** **：**当调用函数fileUploadClearQueue时触发。有event和data两个参数，同**onCancel** 中的两个对应参数。

**onQueueFull ：**当设置了queueSizeLimit并且选择的文件个数超出了queueSizeLimit的值时触发。该函数有两个参数event和queueSizeLimit。

**onError ：**当上传过程中发生错误时触发。该函数有event、queueId、fileObj、errorObj四个参数，其中前三个参数同上，errorObj对象有type和info两个属性。

* type：错误的类型，有三种‘HTTP’, ‘IO’, or ‘Security’
* info：错误的描述

**onOpen ：**点击上传时触发，如果auto设置为true则是选择文件时触发，如果有多个文件上传则遍历整个文件队列。该函数有event、queueId、fileObj三个参数，参数的解释同上。

**onProgress** ：点击上传时触发，如果auto设置为true则是选择文件时触发，如果有多个文件上传则遍历整个文件队列，在onOpen之后触发。该函数有event、queueId、fileObj、data四个参数，前三个参数的解释同上。data对象有四个属性percentage、bytesLoaded、allBytesLoaded、speed：

* percentage：当前完成的百分比
* bytesLoaded：当前上传的大小
* allBytesLoaded：文件队列中已经上传完的大小
* speed：上传速率 kb/s

**onComplete**：文件上传完成后触发。该函数有四个参数event、queueId、fileObj、response、data五个参数，前三个参数同上。response为后台处理程序返回的值，在上面的例子中为1或0，data有两个属性fileCount和speed

* fileCount：剩余没有上传完成的文件的个数。
* speed：文件上传的平均速率 kb/s

注：fileObj对象和上面讲到的有些不太一样，onComplete 的fileObj对象有个filePath属性可以取出上传文件的路径。

**onAllComplete：**文件队列中所有的文件上传完成后触发。该函数有event和data两个参数，data有四个属性，分别为：

* filesUploaded :上传的所有文件个数。
* errors ：出现错误的个数。
* allBytesLoaded ：所有上传文件的总大小。
* speed ：平均上传速率 kb/s

相关函数介绍

在上面的例子中已经用了uploadifyUpload和uploadifyClearQueue两个函数，除此之外还有几个函数：

**uploadifySettings**：可以动态修改上面介绍的那些key值，如下面代码

$('#uploadify').uploadifySettings('folder','JS');

如果上传按钮的事件写成下面这样，文件将会上传到uploadifySettings定义的目录中

<a href="javascript:$('#uploadify').uploadifySettings('folder','JS');

$('#uploadify').uploadifyUpload()">上传</a>

**uploadifyCancel：**该函数接受一个queueID作为参数，可以取消文件队列中指定queueID的文件。

$('#uploadify').uploadifyCancel(id);

## 特殊操作

Uploadify本身的默认设置和人们日常的使用习惯可能并不一致，而Uploadify的封装使得Uploadify的所有回调函数都是在默认的函数之后执行的。因此在某些特殊的场景需要用户编写额外的操作。

例如项目中有如下需求：多文件上传时，可以取消已经上传完成的文件，重复上传同名的文件，提示是否覆盖原来的文件，并且在确认后上传后一个文件，并将前一个已经上传的文件取消。

Uploadify插件本身不支持已上传完成文件的删除功能。其中上传的列表可以去除显示，但是文件队列中却仍然保存文件，并且不会主动去后台要求删除已上传的文件，此时需要手动完成删除工作。

设置OnSelect函数，文件选择时的回调函数：

onSelect: **function** (file) {  
 **var** settings = **this**.**settings**;  
 **var** localHtml = **'<a href="javascript:$(\'#'** + settings.**id** + **'\').uploadify(\'cancel\', \''** + file.id  
 + **'\');removeAttach(\''** + file.id + **'\');">X</a>'**;  
 *$*(**"#"** + file.id +**" .cancel"**).html(localHtml);  
 *//将队列对象赋值给queueFile对象* ***queueFiles*** = **this**.**queueData**.**files**;  
 *//清理无效的队列附件  
 clearAttach*();  
}

将原来的文件列表替换为自己定义的文件列表，this.queueData.files可以获取到当前控件队列中的文件列表，并将值赋给全局变量queueFiles，通过特定的清除方法清除队列中无效的或重复的文件，并可以同时设置ajax请求到服务端去删除已经上传完成的文件。

## Uploadify使用过程中存在的问题

### 1、Firefox无法获取用户信息问题：

由于uploadify在firefox上使用flash发送上传请求无法传递会话ID，导致服务器无法验证当前请求所在会话而阻止该请求，必须在uploader属性的url后面加上jsessionid

例如： 'uploader' : '${base}/admin/organize/importOrgnize;jsessionid=${SESSION\_ID}'

同时针对这个问题，Struts如果在拦截器中设置的当前用户验证，将不被通过，请针对上传文件的Action合理设置拦截器

### 2、Chrome浏览器下由于浏览器缓存会出现页面崩溃问题

在Chrome浏览器下由于浏览器缓存会出现页面崩溃问题，需要让浏览器每次都获取新的uploadify的js文件。代码如下：

//采用jQuery动态加载脚本，默认每次都加载脚本不会使用缓存

jQuery.getScript("${base}/js/jqueryUploadify/jquery.uploadify.min.js",initUploadify);

### 3、无效的404请求

Uploadify会自动发起向请求地址的根目录发起无效的请求，并且返回404请求异常，比如页面是/tickets/create，那么请求的就是/tickets/

原来问题的症结在于SWFUpload，这个是Uploadify调用供支持Flash上传用的，但是其没有对button\_image\_url进行判断，导致如果button\_image\_url为空的话就默认请求了当前根路径

修改的方法：

将下面代码：

|  |
| --- |
| this.settings.button\_image\_url = SWFUpload.completeURL(this.settings.button\_image\_url); |

修改成如下：

|  |
| --- |
| if (this.settings.button\_image\_url != "") {  this.settings.button\_image\_url = SWFUpload.completeURL(this.settings.button\_image\_url);  } |
|  |