

# SWFUpload V2.2.0 说明文档

更新时间：2008-11-07

## TOC

- 1. [SWFUpload](#)
- 2. [SWFUpload 2](#)
- 3. [概述](#)
- 4. [入门](#)
- 5. [SWFUpload JavaScript 对象](#)
  - 1. [构造器](#)
  - 2. [全局变量和常量](#)
    - 1. [instances](#)
    - 2. [movieCount](#)
    - 3. [QUEUE\\_ERROR](#)
    - 4. [UPLOAD\\_ERROR](#)
    - 5. [FILE\\_STATUS](#)
    - 6. [Default Event Handlers](#)
    - 7. [BUTTON\\_ACTION\(added in v2.2.0\)](#)
    - 8. [BUTTON\\_CURSOR\(added in v2.2.0\)](#)
    - 9. [BUTTON\\_WINDOW\\_MODE\(added in v2.2.0\)](#)
  - 3. [属性](#)
    - 1. [customSettings](#)
    - 2. [movieName](#)
  - 4. [方法](#)
    - 1. [addSetting](#) (不推荐使用)
    - 2. [getSetting](#) (不推荐使用)
    - 3. [retrieveSetting](#) (v2.1.0中已删除)
    - 4. [destroy](#) (v2.1.0中新增)
    - 5. [displayDebugInfo](#)
    - 6. [selectFile](#) (不推荐使用)
    - 7. [selectFiles](#) (不推荐使用)
    - 8. [startUpload](#)
    - 9. [cancelUpload](#)
    - 10. [stopUpload](#)
    - 11. [getStats](#)
    - 12. [setStats](#)
    - 13. [getFile](#)
    - 14. [addPostParam](#)
    - 15. [removePostParam](#)
    - 16. [addFileParam](#)
    - 17. [removeFileParam](#)
    - 18. [setUploadURL](#)
    - 19. [setPostParams](#)
    - 20. [setFileTypes](#)
    - 21. [setFileSizeLimit](#)
    - 22. [setFileUploadLimit](#)
    - 23. [setFileQueueLimit](#)
    - 24. [setFilePostName](#)
    - 25. [setUseQueryString](#)
    - 26. [setDebugEnabled](#)
    - 27. [setButtonImageURL \(added in v2.2.0\)](#)
    - 28. [setButtonDimensions \(added in v2.2.0\)](#)
    - 29. [setButtonText \(added in v2.2.0\)](#)
    - 30. [setButtonTextStyle \(added in v2.2.0\)](#)
    - 31. [setButtonTextPadding \(added in v2.2.0\)](#)
    - 32. [setButtonDisabled \(added in v2.2.0\)](#)
    - 33. [setButtonAction \(added in v2.2.0\)](#)
    - 34. [setButtonCursor \(added in v2.2.0\)](#)
- 5. [事件](#)
  - 1. [flashReady](#)
  - 2. [swfUploadLoaded](#)
  - 3. [fileDialogStart](#)
  - 4. [fileQueued](#)
  - 5. [fileQueueError](#)
  - 6. [fileDialogComplete](#)
  - 7. [uploadStart](#)
  - 8. [uploadProgress](#)
  - 9. [uploadError](#)
  - 10. [uploadSuccess](#)
  - 11. [uploadComplete](#)
  - 12. [debug](#)
- 6. [SWFUpload 功能对象](#)
  - 1. [Settings Object](#)
  - 2. [Settings Description](#)
  - 3. [File Object](#)
  - 4. [Stats Object](#)
- 6. [SWFUpload 插件（未译）](#)
- 7. [存在的问题（未译）](#)

# SWFUpload

[SWFUpload](#)最初是由[Vinterwebb.se](#)开发的一个客户端的上传工具. 它结合了FLASH和JavaScript的功能，以提供一种超越了传统的浏览器中<input type="file" />标签提供的文件上传功能。

SWFUpload提供的主要功能:

- 在文件选择对话框中能够进行文件多选
- 页面无刷新的上传
- 提供上传进度的事件回调，实时显示上传进度
- 良好的浏览器兼容性
- 采用了命名空间以兼容其它JS的库（例如 jQuery, Prototype, 等等）
- ~~对FLASH 8和FLASH 9播放器的支持~~
- 对FLASH 9和FLASH 10播放器的支持(V2.2.0版本放弃了对Flash 8的支持)

SWFUpload背后的设计思想和其它基于Flash的上传工具是不同的。它将浏览器的中UI交给开发人员来控制。开发人员能够利用XHTML,CSS,Javascript来定制符合他们网站风格的UI上传元素。然后使用它提供的一组简单的JS事件来更新上传状态，开发人员能够利用这些事件来及时更新页面中的上传进度UI。

不幸的是Flash Player 10 更严格的安全机制迫使我们不得不将一个Flash Button放入Flash影片中。SWFUpload提供API供开发者通过图片、文字、CSS的方式来来自定制更灵活的UI显示。

## SWFUpload v2

SWFUpload v2包含了新的高级功能，改善了稳定性，解决了FlashPlayer中的一些bug，并且提供一套有用的插件。新的功能包括:

- 兼容了Flash Player 10的安全限制问题
- 在文件上传的同时能够发送额外的POST数据
- 针对每一个文件上传发送POST/GET数据
- 更直观的事件回调
- 动态修改实例设置
- 接收服务端返回的数据
- 非取消形式的停止文件上传
- 自定义上传的顺序
- 支持单文件、多文件的文件的选择
- 文件入队数量，文件上传数量和文件大小的限制
- 更合理地处理0字节的文件
- 针对每个文件在上传前都提供一个最后确认的时间回调
- 解决了v1.0.2版本中未描述到的关于Flash的bug
- 解决的v1.0.2中的bug:
  - 在IE中，刷新的时候FLASH无法加载（详细可见我之前的[debug过程](#)）
  - 在FireFox中，如果窗口的滚动条没有回滚到顶部，那么Flash无法加载
  - Race-conditions when files are cached
- 兼容ASP.Net Forms

SWFUpload v2 延续了SWFUpload的设计目标，将UI分离以交给开发人员控制和后续扩展

## 概述

### 传统的HTML上传

标准的HTML上传表单为用户提供一个文本框和按钮来选择文件，选中的文件是随着form表单提交的。整个文件上传完成之后，下一个页面才会显示，并且不能对选择的文件做预设的文件检验，例如文件大小限制，文件类型限制。当文件上传时，用户获得的可用的反馈信息很少。

传统的HTML上传模式十分简单，线性的，几乎所有浏览器都支持它。

### SWFUpload

~~SWFUpload使用一个隐藏的Flash影片来控制文件的选择和上传。JavaScript用来激活文件选择对话框。此文件选择对话框是可以设置允许用户选择一个单独的文件或者是多个文件。~~

SWFUpload使用一个Flash影片来控制文件的选择和上传。此FLASH中包含一个用户自定义UI的按钮，点击该按钮能够激活Flash本身的高级文件上传对话框，它能够根据用户的设置来进行单文件或者是多文件的上传。选择的的文件类型也是可以被限制的，因此用户只能选择指定的适当的文件，例如\*.jpg;\*.gif。

**提醒:** Flash Player 10的安全机制更严格，类似打开文件上传的对话框的操作，一定需要用户交互才触发，如果用脚本触发，会报#2176的运行时错误。因此V2.2.0版本在SWF中添加了一个可定制的Button让用户交互来打开文件对话框。

当选定文件以后，每个文件都会被验证和处理。当Flash上传文件的时候，由开发人员预定义的Javascript事件会被定时触发以便来更新页面中的UI，同时还提供上传状态和错误信息。

选定的文件的上传和它所在页面、表单是独立的。每个文件都是单独上传的，这就保证了服务端脚本能够在一个时间点更容易地处理单个文件。虽然Flash提供了上传服务，但是页面并不会提交或者重新载入。相比于标准的HTML Form，SWFUpload的使用方式更像是AJAX程序，页面中的Form会和FLASH控制的文件

## 入门

SWFUpload并不是拖放式的上传控件，它需要JavaScript和DOM的知识。一些可用的演示展示了它能够完成什么事情以及它是如何完成这些常见的任务。

SWFUpload由4部分组成：

1. 初始化和设置(Javascript)
2. JavaScript 库: SWFUpload.js
3. ~~Flash控制元素: SWFUpload\_f8.swf 或者 SWFUpload\_f9.swf~~ SWFUpload.swf(V2.2.0版本放弃了对flash 8的支持)
4. 事件处理(Javascript)

使用SWFUpload遇到的多数问题是由不正确地设置或者定义了糟糕的处理事件引起的。

### 初始化和设置

SWFUpload必须在页面中初始化，一般可以在window.onload事件中完成此操作。它的构造函数需要一个Object类型的设置对象。 这个设置对象一般是一个直接定义的Object类型变量，直接传递给SWFUpload的构造函数。

初始化的SWFUpload对象的引用需要保留下来，因为当显示文件选择对话框和启动文件上传的时候需要这个实例的引用。

例如:用直接定义的Object类型变量设置初始化SWFUpload对象

```
var swfu;

window.onload = function () {
    swfu = new SWFUpload({
        upload_url : "http://www.swfupload.org/upload.php",
        flash_url : "http://www.swfupload.org/swfupload.swf",
        button_placeholder_id : "spanSWFUploadButton",
        file_size_limit : "20480"
    });
};
```

例如:用存储在变量中的设置对象初始化SWFUpload对象

```
var swfu;

window.onload = function () {
    var settings_object = {
        upload_url : "http://www.swfupload.org/upload.php",
        flash_url : "http://www.swfupload.org/swfupload.swf",
        button_placeholder_id : "spanSWFUploadButton",
        file_size_limit : "20480"
    };

    swfu = new SWFUpload(settings_object);
};
```

### JavaScript 库

该JavaScript库文件(swfupload.js)应该包含在需要上传功能的页面中。

当SWFUpload创建完成并能访问它的一系列功能时，开发人员可以来控制此实例。

例如: 添加SWFUpload.js到页面中

```
<script type="text/javascript" src="http://www.swfupload.org/swfupload.js"></script>
```

例如: 根据需要的设置来初始化SWFUpload，~~同时把它的selectFiles方法绑定到一个按钮的Click事件上。~~

```
var swfu = new SWFUpload({
    upload_url : "http://www.swfupload.org/upload.php",
    flash_url : "http://www.swfupload.org/swfupload.swf",
```

```
button_placeholder_id : "spanSWFUploadButton"
});

document.getElementById("BrowseButton").onclick = function () { swfu.selectFiles(); };
```

## Flash 控制元素

SWFUpload JavaScript库动态加载Flash控制元素（swfupload.swf）。~~该Flash控制元素有两个版本。swfupload\_f8.swf支持Flash Player8以及更高版本。它在提供了更好的兼容性同时牺牲了一些功能。swfupload\_f9.swf支持Flash Player9.0.28以及更高版本。它在提供了额外的功能同时牺牲了使用Flash Player8的用户。~~

Flash控制元素的文件地址在初始化的时候就应该在SWFUpload设置对象中定义。

Flash控制元素是一个很小的Flash影片，它提供了文件浏览、检验和上传功能。它在页面中展现给用户的是一个UI可自定制的按钮，但该Flash会在需要时候通过与Javascript通信来通知浏览器处理更新。

## 事件处理

开发人员必须定义一系列JavaScript函数来处理SWFUpload事件回调，当一些不同的重要事件发生的时候，这些函数会被触发。

通过处理SWFUpload的事件，开发人员能够提供关于上传进度、出错信息以及上传完成等的信息反馈。

例如: swfupload的处理事件和初始化

```
// uploadStart处理事件。该函数变量在设置对象中指定给了upload_start_handler属性。
var uploadStartEventHandler = function (file) {
    var continue_with_upload;

    if (file.name === "the sky is blue") {
        continue_with_upload = true;
    } else {
        continue_with_upload = false;
    }

    return continue_with_upload;
};

//uploadSuccess处理事件。 该函数变量在设置对象中指定给了upload_success_handler属性。
var uploadSuccessEventHandler = function (file, server_data) {

    alert("The file " + file.name + " has been delivered to the server.");
    alert("The server responded with " + server_data);
};

//创建SWFUpload实例，设置事件回调函数
var swfu = new SWFUpload({
    upload_url : "http://www.swfupload.org/upload.php",
    flash_url : "http://www.swfupload.org/swfupload.swf",
    file_size_limit : "20480",

    upload_start_handler : uploadStartEventHandler,
    upload_success_handler : uploadSuccessEventHandler
});
```

# SWFUpload JavaScript 对象

## 构造函数

**SWFUpload**(*settings object*)

返回:一个SWFUpload 实例

```
var swfupload_instance = new SWFUpload(settings_object);
```

## 全局变量和常量

SWFUpload定义了一些全局变量和常量，这对SWFUpload的高级应用程序和处理错误都是很有用的，它们都是只读的。

## SWFUpload.instances

SWFUpload.instances是一个存储了页面中所有SWFUpload实例引用的数组。Flash播放器依靠这个数组来调用正确的处理事件。该数组是由movieName属性来索引的关联数组。例如：SWFUpload.instances.SWFUpload\_0访问的是第一个实例引用。

注意: SWFUpload.instances不是一个真正的JavaScript数组，实际上它是一个对象（关联数组）。

## SWFUpload.movieCount

SWFUpoad.movieCount是一个全局变量，用于记录页面中的SWFUpload实例个数，同时确保给每一个Flash影片分配一个惟一的movieName。

## SWFUpload.QUEUE\_ERROR

SWFUpload.QUEUE\_ERROR是一个包含了Queue Error错误码的JS对象，一般用它来查看fileQueueError事件的中发送的错误码，以确定fileQueueError的具体类型。

```
SWFUpload.QUEUE_ERROR = {
    QUEUE_LIMIT_EXCEEDED           : -100,
    FILE_EXCEEDS_SIZE_LIMIT       : -110,
    ZERO_BYTE_FILE                : -120,
    INVALID_FILETYPE              : -130
};
```

## SWFUpload.UPLOAD\_ERROR

SWFUpload.UPLOAD\_ERROR是一个包含了Upload Error错误码的JS对象，一般用它来查看uploadError事件的中发送的错误码，以确定uploadError的具体类型。

```
SWFUpload.UPLOAD_ERROR = {
    HTTP_ERROR                     : -200,
    MISSING_UPLOAD_URL            : -210,
    IO_ERROR                      : -220,
    SECURITY_ERROR                 : -230,
    UPLOAD_LIMIT_EXCEEDED         : -240,
    UPLOAD_FAILED                 : -250,
    SPECIFIED_FILE_ID_NOT_FOUND   : -260,
    FILE_VALIDATION_FAILED        : -270,
    FILE_CANCELLED                : -280,
    UPLOAD_STOPPED                : -290
};
```

## SWFUpload.FILE\_STATUS

SWFUpload.FILE\_STATUS是一个包含File Status状态码的JS对象。它可以用来检测File对象中的file status属性，以确定文件的状态。

```
SWFUpload.FILE_STATUS = {
    QUEUED           : -1,
    IN_PROGRESS     : -2,
    ERROR            : -3,
    SUCCESS          : -4,
    CANCELLED        : -5
};
```

## 默认处理事件

SWFUpload库提供了一系列默认的处理事件。当开发人员没有自定义处理事件时，SWFUpload库将调用这些默认的处理事件。因此当自定义事件时，不要把这些默认的处理事件覆盖了。自定义事件是需要要在settings对象中被单独定义的。

## SWFUpload.BUTTON\_ACTION

SWFUpload.BUTTON\_ACTION是一个包含button的动作预设的JS对象。

```
SWFUpload.BUTTON_ACTION = {
    SELECT_FILE : -100,
    SELECT_FILES : -110,
```

START\_UPLOAD : -120

}

3

4

可以使用button\_action属性来设置Flash本身的文件上传对话框的行为。默认为SWFUpload.BUTTON\_ACTION.SELECT\_FILES，点击按钮将会打开多文件上传的对话框。如果设置为SWFUpload.BUTTON\_ACTION.SELECT\_FILE，则为单文件上传。如果设置为SWFUpload.BUTTON\_ACTION.START\_UPLOAD，则启动文件上传

## SWFUpload.CURSOR

SWFUpload.CURSOR是一个包含button的光标状态码的JS对象。

```
SWFUpload.CURSOR = {  
    ARROW : -1,  
    HAND : -2  
}
```

3

4

可以使用button\_cursor属性来设置鼠标划过button时的光标状态。默认为SWFUpload.CURSOR.ARROW，如果设置为SWFUpload.CURSOR.HAND，则为手形

## SWFUpload.WINDOW\_MODE

SWFUpload.WINDOW\_MODE是一个包含了该SWF插入到页面中的wmode属性的JS对象.可以通过设置button\_window\_mode属性来告诉浏览器具体以哪种模式显示此SWF。

```
SWFUpload.WINDOW_MODE = {  
    WINDOW : "window",  
    TRANSPARENT : "transparent",  
    OPAQUE : "opaque" };
```

3

4

SWFUpload.WINDOW\_MODE.WINDOW是默认的模式. 该SWF将位于页面元素的最高层级。

SWFUpload.WINDOW\_MODE.OPAQUE 该SWF可以被页面类的其他元素通过层级的设置来覆盖它。

SWFUpload.WINDOW\_MODE.TRANSPARENT 该SWF的背景是透明的，可以透过它看到背后的页面元素。

## 属性

下面这个列表是相关属性的具体描述。使用其它属性或者对只读属性进行了写的操作都会造成SWFUpload出现问题。

## customSettings (可读/可写)

customSettings属性是一个空的JavaScript对象，它被用来存储跟SWFUpload实例相关联的数据。它的内容可以使用设置对象中的custom\_settings属性来初始化。

注意:一些插件使用customSettings对象来实现它们的内部控制。当重写整个customSettings对象的时候需要务必小心。

例如:

```
// 初始化包含自定义设置的SWFUpload对象  
var swfu = new SWFUpload({  
    custom_settings : {  
        custom_setting_1 : "custom_setting_value_1",  
        custom_setting_2 : "custom_setting_value_2",  
        custom_setting_n : "custom_setting_value_n",  
    }  
});  
  
swfu.customSettings.custom_setting_1 = "custom_setting_value_1"; // 更改一个存在的自定义设置  
swfu.customSettings.myNewCustomSetting = "new custom setting value"; // 添加一个新的自定义设置  
  
//用一个全新的对象重写customSettings  
swfu.customSettings = {  
    custom_setting_A : "custom_setting_value_A",  
    custom_setting_B : "custom_setting_value_B"  
};
```

3

4

## movieName (只读)

包含了该SWFUpload实例的惟一影片名字。该值被传递给Flash，用来完成Flash和JavaScript的通信。该值被用来索引实例在SWFUpload.instances数组中的位置，你无法更改此值。

## 方法

下面的方法用来操作SWFUpload。其中有些方法可以跟元素（例如，按钮）的点击事件绑定，其它的方法供SWFUpload内部处理事件中调用。

### **object addSetting(*setting\_name*, *value*, *default\_value*)**

不赞成使用 V2.1.0源码中注释：addSetting和getSetting已经不再被使用了，只是考虑到兼容V2版本，才继续保留在库中。

### **object getSetting(*setting\_name*)**

不赞成使用 V2.1.0源码中注释：addSetting和getSetting已经不再被使用了，只是考虑到兼容V2版本，才继续保留在库中。

### **object retrieveSetting(*setting\_value*, *default\_value*)**

v2.1.0中已被删除

### **bool destroy()**

v2.1.0中新增

用于将一个SWFUpload实例从页面中销毁。不但删除DOM中的Flash元素，同时还删除SWFUpload实例的相关引用。成功返回true，失败返回false。

这个方法还没有针对插件做兼容测试，可能会有不兼容问题。（尤其是SWFObject插件）

**提醒：**SWFUpload在v2.1.0中改写了Flash插入的代码，V2版中使用SWFObject插入Flash会造成IE的一个[刷新BUG](#)。

### **void displayDebugInfo()**

调用debug方法，在Debug输出框中显示SWFUpload实例的设置信息，如果设置中的debug属性是true，那么默认是在实例化完成以后自动调用此方法。

### **void selectFile()**

不赞成使用，不兼容Flash Player 10

弹出flash的文件选择对话框，只能选择单个文件。

**提醒：**Flash Player 10的安全机制更严格，类似打开文件上传的对话框的操作，一定需要用户交互才触发，如果用脚本触发，会报#2176的运行时错误。因此V2.2.0版本在SWF中添加了一个可定制的Button让用户交互来打开文件对话框。

### **void selectFiles()**

不赞成使用，不兼容Flash Player 10

弹出flash的文件选择对话框，可一次性选择多个文件。

**提醒：**Flash Player 10的安全机制更严格，类似打开文件上传的对话框的操作，一定需要用户交互才触发，如果用脚本触发，会报#2176的运行时错误。因此V2.2.0版本在SWF中添加了一个可定制的Button让用户交互来打开文件对话框。

### **void startUpload(*file\_id*)**

指定file\_id来启动该文件的上传，如果file\_id被忽略了，那么默认开始上传第一个文件。

### **void cancelUpload(*file\_id*, *trigger\_error\_event*)**

指定file\_id来退出文件的上传，从上传队列中删除该文件。

如果忽略file\_id，那么默认文件上传队列中的第一个文件将被退出上传。

如果取消的文件是正在上传，那么会触发uploadError事件。

如果将可选参数trigger\_error\_event设置为false,那么uploadError事件不会触发。

### **void stopUpload()**

如果当前有文件上传，那么停止上传，并且将文件还原到上传队列中。



停止了正在上传的文件，uploadError事件会被触发。如果此时没有正在上传文件，那么不会发生任何操作，不会触发任何事件。

**object getStats()**

获取当前状态的统计对象，具体见[Stats Object](#)。

**void setStats(stats\_object)**

Stats统计对象是可以被修改的。如果你希望在上传完毕之后修改上传成功或者上传失败的统计数目时，那么可以使用该方法。

**提醒：**可供修改的属性只有successful\_uploads，upload\_errors，upload\_cancelled，queue\_errors，并且值必须是Number类型。

**object getFile(file\_id [index])**

根据file\_id或者index来获取文件队列中的[文件对象](#)。file\_id是文件对象中的id属性，index是文件对象中的index属性。

传递Number类型的参数会被认定为index，那么返回的是文件队列（所有尝试入队文件，包括因没有通过文件大小、类型检测等而触发fileQueueError，没有成功加入文件上传队列的文件对象）数组中下标为index的文件对象。如果index不在队列数组范围内，那么返回null。

传递非Number类型的参数会被认定为file\_id，那么返回的是文件上传队列（通过文件检测，准备好进行上传的文件）数组中id为file\_id的文件对象。如果参数为空，或者没有此id的文件对象，那么返回文件等待队列中的第一个文件对象，如果第一个文件对象为空，那么返回null。

**void addPostParam(name, value)**

给设置中的post\_params对象添加值对，当文件上传的时候，这个值对会一同在POST中发送。

如果设置的时候，post\_params中以及存在该值，那么实际上会被覆盖。

**提醒：**SWFUpload存在[Cookie bug](#),你可以使用此方法避免。

**void removePostParam(name)**

从设置中的post\_params对象中删除name指定的属性，当文件上传的时候，删除的值对不会继续在POST中发送。

**bool addFileParam(file\_id, name, value)**

为指定file\_id的特定文件对象添加POST值对，如果添加的name属性已经存在，那么原值会被覆盖。

如果需要给所有文件对象添加POST值，那么可以使用设置中的post\_params属性。

**bool removeFileParam(file\_id, name)**

删除由addFileParam添加的POST值对。

如果POST设置中没有此属性，那么返回false。

**void setUploadURL(url)**

动态修改设置中的upload\_url属性。

**void setPostParams(param\_object)**

动态修改post\_params，以前的属性全部被覆盖。param\_object必须是一个JavaScript的基本对象，所有属性和值都必须是字符串类型。

**void setFileTypes(types, description)**

动态修改设置中的file\_types 和 file\_types\_description，两个参数都是必须的。

**void setFileSizeLimit(file\_size\_limit )**

动态修改设置中的file\_size\_limit，此修改针对之后的文件大小过滤有效。file\_size\_limit参数接收一个单位，有效的单位有B、KB、MB、GB，默认单位是KB。

例如: 2147483648 B, 2097152, 2097152KB, 2048 MB, 2 GB

**void setFileUploadLimit(file\_upload\_limit )**

动态修改设置中的file\_upload\_limit，特殊值0表示无限制。



**提醒：**这里限制的是一个SWFUpload实例控制上传成功的文件总数。

**void setFileQueueLimit(*file\_queue\_limit*)**

动态修改设置中的file\_queue\_limit，特殊值0表示无限制。

**提醒：**这里限制的是文件上传队列中（入队检测通过的文件会添加到上传队列等待上传）允许排队的文件总数。

**void setFilePostName(*file\_post\_name*)**

动态修改设置中的file\_post\_name，注意在Linux环境下，FlashPlayer是忽略此设置的。

**void setUseQueryString(*use\_query\_string*)**

动态修改设置中的use\_query\_string，设置为true的时候，SWFUpload以GET形式发送数据，如果为false，那么就以POST发送数据。

**void setDebugEnabled(*debug\_enabled*)**

启动/禁止 debug输出，debug\_enabled参数是一个布尔值。

**void setButtonImageURL(*url*)**

动态修改按钮的图片。url参数是相对于该swf文件或者是绝对地址的图片（或者是SWF）。所有FLASH支持的图片类型都可以使用（gif,jpg,png,或者是一个SWF）。

该按钮图片需要经过一定规则（CSS Sprite）的处理。按钮图片中需要包括按钮的4个状态，从上到下依次是normal, hover, down/click, disabled.（可以参照官方demo中的图片）

**void setButtonDimensions(*width, height*)**

动态修改SWF影片的尺寸以适应Button的图片大小。

**void setButtonText(*text*)**

动态设置Flash Button中显示的文字，支持HTML。HTML文本的样式可以通过CSS选择器并配合setButtonTextStyle方法来设置。关于Flash文本对HTML的支持详细可见 [Adobe's Flash documentation](#)。

**void setButtonTextStyle(*css\_style\_text*)**

配合setButtonText方法，可以通过CSS样式来动态设置Flash Button中的文字样式。关于Flash文本对CSS的支持详细可见[Adobe's Flash documentation](#)

**void setButtonTextPadding(*left, top*)**

Sets the top and left padding of the Flash button text. The values may be negative.

**void setButtonDisabled(*isDisabled*)**

When 'true' changes the Flash Button state to disabled and ignores any clicks.

**void setButtonAction(*buttonAction*)**

Sets the action taken when the Flash button is clicked. Valid action values are taken from the BUTTON\_ACTION constants.

**void setButtonCursor(*buttonCursor*)**

Sets the mouse cursor shown when hovering over the Flash button. Valid cursor values are taken from the BUTTON\_CURSOR constants.

## 事件

SWFUpload在操作过程中会触发一系列事件，开发者可以利用这些回调的处理事件来控制UI，控制操作或者报告错误。

所有的事件都是在SWFUpload实例的上下文中调用的，因此在这些回调的事件中使用this能够直接访问到该触发该事件的实例对象。

所有事件应该在实例初始化时setting参数中预设完成。

**flashReady()**

该事件函数是内部事件，因此不能被重写。当SWFupload实例化，加载的FLASH完成所有初始化操作时触发此事件。

**提醒：** 对应设置中的自定义事件swfupload\_loaded\_handler

## swfUploadLoaded()

V2.1.0版本中已经删除了此事件

## fileDialogStart()

此事件在selectFile或者selectFiles调用后，文件选择对话框显示之前触发。只能同时存在一个文件对话框。

**提醒：** 对应设置中的自定义事件file\_dialog\_start\_handler

## fileQueued(file object)

当文件选择对话框关闭消失时，如果选择的文件成功加入上传队列，那么针对每个成功加入的文件都会触发一次该事件（N个文件成功加入队列，就触发N次此事件）。

**提醒：** 对应设置中的自定义事件file\_queued\_handler

## fileQueueError(file object, error code, message)

当选择文件对话框关闭消失时，如果选择的文件加入到上传队列中失败，那么针对每个出错的文件都会触发一次该事件(此事件和fileQueued事件是二选一触发，文件添加到队列只有两种可能，成功和失败)。

文件添加队列出错的原因可能有：超过了上传大小限制，文件为零字节，超过文件队列数量限制，设置之外的无效文件类型。

具体的出错原因可由error code参数来获取，error code的类型可以查看[SWFUpload.QUEUE\\_ERROR](#)中的定义。

**提醒：** 对应设置中的自定义事件file\_queue\_error\_handler

**注意：** 如果选择入队的文件数量超出了设置中的数量限制，那么所有文件都不入队，此事件只触发一次。如果没有超出数目限制，那么会对每个文件进行文件类型和大小的检测，对于不通过的文件触发此事件，通过的文件成功入队。

## fileDialogComplete(number of files selected, number of files queued)

当选择文件对话框关闭，并且所有选择文件已经处理完成（加入上传队列成功或者失败）时，此事件被触发，number of files selected是选择的文件数目，number of files queued是此次选择的文件中成功加入队列的文件数目。

**提醒：** 对应设置中的自定义事件file\_dialog\_complete\_handler

**注意：** 如果你希望文件在选择以后自动上传，那么在这个事件中调用this.startUpload() 是一个不错的选择。如果需要更严格的判断，在调用上传之前，可以对入队文件的个数做一个判断，如果大于0，那么可以开始上传。

## uploadStart(file object)

在文件往服务端上传之前触发此事件，可以在这里完成上传前的最后验证以及其他你需要的操作，例如添加、修改、删除post数据等。

在完成最后的操作以后，如果函数返回false，那么这个上传不会被启动，并且触发uploadError事件（code为ERROR\_CODE\_FILE\_VALIDATION\_FAILED），如果返回true或者无返回，那么将正式启动上传。

**提醒：** 对应设置中的自定义事件upload\_start\_handler

## uploadProgress(file object, bytes complete, total bytes)

该事件由flash定时触发，提供三个参数分别访问上传文件对象、已上传的字节数，总共的字节数。因此可以在这个事件中来定时更新页面中的UI元素，以达到及时显示上传进度的效果。

**注意：** 在Linux下，Flash Player只在所有文件上传完毕以后才触发一次该事件，官方指出这是Linux Flash Player的一个bug，目前SWFupload库无法解决（我没有测试过）。

**提醒：** 对应设置中的自定义事件upload\_progress\_handler

## uploadError(file object, error code, message)

无论什么时候，只要上传被终止或者没有成功完成，那么该事件都将被触发。error code参数表示了当前错误的类型，更具体的错误类型可以参见[SWFUpload.UPLOAD\\_ERROR](#)中的定义。Message参数表示的是错误的描述。File参数表示的是上传失败的文件对象。

例如，我们请求一个服务端的一个不存在的文件处理页面，那么error code会是-200，message会是404。停止、退出、uploadStart返回false、HTTP错误、IO错误、文件上传数目超过限制等，都将触发该事件，Upload error will not fire for files that are cancelled but still waiting in the queue。（对于官方的这句话我还存在疑问，文件退出以后怎么还会保留在文件上传队列中保留呢？）

**提醒：**对应设置中的自定义事件upload\_error\_handler

**注意：**此时文件上传的周期还没有结束，不能在这里开始下一个文件的上传。

### uploadSuccess(file object , server data)

当文件上传的处理已经完成（这里的完成只是指向目标处理程序发送了Files信息，只管发，不管是否成功接收），并且服务端返回了200的HTTP状态时，触发此事件。

**提醒：**对应设置中的自定义事件upload\_success\_handler

**注意：**

- 1. server data是服务端处理程序返回的数据。
- 2. 此时文件上传的周期还没有结束，不能在这里开始下一个文件的上传。
- 3. 在window平台下，那么服务端的处理程序在处理完文件存储以后，必须返回一个非空值，否则此事件不会被触发，随后的uploadComplete事件也无法执行。

### uploadComplete(file object )

当上传队列中的一个文件完成了一个上传周期，无论是成功(uoloadSuccess触发)还是失败(uploadError触发)，此事件都会被触发，这也标志着一个文件的上传完成，可以进行下一个文件的上传了。

如果要进行多文件自动上传，那么在这个时候调用this.startUpload()来启动下一个文件的上传是不错的选择。

**提醒：**对应设置中的自定义事件upload\_complete\_handler

**注意：**当在进行多文件上传的时候，中途用cancelUpload取消了正在上传的文件，或者用stopUpload停止了正在上传的文件，那么在uploadComplete中就要很小心地使用this.startUpload()，因为在上述情况下，uploadError和uploadComplete会顺序执行，因此虽然停止了当前文件的上传，但会立即进行下一个文件的上传，你可能会觉得这很奇怪，但事实上程序并没有错。如果你希望终止整个队列的自动上传，那么你需要做额外的程序处理了。

### debug(message)

如果debug setting设置为true，那么页面底部会自动添加一个textArea，SWFUpload库和Flash都会调用此事件来在页面底部的输出框中添加debug信息供调试使用。

**提醒：**对应设置中的自定义事件debug\_handler

## SWFUpload功能对象

### Settings object

它是一个Object类型的变量，为SWFUpload的实例初始化提供配置。 其中的每一个配置属性都只能出现一次。 很多属性都是可选的，如果可选属性没有被配置，那么会使用SWFUpload库中默认指定的合适的值，具体可查看setting的详细介绍。

例 如：（setting可以配置的所有属性）

```
{
    upload_url : "http://www.swfupload.org/upload.php",
    flash_url  : "http://www.swfupload.org/swfupload.swf",

    file_post_name : "Filedata",
    post_params : {
        "post_param_name_1" : "post_param_value_1",
        "post_param_name_2" : "post_param_value_2",
        "post_param_name_n" : "post_param_value_n"
    },
    use_query_string : false,
    requeue_on_error : false,
    http_success : [201, 202],
    file_types : "*.jpg;*.gif",
    file_types_description: "Web Image Files",
    file_size_limit : "1024",
    file_upload_limit : 10,
    file_queue_limit : 2,

    debug : false,
```

```
prevent_swf_caching : false,
button_placeholder_id : "element_id",
button_image_url : "http://www.swfupload.org/button_sprite.png",
button_width : 61,
button_height : 22,
button_text : "<b>Click</b> <span class='redText'>here</span>",
button_text_style : ".redText { color: #FF0000; }",
button_text_left_padding : 3,
button_text_top_padding : 2,
button_action : SWFUpload.BUTTON_ACTION.SELECT_FILES,
button_disable : false,
button_cursor : SWFUpload.BUTTON_CURSOR.HAND,

button_window_mode : SWFUpload.WINDOW_MODE.TRANSPARENT,

swfupload_loaded_handler : swfupload_loaded_function,

file_dialog_start_handler : file_dialog_start_function,
file_queued_handler : file_queued_function,
file_queue_error_handler : file_queue_error_function,
file_dialog_complete_handler : file_dialog_complete_function,
upload_start_handler : upload_start_function,
upload_progress_handler : upload_progress_function,
upload_error_handler : upload_error_function,
upload_success_handler : upload_success_function,
upload_complete_handler : upload_complete_function,
debug_handler : debug_function,

custom_settings : {
    custom_setting_1 : "custom_setting_value_1",
    custom_setting_2 : "custom_setting_value_2",
    custom_setting_n : "custom_setting_value_n",
}
}
```

## Settings 描述

upload_url
默认值：空字符串
<p>upload_url设置接收的是一个绝对的或者相对于SWF文件的完整URL。推荐使用完整的绝对路径，以避免由浏览器和FlashPlayer修改了基准路径设置而造成的请求路径错误。</p> <p><b>注意：</b>这里需要考虑FlashPlayer的安全域模型。</p>
file_post_name
默认值：Filedata
<p>该参数设置了POST信息中上传文件的name值（类似传统Form中设置了&lt;input type="file" name="uploadImg"/&gt;的name属性）。</p> <p><b>注意：</b>在Linux下面此参数设置无效，接收的name总为Filedata，因此为了保证最大的兼容性，建议此参数使用默认值。</p>
post_params
默认值：空的Object对象
<p>post_params定义的是一个包含值对的object类型数据，每个文件上传的时候，其中的值对都会被一同发送到服务端。</p> <p><b>注意：</b>设置值对的时候，值只能是字符串或者数字。</p>
use_query_string
默认值：false
<p>该属性可选值为true和false，设置post_params是否以GET方式发送。如果为false，那么则以POST形式发送。</p>
requeue_on_error
默认值：false
<p>该属性可选值为true和false。如果设置为true，当文件对象发生uploadError时（除开fileQueue错误和FILE_CANCELLED错误），该文件对象会被重新插入到文件上传队列的前端，而不是被丢弃。如果需要，重新入队的文件可以被再次上传。如果要从上传队列中删除该文件对象，那么必须使用cancelUpload方法。</p>

跟上传失败关联的所有事件同样会被一一触发，因此将上传失败的文件重新入队可能会和Queue Plugin造成冲突（或者是自动上传整个文件队列的自定义代码）。如果代码中调用了startUpload方法自动进行下一个文件的上传，同时也没有采取任何措施让上传失败的文件退出上传队列，那么这个重新入队的上传失败的文件又会开始上传，然后又会失败，重新入队，重新上传...，进入了无止境的循环。

该设置是在v2.1.0中引入的。

http\_success

默认值： []

该数组可自定义触发success事件的HTTP状态值。200的状态值始终会触发success，并且只有200的状态会提供serverData。

当接受一个200以外的HTTP状态值时，服务端不需要返回内容。

file\_types

默认值： \*.\*

设置文件选择对话框的文件类型过滤规则，该属性接收的是以分号分隔的文件类型扩展名，例如 “\*.jpg;\*.gif”，则只允许用户在文件选择对话框中可见并可选jpg和gif类型的文件。默认接收所有类型的文件。

**提醒：**该设置只是针对客户端浏览器的过滤，对服务端的文件处理中的文件类型过滤没有任何限制，如果你需要做严格的文件过滤，那么服务端同样需要程序检测。

file\_types\_description

默认值： All Files

设置文件选择对话框中显示给用户的文件描述。

file\_size\_limit

默认值： 0

设置文件选择对话框的文件大小过滤规则，该属性可接收一个带单位的数值，可用的单位有B,KB,MB,GB。如果忽略了单位，那么默认使用KB。特殊值0表示文件大小无限制。

**提醒：**该设置只对客户端的浏览器有效，对服务端的文件处理没有任何限制，如果你需要做严格文件过滤，那么服务端同样需要程序处理。

file\_upload\_limit

默认值： 0

设置SWFUpload实例允许上传的最多文件数量，同时也是设置对象中file\_queue\_limit属性的上限。一旦用户已经上传成功或者添加文件到队列达到上最大数量，那么就不能继续添加文件了。特殊值0表示允许上传的数量无限制。只有上传成功（上传触发了uploadSuccess事件）的文件才会在上传数量限制中记数。使用setStats方法可以修改成功上传的文件数量。

**注意：**该值不能跨页面使用，当页面刷新以后该值也被重置。严格的文件上传数量限制应该由服务端来检测、管理。

file\_queue\_limit

默认值： 0

设置文件上传队列中等待文件的最大数量限制。当一个文件被成功上传，出错，或者被退出上传时，如果文件队列中文件数量还没有达到上限，那么可以继续添加新的文件入队，以顶替该文件在文件上传队列中的位置。如果允许上传的文件上限（file\_upload\_limit）或者剩余的允许文件上传数量小于文件队列上限（file\_queue\_limit），那么该值将采用这个更小的值。

flash\_url

默认值： 空字符串

设置绝对或者相对于此上传页面的完整URL，一旦SWFUpload实例化以后，此设置不能再被修改。

**提醒：**测试发现使用setUploadURL方法是可以修改此设置的。

flash\_width

固定值： 1px

(v2.1.0已删除) 设置插入flash影片的HTML元素容器的宽度。如果此设置小于1像素，一些浏览器会出现功能异常。因此该值在v2.1.0中删除了自定义设置，默认设置为1像素了。

flash\_height

固定值： 1px

(v2.1.0已删除) 设置插入flash影片的HTML元素容器的高度。如果此设置小于1像素，一些浏览器会出现功能异常。 因此该值在v2.1.0中删除了自定义设置，默认设置为1像素了。

flash\_color

默认值： #FFFFFF

(v2.2.0已删除) 设置HTML页面中的flash背景色，默认为#FFFFFF

prevent\_swf\_caching

默认值： true

(v2.2.0新增)该布尔值设置是否在Flash URL后添加一个随机值，用来防止浏览器缓存了该SWF影片。这是为了解决一些基于IE-engine的浏览器上的出现一个BUG。

**提醒：**SWFUpload是直接在flash\_url后添加了一个swfuploadrnd的随机参数。如果你给定的flash\_url中已经存在了GET类型的参数，那么就会出现两个问号连接符导致错误。

debug

默认值： false

该值是布尔类型，设置debug事件是否被触发。

**注意：**SWFUpload代码中是将此变量和字符串true做的恒等判断，因此它只认定true为DEBUG模式开启，如果设置为1，虽然JS认定是开启模式，并且在初始化完毕后会有生成Debug Console，但后续操作中FLASH不会输出调试信息。（因为我习惯用1和0代表布尔变量，因此一度疑惑为何Flash的debug信息无法输出。）

button\_placeholder\_id

默认值： null

(v2.2.0新增) 该必要参数指定了swfupload.swf将要替换的页面内的DOM元素的ID值。当对应的DOM元素被替换为SWF元素时，SWF的容器会被添加一个名称为"swfupload"的样式选择器供CSS自定义使用。

button\_image\_url

默认值： 空字符串

(v2.2.0新增) V2.2.0版最大的改变就是引入了一个按钮到SWF中，利用该参数可以设置一个相对于该swf文件或者是绝对地址的图片（或者是SWF），作为按钮的UI展现。所有FLASH支持的图片类型都可以使用（gif,jpg,png,或者是一个SWF）。

该按钮图片需要经过一定规则（CSS Sprite）的处理。按钮图片中需要包括按钮的4个状态， 从上到下依次是normal, hover, down/click, disabled.(可以参照官方demo中的图片)

button\_width

默认值： 1

(v2.2.0新增) 设置该SWF的宽度属性。

button\_height

默认值： 1

(v2.2.0新增)设置该SWF的高度属性（按钮图片高度的1/4）

button\_text

默认值： 空字符串

(v2.2.0新增) 该属性设置Flash Button中显示的文字，支持HTML。HTML文本的样式可以通过CSS选择器并配合button\_text\_style参数来设置。关于Flash文本对HTML的支持详细可见 [Adobe's Flash documentation](#)。

button\_text\_style

默认值： "color: #000000; font-size: 16pt;"

(v2.2.0新增)此参数配合button\_text参数，可以通过CSS样式来设置Flash Button中的文字样式。关于Flash文本对CSS的支持详细可见[Adobe's Flash documentation](#)

button\_text\_top\_padding

默认值： 0



(v2.2.0新增) 设置Flash Button上文字距离顶部的距离，可以使用负值。

#### button\_text\_left\_padding

默认值：0

(v2.2.0新增) 设置Flash Button上文字距离左侧的距离，可以使用负值。

#### button\_action

默认值：SWFUpload.BUTTON\_ACTION.SELECT\_FILES(多文件上传)

(v2.2.0新增) 设置Flash Button点击以后的动作。默认为SWFUpload.BUTTON\_ACTION.SELECT\_FILES，点击按钮将会打开多文件上传的对话框。如果设置为SWFUpload.BUTTON\_ACTION.SELECT\_FILE，则为单文件上传。如果设置为SWFUpload.BUTTON\_ACTION.START\_UPLOAD，则启动文件上传。

#### button\_disabled

默认值：false

(v2.2.0新增) 该布尔值设置Flash Button是否是禁用状态。当它处于禁用状态的时候，点击不会执行任何操作。

#### button\_cursor

默认值：SWFUpload.CURSOR.ARROW(箭头光标)

(v2.2.0新增) 此参数可以设置鼠标划过Flash Button时的光标状态。默认为SWFUpload.CURSOR.ARROW，如果设置为SWFUpload.CURSOR.HAND，则为手形

#### button\_window\_mode

默认值：SWFUpload.WINDOW\_MODE.WINDOW

(v2.2.0新增) 此参数可以设置浏览器具体以哪种模式显示该SWF影片。

#### custom\_settings

默认值：空的Object对象

该属性接收一个Object类型数据，可用于安全地存储与SWFUpload实例关联的自定义信息，例如属性和方法，而不用担心跟SWFUpload内部的方法和属性冲突以及版本升级的兼容。

设置完毕以后，可以通过实例的customSettings属性来访问。

```
var swfu = new SWFUpload({
    custom_settings : {
        "My Setting" : "This is my setting",
        myothersetting : "This is my other setting",
        integer_setting : 100,
        a_dom_setting : document.getElementById("some_element_id")
    }
});

var my_setting = swfu.customSettings["My Setting"];
swfu.customSettings["My Setting"] = "This is my new setting";
swfu.customSetting.myothersetting = "another new value";
swfu.customSetting.integer_setting += 25;
swfu.customSetting["a_dom_setting"].style.visibility = "hidden";
```



#### Event Handlers

默认值：null

剩下的设置定义的是一系列事件处理的回调函数，在SWFUpload的操作过程中[相应的事件](#)会被触发。如果你需要在这些回调中进行自定义操作，那么你应该在设置中定义对应的JavaScript函数。

## File Object

File Object包含了一组可用的文件属性，很多处理事件都会传递一个File Object参数来访问该文件的相关属性。

```
{
    id : string,           // SWFUpload控制的文件的id,通过指定该id可启动此文件的上传、退出上传等
    index : number,       // 文件在选定文件队列（包括出错、退出、排队的文件）中的索引，getFile可使用此索引
    name : string,        // 文件名，不包括文件的路径。
    size : number,        // 文件字节数
    type : string,        // 客户端操作系统设置的文件类型
```



```
creationdate : Date, // 文件的创建时间
modificationdate : Date, // 文件的最后修改时间
filestatus : number // 文件的当前状态, 对应的状态代码可查看SWFUpload.FIL
```

3

4

## Stats Object

该对象提供了上传队列的状态信息，访问实例的`getStats`方法可获取此对象。

该对象包括下面属性：

```
{
  in_progress : number           // 值为1或0，1表示当前有文件正在上传，0表示当前没有文件正在上传
  files_queued : number          // 当前上传队列中存在的文件数量
  successful_uploads : number    // 已经上传成功（uploadSuccess触发）的文件数量
  upload_errors : number         // 已经上传失败的文件数量（不包括退出上传的文件）
  upload_cancelled : number      // 退出上传的文件数量
  queue_errors : number          // 入队失败（fileQueueError触发）的文件数量
}
```

所有这些属性的值都可以使用`setStats`方法来修改，除了`in_progress` 和 `files_queued`。

## SWFUpload 插件

With SWFUpload v2.0 several plugins have been introduced. They are provided to help with common tasks associated with implementing SWFUpload.

Currently most of the documentation for using the plugins is contained in the plugin JavaScript file.

### SWFObject

The SWFObject plugin uses the [SWFObject library](#) to handle the embedding of the SWFUpload Flash Component into the page.

This plugin also provides support for Document Ready loading and Flash Version Detection. Usage details are documented in the plugin file itself.

Using this plugin can eliminate the flicker seen in the Graceful Degradation plugin and provides events for loading failure.

**Flash Player 10:** Because Flash Player 10 requires the SWFUpload swf to act is a button the movie must be visible in order for it to load. If the `button_placeholder_id` is set to an element that is hidden (visibility set to hidden or display set to none) SWFUpload will fail to load.

### Graceful Degradation

**This plugin has been superseded by the SWFObject plugin.**

This plugin provides additional settings and logic for automatically hiding and showing page elements based on whether SWFUpload loads.

**Flash Player 10:** The Graceful Degradation plugin has been updated to support Flash Player 10. However, because the Flash Movie won't execute if it is hidden (inside an element with visibility set to hidden or display set to none) the movie must be loaded in a visible area of the page. This is handled automatically. However, the side affect is that the movie is executed twice. The plugin handles this but developers should be aware that this is happening. In most cases this simply means that you will see the SWFUpload initialization (if debug is enable) run twice. There is still only a single movie per instance and events are not fired twice.

### SWFUpload v1.0.2 Compatibility

Creates compatibility with v1.0.2. Using this plugin you should be able to drop SWFUpload v2.0 in to a v1.0.2 page. This plugin is not supported and is only intended for users of SWFUpload v1.0.2 who wish to upgrade to v2 without recoding their pages.

### Cookies

In response to the Flash Cookie Bug the Cookies Plugin automatically retrieves your browser's cookies and sends them with the upload. The are sent as POST or GET variables to the upload url.

### Queue Handling

This plugin provides Queue Handling features such as entire queue uploading, entire queue cancelling and automatic starting of uploads after being queued.

## Known Issues

The Flash Player and many Browsers have bugs that have a direct impact on the performance of SWFUpload. While we have worked hard to get around many issues there are some things that cannot be fix.

## Cancelling in Linux

Older Flash 9 Players for Linux cause the browser to crash if an upload is cancelled. Newer Flash 9 Players behave better.

## Upload Progress in Linux

The Flash Player in Linux sends a single uploadProgress event after the file has finished uploading.

## Upload Progress in OS X

There have been some reports that uploadProgress events are not fired in MAC Flash Players. The specifics haven't been pinned down but be aware of the possible issue.

## MIME Type

The Flash Player uploads all files with a mime type of *application/octet-stream* regardless of the file's actual mime type.

## Maximum number of selected files

The Flash Player does not impose a maximum number of selected files. However, it builds a selected files string which does have a maximum length. The string is built using the file's name and the separator [quote][space][quote]. The total number of files selected is determined by the sum of the lengths of the file names and a prefixed and postfixed [quote] character (2 characters) and the number of files selected minus one times 3 (for the separator string)

This limitation may vary from system to system. If a use selects too many files they will be receive a Flash Player generated warning message and will be left at the File Selection Dialog.

## Proxies

The Flash Player may not properly use proxies. It does not handle authenticating proxies well (if at all) and will some-times crash.

Some anti-virus software uses a proxy to scan uploads and cause SWFUpload to believe the entire file has been uploaded. SWFUpload will fire uploadProgress events very quickly until it reaches 100% and will then seem to pause until the proxy completes uploading the file to the server.

## Apache mod\_security

Apache's mod\_security validates POST to the server. Flash Player has implemented an edge case (there is argument as to whether it is invalid or note) POST for file uploads and so servers implementing mod\_security will reject the upload. mod\_security can be disabled using your .htaccess file

## SSL

There have been some reports that the Flash Player cannot upload through SSL. The issue has not been pinned down but uploading over SSL may be unreliable. There especially seems to be an issue with using self-signed certificates.

Also, SSL tickets from untrusted Certificate Authorities (CA) do not work as Flash does not provide a method for accepting the certificate. It has been noted that, like the cookie bug, that Flash Player on Windows obtains its trusted CA list from Internet Explorer regardless of the browser in use.

## Authentication

HTTP Authentication is not well supported by the Flash Player. Later versions of Flash Player behave better. Old version of Flash Player would crash the browser.

## Prematurely terminated connections

Prematurely ending the response (such as a Response.end() in ASP.Net) can sometimes cause successful uploads to be reported as failed.

## Filedata in Linux

Changing the Filedata value (file\_post\_name setting) is ignored in Linux Flash Player.

## uploadSuccess & Server Data

The uploadSuccess will not fire if no data is returned from the server. This is especially problematic on the Mac version of the Flash Player.

## Cookie issue

On Windows the Non-IE Flash Player plugin (FireFox, Opera, Safari, etc) sends the IE cookies regardless of the browser used. This breaks authentication and sessions for many server-side scripting technologies.

Developers should manually pass Session and Authentication cookie information and manually restore Sessions on the Server Side if they wish to use Sessions

The SWFUpload package contains work-around sample code for PHP and ASP.Net

## ExternalInterface bugs

Flash Player does not properly escape data when communication with the browser/JavaScript. SWFUpload goes to great lengths to work-around this issue. If this bug is fixed in future Flash Players/Browsers then SWFUpload will send extra escaped data.

## Server Data length bugs

Very long server data is corrupted on Mac and Linux Flash Players. Server data will be truncated, garbled, and/or repeated. We recommend keeping data returned from the server short and concise.

## Avant Browser

For some users the Avant Browser does not work with SWFUpload after the Flash Control has been cached. This has been reproduced by the SWFUpload developers but the Avant Browser developers did not experience any problems.

When the page is reloaded SWFUpload loads and fires the swfupload\_loaded event. However, none of the ExternalInterface callback functions are defined on the movie element. SWFUpload v2.0.2 has added checks which prevent swfupload\_loaded from firing if the callback functions are not detected.

SWFUpload v2.2.0 added the prevent\_swf\_caching setting that attempts to work around this issue.

## File Dialog & Page Changing

Leaving or reloading the current page while the File Browser Dialog window is open will cause the browser to crash (all browsers, all OSs). Most commonly this is caused by failing to cancel a click event on an <a> tag where the onclick event calls the selectFile or selectFiles function.

## Long Running Upload Scripts

After Flash has uploaded the file to the webserver the upload script is executed. This script handles the file whether that means saving it, creating a thumbnail, scanning for viruses, etc. If the upload script does not return any data within 30 seconds Flash will disconnect and return an IO Error. You can prevent this by returning characters or data while the script runs (if possible). For PHP, the script continues to run and complete successfully even though Flash has terminated the connection. Any data returned by the script after Flash has disconnected is lost.

## Other Mac Issues

- Users have reported that uploading to subdomains does not work with the Mac Flash Player.
- Users have reported that pages that redirect (HTTP Status 302) are not handled by the Mac Flash Player. Windows clients seem to handle this issue.
- The flash documentation states that on OS early than OS X 10.3 the bytes loaded is always reported as -1. SWFUpload converts this to 0 but the total bytes will never be sent and 100% won't be reached. The UI should be updated to display 100% complete in uploadSuccess or uploadComplete events to maintain a consistent UI.
- Some users have reported issues if there is a space character in the upload\_url for the Mac Flash Player. Avoid spaces or try replacing them with + or %20.
- Users have reported that the Flash Player for Mac adds the PORT to the HTTP\_HOST server variable (e.g., http://www.example.com:80). If you are checking this variable in your server-side script be aware of the possible issue.

