

# Uploadify v1.6.2

©2009 by Ronnie Garcia

Developed by Ronnie Garcia and Travis Nickels

www.ronniesan.com

## WHAT IS IT?

This plugin allows you to change any element with an ID on your page into a single or multiple file upload tool. The plugin uses a mix of JQuery, Flash, and a backend upload script of your choice to send files from your local computer to your website server.

## HOW DO I IMPLEMENT IT?

Implementation of the plugin is very easy and only relies on one JQuery function call to initiate.

1. Download the latest zip package from <http://www.ronniesan.com/articles/jquery-multiple-file-upload.php>
2. Extract the files and upload them to your server.
3. Link the jquery script and the fileupload plugin to the page you will be using the plugin on. You can do so with the following code in the <head> section of your page:

```
<script type="text/javascript" src="/path/to/jquery-1.3.2.min.js"></script>
<script type="text/javascript" src="/path/to/jquery.uploadify.v.1.6.0.js"></script>
```

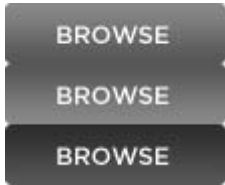
4. Add the call to the plugin using the \$.ready event in the <head> section of your page:

```
<script type="text/javascript">
$(document).ready(function() {
    $('#someID').fileUpload({
        'uploader': '/path/to/uploader.swf',
        'script': '/path/to/upload.php',
        'folder': '/path/to/uploads-folder',
        'cancelImg': '/path/to/cancel.png'
    });
});
</script>
```

When the document is done loading, the elements that the function was called against will be hidden and replaced by the browse button. The component is now ready to use on your page.

## AVAILABLE OPTIONS

<b>uploader</b>	The path to the uploader.swf file. Default = 'uploader.swf'
<b>script</b>	The path to the backend script that will be processing your uploaded files. Default = 'upload.php'
<b>scriptData</b>	An object containing name/value pairs of additional information you would like sent to the upload script. {'name': 'value'}

<b>scriptAccess</b>	The access mode for scripts in the flash file. If you are testing locally, set to 'always'. Default = 'sameDomain'
<b>folder</b>	The path to the folder you would like to save the files to. Do not end the path with a '/'.
<b>multi</b>	Set to <b>true</b> if you want to allow multiple file uploads.
<b>auto</b>	Set to <b>true</b> if you would like the files to be uploaded when they are selected.
<b>fileDesc</b>	The text that will appear in the file type drop down at the bottom of the browse dialog box.
<b>fileExt</b>	A list of file extensions you would like to allow for upload. Format like '*.ext1;*.ext2;*.ext3'.
<b>sizeLimit</b>	A number representing the limit in bytes for each upload.
<b>buttonText</b>	The text you would like to appear on the default button. Default = 'BROWSE'
<b>buttonImg</b>	The path to the image you will be using for the browse button.
<b>rollover</b>	Set to <b>true</b> if you would like to activate rollover states for your browse button. To prepare your browse button for rollover states, simple add the 'over' and 'press' states below the normal state in a single file. See below for an example:   <p>*Mouseover events are inconsistent in Flash 9 so you may see a short lag when using the rollover option.</p>
<b>width</b>	The width of the button image / flash file. Default = 110
<b>height</b>	The height of the button image / flash file. If <b>rollover</b> is set to <b>true</b> , this should be 1/3 the height of the actual file. Default = 30
<b>wmode</b>	Set to <b>transparent</b> to make the background of the flash file transparent. If set to <b>transparent</b> , the flash file will be at the top-most layer of the page. By omitting the <b>buttonImg</b> option and setting <b>wmode</b> to <b>transparent</b> , the entire flash file will be transparent, allowing you layer content below it or style the button using CSS. Default = 'opaque'
<b>cancelImg</b>	The path to the default cancel image. Default = 'cancel.png'
<b>displayData</b>	Setting to 'percentage' will display the percentage complete next to the file name. Setting to 'speed' will show the current upload speed. Omit this option to not show anything.

---

<b>onInit</b>	<p>A function that triggers when the script is loaded. The default event handler hides the targeted element on the page and replaces it with the flash file, then creates a queue container after it. The default function will not trigger if the value of your custom function returns <b>false</b>. For custom functions, you can access the html for the flash file using the variable <b>flashElement</b>.</p>
<b>onSelect</b>	<p>A function that triggers for each element selected. The default event handler generates a 6 character random string as the unique identifier for the file item and creates a file queue item for the file. The default event handler will not trigger if the value of your custom function returns <b>false</b>.</p> <p>Three arguments are passed to the function:</p> <p><b>event:</b> The event object.</p> <p><b>queueID:</b> The unique identifier of the file that was selected.</p> <p><b>fileObj:</b> An object containing details about the file that was selected.</p> <ul style="list-style-type: none"><li>• <b>name</b> – The name of the file</li><li>• <b>size</b> – The size in bytes of the file</li><li>• <b>creationDate</b> – The date the file was created</li><li>• <b>modificationDate</b> – The last date the file was modified</li><li>• <b>type</b> – The file extension beginning with a '.'</li></ul>
<b>onSelectOnce</b>	<p>A function that triggers once for each select operation. There is no default event handler.</p> <p>Two arguments are sent to the function:</p> <p><b>event:</b> The event object.</p> <p><b>data:</b> An object containing details about the select operation.</p> <ul style="list-style-type: none"><li>• <b>fileCount</b> – The total number of files in the queue</li><li>• <b>filesSelected</b> – The number of files selected in the select operation</li><li>• <b>filesReplaced</b> – The number of files that were replaced in the queue</li><li>• <b>allBytesTotal</b> – The total number of bytes for all files in the queue</li></ul>
<b>onCancel</b>	<p>A function that triggers when a file upload is cancelled or removed from the queue. The default event handler removes the file from the upload queue. The default event handler will not trigger if the value of your custom function returns <b>false</b>.</p> <p>Four arguments are sent to the function:</p> <p><b>event:</b> The event object.</p> <p><b>queueID:</b> The unique identifier of the file that was cancelled.</p> <p><b>fileObj:</b> An object containing details about the file that was selected.</p> <ul style="list-style-type: none"><li>• <b>name</b> – The name of the file</li><li>• <b>size</b> – The size in bytes of the file</li><li>• <b>creationDate</b> – The date the file was created</li><li>• <b>modificationDate</b> – The last date the file was modified</li><li>• <b>type</b> – The file extension beginning with a '.'</li></ul> <p><b>data:</b> Details about the file queue.</p> <ul style="list-style-type: none"><li>• <b>fileCount</b> – The total number of files left in the queue</li><li>• <b>allBytesTotal</b> – The total number of bytes left for all files in the queue</li></ul>

---

---

## onClearQueue

A function that triggers when the **fileUploadClearQueue** function is called. The default event handler removes all queue items from the upload queue. The default event handler will not trigger if the value of your custom function returns **false**.

Two arguments are sent to the function:

**event:** The event object.

**data:** An object containing details about the file queue.

- **fileCount** – The number of files remaining in the upload queue
- **allBytesTotal** – The number of bytes remaining in the upload queue

---

## onError

A function that triggers when an error occurs during the upload process. The default event handler attaches an error message to the queue item returning the error and changes it's queue item container to red.

Four arguments are sent to the function:

**event:** The event object.

**queueID:** The unique identifier of the file that was cancelled.

**fileObj:** An object containing details about the file that was selected.

- **name** – The name of the file
- **size** – The size in bytes of the file
- **creationDate** – The date the file was created
- **modificationDate** – The last date the file was modified
- **type** – The file extension beginning with a '.'

**errorObj:** An object containing details about the error returned.

- **type** – Either 'HTTP', 'IO', or 'Security'
- **status** – (For HTTP only) The HTTP status that was returned
- **text** – (For IO or Security only) An error message describing the type of error returned

---

## onProgress

A function that fires each time the progress of a file upload changes. The default function updates the progress bar in the file queue item. The default function will not trigger if the value of your custom function returns **false**.

Four arguments are sent to function:

**event:** The event object.

**queueID:** The unique identifier of the file that was cancelled.

**fileObj:** An object containing details about the file that was selected.

- **name** – The name of the file
- **size** – The size in bytes of the file
- **creationDate** – The date the file was created
- **modificationDate** – The last date the file was modified
- **type** – The file extension beginning with a '.'

**data:** An object containing details about the upload and queue.

- **percentage** – The current percentage completed for the upload
  - **bytesLoaded** – The current amount of bytes uploaded
  - **allBytesLoaded** – The current amount of bytes loaded for all files in the queue
  - **speed** – The current upload speed in KB/s
-

---

## onComplete

A function that triggers when a file upload has completed. The default function removes the file queue item from the upload queue. The default function will not trigger if the value of your custom function returns **false**.

Four arguments are sent to the function:

**event:** The event object.

**queueID:** The unique identifier of the file that was cancelled.

**fileObj:** An object containing details about the file that was selected.

- **name** – The name of the file
- **filepath** – The path on the server to the uploaded file
- **size** – The size in bytes of the file
- **creationDate** – The date the file was created
- **modificationDate** – The last date the file was modified
- **type** – The file extension beginning with a '.'

**response:** The data sent back from the server.

**data:** Details about the file queue.

- **fileCount** – The total number of files left in the queue
- **speed** – The average speed of the file upload in KB/s

---

## onAllComplete

A function that triggers when all file uploads have completed. There is no default event handler.

Two arguments are sent to the function:

**event:** The event object.

**data:** An object containing details about the upload process.

- **filesUploaded** – The total number of files uploaded
  - **errors** – The total number of errors while uploading
  - **allbytesLoaded** – The total number of bytes uploaded
  - **speed** – The average speed of all uploaded files
-

## RELATED FUNCTIONS

**fileUploadSettings** A function used to change options for a fileUpload object.

Two arguments are required:

**setting:** The option to change.

**value:** The value to change the option to.

The following example will change the upload folder to '/uploads':

```
$('#someID').fileUploadSettings('folder', '/uploads');
```

The following options can be changed:

buttonImg, buttonText, cancellmg, fileDesc, fileExt, multi, btnWidth, btnHeight, folder, script, scriptData, simUploadLimit, sizeLimit, checkScript, hideButton, fileName, auto, rollover

**fileUploadStart** A function used to begin an upload of a single file or all files in the queue.

One argument is optional:

**queueID:** The unique queue identifier for the file to upload.

The following example will upload all files in the queue:

```
$('#someID').fileUploadStart();
```

**fileUploadCancel** A function used to remove a file from the queue or stop an upload in progress.

One argument is required.

**queueID:** The unique queue identifier of the file you wish to cancel.

The following example will remove a file from the upload queue:

```
$('#someID').fileUploadCancel('NFJSHS');
```

**fileUploadClearQueue** A function used to clear all files from the upload queue.

The following example clears the file upload queue:

```
$('#someID').fileUploadClearQueue();
```