## Introduction

File uploads need to be dealt with by the client and the server.  Here we explain how uploading a file can be done with simplisity.js.  Although the sever side code can be created in any language and any server, in this tutorial we will be using C#.

## JS files required

When simplisity is required to upload files we must include 2 JS files on the page:

<script type="text/javascript" src="/DesktopModules/DNNrocket/Simplisity/js/simplisity.js">

</script>

<script type="text/javascript" src="/DesktopModules/DNNrocket/Simplisity/js/jquery.fileupload.js">

</script>

## Client Side HTML/JS

We need to create an input field and for file upload. (See example below)

<input

class="simplisity\_fileupload"

s-cmd="rocketimages\_upload"

s-reload="true"

s-return=""

s-uploadcmdurl="/Desktopmodules/dnnrocket/api/fileupload/upload"

s-cmdurl="/Desktopmodules/dnnrocket/api/rocket/action"

s-fields='{"resize":"640"}'

id="imagefileupload"

type="file"

name="files[]"

multiple />

**s-cmd** : command to run on server.  
**s-reload** : Reload the page after upload, this is sometimes required to get a correct display of the uploaded files, however this depends on if the server is returning a html rendered to display the new files.  
**s-return** : In this case we reload the page, so we have no return.  
**s-uploadcmdurl** : The upload is done in 2 part, this command URL identifies the end point for the file upload. Obviously the server side code must be setup to deal with the file list.   
**s-cmdurl** : After the upload has been completed, we will want to run a command (s-cmd) to do the required processing, server side. This is the endpoint for the s-cmd action.   
**s-fields** : Parameters for the server side processing can also be passed using the s-fields property.

*This example is for demonstration, the actual functionality and setup will be adjusted to match the server-side code.*

### What simplisityjs does with this:

When this input field is placed in a simplisity\_panel, simplisity will activate a fileupload event, in preparation for a file upload to be performed.

The “initFileUpload” in simplisity.js function is the initialization for jquery upload and sets the event action when the “simplisity\_fileupload” input field is activated by a file select. It is the jquery functions in “jquery.fileupload.js” that does the file upload.

When an image is selected using the file input the selected files will be uploaded to the server.

*REMEMBER: to have any usefulness the server-side code, which is activated by simplisity, must take the required action on the uploaded files.*

***TIPS:***

* Each file input field has a unique id on the page.
* Check for JS errors if the upload does not work.
* The JS files must be included on the page.

## Progress bar

The simplsaity.js allows for a progress bar to be show. Any progress bar with the class “simplisity-file-progress-bar” will be updated with the progress of the file upload.

<div class=" w3-row w3-light-grey w3-margin">

<div class="w3-blue simplisity-file-progress-bar" style="width:0%">0%</div>

</div>

## Large Files

Large files will be partitioned and uploaded in chucks. This makes it possible to upload large files without effecting the server memory too much.

To control the chuck size the attribute “s-maxchunksize” is used on the file input field.   
This value is in bytes, the default value is 10MB [s-maxchunksize="10000000"].

*NOTE: The server-side code must deal with chuck upload, if large files are being uploaded.*

## Server-Side Code