[**How to upload large files in PHP**](http://www.techawaken.com/upload-large-files-php/)

January 11, 2017 [PHP](http://www.techawaken.com/category/php/)

[Share](https://www.facebook.com/sharer/sharer.php?s=100&p%5burl%5d=http://www.techawaken.com/upload-large-files-php/)

If you are planning to implement file upload functionality in PHP, the first thing that you need to verify is that whether file upload functionality has been turned on or not in your php.ini configuration file. The setting that you need to look for is *“file\_uploads”*.

|  |  |  |
| --- | --- | --- |
| 1 | ; Whether to allow HTTP file uploads. | |
| 2 | ; http://php.net/file-uploads |

|  |  |
| --- | --- |
| 3 | file\_uploads = On |

If you don’t know where your php configuration file is located, then create the below single line PHP file *‘phpinfo.php’* and place it in your docroot folder.

|  |  |
| --- | --- |
| 1 | <?php |
| 2 | phpinfo(); | |

|  |  |
| --- | --- |
| 3 | ?> |

Now access the file via browser http://somedomain.com/phpinfo.php and look for the *“Loaded Configuration File”* value.

**1. Configure PHP**

By default PHP is configured to allow files upload of size upto 2M.

Now if we have to upload files say upto 500M then how do we do it?

We need to adjust the following PHP configuration directives:

1. **upload\_max\_filesize**: By default this value is 2M. We need to increase it to the maximum size of single file that we want to upload.
2. **post\_max\_size**: It defines the maximum size of POST data that PHP will accept. This value should be greater than *‘upload\_max\_filesize’*.
3. **memory\_limit:** This sets the amount of memory a PHP script is allowed to use during its execution. Set this to a value greater than *‘post\_max\_size’* so that PHP script can load and process the uploaded file.

If you notice PHP execution timeout in your log file which may be the case when you are processing the uploaded file like uploading an image, manipulating it and then saving it, then you must consider increasing the following timeout values. The values are to be specified in seconds.

1. **max\_input\_time:** This sets the maximum time in seconds a script is allowed to parse input data, like POST and GET. Timing begins at the moment PHP is invoked at the server and ends when execution begins. This would include populating $\_FILES superglobal.
2. **max\_execution\_time:** The time a script is allowed to run after its input has been parsed. This would include any processing of the file itself.

If you are getting memory related error then turn off the output buffering, the PHP configuration directive to be considered is “output\_buffering”

|  |  |
| --- | --- |
| 1 | output\_buffering = Off |

**Temporary directory for HTTP uploaded files**

You might need to consider the your temp file location (uploaded file is initially stored at this location) also. It should have enough space so that it can save the uploaded file. You must keep in mind that multiple users might be uploading files at the same time when deciding the temporary location.

|  |  |  |
| --- | --- | --- |
| 1 | ; Temporary directory for HTTP uploaded files (will use system default if not | |
| 2 | ; specified). |

|  |  |  |
| --- | --- | --- |
| 3 | ; http://php.net/upload-tmp-dir | |
| 4 | ;upload\_tmp\_dir = |

For these php.ini configuration changes to take effect, you will have restart the apache server if you are using PHP as an apache module. In case you are using PHP-FPM with nginx then you need to restart PHP-FPM.

**What if you don’t have access to the php.ini file?**

* Configure the setting in your application’s .htaccess file

|  |  |  |
| --- | --- | --- |
| 1 | php\_value upload\_max\_filesize 500M | |
| 2 | php\_value post\_max\_size 550M |

|  |  |
| --- | --- |
| 3 | php\_value memory\_limit 1024M |
| 4 | php\_value max\_input\_time 300 |

|  |  |
| --- | --- |
| 5 | php\_value max\_execution\_time 300 |

* Alternatively, you can define the values in your PHP script itself

|  |  |  |
| --- | --- | --- |
| 1 | ini\_set('upload\_max\_filesize', '500M'); | |
| 2 | ini\_set('post\_max\_size', '550M'); |

|  |  |  |
| --- | --- | --- |
| 3 | ini\_set('memory\_limit', '1024M'); | |
| 4 | ini\_set('max\_input\_time', 300); |

|  |  |
| --- | --- |
| 5 | ini\_set('max\_execution\_time', 300); |

**2. Configure Webserver**

Check your web server configuration files for the following setting as they may not allow to upload large size files.

* **Nginx**: [client\_max\_body\_size](http://nginx.org/en/docs/http/ngx_http_core_module.html#client_max_body_size), [client\_body\_buffer\_size](http://nginx.org/en/docs/http/ngx_http_core_module.html#client_body_buffer_size), [client\_body\_temp\_path](http://nginx.org/en/docs/http/ngx_http_core_module.html#client_body_temp_path), [fastcgi\_read\_timeout](http://nginx.org/en/docs/http/ngx_http_fastcgi_module.html#fastcgi_read_timeout)(nginx php location block)
* **Apache:** [LimitRequestBody](https://httpd.apache.org/docs/current/en/mod/core.html#limitrequestbody), [SSLRenegBufferSize](https://httpd.apache.org/docs/current/mod/mod_ssl.html#sslrenegbuffersize), [RequestReadTimeout](https://httpd.apache.org/docs/current/mod/mod_reqtimeout.html) (If you are using “mod\_reqtimeout” apache module and having issues with large file uploads then either disable the module or increase the timeout.)
* **Apache with mod\_fcgid:** [FcgidMaxRequestLen](https://httpd.apache.org/mod_fcgid/mod/mod_fcgid.html#fcgidmaxrequestlen)
* **IIS:** [maxAllowedContentLength](https://www.iis.net/configreference/system.webserver/security/requestfiltering/requestlimits#005)

I was using Nginx with PHP- FPM, so for allowing upto 500M uploads I set the *‘client\_max\_body\_size’* as ‘510M’ in my nginx config file (/etc/nginx/nginx.conf).

|  |  |
| --- | --- |
| 01 | http { |
| 02 | ## |

|  |  |  |
| --- | --- | --- |
| 03 | # Basic Settings | |
| 04 | ## |

|  |  |
| --- | --- |
| 05 | sendfile on; |
| 06 | tcp\_nopush on; | |

|  |  |
| --- | --- |
| 07 | tcp\_nodelay on; |
| 08 | keepalive\_timeout 65; | |

|  |  |  |
| --- | --- | --- |
| 09 | types\_hash\_max\_size 2048; | |
| 10 | # server\_tokens off; |

|  |  |
| --- | --- |
| 11 | client\_max\_body\_size 510M; |
| 12 | server\_names\_hash\_bucket\_size 64; | |

|  |  |
| --- | --- |
| 13 | # server\_name\_in\_redirect off; |
| 14 | include /etc/nginx/mime.types; |

|  |  |  |
| --- | --- | --- |
| 15 | default\_type application/octet-stream; | |
| 16 | . |

|  |  |
| --- | --- |
| 17 | . |
| 18 | . |

|  |  |
| --- | --- |
| 19 | } |

**To Begin with…**

1. I would say, configure the following two ***php.ini*** settings:
   1. **upload\_max\_filesize**
   2. **post\_max\_size**
2. Test the file upload functionality
3. Check errors and log files
4. Reconfigure php and the webserver
5. Repeat from step 2 until you are successful in uploading the file.

|  |
| --- |
| **How to Create an Upload File PHP Script** |
| http://www.atksolutions.com/images/myline.gif |
| http://www.atksolutions.com/images/trans_1x1.gif |
| |  |  |  | | --- | --- | --- | | |  |  | | --- | --- | | http://www.atksolutions.com/images/trans_1x1.gif |  |   **This article will require some PHP programming knowledge.** It is a simple form script that allows you to upload a file to a web server. It uses HTML for the front end and PHP programming language for the back end. In order to successfully accomplish this I would suggest creating a directory on the web server and let's say we will call it "**upload**".   Next, we need to create a simple HTML form, with a file browser button that lets you choose the file to upload to the server. Once you press submit, that file will be uploaded. Below is some sample code with explanations on each section. The first part of the script is PHP code. This code is executed once the user presses "**Upload File**" is pressed.   Below that where the HTML part starts is the form for the user. There is a lot of upload file script variations out on the web. I tried to break this down to its simplest form for ease. Obviously, you can change it to fit your needs... | |
| **Here are the steps to create a file upload script:**  **Step 1:** Create a file name called **uploadfile.php**, open it and copy the following script into the file.    **Step 2:** Copy this file "**uploadfile.php**" to your web server using FTP. so if your website name is **http://www.mywebsite.com** then to run this script. simply enter **http://www.mywebsite.com/uploadfile.php**   **Step 3:** Select a small file, preferably a picture file from your computer and press "**Upload File**". The file should then upload to the same folder as this script.     **Several Things to Remember:**  **#1:** This PHP code should for the most part be self explanatory.   It checks the if block for **if (trim($\_POST["action"]) == "Upload File")** to see if the upload file button pressed and executes the commands after that. It simply grabs the file name that you selected from the "Browse" button and uploaded the file to the web server using that same name.    **#2:** Make sure the form tag has an **enctype='multipart/form-data'**. This will tell the php code that there is a select file block otherwise the PHP form will not know how to grab the filename from the HTML form.  and thats about it. make sure to keep your files small because by default PHP does not allow anything over 2MB. You can change this in the php.ini if you want to upload larger files. |

Uploading a file from a web form in PHP is easy. The online manual provides a [Handling File Uploads](http://www.php.net/manual/en/features.file-upload.php) section, and there are several articles on sitepoint.com, including [How To Handle File Uploads With PHP](http://articles.sitepoint.com/article/handle-file-uploads-php) by Kevin Yank.

One of the most popular uses is image uploads. Your users can submit photographs from a form without resorting to FTP or other convoluted methods. HTML5 and Flash also permit drag and drop, so the operation is likely to become easier as browsers evolve.

This is where the problems can begin. Camera manufacturers continually brag that they have a larger set of megapixels than their competitors. It’s all rubbish, of course — unless you’re a professional photographer or need to print extremely large images, anything over 4MP is fairly pointless and lens quality is much more important. However, even low-end compacts have 12MP and mobile phones have more than 5MP. The result is that a typical snapshot can easily be 6MB in size.

By default, PHP permits a maximum file upload of 2MB. You can ask users to resize their images before uploading but let’s face it: they won’t. Fortunately, we can increase the limit when necessary.

Two PHP configuration options control the maximum upload size: upload\_max\_filesize and post\_max\_size. Both can be set to, say, “10M” for 10 megabyte file sizes.

However, you also need to consider the time it takes to complete an upload. PHP scripts normally time-out after 30 seconds, but a 10MB file would take at least 3 minutes to upload on a healthy broadband connection (remember that upload speeds are typically five times slower than download speeds). In addition, manipulating or saving an uploaded image may also cause script time-outs. We therefore need to set PHP’s max\_input\_time and max\_execution\_time to something like 300 (5 minutes specified in seconds).

These options can be set in your server’s php.ini configuration file so that they apply to all your applications. Alternatively, if you’re using Apache, you can configure the settings in your application’s .htaccess file:

php\_value upload\_max\_filesize 10M

php\_value post\_max\_size 10M

php\_value max\_input\_time 300

php\_value max\_execution\_time 300

Finally, you can define the constraints within your PHP application:

ini\_set('upload\_max\_filesize', '10M');

ini\_set('post\_max\_size', '10M');

ini\_set('max\_input\_time', 300);

ini\_set('max\_execution\_time', 300);

PHP also provides a [set\_time\_limit()](http://www.php.net/manual/en/function.set-time-limit.php) function so you don’t need to set max\_execution\_time directly.

Setting the options in your PHP code is possibly more practical, since you can extend the execution time and increase the file size when your application is expecting a large upload. Other forms would revert to the default 30-second time-out and 2MB limit.

Do you have any other tips for uploading large files in PHP?