Intro

File upload in php is done in 2 ways. Either via HTTP or FTP.

We will focus on HTTP.

We will be using a form as our user interface, for selecting and uploading files but they are other ways of having a user select some files for upload.

This can also be done using a drag and drop interface.

As of PHP 5.2, the progress of a file being uploaded can be tracked.

Check list

They are a few checks that should be carried out before we begin uploading file.

- Are file uploads allowed on your server.
- Is there any limit to the size of a file that can be uploaded.
- Is the temporary file folder for upload set.

The directive must be set inside of the php.ini file in your servers php folder.

MAMP path: MAMP/conf/PHP5/php.ini

WAMP path: C:\wamp\bin\php\php5.2.x

Setting the directives

You must find the following lines in your php.ini file.

file_uploads
This takes a value of ON or OFF

upload_tmp_dir
The directory where temporary files are stored on the server.

upload_max_filesize
Maximum value of the file to be uploaded.

Example

```
file_uploads = On
upload_tmp_dir = "c:/wamp/tmp"; // MAMP/tmp/PHP5
upload_max_filesize = 2M
```

User interface (form)

To send legitimate files to the server, we use a form.

However they are a few new attributes and values that we'll use for the first time.

When declaring the form its ***enctype*** attribute must be set to **multipart/form-data**.
This allows PHP to process the information.

Example:

```
<form action='upload.php' method='post' enctype='multipart/form-data'>
```

Browse and Select

To browse a client directory for files via the form, an input tag must be declared and its ***type*** attribute set to ***file***.

- The input tags *name* attribute must also be set to a value.
- This value will serve as the ***key*** which is mapped to the file data sent to the server.

```
<input name='file1' type='file' />
```

The above renders the following in the browser.

Form example 1

Completed form for single file upload.

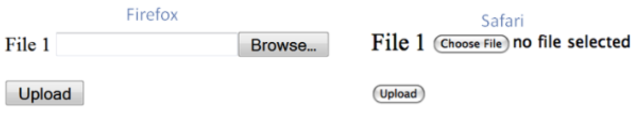
```
<form action='upload.php' method='post' enctype='multipart/form-data'>

   <label>File 1</label>
   <input name='file1' type='file'  /></br>

   <input name= 'submit' type='submit' value='Upload' />

</form>
```

The above renders the following in the browser.

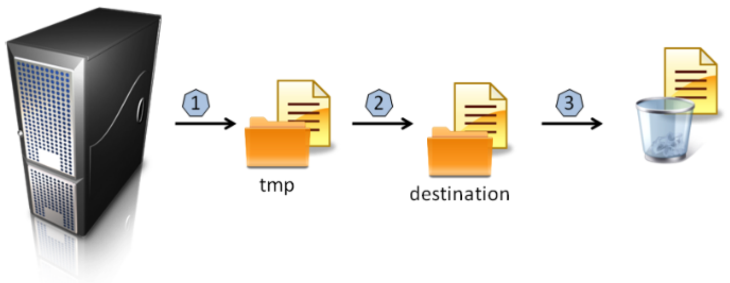


The temporary file

When the form is submitted, any files selected are moved to the temporary directory specified in the php.ini directive.

The temporary file resides in this directory on the server, until you move the file to its permanent location.

The temporary file



1) The file gets moved to a temporary folder on the server
1a) ***file can now be processed.***
2) PHP moves the file to a permanent location
3) The temporary file is deleted from the server

Types of file processing

Before any files are moved to their final destination you may want to:

- check the file is a valid file for uploading.
- The file size is not larger than that specified in the php.ini directive, and its greater than 0 bytes.
- Resize it, make a thumbnail from it or add a watermark to it.
- or some other form of processing, its up to you.

Processing the uploaded file

Although the forms' method is set to ***post***, the variables wont be found in the $_POST superglobals on the server but rather in the $_FILES superglobals.

Example:

```
<input name='file1' type='file'  />
```

To access the file data sent to the server, refer to its key through the superglobals.

```
$_FILES[' file1 ']
```

File data

The $_FILES superglobals compared to others is a multidimensional array.

$_FILES contains one to many associative arrays containing information about the file.

Example: ***site_estimate.doc***

```
Array
(
    [file1] => Array
    (
            [name] => site_estimate.doc
            [type] => application/msword
            [tmp_name] => C:\wamp\tmp\php29D6.tmp
            [error] => 0
            [size] => 53200
    )
)
```

File data table

$_FILES['file']

Key	Description
type	The uploaded file's MIME type http://www.w3schools.com/media/media_mimeref.asp. If the file has an extension not listed on W3C Schools its type would be application/octet-stream.
tmp_name	The temporary location of the file on the server.
error	An integer indicating any status of the file to upload.
size	Size of the file in bytes.
name	The original name of the uploaded file.

Error code

Different error codes are generated based on the type of error occurred whilst

Error Code	Description
0	File uploaded successfully.
1	File size is larger that what is specified directive in php.ini (seems to crash the browser)
2	File size is larger than, the MAX_FILE_SIZE. **(FLAWED)**
3	Partial file sent from client, cannot upload.
4	No file to upload, user didn't select any, wasn't passed from form.

Uploading the file

The move_uploaded_file($remote_file, $new_file) is used to move the *temporary file* to its *new location*.

The function takes the following arguments:

$remote_file - the temporary path to the file on the remote server.
$new_file - the new path of remote file.

The function returns a *Boolean* of true if successful upload was achieved and false otherwise.

Example:
```
move_uploaded_file( $_FILES['file1']['tmp_name'], $_FILES['file1']['name'] );
```

Validating the file

To see if the file was submitted via a form and is a valid *multipart/form-data requests*, use the is_uploaded_file($remote_file) function.

The function takes the *temporary remote file* as its only argument.

This function also returns a Boolean, so use it to check validity using a conditional.

Example:
```
If( is_uploaded_file($_FILES['file1']['tmp_name']) )
{
    echo "This  file was submitted via a form.";
}
```

Replacing white spaces

```
$file = str_replace(' ', '_', $_FILES['file1']['name']);
```

The function str_replace() takes the following three arguments:

- The character or substring that you want to replace—in this case, a space
- The character or substring that you want to insert—in this case, an underscore
- The string that you want to update—in this case, $_FILES['file1']['name']

Notes

Get familiar with the $_FILES superglobals array, and remember its an Associative array unless you assign numeric values in the XHTML form. E.g. <input name='0' />

Remember the $_FILES superglobals array is a multidimensional array, this will help you traverse it and handle with multiple file upload.

Since the move_uploaded_file() function returns a Boolean use it to your advantage.

Do not forget to check your php.ini settings before using file upload.