

File Upload Forms Using PHP by R. Berdan

In some instances you may want to allow users to upload files to a folder. These files might be images for instance or even PHP scripts for testing. You can limit the size and type of files they can upload – and by using a file upload form – the user can not directly access your web account like they would if you provided them with a FTP log on.

You could also include a script that takes someone's digital photos, embeds them into a web page and display them (another separate tutorial).

Here we are going to create a file upload form using two different approaches.

- 1) First will we create a separate form page and another php script to process the form.
- 2) In the second approach we will create a single php script that creates the form on the fly and permits the user to upload a file.

1. Open up notepad, or your favorite HTML editor and create a simple form.

```
<html>
<head>
<title>Upload your files</title>
</head>
<body>

<form method="post" action="upload.php" enctype="multipart/form-data">
<input type="hidden" name="MAX_FILE_SIZE" value="1000000">
Select a file to upload<br>
<input type="file" name="userfile"><br>
<input type="submit" value="Upload">
</form>

</body>
</html>
```

Save the file as upload1.html in your local web server or post it later on your web account that is PHP enabled.

Note setting the maximum file size is not required, the actual max size will depend on the configuration of the server – it is set in the php.ini file – the default value is set to 2MB.

2. Create a php script that will process the form and upload the file. The php script will be saved as **upload.php** into the same location as your upload1.html file. This form requires PHP version 4.2.0 or greater.

```
<?php
```

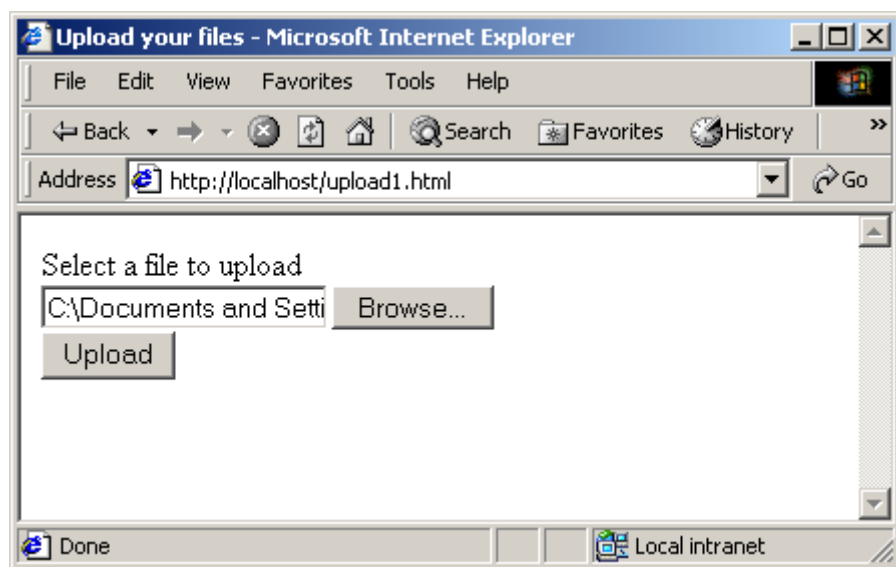
```
if (!(copy($_FILES['username'] ['tmp_name'], "upload/"  
.$_FILES['userfile']['name'])))  
die ("Cannot upload file");  
echo "file upload complete";
```

```
?>
```

Save the file as upload.php. On your server in the same location you uploaded the upload1.html and upload.php file create a new folder and name it **upload**. You uploaded files will be put into this folder.

If the file does not upload, check syntax, check you included `enctype="multipart/form-data"`. Note: the upload feature may not work on your local server (didn't on mine) but works after uploading. If you are using an older version of PHP replace `$_FILES` with `$HTTP_POST_FILES`.

The script basically copies the file "temporarily" uploads the file and calls the file its name. If the server is unable to upload the file you will see an error message "Cannot upload file"; The browse button will work on a local web server, but it won't pass a file to the root folder on your machine. – you must test this script on the Internet in a PHP enabled script. See screen shot below.

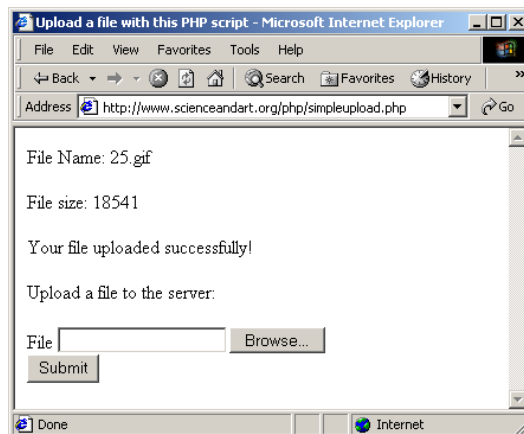


2. In this next script you will create a single PHP file that will create the html for the form and upload the file to a directory called **uploads** on your server – uploads is just another folder you will create in the same location as you new script which we will call **simpleupload.php**.

```
<html>
<head>
<title>Upload a file with this PHP script</title>
</head>
<body>

<?php

if($File) // if file exists
{
    print "File Name: $File_name<p>\n";
    print "File size: $File_size<p>\n";
    if (copy($File, "uploads/$File_name"))
    {
        print "Your file uploaded successfully! <p>\n";
    }
    else
    {
        print "Was unable to upload file <p>\n";
    }
}
unlink ($File); // deletes file
}
print "Upload a file to the server: \n";
print " <form method=\"post\" action=\"simpleupload.php\"
enctype=\"multipart/form-data\"> \n";
print "File <input type=\"file\" name=\"File\" size=20><br> \n";
print "<input type=\"submit\" value=\"Submit\"></form> \n";
?>
```



Comments regarding the upload script :

Name=File is where the \$File variable is derived it is case sensitive

enctype="multipart/form-data" lets HTML know to expect a file or other data

printing the file name and size – these lines are not necessary, but the variable can be pulled from the \$File by appending \$File_name \$File_size

if (copy(\$File, "foldername/\$File_name")) if you remove folder name the file will be uploaded to the same directory where the fileupload.php resides.

For example:

```
if (copy($File, $File_name))
```

note no semicolon follows the if conditional statement

```
print "string";  
print ("string");
```

are all equivalent printf () - the f stands for formatted, it outputs formatted text

Basically this script uses copy() to move the file to a specific location, then you delete the file (from memory) you copied using unlink() function.

E.g. Copy("Source File", "Destination File"):

You will also notice that it is possible to use a single file to process the script and create the HTML page. As forms become more complex the chances of introducing a syntax error (e.g. missing \ escape slash etc) then if you have two separate files – but it is up to the programmer.

The maximum File Size that can be uploaded is determined by 1) server may have a limit set b) PHP can set a limit, view the php.ini file by default it is set to 2 Megabytes and 3) you can set a limit in the script e.g.

```
<input type="hidden" name="MAX_FILE_SIZE" value="2048">
```