PHP - File Uploading

One of the common features required in a typical PHP web application is the provision of letting the user upload files. Uploading files from a client is very easy in PHP. In this chapter, we shall learn how to use PHP script for the file upload process.

The process of uploading a file follows these steps –

- The user opens the page containing a HTML form featuring a text files, a browse button and a submit button.
- The user clicks the browse button and selects a file to upload from the local PC.
- The full path to the selected file appears in the text filed then the user clicks the submit button.
- The selected file is sent to the temporary directory on the server.
- The PHP script that was specified as the form handler in the form's action attribute checks that the file has arrived and then copies the file into an intended directory.
- The PHP script confirms the success to the user.

In order to perform this activity, we must first ensure that configuration settings related to file upload are enabled in "php.ini".

Open the "php.ini" file and ensure that the following settings are enabled by removing the leading semicolon (;) symbol in file_uploads, upload_tmp_dir, upload_max_filesize and max_file_uploads parameters –

```
;;;;;;;;;;;;;;;;
; File Uploads ;
;;;;;;;;;;;;;;;;

; Whether to allow HTTP file uploads.
; http://php.net/file-uploads
file_uploads=On

; Temporary directory for HTTP uploaded files (will use system
; default if not specified).
; http://php.net/upload-tmp-dir
upload_tmp_dir="C:\xampp\tmp"

; Maximum allowed size for uploaded files.
```

```
; http://php.net/upload-max-filesize
upload_max_filesize=40M

; Maximum number of files that can be uploaded via a single request
max_file_uploads=20
```

It is necessary that the folders for both temporary and final locations have permissions set that enable file writing. If either is set to be read-only then process will fail.

Creating a File Upload Form

Next, we need to design a HTML form for file upload. The form's method attribute must be POST and enctype must be multipart/form-data. Use the input type as file to let the user browse and select the file to be uploaded.

</> Open Compiler

```
<h2>File Upload Form</h2>
<form method = "POST" action = "uploadfile.php" enctype="multipart/form-data">
   <label for="file">File name:</label>
   <input type="file" name="uploadfile" />
   <input type="submit" name="submit" value="Upload" />
</form>
```

Creating an Upload Script

The uploadfile.php script receives the uploaded file. The file data is collected in a superglobal variable $_FILES. Fetch the name, file type, size and the tmp_name attributes of the uploaded file.

The move_uploaded_file() function copies the selected file to the document folder.

```
<?php
   echo "<b>File to be uploaded: </b>" . $_FILES["uploadfile"]["name"] . "<br>";
   echo "<b>Type: </b>" . $_FILES["uploadfile"]["type"] . "<br>";
   echo "<b>File Size: </b>" . $_FILES["uploadfile"]["size"]/1024 . "<br>";
   echo "<b>Store in: </b>" . $_FILES["uploadfile"]["tmp_name"] . "<br>";

   if (file_exists($_FILES["uploadfile"]["name"])){
      echo "<h3>The file already exists</h3>";
   } else {
      move_uploaded_file($_FILES["uploadfile"]["tmp_name"], $_FILES["uploadfile"][
```

```
      echo "<h3>File Successfully Uploaded</h3>";
   }
?>
```

Assuming that both the files myform.php and uploadfile.php are stored in the document folder.

Open "myform.php" in the browser **(http://localhost/myform.php)** –

Click the **File** button, browse to the desired file to be uploaded, and click the **Upload** button.

The server responds with the following message –