PHP File Handling

PHP File Handling

- PHP file handling involves different tasks:
  - Creating a file
  - Reading data from a file
  - Updating file content
  - Uploading files

Opening a File with PHP fopen() Function

- The PHP fopen() function is used to open a file. The basic syntax of the fopen() function can be given with:

```
fopen(filename, mode)
```

- The first parameter passed to fopen() specifies the name of the file you want to open.
- The second parameter specifies in which mode the file should be open.

```
<?php
$handle = fopen("data.txt", "r");
?>
```

The file may be opened in one of the following modes:

Modes	What it does
r	Opens the file for reading only.
w	Opens the file for writing only and clears the contents of file. If files does not exist then it attempts to create a file.
a	Append. Opens the file for writing only. Preserves file content by writing to the end of the file. If files does not exist then it attempts to create a file.
x	Create a new file . Returns FALSE and an error if file already exists.

Closing a File with PHP fcolse() Function

- The fclose() function is used to close the file.

```
<html>
    <head>
        <title>File Closing Example</title>
    </head>
    <body>
        <?php
        if (file_exists("data.txt")) {
            $file = fopen("data.txt", "r");
        } else {
            die("Error: The file you are trying to access doesn't exist.");
        }

        // Some code to be executed

        // Closing the file handle
        fclose($file);
        ?>
    </body>
</html>
```

Reading from Files Using the PHP fread() Function

- PHP has several functions for reading data from a file.
- We can read from just one character to the entire file with a single operation.

Reading Strings of Characters

- The fread() function can be used to read a string of characters from a file.
- The basic syntax of this function can be given with.

```
fread(file handle, length in bytes)
```

Example:

```
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php
        $file = "data.txt";
        $handle = fopen($file, "r") or die("ERROR: Cannot open the file");
        $content = fread($handle, "20");
        fclose($handle);
        echo $content;
        ?>
    </body>
</html>
```

Reading an Entire File

- The fread() function uses the filesize() function to read the entire file at once.
- The filesize() function is used to determine the number of characters that should be read in.
- The file_get_contents() function also can be used to read the entire file into a string variable without needing to open it.

Example - UsingThe filesize() function

```
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php
        $file = "data.txt";
        $handle = fopen($file, "r") or die("ERROR: Cannot open the file");
        $content = fread($handle, filesize($file));
        fclose($handle);
        echo $content;
        ?>
    </body>
</html>
```

Example - Using file_get_contents()

```php
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php
        $file = "data.txt";

        // Reading the entire file into a string
        $content = file_get_contents($file) or die("ERROR: Cannot open the file");

        // Display the file content
        echo $content;
        ?>
    </body>
</html>
```

Writing the Files Using PHP

- In PHP, to write data to a file or append to an existing file the PHP fwrite() function is used .
- The basic syntax of this function can be given with:

```
fwrite(file handle, vstring)
```

- The fwrite() function takes two parameter , a file handle and the string of data that is to be written.
- The file_put_contents() function can be used to write data to a file without needing to open it.
- The file_put_contents() function accepts the name of a file together with the data to be written to the file.

Example – Using fwrite()

```
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php
        $file = "data.txt";
        $newData = "The capital city of Libya is Tripoli.";
        $handle = fopen($file, "w") or die("ERROR: Cannot open the file");
        fwrite($handle, $newData) or die("ERROR: Cannot write the file");
        fclose($handle);
        echo "Data written to the file successfully";
        ?>
    </body>
</html>
```

Example – Using file_put_contents()

```
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php
        $file = "data.txt";
        $data = "The capital city of Libya is Tripoli.";
        file_put_contents($file, $data) or die("ERROR: Cannot write the file");
        echo "Data written to the file successfully";
        ?>
    </body>
</html>
```

Example - FILE_APPEND flag

- If the file specified in the file_put_contents() already exists, it will overwirte it by default.
- If you would like to preserve the file's contents you can pass the special FILE_APPEND flag as a third parameter to the file_put_contents() function.

```
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php
        $file = "data.txt";
        $data = "The capital city of Libya is Tripoli.";
        file_put_contents($file, $data, FILE_APPEND) or die("ERROR: Cannot write the file");
        echo "Data written to the file successfully";
        ?>
    </body>
</html>
```

PHP Filesystem Functions

- The following table provides the overview of some other useful PHP filesystem functions that can be used for reading and writing the files dynamically.

Function	Description
fread()	Reads a string of characters from a file.
fwrite()	Writes a string of characters to a file.
fgetc()	Reads a single character at a time.
feof()	Checks to see if the end of the file has been reached.
fgets()	Reads a single line at a time.
fgetcsv()	Reads a line of comma - separated values.
file_get_contents()	Reads an entire file into a string without needing to open it.
file_put_contents()	Writes a whole string to a file without needing to open it.
fseek()	Moves the file pointer to a specific location within an open file.
rewind()	Moves the file pointer to the start of the file.

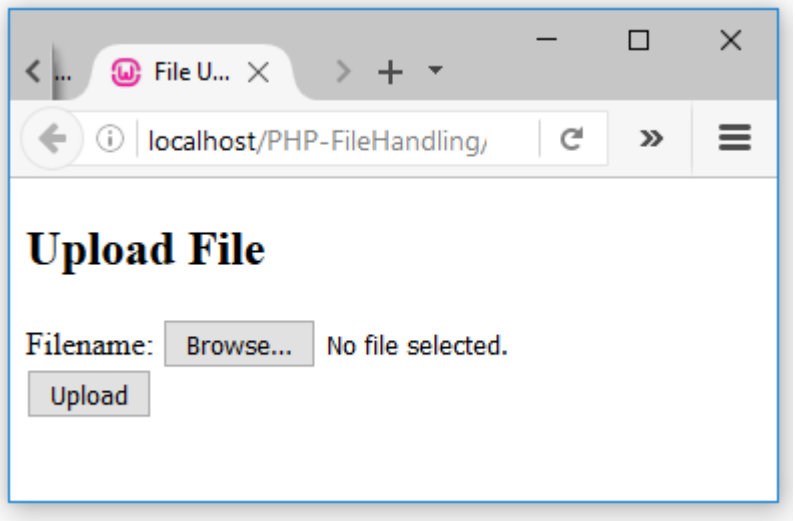
Uploading Files with PHP

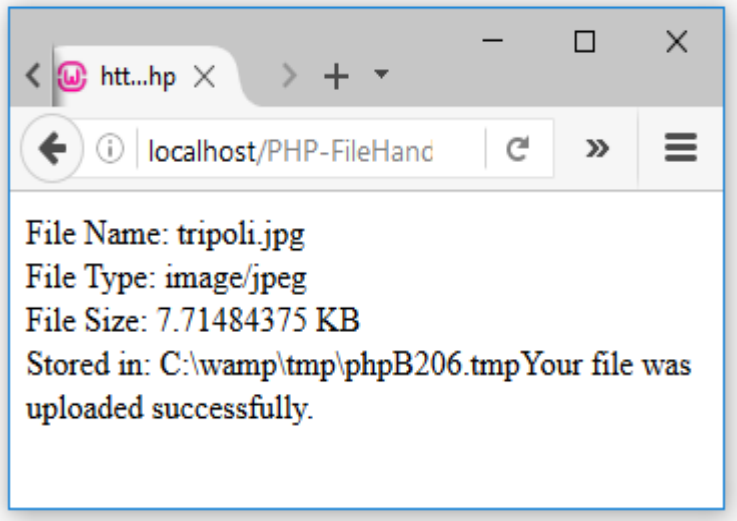
- Using PHP we can upload any kind of file like:
  - images
  - videos
  - Microsoft Office documents
  - PDFs

Example:

- This example uses a simple HTML form to upload files to the server.
  - The form must use the post method.
  - It must contain also an enctype="multipart/form-data" attribute.

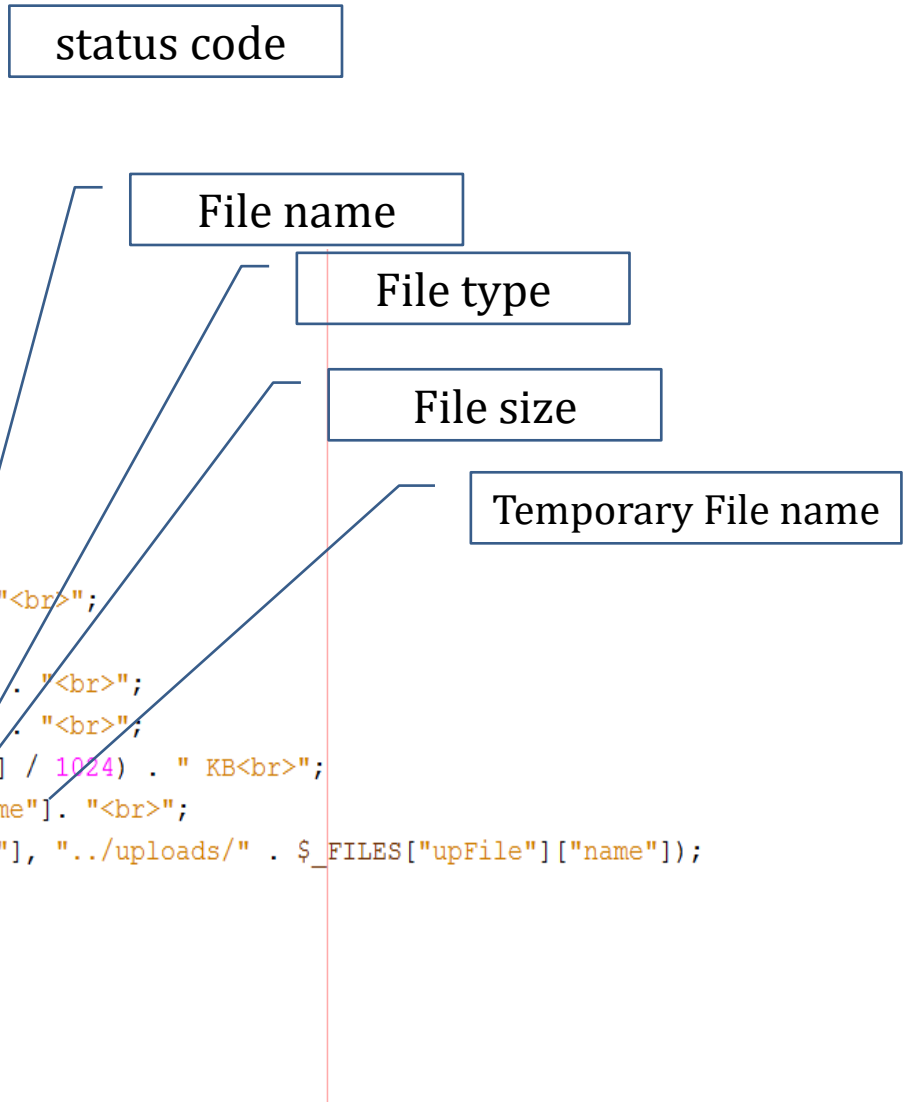
```
<html>
    <head>
        <title>File Upload Form</title>
    </head>
    <body>
        <form action="upload_manager.php" method="post" enctype="multipart/form-data">
            <h2>Upload File</h2>
            Filename:
            <input type="file" name="upFile" ><br>
            <input type="submit" name="submit" value="Upload">
        </form>
    </body>
</html>
```


File U...
localhost/PHP-FileHandling/
Upload File
Filename:
Browse...
No file selected.
Upload


htt...hp
localhost/PHP-FileHand
File Name: tripoli.jpg
File Type: image/jpeg
File Size: 7.71484375 KB
Stored in: C:\wamp\tmp\phpB206.tmpYour file was uploaded successfully.

upload_manager.php

```
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php
        if ($_FILES["upFile"]["error"] > 0) {
            echo "Error: " . $_FILES["upFile"]["error"] . "<br>";
        } else {
            echo "File Name: " . $_FILES["upFile"]["name"] . "<br>";
            echo "File Type: " . $_FILES["upFile"]["type"] . "<br>";
            echo "File Size: " . ($_FILES["upFile"]["size"] / 1024) . " KB<br>";
            echo "Stored in: " . $_FILES["upFile"]["tmp_name"]. "<br>";
            move_uploaded_file($_FILES["upFile"]["tmp_name"], "../uploads/" . $_FILES["upFile"]["name"]);
            echo "Your file was uploaded successfully.";
        }
        ?>
    </body>
</html>
```



Example:

- In this example, we check the file type and file size to ensure that the correct file type and within the allowed limit is uploaded.

```
<html>
    <head>
        <title>image uploading</title>
    </head>
    <body>
        <form action="upload.php" method="post" enctype="multipart/form-data">
            Select image to upload:
            <input type="file" name="upFile" ><br>
            <input type="submit" value="Upload Image" name="submit">
        </form>
    </body>
</html>
```

```php
<html>
        <title></title>
    </head>
    <body>
        <?php
        if ($_FILES["upFile"]["error"] > 0) {
            echo "Error: " . $_FILES["upFile"]["error"] . "<br>";
        } else {
            $filename = $_FILES["upFile"]["name"];
            $filetype = $_FILES["upFile"]["type"];
            $filesize = $_FILES["upFile"]["size"];
            $fileExtention = pathinfo($filename, PATHINFO_EXTENSION);
            // Verify file type
            if ($fileExtention != "jpg" && $fileExtention != "jpeg"
                                && $fileExtention!= "png" && $fileExtention != "gif") {
                die("Error: Please select a valid file format.");
            }
            // Verify file size - 5MB maximum
            $maxsize = 5 * 1024 * 1024;
            if ($filesize > $maxsize) {
                die("Error: File size is larger than the allowed limit.");
            }
            //upload file
            move_uploaded_file($_FILES["upFile"]["tmp_name"], "../uploads/" . $filename );
            echo "Your file was uploaded successfully.";
        }
        ?>
    </body>
</html>
```

Thanks!