File Handling in PHP

Introduction to File Handling

- MySQL is not the only way to store all data on a web server. Sometimes it can be quicker and more convenient to directly access files on the hard disk.

- It can be used to modify images such as uploaded user avatars, or log files that you wish to process.

Creating a File

```
<?php
$fh = fopen(“abc.txt”, ”w”) or die(“Failed to create a File”);

$text = <<<_END
Line 1
Line 2
_END;

fwrite($fh, $text) or die(“Could not write to File”);
fclose($fh);

echo “File abc.txt written successfully”;

?>
```

Continue...

fopen()-: This function is used to open file and it returns a **resource type** value. It contains two parameter **filename** and **mode**.

fwrite()-: This function is used to write the content to the file. It contains 3 parameters **filehandler($fh), text and length**(optional parameter).

fclose()- It is used to close the file and it contains one parameter **filehandler($fh).**

Checking Whether File Exists

Use the **file_exists()** function, this function returns either TRUE or FALSE.

Example:

```
if(file_exists(“abc.txt”)) echo “File exists”;
```

The supported fopen modes

Mode	**Action**	**Description**
‘r’	Read from file start	Open for reading only; place the file pointer at the beginning of the file. Return FALSE if the file doesn’t already exist.
‘r+’	Read from file start and allow writing	Open for reading and writing; place the file pointer at the beginning of the file. Return FALSE if the file doesn’t already exist.
‘w’	Write from file start and truncate file	Open for writing only; place the file pointer at the beginning of the file and truncate the file to **zero** length. If the file doesn’t exist, attempt to create it.
‘w+’	Write from the start, truncate file, and allow reading	Open for reading and writing; place the file pointer at the beginning of the file and truncate the file to zero length. If the file doesn’t exist, attempt to create it.
‘a’	Append to file end	Open for writing only; place the file pointer at the end of the file. If the file doesn’t exist, attempt to create it.
‘a+’	Append to file end and allow reading	Open to reading and writing; place the file pointer at the end of the file. If the file doesn’t exist, attempt to create it.

Reading from Files

Reading a file with fgets

fgets():- this function will return whole line.

```
<?php
        $fh = fopen(“abc.txt”,’r’) or die(“File does not exist or you lack permission to open it”);

        $line = fgets($fh);
        fclose($fh);
        Echo $line;
?>
```

Continue...

fread():- this function can be used to retrieve multiple lines or portions of lines.

```
<?php
$fh = fopen(“abc.txt”,”r”) or die(“File does not exist or you lack permission to open it”);
$text = fread($fh, 3);
fclose($fh);
?>
```

Updating Files

Note:- The *file pointer* is the position within a file at which the next file access will take place, whether it’s read or write.

```
<?php
$fh = fopen(“abc.txt”,”r+”) or die(“Failed to open file”);
$text = fgets($fh);

fseek($fh,0,SEEK_END);
fwrite($fh, $text)  or die(“Could not write to file”);
fclose($fh);
?>
```

Continue...

- **fseek():-** this function contains 3 parameters filehandler($fh), 0 and SEEK_END. **SEEK_END** tells the function to move the file pointer to the **end of the file**, and **0** tells it how many positions it should then be moved backward from that point.

- There are two other seek options available to the fseek function: SEEK_SET and SEEK_CUR.

  - **SEEK_SET-** This option tells the function to set the file pointer to the exact position given by the preceding parameter. Eg. **fseek($fh, 18, SEEK_SET);**

  - **SEEK_CUR-** This option tells the function to set the file pointer to the current position plus the value of the given offset. eg. **fseek($fh, 5, SEEK_SET);**

Uploading Files

```
<?php   //upload.php

echo <<<_END
<html><body>
<form method = ‘post’ action = ‘upload.php’, enctype = ‘multipart/form-data’>
Select File: <input type = ‘file’ name=’abc’ size=’10’>
<input type = ‘submit’ value=’Upload’></form>
_END

if($_FILES)
{
        $name = $_FILES[‘abc’][‘name’];
        move_uploaded_file($_FILES[‘abc’][‘temp_name’], $name);
        Echo “Uploaded image ‘$name’ <br> <img src= ‘$name’>”;

}
?>
```

$_FILES

Five things are stored in the $_FILES array when a file is uploaded.

Array element	**Contents**
$_FILES['file']['name']	The name of the uploaded file(eg. smiley.jpg).
$_FILES['file']['type']	The content type of the file(eg. image/jpeg).
$_FILES['file']['size']	The file's size in bytes.
$_FILES['file']['tmp_name']	The name of the temporary file stored on the server.
$_FILES['file']['error']	The error code resulting from the file upload.