PHP File System

PHP File System allows to create file, read file (line by line or character by character), write file, append file, delete file and close file.

PHP provides pre-defined functions for all these operations:

1. Create a File: fopen()
2. Open a File: fopen()
3. Read a File: fread()
4. Write to a File: fwrite()
5. Append to a File: fwrite()
6. Close a File: fclose()
7. Delete a File: unlink()

Create/Open File:

- fopen() function is used to open/create a file.
- Syntax: fopen(filename, mode)
- Example:

```
<?php
    $filename="data.txt";
    $fp = fopen($filename, "w");
    //$fp = fopen("c:\\folder\\file.txt", "r");
?>
```

Mode	Description
r	Opens file in **read-only** mode. It places the file pointer at the beginning of the file.
r+	Opens file in **read-write** mode. It places the file pointer at the beginning of the file.
w	Opens file in **write-only** mode. It places the file pointer to the beginning of the file and truncates the file to zero length. If file is not found, it creates a new file.
w+	Opens file in **read-write** mode. It places the file pointer to the beginning of the file and truncates the file to zero length. If file is not found, it creates a new file.
a	Opens file in **write-only** mode. It places the file pointer to the end of the file. If file is not found, it creates a new file.

a+	Opens file in **read-write** mode. It places the file pointer to the end of the file. If file is not found, it creates a new file.
x	Creates and opens file in **write-only** mode. It places the file pointer at the beginning of the file. If file is found, fopen() function returns FALSE.
x+	It is same as x but it creates and opens file in **read-write** mode.
c	Opens file in **write-only** mode. If the file does not exist, it is created. If it exists, it is neither truncated (as opposed to 'w'), nor the call to this function fails (as the case with 'x'). The file pointer is positioned on the beginning of the file
c+	It is same as c but it opens file in **read-write** mode.

Close File:

- fclose() function is used to close the file.
- Syntax: fclose(file pointer);
- Example:

```
<?php
    $filename="data.txt";
    $fp = fopen($filename, "w");
    fclose($fp);//close file
?>
```

Read File:

- PHP provides various functions to read data from file.
- There are different functions that allow you to read all file data, read data line by line and read data character by character as follow:
  fread(), fgets(), fgetc()

1. **fread() function:**
   - It is used to read data of the file.
   - It requires two arguments: file pointer and file size.
   - **Syntax: string fread (file pointer , file size)**
     $fp represents file pointer that is created by fopen() function.
     File size represents length of byte to be read.

- **Example:**

```
<?php
    $filename="data.txt";
    $fp = fopen($filename, "r");
    $contents = fread($fp, filesize($filename));//read file
    echo "<pre>$contents</pre>";//printing data of file
    fclose($fp);//close file
?>
```

2. **fgets() function:**
   - It is used to read single line from the file.
   - Syntax: string fgets (file pointer, file size)
   - Example:

```
<?php
    $filename="data.txt";
    $fp = fopen($filename, "r");
    echo fgets($fp);
    fclose($fp);//close file
?>
```

3. **fgetc() function:**
   - It is used to read single character from the file.
   - To get all data using fgetc() function, use !feof() function inside the while loop.
   - Syntax: string fgetc (file pointer)
   - Example:

```
<?php
    $filename="data.txt";
    $fp = fopen($filename, "r");
    while(!feof($fp)) {
       echo fgetc($fp);
      }
```

```
        fclose($fp);//close file
    ?>
```

Write File:

- fwrite() and fputs() functions are used to write data into file.
- To write data into file, you need to use w, r+, w+, x, x+, c or c+ mode.

1. **fwrite():**
    - It is used to write content of the string into file.
    - Syntax: fwrite (file pointer , content, file size)
    - Example:

```
<?php
    $filename="data.txt";
    $fp = fopen($filename, "w");
    fwrite($fp,"WT content");
    fclose($fp);//close file
    echo "File written successfully";
?>
```

Append to File:

- To append data into file by using a or a+ mode in fopen() function.
- fwrite() function is used to write and append data into file.
- Example:

```
<?php
      $filename="data.txt";
      $fp = fopen($filename, "a");
      fwrite($fp,"\nLearning PHP");
      fclose($fp);//close file
      echo "Data appended successfully";
?>
```

Delete File:

- unlink() function is used to delete any file.
- The unlink() function accepts one argument only: file name.
- Syntax: unlink (filename)
- unlink() function generates E_WARNING level error if file is not deleted. It returns TRUE if file is deleted successfully otherwise FALSE.
- Example:

```
<?php
    $filename="data.txt";
    if(unlink($filename))
    {
       echo "File deleted successfully";
    }
    else{
       echo "Sorry!! File is used by some software";
    }
?>
```

More Filesystem Functions:

Function	**Description**
fgetc()	Reads a single character at a time.
fgets()	Reads a single line at a time.
fgetcsv()	Reads a line of comma-separated values.
filetype()	Returns the type of the file.
feof()	Checks whether the end of the file has been reached.
is_file()	Checks whether the file is a regular file.
is_dir()	Checks whether the file is a directory.
is_executable()	Checks whether the file is executable.

Function	Description
realpath()	Returns canonicalized absolute pathname.
rmdir()	Removes an empty directory.
filesize()	Returns the size of the file in bytes
file_exists()	Checks whether a file or directory exists or not.

File Upload:

- PHP allows you to upload single and multiple files through few lines of code only.
- It allows to upload binary and text files both.

$_FILES:

- The PHP global $_FILES contains all the information of file like file name, file type, file size, temp file name and errors associated with file.
- $_FILES['filename']['name'] : returns file name.
- $_FILES['filename']['type']: returns MIME type of the file.
- $_FILES['filename']['size']: returns size of the file (in bytes).
- $_FILES['filename']['tmp_name']: returns temporary file name of the file which was stored on the server.
- $_FILES['filename']['error']: returns error code associated with this file.

move_uploaded_file() function:

- The move_uploaded_file() function moves the uploaded file to a new location.
- The move_uploaded_file() function checks internally if the file is uploaded thorough the POST request. It moves the file if it is uploaded through the POST request.

- Syntax: move_uploaded_file (filename , destination)
- Return type is Boolean.
- Example:

File1.html

```
<form action="file1.php" method="post"
enctype="multipart/form-data">
   Select File:
   <input type="file" name="fileToUpload"/>
   <input type="submit" value="Upload Image" name="submit"/>
</form>
```

File1.php

```
<?php
$target_path = "c:/wamp/www/php/documents/";
$target_path1 = $target_path.$_FILES['fileToUpload']['name'];

if(move_uploaded_file($_FILES['fileToUpload']['tmp_name'],
$target_path1)) {
   echo "File uploaded successfully!";
} else{
   echo "Sorry, file not uploaded, please try again!";
}
?>
```

Rename file:

- It is used to change name of file or directory.
- Syntax: rename(old file, new file)

```
<?php
$target_path = "c:/wamp/www/php/documents/";
$target_path1 = $target_path.$_FILES['fileToUpload']['name'];

if(rename($target_path1,$target_path."pso1.docx"))
```

```
{
   echo "Success";
}
else
{
   echo "Fail";
}
?>
```

Download File:

- It enables to download file easily using built-in readfile() function.
- The readfile() function reads a file.
- It returns the number of bytes read from the file and writes it to the output buffer.
- It is used to download any file from the current location.

- **Example: Download text file**

```
<?php
    $file_url = 'c:/wamp/www/php/documents/test.txt';
    header('Content-Type: application/octet-stream');
    header("Content-Transfer-Encoding: utf-8");
    header("Content-disposition:   attachment;   filename=\""   .
    basename($file_url) . "\"");
    readfile($file_url);
?>
```