**Opening & Closing Files**

The ability to manipulate files can be both fun and useful. The first step of the process would be to identify the file that you want to manipulate, and open it.

PHP uses the fopen() function for the purpose of opening an existing file or creating a new file if the specified file does not already exist. The function syntax is: fopen(filename, method);

It is probably obvious that "filename" refers to the name (and also the path) of the file, but method may not be quite as obvious. It refers to the type of access that is allowed each time the file is opened. The most common options are:

|  |  |
| --- | --- |
| **Mode** | **Description** |
| **r** | Open for reading only; place the file pointer at the beginning of the file. |
| **r+** | Open for reading and writing; place the file pointer at the beginning of the file. |
| **w** | Open for writing only; place the file pointer at the beginning of the file and truncate the file to zero length. If the file does not exist, attempt to create it. |
| **w+** | 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 does not exist, attempt to create it. |
| **a** | Open for writing only; place the file pointer at the end of the file. If the file does not exist, attempt to create it. |
| **a+** | Open for reading and writing; place the file pointer at the end of the file. If the file does not exist, attempt to create it. |
| **x** | Create and open for writing only; place the file pointer at the beginning of the file. If the file already exists, the fopen() call will fail by returning FALSE and generating an error of level E\_WARNING. If the file does not exist, attempt to create it. |
| **x+** | Create and open for reading and writing; otherwise it has the same behavior as 'x'. |
| **c** | Open the file for writing only. 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 is the case with 'x'). The file pointer is positioned on the beginning of the file. |
| **c+** | Open the file for reading and writing; otherwise it has the same behavior as 'c'. |

When using fopen(), a file handle is returned which identifies the open file connection and will later be used to read/write/close the file, all very important steps. We can store that file handle in a variable. Let's look at some examples of how to open a file.

<?php

Error\_reporting(E\_ALL ^ E\_NOTICE);

 $FileHandle1 = fopen("storage.txt", 'w');

 echo $FileHandle1 . "<br>";

 $FileName = "demo.txt";

 $FileHandle3 = fopen($FileName, "r");

 echo $FileHandle3. "<br>";

 $FileHandle2 = fopen("http://www.your-website.com/", 'a+') or die("Be careful Errors!");

?>

So as you can see, you can mix and match your options as needed to get the job done.

The most important part of opening a file is remembering to close it when you are done editing. The fclose() function accepts one parameter, that parameter being the file handle of the open file that needs closed.

<?php
  $FileHandle1 = **fopen**('storage.txt', 'w');
  fclose($FileHandle1);

  $FileHandle2 = fopen('http://www.your-website.com/', 'a+') or die("Error can not be open!");
  **fclose**($FileHandle2);

  $FileName = "/directory-path-to-file/filename.php";
  $FileHandle3 = fopen($FileName, 'r');
  fclose($FileHandle3);
?>

If you are getting errors when trying to open/edit/close files, it might be because PHP does not have permission to edit files on the server. You will need write permission enabled in the directory that the file is stored in.



**Ανάγνωση Αρχείων**

In order to read a file, the file must first be opened with the appropriate mode to allow you to perform the function(s) that you have planned.

Three different functions are useful for reading files.

|  |  |
| --- | --- |
| Function | Description |
| feof() | Tests For End-of-File On a File Pointer |
| fgets() | Gets Line From File Pointer |
| fread() | Binary-Safe File Read |
| filesize() | Gets the Size Of a Given File |

The end-of-file test function, feof(), is commonly used along with the fgets() function to get (read) the entire contents of a file, one line at a time. The filesize() function is commonly used along with the fread() function to read an entire file, otherwise fread() requires that a length/limit be specified and will stop reading the file once that length (number of bytes) has been reached.

Let's look at some examples before we continue.

<?php
  $FileName = "demo1.txt";
  $FileHandle = fopen($FileName, 'r') or die("File Cannot Be Opened");
  $FileData = fread($FileHandle, filesize($FileName));
  fclose($FileHandle);
?>

The above example opens a file and stores the entire contents of the file in a variable before closing the file. The variable containing the file's contents can then be used as needed.

**Εκτύπωση ανά γραμμή του txt αρχείου μας (βρίσκεται στο htdocs)**

<?php

Error\_reporting(E\_ALL ^ E\_NOTICE);

 $FileName = "demo.txt";

 $fp = fopen($FileName, "r") or die("File Cannot Be Opened");

 while (!**feof**($fp)) {

 $line=**fgets**($fp,1024);

 echo $line . "<br>";

 }

**Ανάγνωση κι εκτύπωση τυχαίου ποσού δεδομένων από ένα file με την συνάρτηση fread()**

<?php

Error\_reporting(E\_ALL ^ E\_NOTICE);

 $FileName = "demo.txt";

 $fp = fopen($FileName, "r") or die("File Cannot Be Opened");

 while (!**feof**($fp)) {

 $komati=**fread**($fp,8);

 echo $komati . "<br>";

 }

**Μετακίνηση στα περιεχόμενα ενός αρχείου με την συνάρτηση fseek()**

<?php

Error\_reporting(E\_ALL ^ E\_NOTICE);

 $FileName = "demo2.txt";

 $fp = fopen($FileName, "r") or die("File Cannot Be Opened");

 $fsize=filesize("demo2.txt");

 echo $fsize."<br>";

 $half= (int) ($fsize/2);

 echo "Το μισό κομμάτι είναι : " .$half. "<br>";

 fseek($fp, $half);

 $komati= fread($fp, ($fsize-$half));

 echo $komati . "<br>";

?>

**Εκτύπωση ανά χαρακτήρα των περιεχομένων ενός αρχείου**

<?php

Error\_reporting(E\_ALL ^ E\_NOTICE);

 $FileName = "demo.txt";

 $fp = fopen($FileName, "r") or die("File Cannot Be Opened");

 while (!feof($fp)) {

 $char= fgetc($fp);

 echo $char . "<br>";

 }

**Εγγραφή ή προσθήκη δεδομένων σε ένα Αρχείο με fwrite() ή fputs()**

 $fp = fopen($FileName, "w") για εγγραφή (τα προηγούμενα περιεχόμενά του κατστρέφονται )

 $fp = fopen($FileName, "a") για προσθήκη

<?php

Error\_reporting(E\_ALL ^ E\_NOTICE);

 $filename = "demo3.txt";

 echo "ΘΑ ΚΑΝΟΥΜΕ ΕΓΓΡΑΦΗ ΣΤΟ ΑΡΧΕΙΟ " . $filename . "<br>";

 $fp = fopen($filename, "w") or die("File Cannot Be Opened");

 fwrite($fp, "ΔΙΩΞΤΕ ΡΕ ΤΟΝ ΠΡΟΠΟΝΗΤΗ ΣΤΟΝ ΠΑΟΚ\n");

 fclose($fp);

 echo " "."<br>"; // ΚΕΝΗ ΓΡΑΜΜΗ

 echo "ΘΑ ΚΑΝΟΥΜΕ ΠΡΟΣΘΗΚΗ ΣΤΟ " . $filename . "<br>";

 $fp = fopen($filename, "a") or die("File Cannot Be Opened");

 fputs($fp, "ΚΑΙ ΠΑΡΤΕ ΤΟΝ ΜΟΥΡΙΝΙΟ")."<br>";

 fclose($fp);

 echo " "."<br>"; // ΚΕΝΗ ΓΡΑΜΜΗ

 $fileName = "demo3.txt";

 $fp = fopen($fileName, "r") or die("File Cannot Be Opened");

 while (!feof($fp)) {

 $line=fgets($fp,1024);

 echo $line . "<br>";

 }

?>

**Deleting Files**

Files can be deleting using the unlink() function. As will all functions that remove data, make double sure that you really want the data gone for good, and that you don't delete the wrong data!

To lessen confusion, make sure that your file is closed before you try to delete/unlink it. Then, a single line of code with do the trick:

<?php
  unlink("storage.txt");

?>

It's as easy as that!