HANDLING THE FILESYSTEM WITH PHP

IN PHP WE HAVE A LOT OF FUNCTIONS TO HANDLE THE FILESYSTEM, BESIDES THE ONES WE HAVE SEEN TO WORK WITH SPECIFIC FILES.

SOME EXAMPLES FOR WORKING WITH DIRECTORIES:

- `chdir(string $directory): bool` – changes directory
- `mkdir(string $directory, int $permissions, bool $recursive);`
  creates a directory with the specifed permissions. $recursive must be true if we are trying to create a nested structure of directories
- `rmdir(string $directory);` – removes the directory
- `getcwd(): string` – returns the current directory
- `scandir(string $directory...): array`
  – returns an array with the files and directories of the specified path

HANDLING THE FILESYSTEM WITH PHP

OTHER USEFUL FUNCTIONS TO HANDLE THE FILESYSTEM:

- **`chmod, chown`** – to change permissions and owners
- **`copy, delete`** – to copy and delete files
- **`disk_free_space, disk_total_space`** – info about disk space
- **`is_dir, is_executable,is_file, is_link, is_readable, is_writable`** – to test the type of a file

More information about filesystem functions following the next link:

https://www.php.net/manual/en/book.filesystem.php

UPLOAD FILES

The <input> element with the type="file" allows you to select one or more files from their storage and upload them to the server via the form submission.

The following shows the file input element:

```
<input type="file" id="file" name="file">
```

The value of the <input> element will hold the path to the selected file.

To upload multiple files, you add the multiple attribute to the <input> element like this:

```
<input type="file" id="file" name="file" multiple>
```

The <form> element that contains the file input element must have the enctype attribute with the value multipart/form-data:

```
<form enctype="multipart/form-data" action="index.php" method="post">

</form>
```

To access the information of an uploaded file, you use the $_FILES array.

UPLOAD FILES

For example, if the name of the file input element is file, you can access the uploaded file via $_FILES['file'].

The $_FILES[‘file’] is an associative array that consists of the following keys:

NAME	Description
name	is the name of the uploaded file
tmp_name	is the temporary file on the server that stored the uploaded filename. If the uploaded file is too large, the tmp_name is "none".
Type	is the MIME type of the upload file e.g., image/jpeg for JPEG image or application/pdf for PDF file.
error	is the error code that describes the upload status e.g., UPLOAD_ERR_OK means the file was uploaded successfully
size	is the size of the uploaded file in bytes.

UPLOAD FILES

If you want to get a message based on an error code, you can simply look it up in the MESSAGES array like this:

```
$message = MESSAGES[$_FILES['file']['error']];
```

Value	Constant	Meaning
0	UPLOAD_ERR_OK	No errors
1	UPLOAD_ERR_INI_SIZE	File is too big to upload: directive upload_max_filesize in php.ini
2	UPLOAD_ERR_FORM_SIZE	File is too big to upload hidden field max_file_size
3	UPLOAD_ERR_PARTIAL	File was only partially uploaded
4	UPLOAD_ERR_NO_FILE	No file was uploaded

UPLOAD FILES

Limit file size:

- **In php.ini: The upload_max_filesize specifies the maximum size of the uploaded file.**
- **If you place a field with the name MAX_FILE_SIZE before a file input element in the form, PHP will use that value instead of upload_max_filesize for validating the file size.**

```
<form enctype="multipart/form-data" action="upload.php" method="post">
    <div>
        <label for="file">Select a file:</label>
        <input type="hidden" name="MAX_FILE_SIZE" value="10240"/>
        <input type="file" id="file" name="file"/>
    </div>
    <div>
        <button type="submit">Upload</button>
    </div>
</form>
```

When a file is uploaded successfully, it is stored in a temporary directory on the server (tmp_name). And you can use the move_uploaded_file() or copy() function to move the file from the temporary directory to another one.

```
move_uploaded_file($_FILES['file']['tmp_name'], $path.$filename)
copy($_FILES['file']['tmp_name'],$_FILES['file']['name']);
```

UPLOAD FILES EXAMPLE

```
<form action="mypage.php" method="post" enctype="multipart/form-data">
<input name="myfile" type="file">
<input type="submit" value="Upload">
</form>
```

The uploaded file is stored in the $_FILES array. To upload the file with move_uploaded_file

```
// IN linux OS
$dest = 'upload\\' . basename($_FILES['myfile']['name']);
// IN Windows OS  'img/'.$_FILES['foto']['name']
$dest = 'upload/' . basename($_FILES['myfile']['name']);

$file = $_FILES['myfile']['tmp_name'];
$err = $_FILES['myfile']['error']; //checks the error code
if($err == 0 && move_uploaded_file($file, $dest)) //moves the file out of the
//temporary folder to the path and with the name specified by the second
//argument.
echo 'File successfully uploaded';
```

Basename: It returns the file name component of a path, including the file extension

You can save the name of the file in a field of a table: photo varchar(50) when you do an INSERT-You will practice it in an exercise

FILE UPLOADING: SECURITY CONCERNS

- It's a good practice to limit the size: upload_max_filesize = 2M
- Also,check if is_uploaded_file ($_FILES['imagen']['tmp_name']): Tells whether the file was uploaded via HTTP POST. Prevents a user from trying to read files not loaded by POST. e.g. / etc / passwd

```
<?php
if (is_uploaded_file($_FILES['userfile']['tmp_name'])) {
   echo "File ". $_FILES['userfile']['name'] ." uploaded successfully.\n";
   echo "Displaying contents\n";
   readfile($_FILES['userfile']['tmp_name']);
} else {
   echo "Possible file upload attack: ";
   echo "filename '". $_FILES['userfile']['tmp_name'] . "'.";
}
?>
```

FPDF LIBRARY

We have some libraries to generate a pdf: htmltopdf, tcpdf, dompdf, fpdf.

Let’s see FPDF libray: is a PHP class which allows to generate PDF files with pure PHP

http://www.fpdf.org/: Download the library. You have all the methods, in “manual“ section

Basic example:

```php
<?php
include(__DIR__.'/fpdf/fpdf.php');
$pdf = new FPDF();
// There's no page at the moment, so we have to add one with  AddPage().
$pdf->AddPage();
//Before we can print text, it's mandatory to select a font
$pdf->SetFont('Arial','B',16);
/*A cell is a rectangular area, possibly framed, which contains a line of text. It is
output at the current position. We specify its dimensions*/
$pdf->Cell(40,10,'Hello World!');
// Finally, the document is closed and sent to the browser with Output().
$pdf->Output();
?>
```

You can save the file with output:

```php
$pdf->Output('text.pdf','F');
```

SEND MAIL FROM PHP

In order to work, it must be in a server computer with Apache + mail service.

We use the mail function. To send an email to the teacher.

```
mail("m.gimenezgomez@edu.gva.es","subject","How are you?") ;

//for sending in HTML format
$headers = "MIME-Version: 1.0\r\n";
$headers .= "Content-type: text/html; charset=iso-8859-1\r\n";

//sender address
$headers .= "From: studentpepe <pepestudent@alu.gva.es>\r\n";

//reply address, if we want it to be different from the sender
$headers .= "Reply-To: pepe_personal@gmail.com\r\n";

//Copy addresses. Several: separated by commas
$headers .= "Cc: enrique@edu.gva.es.com\r\n";

mail("m.gimenezgomez@edu.gva.es","subject","How are you?",$headers) ;
```