

advance

PHP Hypertext Preprocessor language

By Kouros

Include Files

- The include (or require) statement takes all the text/code/markup that exists in the specified file and copies it into the file that uses the include statement.
- Including files is very useful when you want to include the same PHP, HTML, or text on multiple pages of a website.

```
include 'filename';
```

or

```
require 'filename';
```

Example of include

```
<!DOCTYPE html>
<html>
<body>

<div class="menu">
<?php include 'menu.php';?>
</div>

<h1>Welcome to my home page!</h1>
<p>Some text.</p>
<p>Some more text.</p>

</body>
</html>
```

PHP include vs. require

- The require statement is also used to include a file into the PHP code.
- However, there is one big difference between include and require; when a file is included with the **include** statement and PHP cannot find it, the script will continue to execute.
- If we do the same example using the **require** statement, the echo statement will not be executed because the script execution dies after the require statement returned a fatal error.

File and file handling

Simple syntax to read file in php

```
<?php  
echo readfile("webdictionary.txt");  
?>
```

File Open/Read/Close

```
<?php
$myfile = fopen("webdictionary.txt", "r") or
die("Unable to open file!");
echo fread($myfile,filesize("webdictionary.txt"));
fclose($myfile);
?>
```

r Open a file for read only. File pointer starts at the beginning of the file

w Open a file for write only. Erases the contents of the file or creates a new file if it doesn't exist. File pointer starts at the beginning of the file

Create and write on File

```
// create file
```

```
$myfile = fopen("testfile.txt", "w")
```

```
// create and write on file
```

```
<?php  
$myfile = fopen("newfile.txt", "w") or die("Unable to open file!");  
$txt = "John Doe\n";  
fwrite($myfile, $txt);  
$txt = "Jane Doe\n";  
fwrite($myfile, $txt);  
fclose($myfile);  
?>
```


Forms and Files

Create The HTML Form

```
<!DOCTYPE html>
<html>
<body>

<form action="upload.php" method="post" enctype="multipart/form-data">
  Select image to upload:
  <input type="file" name="fileToUpload" id="fileToUpload">
  <input type="submit" value="Upload Image" name="submit">
</form>

</body>
</html>
```

The type="file" attribute of the <input> tag shows the input field as a file-select control, with a "Browse" button next to the input control

Create The Upload File PHP Script

```
<?php
$target_dir = "uploads/";
$target_file = $target_dir . basename($_FILES["fileToUpload"]["name"]);
$uploadOk = 1;
$imageFileType = pathinfo($target_file,PATHINFO_EXTENSION);
// Check if image file is a actual image or fake image
if(isset($_POST["submit"])) {
    $check = getimagesize($_FILES["fileToUpload"]["tmp_name"]);
    if($check !== false) {
        echo "File is an image - " . $check["mime"] . ".";
        $uploadOk = 1;
    } else {
        echo "File is not an image.";
        $uploadOk = 0;
    }
}
?>
```

PHP script explained:

`$target_dir = "uploads/"` - specifies the directory where the file is going to be placed
`$target_file` specifies the path of the file to be uploaded
`$uploadOk=1` is not used yet (will be used later)
`$imageFileType` holds the file extension of the file
Next, check if the image file is an actual image or a fake image

Check if File Already Exists

```
// Check if file already exists
if (file_exists($target_file)) {
    echo "Sorry, file already exists.";
    $uploadOk = 0;
}
```

Limit File Size

```
// Check file size  
if ($_FILES["fileToUpload"]["size"] > 500000) {  
    echo "Sorry, your file is too large.";  
    $uploadOk = 0;  
}  
  
//If the file is larger than 500KB, an error message is  
displayed
```

Limit File Type

```
// Allow certain file formats
if($imageFileType != "jpg" && $imageFileType != "png"
&& $imageFileType != "jpeg"
&& $imageFileType != "gif" ) {
    echo "Sorry, only JPG, JPEG, PNG & GIF files are
allowed.";
    $uploadOk = 0;
}
```

//The code only allows users to upload JPG, JPEG, PNG, and GIF files. All other file types gives an error message

Complete Upload File PHP Script (part 1)

```
<?php
$target_dir = "uploads/";
$target_file = $target_dir . basename($_FILES["fileToUpload"]["name"]);
$uploadOk = 1;
$imageFileType = pathinfo($target_file,PATHINFO_EXTENSION);
// Check if image file is a actual image or fake image
if(isset($_POST["submit"])) {
    $check = getimagesize($_FILES["fileToUpload"]["tmp_name"]);
    if($check !== false) {
        echo "File is an image - " . $check["mime"] . ".";
        $uploadOk = 1;
    } else {
        echo "File is not an image.";
        $uploadOk = 0;
    }
}
```

Complete Upload File PHP Script (part 2)

```
// Check if file already exists
if (file_exists($target_file)) {
    echo "Sorry, file already exists.";
    $uploadOk = 0;
}
// Check file size
if ($_FILES["fileToUpload"]["size"] > 500000) {
    echo "Sorry, your file is too large.";
    $uploadOk = 0;
}
// Allow certain file formats
if($imageFileType != "jpg" && $imageFileType != "png" && $imageFileType !=
"jpeg"
&& $imageFileType != "gif" ) {
    echo "Sorry, only JPG, JPEG, PNG & GIF files are allowed.";
    $uploadOk = 0;
}
```


Complete Upload File PHP Script (part 3)

```
// Check if $uploadOk is set to 0 by an error
if ($uploadOk == 0) {
    echo "Sorry, your file was not uploaded.";
// if everything is ok, try to upload file
} else {
    if
(move_uploaded_file($_FILES["fileToUpload"]["tmp_name"],
$_target_file)) {
        echo "The file ". basename(
$_FILES["fileToUpload"]["name"]). " has been uploaded.";
    } else {
        echo "Sorry, there was an error uploading your file.";
    }
}
?>
```

PHP Images from Folder

Glob ()

- The glob() function returns an array of filenames or directories matching a specified pattern.
- This function returns an array of files/directories, or FALSE on failure.

```
<?php  
print_r(glob("*.txt"));  
?>
```

```
// output:  
Array  
(  
[0] => target.txt  
[1] => source.txt  
[2] => test.txt  
[3] => test2.txt  
)
```

Count ()

- The count() function returns the number of elements in an array.

```
<?php
$cars=array("Volvo","BMW","Toyota");
echo count($cars);

// it returns 3
?>
```

Opendir () & Readdir ()

- Opendir is function to open a directory
- Readdir is function to read files in a directory that is open

```
<?php
$dir = "/images/";

// Open a directory, and read its contents
if (is_dir($dir)){
    if ($dh = opendir($dir)){
        while (($file = readdir($dh)) !== false){
            echo "filename:" . $file . "<br>";
        }
        closedir($dh);
    }
}
?>
```

PHP pathinfo() Function

The pathinfo() function returns an array that contains information about a path.

The following array elements are returned:

- [dirname]
- [basename]
- [extension]

```
<?php
print_r(pathinfo("/testweb/test.txt"
));
```

```
// output:
Array
(
    [dirname] => /testweb
    [basename] => test.txt
    [extension] => txt
)
```

The print_r() function is used to print human-readable information about a variable. So it is similar to echo 😊

in_array ()

- It searches for a value in an array

```
<?php
$people = array("Peter", "Joe", "Glenn", "Cleveland");

if (in_array("Glenn", $people))
{
    echo "Match found";
}
else
{
    echo "Match not found";
}
?>
```

```
<?php
$spath = "images/"; // path to images folder
$file_count = count(glob($spath . "*.{png,jpg,jpeg,gif}", GLOB_BRACE));
if($file_count > 0)
{
    $fp = opendir($spath);
    while($file = readdir($fp))
    {
        $sxt = pathinfo($file, PATHINFO_EXTENSION);
        $sxt_array = ['png', 'jpg', 'jpeg', 'gif'];
        if (in_array($sxt, $sxt_array))
        {
            $file_path = $spath . $file;?>
            <div class="col-md-4 col-xs-6">
                <a href="<?php echo $file_path; ?>" title="My Favorites" data-gallery></a>
            </div>
        }
    }
    closedir($fp);
}
?>
```


PHP Error Handling

Error Handling

When creating scripts and web applications, error handling is an important part. If your code lacks error checking code, your program may look very unprofessional and you may be open to security risks.

This tutorial contains some of the most common error checking methods in PHP.

We will show different error handling methods:

- Simple "die()" statements
- Custom errors and error triggers
- Error reporting

Example of Error Handling

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

Warning: fopen(welcome.txt) [function.fopen]: failed to open stream:
No such file or directory in C:\webfolder\test.php on line 2

```
<?php
if(!file_exists("welcome.txt")) {
    die("File not found");
} else {
    $file=fopen("welcome.txt","r");
}
?>
```

Exception handling

Exception handling is used to change the normal flow of the code execution if a specified error (exceptional) condition occurs. This condition is called an exception.

This is what normally happens when an exception is triggered:

- The current code state is saved
- The code execution will switch to a predefined (custom) exception handler function
- Depending on the situation, the handler may then resume the execution from the saved code state, terminate the script execution or continue the script from a different location in the code

Exception handling example

- When an exception is thrown, the code following it will not be executed, and PHP will try to find the matching "catch" block.
- If an exception is not caught, a fatal error will be issued with an "Uncaught Exception" message.

```
<?php
//create function with an exception
function checkNum($number) {
    if($number>1) {
        throw new Exception("Value must be 1 or below");
    }
    return true;
}
//trigger exception
checkNum(2);
?>
```

```
Fatal error: Uncaught exception
'Exception'
with message 'Value must be 1 or
below' in
C:\webfolder\test.php:6
Stack trace: #0
C:\webfolder\test.php(12):
checkNum(28) #1 {main} thrown
in C:\webfolder\test.php on line
```

Example of Try, throw and catch

```
<?php
//create function with an exception
function checkNum($number) {
    if($number>1) {
        throw new Exception("Value must be 1 or below");
    }
    return true;
}

//trigger exception in a "try" block
try {
    checkNum(2);
    //If the exception is thrown, this text will not be shown
    echo 'If you see this, the number is 1 or below';
}

//catch exception
catch(Exception $e) {
    echo 'Message: ' . $e->getMessage();
}
?>
```

Output will be:
Message: Value must be 1 or