Continue to Advance PHP

5.1 PHP 5 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.

PHP include and require Statements

It is possible to insert the content of one PHP file into another PHP file (before the server executes it), with the include or require statement. The include and require statements are identical, except upon failure:

require will produce a fatal error (E_COMPILE_ERROR) and stop the script include will only produce a warning (E_WARNING) and the script will continue So, if you want the execution to go on and show users the output, even if the include file is missing, use the include statement. Otherwise, in case of FrameWork, CMS, or a complex PHP application coding, always use the require statement to include a key file to the flow of execution. This will help avoid compromising your application's security and integrity, just in-case one key file is accidentally missing.

Including files saves a lot of work. This means that you can create a standard header, footer, or menu file for all your web pages. Then, when the header needs to be updated, you can only update the header include file.

Syntax

include '*filename*';

or

require '*filename*';

PHP include Examples

Example 1

Assume we have a standard footer file called "footer.php", that looks like this:

```
<?php
echo "<p>Copyright &copy; 1999-" . date("Y") . " W3Schools.com</p>";
?>
```

To include the footer file in a page, use the include statementexample:

```
<!DOCTYPE html>
<html>
<body>
<h1>Welcome to my home page!</h1>
<p>Some text.</p>
<p>Some more text.</p>
<?php include 'footer.php';?>
</body>
</html>
```

Output:

```
Welcome to my home page!

Some text.

Some more text.

Copyright © 1999-2018 W3Schools.com
```

Example 2

Assume we have a standard menu file called "menu.php":

```
<?php
echo '<a href="/default.asp">Home</a> -
<a href="/html/default.asp">HTML Tutorial</a> -
<a href="/css/default.asp">CSS Tutorial</a> -
<a href="/js/default.asp">JavaScript Tutorial</a> -
<a href="default.asp">PHP Tutorial</a>';
?>
```

All pages in the Web site should use this menu file. Here is how it can be done (we are using a <div> element so that the menu easily can be styled with CSS later) Example:

```
<!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>
```

Output: Home - HTML Tutorial - CSS Tutorial - JavaScript Tutorial - PHP Tutorial

```
Welcome to my home page!

Some text.

Some more text.
```

Example 3 : Assume we have a file called "vars.php", with some variables defined:

```
<?php
$color='red';
$car='BMW';
?>
```

Then, if we include the "vars.php" file, the variables can be used in the calling file Example:

```
<!DOCTYPE html>
<html>
<body>
<h1>Welcome to my home page!</h1>
<?php include 'vars.php';
echo "I have a $color $car.";
?>
```

```
</body>
</html>
```

Output: Welcome to my home page!

I have a red BMW.

5.2 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:

Example:

```
<!DOCTYPE html>
<html>
<body>
<h1>Welcome to my home page!</h1>
<?php include 'noFileExists.php';
echo "I have a $color $car.";
?>
</body>
</html>
```

Output: Welcome to my home page!

I have a .

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:

Example:

```
<!DOCTYPE html>
<html>
<body>
<h1>Welcome to my home page!</h1>
<?php require 'noFileExists.php';
echo "I have a $color $car.";
```

```
?>
</body>
</html>
```

Output: Welcome to my home page!

Use require when the file is required by the application.

Use include when the file is not required and application should continue when file is not found.

6.1 PHP Open File - fopen()

A better method to open files is with the fopen() function. This function gives you more options than the readfile() function.

We will use the text file, "webdictionary.txt", during the lessons:

```
AJAX = Asynchronous JavaScript and XML
CSS = Cascading Style Sheets
HTML = Hyper Text Markup Language
PHP = PHP Hypertext Preprocessor
SQL = Structured Query Language
SVG = Scalable Vector Graphics
XML = EXtensible Markup Language
```

The first parameter of fopen() contains the name of the file to be opened and the second parameter specifies in which mode the file should be opened. The following example also generates a message if the fopen() function is unable to open the specified file:

Example:

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

```
</body>
</html>
```

Output:

```
AJAX = Asynchronous JavaScript and XML CSS = Cascading Style Sheets HTML = Hyper Text
Markup Language PHP = PHP Hypertext Preprocessor SQL = Structured Query Language SVG =
Scalable Vector Graphics XML = EXtensible Markup Language
```

Tip: The fread() and the fclose() functions will be explained below.

The file may be opened in one of the following modes:

Modes	Description
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
a	**Open a file for write only**. The existing data in file is preserved. File pointer starts at the end of the file. Creates a new file if the file doesn't exist
x	**Creates a new file for write only**. Returns FALSE and an error if file already exists
r+	**Open a file for read/write**. File pointer starts at the beginning of the file
w+	**Open a file for read/write**. 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
a+	**Open a file for read/write**. The existing data in file is preserved. File pointer starts at the end of the file. Creates a new file if the file doesn't exist
x+	**Creates a new file for read/write**. Returns FALSE and an error if file already exists

6.2 PHP Read File - fread()

The fread() function reads from an open file. The first parameter of fread() contains the name of the file to read from and the second parameter specifies the maximum number of bytes to read.The following PHP code reads the "webdictionary.txt" file to the end:

```
fread($myfile,filesize("webdictionary.txt"));
```

6.2 PHP Close File - fclose()

The fclose() function is used to close an open file. It's a good programming practice to close all files after you have finished with them. You don't want an open file running around on your server taking up resources!

The fclose() requires the name of the file (or a variable that holds the filename) we want to close:

```
<?php
$myfile = fopen("webdictionary.txt", "r");
// some code to be executed....
fclose($myfile);
?>
```

6.3 PHP Read Single Line - fgets()

The fgets() function is used to read a single line from a file.

The example below outputs the first line of the "webdictionary.txt" file example:

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

```
</body>
</html>
```

Output: AJAX = Asynchronous JavaScript and XML

Note: After a call to the fgets() function, the file pointer has moved to the next line.