# PHP Include and Require Files

## Including a PHP File into Another PHP File: The include() and require() statement allow you to include the code contained in a PHP file within another PHP file. Including a file produces the same result as copying the script from the file specified and pasted into the location where it is called.

You can save a lot of time and work through including files — Just store a block of code in a separate file and include it wherever you want using the include() and require() statements instead of typing the entire block of code multiple times. A typical example is including the header, footer and menu file within all the pages of a website.

The basic syntax of the include() and require() statement can be given with:

include("path/to/filename"); -Or- include "path/to/filename";  
require("path/to/filename"); -Or- require "path/to/filename";

**Note:**  Like the print and echo statements, you can omit the parentheses while using the include and require statements as demonstrated above.

The following example will show you how to include the common header, footer and menu codes which are stored in separate 'header.php', 'footer.php' and 'menu.php' files respectively, within all the pages of your website. Using this technique you can update all pages of the website at once by making the changes to just one file, this saves a lot of repetitive work.

#### Example

<!DOCTYPE html>

<html lang="en">

<head>

<title>Tutorial Republic</title>

</head>

<body>

<?php include "header.php"; ?>

<?php include "menu.php"; ?>

<h1>Welcome to Our Website!</h1>

<p>Here you will find lots of useful information.</p>

<?php include "footer.php"; ?>

</body>

</html>

**Difference Between include and require Statements**

You might be thinking if we can include files using the include() statement then why we need require(). Typically the require() statement operates like include().

The only difference is — the include() statement will only generate a PHP warning but allow script execution to continue if the file to be included can't be found, whereas the require() statement will generate a fatal error and stops the script execution.

Example

<?php require "my\_variables.php"; ?>

<?php require "my\_functions.php"; ?>

<!DOCTYPE html>

<html lang="en">

<head>

<title><?php displayTitle($home\_page); ?></title>

</head>

<body>

<?php include "header.php"; ?>

<?php include "menu.php"; ?>

<h1>Welcome to Our Website!</h1>

<p>Here you will find lots of useful information.</p>

<?php include "footer.php"; ?>

</body>

</html>

**Note:** It is recommended to use the require() statement if you're including the library files or files containing the functions and configuration variables that are essential for running your application, such as [database configuration file](https://www.tutorialrepublic.com/php-tutorial/php-mysql-crud-application.php).

**The include\_once and require\_once Statements**

If you accidentally include the same file (typically [functions](https://www.tutorialrepublic.com/php-tutorial/php-functions.php) or [classes](https://www.tutorialrepublic.com/php-tutorial/php-classes-and-objects.php) files) more than one time within your code using the include or require statements, it may cause conflicts. To prevent this situation, PHP provides include\_once and require\_once statements. These statements behave in the same way as include and require statements with one exception.

The include\_once and require\_once statements will only include the file once even if asked to include it a second time i.e. if the specified file has already been included in a previous statement, the file is not included again. To better understand how it works, let's check out an example. Suppose we've a 'my\_functions.php' file with the following code:

Example

<?php

function multiplySelf($var){

$var \*= $var; // multiply variable by itself

echo $var;

}

?>

Here's is the PHP script within which we've included the 'my\_functions.php' file.

Example

<?php

// Including file

require "my\_functions.php";

// Calling the function

multiplySelf(2); // Output: 4

echo "<br>";

// Including file once again

require "my\_functions.php";

// Calling the function

multiplySelf(5); // Doesn't execute

?>

When you run the above script, you will see the error message something like this: **"Fatal error: Cannot redeclare multiplySelf()"**. This occurs because the 'my\_functions.php' included twice, this means the function multiplySelf() is defined twice, which caused PHP to stop script execution and generate fatal error. Now rewrite the above example with require\_once.

Example

<?php

// Including file

require\_once "my\_functions.php";

// Calling the function

multiplySelf(2); // Output: 4

echo "<br>";

// Including file once again

require\_once "my\_functions.php";

// Calling the function

multiplySelf(5); // Output: 25

?>

As you can see, by using require\_once instead of require, the script works as we expected.