Chapter 4: PHP File Handling

You can include the content of a PHP file into another PHP file before the server executes it. There are two PHP functions which can be used to included one PHP file into another PHP file.

- The include() Function
- The require() Function

This is a strong point of PHP which helps in creating functions, headers, footers, or elements that can be reused on multiple pages. This will help developers to make it easy to change the layout of complete website with minimal effort. If there is any change required then instead of changing thousand of files just change included file.

The include() Function

The include() function takes all the text in a specified file and copies it into the file that uses the include function. If there is any problem in loading a file then the **include()** function generates a warning but the script will continue execution.

Assume you want to create a common menu for your website. Then create a file menu.php with the following content.

```
<a href="http://localhost/index.htm">Home</a> -
<a href="http://localhost/ebxml">ebXML</a> -
<a href="http://localhost/ajax">AJAX</a> -
<a href="http://localhost/perl">PERL</a> <br />
```

Now create as many pages as you like and include this file to create header. For example now your test.php file can have following content.

```
<html>
<body>
<?php include("menu.php"); ?>
<p>This is an example to show how to include PHP file!</p>
</body>
</html>
```

This will produce following result

The require() Function

The require() function takes all the text in a specified file and copies it into the file that uses the include function. If there is any problem in loading a file then the **require()** function generates a fatal error and halt the execution of the script.

So there is no difference in require() and include() except they handle error conditions. It is recommended to use the require() function instead of include(), because scripts should not continue executing if files are missing or misnamed.

You can try using above example with require() function and it will generate same result. But if you will try following two examples where file does not exist then you will get different results.

```
<html>
<body>
<?php include("xxmenu.php"); ?>
<p>This is an example to show how to include wrong PHP file!</p>
</body>
</html>
```

This will produce following result

```
This is an example to show how to include wrong PHP file!
```

Now lets try same example with require() function.

```
<html>
<body>
<?php require("xxmenu.php"); ?>
<p>This is an example to show how to include wrong PHP file!</p>
</body>
</html>
```

This time file execution halts and nothing is displayed.

File handling is an important part of any web application. You often need to open and process a file for different tasks.

File Handling

readfile()

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

This chapter will explain following functions related to files:

- Opening a file
- Reading a file
- Writing a file
- Closing a file

Opening and Closing Files

The PHP **fopen()** function is used to open a file. It requires two arguments stating first the file name and then mode in which to operate.

Files modes can be specified as one of the six options in this table.

Mode	Purpose
r	Opens the file for reading only. Places the file pointer at the beginning of the file.
r+	Opens the file for reading and writing. Places the file pointer at the beginning of the file.
w	Opens the file for writing only. Places the file pointer at the beginning of the file. and truncates the file to zero length. If files does not exist then it attemts to create a file.
w+	Opens the file for reading and writing only. Places the file pointer at the beginning of the file. and truncates the file to zero length. If files does not exist then it attemts to create a file.
a	Opens the file for writing only. Places the file pointer at the end of the file. If files does not exist then it attemts to create a file.
a+	Opens the file for reading and writing only. Places the file pointer at the end of the file. If files does not exist then it attemts to create a file.

If an attempt to open a file fails then **fopen** returns a value of **false** otherwise it returns a **file pointer** which is used for further reading or writing to that file.

After making a changes to the opened file it is important to close it with the **fclose()** function. The **fclose()** function requires a file pointer as its argument and then returns **true** when the closure succeeds or **false** if it fails.

Reading a file

Once a file is opened using **fopen()** function it can be read with a function called **fread()**. This function requires two arguments. These must be the file pointer and the length of the file expressed in bytes.

The files's length can be found using the **filesize()** function which takes the file name as its argument and returns the size of the file expressed in bytes.

So here are the steps required to read a file with PHP.

- Open a file using **fopen()** function.
- Get the file's length using **filesize()** function.
- Read the file's content using **fread()** function.
- Close the file with **fclose()** function.

The following example assigns the content of a text file to a variable then displays those contents on the web page.

```
<html>
<head>
<title>Reading a file using PHP</title>
</head>
<body>

<?php
$filename = "/home/user/guest/tmp.txt";
$file = fopen( $filename, "r" );
if( $file == false )
{
   echo ( "Error in opening file" );
   exit();
}
$filesize = filesize( $filename );
$filetext = fread( $file, $filesize );

fclose( $file );

echo ( "File size : $filesize bytes" );
echo ( "<pre>$filetext</pre>" );
?>

</body>
</html>
```

Writing a file

A new file can be written or text can be appended to an existing file using the PHP **fwrite()**function. This function requires two arguments specifying a **file pointer** and the string of data that is to be written. Optionally a third integer argument can be included to specify the length of the data to write. If the third argument is included, writing would will stop after the specified length has been reached.

The following example creates a new text file then writes a short text heading insite it. After closing this file its existence is confirmed using **file_exist()** function which takes file name as an argument

```
<?php
$filename = "/home/user/guest/newfile.txt";
```

```
$file = fopen( $filename, "w" );
if( $file == false )
{
   echo ( "Error in opening new file" );
   exit();
}
fwrite( $file, "This is  a simple test\n" );
fclose( $file );
?>

<html>
<head>
<title>Writing a file using PHP</title>
</head>
<body>

<?php
if( file_exist( $filename ) )
{
   $filesize = filesize( $filename );
   $msg = "File  created with name $filename ";
   $msg .= "containing $filesize bytes";
   echo ($msg );
}
else
{
   echo ("File $filename does not exit" );
}
?>
</body>
</html>
```

PHP XML File

xml_parse()

The xml_parse() function parses an XML document.

Example 1

XML File

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

PHP Code

```
<?php
$parser=xml_parser_create();

function char($parser,$data)
  {
  echo $data;
  }

xml_set_character_data_handler($parser,"char");
$fp=fopen("test.xml","r");

while ($data=fread($fp,4096))
  {
  xml_parse($parser,$data,feof($fp)) or
  die (sprintf("XML Error: %s at line %d",
  xml_error_string(xml_get_error_code($parser)),
  xml_get_current_line_number($parser)));
  }

xml_parser_free($parser);
?>
```

The output of the code above will be:

```
Tove Jani Reminder Don't forget me this weekend!
```

PHP XML File

str getcsv()

PHP function str_getcsv

This function parses a CSV string into an array.

Syntax str_getcsv

- String: CSV String input
- delimiter: Set the delimiter (only single character is allow)
- enclosure: set the field enclosure (only single character is allow)
- escape: escapse character. Default- backslash (\)

```
str_getcsv (string, delimiter,enclosure, escape);
```

Relate function

If you want to split string into array by specifies a specific delimiter sees explode() function.

Example 1

```
<?php
$input=<<<MYRECORD
Name,Age,School,Score
Yilin,21,MIT,80
Baobao,30,"Stanford University",90
Linlin,22,"A&M University of Texas",89
Mark Lee,32,"Grambling State University",88
MYRECORD;

$record_csv = str_getcsv($input,"\n");
print_r($record_csv);
?>
```

Output

```
Array
(
    [0] => Name,Age,School,Score
    [1] => Yilin,21,MIT,80
    [2] => Baobao,30,"Stanford University",90
    [3] => Linlin,22,"A&M University of Texas",89
```

```
    [4] => Mark Lee,32,"Grambling State University",88
)
```

Example 2

```
<?php
$input=<<<MYRECORD
Product: Price: ID:Location;
Doggy Item:399 : 100B : Dallas, Tx;
PHP prog:430: P340: Grambling, LA

MYRECORD;
$record_csv = str_getcsv($input,";");
print_r($record_csv);
?>
```

Output

```
Array
(
    [0] => Product: Price: ID:Location
    [1] => Doggy Item:399 : 100B : Dallas, Tx
    [2] => PHP prog:430: P340: Grambling, LA
)
```

Example 3

```
<?php
$input=<<<MYRECORD
Product:Price:ID:Location:
'Doggy Item':399:100B:Dallas, Tx:
'PHP prog':430:P340:Grambling, LA
MYRECORD;
$record_csv = str_getcsv($input,":","'");
print_r($record_csv);
?>
```

Output

```
Array
```

```
(
    [0] => Product
    [1] => Price
    [2] => ID
    [3] => Location
    [4] => Doggy Item
    [5] => 399
    [6] => 100B
    [7] => Dallas, Tx
    [8] => PHP prog
    [9] => 430
    [10] => P340
    [11] => Grambling, LA
)
```

Read CSV

test.csv

```
Apple,RED
Orange,ORANDE
Banana,YELLOW
Grapes,VIOLET
KIWI,GREEN
Dates,BROWN
```

Php File

```
<html>
<head>
<title>Read CSV</title>
</head>
<body>
<?php
$filename = "test.csv";
$file = fopen( $filename, "r" );
if( $file == false )
{
   echo ( "Error in opening file" );
   exit();
}
$filesize = filesize( $filename );
```

```
$filetext = fread( $file, $filesize );
fclose( $file );
$record_csv = str_getcsv( $filetext,"\n");
print_r($record_csv);
?>
</body>
</html>
```