PHP – Functions

PHP - File Inclusion

Assistant Prof. Dr. Rafah M. Almuttairi

PHP - Functions

- PHP functions are similar to other programming languages. A **function** is a piece of code which takes one more input in the form of parameter and does some processing and returns a value.
- You already have seen many functions like **fopen()** and **fread()** etc. They are built-in functions but PHP gives you option to create your own functions as well.
- There are two parts which should be clear to you –
  - **Creating a PHP Function**
  - **Calling a PHP Function**
- In fact you hardly need to create your own PHP function because there are already more than 1000 of built-in library functions created for different area and you just need to call them according to your requirement.

Creating PHP Function

- Its very easy to create your own PHP function. Suppose you want to create a PHP function which will simply write a simple message on your browser when you will call it. Following example creates a function called writeMessage() and then calls it just after creating it.
- Note that while creating a function its name should start with keyword **function** and all the PHP code should be put inside { and } braces as shown in the following example below –

```
<html>

   <head>
      <title>Writing PHP Function</title>
   </head>

   <body>

      <?php
         /* Defining a PHP Function */
         function writeMessage() {
            echo "You are really a nice person, Have a nice time!";
         }

         /* Calling a PHP Function */
         writeMessage();
      ?>

   </body>
</html>
```

This will display following result –
You are really a nice person, Have a nice time!

PHP Functions with Parameters

- PHP gives you option to pass your parameters inside a function. You can pass as many as parameters your like. These parameters work like variables inside your function. Following example takes two integer parameters and add them together and then print them.

```
<html>

   <head>
      <title>Writing PHP Function with Parameters</title
   </head>

   <body>

      <?php
         function addFunction($num1, $num2) {
            $sum = $num1 + $num2;
            echo "Sum of the two numbers is : $sum";
         }

         addFunction(10, 20);
      ?>

   </body>
</html>
```

This will display following result–
Sum of the two numbers is : 30

Passing Arguments by Reference

- It is possible to pass arguments to functions by reference. This means that a reference to the variable is manipulated by the function rather than a copy of the variable's value.
- Any changes made to an argument in these cases will change the value of the original variable. You can pass an argument by reference by adding an ampersand to the variable name in either the function call or the function definition.
- Following example depicts both the cases.

```
<html>

   <head>
      <title>Passing Argument by Reference</title>
   </head>

   <body>

      <?php
         function addFive($num) {
            $num += 5;
         }

         function addSix(&$num) {
            $num += 6;
         }

         $orignum = 10;
         addFive( $orignum );

         echo "Original Value is $orignum<br />";

         addSix( $orignum );
         echo "Original Value is $orignum<br />";
      ?>

   </body>
</html>
```

This will display following result

```
Original Value is 10
Original Value is 16
```

PHP Functions returning value

- A function can return a value using the **return** statement in conjunction with a value or object. return stops the execution of the function and sends the value back to the calling code.
- You can return more than one value from a function using **return array(1,2,3,4)**.
- Following example takes two integer parameters and add them together and then returns their sum to the calling program. Note that **return** keyword is used to return a value from a function.

```php
<html>

   <head>
      <title>Writing PHP Function which returns value</title>
   </head>

   <body>

      <?php
         function addFunction($num1, $num2) {
            $sum = $num1 + $num2;
            return $sum;
         }
         $return_value = addFunction(10, 20);

         echo "Returned value from the function : $return_value";
      ?>

   </body>
</html>
```

This will display following result –

Returned value from the function : 30

Setting Default Values for Function Parameters

- You can set a parameter to have a default value if the function's caller doesn't pass it.
- Following function prints NULL in case user does not pass any value to this function.

```
<html>

   <head>
      <title>Writing PHP Function which returns value</title>
   </head>

   <body>

      <?php
         function printMe($param = NULL) {
            print $param;
         }

         printMe("This is test");
         printMe();
      ?>

   </body>
</html>
```

This will produce following result –
This is test

Dynamic Function Calls

- It is possible to assign function names as strings to variables and then treat these variables exactly as you would the function name itself. Following example depicts this behavior.

```
<html>

   <head>
      <title>Dynamic Function Calls</title>
   </head>

   <body>

      <?php
         function sayHello() {
            echo "Hello<br />";
         }

         $function_holder = "sayHello";
         $function_holder();
      ?>

   </body>
</html>
```

This will display following result –
Hello

PHP - File Inclusion

- 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://www.tutorialspoint.com/index.htm">Home</a> -
<a href="http://www.tutorialspoint.com/ebxml">ebXML</a> -
<a href="http://www.tutorialspoint.com/ajax">AJAX</a> -
<a href="http://www.tutorialspoint.com/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>
```

It will produce the following result –

Home -
ebXML -
AJAX -
PERL

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

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 the 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!</

   </body>
</html>
```

This time file execution halts and nothing is displayed.

NOTE – You may get plain warning messages or fatal error messages or nothing at all. This depends on your PHP Server configuration.