## PHP User Defined Functions

**Besides the built-in PHP functions, we can create our own functions.**

**A function is a block of statements that can be used repeatedly in a program.**

**A function will not execute immediately when a page loads.**

**A function will be executed by a call to the function.**

## Create a User Defined Function in PHP

**A user-defined function declaration starts with the word function:**

### Syntax

**function *functionName*() {  
*code to be executed*;  
}**

**Note: A function name can start with a letter or underscore (not a number).**

**Tip: Give the function a name that reflects what the function does!**

**Function names are NOT case-sensitive.**

**In the example below, we create a function named "writeMsg()". The opening curly brace ( { ) indicates the beginning of the function code and the closing curly brace ( } ) indicates the end of the function. The function outputs "Hello world!". To call the function, just write its name:**

### Example

**<!DOCTYPE html>  
<html>  
<body>  
  
<?php  
function writeMsg() {  
    echo "Hello world!";  
}  
  
writeMsg();  
?>  
  
</body>  
</html>**

## PHP Function Arguments

**Information can be passed to functions through arguments. An argument is just like a variable.**

**Arguments are specified after the function name, inside the parentheses. You can add as many arguments as you want, just separate them with a comma.**

**The following example has a function with one argument ($fname). When the familyName() function is called, we also pass along a name (e.g. Jani), and the name is used inside the function, which outputs several different first names, but an equal last name:**

### Example

**<!DOCTYPE html>**

**<html>**

**<body>**

**<?php**

**function familyName($fname) {**

**echo "$fname.<br>";**

**}**

**familyName("san");**

**familyName("dan");**

**?>**

**</body>**

**</html>**

**The following example has a function with two arguments ($fname and $year):**

### Example

**<?php  
function familyName($fname, $year) {  
    echo "$fname Refsnes. Born in $year <br>";  
}  
  
familyName("Hege", "1975");  
familyName("Stale", "1978");  
familyName("Kai Jim", "1983");  
?>**

**<!DOCTYPE html>  
<html>  
<body>  
  
<?php  
function familyName($fname, $year) {  
    echo "$fname Refsnes. Born in $year <br>";  
}  
  
familyName("Hege","1975");  
familyName("Stale","1978");  
familyName("Kai Jim","1983");  
?>  
  
</body>  
</html>**

## PHP Default Argument Value

**The following example shows how to use a default parameter. If we call the function setHeight() without arguments it takes the default value as argument:**

### Example

**<!DOCTYPE html>  
<html>  
<body>  
  
<?php  
function setHeight($minheight = 50) {  
    echo "The height is : $minheight <br>";  
}  
  
setHeight(350);  
setHeight();  
setHeight(135);  
setHeight(80);  
?>  
  
</body>  
</html>**

## PHP Functions - Returning values

**To let a function return a value, use the return statement:**

### Example

**<!DOCTYPE html>  
<html>  
<body>  
  
<?php  
function sum($x, $y) {  
    $z = $x + $y;  
    return $z;  
}  
  
echo "5 + 10 = " . sum(5,10) . "<br>";  
echo "7 + 13 = " . sum(7,13) . "<br>";  
echo "2 + 4 = " . sum(2,4);  
?>  
</body>  
</html>**

**PHP has three different variable**

**local**

**global**

**static**

**Local variable: - A variable declared within a PHP function is local and can only be accessed**

**within that function.**

**<?php**

**function test(){**

**$a = 10;**

**echo 'The Value of $a inside function is : ',$a,'<br>';**

**}**

**test();**

**echo 'The Value of $a Outside Function is : ',$a,'<br>';**

**?>**

**output**

The Value of $a inside function is : 10  
The Value of $a Outside Function is :  
**Warning**: Undefined variable $a in **C:\xampp\htdocs\san\summ.php** on line **7**

## PHP The static Keyword

Normally, when a function is completed/executed, all of its variables are deleted. However, sometimes we want a local variable NOT to be deleted. We need it for a further job.

To do this, use the static keyword when you first declare the variable:

# <!DOCTYPE html>

# <html>

# <body>

# <?php

# function myTest()

# {

# static $x = 0;

# $b=5;

# echo "x= ". $x ."<br>";

# echo "b= " . $b . "<br>";

# $x++;

# }

# myTest();

# echo "<br>";

# myTest();

# echo "<br>";

# myTest();

# ?>

# </body>

# </html>

output

x= 0  
b= 5  
  
x= 1  
b= 5  
  
x= 2  
b= 5

## Global and Local Scope

A variable declared **outside** a function has a GLOBAL SCOPE and can only be accessed outside a function:

# <!DOCTYPE html>

# <html>

# <body>

# <?php

# $x = 5; // global scope

# 

# function myTest() {

# // using x inside this function will generate an error

# echo "<p>Variable x inside function is: $x</p>";

# }

# myTest();

# echo "<p>Variable x outside function is: $x</p>";

# ?>

# </body>

# </html>

**Output**

**Warning**: Undefined variable $x in **C:\xampp\htdocs\san\summ.php** on line **10**

Variable x inside function is:

Variable x outside function is: 5

## PHP Global Variables - Superglobals

**Several predefined variables in PHP are "superglobals", which means that they are always accessible, regardless of scope - and you can access them from any function, class or file without having to do anything special.**

**The PHP superglobal variables are:**

* **$GLOBALS**
* **$\_SERVER**
* **$\_REQUEST**
* **$\_POST**
* **$\_GET**
* **$\_FILES**
* **$\_ENV**
* **$\_COOKIE**
* **$\_SESSION**

## PHP $GLOBALS

**$GLOBALS is a PHP super global variable which is used to access global variables from anywhere in the PHP script (also from within functions or methods).**

**PHP stores all global variables in an array called $GLOBALS[*index*]. The**index**holds the name of the variable.**

**The example below shows how to use the super global variable $GLOBALS:**

### Example

**<!DOCTYPE html>  
<html>  
<body>  
  
<?php   
$x = 75;  
$y = 25;   
  
function addition() {  
    $GLOBALS['z'] = $GLOBALS['x'] + $GLOBALS['y'];  
}  
  
addition();  
echo $z;  
?>  
  
</body>  
</html>**

**In the example above, since z is a variable present within the $GLOBALS array, it is also accessible from outside the function!**

## PHP $\_SERVER

**$\_SERVER is a PHP super global variable which holds information about headers, paths, and script locations.**

**The example below shows how to use some of the elements in $\_SERVER:**

### Example

**<!DOCTYPE html>  
<html>  
<body>  
  
<?php   
echo $\_SERVER['PHP\_SELF'];  
echo "<br>";  
echo $\_SERVER['SERVER\_NAME'];  
echo "<br>";  
echo $\_SERVER['HTTP\_HOST'];  
echo "<br>";  
echo $\_SERVER['HTTP\_REFERER'];  
echo "<br>";  
echo $\_SERVER['HTTP\_USER\_AGENT'];  
echo "<br>";  
echo $\_SERVER['SCRIPT\_NAME'];  
?>  
</body>  
</html>**

**output**

/san/summ.php  
localhost  
localhost  
  
**Warning**: Undefined array key "HTTP\_REFERER" in **C:\xampp\htdocs\san\summ.php** on line **8**  
  
Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/101.0.4951.54 Safari/537.36  
/san/summ.php

## PHP $\_REQUEST

**PHP $\_REQUEST is used to collect data after submitting an HTML form.**

**The example below shows a form with an input field and a submit button. When a user submits the data by clicking on "Submit", the form data is sent to the file specified in the action attribute of the <form> tag. In this example, we point to this file itself for processing form data. If you wish to use another PHP file to process form data, replace that with the filename of your choice. Then, we can use the super global variable $\_REQUEST to collect the value of the input field:**

**<!DOCTYPE html>  
<html>  
<body>  
<form method="post" action="<?php echo $\_SERVER['PHP\_SELF'];?>">  
  Name: <input type="text" name="fname">  
  <input type="submit">  
</form>  
  
<?php  
if ($\_SERVER["REQUEST\_METHOD"] == "POST") {  
    // collect value of input field  
    $name = htmlspecialchars($\_REQUEST['fname']);   
    if (empty($name)) {  
        echo "Name is empty";  
    } else {  
        echo $name;  
    }  
}  
?>  
  
</body>  
</html>**

## PHP $\_POST

**PHP $\_POST is widely used to collect form data after submitting an HTML form with method="post". $\_POST is also widely used to pass variables.**

**The example below shows a form with an input field and a submit button. When a user submits the data by clicking on "Submit", the form data is sent to the file specified in the action attribute of the <form> tag. In this example, we point to the file itself for processing form data. If you wish to use another PHP file to process form data, replace that with the filename of your choice. Then, we can use the super global variable $\_POST to collect the value of the input field:**

### Example

**<!DOCTYPE html>  
<html>  
<body>  
  
<form method="post" action="<?php echo $\_SERVER['PHP\_SELF'];?>">  
  Name: <input type="text" name="fname">  
  <input type="submit">  
</form>  
  
<?php  
if ($\_SERVER["REQUEST\_METHOD"] == "POST") {  
    // collect value of input field  
    $name = $\_POST['fname'];   
    if (empty($name)) {  
        echo "Name is empty";  
    } else {  
        echo $name;  
    }  
}  
?>  
</body>  
</html>**

# PHP 5 Form Handling

**The PHP superglobals $\_GET and $\_POST are used to collect form-data.**

## PHP - A Simple HTML Form

**The example below displays a simple HTML form with two input fields and a submit button:**

### Example

**<!DOCTYPE HTML>  
<html>    
<body>  
<form action="welcome.php" method="post">  
Name: <input type="text" name="name"><br>  
E-mail: <input type="text" name="email"><br>  
<input type="submit">  
</form>  
</body>  
</html>**

**When the user fills out the form above and clicks the submit button, the form data is sent for processing to a PHP file named "welcome.php". The form data is sent with the HTTP POST method.**

**To display the submitted data you could simply echo all the variables. The "welcome.php" looks like this:**

**<html>  
<body>  
Welcome <?php echo $\_POST["name"]; ?><br>  
Your email address is: <?php echo $\_POST["email"]; ?>  
</body>  
</html>**

**The output could be something like this:**

**Welcome John  
Your email address is john.doe@example.com**

**The same result could also be achieved using the HTTP GET method:**

### Example

**<!DOCTYPE HTML>  
<html>    
<body>  
<form action="welcome\_get.php" method="get">  
Name: <input type="text" name="name"><br>  
E-mail: <input type="text" name="email"><br>  
<input type="submit">  
</form>  
</body>  
</html>**

**and "welcome\_get.php" looks like this:**

**<html>  
<body>  
Welcome <?php echo $\_GET["name"]; ?><br>  
Your email address is: <?php echo $\_GET["email"]; ?>  
  
</body>  
</html>**

**Program1**

**<html>**

**<body>**

**<form method="post">**

**Enter First Number:**

**<input type="number" name="number1" /><br><br>**

**Enter Second Number:**

**<input type="number" name="number2" /><br><br>**

**<input type="submit" name="submit" value="Add">**

**</form>**

**<?php**

**if(isset($\_POST['submit']))**

**{**

**$number1 = $\_POST['number1'];**

**$number2 = $\_POST['number2'];**

**$sum = $number1+$number2;**

**echo "The sum of $number1 and $number2 is: ".$sum;**

**}**

**?>**

**</body>**

**</html>**

**Program2**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<title>PHP insertion</title>**

**<body>**

**<div class="title">**

**<h2>Insert Data In Database Using PHP.</h2>**

**</div>**

**<form action="http://localhost/san/insert.php" method="post">**

**<h2>Form</h2>**

**<label>Code:</label>**

**<input class="input" name="cd" type="text" value="">**

**<label>FirstName:</label>**

**<input class="input" name="fn" type="text" value="">**

**<label>LastName:</label>**

**<input class="input" name="ln" type="text" value="">**

**<input class="submit" name="submit" type="submit" value="Insert">**

**</form>**

**</body>**

**</html>**

## Definition and Usage

The isset() function checks whether a variable is set, which means that it has to be declared and is not NULL.

This function returns true if the variable exists and is not NULL, otherwise it returns false.

**insert.php**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<title>Hello!</title>**

**</head>**

**<body>**

**<?php**

**if(isset($\_POST['submit']))**

**{**

**$ecode = $\_POST['cd'];**

**$efname = $\_POST['fn'];**

**$elname = $\_POST['ln'];**

**echo "$ecode " . "$efname " . "$elname "."<br>" ;**

**}**

**?>**

**</body>**

**</html>**