Lecture 5 – ICT1233

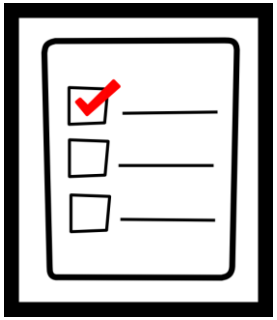
PHP Functions

Department of ICT

Faculty of Technology

Objective

- After the successful completion of this lecture, students should be able to,
  - Identify the use of functions in web development
  - Use user defined functions in web developing



Introduction

- A *function* is a named block of code
- Function performs a specific task, possibly acting upon a set of values given to it, or *parameters*, and possibly returning a single value
- Two types
  - Built-in
  - User defined

Advantages of Using Functions

- Allows to only write the code once and save time and space
- Improve reliability by allowing to fix any bugs in one place, rather than everywhere you perform a task
- Improve readability by isolating code that performs specific tasks

User Defined Functions

- Functions that are defined by users according to their requirements
- User defined function declaration starts with the word "function"

Syntax

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

User Defined Function Guidelines

- Give a function name that reflects what the function does
- A valid function name starts with a letter or underscore, followed by any number of letters, numbers, or underscores
- Function names are case sensitive

Arguments of Functions

- Values that are sent to a function are called as arguments
- Information may be passed to functions via the argument list,
  - which is a comma-delimited list of expressions
  - The arguments are evaluated from left to right

- Syntax

```
function Myfunction ($arg1, $arg2)
```

Passed by Value

- In most cases, you pass parameters by value
- Copy of the variable's value is manipulated by the function

```
<?php
        function addFive($num) {
           $num += 5;
        }
        $orignum = 10;
        addFive( $orignum );
        echo "Original Value is $orignum<br />";

?>
```

Output:
Original Value is 10

Passed by Reference

- A reference to the variable is manipulated by the function rather than a copy of the variable's value
- **To be passed by reference, the argument must be a variable**
- Any changes made to an argument in these cases will change the value of the original variable
- Pass an argument by reference by adding an ampersand (&) to the variable name in either the function call or the function definition

Passed by Reference

```
<?php
        function addSix(&$num) {
           $num += 6;
        }
       $orignum = 10;

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

Output:
Original Value is 16

Default Argument Value

- When designing functions, it is often helpful to be able to assign default values for parameters that aren't passed
- To define default parameters for a function, simply include the constant value you would like to set to the parameter

Variable Length Argument Function

- A function may require a variable number of arguments. rather than just one
- To declare a function with a variable number of arguments, leave out the parameter block entirely
- PHP provides three functions which you can use in the function to retrieve the parameters passed to it
  1. func_num_args()
  2. func_get_arg()
  3. func_get_args()

Example: func_num_args()

```
<?php
function printArg()
{
   $numargs = func_num_args();
   echo "Number of arguments: $numargs\n";
}

printArg(1, 2, 3);
?>
```

Number of arguments: 3

Example: func_get_arg(position)

```
<?php
function showArg()
{
    $numargs = func_num_args();
    echo "Number of arguments: $numargs\n";
    if ($numargs >= 2) {
        echo "Second argument is: " . func_get_arg(1) . "\n";
    }
}

showArg(1, 5, 3);
?>
```

Number of arguments: 3
Second argument is : 5

Example : func_get_args()

```
<?php
function showArray()
{
   $numargs = func_num_args();
   echo "Number of arguments: $numargs \n";
   if ($numargs >= 2) {
      echo "Second argument is: " . func_get_arg(1) . "\n";
   }
   $arg_list = func_get_args();
   for ($i = 0; $i < $numargs; $i++) {
      echo "Argument $i is: " . $arg_list[$i] . "\n";
   }
}

showArray(1, 5, 3);
?>
```

Output

Number of arguments: 3
Second argument is: 5
Argument 0 is: 1
Argument 1 is: 5
Argument 2 is: 3

Variable Length Argument Function

- With all these 3 functions, PHP supports variable-length argument lists with some basic rules:
  - The function call can provide more arguments than the number of argument variables defined in the function statement
  - You can pass arguments to a function that has no argument variable defined

Variable Length Argument List

```
<?php
 function dynamic_args() {
     for($i = 0 ; $i < func_num_args(); $i++) {
          echo "Argument $i = ".func_get_arg($i)."<br />";
     }
     $args = func_get_args();
     foreach ($args as $key => $value) {
          echo "Argument {$key}: {$value} <br />";
     }
 }
 dynamic_args("a", "b", "c", "d", "e");
?>
```

Summary

- PHP functions
  - User defined functions
- Advantages of using functions
- Function arguments

Questions…