PHP Programming
COMP203TH
Lecture: 3
Using Variables in PHP

Mixing PHP with HTML:

When the PHP parser reads a script, it executes only the code found between PHP tags, everything outside these tags is ignored by the parser and returned as is. This makes it extremely easy to embed PHP code within an HTML document to create Web Pages.

e.g.

```
<html>
<body>
<h1> Mixing PHP and HTML </h1>
<?php
echo "My first php line";
?>
</body>
</html>
```

PHP statements end with a semicolon (;).

PHP Variables (storing data in variables):

A variable is simply a container that's used to store both numeric and non-numeric information.

Some important rules about naming variables in PHP:

- All variables in PHP are denoted with a leading dollar sign ($).
- Variable name must begin with a letter or underscore character optionally followed by more letters, numbers or underscore characters.
- Common punctuation characters such as commas, quotation marks or periods are not permitted in variable names neither are spaces.

Assigning Values to Variables:

Use the equality (=) symbol, which is PHP's assignment operator. This assigns the value on the right side of the equation to the variable on the left.

To use a variable in a script, simply call it by name in an expression and PHP will replace it with its value when the script is executed.

e.g.

```
<html>

<head>

<title> </title>

</head>

<body>

<?php

// assign value to variable

$name = 'Victor'

?>

<h2> Welcome to <?php echo $name; ?>'s Blog!</h2>

</body>

</html>
```

In this example, the variable $name is assigned the value 'Victor'. The echo statement is then used to print the value of this variable to the Web page.

Important Points:

- ***Variables in PHP do not have intrinsic types i.e. a variable does not know in advance whether it will be used to store a number or a string of characters. In other words, PHP is a loosely typed language, it means PHP automatically converts the variable to its correct data type.***
- ***Variable names are case sensitive.***

You can also assign a variable the value of another variable or the result of a calculation.

e.g.

```
<?php

$now=  2020;

$currentyear = $now;

$nextyear =$currentyear+1;

echo "$currentyear is going to end. Welcome the $nextyear!";

?>
```

Note:

A user defined function in php starts with the word function:

Syntax: function functionname() {

code to be executed;

}

PHP Variable Scope:

The scope of a variable is defined as its range in the program under which it can be accessed.

PHP has three types of variable scopes:

1. Local Variable
2. Global Variable
3. Static Variable

Local Variable:

The variables that are declared within a function are called local variables for that function. These local variables have their scope only in that particular function in which they are declared i.e. they cannot be accessed outside the function.

e.g.

```
<?php
function local_var()
{
$num = 30;  //local variable
echo "the value of local variable=".$num;
}
local_var();
?>
```

Global Variable:

The global variables are the variables that are declared outside the function. These variables can be accessed anywhere in the program.

To access the global variable within a function, use the ***global*** *keyword before the variable.* However, these variables can be directly accessed or used outside the function without any keyword. Therefore, there is no need to use any keyword to access a global variable outside the function.

e.g.

```
<?php
$name = “Victor”;    //global variable
functio global_var()
{
        global $name;
        echo “Variable inside the function:”.$name;
        echo “</br>”;
}
global_var();
echo “Variable outside the function:”.$name;
?>
```

Without using the global keyword, if you try to access a global variable inside the function, it will generate an error that the variable is undefined.

Another way to use the global variable inside the function is predefined $GLOBALS array.

e.g.

```
<?php
$num1 =2; //global variable
$num2=4;  //global variable
function global_var()
{
```

```
    $num = $GLOBALS['num1'] + $GLOBALS['num2'];
    echo "Sum of global variables is:".$num;
}
global_var();
?>
```

If two variables local and global have the same name, then the local variable has higher priority than the global variable inside the function.

Static Variable:

It is a feature of PHP to delete the variable once it completes it execution and memory is freed. Sometimes, we need to store a variable even after completion of unction execution. In such scenario, we use static variable.

We use static keyword before the variable to define a variable and this variable is called static variable. It does not free its memory after the program execution leaves the scope.

e.g.

```
<?php
    function static_var()
    {
        static $num1 = 2;   //static variable
        $num2 = 3;    // non-static vriable
        $num1++;
        $num2++;
        echo "Static".$num1."</br>";
        echo "Non-static:".$num2."</br>";
    }
    //first function call
    static_var();
    //second function call
    static_var();
?>
```

Here, in the output you will observe $num1 regularly increments after each function call, whereas $num2 does not. Because $num2 is not a static variable, so it freed its memory after the execution of each function call.