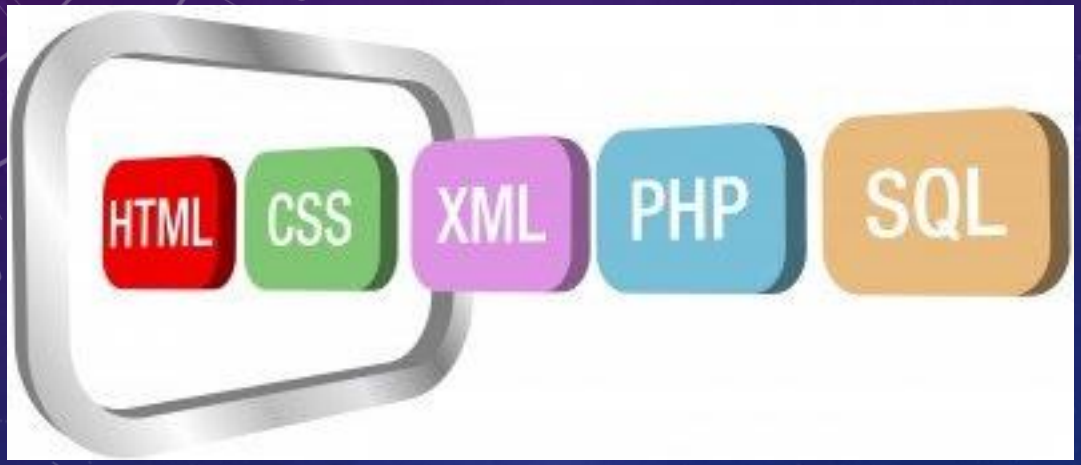
LECTURE 06: PHP

Web Programming



INSTRUCTOR: DR. HOSSAM ZAWBAA

VARIABLES

- PHP variables must begin with a “$” sign
- Case-sensitive ($Foo != $foo != $fOo)
- Global and locally-scoped variables
  - Global variables can be used anywhere
  - Local variables restricted to a function or class
- Certain variable names reserved by PHP
- Form variables ($_POST, $_GET)
- Server variables ($_SERVER)

VARIABLES

```php
<?php
    $name = “ali”
    echo( $name);
?>
```

VARIABLES

```
<?php
    $name = “Mohamed”;
    $age = 23;
    Echo “ My name is $name and I am $age years old”;
?>
```

VARIABLES

```php
<?php
    $name = ‘Ahmed';
    $yearborn = 1995;
    $currentyear = 2018;
    $age = $currentyear - $yearborn;
    echo ("$name is $age years old.");
?>
```

VARIABLES

```php
<?php
    $age = 37;
    $stringTest = 'I am $age years old';
    // output: I am $age years old

    $stringTest = “I am $age years old”;
    // output: I am 37 years old
?>
```

VARIABLES

```
<?php   $name = “Ali";    // declaration     ?>
    <html>
        <head> <title>A simple PHP document</title> </head>
        <body style = "font-size: 2em">
            <p> <strong>
                    <!-- print variable name’s value -->
                    Welcome to PHP, <?php echo( "$name" ); ?>!
                </strong> </p>
        </body>
    </html>
```

ANATOMY OF A PHP SCRIPT

Comments

- // or # for single line
- /* */ for multiline
- /*

this is my comment one

this is my comment two

this is my comment three

*/

ANATOMY OF A PHP SCRIPT

❑ Whitespace

- You can not have any whitespace between <? and php.
- You can not break apart keywords (e.g :while, function, for)
- You can not break apart variable names and function names (e.g: $var_name, function f2).

ANATOMY OF A PHP SCRIPT

- Code Block

- Is simply a series of statements' enclosed between two braces:

```
{
//some comand
}
```

CONCATENATION

- Use a period to join strings into one.

```
<?php
        $string1=“Hello”;
        $string2=“PHP”;
        $string3=$string1 . “ ” . $string2;
        Print $string3;
?>
```

```
Hello PHP
```

EXAMPLE

```
<?php
    $foo = 25;       // Numerical variable
    $bar = “Hello”;   // String variable
    echo $bar;       // Outputs Hello
    echo $foo, $bar;    // Outputs 25Hello
    echo “5x5=”, $foo;    // Outputs 5x5=25
    echo “5x5=$foo”;// Outputs 5x5=25
    echo ‘5x5=$foo’;  // Outputs 5x5=$foo
?>
```

DATA TYPE

Data type	Description
int, integer	Whole numbers (i.e., numbers without a decimal point).
float, double	Real numbers (i.e., numbers containing a decimal point).
string	Text enclosed in either single (' ') or double (" ") quotes.
bool, Boolean	True or false.
array	Group of elements of the same type.
object	Group of associated data and methods.
Resource	
NULL	No value.

PHP data types.

ARITHMETIC OPERATORS

```php
<?php
    $a=15;
    $b=30;
    $total=$a+$b;
    echo $total;

    echo"<p><h1>$total</h1>";
    // total is 45
?>
```

- $a - $b // subtraction
- $a * $b // multiplication
- $a / $b // division
- $a += 5 // $a = $a+5 Also works for *= and /=

LOGIC OPERATIONS

Example	Name	Result
$a == $b	Equal	TRUE if $a is equal to $b after type juggling.
$a === $b	Identical	TRUE if $a is equal to $b, and they are of the same type.
$a != $b	Not equal	TRUE if $a is not equal to $b .
$a <> $b	Not equal	TRUE if $a is not equal to $b
$a !== $b	Not identical	TRUE if $a is not equal to $b, or they are not of the same type.
$a < $b	Less than	TRUE if $a is strictly less than $b.
$a > $b	Greater than	TRUE if $a is strictly greater than $b.
$a <= $b	Less than or equal to	TRUE if $a is less than or equal to $b.
$a >= $b	Greater than or equal to	TRUE if $a is greater than or equal to $b.

IF STATEMENT

- if (condition)
  {
   statements;
  }
  else
  {
   statement;
  }

```
<?php
    $user = "Ahmed";
    if($user=="Ahmed")
    {
        print "hello Ahmed.";
    }
    else
    {
        print "you are not Ahmed.";
    }
?>
```

```
hello Ahmed
```

IF/ELSE STATEMENT

```
<?php
    if ($foo == 0) {
      echo ‘The variable foo is equal to 0’;
    }
    else if (($foo > 0) && ($foo <= 5)) {
      echo ‘The variable foo is between 1 and 5’;
    }
    else {
      echo ‘The variable foo is equal to ‘.$foo;
    }
?>
```

SWITCH STATEMENT

```
switch(expression )
   {
    case value:
    break;
    .
    .
     default:
    break;
   }
```

```
<?php
    $count=0;
    switch($count)
    {
         case 0:
            Print "hello PHP3. ";
        break;
        case 1:
            Print "hello PHP4. ";
        break;
         default:
            Print "hello PHP5. ";
        break;

?>
```

```
hello PHP3
```

FOR LOOP

- for ($variable = value; condition; $value assignment)
  {
  statements;
  }

```
<?php
$count=0;
for($count = 0;$count <3,$count++)
{
    Print "hello PHP. ";
}
?>
```

```
hello PHP. hello PHP. hello PHP.
```

WHILE LOOP

```
while (condition)
  {
   statements;
  }
```

```php
<?php
$count=0;
while($count<3)
{
    echo "hello PHP. ";
    $count += 1;
    // $count = $count + 1;
    // or
    // $count++;
}
?>
```

```
hello PHP. hello PHP. hello PHP.
```

WHILE - EXAMPLE

```
<?php
   $i = 0;
   while ($i++ < 5)
    {
       echo “loop number : “.$i;
    }
?>
```

DO ... WHILE LOOP

```
do
{
    statements;
}
while (condition);
```

```php
<?php
$count=0;
do
{
echo "hello PHP. ";
    $count += 1;
    // $count = $count + 1;
    // or
    // $count++;
}
while($count<3);
?>
```

```
hello PHP. hello PHP. hello PHP.
```

ISSET FUNCTION

```
<?php
    $var = '';
    // This will evaluate to TRUE so the text will be printed.
    if (isset($var))
     {
        echo "This var is set so I will print.";
    }
?>
```

GOTO

```php
<?php
  goto a;
  echo 'Foo';

  a:
  echo 'Bar';
?>
```

PHP BASIC SERIES – POST, GET & REQUEST

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

```
<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 = $_REQUEST['fname'];
    if (empty($name)) {
        echo "Name is empty";
    } else {
        echo $name;
    }
}
?>

</body>
</html>
```

PHP BASIC SERIES – POST, GET & REQUEST

- PHP $_GET
  - PHP $_GET can also be used to collect form data after submitting an HTML form with method="get".

```
<html>
<body>

<?php
echo "Study " . $_GET['subject'] . " at " . $_GET['web'];
?>

</body>
</html>
```

PHP SESSION

- $_SESSION.
- A session is started with the session_start() function.

```
<?php
// Start the session
session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
// Set session variables
$_SESSION["favcolor"] = "green";
$_SESSION["favanimal"] = "cat";
echo "Session variables are set.";
?>

</body>
</html>
```

PHP COOKIES

- A cookie is created with the setcookie() function.
- Syntax

```
setcookie(name, value, expire, path, domain, secure, httponly);
```

```
<?php
$cookie_name = "user";
$cookie_value = "John Doe";
setcookie($cookie_name, $cookie_value, time() + (86400 * 30), "/"); // 86400 = 1 day
?>
<html>
<body>

<?php
if(!isset($_COOKIE[$cookie_name])) {
    echo "Cookie named '" . $cookie_name . "' is not set!";
} else {
    echo "Cookie '" . $cookie_name . "' is set!<br>";
    echo "Value is: " . $_COOKIE[$cookie_name];
}
?>

</body>
</html>
```

PHP AND MYSQL

```
<?php
$con=mysqli_connect("localhost","my_user","my_password","my_db");
// Check connection
if (mysqli_connect_errno())
  {
  echo "Failed to connect to MySQL: " . mysqli_connect_error();
  }

// Perform queries
mysqli_query($con,"SELECT * FROM Persons");
mysqli_query($con,"INSERT INTO Persons (FirstName,LastName,Age)
VALUES ('Glenn','Quagmire',33)");

mysqli_close($con);
?>
```