Programming In PHP
HM...

oh ok
PHP

• Imperative programming language

• Dynamic type checking

• based on C programming Style

• Support OO (object oriented) programming
Let’s go
Ups... sorry HTML first

<!DOCTYPE html>
<html>
<body>
<form action="testaction.php" method="get">
<input type="text" name="var1" />
<button type="submit">Submit</button>
</form>
<?php

?>

</body>
</html>
Comments

/* multi
line
comments*/

// single line comment
Variable

\$ is the key

\$txt = "Hello world!";
\$x = 5;
\$y = 10.5;
Data and Types

- String: "Hello world"
- Integer: $x = 3424$
- Float (floating point numbers - also called double) = $10.344$
- Boolean: true or false
- Array: $pl = array(\text{"php"}, \text{"python"}, \text{"scala"})$
- Object: (will see next)
- NULL: $x = \text{null};;$
- Resource: this special resource type is not an actual data type. I will not talk about it, since it is an advanced topic.
Output with echo and print

// echo
<?php
$txt1 = "PHP";
$txt2 = "Web programming with Framework ";
$x = 5;
$y = 4;

echo "<h2>" . $txt1 . "</h2>";
echo "Study PHP at " . $txt2 . "<br>";
echo $x + $y;
?>

// print
<?php
$txt1 = "PHP";
$txt2 = "Web programming with Framework ";
$x = 5;
$y = 4;

print "<h2>" . $txt1 . "</h2>";
print "Study PHP at " . $txt2 . "<br>";
print $x + $y;
?>
Operator

- Arithmetic operator: +, -, *, /, %, **
- Assignment operator: =, +=, -=, *=, /=, %= 
- Comparison Operator: ==, ===, != (<>), !==, >, <, >=, <=, <=>
- Increment/ decrement operator: ++$x, $x++, -$x, $x--
- Logical operator: and, or, xor, &&, ||, !
- String operator: ., .=
- Array operator: +, ==, ===, !=, <>, !==
- Conditional assignment operator: ?:, ??
Arithmetic Operator

```php
<?php
//Arithmatic operator
$a = 50;
$b = 10;
echo "a = " . $a . " and b = " . $b . "<br/>";
echo "a + b = " . ($a + $b) . "<br/>";
echo "a - b = " . ($a - $b) . "<br/>";
echo "a * b = " . ($a * $b) . "<br/>";
echo "a / b = " . ($a / $b) . "<br/>";
echo "a % b = " . ($a % $b) . "<br/>";
echo "a**2 = " . ($a**2) . "<br/>";
echo "b**3 = " . ($b**3) . "<br/>";
?>
```

**OUTPUT**
a = 50 and b = 10
a + b = 60
a - b = 40
a * b = 500
a / b = 5
a % b = 0
a**2 = 2500
b**3 = 1000
Assignment Operator

- **Addition:** $a += b$ is equal to $a = a + b$
- **Subtraction:** $a -= b$ is equal to $a = a - b$
- **Multiplication:** $a *= b$ is equal to $a = a * b$
- **Division:** $a /= b$ is equal to $a = a / b$
- **Modulus:** $a %= b$ is equal to $a = a % b$
- **Adding one:** $++a$ is equal to $a = a + 1$
- **Subtract one:** $-a$ is equal to $a = a - 1$

```php
<?php
// Assignment Operator
echo "<h1>Assignment operator</h1>

$a = 50;
$b = 10;
$c = 0;
$d = 10;
$e = 3;

echo "c = " . $c . "<br/>";
echo "c = a => c = " . ($c = $a) . "<br/>";
echo "b += a => b = " . ($b += $a) . "<br/>";
echo "b -= a => b = " . ($b -= $a) . "<br/>";
echo "b *= a => b = " . ($b *= $a) . "<br/>";
echo "b /= a => b = " . ($b /= $a) . "<br/>";
echo "d %= e => d = " . ($d %= $e) . "<br/>";
?>
```

**OUTPUT**

```
c = 0
b += a => b = 60
b -= a => b = 10
b *= a => b = 500
b /= a => b = 10
d %= e => d = 1
```
Demo [Operator]
Control statement

• if statement - executes some code if one condition is true

• if...else statement - executes some code if a condition is true and another code if that condition is false

• if...elseif...else statement - executes different codes for more than two conditions

• switch statement - selects one of many blocks of code to be executed
if (condition) {
    code to be executed if condition is true;
}

if (condition) {
    code to be executed if condition is true;
} else {
    code to be executed if condition is false;
}

if (condition) {
    code to be executed if this condition is true;
} elseif (condition) {
    code to be executed if first condition is false and this condition is true;
} else {
    code to be executed if all conditions are false;

if (condition) {
    ...
} else {
    if (condition2) {
        ...
    } else {
        ...
    }
}
switch - case

switch (n) {
    case label1:
        code to be executed if n=label1;
        break;
    case label2:
        code to be executed if n=label2;
        break;
    case label3:
        code to be executed if n=label3;
        break;
    ...
    default:
        code to be executed if n is different from all labels;
}
Demo [Control statement]
Looping

• while - loops through a block of code as long as the specified condition is true

• do...while - loops through a block of code once, and then repeats the loop as long as the specified condition is true

• for - loops through a block of code a specified number of times

• foreach - loops through a block of code for each element in an array
while

while (condition is true) {
    code to be executed;
}

<?php
$x = 7;

while($x <= 8) {
    echo "The number is: $x <br>";
    $x++;
}
?>
do - while

do {
    code to be executed;
} while (condition is true);

<?php
$x = 7;

    do {
        echo "The number is: $x <br>";
        $x++;;
    } while ($x <= 8);
?>
for

for (init counter; test counter; increment counter) {
    code to be executed;
}

Parameters:
• init counter: Initialize the loop counter value
• test counter: Evaluated for each loop iteration. If it evaluates to TRUE, the loop continues. If it evaluates to FALSE, the loop ends.
• increment counter: Increases the loop counter value

```php
<?php
for ($x = 0; $x <= 10; $x++) {
    echo "The number is: $x <br>";
}
?>
```
The `foreach` loop works only on arrays, and is used to loop through each key/value pair in an array.

```
foreach ($array as $value) {
    code to be executed;
}
```

```php
<?php
$colors = array("red", "green", "blue", "yellow");

foreach ($colors as $value) {
    echo "$value <br>";
}
?>
```
Function

- 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.
Function (cont)

Syntax

```php
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!

```php
<?php
function writeHello() {
    echo "Hello world!";
}
writeHello(); // call the function
?>
```
Function is a variable ??
4 Types of Function

• Without parameter, and no return value.
  ```php
  function function_name() {
    echo "Halo";
  }
  ```

• Without parameter, and with return value.
  ```php
  function function_name() {
    return "This is return value";
  }
  ```

• With parameter, and no return value.
  ```php
  function function_name($param1, $param2) {
    echo "$param1 and $param2";
  }
  ```

• With parameter, and with return value.
  ```php
  function function_name($param1, $param2) {
    return ($param1 + $param2);
  }
  ```
Arrays

“An array is a special variable, which can hold more than one value at a time.”

```php
$car1 = "VW";
$car2 = "Mercedes";
$car3 = "Porsche";

Or

$cars = array("Volvo", "BMW", "Toyota");
echo count($cars);
```
Indexed Arrays

$cars = array();
$cars[0] = "BMW";
$cars[1] = "Mercedes";
$cars[2] = "Porsche";
Associative Arrays

$sex = array("Steven" => "male", "Margaret" => "female", "Man" => "male, "Woman" => "female");

$sex["Steven"] = ???
Loop through an associative Arrays

```
$sex = array("Steven" => "male", "Birgit" => "female", "Man" => "male, "Woman" => "female");

foreach($sex as $name => $n_sex){
    echo "Key = " . $name . " , Value = " . $n_sex . "<br/>";
```
Multidimensional Arrays

$cars = array
    (
        array("Mercedes",22,18),
        array("VW",15,13),
        array("Porsche",5,2),
    );