PHP
Part #1

Topics

- PHP Intro
- Basic PHP Syntax
- PHP Variables
- PHP Data Types
- PHP Operators

What is PHP?

- PHP is an acronym for "PHP Hypertext Preprocessor"
- PHP is a widely-used, open source scripting language
- PHP scripts are executed on the server
- PHP costs nothing, it is free to download and use

What is a PHP File?

- PHP files can contain text, HTML, CSS, JavaScript, and PHP code
- PHP code are executed on the server, and the result is returned to the browser as plain HTML
- PHP files have extension ".php"

What Can PHP Do?

- PHP can generate dynamic page content
- PHP can create, open, read, write, delete, and close files on the server
- PHP can collect form data
- PHP can send and receive cookies
- PHP can add, delete, modify data in your database
- PHP can restrict users to access some pages on your website
- PHP can encrypt data

Why PHP?

- PHP runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.)
- PHP is compatible with almost all servers used today (Apache, IIS, etc.)
- PHP supports a wide range of databases
- PHP is free. Download it from the official PHP resource: www.php.net
- PHP is easy to learn and runs efficiently on the server side

Basic PHP Syntax

- A PHP script can be placed anywhere in the document.
- A PHP script starts with

<?php and ends with **?>**:

```
<?
// PHP CODE HERE
?>
```

Example of a simple PHP file

```
<!DOCTYPE html>
<html>
<body>

<h1>My first PHP page</h1>

<?php
echo "Hello World!";
?>

</body>
</html>
```

Comments in PHP

- To let others understand what you are doing
- To remind yourself what you did
- PHP supports three ways of commenting
  - //
  - #
  - /* */

Example of a comment

```
<!DOCTYPE html>
<html>
<body>

<?php
// This is a single line comment

# This is also a single line comment

/*
This is a multiple lines comment block
that spans over more than
one line
*/
?>

</body>
</html>
```

PHP Case Sensitivity

- all variables are case-sensitive
- all user-defined functions, classes, and keywords (e.g. if, else, while, echo, etc.) are NOT case-sensitive

PHP Case Sensitivity

```
<!DOCTYPE html>
<html>
<body>

<?php
ECHO "Hello World!<br>";
echo "Hello World!<br>";
EcHo "Hello World!<br>";
?>

</body>
</html>
```

PHP Case Sensitivity

```
<!DOCTYPE html>
<html>
<body>

<?php
$color="red";
echo "My car is " . $color . "<br>";
echo "My house is " . $COLOR . "<br>";
echo "My boat is " . $coLOR . "<br>";
?>

</body>
</html>
```

PHP 5 Variables

- A variable starts with the $ sign, followed by the name of the variable
- A variable name must start with a letter or the underscore character
- A variable name cannot start with a number
- A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)
- Variable names are case sensitive ($y and $Y are two different variables)

Creating (Declaring) PHP Variables

- PHP has no command for declaring a variable.
- A variable is created the moment you first assign a value to it

Creating (Declaring) PHP Variables

```
<?php
$txt="Hello world!";
$x=5;
$y=10.5;
?>
```

PHP is a Loosely Typed Language

- PHP automatically converts the variable to the correct data type, depending on its value.
- In other languages such as C, C++, and Java, the programmer must declare the name and type of the variable before using it

PHP Variables Scope

- The scope of a variable is the part of the script where the variable can be referenced/used.
- PHP has three different variable scopes:
  - local
  - global
  - static

Local and Global Scope

```
<?php
$x=5; // global scope

function myTest() {
  $y=10; // local scope
  echo "<p>Test variables inside the function:</p>";
  echo "Variable x is: $x";
  echo "<br>";
  echo "Variable y is: $y";
}

myTest();

echo "<p>Test variables outside the function:</p>";
echo "Variable x is: $x";
echo "<br>";
echo "Variable y is: $y";
?>
```

PHP The global Keyword

```
<?php
$x=5;
$y=10;

function myTest() {
  global $x,$y;
  $y=$x+$y;
}

myTest();
echo $y; // outputs 15
?>
```

PHP The static Keyword

```
<?php

function myTest() {
  static $x=0;
  echo $x;
  $x++;
}

myTest();
myTest();
myTest();

?>
```

PHP 5 echo and print Statements

- There are some differences between echo and print:
  - echo - can output one or more strings
  - print - can only output one string, and returns always 1
- **Tip:** echo is marginally faster compared to print as echo does not return any value.

The PHP echo Statement

- can be used with or without parentheses: echo or echo().
  - **Display Strings**
  - **Display Variables**

echo for display strings

```
<?php
echo "<h2>PHP is fun!</h2>";
echo "Hello world!<br>";
echo "I'm about to learn PHP!<br>";
echo "This", " string", " was", " made";
?>
```

echo for display variables

```
<?php
$txt1="Learn PHP";
$txt2="crru.ac.th";
$cars=array("Volvo","BMW","Toyota");

echo $txt1;
echo "<br>";
echo "Study PHP at $txt2";
echo "My car is a {$cars[0]}";
?>
```

print for display strings

```
<?php
print "<h2>PHP is fun!</h2>";
print "Hello world!<br>";
print "I'm about to learn PHP!";
?>
```

print for display variables

```
<?php
$txt1="Learn PHP";
$txt2="crru.ac.th";
$cars=array("Volvo","BMW","Toyota");

print $txt1;
print "<br>";
print "Study PHP at $txt2";
print "My car is a {$cars[0]}";
?>
```

PHP 5 Data Types

- String
- Integer
- Floating point numbers
- Boolean
- Array
- Object
- NULL

PHP Strings

- You can use single or double quotes

```
<?php
$x = "Hello world!";
echo $x;
echo "<br>";
$y = 'Hello world!';
echo $y;
?>
```

PHP Integers

- An integer is a number without decimals.
- Rules for integers:
  - An integer must have at least one digit (0-9)
  - An integer cannot contain comma or blanks
  - An integer must not have a decimal point
  - An integer can be either positive or negative
  - Integers can be specified in three formats: decimal (10-based), hexadecimal (16-based - prefixed with 0x) or octal (8-based - prefixed with 0)

PHP Integers

- The PHP var_dump() function returns the data type and value of variables

```
<?php
$x = 5985;
var_dump($x);
echo "<br>";
$x = -345; // negative number
var_dump($x);
echo "<br>";
$x = 0x8C; // hexadecimal number
var_dump($x);
echo "<br>";
$x = 047; // octal number
var_dump($x);
?>
```

PHP Floating Point Numbers

- A floating point number is a number with a decimal point or a number in exponential form.

PHP Floating Point Numbers

```
<?php
$x = 10.365;
var_dump($x);
echo "<br>";
$x = 2.4e3;
var_dump($x);
echo "<br>";
$x = 8E-5;
var_dump($x);
?>
```

PHP Booleans

- Booleans can be either TRUE or FALSE

```
$x=true;
$y=false;
```

PHP Arrays

- An array stores multiple values in one single variable

```
<?php
$cars=array("Volvo","BMW","Toyota");
var_dump($cars);
?>
```

PHP Objects

- An object is a data type which stores data and information on how to process that data
- OOP

PHP Objects

```
<?php
class Car
{
    var  $color;
    function Car($color="green") {
      $this->color = $color;
    }
    function what_color() {
      return $this->color;
    }
}
function print_vars($obj) {
   foreach (get_object_vars($obj) as $prop => $val) {
     echo "\t$prop = $val\n";
   }
}
$herbie = new Car("white");
echo “herbie: Properties\n";
print_vars($herbie);
?>
```

PHP NULL Value

- The special NULL value represents that a variable has no value.
- NULL is the only possible value of data type NULL

```
<?php
$x="Hello world!";
$x=null;
var_dump($x);
?>
```

PHP 5 String Functions

- The strlen( ) function returns the length of a string, in characters.
- The strpos( ) function is used to search for a specified character or text within a string.

The PHP strlen ( ) function

```
<?php
echo strlen("Hello world!");
?>
```

The PHP strpos ( ) function

```
<?php
echo strpos("Hello world!","world");
?>
```

PHP Constants

- A constant is an identifier (name) for a simple value.
- The value cannot be changed during the script.
- A valid constant name starts with a letter or underscore (no $ sign before the constant name).
- **case-insensitive constant**

PHP Constants

```
<?php
define("GREETING", "Welcome to MIROT!");
echo GREETING;
?>
```

PHP 5 Operators

- PHP Arithmetic Operators
- PHP Assignment Operators
- PHP String Operators
- PHP Increment / Decrement Operators
- PHP Comparison Operators
- PHP Logical Operators
- PHP Array Operators

PHP Arithmetic Operators

PHP Arithmetic Operators

Operator	Name	Example	Result
+	Addition	$x + $y	Sum of $x and $y
-	Subtraction	$x - $y	Difference of $x and $y
*	Multiplication	$x * $y	Product of $x and $y
/	Division	$x / $y	Quotient of $x and $y
%	Modulus	$x % $y	Remainder of $x divided by $

PHP Arithmetic Operators

```
<?php
$x=10;
$y=6;
echo ($x + $y); // outputs 16
echo ($x - $y); // outputs 4
echo ($x * $y); // outputs 60
echo ($x / $y); // outputs 1.6666666666667
echo ($x % $y); // outputs 4
?>
```

PHP Assignment Operators

Assignment	Same as...	Description
x = y	x = y	The left operand gets set to the value of the expression on
x += y	x = x + y	Addition
x -= y	x = x - y	Subtraction
x *= y	x = x * y	Multiplication
x /= y	x = x / y	Division
x %= y	x = x % y	Modulus

PHP Assignment Operators

```
<?php
$x=10;
echo $x; // outputs 10
$y=20;
$y += 100;
echo $y; // outputs 120
$z=50;
$z -= 25;
echo $z; // outputs 25
$i=5;
$i *= 6;
echo $i; // outputs 30
$j=10;
$j /= 5;
echo $j; // outputs 2
$k=15;
$k %= 4;
echo $k; // outputs 3
?>
```

PHP String Operators

Operator	Name	Example	Res
.	Concatenation	$txt1 = "Hello" $txt2 = $txt1 . " world!"	Now
.=	Concatenation assignment	$txt1 = "Hello" $txt1 .= " world!"	Now

PHP String Operators

```
<?php
$a = "Hello";
$b = $a . " world!";
echo $b; // outputs Hello world!

$x="Hello";
$x .= " world!";
echo $x; // outputs Hello world!
?>
```

PHP Increment / Decrement Operators

Operator	Name	Description
++$x	Pre-increment	Increments $x by one, then returns $
$x++	Post-increment	Returns $x, then increments $x by on
--$x	Pre-decrement	Decrements $x by one, then returns $
$x--	Post-decrement	Returns $x, then decrements $x by o

PHP Increment / Decrement Operators

```
<?php
$x=10;
echo ++$x; // outputs 11
$y=10;
echo $y++; // outputs 10
$z=5;
echo --$z; // outputs 4
$i=5;
echo $i--; // outputs 5
?>
```

PHP Comparison Operators

Operator	Name	Example	Result
==	Equal	$x == $y	True if $x is equal to $y
===	Identical	$x === $y	True if $x is equal to $y, and the same type
!=	Not equal	$x != $y	True if $x is not equal to $y
<>	Not equal	$x <> $y	True if $x is not equal to $y
!==	Not identical	$x !== $y	True if $x is not equal to $y, not of the same type
>	Greater than	$x > $y	True if $x is greater than $y
<	Less than	$x < $y	True if $x is less than $y
>=	Greater than or equal to	$x >= $y	True if $x is greater than or e
<=	Less than or equal to	$x <= $y	True if $x is less than or equa

PHP Comparison Operators

```
<?php
$x=100;
$y="100";
var_dump($x == $y);
echo "<br>";
var_dump($x === $y);
echo "<br>";
var_dump($x != $y);
echo "<br>";
var_dump($x !== $y);
echo "<br>";
$a=50;
$b=90;
var_dump($a > $b);
echo "<br>";
var_dump($a < $b);
?>
```

PHP Logical Operators

Operator	Name	Example	Result
and	And	$x and $y	True if both $x and
or	Or	$x or $y	True if either $x or
xor	Xor	$x xor $y	True if either $x or both
&&	And	$x && $y	True if both $x and
||	Or	$x || $y	True if either $x or
!	Not	!$x	True if $x is not true

PHP Array Operators

Operator	Name	Example	Result
+	Union	$x + $y	Union of $x and $y (but duplicate k are not overwritten)
==	Equality	$x == $y	True if $x and $y have the same key/value pairs
===	Identity	$x === $y	True if $x and $y have the same key/value pairs in the same order a the same types
!=	Inequality	$x != $y	True if $x is not equal to $y
<>	Inequality	$x <> $y	True if $x is not equal to $y
!==	Non-identity	$x !== $y	True if $x is not identical to $y

PHP Array Operators

```
<?php
$x = array("a" => "red", "b" => "green");
$y = array("c" => "blue", "d" => "yellow");
$z = $x + $y; // union of $x and $y
var_dump($z);
var_dump($x == $y);
var_dump($x === $y);
var_dump($x != $y);
var_dump($x <> $y);
var_dump($x !== $y);
?>
```