college CDI

Server-side technologies (TCS)

Class 2

Working with variable (suite)

Variables scope

The scope of variables depends on the context in which it is used. A variable declared within a script but out of a function will be global. Although global, it will be usable within the script where it has been declared only and won't be usable in functions nor other scripts.

```
<?php
$a=1;                   // global (default)
function hello() {
        echo $a;        // local to function (no data to output)
}
?>
```

In order to access and/or create global variables from within a function, the ***key-word global*** must be used.

```
<?php
$a=1;                   // global (default)
$b=2;                   // global (default)
function plus() {
        global $a,$b;   // retrieves global variables
        global $result; // declares new global variable
        $result=$a+$b;  // assigns value to new variable
}
plus();                 // launches function
echo $result;           // outputs final data to screen : 3
?>
```

Another way to retrieve global variables consists into using ***pre-defined $GLOBALS associative array***.

```
<?php
$a = 1;
$b = 2;
function plus() {
        $GLOBALS["b"] = $GLOBALS["a"] + $GLOBALS["b"];
}
plus();
echo $b;                //b now equals 3
?>
```

Constants

Constants are used as identifiers to store data, just like variable, and they are global. Although, the values of constants cannot be changed but only used by programs and functions.

The function ***define()*** is used to assign values to the constants name. By convention, constants are written in upper case.

```
<?php
$price = 9.99;              // variable
define("TAX", 1.15);        // constant representing 15% tax to apply
echo $price * TAX;
?>
```

Some predefined constants

__FILE__:	Returns the name of the file including its full path.
__DIR__:	Returns the name of the directory of the file including its path.
__LINE__:	Returns the line number in the source file.

```
<?php
echo "<b>Full path :</b> " . __FILE__ . "<br />";
echo "<b>Folder path :</b> " . __DIR__ . "<br />";
echo "<b>Line number :</b> " . __LINE__ . "<br />";
?>
```

Result :
Full path : /Applications/MAMP/htdocs/test.php
Folder path : /Applications/MAMP/htdocs
Line number : 13

Booleans

Booleans supply a ***true (1)*** or ***false (0)*** value. Both true and false constants can be written lowercase or uppercase.

Boolean is *false* when the value is :

- 0 (zero) integer
- 0.0 (zero) float
- Empty or zero string (e.g. : " " or "0")
- Empty array
- Empty object
- NULL special constant

Boolean is *true* in all other circumstances.

Working with numbers

Integers

Integer numbers can be specified in decimal (base 10), hexadecimal (base 16), octal (base 8) or binary (base 2) notation and can be both positive or negative.

```
<?php
$a = 1234;              // decimal
$a = -123;              // a negative
$a = 0123;              // octal (equivalent to 83 decimal)
$a = 0x1A;              // hexadecimal (equivalent to 26 decimal)
$a = 0b11111111;        // binary (equivalent to 255 decimal)
?>
```

Operators

Usual operators and mathematics calculation order are in use in PHP just like in JavaScript (addition, subtraction, multiplication, division and modulo).

```
<?php
echo 10 + 2;
echo 10 - 2;
echo 10 * 2;
echo 10 / 2;
?>
```

Division of integers generating decimals in result

If the result of integers division results into an answer with decimals, the resulting number won't be considered as an integer but as a float (even if it is rounded).

Example :

```
<?php
var_dump(41/8);
?>
```

Result :

```
float(5.125)
```

Converting a value into an integer

To convert a value into an integer, ***(int)***, ***(intval)*** or ***(integer)*** casts can be used.

```
<?php
var_dump(41/8);                    // result : float(5.125)
var_dump((int) (41/8));            // result : int(5)
var_dump((integer) (41/8));        // result : int(5)
var_dump((intval) (41/8));         // result : int(5)
?>
```

Note

Converting a float to an integer rounds the value to the lowest integer.

Example :

```
<?php
var_dump(28/5);                    // result : float(5.6)
var_dump((int) (28/5));            // result : int(5)
?>
```

Important

The maximum length of an integer being of approximately 2 billion digits, any result longer than the maximum limit would generate an ***undefined*** error message (PHP prior to version 7).

Numerical strings

PHP treats numbers in a different way JavaScript does. Numbers in string do not always behave as strings but as integers.

```
<?php
echo 123;               // result : 123
echo 0123;              // result : 83 (octal numbers start with 0)
echo "0123";            // result : 0123 (string)
echo "0123" + "10";     // result : 133 (123 + 10)
echo "0123" + 10;       // result : 133 (123 + 10)
?>
```

Comparing values

PHP have problems comparing a number value with the result from operations. In such cases, you will need to use the ***round()*** function.

```
<?php
$x = 8 - 6.4;                          // equals 1.6
$y = 1.6;
var_dump($x == $y);                    // boolean = false
?>

<?php
$x = 8 - 6.4;                          // equals 1.6
$y = 1.6;
var_dump(round($x) == round($y));      // boolean = true
?>
```

Booleans and numbers

These two syntaxes provide the same result :

```
if($var==true) echo "ok";
if($var) echo "ok";
```

Important :

A negative value (e.g. : -2) is considered true (it isn't 0).

Booleans converted into integers are :

False = 0
True = 1

Booleans converted into floats are rounded to the lowest integer :

Examples :

1.9 = 1
0.8= 0

Math functions

ceil()	Rounds a number up to the nearest integer
exp()	Calculates the exponent of e
floor()	Rounds a number down to the nearest integer
lcg_value()	Returns a pseudo random number in a range between 0 and 1
is_float()	Returns true if a value is a float
is_int()	Returns true if a value is an integer
is_nan()	Checks whether a value is 'not-a-number'
is_numeric()	Returns true if a value is numerical
max()	Returns the highest value in an array, or the highest value of several specified values
min()	Returns the lowest value in an array, or the lowest value of several specified values
mt_rand()	Generates a random integer using Mersenne Twister algorithm
number_format()	Formats a number with groups of thousands
pi()	Returns the value of PI
pow()	Returns x raised to the power of y
rand()	Generates a random integer
round()	Rounds a floating-point number
sqrt()	Returns the square root of a number

Math constants

M_PI	Returns Pi
PHP_ROUND_HALF_UP	Round halves up
PHP_ROUND_HALF_DOWN	Round halves down

Getting the date

Date()

This function is used to format date.

Syntax
date(format)

Year
Y : 4 digits number (e.g : 2019)
y : 2 digits number (e.g. : 19)

Month
M : 3 letters string (e.g : Mar)
m : 2 digits number (e.g. : 03)

Day
D : 3 letters string (e.g : Sun)
d : 2 digits number, date of the month (e.g. : 31)

```
<?php
echo "<b>Today is :</b> " . date("l M d, Y") . "<br>";
echo "<b>Today is :</b> " . date("Y/m/d") . "<br>";
echo "<b>Today is :</b> " . date("Y.m.d") . "<br>";
echo "<b>Today is :</b> " . date("Y-m-d") . "<br>";
echo "<b>Today is :</b> " . date("l");                    // Lower case «L»
?>
```

Result :

Today is : Sunday Mar 31, 2019
Today is : 2019/03/31
Today is : 2019.03.31
Today is : 2019-03-31
Today is : Sunday

Getting the time

Date()

This function is also used to format time.

Syntax
date(timestamp)

Hours
H : 24-hour format (00 to 23)
h : 12-hour format (01-12)

Minutes
i : Minutes with leading zeros (00-59)

Seconds
s : Seconds with leading zeros (00-59)
d : 2 digits number, date of the month (e.g. : 31)

A.M. / P.M.
a : Ante / Post meridiem (lower or upper case)

Adjusting timezone

date_default_timezone_set("America/New_York")

```
<?php
date_default_timezone_set("America/New_York");
echo "<b>The time is :</b> " . date("H:i:s");
?>
```

Result :
The time is : 21:34:28

```
<?php
echo "<b>The time is :</b> " . date("h:i:sa");
?>
```

Result :
The time is : 09:35:33pm

Assignment 2 : Your first PHP script

In a HTML document, use PHP to obtain the following :

The user will enter different values for the main global variables (table diameter and side a and be of a triangle) and the program will output a message such as the following :

Date :
Saturday, April 6th 2019

Time :
10:25 am

Your table details :

Diameter :	*User enters diameter*
Circumference :	*Program output*
Surface area :	*Program output*

The length of your triangle hypotenuse :

Side a :	*User enters length of side a*
Side b :	*User enters length of side b*
Hypotenuse :	*Program output*