**PHP**

PHP: Hypertext Preprocessor. A server-side scripting language that can embed or contain text, HTML, CSS, and JavaScript along with PHP code. The PHP code is executed on the server, and the result is returned to the browser as plain HTML.

To *run* PHP, you should download the **XAMPP** server or similar applications, including a webserver, PHP, and a database like SQL. <https://www.youtube.com/watch?v=NG1SJGYYbTk>.

Basic PHP **syntax** is like that of HTML:

* Starts with `<?php` and ends with `?>`.
* Statements should end with `;`.
* Keywords like if and echo are not *case-sensitive*, but variable names like $name, $color, and $age are.
* Single-line *comments* can be made using `//` and `#`. Multiple-line comments use `/\* \*/`.

*Variables* in PHP start with the `$` sign, followed by the variable’s name. There is no need to declare variables in PHP.

* Variable names cannot start with a number but only with a letter or underscore. Can have only A-z, 0-9, and \_.
* Type declarations were added, so enabling strict requirement mode will throw a “Fatal Error” on type mismatch.
* **Scope** of variables is local, global, and static.
* *Global* variables can only be accessed outside of a function. *Local* ones only within that function.
* You can have local variables with the same name in different functions, as they are only recognized by the function in which they are declared.
* To access global variables from within a function, we use the `global` keyword inside that function.
* If $x is a global variable declared outside a function Test(), we use global $x, or $GLOBALS[‘x’], from within Test() to access it.
* `static` keyword keeps a local variable from being deleted after that function is executed. So, if you define two numbers and make one static, then increment them both, you will see that the static one shows the incremented value while the non-static variable shows the same value as before.

The difference between `echo` and `print` statements is minor. They are both used to output data to the screen. `echo` has no return value while `print` returns 1. `echo` can take multiple parameters while `print` can only take one argument. `echo` is the faster of these two. Multiple parameters mean `echo "these", "are", "multiple", "parameters. ";`.

PHP **Data Types**

String: $x = "Hello"; (' can also be used instead of “)

Integer: $x = 4353;

Float: $x = 10.23;

Boolean: $x = true;

Array: $attacks = array("SQL Injection","XSS","DoS");

Object: A class is a template, and an object is an instance of a class.

<?php  
class Car {  
  public $color;  
  public $model;  
  public function \_\_construct($color, $model) {  
    $this->color = $color;  
    $this->model = $model;  
  }  
  public function message() {  
    return "My car is a " . $this->color . " " . $this->model . "!";  
  }  
}  
  
$myCar = new Car("black", "Volvo");  
echo $myCar -> message();  
echo "<br>";  
$myCar = new Car("red", "Toyota");  
echo $myCar -> message();  
?>

This will print: "My car is a black Volvo!" in the 1st line and "My car is a red Toyota!" in the 2nd line.

NULL: It’s a particular data type with only one value – NULL. It doesn’t have any value assigned to it. If a variable is created without a value, it is automatically assigned a value of NULL.

Resource: This special data type isn’t an actual data type. It is a storing of a reference to functions and resources external to PHP. A typical example of using it is a database call. It is an advanced topic.

String Functions

strlen(): Returns the length of a string.

str\_word\_count() – Counts the words in a string.

strrev() – Reverses a string.

strops() – Search for the position of a text within a string.

str\_replace() – Replaces the text within a string.

Type casting

Casting a numerical value into another data type.

<?php  
// Cast float to int  
$x = 23465.768;  
$int\_cast = (int)$x;  
echo $int\_cast;  
  
echo "<br>";  
  
// Cast string to int  
$x = "23465.768";  
$int\_cast = (int)$x;  
echo $int\_cast;  
?>

PHP Math

The pi() function returns the value of PI (3.14). sqrt() returns the square root.

min() and max() functions find the lowest and highest value in a list of arguments.

abs() function returns the absolute value of a number. round() function rounds off to the closest integer.

rand() generates a random number. rand(10, 100) generates within 10 to 100 inclusively. <https://www.w3schools.com/php/php_ref_math.asp> is the list of other math functions.

PHP Constants

Constant’s value cannot be changed during the script. It doesn’t have a $ sign. define() is used to create constants.

Syntax of define() is: define(name, value, case-insensitive). Case-insensitive is false by default.

<?php  
define("GREETING", "Welcome to W3Schools.com!", true);  
echo greeting;  
?>

Even arrays can be defined as constants. Constants are, by default, global and can even be used inside functions.

<https://www.w3schools.com/php/php_operators.asp> (A little bit tough to remember)

*Operators* are used to performing operations on variables and values. Some important ones are % for modulus (remainder of $x divided by $y) and \*\* for exponentiation (raising $x to $y’th power). Assignment operators are written as x+=y is the same as x=x+y. It’s true for all other operators.

The difference between == and === is that == returns true if a={1,2} and b={2,1}, but === returns true only if a=b={1,2}.

Some essential comparison operators are == is equal, === is identical, != and <> is not equal. !== is not identical. <=> is the spaceship, which returns an integer less than, equal to, or greater than zero, depending on if $x is less than, equal to, or greater than $y. So, it returns -1, 0, 1 if the $x is less than, equal to, or greater than $y.

++$x (pre-increment) increments $x by one and then returns $x, so the output will be ‘initial $x’+1, $x++ (post-increment) returns $x and then increments $x, so the output will be just the ‘initial $x’.

And - used as both `and`, `&&`. Or - used as both `or`, `||`. Xor - used as `xor`. Not - used as `!`.

String operations are:

The `.` operator is used for the concatenation of two strings. `.=` is a concatenation assignment and appends two strings. The white spaces are removed if there is more than one.

Array operators are:

`+` is union of two arrays. `==` returns true if two arrays have the same key/value pairs. === returns true for an exact match. !== is for non-identity.

I did not understand PHP Conditional Assignment Operators, Ternary, and Null coalescing.

There is a part about “if”, “if else”, and “if elseif” which is very easy.

The `switch` statement selects one of many code blocks to execute. `break` prevents the code from automatically running into the next case. `default` is used if no case is matched.

<?php  
$favcolor = "red";  
  
switch ($favcolor) {  
  case "red":  
    echo "Your favorite color is red!";  
    break;  
  case "blue":  
    echo "Your favorite color is blue!";  
    break;  
  case "green":  
    echo "Your favorite color is green!";  
    break;  
  default:  
    echo "Your favorite color is neither red, blue, nor green!";  
}  
?>

PHP Loops

`while` - as long as the condition is true

`do…while` - loops once, then repeats as long as the condition is true

`for` - loops for a specified number of times

`foreach` - Used only on arrays to loop through each key/value pair.

`break` will stop the loop if the condition is met. `continue` will break only the iteration which meets the condition and continues with the next iteration in the loop.

PHP Functions

A block of statements that can be used repeatedly in a program and can only be executed by a call to the function. Function names must start with a letter or an underscore and are not case-sensitive. PHP converts the data type of a variable accordingly. To make it work as it should, we use `strict` (<?php declare(strict\_types=1);). Using strict will give a “Fatal Error” if a data type mismatch exists. If you are using `return`, using `: float` before the ({) of the function will give the specific data type for the return value. Did not understand Passing Arguments by Reference.

PHP Arrays

Arrays hold more than one value at a time. Format for PHP array is `$colors = array(“White”, “Grey”, “Black”);`. Indexing starts from 0 for the first element. Unlike the languages I learned until now, PHP has a new type of array called Associative arrays. They can use named keys assigned to the values of the array. The format is `$hacker = array(“Elliot”=> “Grey”, “Darlene”=> “Black”, “Tyrell”=> “White”);`. These values can be called with their named keys `echo “Elliot is a “ . $hacker[‘Elliot’] . “ hat hacker.”;`.