PHP : Hypertext Processor

- is a general purpose scripting language use for web development and can be embedded into HTML

- is a dynamic programming language that can connect to the databases

Sample php in html file

&lt;!DOCTYPE HTML&gt;

&lt;html&gt;

&lt;head&gt;

&lt;title&gt;Sample&lt;/title&gt;

&lt;/head&gt;

&lt;body&gt;

&lt;?php

echo "Hi!";

?&gt;

&lt;/body&gt;

&lt;/html&gt;

The opening tag for php is &lt;?php to be close by the ?&gt; tag while echo tag is used to output data on screen.

**I. COMMENT**

//This is a one line comment

/\*

This is a

block comment \*/

II. **SYNTAX**

· PHP commands ends with a semicolon.

echo “Hello World”;

Without the semicolon, PHP will interpret the other lines as a whole statement which would lead to errors.

· &36;=$ - PHP uses the dollar sign for all variables that will be declared.

&lt;TRY ME - - uploaded a file in github &gt;

&lt;?

&36;=$\_name = “Web Technologies”; //this is a string

Echo $\_name;

?&gt;

&lt;END OF TRY ME&gt;

III. Variables

String Variables

-to declare a string variable you need first the dollar sign and the variable name like : &36;=$user and after that the string should be placed on a quotation marks (specify mamaya).

&36;=$\_user = “Mary Ann”;

Also you can declare a variable with another variable.s

Numeric variable

&36;=$\_integer = 15;

&36;=$\_float = 15.5;

Rules for naming a variable:

1. A variable must start with a letter of the alphabet or the underscore “\_”

&36;=$hello //correct

&36;=$12hello //wrong

&36;=$\_hello //correct

1. A variable can only contain the symbol underscore. Other symbols are not recommended.

&36;=$\*hello //wrong

&36;=$\_12hello //correct

1. When a variable composes of two words, do no use whitespace rathen use an underscore.

&36;=$\_hello world //wrong

&36;=$\_hello\_world //correct

1. Variables in PHP are case-sensitive. $name is different from $Name.

IV. Arrays

Declaring an array:

&36;=$members = array(“one”, “two”, “three”);

Accessing an array per index:

&36;=$members[1]; //returns the string “one”

Declaring a two-dimensional array

&36;=$group = array(array(“one”, “two”, “three”),

array(“a”, “b”, “c”),

array(1, 2, 3)

);

Accessing a two-dimensional array:

&36;=$group[0][0]; //returns “one”

&36;=$group[2][1]; //returns 2

V. String

In php, a period “.” is used to concatenate a String.

<?php

&36;=$\_hello = “hello”;

echo &36;=$\_hello . ” world”;

?>

OUTPUT: hello world

The “.=” operation is used to append a variable to another variable

<?php

&36;=$\_hello = "hello";

&36;=$\_world = "world";

&36;=$\_hello .= $\_world;

echo &36;=$\_hello;

?>

OUTPUT: helloworld

Two types of String:

* Single quotation mark ‘ ‘

<?php

&36;=$\_hello = "hello";

&36;=$\_world = "world";

echo '&36;=$\_hello &36;=$\_world';

?>

OUTPUT: $\_hello $\_world

* Double quotation mark “ “

<?php

&36;=$\_hello = "hello";

&36;=$\_world = "world";

echo "&36;=$\_hello &36;=$\_world";

?>

OUTPUT: hello world

Variable Naming rules

There are Four rules in naming PHP variables

1. The names of the variables must start with alphabets or the underscore character ( \_ ).
2. The variables names should only have the alphabets (including the capital and and lowercase letters) and numbers (0-9), and the underscore character (\_).
3. In naming variable names, it may not contain spaces. It should be compromise of one word. If naming a variable with two words is should be seperated by an underscore character ( \_ ) (ex: $web\_systems\_and\_technology)
4. This variable name $webtech is not the same as $WEBTECH. In other words it is case-sensitive

Operators

-consist of mathematical, string, comparison, and logical commands (+, -, \*, / ).

Arithmetic Operators

* This operators are used to perform mathematics

|  |  |
| --- | --- |
| Operator | Description |
| + | Addition |
| - | Subtraction |
| \* | Multiplication |
| / | Division |
| % | Modulus (Division remainder) |
| ++ | Increment |
| -- | Decrement |

.

Assignment Operators

-These Operators are used to assign values to the variables

- the dot or period (.) is a string operator and this concatenate string

|  |  |  |
| --- | --- | --- |
| Operator | Example | Equivalent to |
| = | &36;=$j = 15 | &36;=$j = 15 |
| += | &36;=$j += 5 | &36;=$j = &36;=$j + 5 |
| -= | &36;=$j -= 2 | &36;=$j = &36;=$j - 2 |
| \*= | &36;=$j \*= 6 | &36;=$j = &36;=$j \* 6 |
| /= | &36;=$j /= 9 | &36;=$j = &36;=$j / 9 |
| .= | &36;=$j .= $p | &36;=$j = &36;=$j . &36;=$p |
| %= | &36;=$j %= 12 | $j = $j % 12 |

Comparison Operators

* These operators are used on a construct like *if* statements in which you need to compare two items.
* The = and == is different. = is an assignment operator and == is a comparison operator which compares if the two values or items are equal.

|  |  |
| --- | --- |
| Operator | Description |
| == | Is equal to |
| != | Is not equal to |
| &gt;=> | Is greater than |
| &lt;=< | Is lesser than |
| &gt;=>= | Is greater than equal to |
| &lt;=<= | Is lesser than equal to |

Logical operators

* These operators are used to combine the results of the comparison operators
* It takes an input of TRUE or FALSE and give a result of TRUE or FALSE

|  |  |  |
| --- | --- | --- |
| Operator | Description | Example |
| && | And | $j == 3 && $p == 6 |
| and | Low-precedence and | $j == 3 and $p == 6 |
| || | or | $j > 2 || $p > 4 |
| or | Low-precedence or | $j > 2 or $p > 4 |
| ! | not | !($j == $p) |
| xor | Exclusive or | $j xor $p |

* Using the && and || and their precedence is usually interchangeable. Is some cases to use the low-precedence of or and and, you might need a parenthesis to force the precedence.

Variable Assignment

* The syntax in assigning a value to a variable is Variable = value (ex $num = 3)
* In reassigning the value to another variable the syntax will be variable = variable.

String concatenations

* In string concatenation uses the period

Ex:

String types

* PHP supports of two types of strings. It is either be denoted by a quotation marks “ “ or single quotation mark or also known as the apostrophe ‘ ‘
* Difference of double quotation and apostrophe
  + apostrophe - preserves the exact contents

Ex: $txt = ‘assign variables with $ like $number’;

Echo $txt //output: assign variables with $ like $number

* Double Quotation - it is used if you want to include the value of the variable inside a string

Ex: echo “You have $count friends request” // You have 2 friends request

Variable typing

* PHP is very loosely typed language, this means that the variables doesn’t have to be declared.
* PHP always converts variables to the type required when it is accessed.

Constants

* It also holds information that will be accessed, but they maintain it constant.
* Once defined, the value is will not be altered.

Predefined Constants

* PHP has dozens of predefined constants.
* PHP has also magic constants that has two underscores, one at the beginning and one at the end.

Here are some examples of magic constants.

\_LINE\_ - the current number of the file

\_FILE\_ - the full path and the filename of the file

\_DIR\_ - The directory of the file

\_FUNCTION\_ the function name

\_CLASS\_ - the class name

\_METHOD\_ - the class method name

\_NAMESPACE\_ - the name of the current namespace

* These constants will become handy when it also comes in debugging.

ECHO vs PRINT

Echo and print is similar, they are used to perform some output.

echo

-purely PHP language construct

-outputs faster that print in general text

-doesn’t return value

-cannot be used on more complex expressions

print

-it can be use to output complex expressions

Functions

* Are used to separate out sections in the code to perform particular tasks.
* Placing codes in functions shortens the source code and it will make the source code more readable

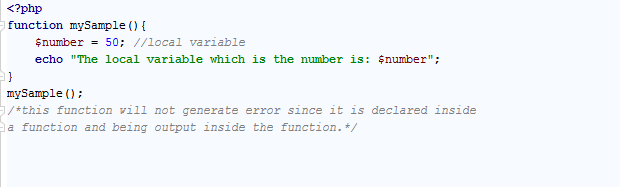
Variable Scope

* There are three types of variables scope namely, the local variables, global variables, and the static variables.

Local variables

-it is created within the function and can only be accessed by the function.

SAMPLE CODE

<local variable>

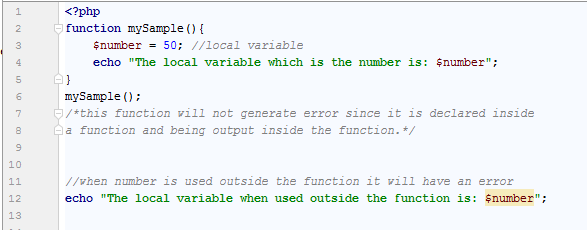
OUTPUT

C:\Users\sammy\Desktop\pictures for PHP notes\localvariableOUTPUT.PNG

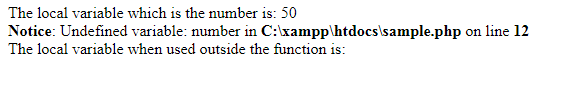
<local variable output>

But when a local variable will be echo outside the function it will return an error.

SAMPLE PROGRAM

<local variable with error.png>

OUTPUT

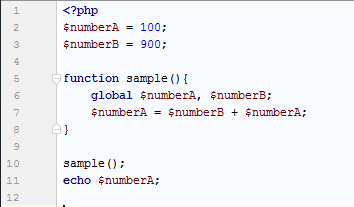


<local variable with error ouptut.png>

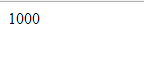
GLOBAL Variables

-global variables are variables that can be accessed by the whole code even inside the function.

-to declare the global variable it should be declared with the keyword global

SAMPLE CODE

<global variable.png>



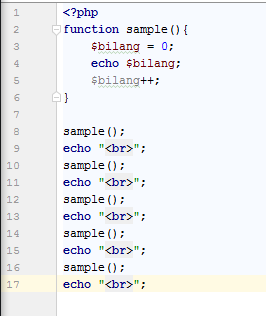
Output: sdf

<global variable output.png>

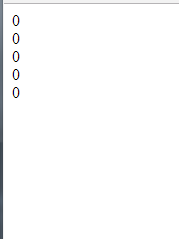
STATIC VARIABLES

* The value of the variable will be deleted when the program executed successfully.

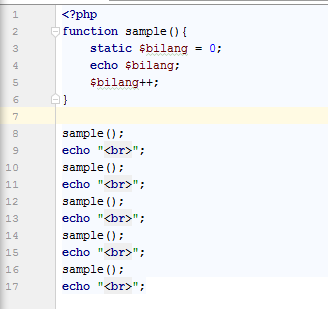
SAMPLE CODE

 <static variable na di static panay 0 ang display>

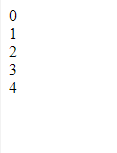
OUTPUT

 <output ng di na declare na static>

* To keep track on how many times the function is called, you need to declare the static variable

 <static variable na talaga.png>

Output

 <output ng static variable.png>

The Superglobal Variables

* Superglobal variables are provided with PHP environment
* It can be accessible everywhere

|  |  |
| --- | --- |
| SUPERGLOBAL NAME | Constants |
| &36;=$GLOBALS | All variables that are currently defined in the global scope of the script. The variable names are the keys of the array |
| &36;=$\_SERVER | Information such as headers, paths, and script locations. The entities in this array are created by the web server, and there is no guarantee that every web server will provide any of all these. |
| &36;=$\_GET | Variables passed to the current script via the HTTP Get method |
| &36;=$\_POST | Variables passed to the current script via the HTTP Post method |
| &36;=$\_FILES | Items uploaded to the current script via the HTTP Post method |
| &36;=$\_COOKIE | Variables passed to the current script via HTTP cookies. |
| &36;=$\_SESSION | Session variables available to the current script. |
| &36;=$\_REQUEST | Contents of information passed from the browser; by default, &36;=$\_GET, &36;=$\_POST, and &36;=$\_COOKIE |
| &36;=$\_ENV | Variables passed to the current script via the environment method. |

ACCESSING MySQL USING PHP

The Process

The following process of using MySQL with PHP is as follows

1. Connect to MySQL and select the database to use
2. Build a query string
3. Perform the query
4. Retrieve the results and output them to a web page
5. Repeat steps 2 to 4 until all desired data has been retrieved
6. Disconnect from MySQL

CREATING A LOGIN FILE

* Most websites that are developed in PHP has a multiple program that will access to MySQL that will need the login and password details.
* It is required a separate file to store the information. Example:

<?php //connectDB.php

&36;=$lc = ‘localhost’;

&36;=$db = ‘quiz’;

&36;=$un = ‘username’;

&36;=$pw = ‘password’;

?>

* You can replace the username and the password, depending on your MySQL.

CONNECTING TO MYSQL DATABASE

When the connectDB.php has been created. You can include it to any PHP file that needs accessing the database.

<?php

require\_once ‘connectDB’;

&36;=$conn = new mysqli(&36;=$lc, &36;=$db, &36;=$un, &36;=$pw);

if (&36;=$conn->connect\_error) die (&36;=$conn->connect\_error);

?>

Why use require\_once over include and require? require\_once will generate a fatal error if the file is not found. Also, the file will be in read only when it has not previously been included, which prevent wasteful duplicate disk access.

Sample database



Output:

COOKIES

Using Cookies in PHP

**Cookie** – is an item of data that a web server saves to your computer’s hard disk via a web browser.

-it contains almost alphanumeric information and can be retrieved from your computer and return to the server.

-the common uses of cookie are session tracking, maintaining data across multiple visits.

**Third-party cookies** – Multiple elements on a web page can be embedded form multiple domains, each of which can issue its own cookies.

-these are created by advertising companies in order to track users across multiple websites.

ANATOMY OF COOKIE

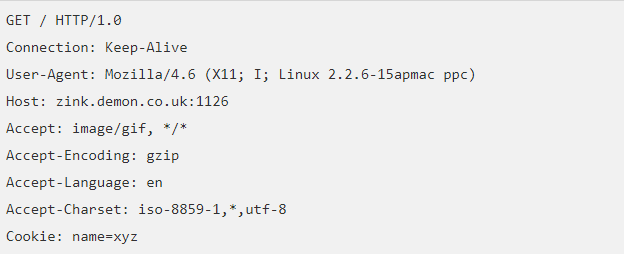
The cookie from PHP might look something like this:



from tutorials point

The Set-Cookie header contains the name and the value, it also contains the date, path and the domain. It also contains when it will expires. If the browser is set to store cookies, it will keep the information of the cookie until it expires.

If the user go to other pages in the browser and matches the path and domain cookie, it will resent the cookie to the server. In the browser header might also look something like this:



From tutorials point

Setting Cookies with PHP

PHP has provided a function setcookie().

setcookie(name, value, expire, path, domain, security);

name- sets the name of the cookie and it will be stored to HTTP\_COOKIE\_VARS, an environment variable

value-sets the value of the named variable.

Expiry – if it is time seconds the expiry of cookies is 00:00:00 GMT on day Month Year. the cookie will automatically expires when the web browser is closed.

Path- specifies the directories

Domain – used to specify the domain name

Security – it can be set to 1 to tell that the cookie should only be sent by secure transmission (HTTPS). It can be set to 0 if it is HTTP

SESSION

* It creates a temporary directory of the file on the server where registered session variables are being stored.
* Temporary file is determined setting the session.save\_path in the php.ini.
* PHP automatically retrieve session string identifier from the PHPSESSID cookie.
* The session ends if the user leave the site, then the server will automatically terminate the session after predetermined period of time.

Starting a PHP Session

-The PHP session start by starting or calling the function of session\_start(). It is recommended to start the function at the beginning of the page.

-the &36;=$\_SESSION[] this is an associative array where session variables are stored

-use the isset() function in checking if the session variable is set or not.

Destroying a PHP Session

-to destroy a session in PHP use the funciton session\_destroy().

-the session\_destroy() does’t need any arguments, one call can destroy all session variables.\

-to destroy a specific session variable use unsetI() function

Auto Session

To automatically start the session use set session.auto\_start variable to 1 in the php.ini file.

<https://www.tutorialspoint.com/php/php_sessions.htm>

<https://www.tutorialspoint.com/php/php_cookies.htm>

Learning PHP, MySQL & JavaScript with Jquery, CSS, and HTML5 author Ribin Rixon, 2014