Unit 1
Introduction to PHP

Contents

- Basic's of Web Concept
- Introduction to PHP
- Why use PHP
- Tools & Software Requirement
- Importance, Features and Application of PHP
- Compare with Other Language
- Create first script
- Variable and it's types
- Scopes of variable
- Constant
- Types of data
- Expression
- Types of operators
- Operators Precendence & Associativity

Web Browser

- Web browser is a client, program, software or tool through which we sent HTTP request to web server.
- The main purpose of web browser is to locate the content on the World Wide Web and display in the shape of web page, image, audio or video form.
- You can call it a client server because it contacts the web server for desired information.
- Microsoft Internet Explorer, Mozilla Firefox, Safari, Opera and Google Chrome are examples of web browser

HOW POPULAR WEB BROWSERS WORK

Mozilla Firefox

Google Chrome

Microsoft Internet Explorer

Apple Safari

Opera

Web Server

- Web server is a computer system or program, which provides the web pages via HTTP (Hypertext Transfer Protocol).
- IP address and a domain name is essential for every web server.
- Whenever, you insert a URL or web address into your web browser, this sends request to the web address where domain name of your URL is already saved.
- Then this server collects the all information of your web page and sends to browser, which you see in form of web page on your browser.

Web page

- A **web page** or **webpage** is a document commonly written in HTML (Hypertext Markup Language) that is accessible through the Internet.
- A web page is accessed by entering a URL address and may contain text, graphics, and hyperlinks to other web pages and files.
- This web page consists of several elements including CSS, images, and JavaScript.
- For example:

  Example.com is a website.

  Example.com/aboutus.html is a web page.

- **NOTE:** You can view all of the code by viewing the source code of the web page.

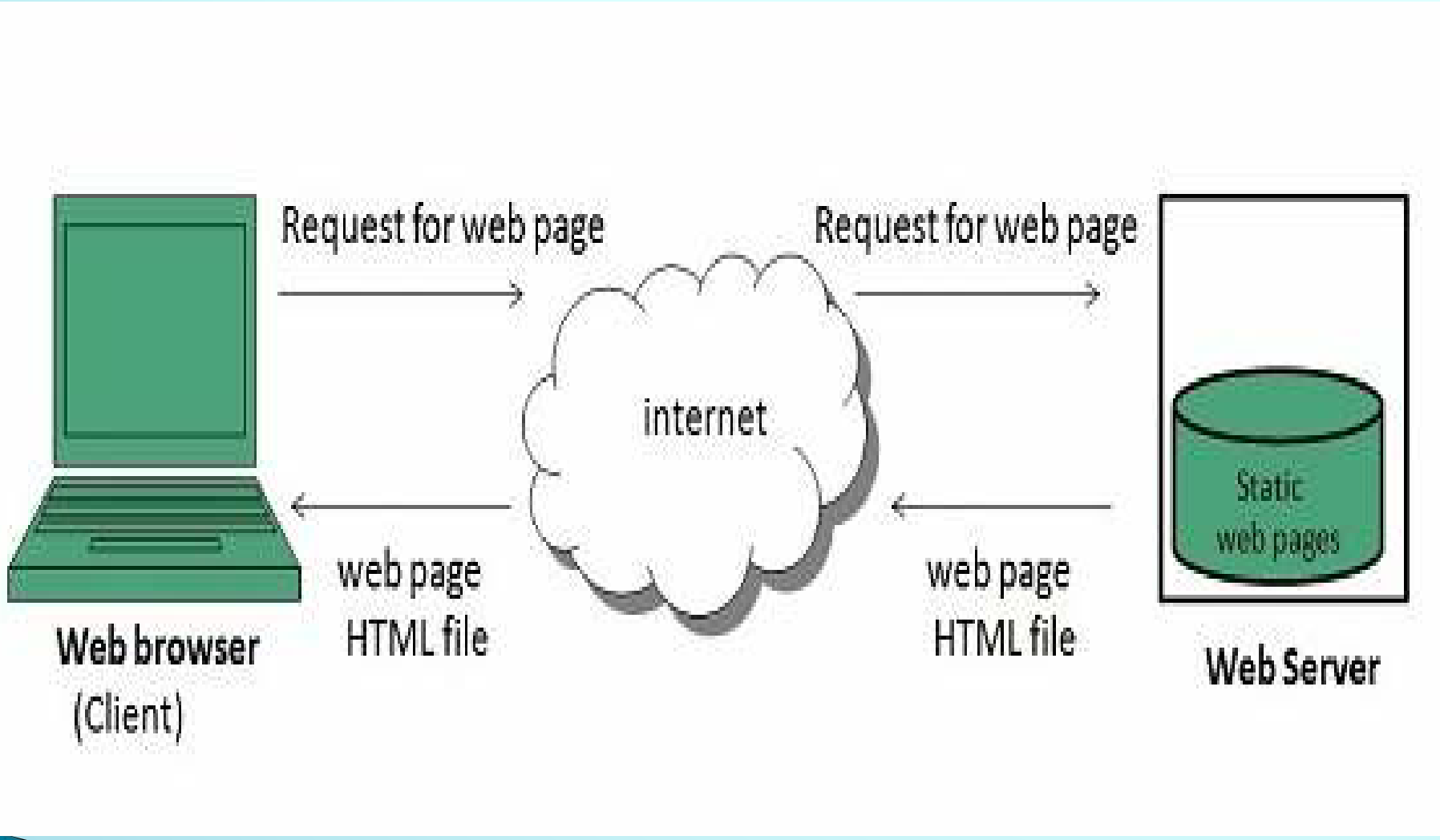
Static and Dynamic page

- A **static website** is something in which the data is fixed and doesn't change unless it has been added in the html code.
- A **dynamic website** is something which fetches data from the server. Name is PHP, Java, Angular, dotNet whatever all do the same thing.

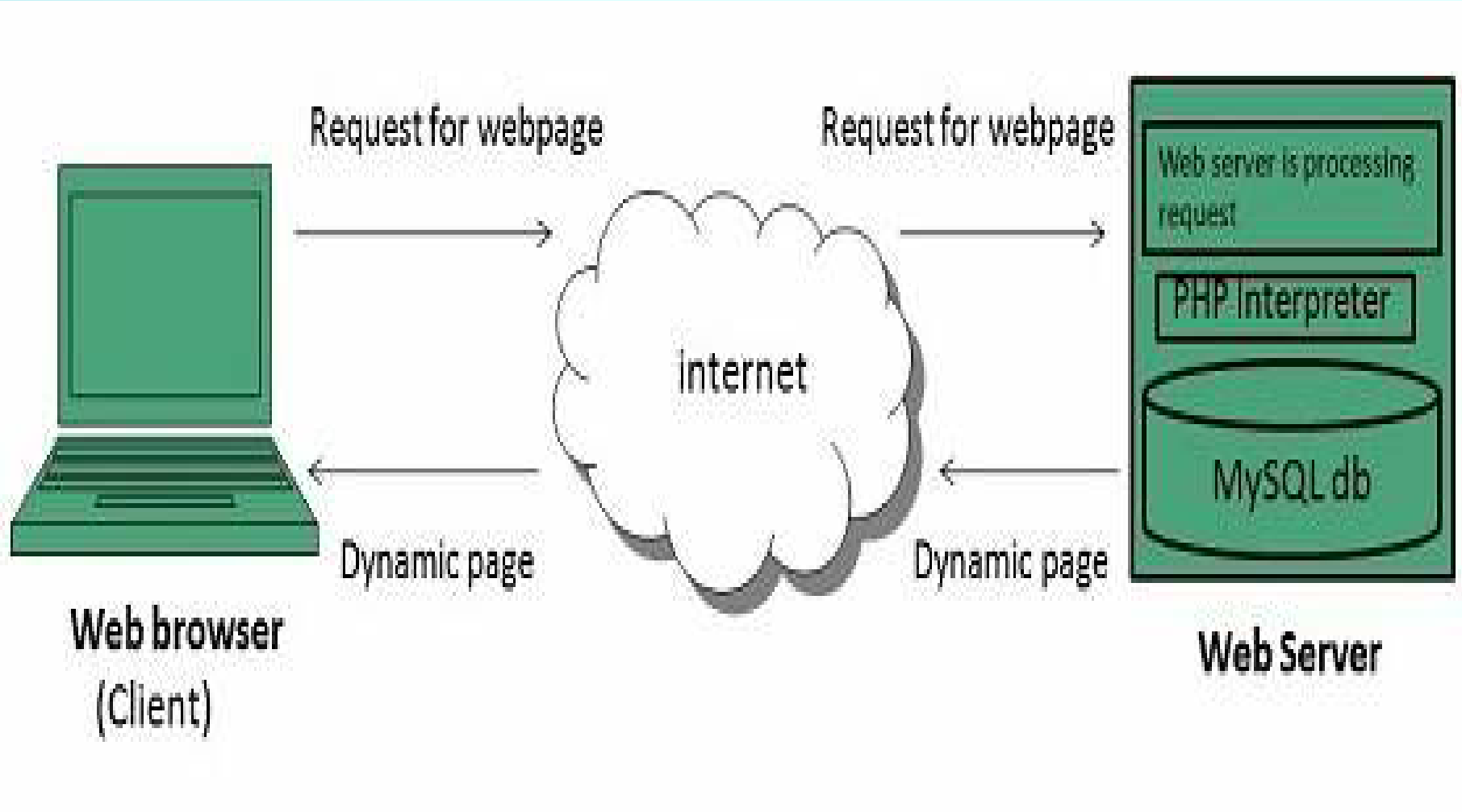
  The most common example is Quora.com, Facebook.com, YouTube etc., All are dynamic websites in which data is updated every moment.

- A dynamic website has a communication system to do the interaction between the browser and database. All visitors will not be served with same data and users are given with data based on users' activities.
- A static website provides the same content for all users.
- If the users can create an account on a website, it will be a considered as a dynamic website. All social media websites are dynamic websites.

Static web page

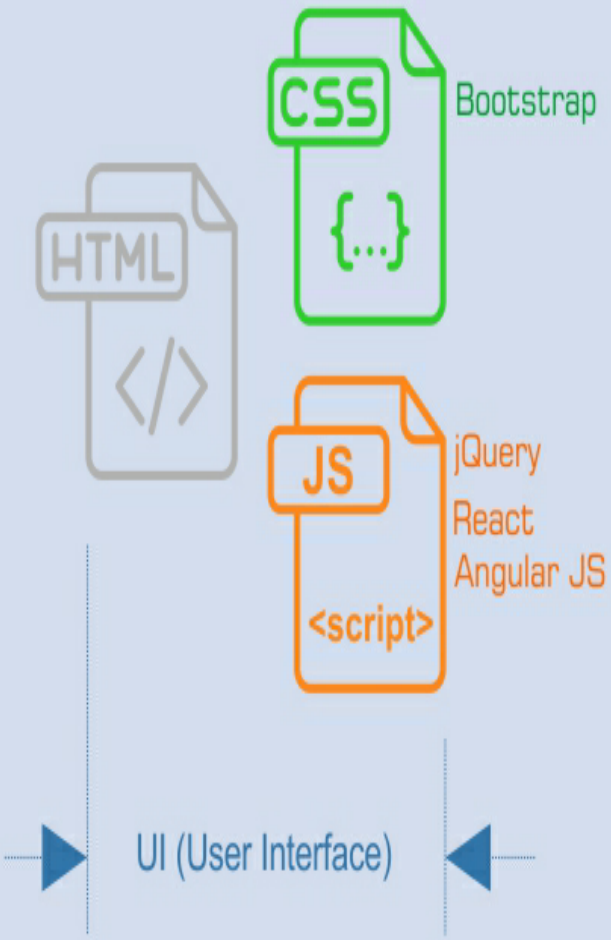


Dynamic web page



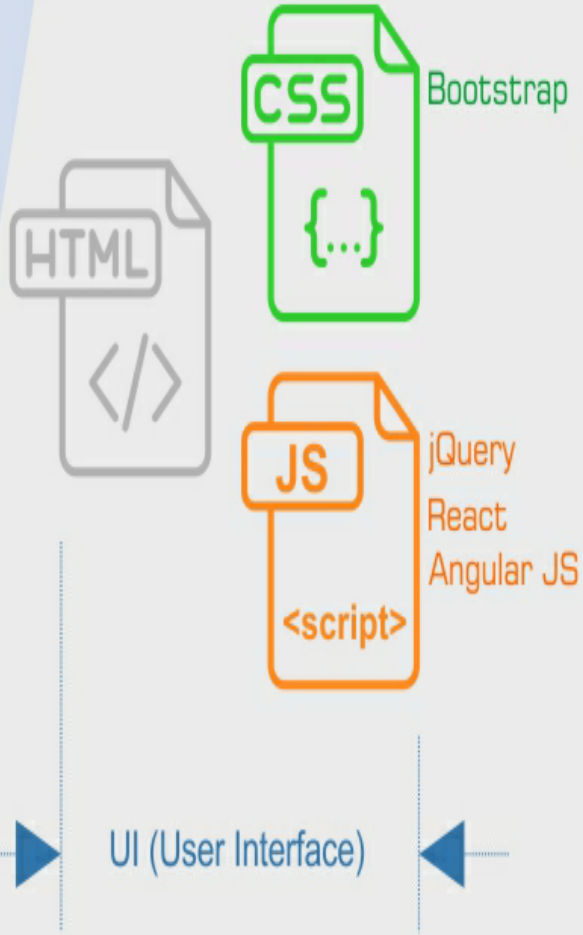
static website

client- side

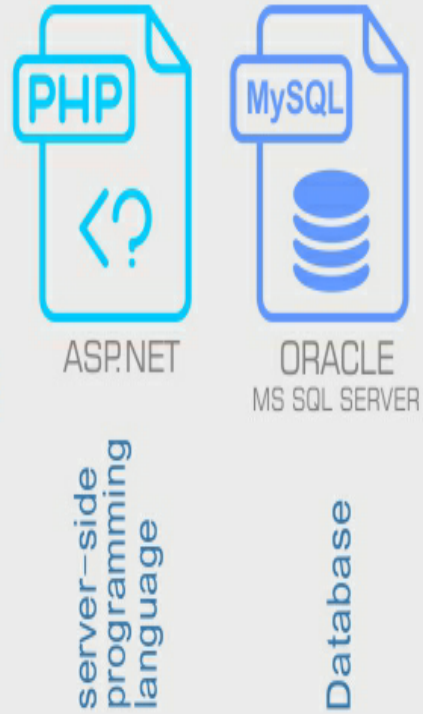


dynamic website

client- side



server- side



Website

- A Web site is a related collection of World Wide Web (WWW) files that includes a beginning file called a home page.
- A company or an individual tells you how to get to their Web site by giving you the address of their home page.
- From the home page, you can get to all the other pages on their site.
- A website refers to a central location that contains more than one web page. For example, Computer Hope is considered a website, which includes thousands of different web pages.

Introduction to PHP

- PHP stands for HyperText Preprocessor.
- It is called the Preprocessor because the PHP codes are processed before sending them to the browser.
- Initially, PHP was the abbreviation for Personal Home Page.

- History of PHP

- First version was developed by Rasmus Lerdorf (S/w Engg) for his personal use in 1994. By which he discovered that how many people visited his websites.

Php history

- 1st version 1994 (Rasmus Leadorf), in c programming language.
- 2nd version 1995
- PHP 3 was released in 1998, stands for "PHP: Hypertext Preprocessor "
- PHP 4 was released in 2000, with a new core, known as the Zend Engine [Zend stands for Zeev and Andi]
- PHP 5 was released in 2004, with the feature of OOPs.

[5.5.8 , released on 10th jan 2014]

What is PHP

- PHP is server-side scripting language specially designed for web development but also used as a general-purpose programming language.
- PHP is a server side scripting language because php code are executed on server.
- PHP is an interpreted language, i.e. there is no need for compilation.
- PHP is an object-oriented language.
- PHP is an open-source scripting language.
- PHP is simple and easy to learn language.

Scripting Language

- A script is a set of programming instructions that is interpreted at runtime.
- If code of programming language can embed with other language or integrate with other language called **Scripting Language**.
- The purpose of the scripts is usually to enhance the performance of any application.
- Server side scripts are interpreted on the server while client side scripts are interpreted by the client application.
- PHP is a server side script that is interpreted on the server while JavaScript is an example of a client side script that is interpreted by the client browser.

Programming Language Vs Scripting Language

Programming lang. has all the features needed to develop complete applications while S.L. Mostly used for routine tasks.

P.L. code has to be compiled before it can be executed while S.L. code is usually executed(interpreted) without compiling.

P.L. Does not need to be embedded into other languagesIs while S.L.usually embedded into other software environments.

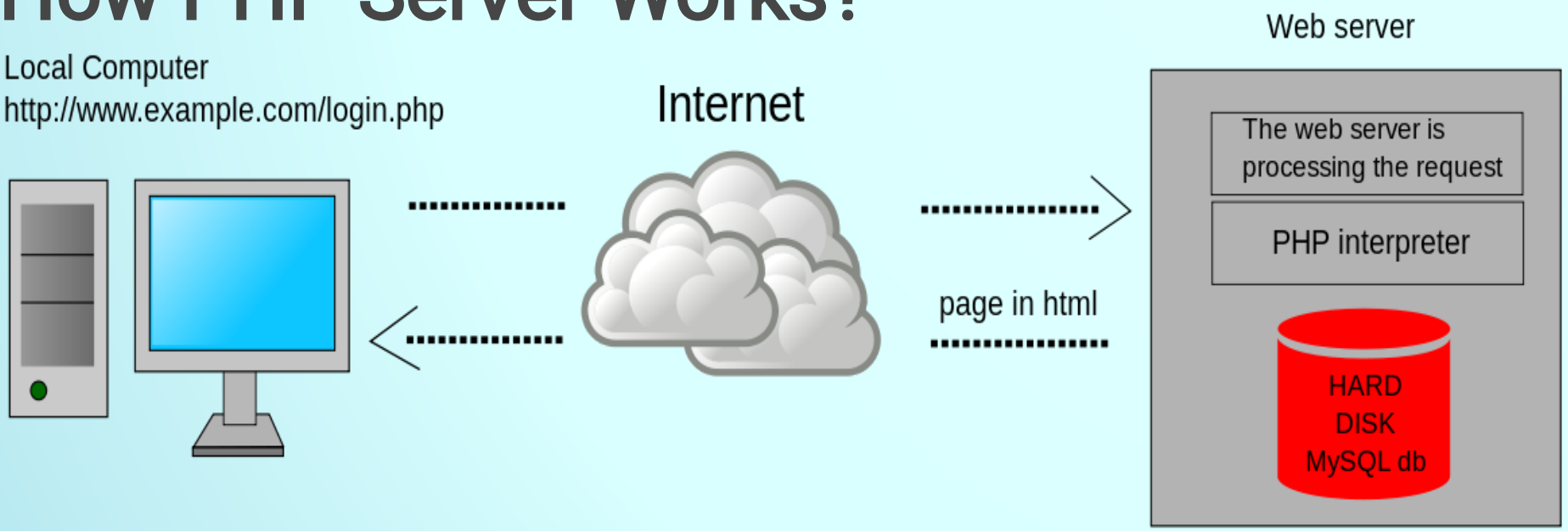
Client side scripting

- Used when the users browser already has all the code
- The Web Browser executes the client side scripting
- Cannot be used to connect to the databases on the web server
- Can't access the file system that resides at the web server
- Response from a client-side script is faster as compared to a server-side script

Server side scripting

- Used to create dynamic pages
- The Web Server executes the server side scripting
- Used to connect to the databases that reside on the web server
- Can access the file system residing at the web server
- Response from a server-side script is slower as compared to a client-side script

How PHP Server Works?



- URL is typed in the browser.
- The browser sends a request to the web server.
- The web server then calls the PHP script on that page.
- The PHP module executes the script, which then sends out the result in the form of HTML back to the browser, which can be seen on the screen.

Features & Importance of PHP

- It is open source
- Platform Independent
- Loosely Typed Language(because we did not have to tell PHP which data type the variable is hold. It automatically converts the variable to correct data type, depending on its value.)
- Case sensitive(in case of variable) but not(in case of function, classes.)
- Interact with web server
- PHP can be embedded with html, css, javascript.

Limitation: The server might be slow slowed down while multiple application are executed.

Software Required for PHP

- Text Editor
- Browser
- Server

php code start from:

```
<?php
  ……
  ……
?>
```

Save in wamp folder

PHP Server's

- WAMP 

(Windows ,Apache, MySQL, PHP engine)

- LAMP 

(Linux ,Apache, MySQL ,PHP engine)

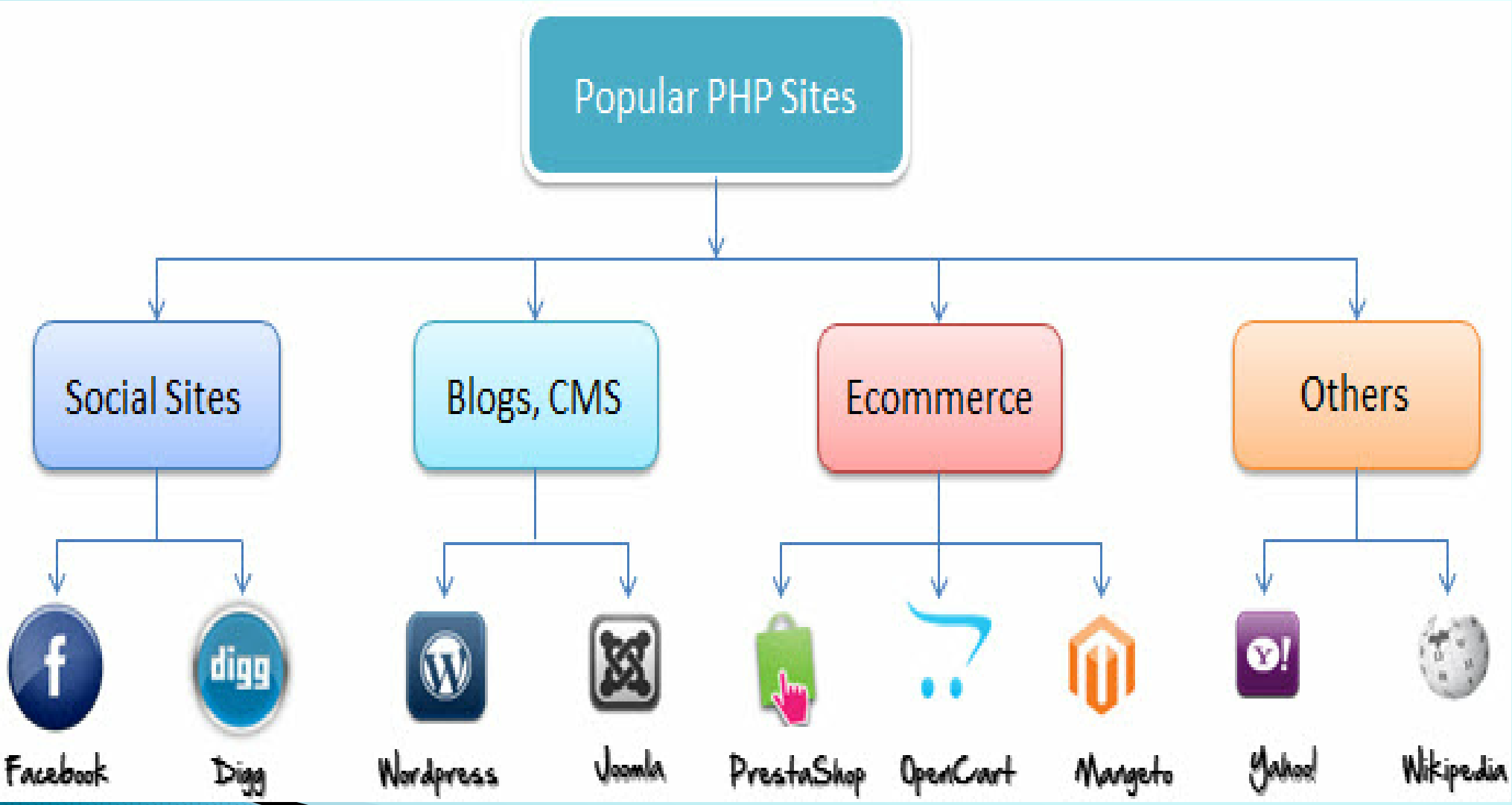
- XAMPP 

(Cross platform ,Apache, MySQL, PHP engine)

- Mamp

(Mac ,Apache, MySQL ,PHP engine)

Application of php programming language



PHP Variables

- A variable in PHP is a name of memory location that holds data. A variable is a temporary storage that is used to store data temporarily.
- In PHP, a variable is declared using $ sign followed by variable name.
- Syntax of declaring a variable in PHP is given below:
- $variablename=value;

PHP echo statement

- In PHP ‘echo’ statement is a language construct and not a function, so it can be used without paranthesis.

1. **Displaying Strings**:

```
<?php
echo "Hello,This is a display string example!";
?>
```

2. **Displaying Strings as multiple arguments**:

```
<?php
echo "Multiple ","argument ","string!";
?>
```

3. **Displaying Variables**

```
<?php
$str="hello string";
$x=200;
$y=44.6;
echo "string is: $str <br/>";
echo "integer is: $x <br/>";
echo "float is: $y <br/>";
Or
Echo “string is”.$str;
?>
```

PHP Variable: Sum of two variables

```
<?php
$x=5;
$y=6;
$z=$x+$y;
echo $z;
?>
```

Output:

11

PHP Variable: case sensitive

- In PHP, variable names are case sensitive. So variable name " color" is different from Color, COLOR, COLor etc.

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

Output:

My car is red

Notice: Undefined variable: COLOR in C:\wamp\www\variable.php on line 4

My house is Notice: Undefined variable: coLOR in C:\wamp\www\variable.php on line 5

My boat is

```
<html>
<body>
   <h1>Displaying text from PHP</h1>
   <br>
   <?php
      echo “<i>Welcome</i> <br>”;
      echo “<u>to</u><br>”;
      echo “<b>PHP</b>”;
      ?>
   </body>
   </html>
```

Create a simple HTML form and accept the user name and display the name through PHP echo statement.

```
<html> <head> <title></title></head> <body> <form
  method='POST'>
 <h2>Please input your name:</h2>
<input type="text" name="name">
 <input type="submit" value="Submit Name"> </form>
 <?php //Retrieve name from query string and store to
  a local variable
$name = $_POST['name'];
echo "<h3> Hello $name </h3>";
 ?> </body> </html>
```

$_POST

$_POST is a global variable (an assoc array) which holds data received from an HTTP POST request.

For example: if you submit a form with post method then all form fields will be received in $_POST.

```
<form method="post" action="somepage.php">
<input type="text" name="username" />
</form>
```

PHP Variable Scope

- It can be declared anywhere in the script:
  - Local: **The variable is only accessible from within the function that created it.**
  - Global : The variable is accessible from anywhere in the script.
  - A variable declared outside a function has global scope & can be accessed outside a function.

Ex. of local variable

```
<?php
  function calculate()
  {
   $count = 5;
          //will print 5; the value of local variable
   echo $count++;
  }
 ?>
```

//uses of local & global variable
//Ex 1

```
$x="joy";
function test()
{
   echo "var x inside function is: $x <br>";
}
test();
   echo "var x outside function is: $x";
?>
```

Notice: Undefined variable: x in **C:\xampp\htdocs\corephp\var1.php** on line **14**
var x inside function is:
var x outside function is: joy

//Ex 2

```
$x="joy";
function sayHello()
{
   $y="jarry";
   echo "Hello,$y!<br>";
}
sayHello();
echo "The value of global is: $x <br>";
echo "the value of local is : $y <br>";
?>
```

Hello,jarry!
The value of global is: joy

Notice: Undefined variable: y in **C:\xampp\htdocs\corephp\var1.php** on line **28**
the value of local is :

Access global variable from within function using global keyword

```
$g="joy";
function sayHello()
{
   $l="jarry";
   echo "Hello, $l <br>";

   global $g;
   echo "Hello, $g <br>";
}
sayHello();
```

o/p: Hello, jarry
Hello, joy

Static variable

- A static variable exist only in a local function scope, but it does not loose it's value when program execution leaves this scope.

```
function test()
{
   $a=0;
   echo $a;
   $a++;
}
test();
test();
```

o/p: 0 0

- This function is quite useless since every time it is called it sets $a to 0 and prints 0.
- To make useful counting function which will not lose track of current count, $a var is declared static.

```
function test()
{
   static $a=0;
   echo $a;
   $a++;
}
test();
test();
test();
```

o/p: 0 1 2

PHP Constants

- PHP constants are name or identifier that can't be changed during the execution of the script.

Let's see the syntax of define() function in PHP.

- Conventionally, PHP constants should be defined in uppercase letters.

  define(name, value, case-insensitive)

- name: specifies the constant name
- value: specifies the constant value
- case-insensitive: Default value is false. It means it is case sensitive by default.

File: constant1.php

```
<?php
define("MEDICAPS", “Welcome in Medicaps
   University");
echo  MEDICAPS;
?>
```

Output:

Welcome in Medicaps University

Ex 2

```
<?php
define("MEDICAPS","Welcome in Medicaps
   University",true); //not case sensitive
echo MEDICAPS;
echo medicaps;
?>
```

Ex 3

```
<?php
define("MEDICAPS", "Welcome in Medicaps
   University",false); // case sensitive
echo MEDICAPS;
echo medicaps;
?>
```

PHP constant: const keyword

```
<?php
const MESSAGE="Hello const ";
echo MESSAGE;
?>
```

PHP | Data Types

- PHP allows eight different types of data types. All of them are discussed below. The first five are called simple data types and the last three are compound data types:

1. **Integer** : Integers hold only whole numbers including positive and negative numbers, i.e., numbers without fractional part or decimal point
2. **Double**: Can hold numbers containing fractional or decimal part including positive and negative numbers.

```php
<?php
$val1 = 50;
$val2 = 654.26;
$sum = $val1 + $val2;
echo $sum;
?>
```

3. String : Hold letters or any alphabets, even numbers are included. These are written within double quotes during declaration. The strings can also be written within single quotes but it will be treated differently while printing variables.

EX;

```
<?php
$name = "Rasmus Lerdorf";
echo "The name of the php developer is $name <br>";
echo 'The name of the php developer is $name';
?>
```

o/p: The name of the php developer is Rasmus Lerdorf
The name of the php developer is $name

- **4. NULL**: These are special types of variables that can hold only one value i.e., NULL. We follow the convention of writing it in capital form, but its case sensitive.

```
<?php
  $nm = NULL;
  echo $nm;    // This will give no output
?>
```

- 5. **Boolean**: Hold only two values, either TRUE or FALSE.

```
<?php
 if(TRUE)
    echo "This condition is TRUE";
if(FALSE)
    echo "This condition is not TRUE";
?>
```

6. **Arrays**: Array is a compound data-type which can store multiple values of same data type.

7. **Objects**: Objects are defined as instances of user defined classes that can hold both values and functions.
8. **Resources**: These are basically used to store references to some function call or to external PHP resources. For example, consider a database call. This is an external resource.

gettype()

PHP **gettype()** only gives the type of variable.

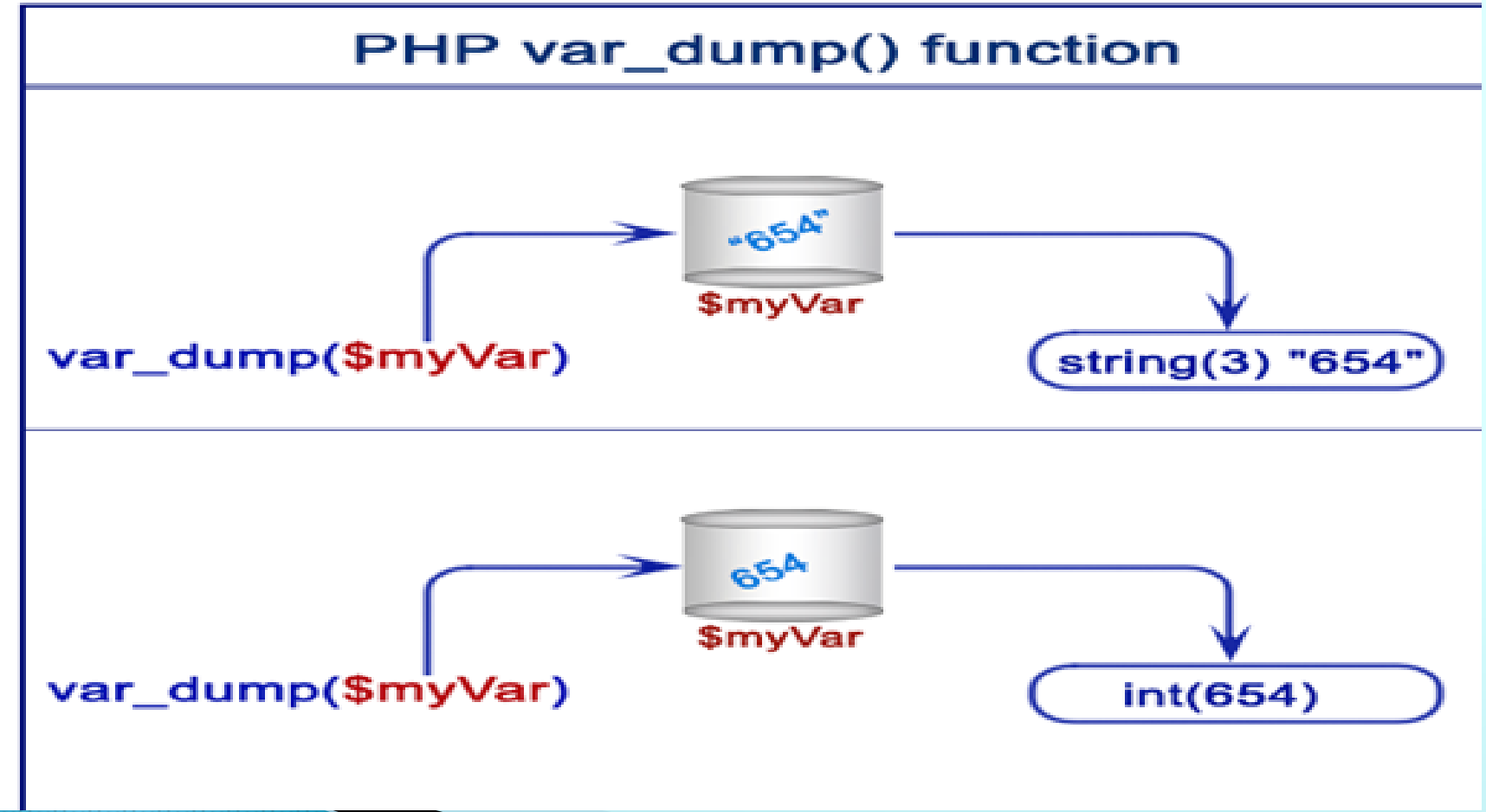
```
<?php
$a=10.50;
echo gettype($a);//double
echo "<br>";

$a="hello string";
echo gettype($a);//String

$b=true;
echo gettype($b);//boolean
?>
```

var_dump()

- PHP function **var_dump()** gives full details of the variable. It is used to display structured information (type and value) about one or more variables.



```
<?php
$var_name1=678;
$var_name2="a678";
$var_name3="678";
$var_name4="scripting language";
$var_name5=698.99;
echo var_dump($var_name1)."<br>";
echo var_dump($var_name2)."<br>";
echo var_dump($var_name3)."<br>";
echo var_dump($var_name4)."<br>";
echo var_dump($var_name5)."<br>";
?>
```

o/p: int(678)
string(4) "a678"
string(3) "678"
string(18) "scripting language"
float(698.99)

PHP|Operators

- Operators are used to perform operations on some values.
- Types of operators:

1. Arithmetic Operators
2. Logical or Relational Operators
3. Comparison Operators
4. Conditional or Ternary Operators
5. Assignment Operators
6. Spaceship Operators (Introduced in PHP 7)
7. Array Operators
8. Increment/Decrement Operators
9. String Operators

EX of logical operator

```
<?php
$x = 50;
$y = 30;
If ($x == 50 and $y == 30)
   echo "and Success \n";
if ($x == 50 or $y == 20)
echo "or Success \n";
if ($x == 50 xor $y == 20)
echo "xor Success \n";
if ($x == 50 && $y == 30)
echo "&& Success \n";
if ($x == 50 || $y == 20)
echo "|| Success \n";
if (!$z)
echo "! Success \n";
?>
```

Ex of Comparison Operators

1. != (Not Equal To) $x != $y

   Returns True if both the operands are not equal

2. <> (Not Equal To) $x != $y

Returns True if both the operands are unequal

3. ===(Identical) $x === $y

Returns True if both the operands are equal and are of the same type

4. !==(Not Identical) $x !== $y

Returns True if both the operands are unequal and are of different types

```php
<?php
   $a = 80;
   $b = 50;
   $c = "80";
   var_dump($a == $c) + "\n";
   var_dump($a != $b) + "\n";
   var_dump($a <> $b) + "\n";
   var_dump($a === $c) + "\n";
   var_dump($a !== $c) + "\n";
   var_dump($a < $b) + "\n";
   var_dump($a > $b) + "\n";
   var_dump($a <= $b) + "\n";
   var_dump($a >= $b);
?>
```

- o/p
- bool(true)
- bool(true)
- bool(true)
- bool(false)
- bool(true)
- bool(false)
- bool(true)
- bool(false)
- bool(true)

Conditional or Ternary Operators

- syntax
- $var = (condition)? value1 : value2;

Ex:

```
<?php
 $x = -12;
echo ($x > 0) ? 'The number is positive' : 'The
number is negative';
?>
```

Assignment operator

```
<?php
// simple assign operator
$y = 75;
echo $y;
  // add then assign operator
$y = 100;
$y += 200;
echo $y;
  // subtract then assign operator
$y = 70;
$y -= 10;
echo $y;
```

Increment/Decrement Operators

- These are called the unary operators as it work on single operands. These are used to increment or decrement values.

```
<?php
  $x = 2;
echo ++$x, " First increments then prints \n";
echo $x, "\n";

$x = 2;
echo $x++, " First prints then increments \n";
echo $x, "\n";
```

```
$x = 2;
echo --$x, " First decrements then prints \n";
echo $x, "\n";

$x = 2;
echo $x--, " First prints then decrements \n";
echo $x;

?>
```

ex

```
$a=10;
$b=2;
$x=0;
$x=a+b*a+10/2*a;
echo "value is“. x;
```

String Operators

- These are implemented over strings.

. (Concatenation)

Syntax **$x.$y** (Concatenated $x and $y)

.= (Concatenation and assignment)

Syntax **$x.=$y**

First concatenates then assigns, same as $x = $x. $y

Ex

```
<?php
   $x = “Welcome";
   $y = “to";
   $z = “PHP!!!";
   echo $x . $y . $z, "\n";
   $x .= $y . $z;
   echo $x;

?>
```

PHP Bitwise Operators

- a& b (And)
- Bits that are set in both *a* and *b* are set.
- Or
- Xor
- Not(~)
- a<<b

PHP Type Operator

- **instanceof** is the type operator used to determine if a PHP variable is an instantiated object of a class or not.

```
<?php
   class MyClass{}
   class OtherClass{}
   $var = new MyClass;
   var_dump($var instanceof MyClass);
      // prints bool(true)
   var_dump($var instanceof OtherClass);
      // prints bool(false) ?>
```

Understanding Precedence and Associativity

- operators have a set precedence, or order, in which they are evaluated. Operators also have associativity, which is the order in which operators of the same precedence are evaluated. This order is generally left to right (called *left* for short), right to left (called *right* for short), or *not relevant*.

Associativity	Operators
	HighestPrecedence
	()
Right	[]
Left	* / %
Left	+ - .
Left	<< >>
	< <= > >=
n/a	== != === !===
Left	&
Left	^
Left	|

```
Left    |
Left    &&
Left    ||
Left    ?  :
Left    = += -= *= /=
```

```php
<?php
   $n1 = 10;
   $n2 = 5;
   $n3 = 2;
   $ans = $n1 + $n2 * $n3;
   echo "$n1 + $n2 * $n3 = $ans<br />";
  $ans = ($n1 + $n2) * $n3;
  echo "($n1 + $n2) * $n3 = $ans<br />";
 ?
```

o/p: 10 + 5 * 2 = 20
(10 + 5) * 2 = 30

Expression

- An *expression* is a bit of PHP that can be evaluated to produce a value