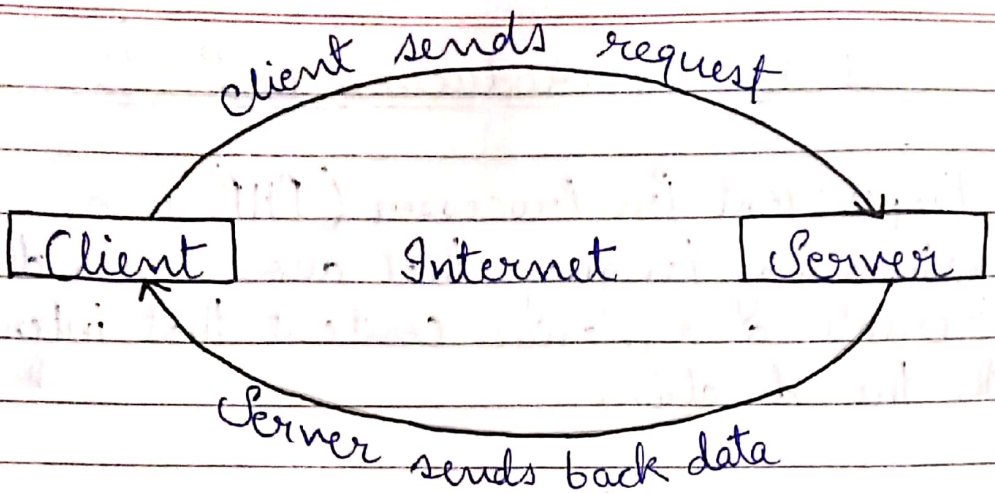
23/8/19

Ch-1 (Introduction to PHP)

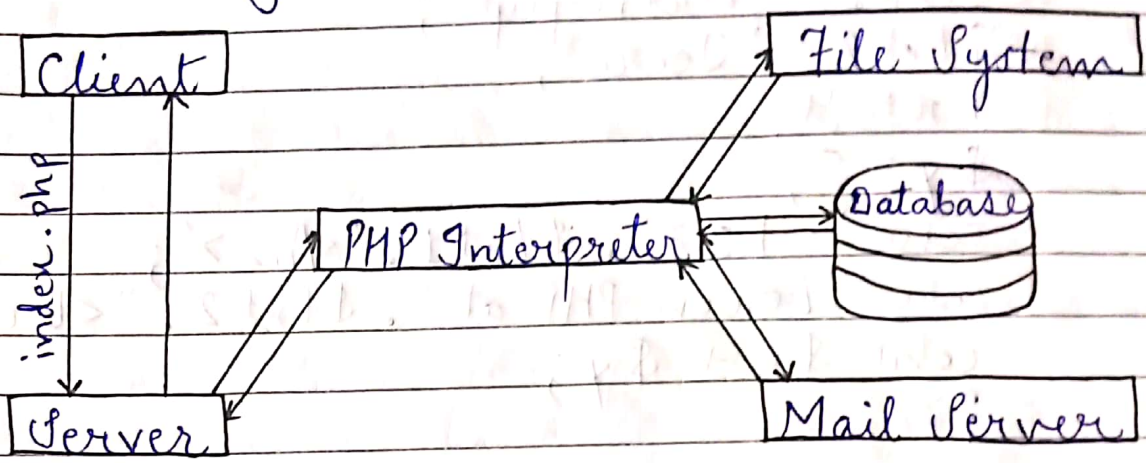
- The Hyper-Text Pre-Processor (PHP) is a programming language that allows web developers to create a dynamic content that interacts with the databases.
- It is a server-side scripting language designed for the web-development as well as for developing web-based softwares application.
- The First-version of PHP was introduced in 1994 by Rasmus lerdorf (earlier called, Personal Homepages).
- It is integrated with a number of databases like MySql, Oracle, Sybase, Informix, PostGreSql.
- It can create dynamic page content and also create, open, read, write, and delete the files on the server.
- Server is a high-configuration computer that are 24X7 online and stores large amount of data with less time consumption.
- It can collect form data, send and receive cookies, add, delete, modify data in your database and can encrypte the data.



- In PHP, Interpreter debugging (resolving error) is easy than compiler debugging because it works on server-side, thus it is known as server-side scripting language.

- Requirements to run PHP program :-
  (i) Laptop / Desktop
  (ii) Server
    - → WAMP (Windows, Apache, Mysql, PHP)
    - → XAMPP (Cross-platform, Apache, Mysql, Perl, PHP)
    - → MAMP (Mac, Apache, Mysql, PHP)
    - → LAMP (Linux, Apache, Mysql, PHP)
  (iii) Editor
    - → Dreamweaver
    - → Brackets
    - → Atom
    - → Notepad++
    - → Sublime Text
    - → Visual Stdio
  (iv) Browser

- PHP Working -



Ex1.
```
<?php
$x = 4;
$y = 5;
$z = $x + $y;
echo $z;
echo "The Sum is $z";
?>
```

Ex2.
```
<?php
$r = 5;
$a = 3.14 * r * r;
echo $a;
?>
```

Ex3.
```
<?php
$l = 6;
$b = 4;
$a = $l * $b;
echo $a;
?>
```

Ex4.
```
<?php
$txt1 = "Learn php";
$txt2 = "Jecrc";
$x = 4;
$y = 5;
echo "<h2>". $txt1. "</h2>";
echo "Learn PHP at ". $txt2. "<br>";
echo $x + $y;
?>
```

Ex5.)
```
<?php
$x = "Hi";
$y = "My name is Neha";
$p = 4;
$q = 5;
$t = $p + $q;
echo "<h2>". $x. " ". $y. "</h2>";
echo $t;
?>
```

Note: 1.)(.) is used as a concatenation operator.
2) Space is also a string type.

★ Variables:

- They are used for storing a value like text, strings, numbers or arrays.
- PHP is a loosely-type language i.e., it

automatically converts the variable to the correct datatype depending on its value.

- It starts with the $ sign followed by the name of the variable. A variable name must start with a letter or underscore.
- It cannot start with a number but it can contain alpha-numeric characters; They are case-sensitive i.e., $x ≠ $X.

★ Echo	Print
(i) It is used to display output on the screen but does not returns a value.	(i) It is used to display output on the screen but returns the value of 1.
(ii) It contains multiple arguments.	(ii) It contains single arguments.
(iii) It is faster in speed.	(iii) It is slower in speed.

★ Datatypes :

Variables can store data of different types and different datatypes can do different things! Basically, there are 8 types of datatypes available :

(i) String
(ii) Integer
(iii) Float also called as double

(iv) Boolean
(v) Array
(vi) Object
(vii) NULL
(viii) Resource
} used for advanced coding

* String:

Ex:
```
<?php
$x = "Hello World!";
var_dump ($x);
?>
```
Output → String (12) "Hello World!"

* Integer:

Ex:
```
<?php
$x = 415-4;
var_dump ($x);
?>
```
Output → int (415-4)

- Non-decimal number range between -2147483648 to +2147483647.
- Rules -
  (i) Must have atleast one digit and should not contain decimal values.
  (ii) Can be either positive or negative.

(iii) Can be specified in three formats -
(a) Decimal (10)
(b) Octal (8)
(c) Hexadecimal (16)

* Float :

Ex: <?php
$x = 10.45;
var_dump($x);
?>

Output → float(10.45)

* Boolean :

Ex: <?php
$x = True
var_dump($x);
?>

- It represents two possible states either true or false.

* Array :

Ex1. <?php
$num = array(10, 20, 30);
echo "First element : $num[0]";
echo "Second element : $num[1]";

echo "Third element : $num[2]";
?>

Output → First element : 10
Second element : 20
Third element : 30

Ex2. <?php
$cars = array ("Volvo", "BMW", "Audi");
var_dump ($cars);
?>

Output → array (3)
{
[0] → String [5] "Volvo"
[1] → String [3] "BMW"
[2] → String [4] "Audi"
}

- An array stores multiple values in one single variable, var_dump function returns the datatype of the variable.

* NULL : Ex: <?php $a = "Hello"; $a = NULL; var_dump ($a); ?> Output :- String ( )

- It is a special datatype which can have only one value. NULL variables can be empty by setting the value to NULL.

* PHP Resource : (used to establish connection)
$con = mysql_connect ("localhost", "root", "pwd", "dbname");

- This isn't an actual datatype, storing of a reference to functions and resources external to PHP. Eg: Database Connection and call.

* Object :

- It is a datatype which stores data and information on how to process data in PHP. An object must be explicitly declared. First, we must declare the class of the object using the class keyword.

* Operators :

(i) Arithmetic Operator
(a) + Addition
(b) - Subtraction
(c) * Multiplication
(d) / Division
(e) % Modulus
(f) ** Exponentation

Ex :

```
<?php
$a = 10;
$b = 2;
$w = $a + $b; echo $w;
$x = $a - $b; echo $x;
$y = $a * $b; echo $y;
$z = $a % $b; echo $z;
```

```
$p = $a / $b; echo $p;
$q = $a ** $b; echo $q;
?>
```

(ii) Comparison Operator

(a) == Equal To
(b) === Identical
(c) != Not Equal To
(d) <> Not Equal To
(e) !== Not identical
(f) > Greater than
(g) < Less than
(h) >= Greater than equal to
(i) <= Less than equal to

Ex:

```
$x = 80;
$y = 40;
$x == $y;
$x != $y;
$x <> $y;
$x === $y;
$x !== $y;
$x < $y;
$x > $y;
$x <= $y;
$x >= $y;
```

(iii) Logical Operator

(a) and

(b) or
(c) xor
(d) && and
(e) || or
(f) ! not

Ex: xor

x	y	Output
0	0	= 0
0	1	= 1
1	0	= 1
1	1	= 0

(iv) Increment and Decrement

(a) ++ increment
(b) -- decrement

Ex: $x = 4;

$y = $x++ + ++$x + $x++ + ++$x;
= 6 + 6 + 6 + 6
= 24

Solve:

y = ++x + x++ + ++x + x-- + --x + --x + x++ ;
z = y++ + ++y + ++x + x-- + y-- + --y;
t = z++ + --z + ++z + x++ + ++x + --x + y-- + ++y + y-- + y++ ;

(v) Assignment Operator

(a) =
(b) += Addition

(c) -= Subtraction
(d) *= Multiplication
(e) /= Division
(f) %= Modulus

Ex:

C += A ⟶ C = C + A
C -= A ⟶ C = C - A
C *= A ⟶ C = C * A
C = A

(vi) Spaceship Operator (<=>)

- It is used to compare two expressions. It returns -1, 0, +1 when the first expression is less than, equal, greater than the second expression.

greater than = 1
Less than = -1
equal = 0

Ex:
```
<?php
print (1 <=> 1); = 0
print (1 <=> 2); = -1
print ("a" <=> "a"); = 0
print ("b" <=> "a"); = 1
print (2.5 <=> 1.5); = 1
?>
```

(vii) Conditionals operator

(a) if

(b) if-else
(c) else-if
(d) switch case
(e) loop : for, while, do-while

Ex: if

```
<?php
$t = date("H");
if ($t < "20")
{
  echo ("Have a good day");
}
?>
```

Ex: Switch

```
<?php
$favcolor = "Red";
switch ($favcolor)
{
  case : red
  echo "Fav color is red"; break;
  case: green
  echo "Fav color is green"; break;
  default:
  echo "no match found";
}
?>
```

Note: String operators: (.) concatenation, (.=) concatenation assignment

* Loop :

(i) Initialization

(ii) Condition and statement
(iii) Increment / Decrement

(i) For (multiple initializations are possible)
Ex:

```
<html>
<body>
<?php
$a = 0;
$b = 0;
for ($i=0; $i<5; $i++)
{
  $a += 10;
  $b += 5;
}
?>
</body>
</html>
```

Output :- a = 50 and b = 25

(ii) while
Ex:

```
<html>
<body>
<?php
$i = 0;
$num = 50;
```

```
while ($i < 10)
{
    $num--;
    $i++;
}
?>
</body>
</html>
```

(iii) Do-while

Ex:

```
do
{
    $num--;
    $i++;
}
while ($i<10);
```

* While	Do-while
(i) Loops through a block of code if ~~and~~ as long as specified condition is true.	(i) Loops through a block of code once & repeats the loop as long as the specified condition is true
(ii) Entry controlled loop.	(ii) Exit controlled loop.
(iii) Condition is at the top.	(iii) Condition is at the bottom.
(iv) There is no semi-colon at the end of while	(iv) There is a semi-colon at the end of do-while.
(v) This should be used when condition is more imp.	(v) This should be used when process is more imp.

* Examples of while, do-while and for:

(i) While

```
<?php
$i=1;
while ($i<=3)
{
  $i++;
  echo "The number is ".$i."<br>";
}
?>
```

Output :- 2
3
4

(ii) Do-while

```
<?php
$i=1;
do
{
  $i++;
  echo "The number is ".$i."<br>";
}
while ($i<=3);
?>
```

Output :- 2
3
4

(iii) for

```
<?php
for ($i=1; $i<=3; $i++)
{
  echo "The number is ".$i."<br>";
}
?>
```

Output :- 1
2
3

* Foreach Loop :

- A foreach loop or statement is used to loop through arrays for each parse the value of the current array element is assigned to $value and array pointer is moved by one and in the next parse, next element will be processed.

Ex 1.
```
<? php
$a = array (1, 2, 3, 4, 5);
foreach ($a as $value)
{
echo "Value is $value". "<br>";
}
?>
```

output :- Value is 1
Value is 2
Value is 3
Value is 4
Value is 5

Ex 2.
```
<? php
$a = array ("Red", "Green", "Blue");
foreach ($a as $value)
{
echo "Value is $value". "<br>";
}
?>
```

Output :- Value is Red
Value is Green
Value is Blue

★ Break	Continue
Break is used to terminate the execution of loop prematurely, after coming out of the loop immediate statement to the loop will be executed. Break escapes the entire logic of other iteration.	Continue is used to hault current iteration of the loop but it does not terminate the loop. It skips out of the particular instance of the loop and moved onto the next one for parse encountering the continue statement, rest of the loop code is skipped and next parse starts.

★ Examples of Break and Continue:

(i) Break

Ex1.

```
for ( $i=0 ; $i<=5 ; $i++)
{
  if ($i==2)
  {
    break;
  }
  echo $i;
  echo "<br>";
}
```

```
echo "End of for loop";
```

Output :- 0
1
End of for loop

Ex 2.

```
<html>
<body>
<?php
$i = 0;
while ($i < 10)
{
  $i++;
  if ($i == 3)
    break;
}
echo "Loop stopped at i= ". $i;
?>
</body>
</html>
```

Output :- Loop stopped at i = 3

(ii) Continue

Ex 1.

```
for( $i=0; $i <= 5; $i++)
{
  if ($i == 2)
  {
```

```
    continue;
}
echo $i;
echo "<br>"; }
echo "End of for loop";
```

Output :- 0
1
3
4
5
End of for loop

Ex2:
```
<?php
$a = array (1, 2, 3, 4, 5);
foreach ($a as $value)
{
    if ($value == 3)
    {
        continue; }
        echo "Value is $value"."<br>";
}
?>
```

Output :- 1
2
4
5

★ Range :

- This is an inbuilt array function in PHP which is used to create an array of elements from low to high or vice-versa. and its default value is 1.

Syntax:

```
array = range (low, high, step)
```

Ex1:

```
<?php
$a = range (0, 6);
foreach ($a as $value)
{
  echo $value ;
}
?>
```

Output :- 0 1 2 3 4 5 6

Ex2:

```
<?php
$a = range (0, 100, 20);
foreach ($a as $value)
{
  echo $value;
}
?>
```

Output :- 0 20 40 60 80 100

Ex3:
```
<?php
$a = range ("p", "a");
foreach ($a as $value)
{
  echo $value;
}
?>
```

Output :- p - - - - - - a

* Pattern Printing :

1.)
```
*
* *
* * *
```

⇒
```
for (i=0; i<=2; i++)
{
  for (j=0; j<=i; j++)
  {
    echo "*";
  }
  echo "<br>";
}
```

2.)
```
* * *
* *
*
```

⇒
```
for (i=0; i<=2; i++)
{
  for (j=3-i; j>=1; j--)
  {
    echo "*";
  }
  echo "<br>";
}
```

3) A
B B
C C C

⇒
```
$a = range ("A", "C");
for (i=0; i<2; i++)
{
    for (j=0; j<=i; j++)
    {
        echo a[i];
    }
    echo "<br>";
}
```

★ Reverse:

Ex:
```
<?php
$num = 121;
$revnum = 0;
while ($num > 1)
{
    $rem = $num % 10;
    $revnum = ($revnum * 10) + $rem;
    $num = ($num / 10);
}
echo "Reverse of 121 is $revnum";
?>
```

Note: Important Topics → prime no., reverse, armstrong no., Fibonnaci series, palindrome no., factorial, even/odd, table of a number, swap two number (with & without using 3rd variable), leap year, pattern.

* Functions:

- It is a block of code that can be executed whenever we need it. There are two types of functions -
  (i) User-Defined
  (ii) Built-in (numeric, string, etc.)

- Rules of functions -
  (i) It should start with the word function.
  (ii) The name can start with a letter or underscore.
  (iii) The body of function should be written in brackets.

Ex1. Adding

```
<?php
function add_no()
{
  echo "Hello";
}
add_no();
?>
```

Ex2. <?php

```
function write_my_name ($fname)
{
    echo "My Name is ". $fname. "<br>";
}
write_my_name ();
?>
```

Ex3. Name with punctuation

```
<?php
function write_my_name ($fname, $punctuation)
{
    echo "My name is". $fname. $punctuation.
        "<br>";
}
fname ("Neha", ".");
?>
```

Ex4. function with return type or parameters

```
<?php
function add ($x, $y)
{
    $total = $x + $y;
    return $total;
}
echo "4+6 = ". add (4, 6);
?>
```