Introduction to PHP

Objective

In this chapter learner able to understand:

- What is web?
- What is script?
- History, Future Scope, Benefits and Importance of PHP in Web World
- What is PHP?
- Data types
- Type juggling
- PHP Variables
- Operators
- Decision Making using PHP

What is web?

- The World Wide Web (WWW) is an open source information space where documents and other web resources are identified by URLs, interlinked by hypertext links, and can be accessed via the Internet.
- It permits the computer users to view and locate and multimedia-based documents.
- It is developed by Tim Berners Lee of CERN (the European Laboratory for Particle Physics) in 1990.
- The Internet combines communications technologies and computing.
- Small businesses and Individuals can receive worldwide exposure on the cloud (Internet).
- Now a days there are millions of websites available on the internet which provides a powerful medium for people can stay in touch to each-other through social networking sites, make online shopping, payments through web.

What is Script?

A computer script is a list of commands that are executed by a certain program or scripting engine. Scripts may be used to automate processes on a local computer or to generate Web pages on the Web .

For example:

DOS scripts and VB Scripts may be used to run processes on Windows machines, while AppleScript scripts can automate tasks on Macintosh computers. ASP, JSP, and PHP scripts are often run on Web servers to generate dynamic Web page content.

Scripting Language

“A scripting language is used to write the scripts on the web pages.”

- These scripts are actually the programs embedded into HTML code and these web pages are run on the client computers.
- These scripts/programs are interpreted by the browsers.
- There are different types of these scripting languages that are commonly used
- today like JavaScript, vbscript , php, perl etc

Father Of PHP

Rasmus Lerdorf , who wrote the original Common Gateway Interface component in 1994 .

Rasmus Lerdorf is created PHP at that time PHP stands for "Personal Home Page".

Including tasks such as displaying his resume and recording how much traffic his page was receiving.

Rasmus Lerdorf

Two programmers, **Zeev Suraski** and **Andi Gutmans** , rebuilt PHP's core, releasing the updated result as PHP/FI 2 in 1997. The acronym was formally changed to

PHP: HyperText Preprocessor

Andi Gutmans

Zeev Suraski and **Andi Gutmans** who

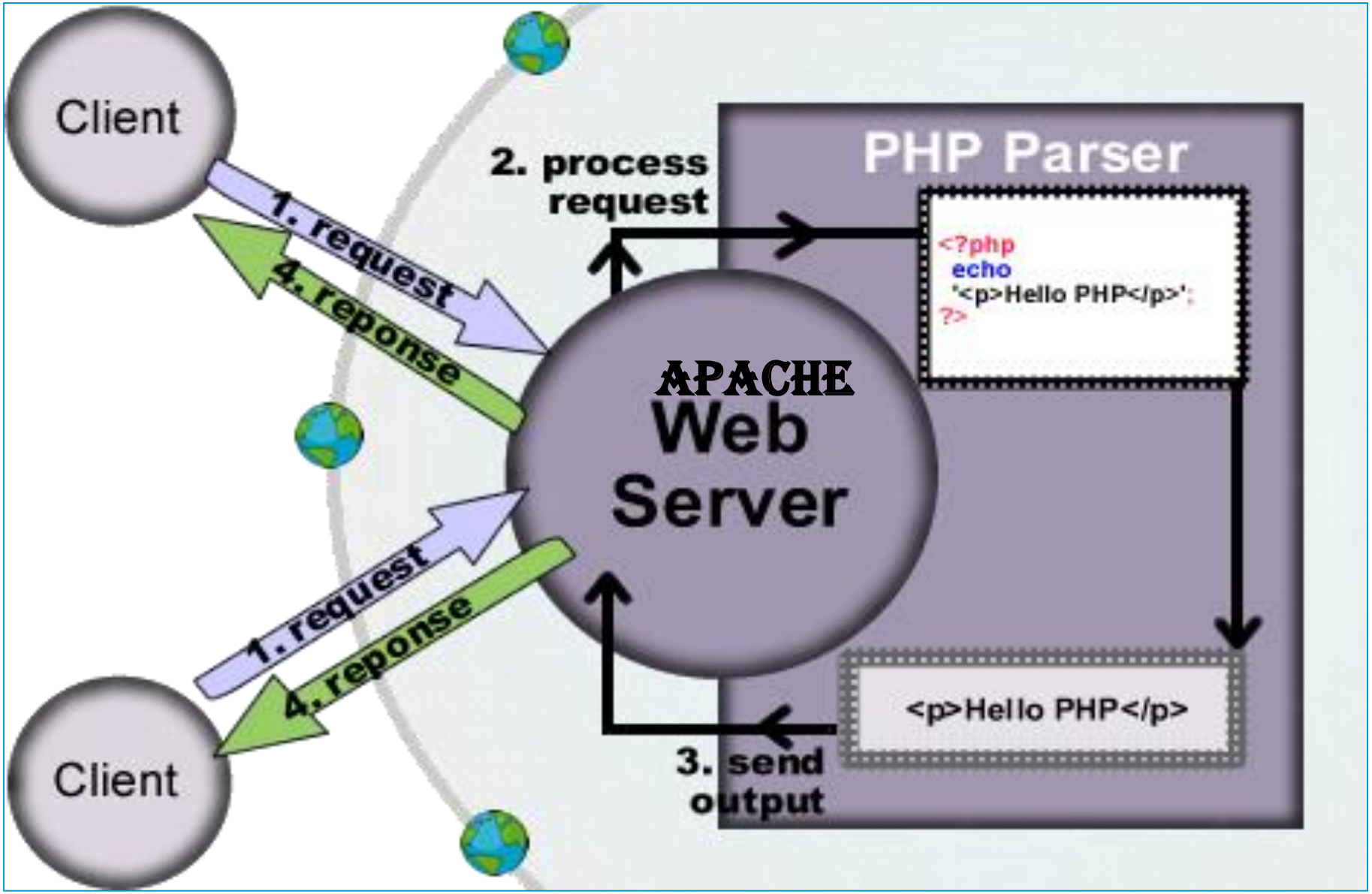
rewrote the parser that formed PHP 3

Zeev Suraski

Release History

Major version	Release date	Notes
1	1995-06-08	Officially called "Personal Home Page Tools (PHP Tools)". This is the first use of the name "PHP".
2	1997-11-01	Considered by its creator as the "fastest and simplest tool" for creating dynamic web pages.
3	1998-06-06	Development moves from one person to multiple developers. Zeev Suraski and Andi Gutmans rewrite the base for this version.
4	2002-12-27	More security enhancements and bug fixes. The last release of the PHP 4.4 series.
5	2004-07-13	Zend Engine II with a new object model
6	Not released	
7	2015-12-03	Zend Engine 3 (performance improvements and 64-bit integer support on Windows), uniform variable syntax etc
8	2020-11-26	Just-In-Time (JIT) compilation,arrays starting with a negative index etc.

PHP Page Life Cycle

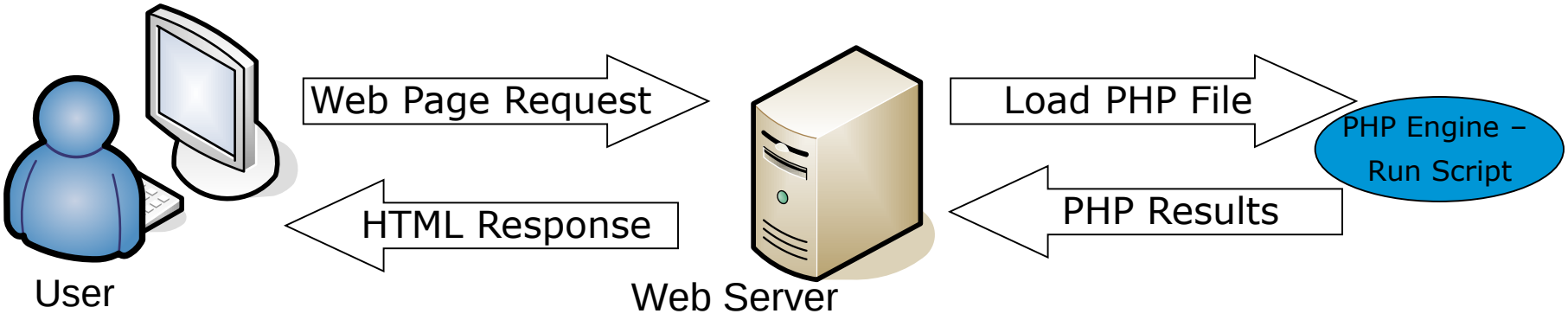


What is PHP ?

- PHP: Hypertext Pre-processor
- PHP scripts are interpreted and executed on the server
- A PHP file can contain plain text, HTML tags and scripts.
- Execution is done before delivering content to the client .
- Contains a vast library of functionality that programmers can harness .
- Executes entirely on the server, requiring no specific features from the client .
- PHP runs on different platforms (Unix, Linux, Windows.)

What is PHP ?

- Static resources such as regular HTML are simply output to the client from the server
- Dynamic resources such as PHP scripts are processed on the server prior to being output to the client
- PHP has the capability of connecting to many database systems making the entire process transparent to the client



Why is PHP used?

- **Cross Platform**
  Runs on almost any Web server on several operating systems.One of the strongest features is the wide range of supported databases

- **Web Servers:**
  Apache, Microsoft IIS, Caudium , Netscape Enterprise Serve

- **Operating Systems:**
  UNIX (HP-UX, OpenBSD , Solaris,Linux), Mac OSX, Windows NT/98/2000/XP/2003.

- **Supported Databases:**
  PHP supports many databases (MySQL ,MS SQL ,SQLLite Sybase, Oracle and many others.)

Why is PHP used?

Easy to Use

Code is embedded into HTML. The PHP code is enclosed in special start and end tags that allow you to jump into and out of "PHP mode".

```
<html>
   <head>
      <title>Example</title>
   </head>
   <body>

      <?php
      echo "Hi, I'm a PHP script!";
      ?>

   </body>
</html>
```

PHP-Syntax and Structure

- All scripts start with <?php and with with ?>
- Line separator: ; (semi-colon)
- Code block: { //code here } (brace brackets)
- White space is generally ignored (not in strings)
- Comments are created using:
  - // single line quote
  - /* Multiple line block quote */
- Precedence
  - Enforced using parentheses
  - E.g. $sum = 5 + 3 * 6; // would equal 23
  - $sum = (5 + 3) * 6; // would equal 48

Displaying Data

- There are two language constructs available to display data: print() and echo().
- They can be used with or without brackets.
- Note that the data ‘displayed’ by PHP is actually parsed by your browser as HTML. View source to see actual output.
- Difference Between Echo And Print()
- Unlike echo, print can accept only one argument.
- Unlike echo , print return a value, which represents whether the print statement succeeded.

Sample Program

```
<?php
    echo "Welcome in PHP Script !!!";
    print "Welcome in PHP Script !!!";
 ?>
```

Displaying Data

- **printf :Output a formatted string**
- **int printf (string $format , mixed $args , mixed $...)**

Type Format	Description
%b	Format the argument as a binary integer (e.g.10010110)
%c	character with the argument's ASCII value
%d	Format the argument as a signed decimal integer
%f	Format the argument as a floating-point number
%o	Format the argument as an octal integer
%s	Format the argument as a string
%u	Format the argument as an unsigned decimal integer
%x	Format the argument as a lowercase hexadecimal integer (e.g. 4fdf87)
%X	Format the argument as an uppercase hexadecimal integer (e.g. 4FDF87)

Escaping Characters

- Some characters are considered ‘special’ Escape these with a backslash \ .
- Special characters will be flagged when they arise, for example a double or single quote belong in this group...

PHP - Data Types

- **PHP is not strictly typed**
  - ❖ Different to JAVA where all variables are declared
- **A data type refers to the type of data a variable can store. PHP has eight (8) different data types you can work with.**

- **Integers:** are whole numbers, without a decimal point, like 4195.
- **Doubles:** are floating-point numbers, like 3.14159 or 49.1.
- **Booleans:** have only two possible values either true or false.
- **NULL:** is a special type that only has one value: NULL.
- **Strings:** are sequences of characters, like 'PHP supports string operations.'
- **Arrays:** are named and indexed collections of other values.
- **Objects:** are instances of programmer-defined classes, which can package up both other kinds of values and functions that are specific to the class.
- **Resources:** are special variables that hold references to resources external to PHP (such as database connections).

Variables in PHP

- “ Variables are used for storing values, like text strings, numbers or arrays ”
- The names of PHP variables must follow a certain number of rules:
- PHP variables must start with the symbol $, followed by a letter or by an underscore.
- PHP variables can contain alphanumeric symbols (A-Z, a-z, 0-9) and underscores (_) only (they cannot contain spaces).
- Assign values with = operator
- Example:
- $author = “Rahul Kumar”;
- No need to define type .
- Variable names are case sensitive.
  - ❖ $author and $Author are different

Managing Variables

- Three language constructs are used to manage variables. They enable you to check if certain variables exist, remove variables, and check variables' truth values.
- **isset() :** isset() determines whether a certain variable has already been declared by PHP. It returns a boolean value true if the variable has already been set, and false otherwise, or if the variable is set to the value NULL .
- Checking an array element:
- if (isset($arr["offset"])) { …… }
- Checking an object property:
- if (isset($obj->property)) { ..... }
- Note:The isset() construct returns false automatically if they are not set.
- isset() is the only one of the three language constructs that accepts an arbitrary amount of parameters.

```
isset($var1, $var2, $var3, ...);
```

Managing Variables

- **unset()**: unset() “undeclares” a previously set variable, and frees any memory that was used by it if no other variable references its value.
- A call to **isset()** on a variable that has been unset() returns false.
- unset() can also be used on array elements and object properties similar to isset().
- **empty()**:empty() may be used to check if a variable has not been declared or its value is false. This language construct is usually used to check if a form variable has not been sent or does not contain data. When checking a variable’s truth value,its value is first converted to a Boolean according to the rules. then it is checked for true/false.

Superglobals

PHP does not support global variables (variables that can automatically be accessed from any scope).

PHP has special internal variables behave like global variables similar to other languages.

These variables are called superglobals and are predefined by PHP .

Some superglobals are:-

- **$_GET[]** -An array that includes all the GET variables that PHP received
- from the client browser.
- **$_POST[]** - An array that includes all the POST variables that PHP received
- **From the client browser.**
- **$_COOKIE[]**- An array that includes all the cookies that PHP received from the client browser.
- **$_ENV[]**- An array with the environment variables.
- **$_SERVER[]**- An array with the values of the web-server variables**.**

PHP - Constants

- ✓ Constants are special variables that cannot be changed .
- ✓ Use them for named items that will not change .
- ✓ In php we can define constant variable in two ways :
  1. Using define() function
  2. Using const keyword

- ✓ To define a constant, use the define function:
  define("CONSTANT_NAME", value)
  - ❖ "CONSTANT_NAME" is a string.
  - ❖ value is any valid PHP expression excluding arrays and objects.

  - ✓ To define a constant, use the const keyword:
    const constantName = Value

  E.G :

  const x = 10;

Differences between constants and variables

- Constants do not have a dollar sign ($) before them;
- Constants may be defined and accessed anywhere without regard to variable scoping rules;
- Constants may not be redefined or undefined once they have been set .
- Constant can’t hold arrays or objects ,like variables can.
- There are some built in constants like- **M_PI , PHP_VERSION**

PHP Operators

Operators are used to operate on values

Type of the Operators:

1- Unary Operator

2- Binary Operator

3- Ternary Operator

Unary Operator

- ❖ Incrementing/Decrementing Operators
- Unary operators act on one operand.

- PHP supports C-style pre- and post-increment and decrement operators.

Increment/decrement Operators

Example	Name	Effect
++$a	Pre-increment	Increments $a by one, then returns $a.
$a++	Post-increment	Returns $a, then increments $a by one.
--$a	Pre-decrement	Decrements $a by one, then returns $a.
$a--	Post-decrement	Returns $a, then decrements $a by one.

Binary Operator

“Binary operators are used on two operands”

PHP can only perform binary operations on two operands that have the same type .

Binary Operators

Operator	Name	Value
+	Addition	The sum of the two operands.
-	Subtraction	The difference between the two operands.
*	Multiplication	The product of the two operands.
/	Division	The quotient of the two operands.
%	Modulus	Both operands are converted to integers. The result is the remainder of the division of the first operand by the second operand.

Ternary Operator

- The ternary operator is a shortcut comparison operator that replaces an if-else statement in a PHP script.
- Ternary operators are used on three operands

  $variable = condition ? if true : if false

  ```
  <?php
  $agestr = ($age < 16) ? 'child' : 'adult';
  ?>
  ```

Concatenation Operator (.)

The concatenation operator concatenate two strings. This operator works only on strings; thus, any non-string operand is first converted to one.

```
<?php
$year = 2000
print "The year is " . $year;
?>
```

Assignment Operators

Assignment operators enable you to write a value to a variable.

Example:

$var = 5 has the value 5 (and assigns 5 to $var).

The following list show the valid composite assignment operators:

+=, -=, *=, /=, %=,

Example:

```
$counter += 2;
// This is identical to $counter = $counter + 2;
```

By-Reference Assignment Operator

```
<?php
$name = “XYZ";
$name_alias =&$name;
print $name_alias;
?>
```

=>The result of this example is

XYZ

Comparison operators

Comparison operators, as their name implies, allow you to compare two values.

Example	Name	Result
$a == $b	Equal	TRUE if $a is equal to $b.
$a === $b	Identical	TRUE if $a is equal to $b, and they are of the same type. (introduced in PHP 4)
$a != $b	Not equal	TRUE if $a is not equal to $b.
$a <> $b	Not equal	TRUE if $a is not equal to $b.
$a !== $b	Not identical	TRUE if $a is not equal to $b, or they are not of the same type. (introduced in PHP 4)
$a < $b	Less than	TRUE if $a is strictly less than $b.
$a > $b	Greater than	TRUE if $a is strictly greater than $b.

Logical Operators

Example	Name	Result
$a and $b	And	TRUE if both $a and $b are TRUE.
$a or $b	Or	TRUE if either $a or $b is TRUE.
$a xor $b	Xor	TRUE if either $a or $b is TRUE, but not both.
! $a	Not	TRUE if $a is not TRUE.
$a && $b	And	TRUE if both $a and $b are TRUE.
$a || $b	Or	TRUE if either $a or $b is TRUE.

Error Control Operators

- PHP supports one error control operator: the at sign (@). When prepended to an expression in PHP, any error messages that might be generated by that expression will be ignored.

  Example:

- $myfile = @file ('test1.ph') or
- die ("failed opening file:error was '$php_errorms'");

String Operators

There are two string operators.

The first is the concatenation operator ('.'), which returns the concatenation of its right and left arguments.

```
<?php
  $a = "Hello ";
  $b = $a . "World!";
?>
```

String Operators

The second is the concatenating assignment operator ('.='), which appends the argument on the right side to the argument on the left side.

```
<?php
   $a = "Hello ";
    $a .= "World!";
   echo $a;
?>
```

Conversion

Type of One of The Operand	Type of the Other Operand	Conversion Performed
Integer	Floating point	The integer operand is converted to a floating point number.
Integer	String	The string is converted to a number. If the converted string's type is real, the integer operand is converted to a real as well.
Real	String	The string is converted to a real.

Type Casting

There are two ways to cast a variable in PHP as a specific type: using the settype() function , or using (int) (bool) (float) etc.

The casts allowed are:

(int), (integer)	cast to integer
(bool), (boolean)	cast to boolean
(float), (double), (real)	cast to float
(string)	cast to string
(array)	cast to array
(object)	cast to object

Type Casting

settype () — Using the settype function you can do this:

```
$v =‘5’
settype($v,”int”)
settype($v,”boolen”)
```

gettype() — Get the type of a variable

```
gettype($v);
```

Type Testing

When converting to boolean , the following values are considered FALSE:

- the boolean FALSE itself
- the integer 0 (zero)
- the float 0.0 (zero)
- the empty string, and the string "0"
- an array with zero elements
- an object with zero member variables (PHP 4 only)

Example :-

```
<?php
    var_dump((bool) "");        // bool(false)
    var_dump((bool) 1);         // bool(true)
    var_dump((bool) -2);        // bool(true)
    var_dump((bool) "foo");     // bool(true)
?>
```

Integer overflow

If you specify a number beyond the bounds of the integer type, it will b interpreted as a float instead.

Example :-

```
<?php
$large_number =  9223372036854775807;
var_dump($large_number);
// output: int(9223372036854775807)
$large_number+=1;
var_dump($large_number);
// output: float(9223372036854775808)
?>
<?php
var_dump(25/7);         // float(3.5714285714286)
var_dump((int) (25/7)); // int(3)
var_dump(round(25/7));  // float(4)
?>
```

Variable variables

Variable variables allow you to access the contents of a variable without knowing its name directly - it is like indirectly referring to a variable.

Example :-

```
<?php
    $a = 'hello';
    $$a = 'world';
    echo "$a ${$a}";
    echo "$a $hello";
?>
```

Please note that variable variables cannot be used with PHP's Super global arrays within functions or class methods

Variable scope

The scope of a variable is the context within which it is defined. For the most part all PHP variables only have a single scope.

Example :-

```
<?php
    $a = 1;
     function Test()
     {
      $a=20;
     echo $a;
}
Test();
echo $a;
?>
```

Global keyword

A global variable can be accessed in any part of the program.
A global variable must be explicitly declared to be global in the function in which it is to be modified.
This is accomplished, conveniently enough, by placing the keyword global in front of the variable that should be recognized as global. Example :-

```
<?php
        $a = 1;
       $b = 2;
       function Sum()
      {
        global $a, $b;
        $b = $a + $b;
       }
    Sum();
    echo $b;
?>
```

$GLOBALS instead of global

- The $GLOBALS array is an associative array with the name of the global variable being the key and the contents of that variable being the value of the array element.
- Example :-

```
<?php
$a = 1;
$b = 2;
function Sum()
{
    $GLOBALS['b'] = $GLOBALS['a'] + $GLOBALS['b'];
}
Sum();
echo $b;
?>
```

- Notice how $GLOBALS exists in any scope, this is because $GLOBALS is a superglobal.

Static variables

A static variable exists only in a local function scope, but it does not lose its value when program execution leaves this scope.
Example :-

```
<?php
function Test()
{
    static $a = 0;
    echo $a;
    $a++;
}
for($i=0;$i<5;$i++)
{
Test();
}
?>
```

Control Structures

PHP supports a variety of the most common control structures available in other programming languages

They can be basically divided into two groups

1. Conditional control structures (or) Selection constructs
2. Loop control structures (or) Iteration constructs

Chapter Summary

In this chapter, you have learned about:

- PHP is the short form of "**PHP: Hypertext Preprocessor**", PHP/FI, was developed by RasmusLerdorf in 1994.
- The PHP scripts must starts with <?php and ends with ?>.
- Data types are types of data and the operations that can perform on those data. PHP supports eight different data-types.
- PHP has different Types of Variables like Local, Global, etc.
- PHP has the different types of operators.
- PHP has the decision making statements.

Thank you