UNIT 2 PHP

- Introduction to Server side Programming
- Introduction to PHP
- PHP and HTML
- essentials of PHP
- Why Use PHP
- Installation of Web Server
- WAMP Configurations
- Writing simple PHP program
- embedding with HTML
- comments in PHP ,Variables, Naming Conventions, Strings, String Concatenation, String functions, float functions.

Server-side Programming :

It is the program that runs on server dealing with the generation of content of web page.
1) Querying the database
2) Operations over databases
3) Access/Write a file on server.
4) Interact with other servers.
5) Structure web applications.
6) Process user input. For example if user input is a text in search box, run a search algorithm on data stored on server and send the results.

Examples :
The Programming languages for server-side programming are :
1) PHP
2) C++
3) Java and JSP
4) Python
5) Ruby on Rails

Introduction of PHP

The **PHP Hypertext Preprocessor (PHP)** is a programming language that allows web developers to create dynamic content that interacts with databases. PHP is basically used for developing web based software applications. This tutorial helps you to build your base with PHP.

Why to Learn PHP?

PHP started out as a small open source project that evolved as more and more people found out how useful it was. Rasmus Lerdorf unleashed the first version of PHP way back in 1994.

PHP is a MUST for students and working professionals to become a great Software Engineer specially when they are working in Web Development Domain. I will list down some of the key advantages of learning PHP:

- PHP is a recursive acronym for "PHP: Hypertext Preprocessor".
- PHP is a server side scripting language that is embedded in HTML. It is used to manage dynamic content, databases, session tracking, even build entire e-commerce sites.
- It is integrated with a number of popular databases, including MySQL, PostgreSQL, Oracle, Sybase, Informix, and Microsoft SQL Server.
- PHP is pleasingly zippy in its execution, especially when compiled as an Apache module on the Unix side. The MySQL server, once started, executes even very complex queries with huge result sets in record-setting time.
- PHP supports a large number of major protocols such as POP3, IMAP, and LDAP. PHP4 added support for Java and distributed object architectures (COM and CORBA), making n-tier development a possibility for the first time.
- PHP is forgiving: PHP language tries to be as forgiving as possible.
- PHP Syntax is C-Like.

Characteristics of PHP

Five important characteristics make PHP's practical nature possible −

- Simplicity
- Efficiency
- Security
- Flexibility
- Familiarity

Hello World using PHP.

```
<html>

  <head>
    <title>Hello World</title>
  </head>

  <body>
    <?php echo "Hello, World!";?>
  </body>

</html>
```

Applications of PHP

As mentioned before, PHP is one of the most widely used language over the web. I'm going to list few of them here:

- PHP performs system functions, i.e. from files on a system it can create, open, read, write, and close them.
- PHP can handle forms, i.e. gather data from files, save data to a file, through email you can send data, return data to the user.
- You add, delete, modify elements within your database through PHP.
- Access cookies variables and set cookies.
- Using PHP, you can restrict users to access some pages of your website.
- It can encrypt data.

What is a Web Server?

A web server is software that listens for requests and returns data (usually a file). When you type "www.mysite.com", the request is forwarded to a machine running web server software which returns a file back to your browser — such as the contents of index.html. The browser might then make further requests based on the HTML content — like CSS, JavaScript, and graphic files.

Since the web server sits between your browser and the requested file, it can perform processing that's not possible by opening an HTML file directly. For example, it can parse PHP code which connects to a database and returns data.

You can use your host's web server for testing, but uploading will become tiresome and changes could go live before they've been fully tested. What you need is a local web server installation.

What is WAMP

- WAMP is an acronym that stands for Windows, Apache, MySQL, and PHP. It's a software stack which means installing WAMP installs Apache, MySQL, and PHP on your operating system (Windows in the case of WAMP). Even though you can install them separately, they are usually bundled up, and for a good reason too.
- What's good to know is that WAMP derives from LAMP (the L stands for Linux). The only difference between these two is that WAMP is used for Windows, while LAMP – for Linux based operating systems.

- "W" stands for Windows, there's also LAMP (for Linux) and MAMP (for Mac).
- "A" stands for Apache. Apache is the server software that is responsible for serving web pages. When you request a page to be seen by you, Apache grants your request over HTTP and shows you the site.
- "M" stands for MySQL. MySQL's job is to be the database management system for your server. It stores all of the relevant information like your site's content, user profiles, etc.

- “P” stands for PHP. It’s the programming language that was used to write WordPress. It acts like glue for this whole software stack. PHP is running in conjunction with Apache and communicating with MySQL.

- Instead of installing and testing WordPress on your hosting account, you can do it on your personal computer (localhost).
- WAMP acts like a virtual server on your computer. It allows you to test all WordPress features without any consequences since it’s localized on your machine and is not connected to the web.
- First of all, this means that you don’t need to wait until files are uploaded to your site, and secondly – this makes creating backups much easier.
- WAMP speeds up the work process for both developers and theme designers alike. What is more, you also get the benefit of playing around with your site to your heart’s content.
- However, to actually make the website go live, you need to get some form of hosting service and a Domain. See our beginner-friendly article about web hosting for more information.
- Or if you’re already prepared to go online, take a look at what Hostinger has to offer. We can guarantee you the best web hosting prices in the market for high-quality hosting solutions.
- In essence, WAMP is used as a safe space to work on your website, without needing to actually host it online.
- WAMP also has a control panel. Once you install the software package, all of the services mentioned above (excluding the operating system that is) will be installed on your local machine.
- Whether you use WAMP or software packages for the other operating systems, it’s a great way to save time. You won’t have to upload files to a site and will be able to learn how to develop in a safe and care-free environment.

<u>Simple PHP Program :-</u>

Sum Of Digit

```
<?php
$num = 14597;
$sum=0; $rem=0;
  for ($i =0; $i<=strlen($num);$i++)  {
  $rem=$num%10;
   $sum = $sum + $rem;
   $num=$num/10;
  }
 echo "Sum of digits 14597 is $sum";
```

```
?>
```

Prime number in PHP

```php
<?php
$count = 0;
$num = 2;
while ($count < 15 ) {
$div_count=0;
for ( $i=1; $i<=$num; $i++) {
if (($num%$i)==0) {
$div_count++;
}
}
if ($div_count<3) {
echo $num." , ";
$count=$count+1;
}
$num=$num+1;
}
?>
```

How to use PHP in Html

If we want to use PHP in the Html document, then we have to follow the steps which are given below. Using these simple steps, we can easily add the PHP code.

Step 1: Firstly, we have to type the Html code in any text editor or open the existing Html file in the text editor in which we want to use the PHP.

1. <!Doctype Html>
2. <Html>
3. <Head>
4. <Title>
5. Use a Php in Html
6. </Title>
7. </Head>
8. <Body>

```
9.  </Body>
10. </Html>
```

Step 2: Now, we have to place the cursor in any tag of the <body> tag where we want to add the code of PHP. And, then we have to type the start and end tag of PHP.

```
1.   <h1>
2.  <?php   ?>
3.  </h1>
```

Step 3: After then, we have to type the code of PHP between the tags of PHP.

```
1.  <h1>
2.  <?php
3.   echo "Hii User!! You are at JavaTpoint Site"
4.  ?>
5.  </h1>
```

Step 4: When we successfully add the PHP code, then we have to save the Html file and then run the file in the browser.

```
1.  <!Doctype Html>
2.  <Html>
3.  <Head>
4.  <Title>
5.  Use a Php in Html
6.  </Title>
7.  </Head>
8.  <Body>
9.   <h1><?php echo "Hii User!! You are at JavaTpoint Site" ?></h1>
10. </Body>
11. </Html>
```

Name Convertion:

In Web Development world, keeping a definite naming convention is a tough job. Every developer follows the practices he feels convenient and you may even identify the code written by a particular developer. In the field of web development we call it as the DNA of the code. But as you know, web development is not meant for a single person’s job, there is always a Team effort somewhere down the line. So it becomes necessary that you follow a specific set of naming conventions in your project or even on the whole Agency level, so that every team member can easily grasp the inners of code by simply looking at a small chunk of code and takeover the further development tasks if in any case you need to switch developers. I have seen developers feeling hesitant to touch the code of peer members just because of the differences in naming conventions.

Variables:

First of all, what is a variable? A variable is what stores data. A data can be of any type, be it String, Integer, Floating point values or Boolean, Arrays. I'd even go further to say that we should follow same conventions when defining Array Keys and Database Field Names as well. It improves consistency across our whole development stack.

```
$first_name = "John"; // Right
$last_name = "Doe"; // Right

$n1 = "John"; // Wrong
$n2 = "Doe"; // Wrong

$my_name = "John Doe"; // Wrong
$my_var = "John Doe"; // Wrong
```

So can you guess what's wrong with $n1, $n2, $my_name and $my_var ? One word – "Ambiguity". Ambiguous names are the biggest reason for unreadable and ugly code. So what is "Ambiguity"? It's simply naming a variable in a way so that you can't guess what exactly it stores. Consider for example following snippet:

```
$i = "Apple iPhone 6S (32GB, Rose Gold)";
$p = "749.00";
$t = "749.00";
```

Can you guess in the above code, what is $t? No? Okay, let's try following code:

```
$item = "Apple iPhone 6S (32GB, Rose Gold)";
$price = "749.00";
$total = "749.00";
```

Can you guess now? Exactly! This is how you differentiate ambiguous variables from absolute variables. Obviously the latter approach is better because it is definitive.

Constants:

Constants as the name intends are meant to store the values which don't change throughout the program. So what kind of values are supposed to be constants? They are Database Credentials, API Keys and Mathematical constants as well. There are two types of constants you can find in PHP language – Global and Class-based.

Global constants can be defined in a common file which is included in each request of our program.

```
define(‘DB_HOST’, ‘localhost’);
define(‘DB_USER’, ‘db_user’);
define(‘DB_PASS’, ‘password$%@#’);
define(‘DB_NAME’, ‘db_app’);
```

See, how we’ve kept the name of constants in ALL CAPS. This is to easily differentiate between constants and variables in a program.

Note : PHP Constants don’t use the $ prefix as the variables do.

Now an example of class constants:

```
class Circle{
  const PI = 3.14;
          public function circumference($radius){
          return 2 * self::PI * $radius;
  }
  public function area($radius){
          return self::PI * $radius * $radius;
  }
}
```

When you need to use a class constant inside class itself, you use the self::CONSTANT_NAME but to use constant outside class you can simply call it by replacing self with the name of the class, e.g. Circle::PI

Class constants are targeted for contained libraries such as when integrating multiple 3rd-Party APIs since a library can have all its constants inside its classes so that they don’t get conflicted with the global constants. So suppose a class decides to use PI upto 6 precisions but you yourself are using PI upto 2 precisions, you can achieve this simply by making a global PI constant and one class-specific constant. Easy, huh! (I know this is a stupid example, in real world you’d never need to use 2 versions of PI)

Functions:

A function is a piece of code which does a specific task. Each programming language consists of a lot of functions which do different tasks. So what conventions are mostly used in functions?

There are two of them majorly used: lowercase with underscore separators and camelcase.

Consider following two examples:

```
function get_name(){
  // Do something
}
```

```
function getName(){
  // Do something
}
```

So which one is best – I'd say none, both are equally good but we personally prefer to use the camelCase convention because for variables I prefer the underscore separations and thus using camelcase for functions give a visual ease of differentiating between a function and a variable.

But (yeah, there is a but), always keep in mind that a function should always start with a "Verb" which defines what that function is trying to do in conjunction with the name of the "Entity" being affected by this function.

```
getName() // Right
namesList() // Wrong
listNames() // Right
sendEmail() // Right
deleteUserById($id) // Right
connectToDatabase() // Right
databaseConnection() // Wrong
prepareOutput() // Right
```

See, how above examples consist of "Verb" and "Entity" in the functions and which ways are correct.

This covers the global/procedural functions, what about "Class Methods"? For those, who don't know what is a "Class Method" – it is simply a function but inside a class, we'll cover them in greater details in our upcoming articles. Here is how you handle the naming convention for Class Methods:

```
class User{
  public function delete($id){
    // Do something
  }
}

$user = new User();
$user->delete(2); // Delete the user with ID = 2
```

So I'm contradicting myself for "Class Methods" – right? Yes, there is a logical explanation for this. When you use a class, a class itself represents an "Entity" of our program so suffixing the "Entity" in a function name doesn't make sense because it becomes self-explanatory in case of class methods.

Note : An " Entity " is what a program is composed of. It is a universal term and is applicable across all kind of programs. A Blog website has following entities: Articles, Authors, Topics, Tags whereas an E-commerce platform consists of following entities: Products, Categories, Orders, Manufacturers, Customers. Got the idea!

Classes:

Each class should be declared in its own file. And class names must be in “StudlyCaps”

```
class ModelUser{

}
```

Private and protected properties in a class MUST BE prefixed with a single underscore, for example:

```
class Circle{
  public PI = 3.14;
  private $_radius;

  private function _calculateArea(){
    return self::PI * $_radius * $_radius;
  }
}
```

Parentheses:

Parenthesis are important to follow correctly. There are only two conventions for parenthesis which are widely used now-a-days. You can read more about all the styles in existence here:

K&R Style Parentheses and Allman Parentheses.

K&R Style Parentheses:

```
if($a == 1){
  // do something
}
```

Allman Parentheses:

```
if($a == 1){
  // do something
}
```

We prefer to use K&R Style parentheses because they save one line and seem natural. But a lot of people prefer Allman parentheses. It doesn’t matter which one you prefer, they both are equally good, what matters is ensuring that you stick to the one you prefer.

Epilogue:

The key to following conventions is consistency. You MUST ensure that you stick to a particular set of conventions. A lot of agencies have developed their own coding guidelines and you can follow the ones we follow here at Hashbyte. If you wish, you can mix and mingle to create yourself a new set of conventions.

PHP | floatval() Function

The floatval() function is an inbuilt function in PHP which returns the float value of a variable.

Syntax:

float floatval ($var)

Parameters: This function accepts one parameter which is mandatory and described below:

$var: The variable whose corresponding float value will be returned. This variable should not be an object.

Return Value: It returns float value of given variable. Empty array returns 0 and non-empty array returns 1.

Note: If an object is passed to the function, it produce an E_NOTICE level error and return 1.

Examples:

```
Input : $var = '41.68127E3'
Output : 41681.27

Input : $var = '47.129mln'
Output : 47.129
```

Program 1:

```
<?php
$var = '41.68127E3';
$float_value = floatval($var);
echo $float_value;
?>
```

Output:

```
41681.27
```