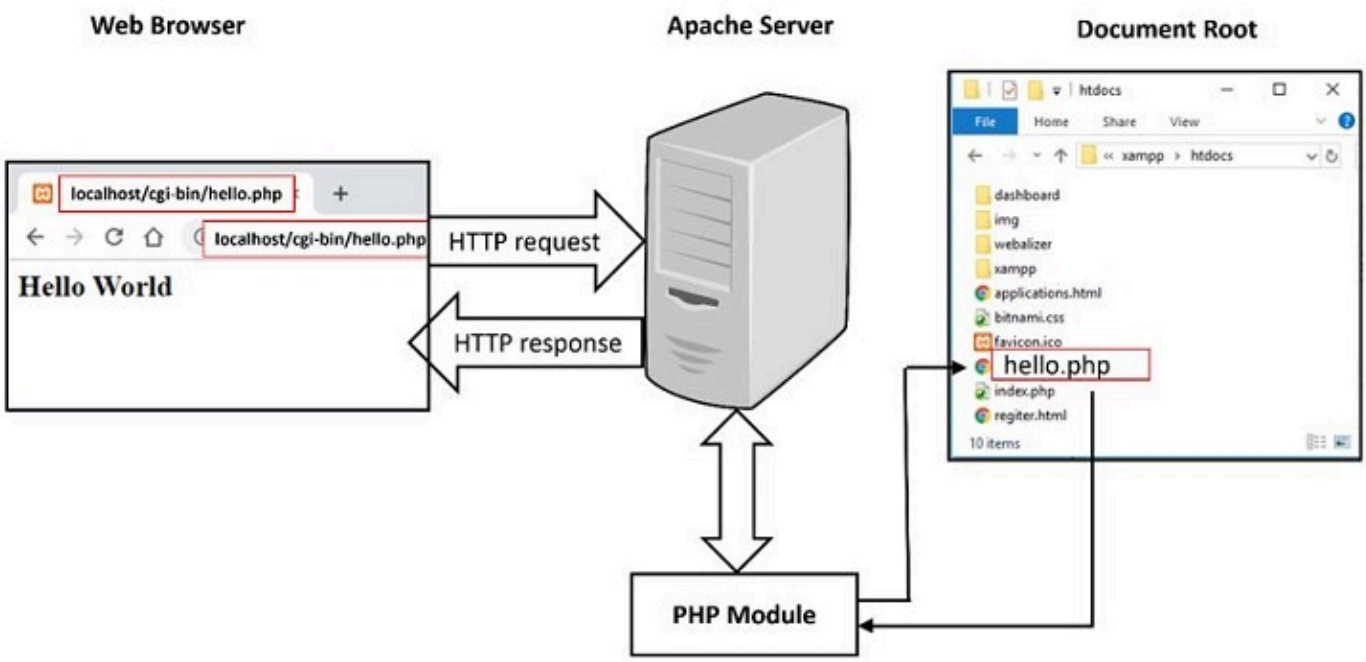
PHP - Web Concepts

PHP is a server-side scripting language that is used to create dynamic webpages. It is one of the most popular programming languages for web development. This chapter aims to let you get familiarized with certain important concepts of web application development using PHP.

A web-based application is a collection of webpages. A webpage is mainly created with HTML tags. HTML consists of different HTML tags which are required to define the appearance of page elements like text, image, table, etc. Hence, HTML essentially creates a static webpage.

A Web application is hosted on a HTTP server with PHP module installed. The browser acts as a http client, to establish communication with the server, following HTTP protocol.



How to Add Dynamic Content on a Webpage?

To add dynamic content io a webpage, there are two possibilities.

JavaScript is a client-side scripting language, that can access the HTML document object model and render dynamic content on the client browser. JavaScript code can be embedded in HTML page.

The browser may collect data from the user in the form of HTML form elements and send it to a HTTP server for processing. PHP is a widely used Server-side processing language. PHP script can also be embedded inside HTML page.

Example

In the following script, JavaScript code embedded in HTML renders the current date as per the client browser, and the PHP code displays the current date as per the server, where this script is hosted.

</> Open Compiler

```
<!DOCTYPE html>
<html>
<body>
   <script type="text/JavaScript">
      document.write("Client's date :"+Date()+"\n");
   </script>
   <?php
      date_default_timezone_set("Asia/Calcutta");
      echo "server's date is " . date("Y-m-d") . "\n";
      echo "The time is " . date("h:i:sa");
   ?>
</body>
</html>
```

PHP can intercept and process the data from HTML forms. This allows you to collect information from your users. The next chapter discusses PHP's form handling.

PHP can be used to interact with databases such as MySQL and PostgreSQL. This allows you to store and retrieve data from your database, and dynamically populate the web pages or to power the web applications. PHP includes mysql, mysqli and PDO extensions for database handling.

PHP can handle the data received from the client with HTTP GET as well as POST methods. We shall discuss in detail, how PHP handles GET/POST methods in one of the latter chapters.

HTTP is a stateless protocol. However, it allows Sessions and cookies to be maintained on server and client respectively. PHP can be used to create and manage sessions and cookies. Sessions allow you to track individual users as they navigate your website, while cookies allow you to store information on the user's computer for later use. In of the subsequent chapters, we shall learn how PHP handles sessions and cookies.

PHP can be used to upload files to your web server. This allows you to create web applications that allow users to upload files, such as images, videos, or documents.

You can use PHP to create a login page for your website. When the user enters their username and password, PHP can check the database to see if the user is valid. If the user is valid, PHP can log the user in and redirect them to the main page of your website.

Identifying Browser & Platform

PHP creates some useful **environment variables** that can be seen in the phpinfo.php page that was used to setup the PHP environment.

One of the environment variables set by PHP is **HTTP_USER_AGENT** which identifies the user's browser and operating system.

PHP provides a function getenv() to access the value of all the environment variables. The information contained in the HTTP_USER_AGENT environment variable can be used to create dynamic content appropriate to the browser.

Example

Following example demonstrates how you can identify a client browser and operating system.

NOTE – The function preg_match()is discussed in PHP Regular expression session.

```
<?php
   function getBrowser() {
      $u_agent = $_SERVER['HTTP_USER_AGENT'];
      $bname = 'Unknown';
      $platform = 'Unknown';
      $version = "";

      //First get the platform
      if (preg_match('/linux/i', $u_agent)) {
         $platform = 'linux';
      } elseif (preg_match('/macintosh|mac os x/i', $u_agent)) {
         $platform = 'mac';
      } elseif (preg_match('/windows|win32/i', $u_agent)) {
         $platform = 'windows';
      }

      // Next get the name of the useragent yes seperately and for good reason
      if(preg_match('/MSIE/i',$u_agent) && !preg_match('/Opera/i',$u_agent)) {
         $bname = 'Internet Explorer';
         $ub = "MSIE";
      } elseif(preg_match('/Firefox/i',$u_agent)) {
```

```
      $bname = 'Mozilla Firefox';
      $ub = "Firefox";
   } elseif(preg_match('/Chrome/i',$u_agent)) {
      $bname = 'Google Chrome';
      $ub = "Chrome";
   } elseif(preg_match('/Safari/i',$u_agent)) {
      $bname = 'Apple Safari';
      $ub = "Safari";
   } elseif(preg_match('/Opera/i',$u_agent)) {
      $bname = 'Opera';
      $ub = "Opera";
   } elseif(preg_match('/Netscape/i',$u_agent)) {
      $bname = 'Netscape';
      $ub = "Netscape";
   }

   // finally get the correct version number
   $known = array('Version', $ub, 'other');
   $pattern = '#(?<browser>' . join('|', $known) . ')
      [/ ]+(?<version>[0-9.|a-zA-Z.]*)#';

   if (!preg_match_all($pattern, $u_agent, $matches)) {
      // we have no matching number just continue
   }

   // see how many we have
   $i = count($matches['browser']);

   if ($i != 1) {
      //we will have two since we are not using 'other' argument yet

      //see if version is before or after the name
      if (strripos($u_agent,"Version") < strripos($u_agent,$ub)){
         $version= $matches['version'][0];
      } else {
         $version= $matches['version'][1];
      }
   } else {
      $version= $matches['version'][0];
   }

   // check if we have a number
```

```
      if ($version == null || $version == "") {$version = "?";}
         return array(
            'userAgent' => $u_agent,
            'name'      => $bname,
            'version'   => $version,
            'platform'  => $platform,
            'pattern'   => $pattern
         );
   }

   // now try it
   $ua = getBrowser();
   $yourbrowser = "Your browser: " . $ua['name'] . " " . $ua['version'] .
   " on " .$ua['platform'] . " reports: <br >" . $ua['userAgent'];

   print_r($yourbrowser);
?>
```

This is producing following result on my machine. This result may be different for your computer depending on what you are using.

It will produce the following result –

```
Your browser: Google Chrome 54.0.2840.99 on windows reports:
Mozilla/5.0 (Windows NT 6.3; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
   Chrome/54.0.2840.99 Safari/537.36
```


Display Images Randomly

The PHP **rand()** function is used to generate a random number.i This function can generate numbers with-in a given range. The random number generator should be seeded to prevent a regular pattern of numbers being generated. This is achieved using the **srand()** function that specifies the seed number as its argument.

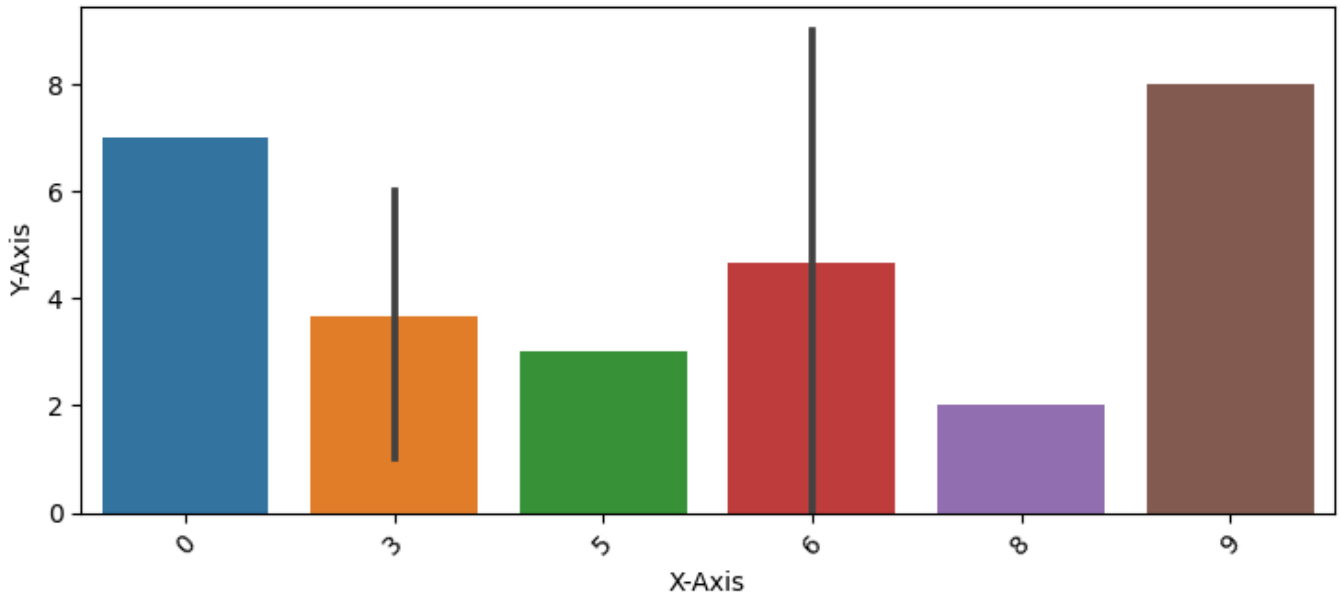
Example

Following example demonstrates how you can display different image each time out of four images –

</> Open Compiler

```php
<?php
   srand( microtime() * 1000000 );
   $num = rand( 1, 4 );

   switch( $num ) {
      case 1: $image_file = "/php/images/php_image_sample_1.jpg";
         break;

      case 2: $image_file = "/php/images/php_image_sample_2.jpg";
         break;

      case 3: $image_file = "/php/images/php_image_sample_3.jpg";
         break;

      case 4: $image_file = "/php/images/php_image_sample_4.jpg";
         break;
   }
   echo "Random Image : <img src=$image_file />";
?>
```

It will produce the following result –



Using HTML Forms

The most important thing to notice when dealing with HTML forms and PHP is that any form element in an HTML page will automatically be available to your PHP scripts.

Example

Try out following example by putting the source code in test.php script.

```
<?php
   if( $_POST["name"] || $_POST["age"] ) {
      if (preg_match("/[^A-Za-z'-]/",$_POST['name'] )) {
         die ("invalid name and name should be alpha");
      }

      echo "Welcome ". $_POST['name']. "<br />";
      echo "You are ". $_POST['age']. " years old.";

      exit();
   }
?>
<form action = "<?php <b>$_PHP_SELF</b> ?>" method = "POST">
   Name: <input type = "text" name = "name" />
   Age: <input type = "text" name = "age" />
   <input type = "submit" />
</form>
```

It will produce the following result –

- The PHP default variable **$_PHP_SELF** is used for the PHP script name and when you click "submit" button then same PHP script will be called and will produce following result –
- The method = "POST" is used to post user data to the server script. There are two methods of posting data to the server script which are discussed in PHP GET & POST chapter.

Browser Redirection

The PHP **header()** function supplies raw HTTP headers to the browser and can be used to redirect it to another location. The redirection script should be at the very top of the page to prevent any other part of the page from loading.

The target is specified by the **Location:** header as the argument to the **header()** function. After calling this function the **exit()** function can be used to halt parsing of rest of the code.

Example

Following example demonstrates how you can redirect a browser request to another web page. Try out this example by putting the source code in test.php script.

</> Open Compiler

```
<?php
   if( $_POST["location"] ) {
      $location = $_POST["location"];
      header( "Location:$location" );

      exit();
   }
?>
<p>Choose a site to visit :</p>
<form action = "<?php <b>$_SERVER['PHP_SELF']</b> ?>" method ="POST">
   <select name = "location">.

      <option value = "http://www.tutorialspoint.com">
         Tutorialspoint.com
      </option>

      <option value = "http://www.google.com">
         Google Search Page
      </option>

   </select>
   <input type = "submit" />
</form>
```

It will produce the following result –