**Task no 1:**

**Provide a descriptions and usage for all the PHP superglobals:**

PHP Superglobals

Superglobals were introduced in PHP 4.1.0, and are built-in variables that are always available in all scopes.

Several predefined variables in PHP are "superglobals", which means that they are always accessible, regardless of scope - and you can access them from any function, class or file without having to do anything special.

The PHP superglobal variables are:

* $GLOBALS
* $\_SERVER
* $\_REQUEST
* $\_POST
* $\_GET
* $\_FILES
* $\_ENV
* $\_COOKIE
* $\_SESSION
  1. PHP $GLOBALS
* $GLOBALS is a PHP super global variable which is used to access global variables from anywhere in the PHP script (also from within functions or methods).
* PHP stores all global variables in an array called $GLOBALS[*index*]. The *index* holds the name of the variable.
* The example below shows how to use the super global variable $GLOBALS:
* Example
* <?php   
  $x = 75;   
  $y = 25;  
     
  function addition() {   
      $GLOBALS['z'] = $GLOBALS['x'] + $GLOBALS['y'];   
  }  
     
  addition();   
  echo $z;   
  ?>

In the example above, since z is a variable present within the $GLOBALS array, it is also accessible from outside the function!

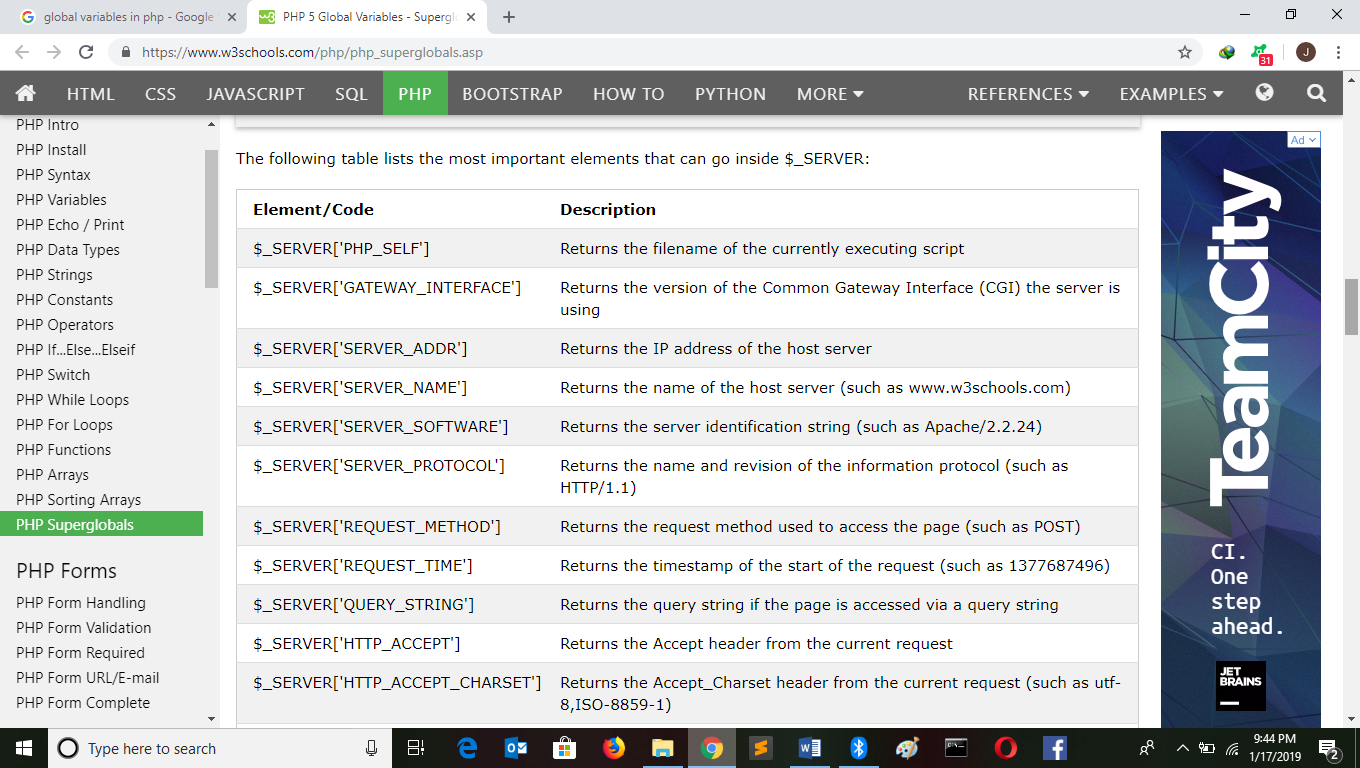
1. PHP $\_SERVER

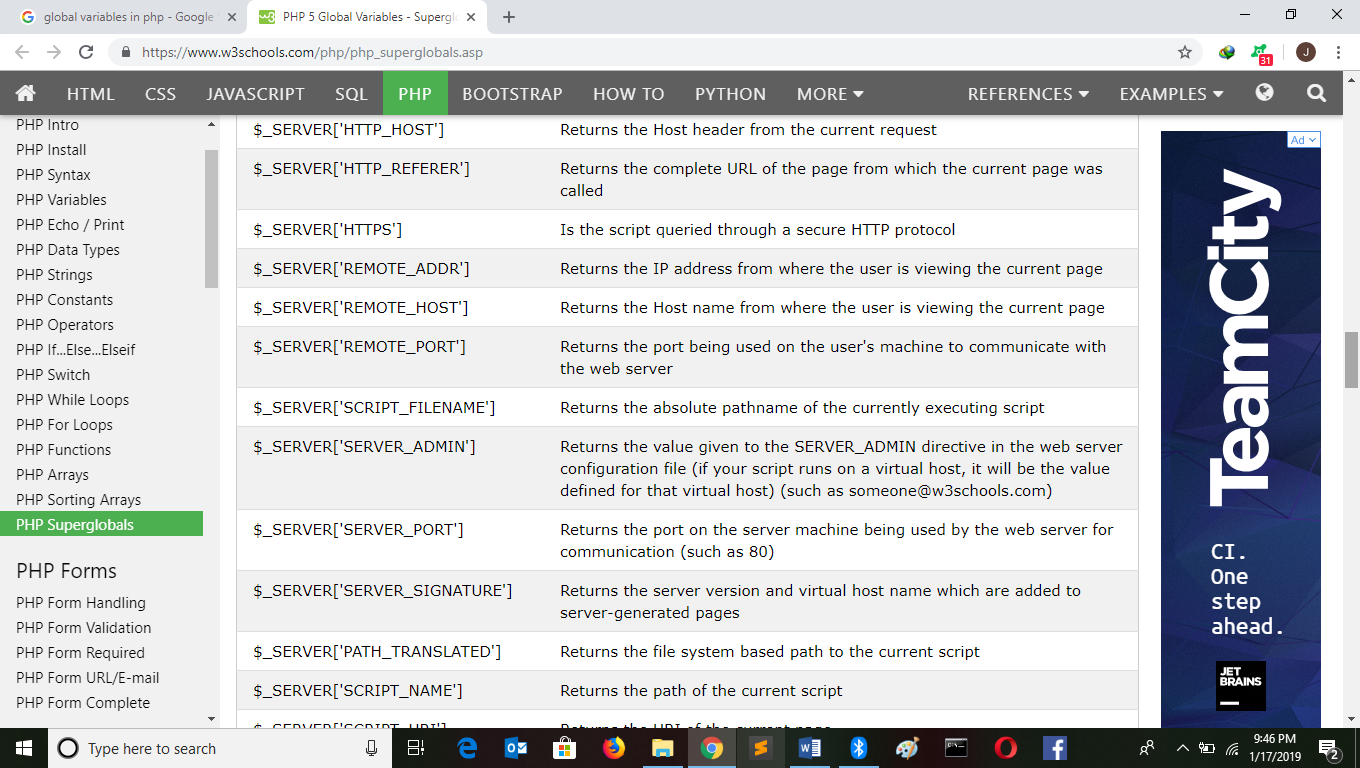
$\_SERVER is a PHP super global variable which holds information about headers, paths, and script locations.

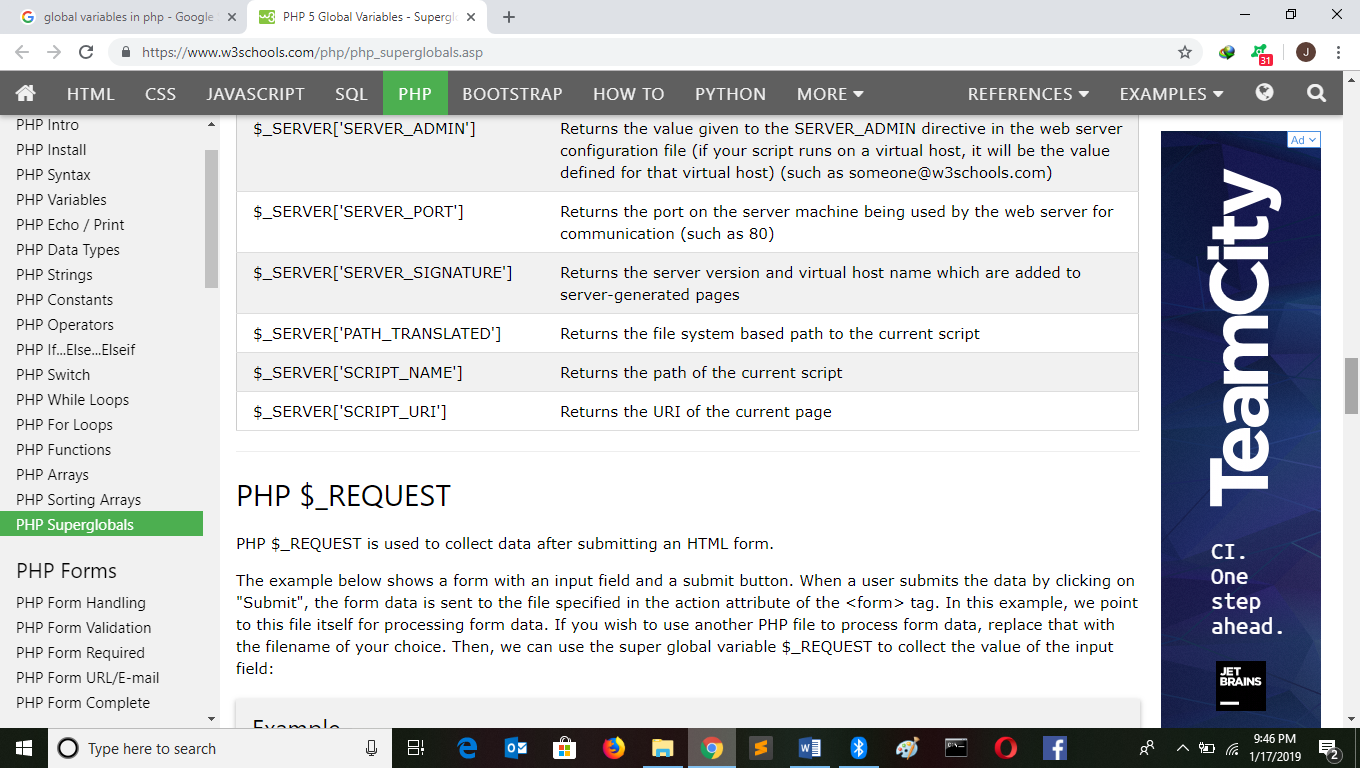
The example below shows how to use some of the elements in $\_SERVER:

Example

<?php   
echo $\_SERVER['PHP\_SELF'];  
echo "<br>";  
echo $\_SERVER['SERVER\_NAME'];  
echo "<br>";  
echo $\_SERVER['HTTP\_HOST'];  
echo "<br>";  
echo $\_SERVER['HTTP\_REFERER'];  
echo "<br>";  
echo $\_SERVER['HTTP\_USER\_AGENT'];  
echo "<br>";  
echo $\_SERVER['SCRIPT\_NAME'];  
?>







1. PHP $\_REQUEST

PHP $\_REQUEST is used to collect data after submitting an HTML form.

The example below shows a form with an input field and a submit button. When a user submits the data by clicking on "Submit", the form data is sent to the file specified in the action attribute of the <form> tag. In this example, we point to this file itself for processing form data. If you wish to use another PHP file to process form data, replace that with the filename of your choice. Then, we can use the super global variable $\_REQUEST to collect the value of the input field:

Example

<html>  
<body>  
  
<form method="post" action="<?php echo $\_SERVER['PHP\_SELF'];?>">  
  Name: <input type="text" name="fname">  
  <input type="submit">  
</form>  
  
<?php  
if ($\_SERVER["REQUEST\_METHOD"] == "POST") {  
    // collect value of input field  
    $name = $\_REQUEST['fname'];  
    if (empty($name)) {  
        echo "Name is empty";  
    } else {  
        echo $name;  
    }  
}  
?>  
  
</body>  
</html>

1. PHP $\_POST

PHP $\_POST is widely used to collect form data after submitting an HTML form with method="post". $\_POST is also widely used to pass variables.

The example below shows a form with an input field and a submit button. When a user submits the data by clicking on "Submit", the form data is sent to the file specified in the action attribute of the <form> tag. In this example, we point to the file itself for processing form data. If you wish to use another PHP file to process form data, replace that with the filename of your choice. Then, we can use the super global variable $\_POST to collect the value of the input field:

Example

<html>  
<body>  
  
<form method="post" action="<?php echo $\_SERVER['PHP\_SELF'];?>">  
  Name: <input type="text" name="fname">  
  <input type="submit">  
</form>  
  
<?php  
if ($\_SERVER["REQUEST\_METHOD"] == "POST") {  
    // collect value of input field  
    $name = $\_POST['fname'];  
    if (empty($name)) {  
        echo "Name is empty";  
    } else {  
        echo $name;  
    }  
}  
?>  
  
</body>  
</html>

1. PHP $\_GET

PHP $\_GET can also be used to collect form data after submitting an HTML form with method="get".

$\_GET can also collect data sent in the URL.

Assume we have an HTML page that contains a hyperlink with parameters:

<html>  
<body>  
  
<a href="test\_get.php?subject=PHP&web=W3schools.com">Test $GET</a>  
  
</body>  
</html>

When a user clicks on the link "Test $GET", the parameters "subject" and "web" are sent to "test\_get.php", and you can then access their values in "test\_get.php" with $\_GET.

The example below shows the code in "test\_get.php":

Example

<html>  
<body>  
  
<?php   
echo "Study " . $\_GET['subject'] . " at " . $\_GET['web'];  
?>  
  
</body>  
</html>

1. PHP : $\_FILES

$\_FILES is a super global variable which can be used to upload files. Here we will see an example in which our php script checks if the form to upload the file is being submitted and generates a message if true.

Example

Here is the html code (upload.php):

<html>

<body>

<form action="upload\_file.php"method="post"

enctype="multipart/form-data">

<label for="file">Filename:</label>

<input type="file"name="file"id="file"/>

<br />

<input type="submit"name="submit"value="Submit"/>

</form>

</body>

</html>

**Code of files.php file:**

<?php

if($\_FILES["file"]>0)

{

echo"You have selected a file to upload";

}

?>

1. PHP: $\_ENV

$\_ENV is used to return the environment variables from the web server.

<?php

echo $\_ENV['username'];

?>

Example

<?php

echo"my username is ".$\_ENV['username'];

?>

1. PHP: $\_COOKIE

Cookies are small text files loaded from a server to a client computer storing some information regarding the client computer, so that when the same page from the server is visited by the user, necessary information can be collected from the cookie itself, decreasing the latency to open the page.

Example

$\_COOKIE retrieves those cookies.

<?php

setrawcookie();

print\_r($\_COOKIE);

?>

1. PHP: $\_SESSION

Sessions are wonderful ways to pass variables. All you need to do is start a session by session\_start();Then all the variables you store within a $\_SESSION, you can access it from anywhere on the server. Here is an example:

Example

<?php

session\_start();

$\_SESSION['w3resource']='The largest online tutorial';

echo $\_SESSION['w3resource'];

?>