PHP - $_SESSION

One of the superglobal variables in PHP, **$_SESSION** is an associative array of session variables available in the current script. **$HTTP_SESSION_VARS** also contains the same information, but it is not a superglobal, and it has now been deprecated.

What is a Session?

A Session is an alternative way to make data accessible across the pages of an entire website. It is the time duration between the time a user establishes a connection with a server and the time the connection is terminated. During this interval, the user may navigate to different pages. Many times, it is desired that some data is persistently available across the pages. This is facilitated by **session variables**.

A session creates a file in a temporary directory on the server where the registered session variables and their values are stored. This data will be available to all the pages on the site during that visit.

The server assigns a unique SESSIONID to each session. Since HTTP is a stateless protocol, data in session variables is automatically deleted when the session is terminated.

The session_start() Function

In order to enable access to session data, the session_start() function must be invoked. session_start() creates a session or resumes the current one based on a session identifier passed via a GET or POST request, or passed via a cookie.

```
session_start(array $options = []): bool
```

This function returns **true** if a session was successfully started, else it returns **false**.

Handling Session Variables

To create a new session variable, add a key-value pair in the $_SESSION array −

```
$_SESSION[ "var"]=value;
```

To read back the value of a session variable, you can use **echo/print** statements, or **var_dump()** or **print_r()** functions.

```
echo $_SESSION[ "var"];
```

To obtain the list of all the session variables in the current session, you can use a **foreach** loop to traverse the **$_SESSION** −

```
foreach ($_SESSION as $key=>$val)
echo $key . "=>" . $val;
```

To manually clear all the session data, there is **session_destroy()** function. A specific session variable may also be released by calling the **unset()** function.

```
unset($_SESSION[ "var"]);
```

List of Session Functions

In PHP, there are many built-in functions for managing the session data.

Session Functions	Description
session_abort	Discard session array changes and finish session
session_cache_expire	Return current cache expire
session_cache_limiter	Get and/or set the current cache limiter
session_commit	Alias of session_write_close
session_create_id	Create new session id
session_decode	Decodes session data from a session encoded string
session_destroy	Destroys all data registered to a session
session_encode	Encodes the current session data as a session encoded string
session_gc	Perform session data garbage collection
session_get_cookie_params	Get the session cookie parameters
session_id	Get and/or set the current session id
session_is_registered	Find out whether a global variable is registered in a

	session
session_module_name	Get and/or set the current session module
session_name	Get and/or set the current session name
session_regenerate_id	Update the current session id with a newly generated one
session_register_shutdown	Session shutdown function
session_register	Register one or more global variables with the current session
session_reset	Re-initialize session array with original values
session_save_path	Get and/or set the current session save path
session_set_cookie_params	Set the session cookie parameters
session_set_save_handler	Sets user-level session storage functions
session_start	Start new or resume existing session
session_status	Returns the current session status
session_unregister	Unregister a global variable from the current session
session_unset	Free all session variables
session_write_close	Write session data and end session

Example

The following PHP script renders an HTML form. The form data is used to create three session variables. A hyperlink takes the browser to another page, which reads back the session variables.

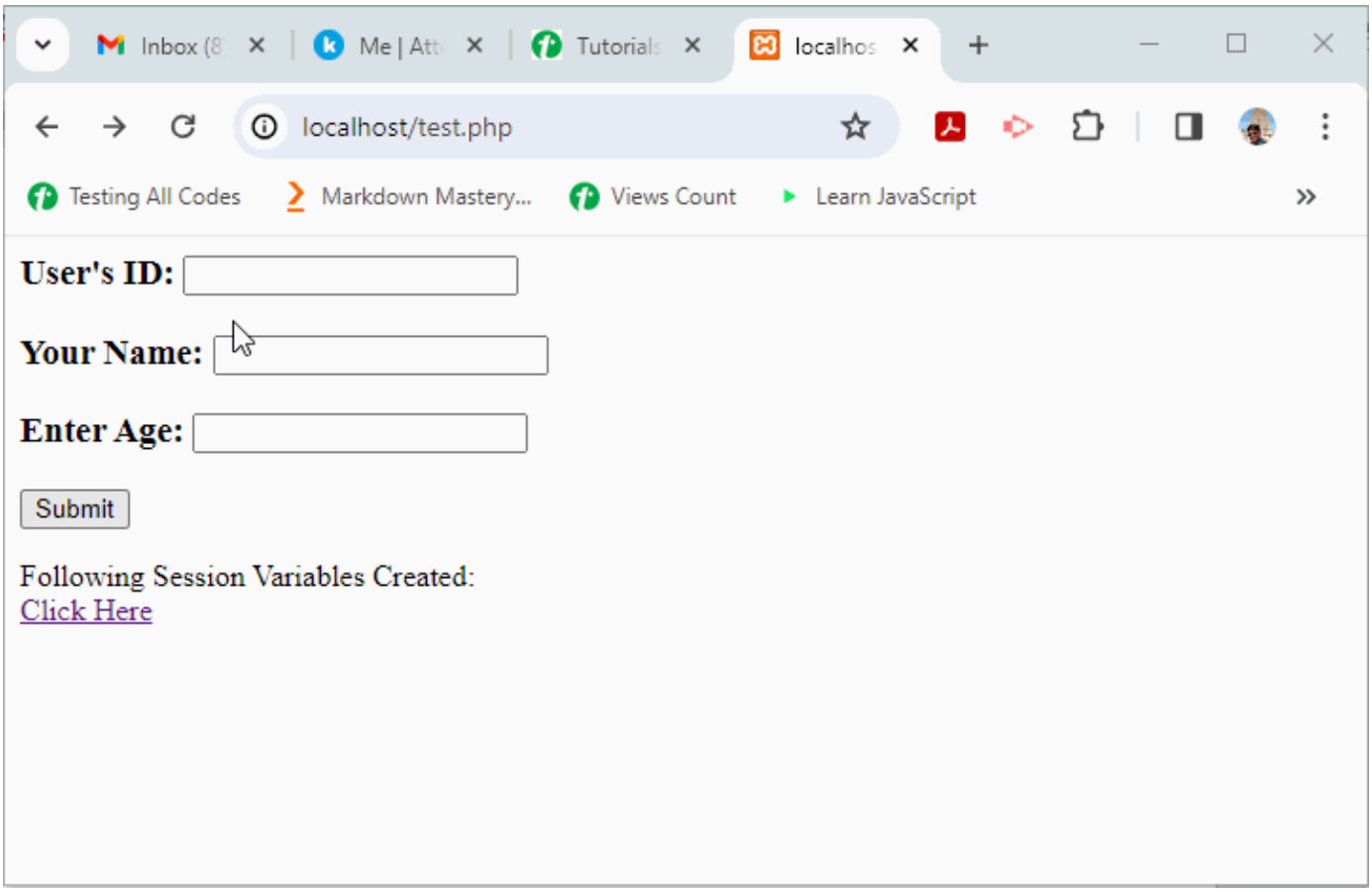
Save this code as "test.php" in the document root folder, and open it in a client browser. Enter the data and press the **Submit** button.

```
<html>
<body>
   <form action="<?php echo $_SERVER['PHP_SELF'];?>" method="post">
      <h3>User's ID: <input type="text" name="ID"/></h3>
      <h3>Your Name: <input type="text" name="name"/></h3>
      <h3>Enter Age: <input type="text" name="age"/></h3>
      <input type="submit" value="Submit"/>
   </form>
```

```
   <?php
      session_start();
      if ($_SERVER["REQUEST_METHOD"] == "POST") {
         $_SESSION['UserID'] = $_POST['ID'];
         $_SESSION['Name'] = $_POST['name'];
         $_SESSION['age'] = $_POST['age'];
      }
      echo "Following Session Variables Created: \n";

      foreach ($_SESSION as $key=>$val)
      echo "<h3>" . $key . "=>" . $val . "</h3>";
      echo "<br/>" . '<a href="hello.php">Click Here</a>';
   ?>
</body>
</html>
```

When you click the "Submit" button, it will show a list of all the session variables created –



Next, have the following script in the "hello.php" file and save it.

```
<?php
session_start();
   echo "<h2>Following Session variables Read:</h2>";
```

```
    foreach ($_SESSION as $key=>$val)
    echo "<h3>" . $key . "=>" . $val . "</h3>";
?>
```

Now, follow the link on the "test.php" page to navigate to "hello.php". It will show the session variables that are read –