Unit 3. PHP Forms

I.E.S. Zaidín Vergeles (Granada)

PHP Forms

What do we use forms for in a website?

Which different elements can we include in a form?

PHP Forms

Revise the HTML tags that allow us to write forms and answer these questions:

- What attributes must be written in the <form> tag?
- What values can these attributes have?
- What attribute is necessary to add to a tag inside a form in order to save the value a user introduces?

PHP Forms

The ***action*** attribute has two possible receivers:

- An email address.
- A script that will handle the data.

PHP Forms

The ***method*** attribute has two possible values:

- **GET**. The data appears after ? in the URL in the address bar and has a limitation of about 2000 characters.
- **POST**. The data is sent by HTTP request so it is invisible and has no limits on the amount of characters.

PHP Forms

Write an *HTML* form to ask for a person's name and mail and send them to this script. Then add the *var_dump* function to read the content of the *$_POST* variable.

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Greeting</title>
  </head>
  <body>
    <?php echo "<p>Welcome, ", $_POST["name"], "(", $_POST["mail"], ").</p>";?>
  </body>
</html>
```

$_GET and $_POST

They are global variables (*superglobals*) that contain associative arrays in which:

- The values of the index are taken from the ***name*** attribute of each element of the form.
- The values associated to each value of the index is taken from its corresponding ***value*** attribute.

$_GET and $_POST

An example of the former code is:

VALUE	INDEX	REFERENCE
Shirley Smith	name	*$_POST['name']*
ssmith22@gmail.com	mail	*$_POST['mail']*

$_GET and $_POST

Modify the previous example to a ***get*** sending method. Then, apply the *var_dump* function to read the type and the content of the *$_GET* variable.

$_GET and $_POST

Write an HTML document to take two numbers and send them to a script that calculates their addition, subtraction, multiplication, division and modulus.

$_GET and $_POST

When collecting information from a drop-down list that allows multiple selection, the data will be stored in an array inside *$_POST* or *$_GET*. For that purpose, we will add **[ ]**, at the end of the value assigned to the ***name*** attribute in the <*select*> tag.

$_GET and $_POST

Extend the form in slide 6 in this way:

- Add a multiple selection drop-down list containing the three possible roles in a school (student, teacher, service staff).
- The script will deliver a message that shows a person's name and his/her different roles.

$_GET and $_POST

What attribute do we write to make compulsory the filling out of a specific form element?

$_GET and $_POST

It is also possible to get to know if a variable is empty by using this function:

empty(variable_name)

$_GET and $_POST

Improve the previous code in this way:

- If the user hasn't written the name, a warning message will be delivered.
- If the user hasn't filled out the mail, another warning message will also appear.
- Only in case that both pieces of data have been introduced, the welcome message will appear.

$_SERVER

This global variable is an associative array created by the web server and contains information about HTTP headers, paths and script locations.

Web Application Implementation

TRY IT!

$_SERVER

```
<!DOCTYPE html>
<html lang=”en”>
  <head>
    <title>Form</title>
  </head>
  <body>
    <form method="post" action="<?php echo $_SERVER['PHP_SELF'];?>">
      <label>Name: </label><input type="text" name="name"/>
      <input type="submit"/>
    </form>
    <?php
      if ($_SERVER["REQUEST_METHOD"] == "POST") {
        echo $_POST['name'];
      }
    ?>
  </body>
</html>
```

Explain the purpose of the *PHP* code.

$_SERVER

Include in the previous script the code to deliver a table to display all the index-value pairs hold in the *$_SERVER* variable.

$_SERVER

Is it advisable to use this code?

action="<?php echo $_SERVER['PHP_SELF'];?>"

Read about it in:

https://www.w3schools.com/php/php_form_validation.asp

$_SESSION

The variables declared in a specific script are local, so they cannot be used by other scripts. However, it is usually useful to get to them from different pages of the website.

$_SESSION

Think of the situation described in the former slide and tell an example based on your experience.

$_SESSION

$_SESSION is a global array which holds the data that will be available in multiple pages.

It is stored in the server and, once the user closes the browser, this variable information is deleted.

$_SESSION

To create a session the ***session_start()*** function must be written at the beginning of the document (even before HTML tags).

Web Application Implementation

TRY IT!

$_SESSION

```
<?php
  session_start();
?>
<!DOCTYPE html>
<html lang=”en”>
  <body>
    <?php
      $_SESSION["user"] = "123456";
      $_SESSION["role"] = "customer";
    ?>
  </body>
</html>
```

What is *$_SESSION* in this example?

Web Application Implementation

TRY IT!

$_SESSION

```
<?php
  session_start();
?>
<!DOCTYPE html>
<html lang=”en”>
  <body>
    <?php
      echo "<div>User: ", $_SESSION["user"], "</div>";
      echo "<div>Role: ", $_SESSION["role"], "</div>";
    ?>
  </body>
</html>
```

What happens if you run this code without closing the browser?

$_SESSION

Create a project consisting of:

- An HTML document with a form to get a user's name and surname.
- A PHP document that stores this information in a session and also has a link to a second PHP document.
- The linked PHP document which delivers a greeting to that person, including their name and surname.

$_SESSION

Using the code in slide 25, add a script to change the user's role to *"provider"*.

$_SESSION

Add the *session_unset()* function at the end of the former script and check the content of *$_SESSION*. What has happened?

Web Application Implementation

TRY IT!

$_SESSION

```
<?php
  session_start();
?>
<!DOCTYPE html>
<html lang=”en”>
  <body>
    <?php
       if(!isset($_SESSION["user"])){
          echo "The session is closed";
       }
    ?>
  </body>
</html>
```

- What happens if you run this code without closing the browser?
- What is the purpose of the *isset()* function?

$_SESSION

After using the ***session_unset()*** function, the content of *$_SESSION* is deleted but, if needed, new values can be stored in it.

$_SESSION

However, to definitely remove the content of *$_SESSION*, the ***session_destroy()*** function must be used.

$_SESSION

Think of possible situations in which *session_destroy()* function is useful.

$_COOKIE

A **cookie** is a small file that the server embeds on the user's computer, normally, to identify it and save its preferences. Each time a user requests a web page with a browser, the cookie is sent to the server.

This information is stored in the global array ***$_COOKIE***.

$_COOKIE

You can have a look at this web page in order to get to know about the use of cookies in a specific website:

https://www.caminitodelrey.info/es/aviso-cookies/politica

$_COOKIE

$_COOKIE is created by the ***setcookie()*** function, whose syntax is:

setcookie(name, value, expire, path, domain, secure, httponly);

$_COOKIE

In the ***setcookie()*** function:

- Only the name is required, the rest of the parameters are optional.
- The expiring date is expressed as a number of seconds, so it is usually a multiple of 86400 (seconds in a day) from the current time.

*time() + 86400 * number_of_days*

$_COOKIE

In the ***setcookie()*** function:

- If the expiring date is not written or equals zero, the cookie will expire when the browser gets closed.

$_COOKIE

In the ***setcookie()*** function:

- The path parameter points out the part of the domain in which the cookie is available.
  - *'/'*, for the whole domain.
  - *'/folder_name/'*, for this folder and its subfolders.

$_COOKIE

Search the internet for the domain, secure and httponly parameters and their values.

$_COOKIE

TRY IT!

```
<!DOCTYPE html>
  <?php
    $cookie_name = "user";
    $cookie_value = "Margareth Keaton";
    setcookie($cookie_name, $cookie_value, time() + 86400 * 30, "/");
  ?>
<html lang=”en”>
   <head>
     <title>Cookies</title>
  </head>
    ...
```

Web Application Implementation

TRY IT!

$_COOKIE

```
...
<body>
  <?php
     if(!isset($_COOKIE[$cookie_name])) {
      echo "Cookie " , $cookie_name , " is not set!";
     }
    else {
      echo "Value of " , $cookie_name , " is: " , $_COOKIE[$cookie_name];
  }
  ?>
</body>
</html>
```

$_COOKIE

REVISE

- What is *time()*?
- What is the meaning of *time() + 86400 * 30*?
- What is the meaning of '/' in the *setcookie()* function?
- What do we use *isset()* function for?
- What is the type and content of *$_COOKIE*?

$_COOKIE

Add a new cookie to the previous code called *password* with the value *Qpf5n*. Then, check the content of *$_COOKIE* again.

$_COOKIE

The ***setcookie()*** function also allows to change the value of a cookie just writing a new value in its second parameter.

$_COOKIE

Using the previous document, change the value of the *user* to *Donald Watson*. Then, check the content of the *$_COOKIE* variable.

TRY IT!

$_COOKIE

```php
<?php
   setcookie("usuario", "", time() - 3600);
?>
```

What can be the purpose of this code? Why?

$_SESSION AND $_COOKIE

Write a table to compare different aspects of these two superglobals (similarities and differences).

PHP Forms

Take the form you created in Markup Languages subject last year and develop the following tasks:

1. Once the form is fill out, a web page is displayed with the customer's information and the price of the sandwich. You can apply different prices depending on the size, the kind and number of ingredients and if it is sent to home. Add a link that allows the customer to confirm the ordering.
2. Finally, a new page appears with the customer's name and a message about the time in which the sandwich will be delivered or ready to pick up in the establishment.

Don't forget to save the customer's information for future sales.