Web Programming with PHP

Sergio Luján Mora
Department of Software and Computing Systems
University of Alicante (Spain)
WEB APPLICATION FRAMEWORK

Index

- Predefined variables
- Form handling
- Session handling
Predefined variables (1)

- Method for retrieving external variables
- They are superglobal: they are available in every context, sin tener que declararlas con global
- Associate arrays (key, value)
- Content depends on execution mode (web, command line) and operating system

Predefined variables (2)

- Predefined variables:
  - $GLOBALS: access to global variables
  - $_COOKIE: cookies
  - $_ENV: environment variables
  - $_FILES: files uploaded to the current script via the HTTP POST method
  - $_GET: data passed to the current script via the HTTP GET method
  - $_POST: data passed to the current script via the HTTP POST method
  - $_REQUEST: $_GET, $_POST y $_COOKIE
  - $_SERVER: data provided by the web server
  - $_SESSION: session variables available to the current script
Predefined variables (3)

- $_ENV: environment variables
- Example:
  - HOME
  - PATH
  - PWD
  - USER
  - ...

Predefined variables (4)

- $_SERVER: data provided by the web server
- A large number of these variables are from CGI
- Example:
  - HTTP_ACCEPT
  - PATH
  - REMOTE_ADDR
  - REMOTE_PORT
  - SERVER_NAME
  - SERVER_PORT
  - ...
Predefined variables (and 5)

- Example:
  ```php
  echo "$_SERVER['REMOTE_ADDR']";
  ```
- How to get the key and the value of each element:
  ```php
  reset($_SERVER);
  while((list($key, $value) = each($_SERVER)) != FALSE)
    echo "$key => $value<br>";
  ```

Form handling (1)

- Access through global predefined variables:
  - $_GET: data sent by GET method or through URL
  - $_POST: data sent by POST method
  - $_REQUEST: content of $_GET and $_POST
- If the register_globals directive is set on php.ini, then these variables will also be made available in the global scope of the script
  - Due to security problems, this feature is deprecated in PHP 5 and removed in PHP 6
Form handling (2)

- Form:
  ```html
  <form action="respuesta.php?value=10" method="post">
  <input type="text" name="name">
  </form>
  ```
- PHP (are all of them possible?):
  ```php
  <?
  echo 'value: ' . $_GET['value'] . '<br>';
  echo 'value: ' . $_POST['value'] . '<br>';
  echo 'value: ' . $_REQUEST['value'] . '<br>';
  echo 'name: ' . $_GET['name'] . '<br>';
  echo 'name: ' . $_POST['name'] . '<br>';
  echo 'name: ' . $_REQUEST['name'] . '<br>';
  ?>
  ```

Form handling (3)

- Form:
  ```html
  <form action="respuesta.php?value=10" method="post">
  <input type="text" name="name">
  </form>
  ```
- PHP (are all of them possible?):
  ```php
  <?
  echo 'value: ' . $_GET['value'] . '<br>';
  echo 'value: ' . $_POST['value'] . '<br>';
  echo 'value: ' . $_REQUEST['value'] . '<br>';
  echo 'name: ' . $_GET['name'] . '<br>';
  echo 'name: ' . $_POST['name'] . '<br>';
  echo 'name: ' . $_REQUEST['name'] . '<br>';
  ?>
  ```
Form handling (4)

- Array values from a form:
  - `<select multiple>`
  - `<input type="checkbox" />`
  - Write “[]” at the end of the name in the HTML
  - We can use `count()` to get the number of elements in the array

Form handling (and 5)

- Form:
  ```html
  <select name="lista[]" multiple>
  <option>Spain</option>
  <option>Russia</option>
  <option>Germany</option>
  </select>
  ```

- PHP:
  ```php
  <?
  $lista =  $_POST['lista'];
  for($i = 0; $i < count($lista); $i++)
    echo "$lista[$i]<br>";
  ?>
  ```
Session handling (1)

- Session support consists of a way to preserve certain data across subsequent accesses.
- A visitor accessing your web site is assigned a unique id, the so-called session id.
  - This is either stored in a cookie on the user side or is propagated in the URL.
- The session support allows you to register arbitrary numbers of variables to be preserved across requests.

Session handling (2)

- When a visitor accesses your site, PHP will check whether a specific session id has been sent with the request.
  - If this is the case, the prior saved environment is recreated.
Session handling (3)

- **session_start()**:  
  - Creates a session or resumes the current one based on the current session id

- **session_register('variable')**:  
  - Register one or more global variables with the current session

- **session_id()**:  
  - Get and/or set the current session id

Session handling (4)

- **session_unregister('variable')**:  
  - Unregister a global variable from the current session

- **session_is_registered('variable')**:  
  - Returns TRUE if there is a global variable with the name registered in the current session, FALSE otherwise

- **session_destroy()**:  
  - Destroys all data registered to a session
Session handling (5)

- Page 1:
  ```php
  <?
  session_start();
  session_register('counter');
  $counter++;
  ?>
  <html> ... </html>

- Page 2:
  ```php
  <? session_start(); ?>
  <html>
  <body>
  <?
  echo "counter: $counter";
  ?>
  </body>
  </html>
  ```

Session handling (6)

- Alternative way: global predefined variable `$_SESSION`
- You don’t need:
  - `session_register()`
  - `session_unregister()`
  - `session_is_registered()`
- We can use array functions (`count`, `foreach`, etc.) to consult session data
- Recommended for improved security and code readability
Session handling (and 7)

- Page 1:
  ```php
  <?
  session_start();
  $_SESSION['counter']++;
  ?>
  <html> ... </html>
```

- Page 2:
  ```php
  <?
  session_start();
  ?>
  <html>
  <body>
  <?
  echo "counter: " . $_SESSION['counter'];
  ?>
  </body>
  </html>
```