# Desarrollo Web en Entorno Servidor

## 6.- PHP – Programación Web


IES Severo Ochoa

# Índice



- Variables de servidor
- Formularios
  - Subiendo archivos
- Gestión del estado
  - Cookies
  - Sesión
- Gestión de usuarios

# Variables de servidor

ies severo ochoa

- PHP almacena la información del servidor y de las peticiones HTTP en seis arrays globales:
  - `$_ENV`: información sobre las variables de entorno
  - `$_GET`: parámetros enviados en la petición GET
  - `$_POST`: parámetros enviados en el envio POST
  - `$_COOKIE`: contiene las cookies de la petición, las claves del array son los nombres de las cookies
  - `$_SERVER`: información sobre el servidor
  - `$_FILES`: información sobre los ficheros cargados via upload

# $_SERVER I

- https://www.php.net/manual/es/reserved.variables.server.php
- `PHP_SELF`: nombre del script ejecutado, relativo al *document root (p.ej: `/tienda/carrito.php`)*
- `SERVER_SOFTWARE`: (p.ej: Apache)
  - `SERVER_NAME`: dominio, alias DNS (p.ej: `www.elche.es`)
- `REQUEST_METHOD`: GET
  - `REQUEST_URI`: URI, sin el dominio
  - `QUERY_STRING`: todo lo que va después de ? en la URL (p.ej: `heroe=Batman&nombre=Bruce`)

# $_SERVER II

- `PATH_INFO:` ruta extra tras la petición. Si la URL es `http://www.php.com/php/pathInfo.php/algo/cosa?foo=bar`, entonces `$_SERVER['PATH_INFO']` será `/algo/cosa`.
- `REMOTE_HOST`: *hostname* que hizo la petición
  - *`REMOTE_ADDR:`* IP del cliente
- `AUTH_TYPE`: tipo de autenticación (p.ej: Basic)
  - `REMOTE_USER`: nombre del usuario autenticado
- Apache crea una clave para cada cabecera HTTP, en mayúsculas y sustituyendo los guiones por subrayados:
  - `HTTP_USER_AGENT`: agente (navegador)
  - `HTTP_REFERER`: página desde la que se hizo la petición

# Ejemplo $_SERVER

ies severo ochoa

```php
<?php

echo $_SERVER["PHP_SELF"]."<br>"; // /u6/601server.php
echo $_SERVER["SERVER_SOFTWARE"]."<br>"; // Apache/
2.4.46 (Win64) OpenSSL/1.1.1g PHP/7.4.9
echo $_SERVER["SERVER_NAME"]."<br>"; // localhost

echo $_SERVER["REQUEST_METHOD"]."<br>"; // GET
echo $_SERVER["REQUEST_URI"]."<br>"; // /u6/601server.php?
heroe=Batman
echo $_SERVER["QUERY_STRING"]."<br>"; // heroe=Batman

echo $_SERVER["HTTP_USER_AGENT"]."<br>"; // Mozilla/
5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/
537.36 (KHTML, like Gecko) Chrome/87.0.4280.88 Safari/537.36
```

# GET vs POST

ies severo ochoa

- GET: los parámetros se pasan en la URL
  - <2048 caracteres, sólo ASCII
  - Permite almacenar la dirección completa (marcador / historial)
  - *Idempotente*: dos llamadas con los mismos datos siempre debe dar el mismo resultado
  - El navegador puede cachear las llamadas

- POST: parámetros ocultos (no encriptados)
  - Sin límite de datos, permite datos binarios.
  - No se pueden cachear
  - No idempotente → actualizar la BBDD

# Recogiendo los datos

- `$par = $_GET["parametro"]`
- `$par = $_POST["parametro"]`

- Es conveniente siempre hacer validación doble:
  - En el cliente mediante JS
  - En servidor, antes de llamar a negocio, es conveniente volver a validar los datos.
    - Librerías: respect/validation, particle/validator

```
if (isset($_GET["parametro"])) {
  $par = $_GET["parametro"];
  // comprobar si $par tiene el formato adecuado, su valor, etc...
}
```

# Ejemplo de formulario

ies severo ochoa

```
<form action="formulario.php" method="GET">
<p>Nombre del alumno:
<input type="text" name="nombre" value="" />
</p>

<p><input type="checkbox" name="modulos[]" value="DWES" />
Desarrollo web en entorno servidor<br />
</p>

<p><input type="checkbox" name="modulos[]" value="DWEC" />
Desarrollo web en entorno cliente<br />
</p>
<input type="submit" value="Enviar" name="enviar"/>
</form>
```

# Generando el formulario

ies severo ochoa

```
<?php
if (!empty($_POST['modulos']) && !empty($_POST['nombre'])) {
// Aquí se incluye el código a ejecutar cuando los datos son correctos
} else {
// Generamos el formulario
?>
  <form action="<?php echo $_SERVER['PHP_SELF'];?>" method="GET">
  <p>Nombre del alumno:
  <input type="text" name="nombre" value="" />
  </p>
  <p><input type="checkbox" name="modulos[]" value="DWES" />
  Desarrollo web en entorno servidor<br />
  </p>
  <p><input type="checkbox" name="modulos[]" value="DWEC" /> Desarrollo
web en entorno cliente<br />
  </p>
  <input type="submit" value="Enviar" name="enviar"/>
  </form>
<?php } ?>
```

# Recogiendo parámetros multivalor

ies severo ochoa

- Elementos que envían varios valores
  - `select multiple`
  - `checkbox`
- El nombre del elemento debe ser un array

```
<select name="lenguajes[]" multiple="true">
<option name="c">C</option>
<option name="java">Java</option>
<option name="php">PHP</option>
<option name="python">Python</option>
</select>
```

```
<input type="checkbox" name="lenguajes[]" value="c" /> C<br />
<input type="checkbox" name="lenguajes[]" value="java" /> Java<br />
<input type="checkbox" name="lenguajes[]" value="php" /> Php<br />
<input type="checkbox" name="lenguajes[]" value="python" /> Python<br />
```

# Volviendo a rellenar un formulario – sticky forms

ies severo ochoa

```php
if (!empty($_POST['modulos']) && !empty($_POST['nombre'])) {
  // Aquí se incluye el código a ejecutar cuando los datos son correctos
} else {
  // Generamos el formulario
  $nombre = $_POST['nombre'] ?? "";
  $modulos = $_POST['modulos'] ?? [];
  ?>
  <form action="<?php echo $_SERVER['PHP_SELF'];?>" method="POST">
   <p>Nombre del alumno:
   <input type="text" name="nombre" value="<?= $nombre ?>" />
   </p>
   <p><input type="checkbox" name="modulos[]" value="DWES"
   <?php if(in_array("DWES",$modulos)) echo 'checked="checked"'; ?>" />
   Desarrollo web en entorno servidor<br />
   </p>
   <p><input type="checkbox" name="modulos[]" value="DWEC"
   <?php if(in_array("DWEC",$modulos)) echo 'checked="checked"'; ?>" />
   Desarrollo web en entorno cliente<br />
   </p>
   <input type="submit" value="Enviar" name="enviar"/>
  </form>
<?php } ?>
```

# Subiendo archivos

- Se almacenan en `$_FILES` con el nombre del campo del tipo `file`.
  - Propiedad opcional `MAX_FILE_SIZE` restringe el tamaño del archivo a subir (nunca superior al especificado en `php.ini`)

```
<form enctype="multipart/form-data" action="<?php echo $_SERVER['PHP_SELF']; ?>"
 method="POST">
   <input type="hidden" name="MAX_FILE_SIZE" value="10240">
   Archivo: <input name="archivoEnviado" type="file" />
   <br />
   <input type="submit" name="btnSubir" value="Subir" />
</form>
```

# Configuración en `php.ini`

ies severo ochoa

- `file_uploads`: on / off
- `upload_max_filesize`: 2M
- `upload_tmp_dir`: directorio temporal. No es necesario configurarlo, cogerá el predeterminado del sistema
- `post_max_size`: tamaño máximo de los datos POST. Debe ser mayor a upload_max_filesize.
- `max_file_uploads`: número máximo de archivos que se pueden cargar a la vez.
- `max_input_time`: tiempo máximo empleado en la carga (GET/POST y upload → normalmente se configura en 60)
- `memory_limit`: 128M
- `max_execution_time`: tiempo de ejecución de un script (no tiene en cuenta el upload)

# Cargando los archivos

```php
if (isset($_POST['btnSubir']) && $_POST['btnSubir'] == 'Subir') {
    if (is_uploaded_file($_FILES['archivoEnviado']['tmp_name'])) {
        // subido con éxito
        $nombre = $_FILES['archivoEnviado']['name'];
        move_uploaded_file($_FILES['archivoEnviado']['tmp_name'],
        "./uploads/{$nombre}");

        echo "<p>Archivo $nombre subido con éxito</p>";
    }
}
```

- Cada archivo cargado en $_FILES tiene:
  - `name`: nombre
  - `tmp_name`: ruta temporal
  - `size`: tamaño en bytes
  - `type`: tipo MIME
  - `error`: si hay error, contiene un mensaje. Si ok → 0.

# Cabeceras de respuesta

- Debe ser lo primero a devolver.
- `header(cadena)`
  - permite configurar el tipo de contenido, tiempo de expiración, redireccionar el navegador, especificar errores HTTP, etc.

```
<?php header("Content-Type: text/plain"); ?>
<?php header("Location:
http://www.ejemplo.com/inicio.html");
   exit();
```

- Se puede comprobar en `Developer Tools → Network → Headers`

# Expirando documentos

ies severo ochoa

- Permiten evitar consultar la caché o provocar su renovación:

```php
<?php
header("Expires: Sun, 31 Jan 2021 23:59:59 GMT");
// tres horas
$now = time();
$horas3 = gmstrftime("%a, %d %b %Y %H:%M:%S GMT", $now + 60 * 60 * 3);
header("Expires: {$horas3}");
// un año
$now = time();
$anyo1 = gmstrftime("%a, %d %b %Y %H:%M:%S GMT", $now + 365 * 86440);
header("Expires: {$anyo1}");
// se marca como expirado (fecha en el pasado)
$pasado = gmstrftime("%a, %d %b %Y %H:%M:%S GMT");
header("Expires: {$pasado}");
// evitamos cache de navegador y/o proxy
header("Expires: Mon, 26 Jul 1997 05:00:00 GMT");
header("Last-Modified: " . gmdate("D, d M Y H:i:s") . " GMT");
header("Cache-Control: no-store, no-cache, must-revalidate");
header("Cache-Control: post-check=0, pre-check=0", false);
header("Pragma: no-cache");
```

# Gestionando el estado

ies severo ochoa

- HTTP es un protocolo *stateless*
- Se simula el estado mediante el uso de cookies, tokens o la sesión
  - Carrito de la compra, operaciones asociadas a un usuario, etc...
- El mecanismo de PHP para gestionar la sesión emplea cookies por debajo
- Las cookies se almacenan en el navegador, y la sesión en el servidor web

# Cookies I

- https://www.php.net/manual/es/features.cookies.php
- `$_COOKIE`
- Archivos que se guardan en el cliente
  - El cliente puede no querer almacenarlas
  - 20 por dominio, 300 en total en el navegador
- Para crear una *cookie*:
  - **`setcookie`**`(nombre [, valor [, expira [, ruta [, dominio [, seguro [, httponly ]]]]]]);`
  - **`setcookie`**`(nombre [, valor = "" [, opciones = [] ]] )`
    - `nombre`: sin espacios ni `;`
    - `valor` < 4 KB

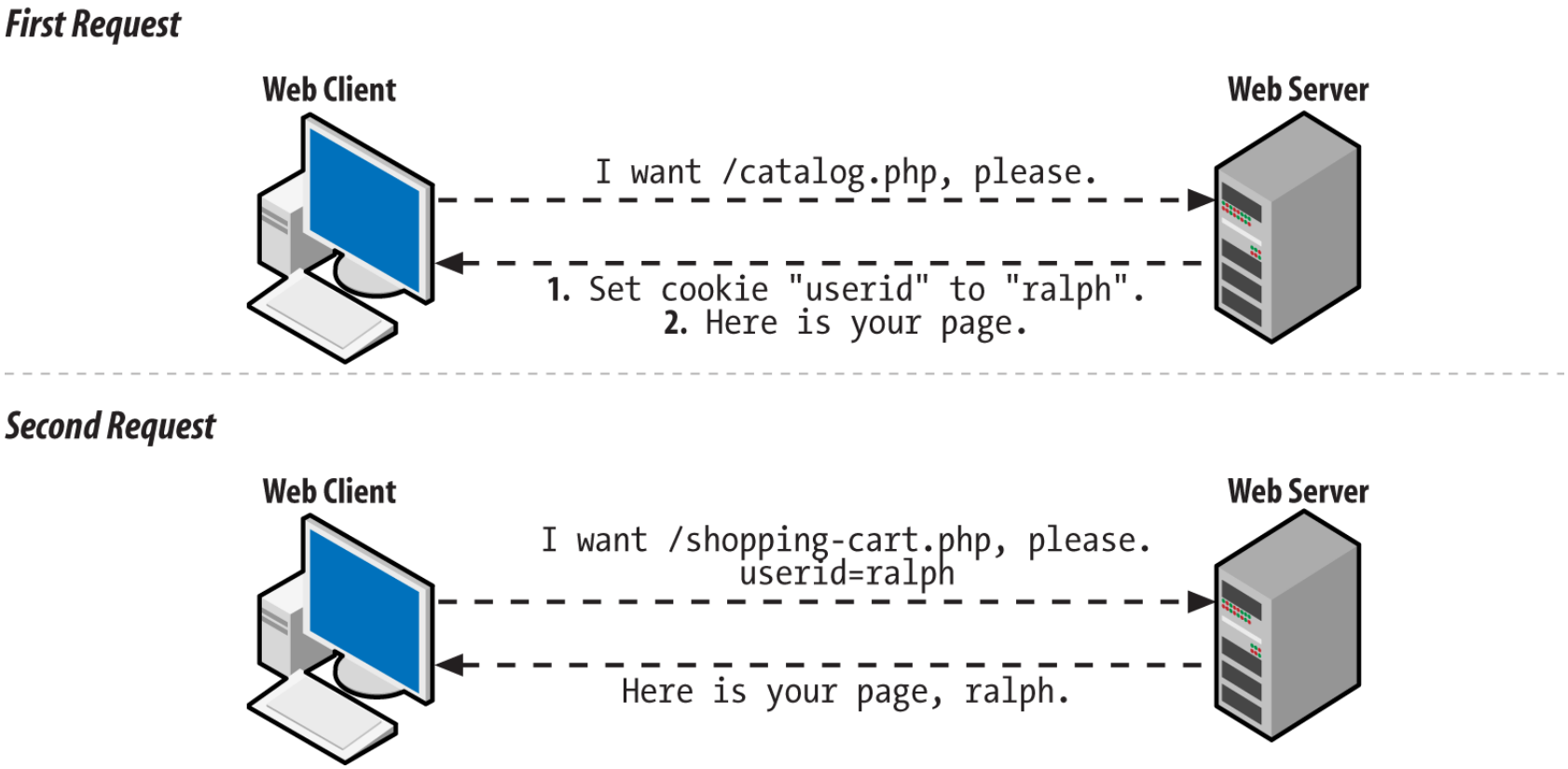
# Cookies II

- Pueden durar tanto como el sitio web. Ellas seguirán ahí, incluso si el navegador está cerrado o abierto.

- Para borrar una cookie (expira en el pasado):
  - `setcookie(nombre, "", 1) // pasado`

- O que caduquen (dentro de una hora):
  - `setcookie(nombre, valor, time + 3600)`

# Comunicación con cookies



# Uso de las Cookies

ies severo ochoa

- Se utilizan para:
  - Recordar los inicios de sesión
  - Almacenar valores temporales de usuario
    - Si un usuario está navegando por una lista paginada de artículos, ordenados de cierta manera, podemos almacenar el ajuste de la clasificación.

- Alternativa en el cliente: *LocalStorage*

# Ejemplo cookies

- Se pueden comprobar su valor en `Dev Tools → Application → Storage`

```php
<?php
$accesosPagina = 0;
if (!isset($_COOKIE['accesos'])) {
  $accesosPagina = $_COOKIE['accesos'
];
  setcookie('accesos', ++
$accesosPagina);
}
?>
```

# Sesión

ies severo ochoa

- https://www.php.net/manual/es/book.session.php
- Añade la gestión del estado a HTTP.
- Cada visitante tiene un ID de sesión único.
- Por defecto se almacena en una cookie denominada `PHPSESSID`.
- Si el cliente no tiene las cookies activas, el ID se propaga en cada URL dentro del mismo dominio.
- Cada sesión tiene asociado un almacén de datos, en el cual podemos almacenar y recuperar información → `$_SESSION`

# Comunicación con sesión

# Operaciones con la sesión

ies severo ochoa

- La sesión comienza al ejecutar un script PHP. Se genera un nuevo ID y se cargan los datos del almacén.

- Operaciones:
  - `session_start(); // carga la sesión`
  - `session_id() // devuelve el id`
  - `session_destroy(); // vacía la sesión`
  - `$_SESSION[clave] = valor; // inserción`
  - `unset($_SESSION[clave]; // borrado`

# Ejemplo de uso de sesión

ies severo ochoa

---- 606sesion1.php ----

```php
<?php
session_start(); // inicializamos
$_SESSION["ies"] = "IES Severo Ochoa"; // asignación
$instituto = $_SESSION["ies"]; // recuperación
echo "Estamos en el $instituto";
?>
<br />
<a href="606sesion2.php">Y luego</a>
```

---- 606sesion2.php ----

```php
<?php
session_start();
$instituto = $_SESSION["ies"]; // recuperación
echo "Otra vez, en el $instituto";
?>
```

# Configurando la sesión en php.ini

ies severo ochoa

- https://www.php.net/manual/es/session.configuration.php
- `session.save_handler`: controlador que gestiona cómo se almacena (`files`)
- `session.save_path`: ruta donde se almacenan los archivos con los datos (`/xampp/sessions`)
- `session.name: nombre de la sesión` (`PHSESSID`)
- `session.auto_start`: Se puede hacer que se autocargue con cada script, por defecto está deshabilitado
- `session.cookie_lifetime`: tiempo de vida

# Gestión de usuarios

ies severo ochoa

- Una sesión establece una relación anónima con un usuario particular.
- Al hacer login, sabremos quien utiliza nuestra aplicación.
- Pasos:
  1) Mostrar el formulario login/password
  2) Comprobar los datos enviados
  3) Añadir el login a la sesión
  4) Comprobar el login en la sesión para realizar tareas específicas del usuario
  5) Eliminar el login de la sesión cuando el usuario la cierra.

# 610index.php

ies severo ochoa

```
<form action='611login.php' method='post'>
  <fieldset>
    <legend>Login</legend>
     <div><span class='error'><?php echo $error; ?></span></div>
     <div class='fila'>
        <label for='usuario'>Usuario:</label><br />
        <input type='text' name='inputUsuario' id='usuario'
maxlength="50" /><br />
     </div>
     <div class='fila'>
        <label for='password'>Contraseña:</label><br />
        <input type='password' name='inputPassword' id='password'
maxlength="50" /><br />
     </div>
     <div class='fila'>
        <input type='submit' name='enviar' value='Enviar' />
     </div>
  </fieldset>
  </form>
```

# 611login.php

ies severo ochoa

```php
<?php
// Comprobamos si ya se ha enviado el formulario
if (isset($_POST['enviar'])) {
    $usuario = $_POST['inputUsuario'];
    $password = $_POST['inputPassword'];

    // validamos que recibimos ambos parámetros
    if (empty($usuario) || empty($password)) {
        $error = "Debes introducir un usuario y contraseña";
        include "610index.php";
    } else {
        if ($usuario == "admin" && $password == "admin") {
            // almacenamos el usuario en la sesión
            session_start();
            $_SESSION['usuario'] = $usuario;
            // cargamos la página principal
            include "612main.php";
        } else {
            // Si las credenciales no son válidas, se vuelven a pedir
            $error = "Usuario o contraseña no válidos!";
            include "610index.php";
        }
    }
}
```

# 612main.php

ies severo ochoa

```php
<?php
    // Recuperamos la información de la sesión
    if(!isset($_SESSION)) {
        session_start();
    }

    // Y comprobamos que el usuario se haya autentificado
    if (!isset($_SESSION['usuario'])) {
       die("Error - debe <a href='610index.php'>identificarse</a>.<br />");
    }
?>
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Listado de XXXX</title>
</head>
<body>
    <h1>Bienvenido <?= $_SESSION['usuario'] ?></h1>
    <p>Pulse <a href="613salir.php">aquí</a> para salir</p>
    <p>Volver al <a href="612main.php">inicio</a></p>
    <h2>Listado de XXXX</h2>
</body>
</html>
```

# 613logout.php

ies severo ochoa

```php
<?php
    // Recuperamos la información de la sesión
    session_start();

    // Y la eliminamos
    session_unset();
    header("Location: 610index.php");
?>
```

ies severo ochoa

# ¿Alguna pregunta?