HLC

2º DAM

I.E.S. POLITÉCNICO H.LANZ
JOSÉ MARÍA MOLINA

TEMA 1-3 – PROGRAMACIÓN BÁSICA EN PHP (II)

TEMA 1 – INTRODUCCIÓN A PHP

Vamos a ver conceptos básicos de programación en PHP.

- ✓1 DEPURACIÓN DE ERRORES
- ✓2 VARIABLES Y CONSTANTES DEL SISTEMA
- ✓3 PASO DE VARIABLES

1 DEPURACIÓN DE ERRORES

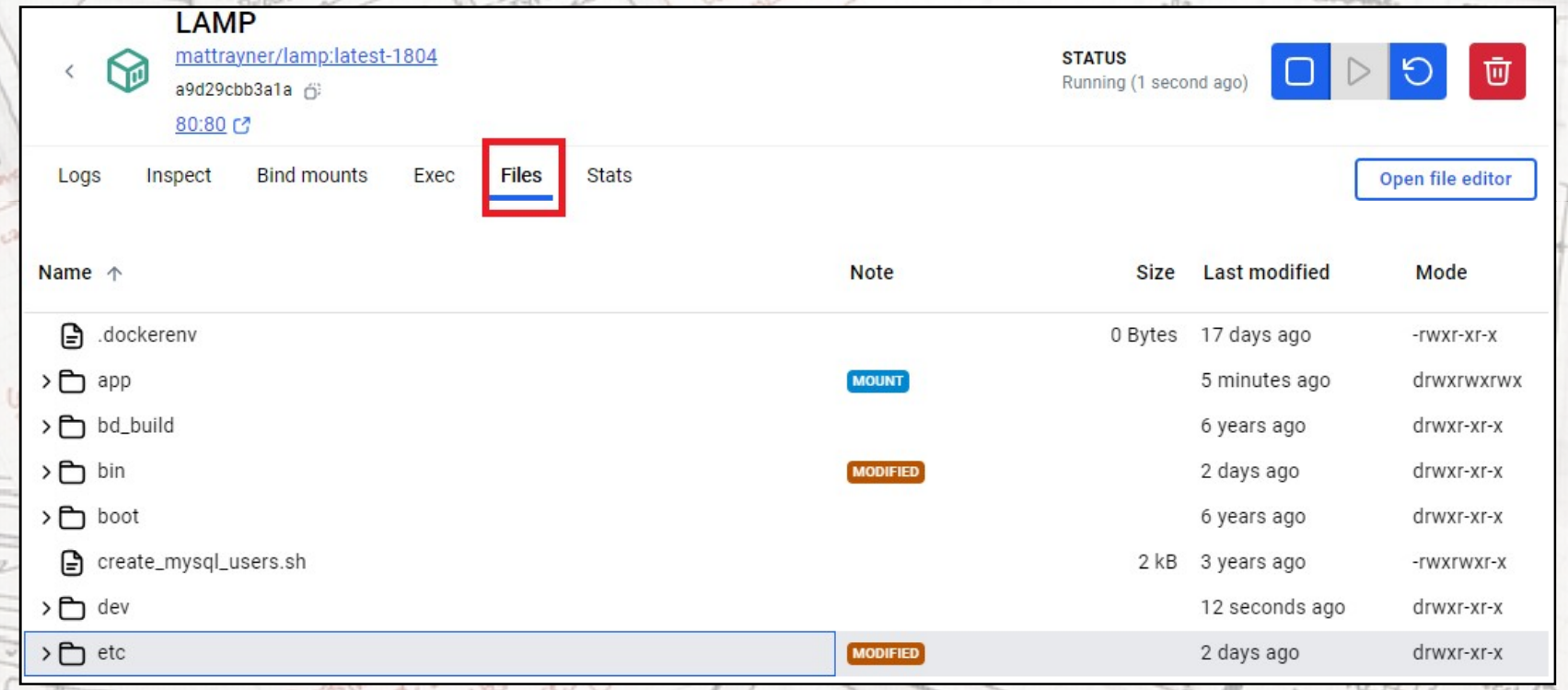
DEPURACIÓN DE ERRORES

Loaded Configuration File	/etc/php/8.0/apache2/php.ini

- Errores de SINTAXIS: ifa, fore, functiona… Provocan que no se procese el PHP!!!
- No detecta este tipo de errores (estos los detectamos desde el editor al usar un plugin como por ejemplo *All-in-One PHP*)
- Sí detecta errores en tiempo de ejecución, de lógica, warnings, etc…
- **ERRORES EN DESARROLLO – CAMBIO EN SERVIDOR**
  - Buscar el fichero php.ini (se puede ver ruta al utilizar phpinfo())
  - Buscar línea y cambiar **display_errors = Off** por **display_errors = On**
  - Se puede instalar nano y buscar la línea (apt-get install nano; grep –n display_errors)
  - Con nano entramos y si pulsamos ^_ vamos a la línea 503.
  - Reiniciar servidor
- Lo ideal es cambiarlo permanentemente en el servidor
- En producción habría que deshabilitarlos!!!.
- Probar un ejemplo con fallo

1 DEPURACIÓN DE ERRORES

- **ERRORES EN DESARROLLO – CAMBIO EN SERVIDOR**
- También lo podemos hacer desde el modo gráfico de Docker Desktop



2 VARIABLES Y CONSTANTES

PHP AVANZADO. VARIABLES DEL SISTEMA

•Dada su naturaleza de lenguaje de lado servidor, PHP es capaz de darnos acceso a toda una serie de variables (de lectura) que nos informan sobre nuestro servidor y sobre el cliente.

• Existen multitud de variables de este tipo, algunas realmente interesantes y con una aplicación directa para nuestro sitio web:

$_SERVER :Variables definidas por el servidor web ó directamente relacionadas con el entorno: donde el script se esta ejecutando.

$_GET : Variables proporcionadas al script por medio de HTTP GET.

$_POST: Variables proporcionadas al script por medio de HTTP POST.
$_COOKIE: Variables proporcionadas al script por medio de HTTP cookies.
$_FILES: Variables proporcionadas al script por medio de la subida de ficheros vía HTTP .
$_ENV: Variables proporcionadas al script por medio del entorno del SSOO.
$_REQUEST: Variables proporcionadas al script por medio de cualquier mecanismo de entrada del usuario y por lo tanto *no se puede confiar en ellas.*
$_SESSION: Variables registradas en la sesión del script.

Ver ejemplos en la carpeta: **ejemplo_variables_sistema.php**

3 PASO DE VARIABLES

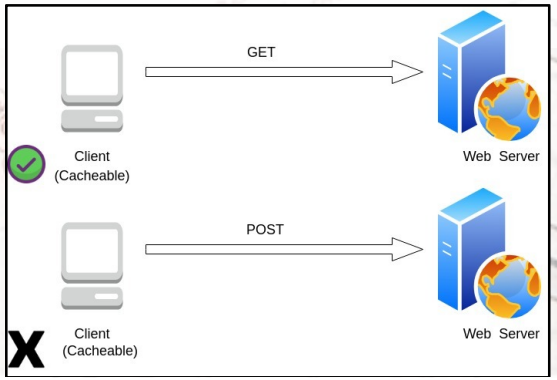
PHP AVANZADO. PASO DE VARIABLES (Ejemplos en paso_de_variables)

- Posiblemente la característica fundamental de un lenguaje de script de servidor es el paso de variables: es la comunicación entre páginas web.
- Si yo tengo un portal Web con un conjunto de páginas y no pueden transmitir datos entre ellas → pierden toda utilidad.
- Por ejemplo, cuando en el google pongo la palabra a buscar: "futbol" , la palabra se envía a la página que procesa la búsqueda , dicha página tendrá que **CAZAR** la palabra de alguna forma para poder consultar la base de datos. El paso de variable permite el envío y recepción de variables por la Web.
- Cuando envío datos casi siempre tendré un formulario (HTML o PHP) y una página obligatoriamente PHP.

3 PASO DE VARIABLES

PHP AVANZADO. PASO DE VARIABLES (Ejemplos en paso_de_variables)

- Hay tres formas básicas de enviar datos:
  - **Método POST:** transporta los datos incluyéndolos en las cabeceras HTML. No podemos verlos por la URL, utilizados para modificar recursos en servidor. Es más seguro que GET.
  - **Método GET:** transporta los datos por la URL. Permite una mayor cantidad de datos que POST pero es relativamente inseguro (pueden quedar en el historial del servidor o en el proxy). Utilizado sobre todo para hacer consultas en servidor
  - **Método REQUEST:** transporta los datos y los recoge independientemente del método que el usuario haya utilizado para su envío (formulario, directamente escribiéndolo en la URL). Por lo tanto es inseguro → no usar!



3 PASO DE VARIABLES

PHP AVANZADO. PASO DE VARIABLES. POST

- Tengo que indicar **method=POST** en el formulario
- No aparece ningún dato por la URL
- En la página PHP recogeré los datos mediante el array **$_POST**
- Ejemplo:

Página formu.html

```
<form name="formu" method="POST" action="recoge_por_post.php">
        <input type="text" name="nombre">
        <input type="submit">
</form>
```

Página **recoge_por_post.php**

```
<?php
$variable_nomb=$_POST["nombre"];
echo "La variable recogida tiene el valor $variable_nomb";
```

3 PASO DE VARIABLES

PHP AVANZADO. PASO DE VARIABLES. GET

- Tengo que indicar **method=GET** en el formulario
- Aparecen los datos por la URL (ej:si envío 3 variables)→
  http://**pagina.php?var1=**valor1**&var2=**valor2**&var3=**valor3
- En la página PHP recogeré los datos mediante el array **$_GET**
- Ejemplo:

Página formu.html

```
<form name="formu" method="GET" action="recoge_por_get.php">
        <input type="text" name="nombre">
        <input type="submit">
</form>
```

Página **recoge_por_get.php**

```
<?php
$variable_nomb=$_GET["nombre"];
echo "La variable recogida tiene el valor $variable_nomb";
```

3 PASO DE VARIABLES

PHP AVANZADO. PASO DE VARIABLES. REQUEST

•Request recoge los datos de cualquier sitio que vengan (formulario o escritura en URL). Por lo tanto es inseguro

Página **recoge_por_requet.php**

<?php

*$variable_nomb*=**$_REQUEST**[“**nombre**”];

echo “La variable recogida tiene el valor *$variable_nomb*”;

3 PASO DE VARIABLES

PHP AVANZADO. PASO DE VARIABLES. FOREACH

•POST, GET y REQUEST son arrays → por lo tanto puedo acceder a sus valores por medio de la función **print_r( );**

•Hay una forma fácil y rápida de sacar todos los valores enviados desde otra página.

•Esto además tiene una **ventaja**: me da igual cuantos valores se envíen (1 o 200) , los va a coger todos, pudiendo almacenarlos para ser procesados.

•Ejemplo:

```
$todas_las_variables=array();
foreach ($_POST as $clave=>$valor){
        $todas_las_variables[$clave]=$valor;

print_r($todas_las_variables); // Aquí ya tengo todos los valores
```

•Vamos a imaginar que se pasan 100 variables de una página a otra. Si esto mismo se tuviera que hacer con $_POST o con $_GET tendríamos que coger uno a uno cada uno de los datos enviados → 100 líneas de código. Aquí en 2 líneas los cogemos todos.

3 PASO DE VARIABLES

- Por lo tanto, todos los tipos de variables enviadas desde un formulario (atributo **name** de cada parte del formulario) son capturadas en otra página (atributo **value**) de dicha variable.
- Si por ejemplo envío la variable **nombre** y el **value** es *paco* si hago:
  - echo $_GET[“nombre”]; → el resultado será *paco*
- Hay dos tipos de variables especiales, que no envían un *value* , sino que envían si han sido pulsadas o no: radiobotones y checkbox.
  - Si un checkbox no ha sido marcado enviará **on**, en otro caso será **off**
  - Lo mismo pasa con los radiobotones, lo que pasa es que normalmente uno de ellos estará marcado (selected) por lo que si enviará su value.
- ¿Cómo sé si ha sido marcado en la página de recepción? No se comprueba si el valor es on u off, se utiliza la función **isset().**Ejemplo:

```
$condiciones=$_POST[“checkbox1”];
If (isset($condiciones)) echo “CHECKBOX HA SIDO MARCADO”
else “CHECKBOX NO HA SIDO MARCADO”
```

3 PASO DE VARIABLES

- Resumiendo, hay 3 formas de recoger las variables enviadas entre páginas Web!
- Los formularios no son la única forma de enviar variables, se pueden **enviar por medio de hipervínculos**. Ejemplo: suponed una tabla de una base de datos, se hace una consulta generando (con un bucle for) un hipervínculo por cada fila. El hipervínculo nos permitiría borrar dicha fila. Por lo tanto el formato sería el siguiente:
- BORRA FILA 1
- BORRA FILA 2
- BORRA FILA 3
- BORRA FILA 4

- Lógicamente, si estoy pasando datos por la URL, se tendrán que cazar con **$_GET**
- Se podrían enviar todas las variables que yo quisiera por medio de un hipervínculo: ej:
- BORRA FILAS ENTRE **a** y **b**

3 PASO DE VARIABLES

•En los ejemplos anteriores se envían datos a otras páginas, y en estas páginas se capturan los datos.

•También se puede enviar datos a la misma página para hacer las comprobaciones en la misma página. → ¿Como? Enviando datos a la misma página por el formulario o por un hipervínculo. →**ej_formulario_get_rellamada.php**

•Al entrar, captura los datos del formulario, pero ¿Cómo sabe la página si estoy entrando la primera vez o acabo de darle al botón? → Puedo utilizar **isset** con la variable del botón de envío (tengo que ponerle un name).

•Ej:

```
<?php
if (isset($_GET["boton_envio"])){
   //echo "BOTON DE ENVÍO MARCADO. CAPTURO TODOS LOS DATOS";
   $nom=$_GET["nombre"];
            if ($nom=="")
                     echo "<script language='javascript'>
                              alert('FALTA EL NOMBRE');
                     </script>";

  }?>
```