

Capítulo 3

El lenguaje PHP

1. ¿Cómo programar en PHP?

Regrese a su editor Visual Studio Code y haga clic en la pestaña **index.php**.

Vamos a escribir nuestro primer código en este archivo.

Todos los nombres de archivos PHP deben terminar con **.php** (por ejemplo: `miArchivo.php`). Evite los caracteres especiales en los nombres de sus archivos (`/`, `>`, `<`, `?`, `!`...) y los caracteres acentuados (`é`, `à`, `ü`...), así como los espacios.

Si da a su archivo un nombre diferente de **index.php**, deberá especificarlo en la barra de direcciones del navegador (veremos esto más adelante).

Comencemos escribiendo código en nuestro archivo **index.php**. Todo el código PHP se encuentra dentro de las mismas instrucciones:

```
<?php
... una línea de código en php
?>
```

Aunque su archivo termine en **.php**, es necesario especificar estas etiquetas dentro de él para escribir código PHP. ¿Por qué?

Porque su archivo PHP también puede contener texto libre o código HTML, que será interpretado tal cual por el navegador.

36 ————— Desarrollar aplicaciones web

con PHP y Symfony

Ejemplo

```
<h1>Título de la página en HTML</h1>
<?php
// mi código PHP
?>
```

La página que se muestre comenzará aquí con un título en negrita: Título de la página en HTML.

Así, un archivo PHP (con la extensión .php) puede contener código mixto con partes en HTML y partes en PHP.

```
Código HTML
<?php
Aquí dentro, código PHP
?>
Código HTML
<?php
Aquí dentro, código PHP
?>
```

También podría únicamente contener código PHP, como será nuestro caso con Symfony:

```
<?php
Aquí dentro, código PHP
?>
```

Cuando un archivo contiene únicamente código PHP, se puede prescindir de la etiqueta de cierre `?>` (y es mejor para evitar la visualización de cualquier carácter que quede al final en la página).

Ejemplo

```
<?php
Aquí dentro, código PHP
```

Los comentarios (líneas que no serán interpretadas por el servidor) se indican con `//` o con `/* ... */` para un bloque de líneas.

Se recomienda agregar comentarios en su código (no demasiados) para explicar lo que hace y facilitar la comprensión cuando usted o alguien más retome el desarrollo del código.

Ejemplo

```
<?php
// comentario en una línea
*/Ejemplo de
comentario en
varias líneas */
?>
```

Visual Studio Code le permite insertar automáticamente comentarios seleccionando la línea o líneas deseadas y pulsando [Ctrl] / (las mismas teclas para deseleccionar).

2. La base del lenguaje: el primer «Hello World!»

Vamos a escribir solo una línea de código para mostrar en el navegador:

```
Hello World!
```

Normalmente, se suele escribir texto en inglés, el idioma más utilizado en informática, pero también podría escribir «¡Hola, mundo!».

La instrucción PHP que permite mostrar algo en el navegador se llama `echo`.

■ Escriba este código en el archivo `index.php`:

```
<?php
echo 'Hello World!';
?>
```

Observamos que es necesario colocar el texto entre ' (comillas simples) para distinguirlo del código y que la instrucción termina con un punto y coma.

■ Observación

Todas las instrucciones de PHP terminan con un punto y coma. Si lo olvida, se mostrará un error en su navegador.

■ No olvide guardar su archivo: [Ctrl] S (cada vez que modifique un archivo, deberá guardar las modificaciones).

38 — Desarrollar aplicaciones web

con PHP y Symfony

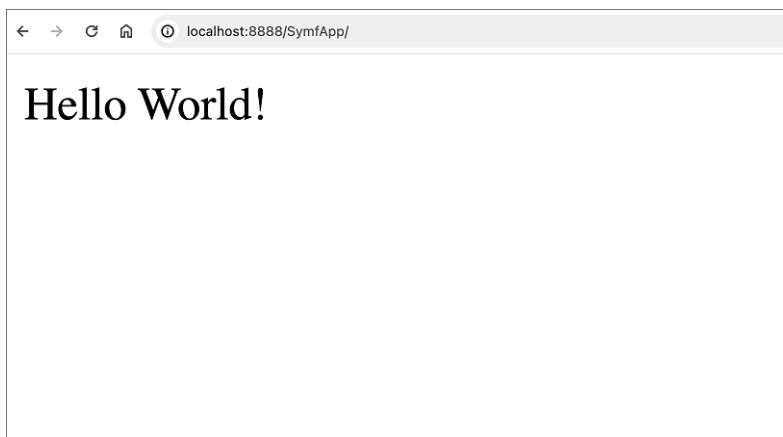
¿Cómo ver el resultado de su código en su navegador como lo vería un internauta?

Si su WAMP está encendido (**W** verde en la barra de tareas; de lo contrario, asegúrese de reiniciarlo): abra un navegador (se recomienda Firefox) y escriba la siguiente dirección en la barra de direcciones: localhost/SymfApp.

■ Observación

«localhost» es el nombre del servidor instalado en su propio ordenador.

El archivo index.php se abrirá automáticamente porque es el que se abre por defecto, y verá que aparece el texto:



Ahí lo tiene: acaba de escribir su primer código PHP.

- Vuelva a su archivo index.php en Visual Studio Code (que llamaremos Vscode en lo sucesivo) y modifique el título:

```
<?php
echo 'Hello everybody';
?>
```

- Guarde (teclas [Ctrl] **S**) y actualice su navegador. Verá que se ha realizado la modificación en el navegador.

Ahora, aprenderemos algunas instrucciones de PHP.

No se trata aquí de explorar el lenguaje PHP en todos sus detalles, sino de tener las herramientas necesarias para sentirse cómodo con el framework Symfony.

Las herramientas que vamos a ver son realmente indispensables, así que sumérjase en esta iniciación con cuidado. Practique creando sus propios ejemplos en el archivo **index.php**.

3. Las variables en PHP

Los primeros de estos instrumentos son **las variables**.

Una variable permite almacenar información para poder reutilizarla tantas veces como se desee en el código. También podemos realizar operaciones entre variables (sumarlas, multiplicarlas, etc.).

Todas las variables en PHP comienzan con \$. Así es como se identifican. El nombre de una variable debe seguir las mismas reglas que los nombres de archivos (sin caracteres especiales ni acentuados, ni espacios).

Ejemplo de variable: \$texto

Si desea utilizar nombres de variables con varias palabras, la convención es distinguir cada palabra, excepto la primera, con una mayúscula (no es obligatorio, pero es un estándar de escritura). Esta convención recibe el nombre de camelCase o dromedaryCase.

Ejemplo: \$miTexto

Puede crear tantas variables como desee en su código.

Todas las variables se eliminan al final de la ejecución del código.

Es posible asignar texto a una variable:

```
<?php
$miTexto='Hello World!';
?>
```

40 _____ Desarrollar aplicaciones web

con PHP y Symfony

¿Por qué es útil?

Cada vez que desee mostrar este texto en cualquier parte de su código, utilizará el nombre de la variable. Una de las ventajas es que, si desea cambiar el texto en todos los lugares donde lo ha insertado (¡imagínese que lo ha insertado 100 veces!), solo tiene que cambiar el texto en su variable y se actualizará automáticamente en todas partes.

Para mostrar el contenido de su variable, seguirá utilizando la instrucción `echo`, pero esta vez sin las comillas simples:

```
<?php
$miTexto='Hello World!';

echo $miTexto;

?>
```

Este código produce el mismo resultado que anteriormente (puede probarlo; no olvide guardar con [Ctrl] **S** cada vez que modifique su código).

También es posible insertar una variable en un texto que se va a mostrar. Para ello, se utilizan las comillas dobles (""):

```
<?php
$miNombre='Francisco';

echo "Hello $miNombre!";

?>
```

Este código mostrará `Hello Francisco!` en el navegador.

Otra forma de hacerlo es utilizar la concatenación para yuxtaponer varias cadenas de caracteres usando el operador . (punto):

```
<?php
    $miNombre='Francisco';

    echo "Hello ".$miNombre."!";

?>
```

Aquí, la cadena de caracteres "Hello" se concatena con la variable \$miNombre, que a su vez se concatena con la cadena de caracteres "!".

A las cadenas de caracteres también las llamamos **strings**. Este es el término que se usará a partir de ahora en el libro.

También es posible insertar números en una variable (para realizar operaciones, por ejemplo). Para distinguir los números de las cadenas de caracteres, se omiten las comillas.

Ejemplo: \$precio=20; \$iva=0.21;

He aquí cómo mostrar una cantidad de productos y un IVA con variables:

```
<?php
    $precio=20;
    $cantidad=10;
    $iva=0.21;

    $total=$precio*$cantidad*(1+$iva);

    echo "Precio Total (IVA incluido): $total euros";

?>
```

Observación

** es el operador de multiplicación (los operadores aritméticos son: +, -, *, /).*

Una vez actualizado, el resultado será: Precio Total (IVA incluido): 242 euros.

42 _____ Desarrollar aplicaciones web

con PHP y Symfony

Las variables son muy útiles, ya que permiten agrupar varios datos en todo el código.

4. Las declaraciones de tipo en PHP

Las variables se definen según su tipo. Llamamos TIPO DE VARIABLE a la naturaleza del valor que puede almacenar. ¿Es la variable de tipo numérico (por ejemplo, `$precio=20;`) o alfanumérico (por ejemplo, `$nombre='Francisco';`)?

Según el tipo de variable, su comportamiento y su visualización serán diferentes. Por ejemplo, puedo realizar operaciones en variables de tipo numérico, pero no en variables de tipo alfanumérico.

En PHP, existen diferentes tipos de variables:

Null: la variable solo puede tomar un valor, el valor null. Existe, pero no contiene nada. Se pueden declarar variables sin asignarles un valor. Basta con escribir: `$miVariable=null`. Se supone que a esta variable se le otorgará un valor más adelante, en el código.

Bool: estas variables solo pueden tomar dos valores: *true* o *false*. Veremos en la siguiente sección (Las estructuras de control) el interés de este tipo.

Int: son variables numéricas enteras. Solo pueden contener números enteros (sin decimales). Ejemplo: `$precio=10;`

Float: son variables numéricas de punto flotante (también conocidas como «floats», «doubles» o «números reales»). Estas variables se pueden especificar utilizando las siguientes sintaxis:

`$a = 1.234;`

`$b = 1.2e3;`

`$c = 7E-3;` // esta última notación es la notación científica (7E-3 significa 0.007).

String: estas variables son de tipo alfanumérico. Sus valores siempre están definidos entre comillas. Por ejemplo: `$ciudad='Madrid';` o `$pais="España"`

Veremos más adelante en el libro que hay otros tipos de variables, como arrays u objetos (Object), pero, por ahora, nos limitaremos a estos.

En PHP, no es necesario declarar el tipo de una variable. El tipo se define automáticamente cuando se asigna un valor a una variable.

Por ejemplo, si escribes: `$altura=1.85`, su variable será automáticamente de tipo float. Esto puede causar confusiones, ya que si luego, en su código, asigna a la variable otro valor con otro tipo, por ejemplo: `$altura='entre 175 y 190 cm'`, su variable cambiará automáticamente de tipo. En este caso, tomará el tipo STRING.

La visualización mediante la instrucción `ECHO` no indica el tipo de una variable. Si quiere mostrar el tipo de una variable, puede utilizar la función `gettype()`.

Ejemplo

```
<?php
$var = null;
echo gettype($var);
```

Este código mostrará el valor NULL.

También puede mostrar el tipo y el valor de una variable utilizando la función `var_dump()`. No es necesario usar la función `echo` en este caso.

Ejemplo

```
<?php
$precio = 101;
var_dump($precio);
```

Este código mostrará: `C:\wamp64\www\SymfonyApp\index.php:4:int 101`.

Nótese que también indica la ruta hacia la variable creada.

El hecho de que la declaración de los tipos de variables sea automática en PHP no es trivial. Veremos más adelante cómo asegurarnos del tipo de nuestras variables.