

UT2: Introducción al desarrollo de aplicaciones Web con PHP

PHP y HTML. Código Incrustado

- PHP es el lenguaje de programación para desarrollo web en el lado del servidor.
- Desde su aparición en 1994 ha tenido gran aceptación y se puede decir que es lenguaje más extendido para el desarrollo en el lado del servidor.
- Aunque no es la única opción, lo normal es que el intérprete de PHP sea un módulo del servidor web.

PHP y HTML. Código Incrustado

- El lenguaje PHP es flexible y permite programar pequeños scripts con rapidez.
- Comparado con lenguajes como Java, requiere escribir menos código y, en general, resulta menos engorroso.
- La sintaxis de los elementos básicos es bastante parecida a la de lenguajes muy extendidos como Java y C.
- Por estos motivos, es un lenguaje rápido de aprender para las personas con alguna experiencia en programación.

Creación de ficheros

- PHP se ejecuta en un fichero plano. Para ello debes crear un documento con la extensión .php y dentro tu HTML.

A screenshot of a code editor window with a dark background. The title bar at the top is labeled 'php' and has standard macOS window control buttons (green, yellow, red) on the right. The editor contains the following HTML code in a light pink font:

```
<html>
  <head></head>
  <body>
    <h1>Mi primer fichero en PHP</h1>
  </body>
</html>
```

A clipboard icon is visible in the top right corner of the editor area.

- Para añadir tus scripts de PHP debes usar `<?php ?>` o `<? ?>` (esta última debes activarla en la configuración).

Creación de ficheros

A screenshot of a code editor window with a dark background. The window title is 'php'. It contains the following code:

```
<html>
  <head></head>
  <body>
    <h1>Mi primer fichero en PHP</h1>
    <?php echo 'Y funciona perfecto!'; ?>
  </body>
</html>
```

- Toda sentencia, la instrucción, debe terminar con un ;. Independientemente de donde se abra o se cierre el script. Eres libre de añadir tabulaciones y saltos de línea para dejarlo más ordenado.

Levantar un servidor de PHP

- Si intentas abrir el archivo comprobarás que no ha pasado nada. Necesitamos procesar el archivo PHP para que se muestre el HTML final.
- La misión de PHP es generar HTML, el navegador no entiende PHP. Todo archivo de PHP debe ejecutarse con un software.
- Utilizaremos XAMPP, en donde aparecerá una carpeta con el nombre htdocs donde colocarás tus activos PHP.

Imprimir

- La palabra reservada ***echo*** sirve para imprimir por pantalla. Eso significa que cualquier texto que pongamos a continuación de echo aparecerá en el lugar donde se ha declarado el script.

A screenshot of a code editor window titled 'php'. The window has a dark background and a light gray border. In the top right corner, there are three colored window control buttons: green, yellow, and red. A clipboard icon is visible in the top right corner of the code area. The code is written in a syntax-highlighted format with the following lines:

```
<html>
  <head></head>
  <body>
    <h1>Mi primer fichero en PHP</h1>
    <?php
      echo '<p>Y funciona perfecto!</p>';
    ?>
  </body>
</html>
```

Imprimir

- Terminará siendo.

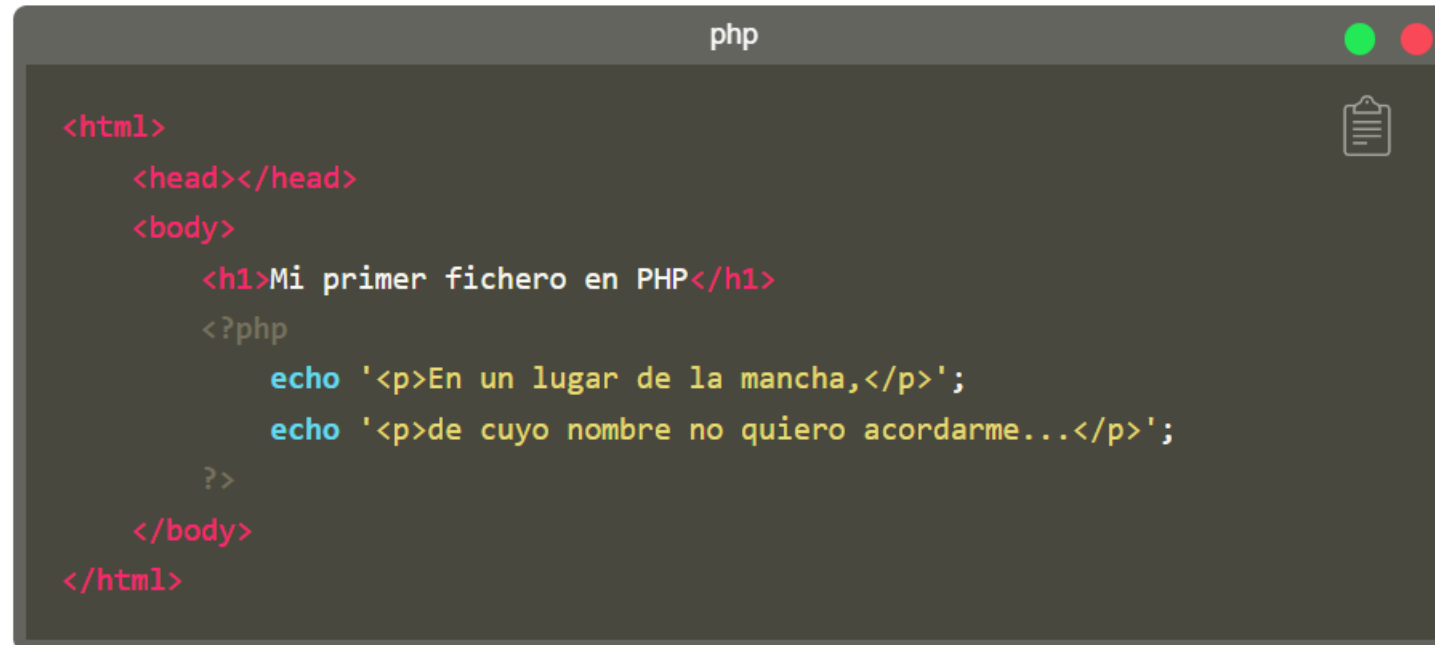
A screenshot of a code editor window with a dark gray background. The window has a title bar at the top with the text 'php' and three colored window control buttons (green, yellow, red) on the right. The code is written in a light gray font and is syntax-highlighted: opening and closing HTML tags are in red, and text content is in light gray. The code is as follows:

```
<html>
  <head></head>
  <body>
    <h1>Mi primer fichero en PHP</h1>
    <p>Y funciona perfecto!</p>
  </body>
</html>
```

- Si deseas realizar varias líneas tienes varias opciones.
- La más sencilla es utilizando varios echo.

Imprimir

- Terminará siendo.



```
<html>
  <head></head>
  <body>
    <h1>Mi primer fichero en PHP</h1>
    <?php
      echo '<p>En un lugar de la Mancha,</p>';
      echo '<p>de cuyo nombre no quiero acordarme...</p>';
    ?>
  </body>
</html>
```

Imprimir

- Otra es añadiendo un salto de línea representado como PHP_EOL.

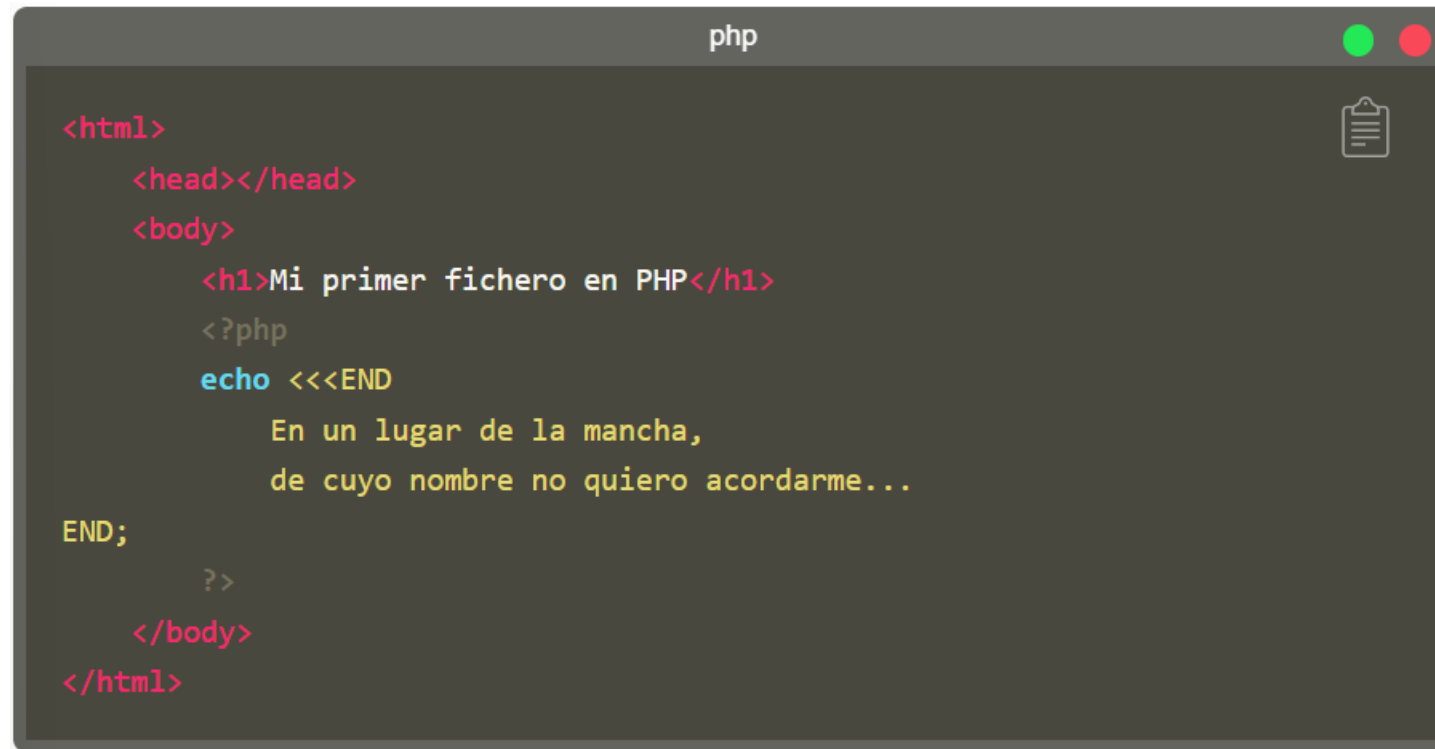


```
<html>
  <head></head>
  <body>
    <h1>Mi primer fichero en PHP</h1>
    <?php
      echo '<p>En un lugar de la mancha,</p>' . PHP_EOL . '<p>de cuyo n
ombre no quiero acordarme...</p>';
    ?>
  </body>
</html>
```

Imprimir

- En la red verás miles de ejemplos donde se usa `\n` para realizar saltos de línea. Si lo utilizas podrías encontrarte problemas de compatibilidad entre sistemas operativos ya que la forma de realizar un salto de línea varía en cada uno. `PHP_EOL` te hace la vida más fácil ya que selecciona automáticamente el correcto.
- Y la última es Usando `<<<`. Recuerda dejar `END;` al principio de la línea.

Imprimir



```
<html>
  <head></head>
  <body>
    <h1>Mi primer fichero en PHP</h1>
    <?php
      echo <<<END
        En un lugar de la Mancha,
        de cuyo nombre no quiero acordarme...
      END;
    ?>
  </body>
</html>
```

Concatenar

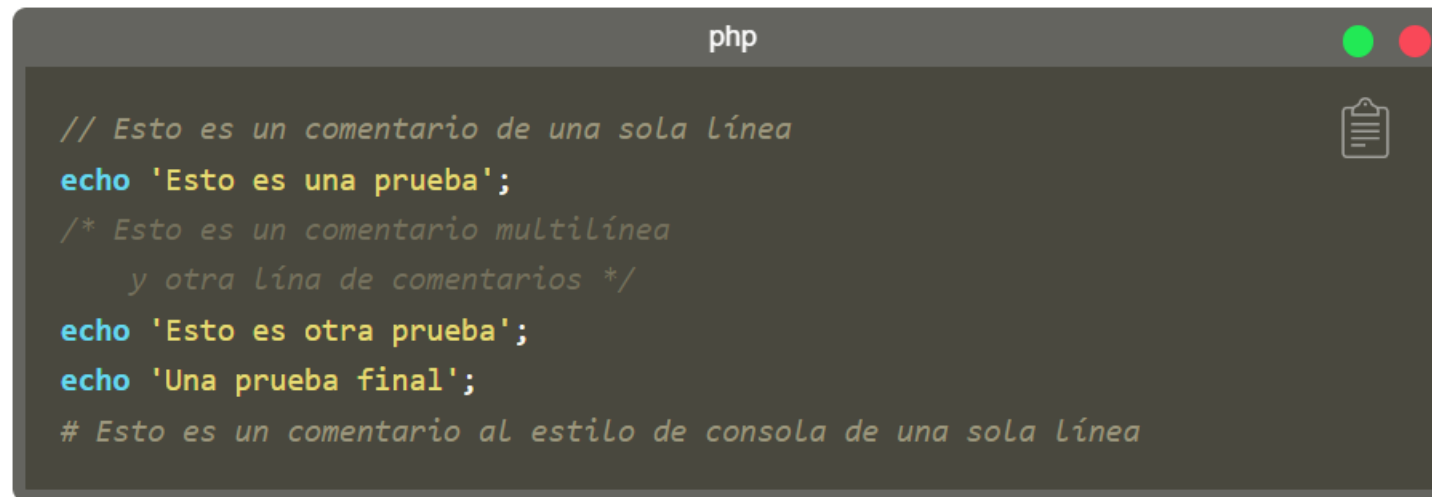
- Debes utilizar un punto entre ambos textos. Los espacios son opcionales, pero te ayudará a visualizar con más facilidad.



```
<html>
  <head></head>
  <body>
    <h1>Dos Beatles</h1>
    <?php
      echo 'John Lennon' . ' y ' . 'Paul McCartney';
    ?>
  </body>
</html>
```

Comentarios

- Existe dos tipos de comentarios: línea y multilinea.

A screenshot of a code editor window titled 'php'. The window has a dark background and a light gray border. In the top right corner, there are three colored circles (green, yellow, red) and a clipboard icon. The code inside the window is as follows:

```
// Esto es un comentario de una sola línea
echo 'Esto es una prueba';
/* Esto es un comentario multilinea
   y otra línea de comentarios */
echo 'Esto es otra prueba';
echo 'Una prueba final';
# Esto es un comentario al estilo de consola de una sola línea
```

- Los de una línea empiezan con // o #, seguido de un espacio.

Comentarios

- Mientras que los multilinea empiezan con `/*` y terminan con `*/`.
- Si quieres dejar secciones bien delimitadas y ordenadas J. Prettyman te propone la siguiente estructura.

```
php
//=====
// CATEGORIA EN MAYUSCULA
//=====

//-----
// Sub-Categoria En Minúscula
//-----

/* Título con Letras Capital */

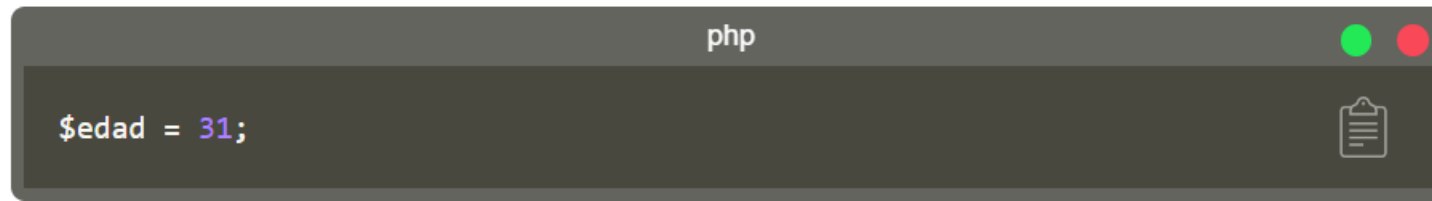
# Opción 1
# Opción 2
# Opción 3

/*
 * Larga explicación de alguna
 * funcionalidad o bloque de código
 * que lo merezca.
 */

// Anotación
```

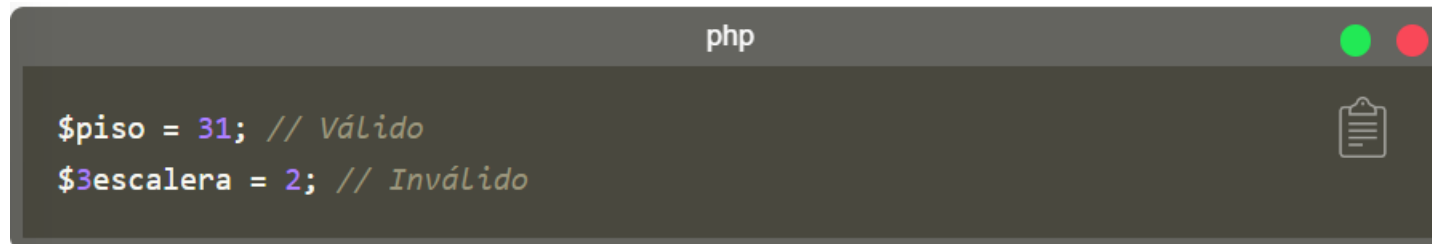
Variables

- Para declarar una variable debe tener la siguiente estructura.




A screenshot of a code editor window titled 'php'. The window has a dark background and a light gray border. In the top right corner, there are three colored circles (green, yellow, red) and a clipboard icon. The code inside the window is: `$edad = 31;`. The variable name '\$edad' is in white, the equals sign is in white, and the value '31' is in blue.

- \$edad es el nombre de la variable. Puede empezar por cualquier carácter salvo un número.



A screenshot of a code editor window titled 'php'. The window has a dark background and a light gray border. In the top right corner, there are three colored circles (green, yellow, red) and a clipboard icon. The code inside the window is: `$piso = 31; // Válido` and `$3escalera = 2; // Inválido`. The variable names '\$piso' and '\$3escalera' are in white, the equals signs are in white, and the values '31' and '2' are in blue. The comments '// Válido' and '// Inválido' are in a lighter gray color.

Variables

- El símbolo = es equivalente a una flecha que apunta de derecha a izquierda () , no confundas con el significado real del símbolo: no es un igual. El valor de la derecha es guardado en la izquierda. O más técnicamente: el valor se guarda en un espacio de memoria con el nombre de la izquierda. De este modo lo podrás reutilizar.
- El valor pueden ser varios tipos.

Variables

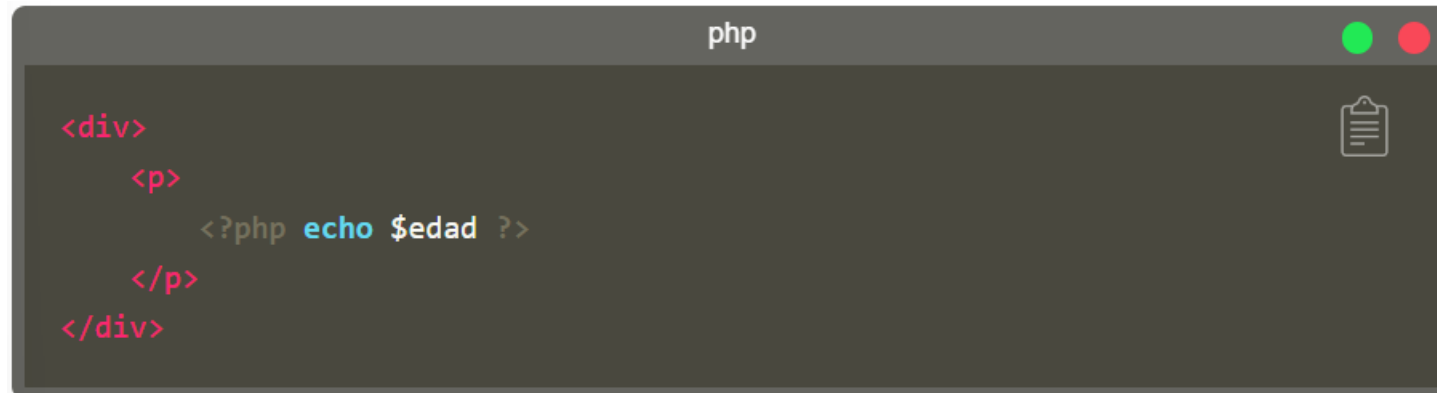
```
php
$nombre = 'Manolo'; // Texto. Puede ser con comillas simples o dobles (String)
$edad = 31; // Enteros (Integer)
$altura = 1.72; // Decimales, usando el punto en lugar de la coma (Float)
$mayorEdad = True; // Verdad o mentira (Boolean)
```

- Si quieres imprimirlo solo debes hacer uso de echo.

```
php
echo $edad;
```

Variables

- En caso de estar dentro de un contexto HTML, deberás abrir y cerrar PHP.

A screenshot of a code editor window with a dark background. The window title is 'php'. It contains the following code:

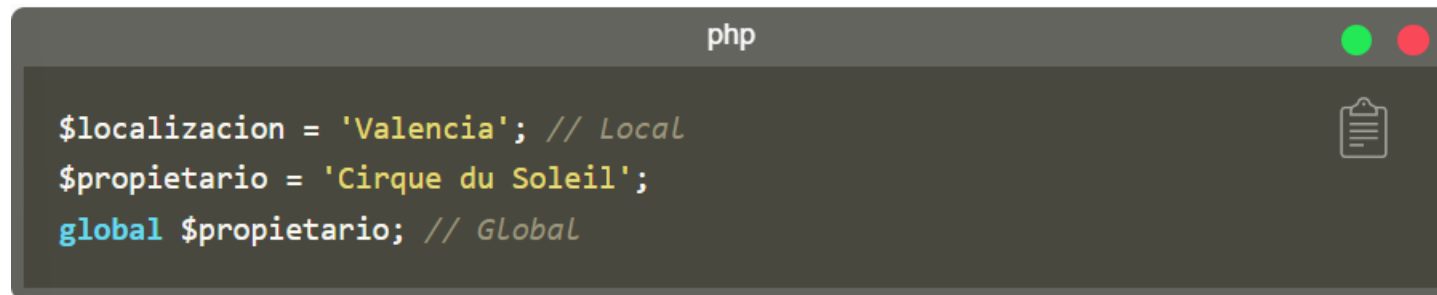
```
<div>
  <p>
    <?php echo $edad ?>
  </p>
</div>
```

 The code is color-coded: HTML tags are pink, the PHP opening tag is light blue, and the variable name is light green. A clipboard icon is visible in the top right corner of the editor area.

- Si solo es una línea puedes ahorrarte el punto y coma al final (;). Incluso existe un atajo para evitar echo.

Variables

- Visibilidad (Ámbito de variables)
 - Una variable por defecto tiene un alcance local. No puede usarse en otro script (o página). Si quieres que sea accesible por cualquier documento, debes usar la palabra reservada global.

A screenshot of a code editor window titled 'php'. The window has a dark background and a light gray border. It contains three lines of PHP code. The first line is '\$localizacion = 'Valencia'; // Local' with 'Valencia' in yellow and 'Local' in gray. The second line is '\$propietario = 'Cirque du Soleil';' with 'Cirque du Soleil' in yellow. The third line is 'global \$propietario; // Global' with 'global' in blue and 'Global' in gray. A clipboard icon is visible in the top right corner of the code area.

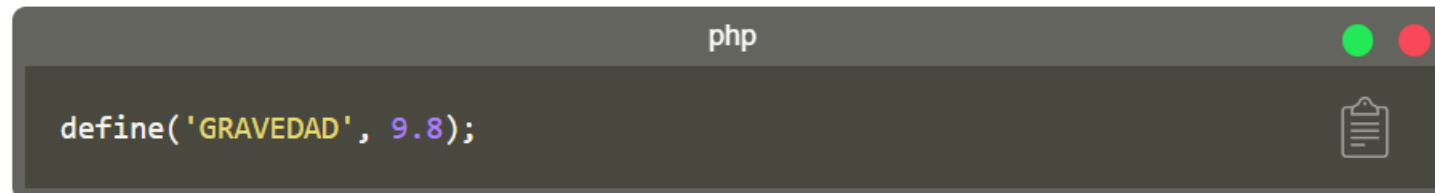
```
php

$localizacion = 'Valencia'; // Local
$propietario = 'Cirque du Soleil';
global $propietario; // Global
```

Variables

- Constantes

- En ocasiones es necesario indicar una variable va a ser inmutable. Ello ocurre porque hay cosas que nunca cambian ni quieres que cambien: número PI, velocidad de la luz, fuerza de la gravedad, días de la semana...
- Para crearla usaremos la función `define()`.

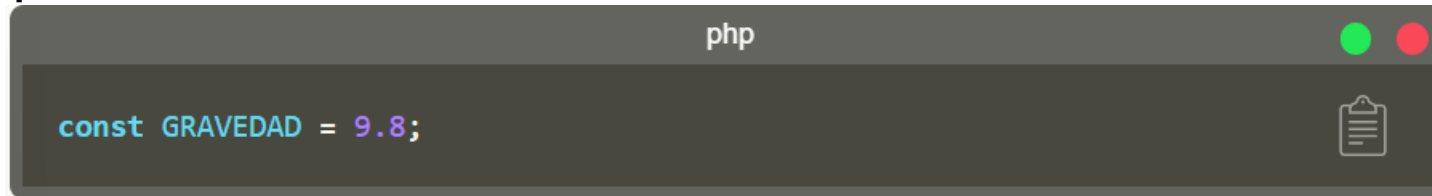
A screenshot of a code editor window titled 'php'. The window has a dark background and a light gray border. The code 'define('GRAVEDAD', 9.8);' is displayed in a monospaced font. The string 'GRAVEDAD' is highlighted in yellow, and the number '9.8' is highlighted in purple. There are three colored window control buttons (green, yellow, red) in the top right corner, and a clipboard icon in the bottom right corner.

- En cambio, para utilizarla solo usaremos su nombre sin necesidad de usar el prefijo \$.

Variables

- Constantes

- Otra sintaxis que nos ofrece PHP es usando la palabra const, como en JavaScript.

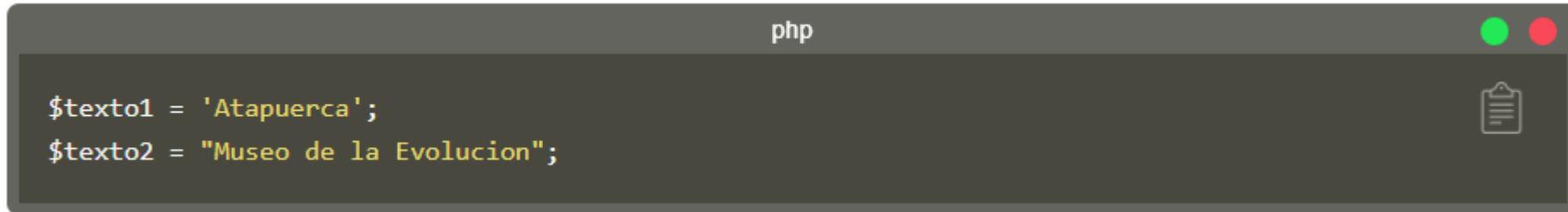
A screenshot of a code editor window titled 'php'. The window has a dark background and a light gray border. Inside, the code 'const GRAVEDAD = 9.8;' is displayed in a light blue font. To the right of the code, there is a small icon of a clipboard with a checkmark. The window also features standard macOS-style window controls (red, yellow, and green buttons) in the top right corner.

```
const GRAVEDAD = 9.8;
```

- Ambas se comportan igual.
 - No intentes concatenar constantes con comillas dobles (""), en PHP no funcionarán, usa punto (.) en su lugar.

Variables

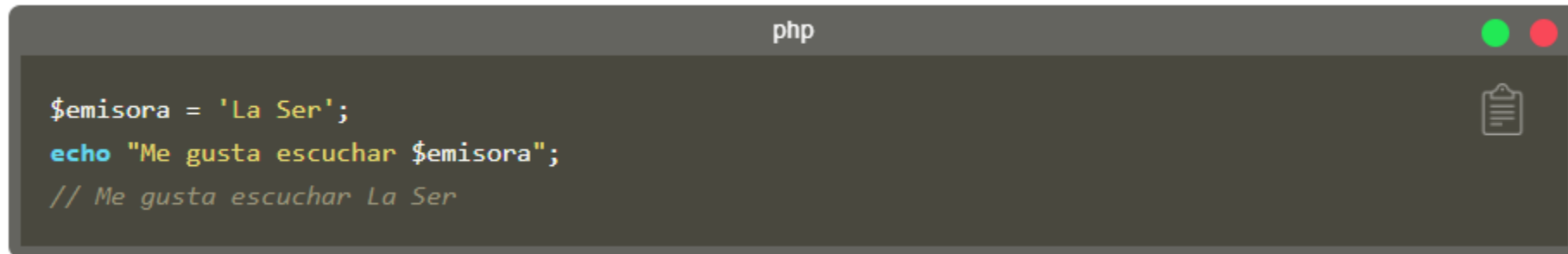
- Concatenar
 - No es lo mismo usar comillas simples o dobles.



```
php

$texto1 = 'Atapuerca';
$texto2 = "Museo de la Evolucion";
```

- Tal vez en apariencia funcionen igual. Pero cuando se utilizan las comillas dobles se le indica que dentro puede existir una variable.

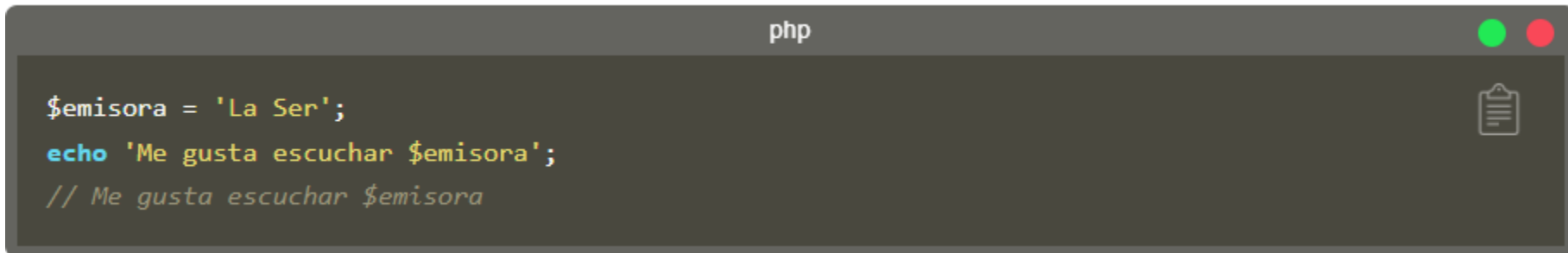


```
php

$emisor = 'La Ser';
echo "Me gusta escuchar $emisor";
// Me gusta escuchar La Ser
```

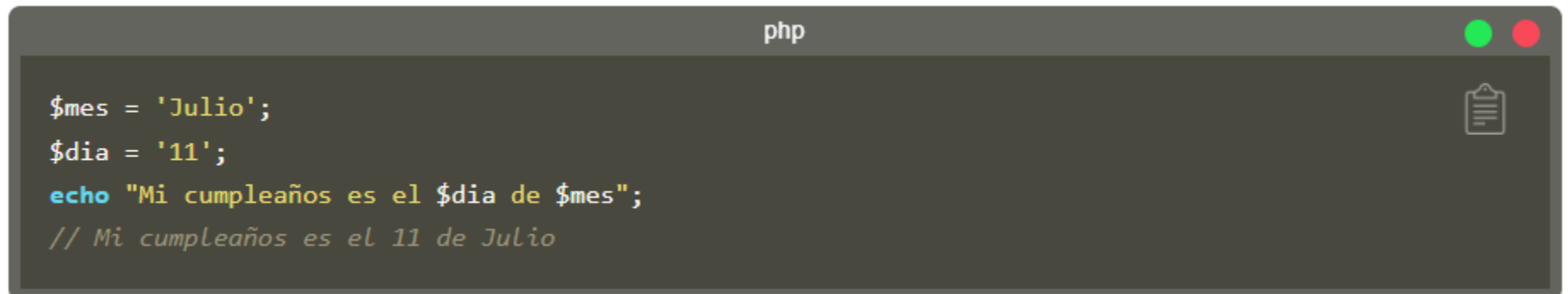
Variables

- Concatenar
 - Si hubiera utilizado comillas simples lo hubiera interpretado tal cual estaba escrito.



```
$emisora = 'La Ser';  
echo 'Me gusta escuchar $emisora';  
// Me gusta escuchar $emisora
```

- Esta técnica es una forma sencilla de concatenar variables sin usar un punto.



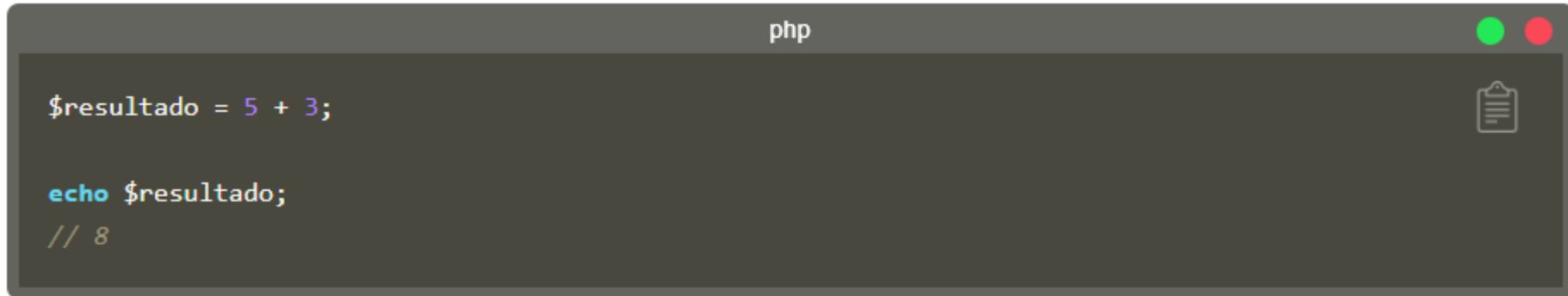
```
$mes = 'Julio';  
$dia = '11';  
echo "Mi cumpleaños es el $dia de $mes";  
// Mi cumpleaños es el 11 de Julio
```


Variables

- Concatenar
 - Acostúmbrate a usar siempre comillas simples salvo que quieras concatenar una variable. Os ayudaréis mutuamente: PHP trabajará menos y tu tendrás páginas que cargarán más rápido.

Variables

- Operaciones aritméticas
 - Para realizar operaciones matemáticas se utiliza los mismos símbolos que estamos acostumbrados (salvo el símbolo igual por las razones mencionadas).

A screenshot of a code editor window titled 'php'. The window has a dark background and a light gray border. It contains two lines of PHP code: '\$resultado = 5 + 3;' and 'echo \$resultado;'. Below the second line, there is a comment '// 8'. The code is color-coded: '\$resultado' is blue, '=' is black, '5' is purple, '+' is black, '3' is purple, ';' is black, 'echo' is blue, '\$resultado' is blue, ';' is black, and '// 8' is yellow. There is a clipboard icon in the top right corner of the editor area.

```
$resultado = 5 + 3;  
  
echo $resultado;  
// 8
```

- Los valores pueden estar en tantas variables como sean necesarios.

Variables

- Operaciones aritméticas

```
php

$num1 = 8;
$num2 = 2;
$resultado = $num1 + $num2;

echo $resultado;
// 10
```

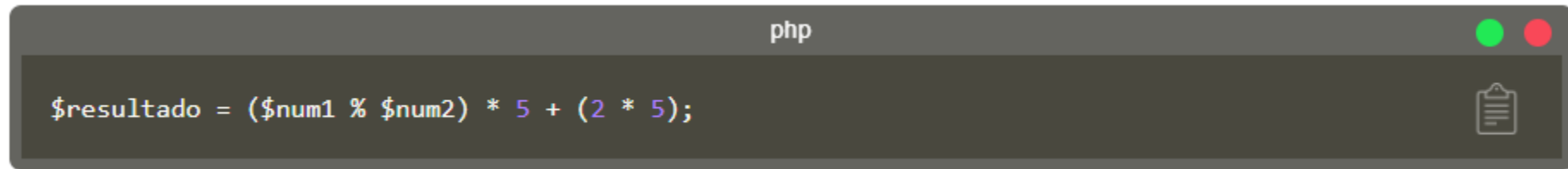
- Otras operaciones disponibles.

```
php

$resultado = $num1 + $num2; # Sumar
$resultado = $num1 - $num2; # Restar
$resultado = $num1 / $num2; # Dividir
$resultado = $num1 * $num2; # Multiplicar
$resultado = $num1 % $num2; # Resto
$resultado = $num1 ** $num2; # Potencia (Elevado a...)
```

Variables

- Operaciones aritméticas
 - Para realizar operaciones complejas te puedes apoyar en los paréntesis para indicar el orden de las operaciones.

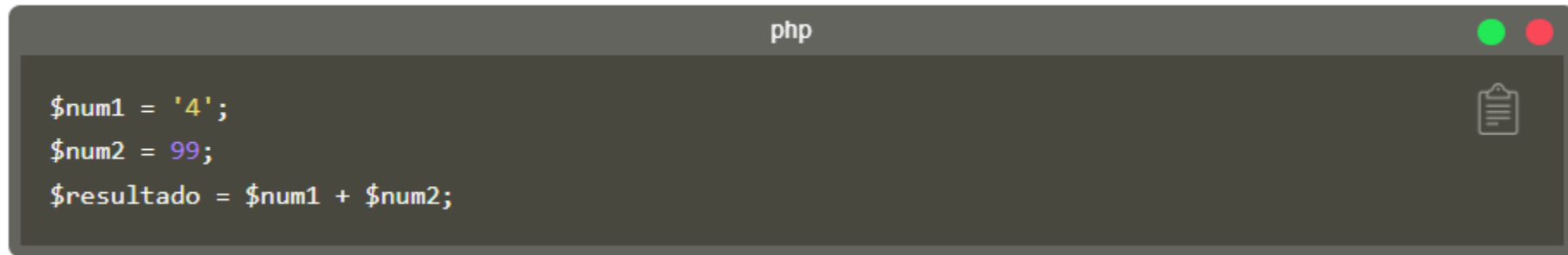


```
$resultado = ($num1 % $num2) * 5 + (2 * 5);
```

- ¿Todo claro? Vamos a jugar a un juego: observar el siguiente código.

Variables

- Operaciones aritméticas

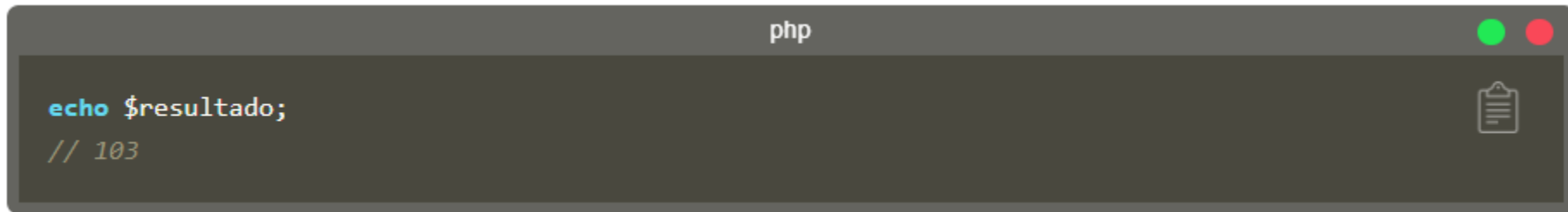


```
$num1 = '4';  
$num2 = 99;  
$resultado = $num1 + $num2;
```

- ¿Cuánto vale \$resultado? ¿499 o 103?

Variables

- Operaciones aritméticas



```
php
echo $resultado;
// 103
```

- ¿Qué ha pasado? PHP ignora que una de las variables sea un String (tipo texto) a la hora de realizar operaciones.
- Este efecto se llama tipado débil o tipado dinámico. Debería fallar ya que es imposible sumar un texto con un número, pero PHP te lo permite. A pesar de que parezca una fortaleza es una debilidad del lenguaje. ¡No ocurre en otros sitios!

Variables

- Actividad 1 - Calculando
 - Realiza una sencilla calculadora.
 - Pon en una variable un número.
 - Pon en otra variable un número.
 - Suma y muestra el resultado.
 - Pro:
 - Realiza otras operaciones (Restar, dividir...).

Variables

- Actividad 2 - Meeting
 - Realiza los siguientes pasos:
 - Guarda en una variable el nombre de una chica.
 - Guarda en otra variable el nombre de un chico.
 - Muestra la frase > “A {chica} le gusta {chico}.”
 - Por ejemplo: A Sonia le gusta Roberto.
 - Pro:
 - Guarda en dos variables un nombre y un año de nacimiento. Muestra la siguiente frase calculando la edad: “Me llamo {nombre} y nací el {año de nacimiento}. Por lo tanto, tengo 23.”

Variables

- Actividad 3 - Impuestos
 - Realiza los siguientes pasos:
 - Guarda un número.
 - Calcula el IVA de esa cifra.
 - Muestra la misma cifra con IVA añadido.
 - Pista: Para calcular el IVA debes aplicar la siguiente formula $\text{precio} * 1.21$.
 - Pro:
 - Muestra además la cifra sin IVA.
 - ¡Se creativo! Formatea el resultado de forma atractiva.

Arrays

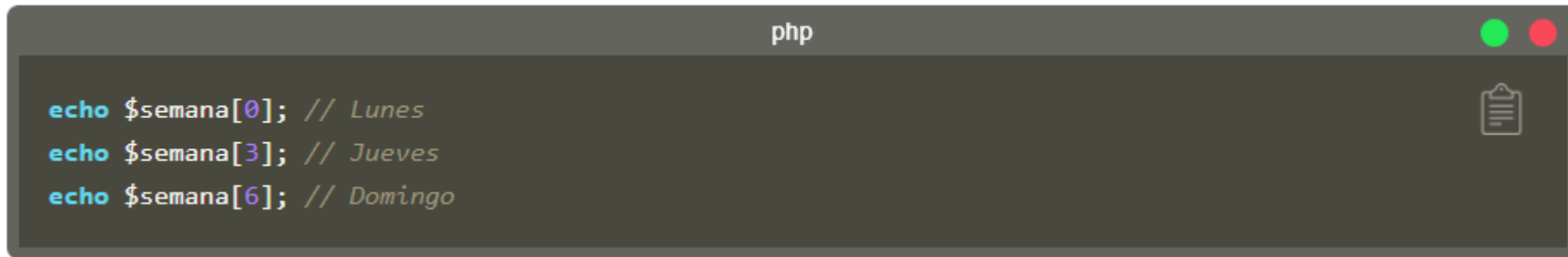
- Una dimensión
 - No es posible almacenar más de un valor en una sola variable, a no ser que utilicemos un array.
- Un array es un mapa ordenado donde los datos tendrán una clave (key) pero muchos valores (values). Por ejemplo, podríamos guardar los días de la semana bajo el mismo nombre de variable.

A screenshot of a code editor window titled 'php'. The window has a dark background and a light-colored text. It contains a PHP array definition for the days of the week. The array is named '\$semana' and contains seven elements: 'Lunes', 'Martes', 'Miércoles', 'Jueves', 'Viernes', 'Sábado', and 'Domingo'. The code is as follows:

```
$semana = [  
    'Lunes',  
    'Martes',  
    'Miércoles',  
    'Jueves',  
    'Viernes',  
    'Sábado',  
    'Domingo'  
];
```

Arrays

- Una dimensión
 - Los valores están dentro de corchetes ([]) separados por comas. De esta forma se asigna automáticamente un marcador que empieza por cero. Si quiero leer por separado cada elemento debo asignar el marcador envuelto en corchetes.

A screenshot of a terminal window titled 'php'. The window has a dark background and a light gray border. Inside, there are three lines of PHP code: `echo $semana[0]; // Lunes`, `echo $semana[3]; // Jueves`, and `echo $semana[6]; // Domingo`. The code is color-coded: 'echo' is blue, '\$semana' is green, and the indices and comments are brown. On the right side of the terminal window, there are three colored circles (green, yellow, red) and a clipboard icon.

```
php

echo $semana[0]; // Lunes
echo $semana[3]; // Jueves
echo $semana[6]; // Domingo
```

- Si quiero ver todo el contenido puedo usar la función nativa de PHP `var_dump`. Nos dirá la longitud del array, la posición de cada elemento, su tipo, longitud de cada valor y el propio valor que almacena.

Arrays

- Una dimensión
 - `var_dump` no necesita añadir un `echo` delante.
 - Usa `var_dump` en lugar de `print_r`, proporciona más información.

```
php

var_dump($semana);

/*
array(7) {
    [0] =>
        string(5) "Lunes"
    [1] =>
        string(6) "Martes"
    [2] =>
        string(10) "Miércoles"
    [3] =>
        string(6) "Jueves"
    [4] =>
        string(7) "Viernes"
    [5] =>
        string(7) "Sábado"
    [6] =>
        string(7) "Domingo"
}
*/
```

Arrays

- Una dimensión
 - Dentro de un array puede existir cualquier tipo, al igual que una variable.

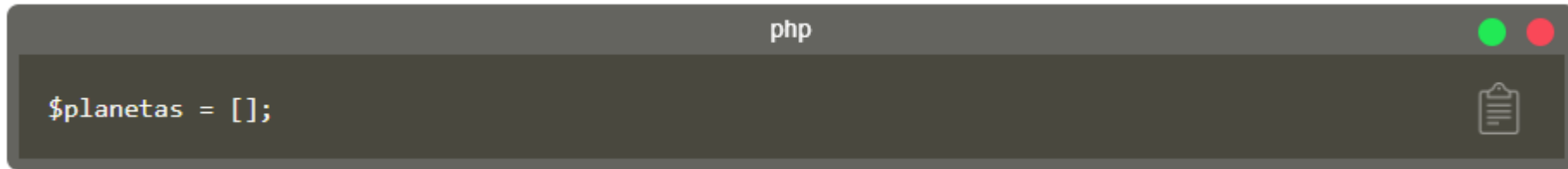


```
$corteIngles = [  
    'Azul',  
    39,  
    True,  
    11  
];
```

Arrays

- Crear

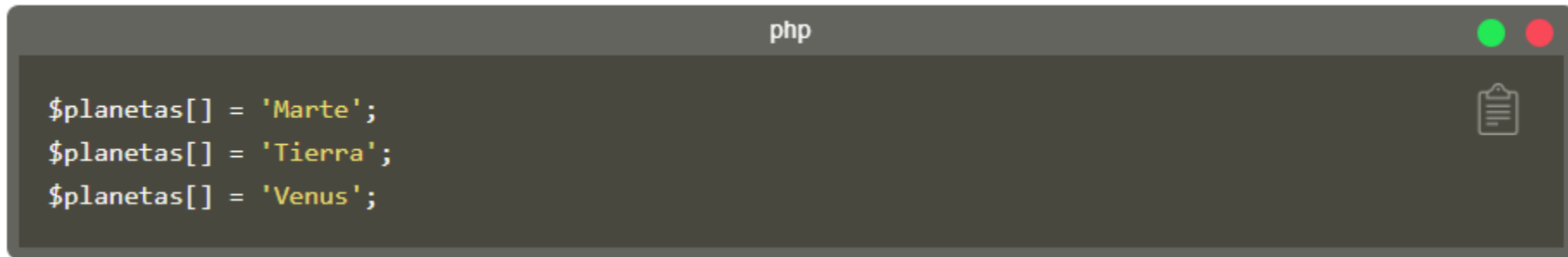
- Para declarar un array vacío solamente debemos crear una variable donde asignemos unos corchetes.

A screenshot of a code editor window with a dark gray background. The title bar at the top is dark gray and contains the text 'php' in white. On the right side of the title bar are three colored window control buttons: green, yellow, and red. The main area of the editor is dark gray and contains the PHP code '\$planetas = [];' in a light gray font. On the right side of the editor area, there is a small icon of a clipboard with a document.

- También lo puedes crear con su función: `$planetas = array();`.

Arrays

- Añadir
 - Para ir añadiendo elementos.



```
$planetas[] = 'Marte';  
$planetas[] = 'Tierra';  
$planetas[] = 'Venus';
```

- Y si queremos utilizar un método más orientado a la programación funcional, podemos usar `array_merge` para crear un nuevo array.

Arrays

- Añadir

```
php
// Array de partida
$planetas = ['Marte', 'Tierra', 'Venus'];

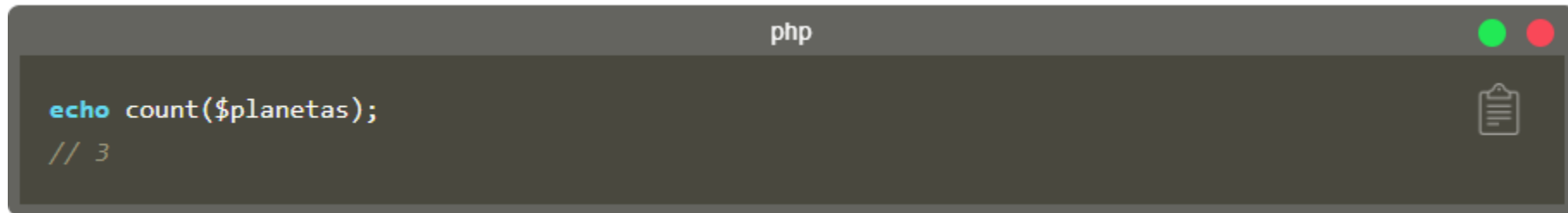
// Añadimos 'Mercurio'
$nuevosPlanetas = array_merge($planetas, ['Mercurio']);

// Vemos el resultado
var_dump($nuevosPlanetas);

/*
array(4) {
    [0]=>
        string(5) "Marte"
    [1]=>
        string(6) "Tierra"
    [2]=>
        string(5) "Venus"
    [3]=>
        string(8) "Mercurio"
}
*/
```


Arrays

- Añadir
 - Una utilidad muy práctica para saber la longitud de un array es usar la función `count()`.



```
php
echo count($planetas);
// 3
```

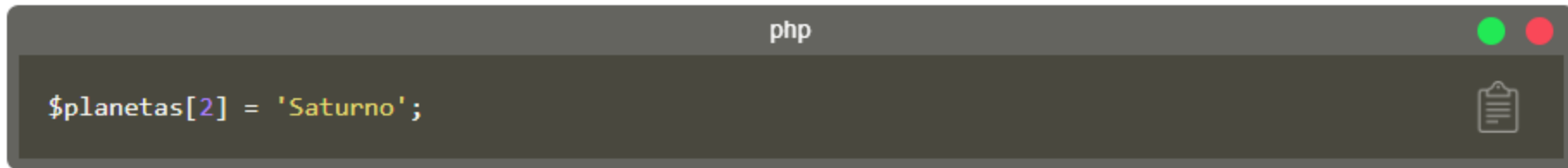
Arrays

- Actividad 4 – Nos vamos de viaje
 - Realiza los siguientes pasos:
 - Guarda en un array los nombres de unos amigos.
 - Imprime la siguiente frase: “{amigo 1} se va de viaje”.
 - Crea otro array con el nombre de varias ciudades.
 - Imprime la siguiente frase: “{amigo 2} se va de viaje a {ciudad 1}”
 - Pro:
 - Imprime aleatoriamente el nombre de un amigo.
 - *Pista:* shuffle(\$amigos).
 - Aleatoriamente busca dos nombres y una ciudad para generar la siguiente frase: “{amigo aleatorio} se va de viaje con {amigo aleatorio} a la bonita ciudad de {ciudad aleatoria}.

Arrays

- Modificar

- Para cambiar un valor hay que indicar la posición y el nuevo valor a introducir.

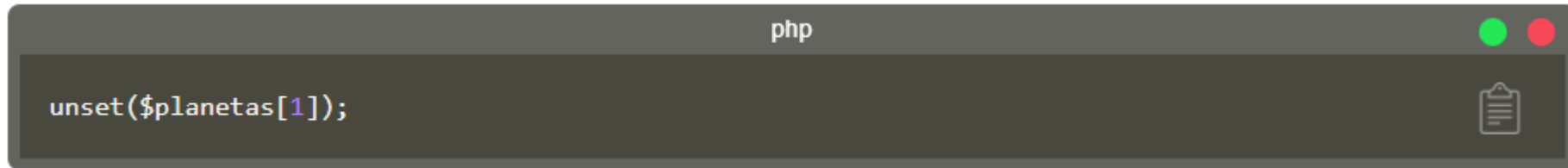
A screenshot of a code editor window with a dark theme. The title bar at the top says 'php' and has standard macOS window control buttons (red, yellow, green) on the right. The main area contains the PHP code `$planetas[2] = 'Saturno';` in a light-colored font. A small clipboard icon is visible in the bottom right corner of the editor area.

- Borrar

- Eliminar un elemento es un poco más marciano, debes usar una función nativa llamada unset.
 - Supongamos que quiero destruir la Tierra antes de que lo haga el hombre. Es el 2º elemento, cuya posición es la 1.

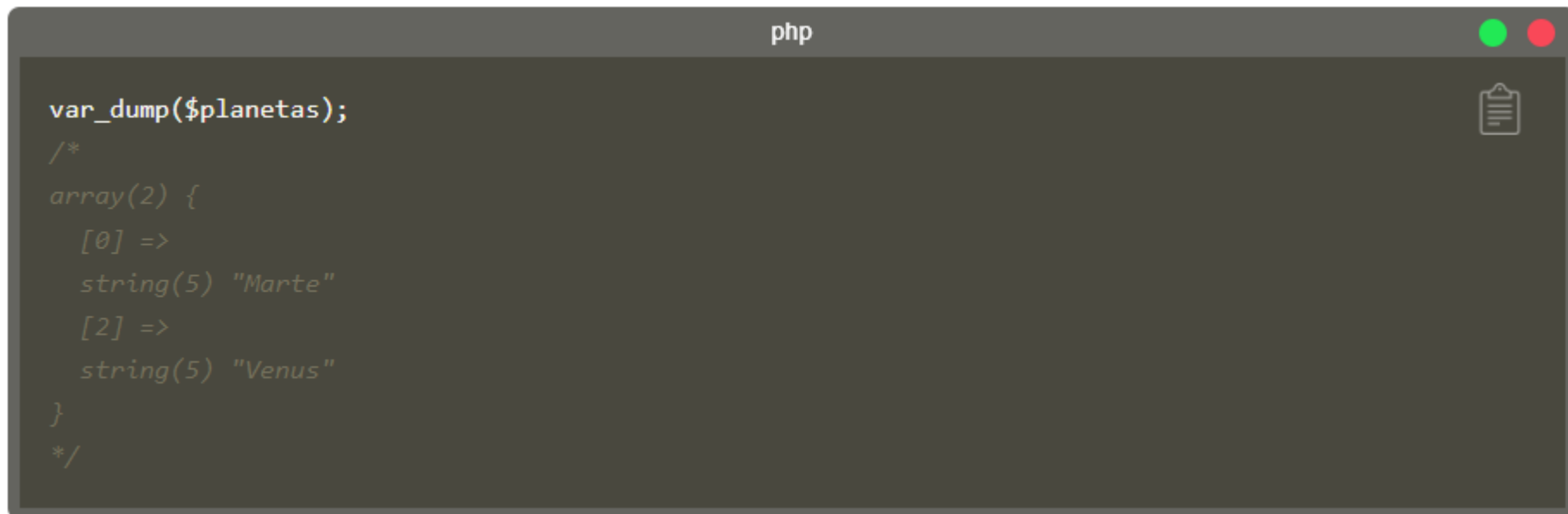
Arrays

- Borrar



```
unset($planetas[1]);
```

- Veamos que ha pasado.



```
var_dump($planetas);  
/*  
array(2) {  
    [0] =>  
        string(5) "Marte"  
    [2] =>  
        string(5) "Venus"  
}  
*/
```

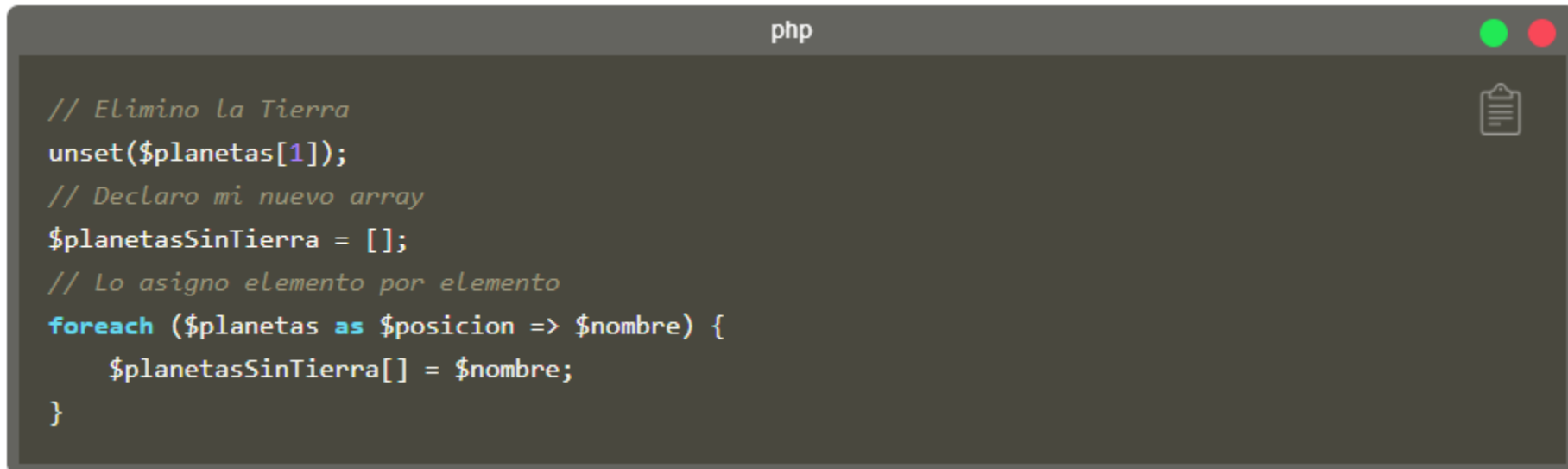
Arrays

- Borrar
 - Se ha ido... pero ¡me ha desordenado las posiciones! Intentamos hacer un Ctrl+Z añadiendo la Tierra de nuevo.

```
php
// Añado
$planetas[] = 'Tierra';
// Muestro todo
var_dump($planetas);
/*
array(2) {
    [0] =>
        string(5) "Marte"
    [2] =>
        string(5) "Venus"
    [3] =>
        string(6) "Tierra"
}
*/
```

Arrays

- Borrar
 - Nuestro array pierde la posición que borramos para siempre.
 - En realidad, no es un problema porque la gran mayoría de las ocasiones lo recorreremos con un bucle (loop) e ignoraremos las posiciones, pero debes ser consciente de cómo funciona para no encontrarte sorpresas.



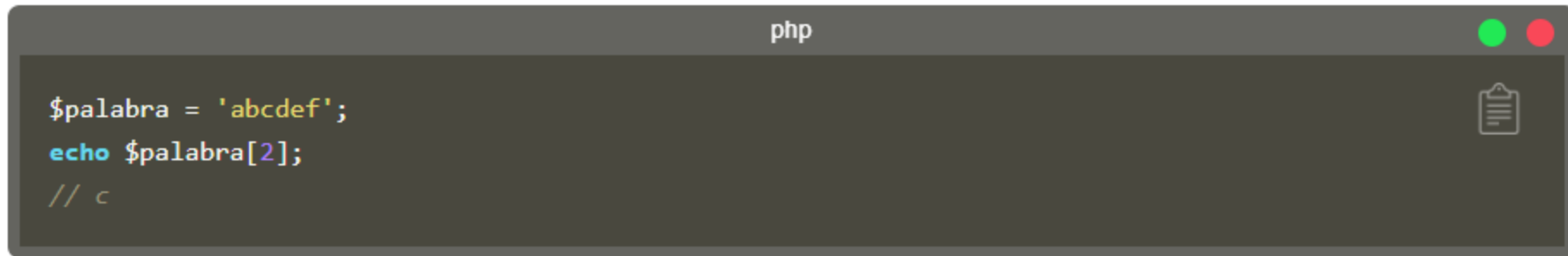
```
// Elimino la Tierra
unset($planetas[1]);
// Declaro mi nuevo array
$planetasSinTierra = [];
// Lo asigno elemento por elemento
foreach ($planetas as $posicion => $nombre) {
    $planetasSinTierra[] = $nombre;
}
```

Arrays

- Actividad 5 – Agenda
 - Realiza los siguientes pasos:
 - Crea un array con el nombre agenda.
 - Añade 2 citas (frases): “Dentista a las 12h” y otra que tú quieras.
 - Imprime con var_dump.
 - Te ha surgido un problema: Modifica la cita del Dentista a las 16h.
 - Imprime con var_dump.
 - Al final se te ha liado el día: Borra la cita con el Dentista
 - Imprime con var_dump.
 - Pro:
 - En lugar del var_dump, crea una lista desordenada ().
 - ¡Pista! join().

Arrays

- Jugando con Strings
 - Observa y dime que ocurre.



```
$palabra = 'abcdef';  
echo $palabra[2];  
// c
```


Arrays

- Jugando con Strings
 - ¿Qué ha pasado? Pues que los string pueden ser manipulados igual que un array.



```
$palabra = 'abcdef';  
$palabra[2] = 'Z';  
echo $palabra;  
// abZdef
```

- Un string se comporta como un array porque en el fondo no existen las palabras en la programación, sino conjuntos de caracteres. Dicho de otra manera: un string es un array de muchas letras.

Arrays

- Convertir un String en un Array
 - En algún momento tendrás la necesidad de pasar un texto a un array por medio de algún separador. Por ejemplo, transformar una frase a array dividido por espacios. La clave está en usar `preg_split`.

```
php

$frase = 'En un lugar de la mancha';
$arrayDeFrase = preg_split('/[\s,]+/', $frase);
echo $arrayDeFrase[2];
// "lugar"
var_dump($arrayDeFrase);
/*
array(6) {
    [0] =>
        string(2) "En"
    [1] =>
        string(2) "un"
    [2] =>
        string(5) "lugar"
    [3] =>
        string(2) "de"
    [4] =>
        string(2) "la"
    [5] =>
        string(6) "mancha"
}
*/
```

Arrays

- Actividad 6 – Concursos de micro relatos
 - Se ha convocado un concurso de micro relatos sobre sillones incómodos. El límite de palabras para ser enviadas son de 10.
 - Crea una variable con el micro relato.
 - Muestra el número de palabras usando `preg_split` y `count`.

Arrays

- Diccionario

- Las claves (key) pueden ser definidas por nosotros. A esto se le denomina diccionario.



A screenshot of a code editor window titled 'php'. The window has a dark background and a light gray border. In the top right corner, there are three colored window control buttons (green, yellow, red) and a clipboard icon. The code inside the editor is as follows:

```
$empleados = [  
    'Juan' => 34,  
    'Luisa' => 56  
];
```

- Para leer será de la misma forma, salvo que ya no tenemos posiciones sino nuestras propias claves.

Arrays

- Diccionario

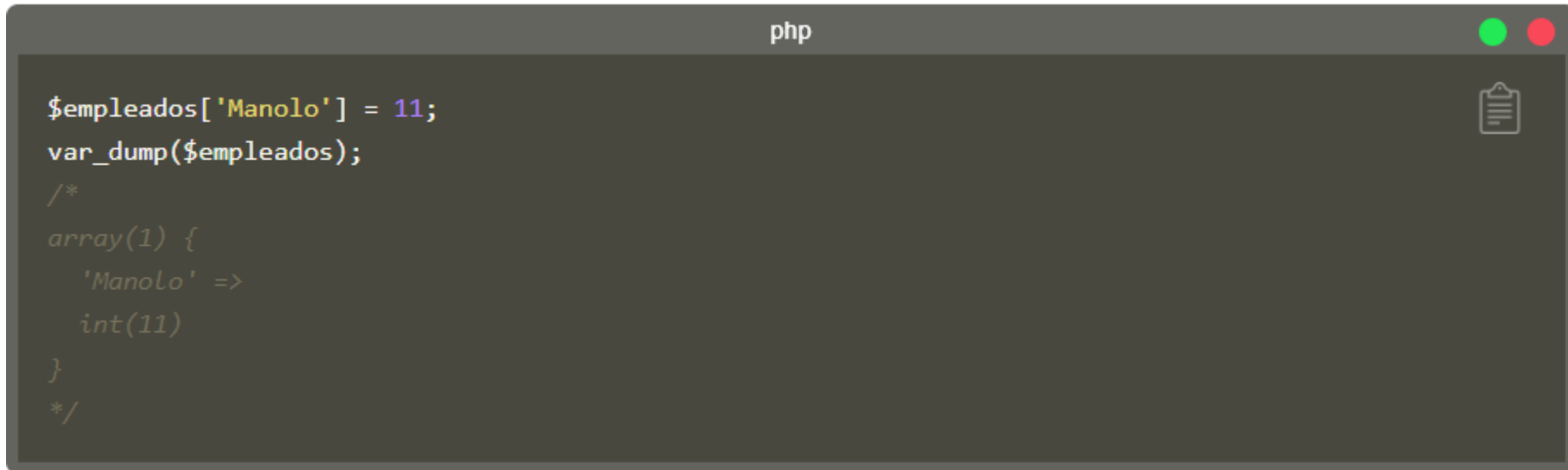
```
php
echo $empleados['Luisa'];
```

- A la hora de añadir tendrás que indicar directamente el nombre que quieras darle.

```
php
$empleados = [];
$empleados['Manolo'] = 99;
var_dump($empleados);
/*
array(1) {
    'Manolo' =>
        int(99)
}
*/
```

Arrays

- Diccionario
 - Modificar será igual, indicando la clave.



```
$empleados['Manolo'] = 11;
var_dump($empleados);
/*
array(1) {
    'Manolo' =>
        int(11)
}
*/
```

Arrays

- Diccionario
 - Y borrar de la misma forma que un array.

```
php

$empleados = [];
$empleados['Manolo'] = 99;
$empleados['Juan'] = 99;
var_dump($empleados);
/*
array(2) {
    'Manolo' =>
        int(99)
    'Juan' =>
        int(99)
}
*/
unset($empleados['Manolo']);
var_dump($empleados);
/*
array(1) {
    'Juan' =>
        int(99)
}
*/
```

Arrays

- Actividad 7 – Censo de población
 - Crea un diccionario con el censo de población de: España, Portugal, Francia, Italia y Grecia. Ayúdate de Internet. Un ejemplo:

```
php

$censo = [
    'España' => 99999,
    ...
]
```

- Ordena de mayor a menor. ¡Pista! `asort` hará el trabajo por ti:

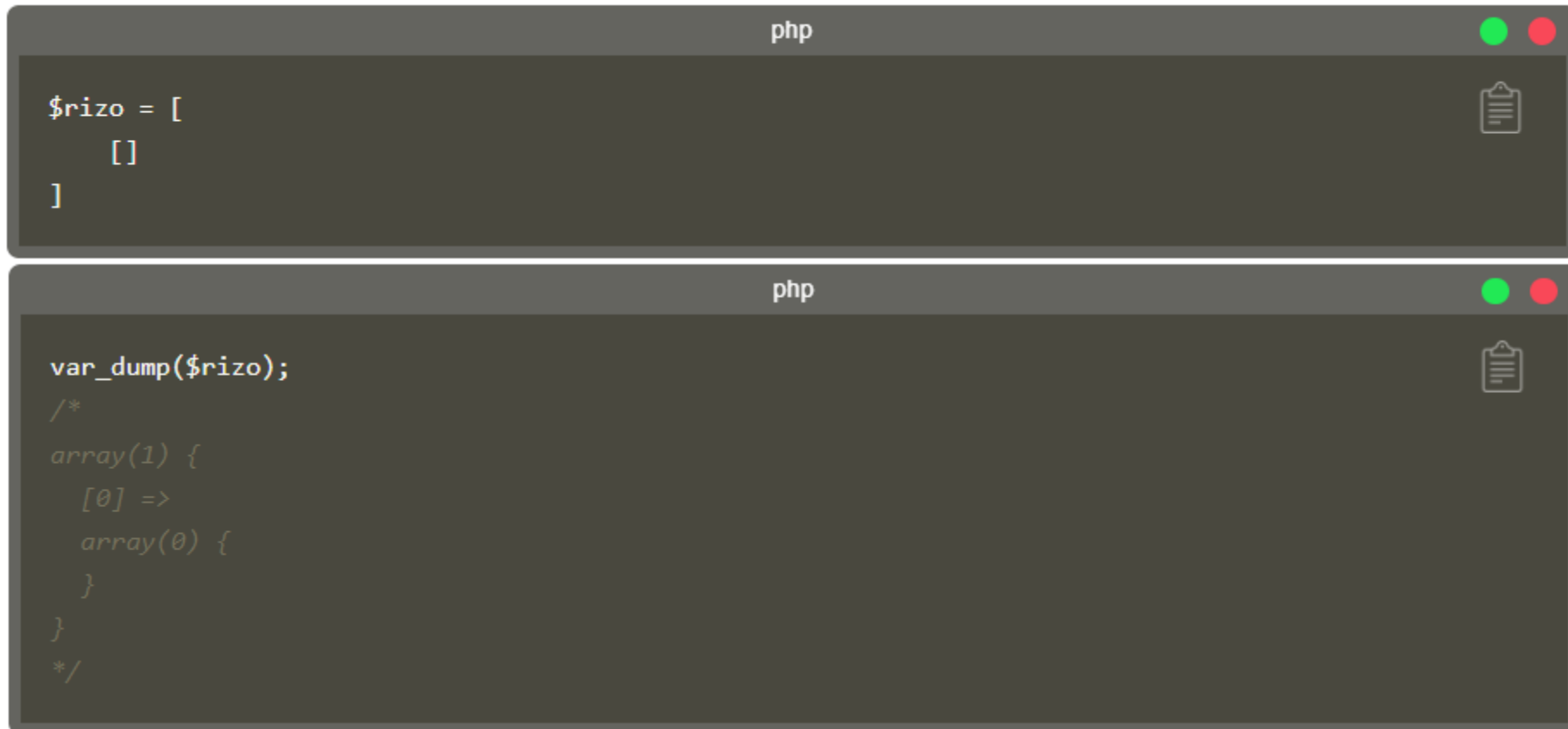
```
php

asort($censo, SORT_DESC);
```

- Imprime con `var_dump`.

Arrays

- Dos dimensiones
 - Cuando un array tiene dentro otro array.



```
php

$rizo = [
    []
]

var_dump($rizo);
/*
array(1) {
    [0] =>
        array(0) {
        }
    }
}
*/
```

Arrays

- Dos dimensiones
 - Tiene infinidad de usos. Pensemos que has heredado de un tío rico una tienda de ropa. Cada producto tiene su propio código de barras, precio, nombre, color y género.

```
php
$zara = [
    123 => [
        'nombre' => 'Camisa a cuadros',
        'precio' => 29.95,
        'sexo' => 'Hombre'
    ],
    234 => [
        'nombre' => 'Falda manga',
        'precio' => 19.95,
        'sexo' => 'Mujer'
    ],
    345 => [
        'nombre' => 'Bolso minúsculo',
        'precio' => 50,
        'sexo' => 'Mujer'
    ]
];
```

Arrays

- Dos dimensiones
 - El mecanismo para gestionarlo es igual al diccionario salvo que debemos ir nodo por nodo.

```
echo $zara[345]['nombre'];  
// Bolso minúsculo
```

```
var_dump($zara);  
/*  
array(3) {  
  [123] =>  
    array(3) {  
      'nombre' =>  
        string(16) "Camisa a cuadros"  
      'precio' =>  
        double(29.95)  
      'sexo' =>  
        string(6) "Hombre"  
    }  
  [234] =>  
    array(3) {  
      'nombre' =>  
        string(11) "Falda manga"  
      'precio' =>  
        double(19.95)  
      'sexo' =>  
        string(5) "Mujer"  
    }  
  [345] =>  
    array(3) {  
      'nombre' =>  
        string(16) "Bolso minúsculo"  
      'precio' =>  
        int(50)  
      'sexo' =>  
        string(5) "Mujer"  
    }  
  }  
}  
*/
```

Bucles

- Un bucle, o loop, se utiliza para repetir un conjunto de instrucciones en una serie de ocasiones definida.
- Ello quiere decir que puedes realizar la misma tarea tantas veces como necesites: una, ninguna, x veces; pero no infinitas porque si no nunca llegaría a terminar de generarse el HTML. Cada repetición se denomina **iteración**.

Bucles

- Si ejecutas el siguiente código...



```
<html>
  <body>
    <h1>¿Cuantos años tienes?</h1>
    <select>
      <?php foreach (range(1, 10) as $num): ?>
        <option value="<?php echo $num; ?>"><?php echo $num . ' años'; ?></option>
      <?php endforeach; ?>
    </select>
  </body>
</html>
```

Bucles

- ... te generará este HTML.

```
html
<html>
  <body>
    <h1>¿Cuántos años tienes?</h1>
    <select>
      <option value="1">1 años</option>
      <option value="2">2 años</option>
      <option value="3">3 años</option>
      <option value="4">4 años</option>
      <option value="5">5 años</option>
      <option value="6">6 años</option>
      <option value="7">7 años</option>
      <option value="8">8 años</option>
      <option value="9">9 años</option>
      <option value="10">10 años</option>
    </select>
  </body>
</html>
```

Bucles

- Dentro de PHP existen 4 tipos de bucles:
 - foreach
 - for
 - while
 - do-while
- Todos son diferentes estrategias a la hora de decidir el número de veces que se va a iterar, en el fondo hacen lo mismo.

Bucles

- foreach
 - Es la forma más sencilla de iterar un array.

```
php

$animalesFantasticos = ['fénix', 'dragón', 'grifo', 'pegaso', 'cerbero'];
foreach ($animalesFantasticos as $animal) {
    echo $animal . ' ';
}

// fénix dragón grifo pegaso cerbero
```

- En caso de que necesitemos la key, existe otra forma para utilizarlo.

```
php

$animalesFantasticos = ['fénix', 'dragón', 'grifo', 'pegaso', 'cerbero'];
foreach ($animalesFantasticos as $posicion => $animal) {
    echo "El animal $animal ocupa la posición $posicion \n";
}

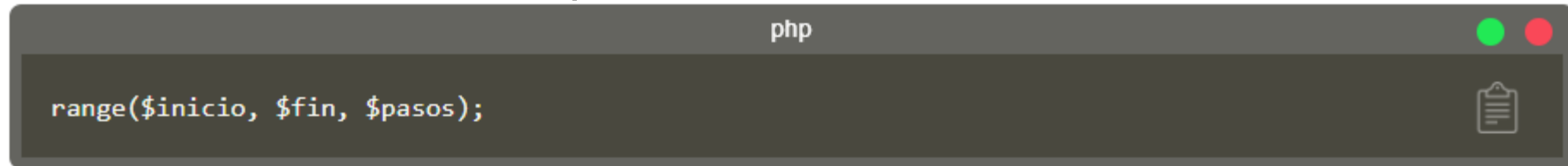
// El animal fénix ocupa la posición 0
// El animal dragón ocupa la posición 1
// El animal grifo ocupa la posición 2
// El animal pegaso ocupa la posición 3
// El animal cerbero ocupa la posición 4
```


Bucles

- Actividad 8 – Lista de películas
 - Realiza los siguientes pasos:
 - Guarda en un array tus 6 películas favoritas.
 - Imprime en párrafos con el siguiente formato: ‘Película: Los Vengadores’
 - Añade la posición de la película: ‘Película 4: Godzilla’
 - Pro:
 - Imprime en lugar de párrafos... ¡una tabla!
 - Añade un poco de CSS para mejorar el diseño. Cada título debe tener un color aleatorio. ¡Pista!: `random_int(0, 255)`

Bucles

- foreach
 - Respecto a range() es una función nativa de PHP que genera un array de elementos. Admite 2 o 3 parámetros.



```
range($inicio, $fin, $pasos);
```

The image shows a dark-themed code editor window with the title 'php'. Inside the editor, the PHP function `range($inicio, $fin, $pasos);` is written in a light-colored font. The window has standard macOS-style window controls (red, yellow, and green buttons) in the top right corner and a clipboard icon in the bottom right corner.

Bucles

- foreach

```
var_dump(range(10, 15));
```

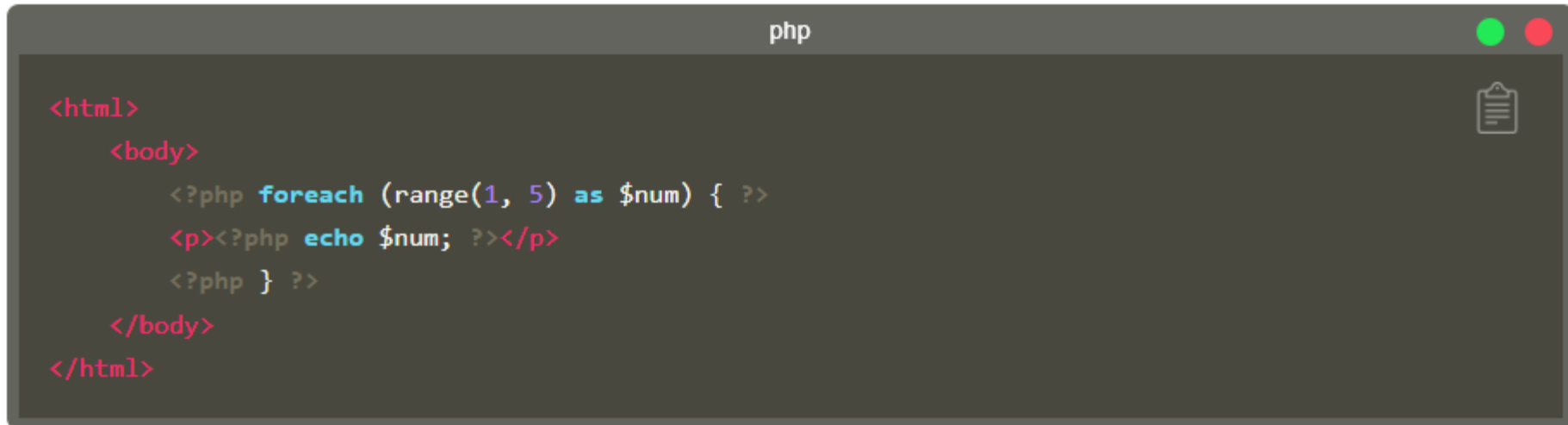
```
/*  
array(6) {  
    [0] =>  
        int(10)  
    [1] =>  
        int(11)  
    [2] =>  
        int(12)  
    [3] =>  
        int(13)  
    [4] =>  
        int(14)  
    [5] =>  
        int(15)  
}  
*/
```

```
var_dump(range(0, 100, 20));
```

```
/*  
array(6) {  
    [0] =>  
        int(0)  
    [1] =>  
        int(20)  
    [2] =>  
        int(40)  
    [3] =>  
        int(60)  
    [4] =>  
        int(80)  
    [5] =>  
        int(100)  
}  
*/
```

Bucles

- foreach
 - Para insertar un bucle dentro de un HTML tienes 2 formas. En el ejemplo está escrito con foreach pero es válido cualquier loop o condicional.
- Sintaxis clásica.



```
<html>
  <body>
    <?php foreach (range(1, 5) as $num) { ?>
      <p><?php echo $num; ?></p>
    <?php } ?>
  </body>
</html>
```

Bucles

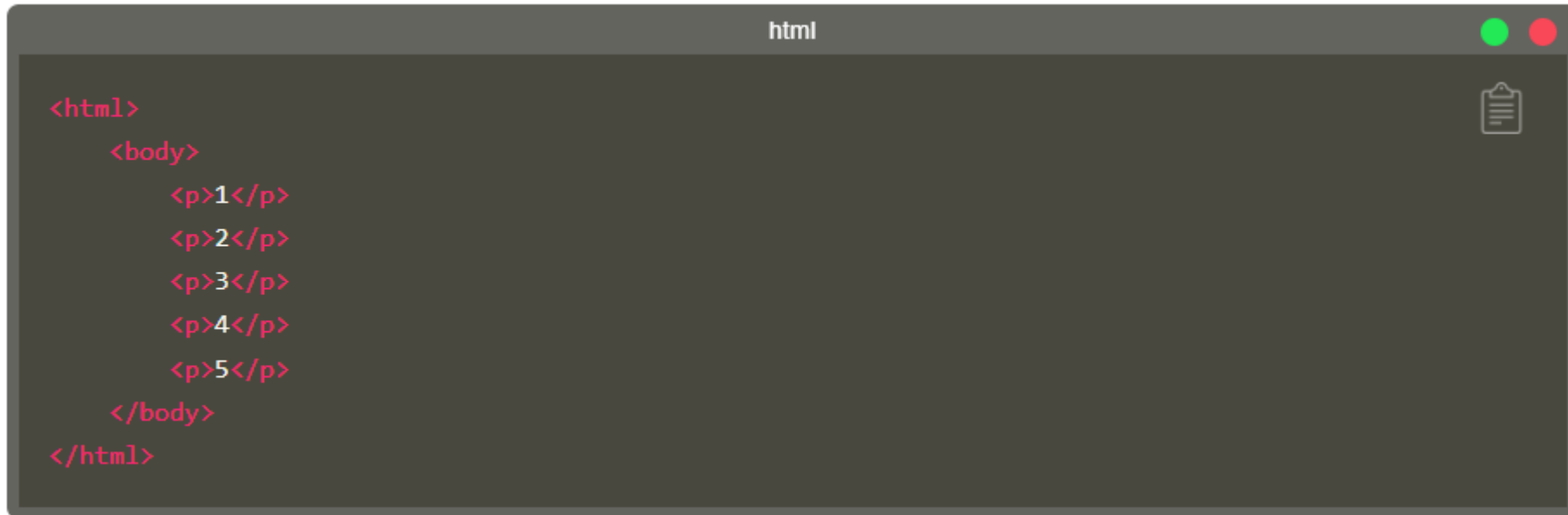
- foreach
 - Sintaxis alternativa.

A code editor window with a dark background and a title bar that says 'php'. The code is written in PHP and HTML, using a syntax-highlighted color scheme. The code is as follows:

```
<html>
  <body>
    <?php foreach (range(1, 5) as $num): ?>
      <p><?php echo $num; ?></p>
    <?php endforeach; ?>
  </body>
</html>
```

Bucles

- foreach
 - Ambos dan el mismo resultado.



```
<html>
  <body>
    <p>1</p>
    <p>2</p>
    <p>3</p>
    <p>4</p>
    <p>5</p>
  </body>
</html>
```

Bucles

- Actividad 9 – Jugando con Bucles
 - Realiza los siguientes pasos:
 - Imprime los números del 1 al 10.
 - Imprime los números de 60 al 70.
 - Imprime los números del 20 al 1.
 - Imprime los números del 1 al 1000
 - Imprime la tabla del 5.
 - Pro:
 - Imprime la tabla del 5 con este formato: $5 \times 3 = 15$
 - Suma los números del 1 al 100.

Bucles

- Recorrer arrays multidimensionales
 - Para poder leer un array con más de una dimensión, tendremos que realizar bucles anidados. O un bucle dentro de otro bucle.
- Tomamos de partida un array, de 2 dimensiones, que vimos en la lección anterior.

A screenshot of a code editor window titled 'php'. The code defines a 2D array named '\$zara' containing three items, each with 'nombre', 'precio', and 'sexo' properties. The first item is a shirt for a man, the second is a skirt for a woman, and the third is a small bag for a woman.

```
$zara = [  
    123 => [  
        'nombre' => 'Camisa a cuadros',  
        'precio' => 29.95,  
        'sexo' => 'Hombre'  
    ],  
    234 => [  
        'nombre' => 'Falda manga',  
        'precio' => 19.95,  
        'sexo' => 'Mujer'  
    ],  
    345 => [  
        'nombre' => 'Bolso minúsculo',  
        'precio' => 50,  
        'sexo' => 'Mujer'  
    ]  
];
```

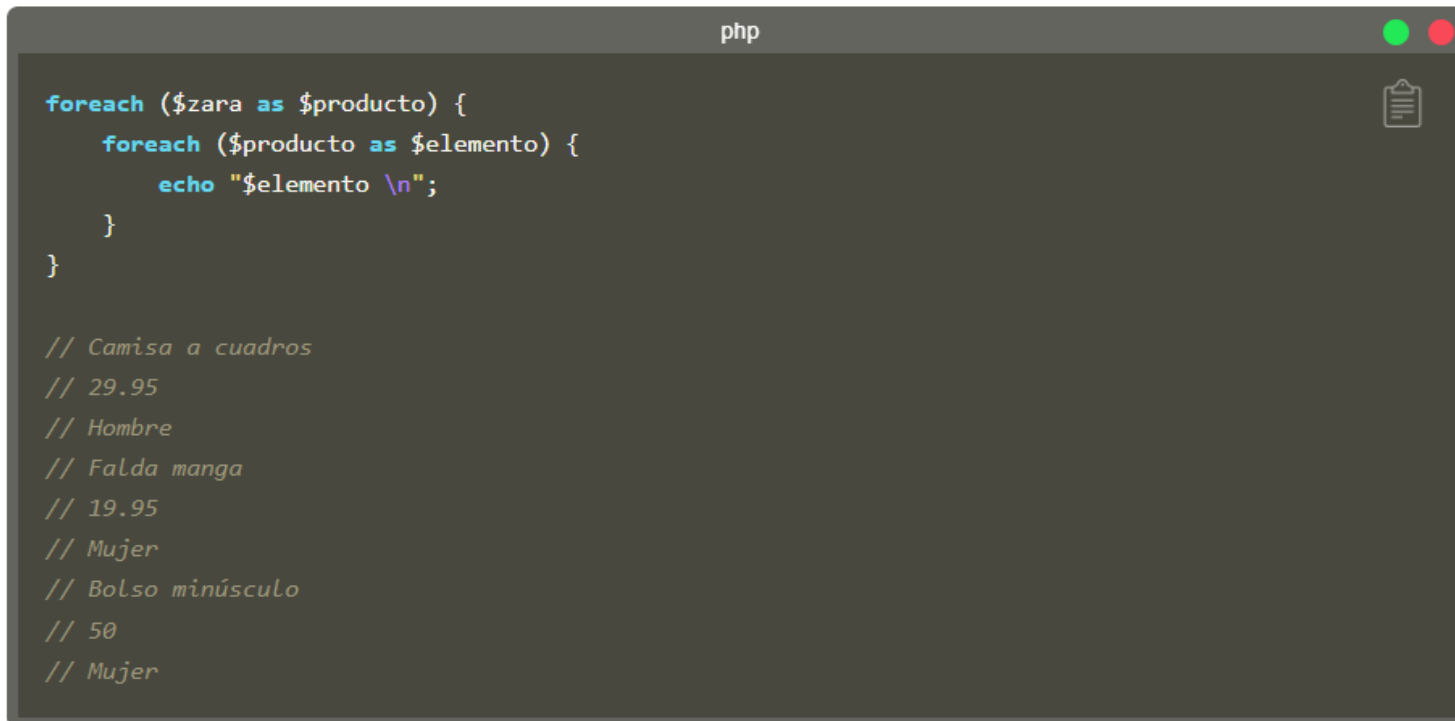

Bucles

- Recorrer arrays multidimensionales
 - Si se quisiera mostrar toda la información de los productos.

```
foreach ($zara as $producto) {  
    var_dump($producto);  
}  
  
/*  
array(3) {  
    'nombre' =>  
        string(16) "Camisa a cuadros"  
    'precio' =>  
        double(29.95)  
    'sexo' =>  
        string(6) "Hombre"  
}  
  
array(3) {  
    'nombre' =>  
        string(11) "Falda manga"  
    'precio' =>  
        double(19.95)  
    'sexo' =>  
        string(5) "Mujer"  
}  
  
array(3) {  
    'nombre' =>  
        string(16) "Bolso minúsculo"  
    'precio' =>  
        int(50)  
    'sexo' =>  
        string(5) "Mujer"  
}  
*/
```

Bucles

- Recorrer arrays multidimensionales
 - Se ha iterado 3 veces, y en cada ocasión ha devuelto un array. Es lo que hay contenido dentro del primer array, otros arrays. Entonces se debe recorrer cada uno de ellos con otro foreach.



```
php

foreach ($zara as $producto) {
    foreach ($producto as $elemento) {
        echo "$elemento \n";
    }
}

// Camisa a cuadros
// 29.95
// Hombre
// Falda manga
// 19.95
// Mujer
// Bolso minúsculo
// 50
// Mujer
```

Bucles

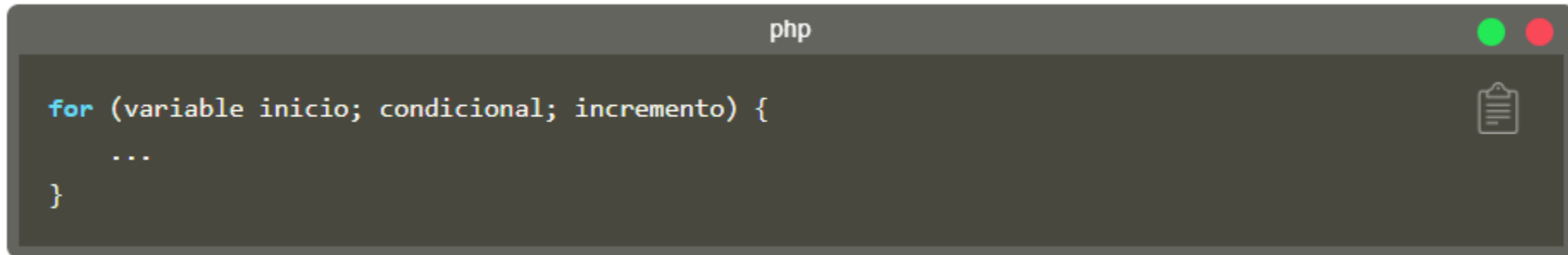
- Recorrer arrays multidimensionales
 - Los bucles anidados se pueden realizar con cualquier tipo de loop.
 - Se recomienda que trabajes siempre que puedas con foreach ya que será más difícil que acabes teniendo un bucle infinito.

Bucles

- Actividad 10 – Test
 - Supongo que el ejemplo anterior está claro y no tienes dudas. ¡Demuéstramelo!
 - ¿Cuántas veces se ejecuta el primer foreach?
 - ¿Cuántas veces se ejecuta el segundo foreach?
 - ¿Cuántos echos se han realizado? ¿Corresponde a las respuestas anteriores?
 - En el ejemplo tienes 2 loops anidados. ¿Cuántos piensas que pueden existir como máximo (un bucle dentro de un bucle de otro bucle...)?

Bucles

- for
 - El loop más complejo y similar a otros lenguajes (C, Java, Javascript...).

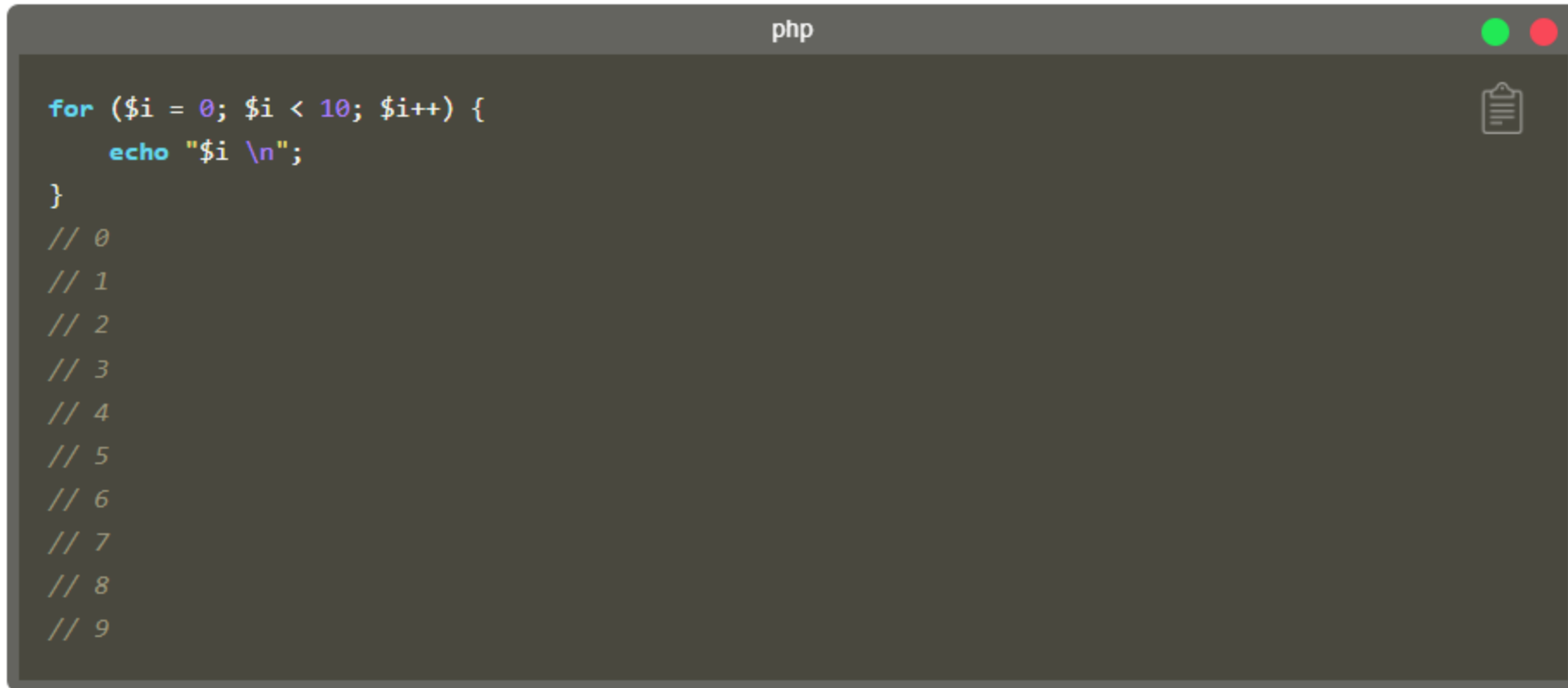


```
for (variable inicio; condicional; incremento) {  
    ...  
}
```

The image shows a code editor window with a dark background. The title bar at the top is labeled 'php' and has three window control buttons (green, yellow, red) on the right. The code inside the editor is a for loop syntax: `for (variable inicio; condicional; incremento) {` on the first line, `...` on the second line, and `}` on the third line. A clipboard icon is visible on the right side of the editor area.

Bucles

- for

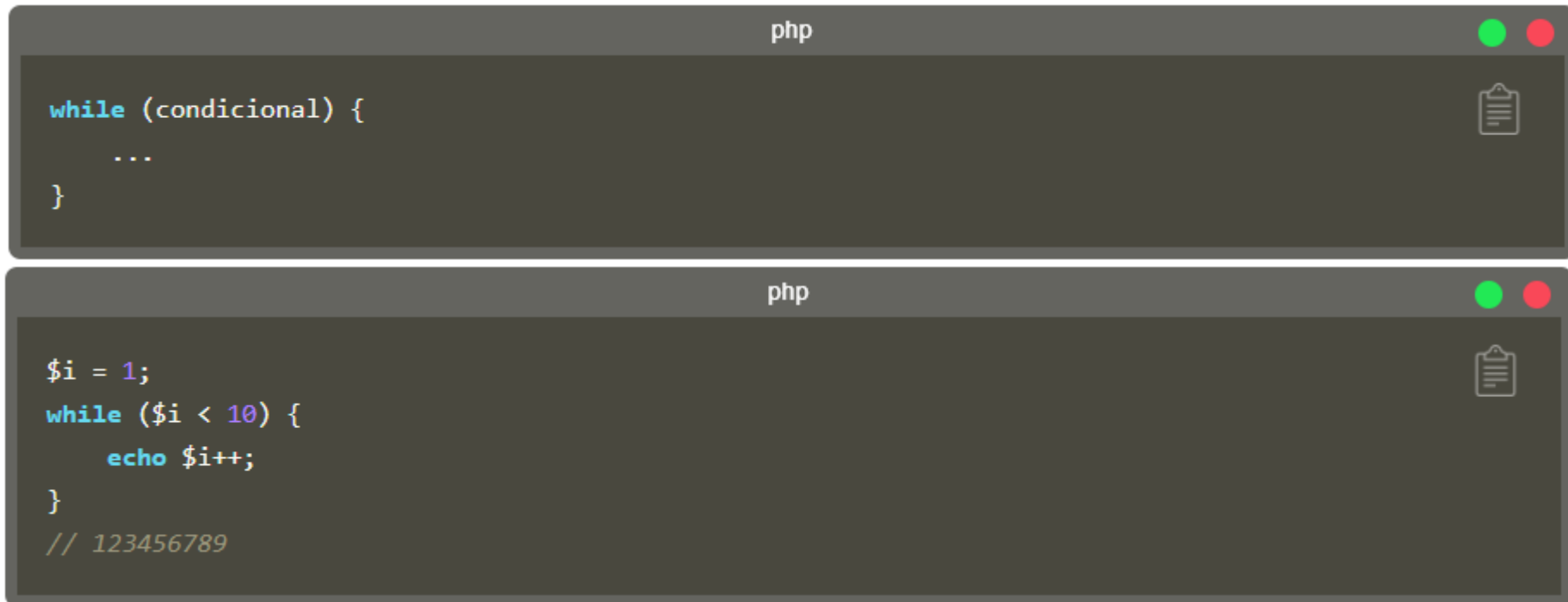
A screenshot of a code editor window titled 'php'. The window has a dark background and a light gray border. In the top right corner, there are three colored window control buttons (green, yellow, red) and a clipboard icon. The code inside the editor is a PHP for loop that prints numbers from 0 to 9. The code is as follows:

```
for ($i = 0; $i < 10; $i++) {  
    echo "$i \n";  
}  
  
// 0  
// 1  
// 2  
// 3  
// 4  
// 5  
// 6  
// 7  
// 8  
// 9
```

- `$i++` es equivalente a `$i += 1` o `$i = $i + 1`. Básicamente incrementa en 1 la variable. Además dispones de su antítesis: `$i--`.

Bucles

- while
 - Es el bucle más sencillo y peligroso. Debes prestar mucha atención para que se acabe en algún momento.



The image shows two screenshots of a code editor window titled 'php'. The top screenshot displays the general syntax of a while loop:

```
while (condicional) {  
    ...  
}
```

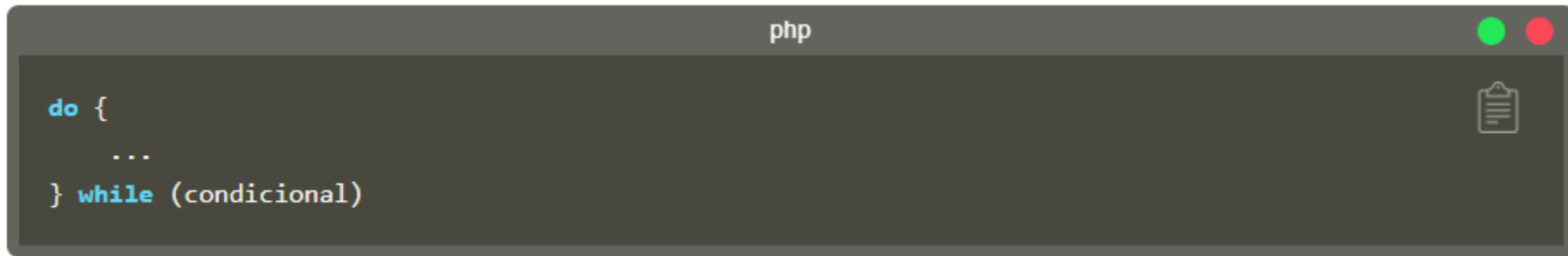
. The bottom screenshot shows a specific example:

```
$i = 1;  
while ($i < 10) {  
    echo $i++;  
}  
// 123456789
```

. Both screenshots include a clipboard icon in the top right corner.

Bucles

- do-while
 - Se comporta igual que while, salvo que se compromete a ejecutarse al menos una vez.
 - Independientemente de si se cumple la condicional. El secreto radica en que primero se ejecuta las instrucciones y a continuación se evalúa el condicional.



```
do {  
    ...  
} while (condicional)
```


Bucles

- do-while

```
php

$i = 1;
do {
    echo $i++;
} while ($i < 10);
// 123456789
```

```
php

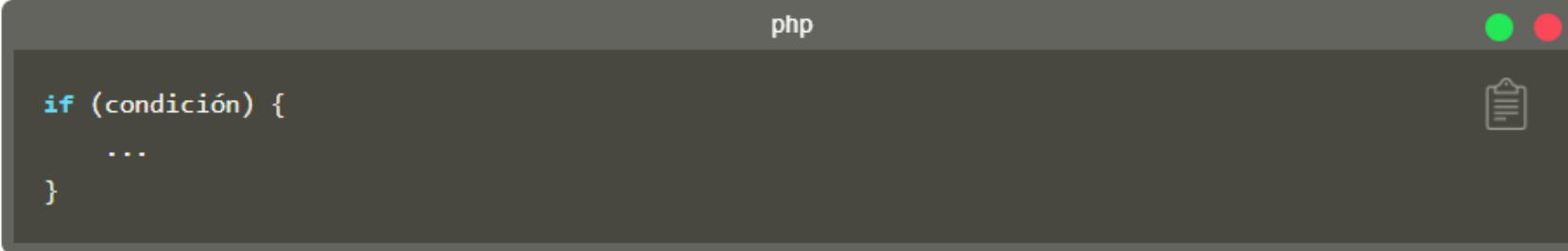
$i = 20;
do {
    echo $i++;
} while ($i < 10);
// 20
```

Bucles

- Actividad 11 – Fecha de nacimiento
 - Crea un select para pedir el día de nacimiento: 1 al 31. Usa un foreach.
 - A su otro lado select para pedir el mes de nacimiento: 1 al 12. Usa un for.
 - Y a continuación otro select para pedir el año de nacimiento: 1900 al año actual. Usa un while.

Condicionales

- Los condicionales son una herramienta esencial en cualquier lenguaje de programación. Sirve para ejecutar instrucciones dependiendo de unos condicionales.



```
if (condición) {  
    ...  
}
```

The image shows a code editor window with a dark background. The title bar at the top is labeled 'php' and has three colored window control buttons (green, yellow, red) on the right. The code inside the editor is a PHP conditional statement: 'if (condición) {' followed by an indented line '...' and a closing brace '}'.

Condicionales

```
php

echo "Inicio \n";
if (2 > 0) {
    echo "Entro en el condicional \n";
}
echo 'Fin';
// Inicio
// Entro en el condicional
// Fin
```

```
php

echo "Inicio \n";
if (2 > 1000) {
    echo "Entro en el condicional \n";
}
echo 'Fin';
// Inicio
// Fin
```

Condicionales

- Tipos de condicionales
 - Las condicionales son símbolos aritméticos.
 - Si la lógica es cierta, entrará. En caso contrario no se cumplirá y nunca llegará a ejecutarse el interior del if o while (ambos usan condicionales).
 - Además, podremos concatenar todas las condicionales que necesitemos. El uso de paréntesis está permitido.

Condicionales

- Tipos de condicionales

```
php
if (10 > 2 && True && 'HBO' != 'Netflix') {
    echo 'Entro seguro';
}
// Entro seguro
```

Símbolo	Explicación	Ejemplo
>	es mayor que	if (1 > 0)
<	es menor que	if (1 < 0)
&&	y	if (1 > 0 && 67 > 0)
	o	if (1 > 10 67 > 0)
!	no	if (!(1 > 0))
==	es igual en valor	if ('3' == 3)
===	es igual en valor y tipo	if ('3' === '3')
!=	no es igual	if ('Doctor' != 'Who')
!==	no es igual en valor o tipo	if ('Doctor' !== 'Who')
>=	es mayor o igual que	if (10 >= 10)
<=	es menor o igual que	if (10 <= 20)
<=>	-1, 0 y 1 dependiendo de los valores si son superados	(10 <=> 20) // 1
True	Verdad	if (True)
False	Falso	if (False)

Bucles

- Actividad 12 – Logicando la lógica
 - Indica en cada caso si entraría el condicional.

```
1. if (True && True)
2. if (False && True)
3. if (1 == 1 && 2 == 1)
4. if ("test" == "test")
5. if (1 == 1 || 2 != 1)
6. if (True && 1 == 1)
7. if (False && 0 != 0)
8. if (True || 1 == 1)
9. if ("test" == "testing")
10. if (1 != 0 && 2 == 1)
```

```
11. if ("test" != "testing")
12. if ("test" == 1)
13. if (!(True && False))
14. if (!(1 == 1 && 0 != 1))
15. if (!(10 == 1 || 1000 == 1000))
16. if (!(1 != 10 || 3 == 4))
17. if (!("testing" == "testing" && "Zed" == "Cool Guy"))
18. if (1 == 1 && !("testing" == 1 || 1 == 0)))
19. if ("chunky" == "bacon" && !(3 == 4 || 3 == 3)))
20. if (3 == 3 && !("testing" == "testing" || "PHP" == "Fun")))
```

Condicionales

- else
 - Existe una herramienta que nos ayudará a que solo una serie de instrucciones se ejecuten.



```
php

if (condición) {
    ...
} else {
    ...
}
```

A screenshot of a code editor window titled 'php'. The window has a dark background and a light gray border. In the top right corner, there are three colored circles (green, yellow, red) and a clipboard icon. The code inside the window is a PHP conditional statement: `if (condición) {`, followed by an indented line `...`, then `} else {`, followed by another indented line `...`, and finally `}`.

- En caso de cumplirse la condición entrará en las primeras llaves ({}), en caso contrario entrará en las segundas llaves. Pero nunca en ambas.

Condicionales

- else

```
php

if ('Ghibli' == 'Ghibli') {
    echo 'Bienvenido';
} else {
    echo 'No eres bien recibido'
}

// Bienvenido
```

```
php

if ('Estudio' == 'Ghibli') {
    echo 'Bienvenido';
} else {
    echo 'No eres bien recibido'
}

// No eres bien recibido
```

Condicionales

- elseif
 - Es posible tener varios condicionales.

```
php

if (condición) {
    ...
} elseif (condición) {
    ...
}
```

```
php

if ('Michael Jackson' == 'Moonwalker') {
    echo 'Gran baile';
} elseif ('Moonwalker' == 'Moonwalker') {
    echo 'Legendario';
}

// Legendario
```

Condicionales

- elseif
 - Al añadir un else al final le daremos un caso “por defecto”. Si no se cumpliera ninguna condicional iría automáticamente.

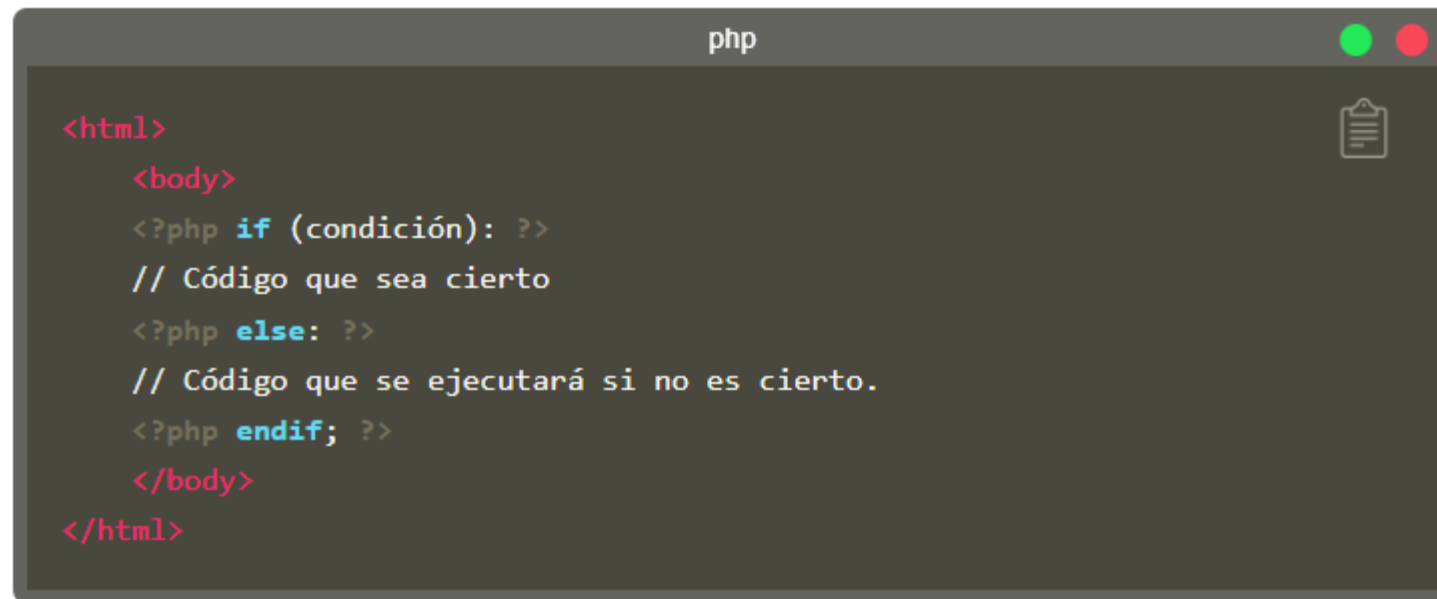


```
if (condición) {  
    ...  
} elseif (condición) {  
    ...  
} else {  
    ...  
}
```

- Se admite tanto elseif como else if. Salvo como sintaxis alternativa, que debe estar junto.

Condicionales

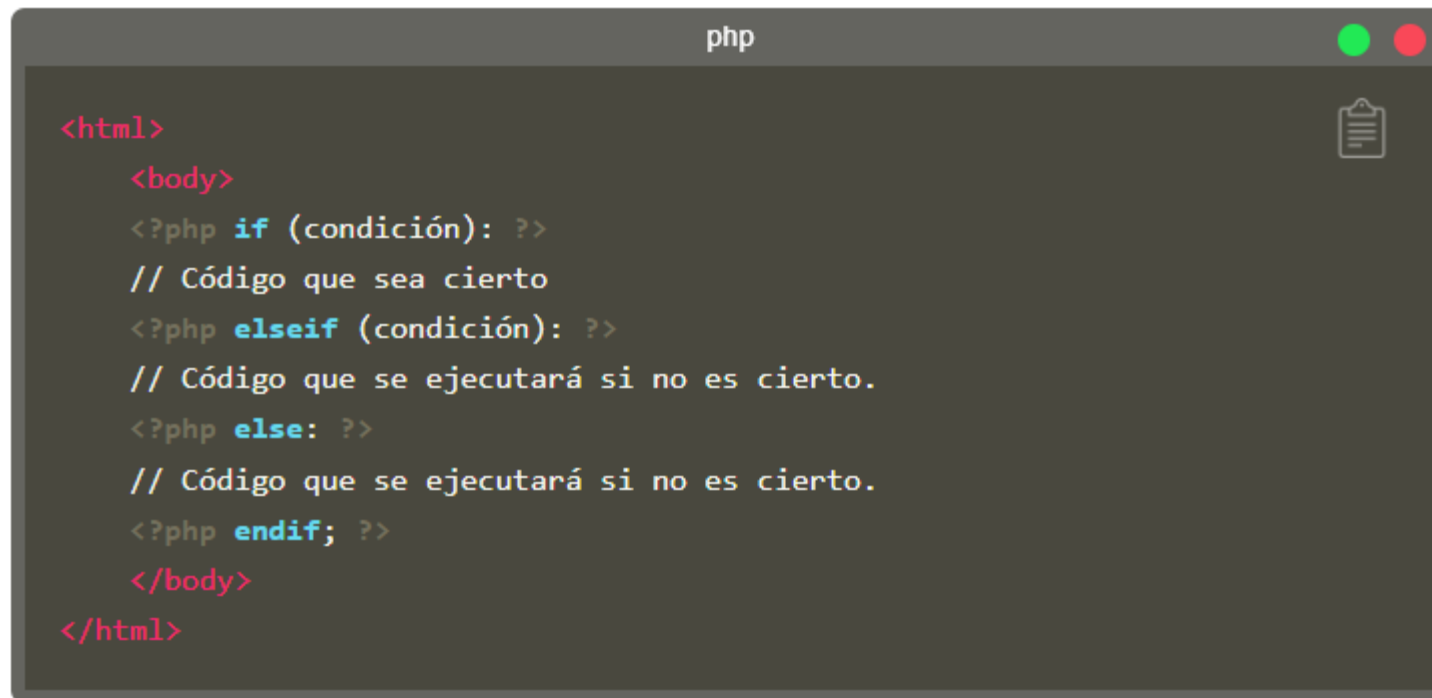
- Sintaxis alternativa
 - Para facilitar la integración con HTML dispones de una sintaxis alternativa y un poco más agradable al ojo.



```
<html>
  <body>
    <?php if (condición): ?>
      // Código que sea cierto
    <?php else: ?>
      // Código que se ejecutará si no es cierto.
    <?php endif; ?>
  </body>
</html>
```

Condicionales

- Sintaxis alternativa
 - O con un elseif.

A screenshot of a code editor window with a dark background. The window title is 'php' and it has standard macOS window controls (red, yellow, green buttons) in the top right corner. A clipboard icon is visible in the top right of the editor area. The code is written in a light-colored font and shows the following structure:

```
<html>
  <body>
    <?php if (condición): ?>
    // Código que sea cierto
    <?php elseif (condición): ?>
    // Código que se ejecutará si no es cierto.
    <?php else: ?>
    // Código que se ejecutará si no es cierto.
    <?php endif; ?>
  </body>
</html>
```

Condicionales

- Forma abreviada (Operador ternario)
 - Es posible ejecutar en una sola instrucción con if con un else. Si estas empezando no te recomiendo usarla, pero no la olvides.

```
php
<?php (condicional) ? 'Valor si se cumple' : 'Valor si no se cumple'; ?>
```

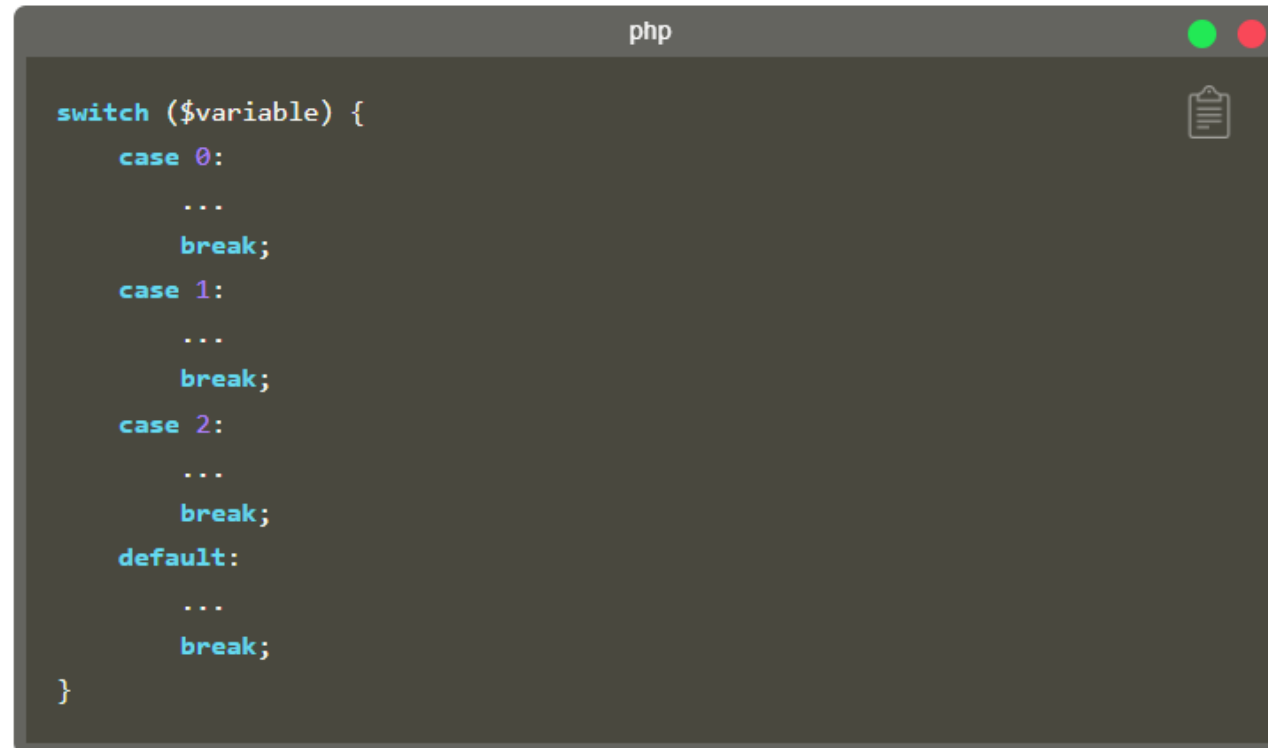
```
php
<?php echo (5 > 10) ? 'Es verdad' : 'Es mentira'; ?>
```

Bucles

- Actividad 13 – Portero de discoteca automático
 - Realiza los siguientes pasos:
 - Pide el año de nacimiento.
 - Calcula la edad.
 - Si es mayor de edad, dile que puede pasar dentro.
 - Si es menor, tírale.
 - Si tiene más de 65 años, dile que es demasiado mayor para entrar.
 - Pro:
 - Obtén el año del sistema en lugar de escribirlo a mano en una variable.
 - Pro2:
 - Pide además el día y el mes de nacimiento para saber si ha cumplido el año actual.

Condicionales

- Switch
 - La funcionalidad de switch es prácticamente igual a if salvo que es más limitada, solo admite la condicional de igualación.



```
switch ($variable) {  
    case 0:  
        ...  
        break;  
    case 1:  
        ...  
        break;  
    case 2:  
        ...  
        break;  
    default:  
        ...  
        break;  
}
```


Condicionales

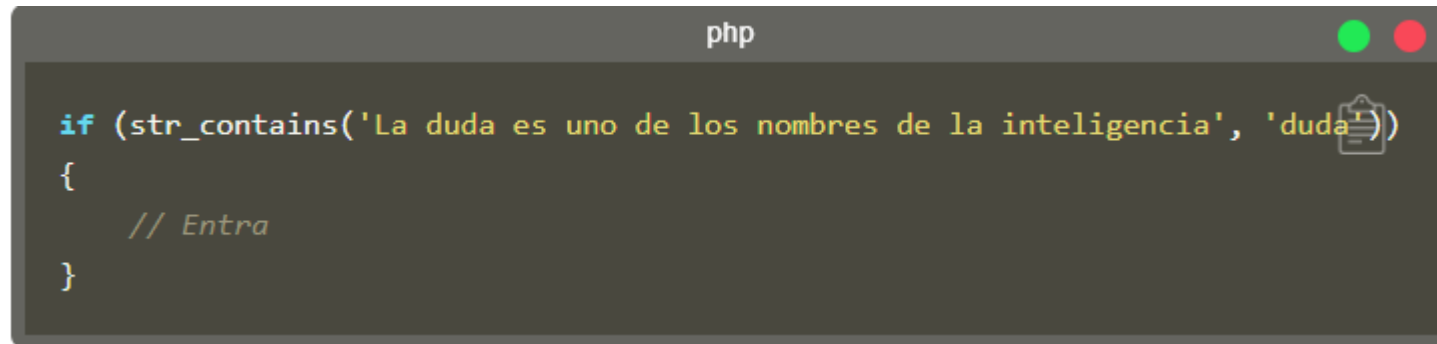
- Switch
 - Una equivalencia entre ambos.

```
php
$num = 1;
if ($num == 0) {
    echo "num es igual a 0";
} elseif ($num == 1) {
    echo "num es igual a 1";
} elseif ($num == 2) {
    echo "num es igual a 2";
} else {
    echo "No se a que es igual";
}
// num es igual a 1

switch ($num) {
    case 0:
        echo "num es igual a 0";
        break;
    case 1:
        echo "num es igual a 1";
        break;
    case 2:
        echo "num es igual a 2";
        break;
    default:
        echo "No se a que es igual";
        break;
}
// num es igual a 1
```

Condicionales

- Strings
 - Funciones que te pueden ser de ayuda cuando trabajas con String.
 - `str_contains` (¿Contiene este texto este otro texto?)



```
if (str_contains('La duda es uno de los nombres de la inteligencia', 'duda'))  
{  
    // Entra  
}
```

Condicionales

- Strings

- `str_starts_with` (¿Empieza este texto con este otro texto?)

```
php

if (str_starts_with('La duda es uno de los nombres de la inteligencia', 'La d
uda es')) {
    // Entra
}
```

- `str_end_with` (¿Termina este texto con este otro texto?)

```
php

if (str_end_with('La duda es uno de los nombres de la inteligencia', 'intelig
encia')) {
    // Entra
}
```

Formularios

- Los formularios es la única manera de que el usuario nos transmita información, y tenemos una gran cantidad de posibilidades para recoger: textos, números, archivos, checks...
- Para construir nuestro formulario necesitaremos la etiqueta `<form>` y dentro todos los `<input>`s que necesitemos.

Formularios

A code editor window with a dark background and a title bar that says 'php'. The code is written in a syntax-highlighted format. It shows an HTML document structure with a body containing a form. Inside the form, there is a PHP code block that uses the var_dump function to display the contents of the \$_REQUEST superglobal. Below the PHP code, there is a text input field with the name 'nombre' and a submit button.

```
<html>
  <body>
    <form>
      <?php
        var_dump($_REQUEST);
      ?>
      <input type="text" name="nombre">
      <input type="submit">
    </form>
  </body>
</html>
```

- Utiliza `var_dump($_REQUEST)` para saber que variables te llegan de un formulario.

Formularios

- `$_REQUEST` es un array que contiene todas las variables que recibimos, lo cual nos simplifica a la hora de extraer cada elemento.
- La función `isset()` te indica si existe cualquier tipo de variable: local, global o dentro de un array.



```
<html>
  <body>
    <?php if (isset($_REQUEST['nombre'])): ??>
      <p>¿De verdad te llamas <?php echo $_REQUEST['nombre']; ??>? Qué n
ombre más bonito.</p>
    <?php endif; ??>
    <form>
      <input type="text" name="nombre">
      <input type="submit">
    </form>
  </body>
</html>
```

Formularios

- Métodos de petición
 - Existe una gran cantidad de verbos HTTP, o métodos de petición, dentro del desarrollo web:
 - GET
 - POST
 - PUT
 - DELETE
 - HEAD
 - CONNECT
 - OPTIONS
 - TRACE
 - PATH

Formularios

- Métodos de petición
 - Todos ellos son como etiquetas que le damos a los datos para marcar su uso.
 - Piensa en ellos como diferentes caminos para llegar al mismo sitio.



```
if ($_SERVER['REQUEST_METHOD'] === 'POST') {  
    echo 'Llega algo por POST';  
}
```


Formularios

- Método GET
 - Por defecto un formulario es enviado con el método, verbo, GET.
 - Tiene una particularidad que no tiene el resto.

Formularios

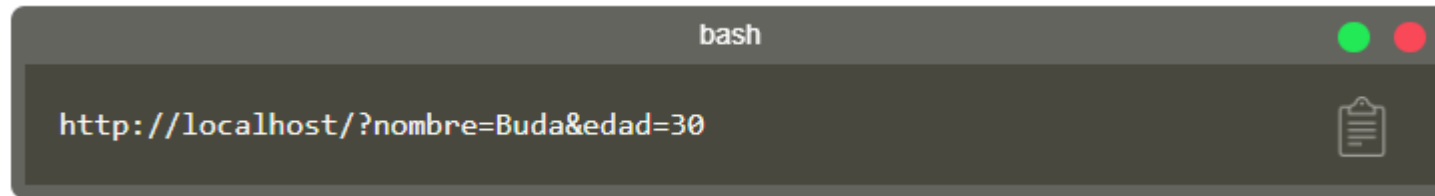
- Método GET
 - Pongamos el siguiente código. Yo lo relleno con el nombre “Buda” y la edad “30”. A continuación, pulso el botón de enviar.

A code editor window with a dark background and a title bar that says 'php'. The code is written in a syntax-highlighted style. It defines an HTML document with a body containing a form. The form uses the GET method and has two input fields: a text field named 'nombre' and a number field named 'edad', followed by a submit button.

```
<html>
  <body>
    <form method="get">
      <input type="text" name="nombre">
      <input type="number" name="edad">
      <input type="submit">
    </form>
  </body>
</html>
```

Formularios

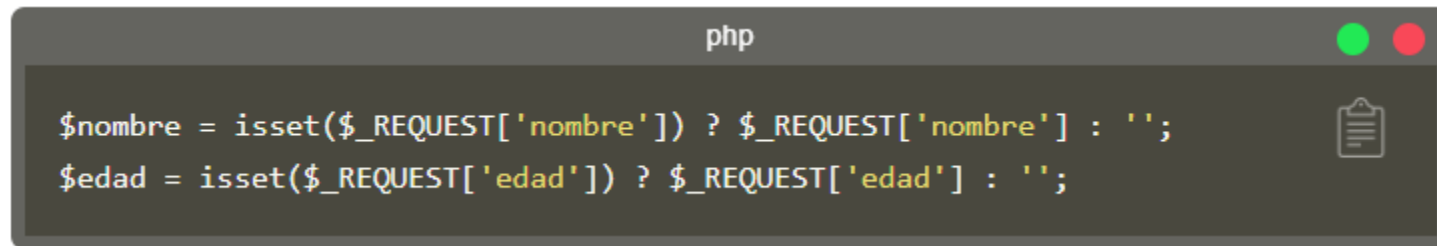
- Método GET
 - Si te fijas en la dirección, la URL, verás una ruta parecida a la siguiente.



- GET nos permite ver en la barra del URL los datos dándonos la posibilidad de modificarlos.
- Cualquier persona observadora podrá ver y modificar los datos. Ten mucho cuidado. Úsalo para información que no comprometa tu sitio: paginadores, mensajes, marcador de idioma...

Formularios

- Método GET
 - Para recoger será usando siempre isset() con \$_REQUEST.



```
$nombre = isset($_REQUEST['nombre']) ? $_REQUEST['nombre'] : '';  
$edad = isset($_REQUEST['edad']) ? $_REQUEST['edad'] : '';
```

Formularios

- Actividad 14 – Contacto fraudulento
 - Realiza los siguientes pasos:
 - Realiza un formulario con los siguientes datos: nombre, teléfono, email y mensaje.
 - Cuando se pulse en enviar debe mostrar la siguiente plantilla.

“Hola **nombre!**

Te voy a enviar spam a **correo** y te llamaré por la madrugada a **teléfono**.

mensaje

Enviado desde un iPhone”

Formularios

- Método POST
 - El método POST será invisible al ojo del usuario. Recomendable cuando modificamos variables o una base de datos.

A code editor window with a dark background and a title bar that says 'php'. The code is written in a syntax-highlighted style. It defines an HTML document with a body containing a form. The form uses the POST method and includes three input fields: a text field for 'nombre', a number field for 'edad', and a submit button.

```
<html>
  <body>
    <form method="post">
      <input type="text" name="nombre">
      <input type="number" name="edad">
      <input type="submit">
    </form>
  </body>
</html>
```

Formularios

- Actividad 15 – ¿Quién saca al perro?
 - Realiza los siguientes pasos:
 - Escribe en un textarea una lista de nombres.
 - Cuando pulses un botón debes mostrar un nombre aleatorio. (Será el encargado de sacar al perro)
 - Muestra con la siguiente plantilla: **nombre** saca al perro.
 - Ejemplo en textarea:
 - Batman Superman Ironman Pescanova
 - Cuando es pulsado el botón...
 - Ironman saca al perro.

Formularios

- Actividad 16 – Cueva de Gollum
 - Realiza los siguientes pasos:
 - Muestra la siguiente adivinanza:
“Esta cosa se devora a todas las cosas;
Pájaros, bestias, árboles, flores;
Carcome el hierro, muerde el acero;
Muele duras piedras y las reduce a harina;
Mata al rey, arruina la ciudad,
Y derriba a la montaña.”
 - En un **input**, pide la respuesta.
 - Añade un botón de **submit**.
 - Si se pulsa el botón debes comprobar si ha acertado. La respuesta es: Tiempo.
 - Si acierta felicítale.
 - Si pierde, muestra la respuesta y cómetelo.

Formularios

- Método POST
 - Para obtener una variable get también dispones de `$_GET['nombre']` y para post `$_POST['nombre']`, pero `$_REQUEST['nombre']` simplifica al unificar cualquier elemento que recibamos.

Formularios

- Action
 - Si no indicas lo contrario, la información se enviará a la misma página donde estemos. Con action podremos decirle al formulario que lleve los datos a otra URL.

A code editor window with a dark background and a title bar that says 'php'. The code is written in a syntax-highlighted style. It shows an HTML document structure with a form. The form has a method of 'post' and an action of 'login.php'. It contains two input fields: one for 'nombre' (text) and one for 'edad' (number), followed by a submit button.

```
<html>
  <body>
    <form method="post" action="login.php">
      <input type="text" name="nombre">
      <input type="number" name="edad">
      <input type="submit">
    </form>
  </body>
</html>
```

Formularios

- Evitar que se borren los campos
 - Cada vez que pulsas en un submit la página se refresca (se realiza una petición) y con ello se reinicia el formulario.
 - ¿Qué pasa si me he equivocado en algún campo? Se pierde como lágrimas en la lluvia.
 - El usuario tendría que volver a rellenarlo. Un truco para solucionarlo sería comprobar si existe el dato, y si es así rellenar su value.

Formularios

- Evitar que se borren los campos

```
php
<html>
  <body>
    <form>
      <input type="text" placeholder="Nombre" name="nombre">
      <input type="number" placeholder="Edad" name="edad">
      <input type="submit">
    </form>
  </body>
</html>
```

Formularios

- Actividad 17 – Calculadora de IVA dinámica
 - Realiza los siguientes pasos:
 - Vuelve a realizar una calculadora de IVA, pero en esta ocasión la cantidad no estará guardada en una variable, sino que nos la proporcionará el usuario.
 - Pista: Para calcular el IVA debes aplicar la siguiente formula $\text{precio} / 1.21$.

Formularios

- Campus ocultos

- En ciertas ocasiones necesitaremos enviar por el formulario una información que no sea visible por el usuario: id, token, un histórico, un cálculo, etc.
- Para ello disponemos de un input especial con el tipo hidden.



```
<html>
  <body>
    <form>
      <input type="hidden" name="maquina-enigma" value="149">

      <input type="submit">
    </form>
  </body>
</html>
```

Formularios

- Campus ocultos
 - En este caso, el visitante solo visualizará un botón pero al enviarse llegará la data maquina-enigma con el valor 149.



```
$codigoSecreto = isset($_REQUEST['maquina-enigma']) ? $_REQUEST['maquina-enigma'] : '';  
echo $codigoSecreto;  
  
// 149
```

- También puede ser utilizado para enviar arrays. Tan solo debemos repetir el name dejando presente [].

Formularios

- Campus ocultos

```
php
<html>
  <body>
    <form>
      <input type="hidden" name="filtros[]" value="precio">
      <input type="hidden" name="filtros[]" value="valoracion">
      <input type="hidden" name="filtros[]" value="fecha">

      <input type="submit">
    </form>
  </body>
</html>
```

```
php
$misFiltros = isset($_REQUEST['filtros']) ? $_REQUEST['filtros'] : [];
var_dump($misFiltros);

// ['precio', 'valoracion', 'fecha']
```


Formularios

- Campus ocultos
 - Aunque también nos podemos apoyar en las funciones `serialize()` y `unserialize()`. Nos convertirá los arrays en texto o texto en arrays.



```
php

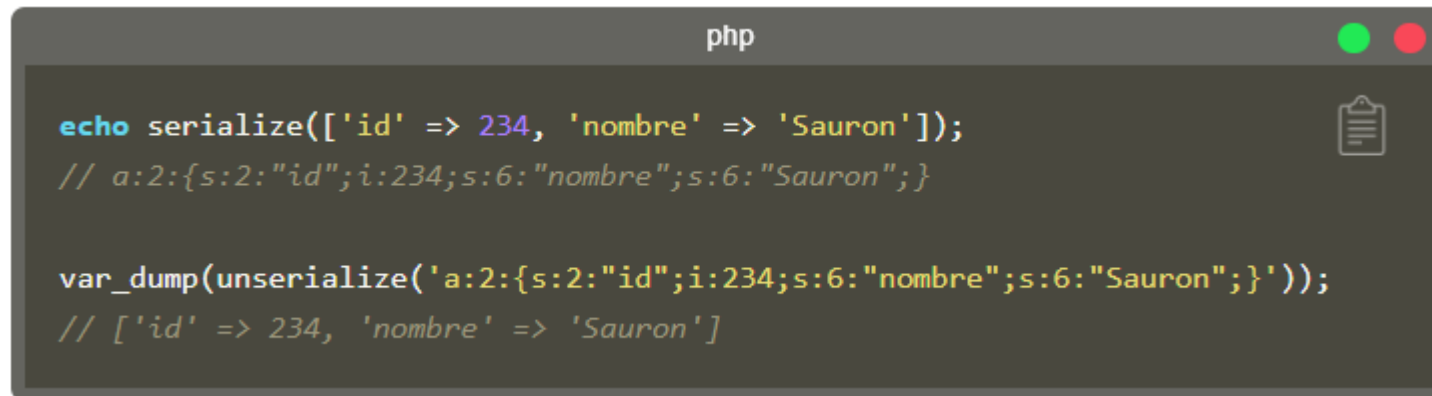
echo serialize(['mañana', 'tarde', 'noche']);
// a:3:{i:0;s:7:"mañana";i:1;s:5:"tarde";i:2;s:5:"noche";}

var_dump(unserialize('a:3:{i:0;s:7:"mañana";i:1;s:5:"tarde";i:2;s:5:"noche"}'));
// ['mañana', 'tarde', 'noche']
```

- Estas funciones nos permiten mantener estructuras avanzadas como los diccionarios.

Formularios

- Campus ocultos



```
php

echo serialize(['id' => 234, 'nombre' => 'Sauron']);
// a:2:{s:2:"id";i:234;s:6:"nombre";s:6:"Sauron";}

var_dump(unserialize('a:2:{s:2:"id";i:234;s:6:"nombre";s:6:"Sauron";}'));
// ['id' => 234, 'nombre' => 'Sauron']
```

- Si volvemos al ejemplo anterior, deberemos tener cuidado con las comillas dobles que entrarán en conflicto con las propias de atributo HTML. En su lugar envuelve con comillas simples.

Formularios

- Campos ocultos

```
php

<?php
$filtros = ['precio', 'valoracion', 'fecha'];
?>

<html>
    <body>
        <form>
            <input type="hidden" name="filtros" value='<?= serialize($filtros); ?>'>

            <input type="submit">
        </form>
    </body>
</html>
```

```
php

$filtroSerializados = isset($_REQUEST['filtros']) ? $_REQUEST['filtros'] : '';
$filtroDeserializados = unserialize($filtroSerializados);
var_dump($filtroDeserializados);

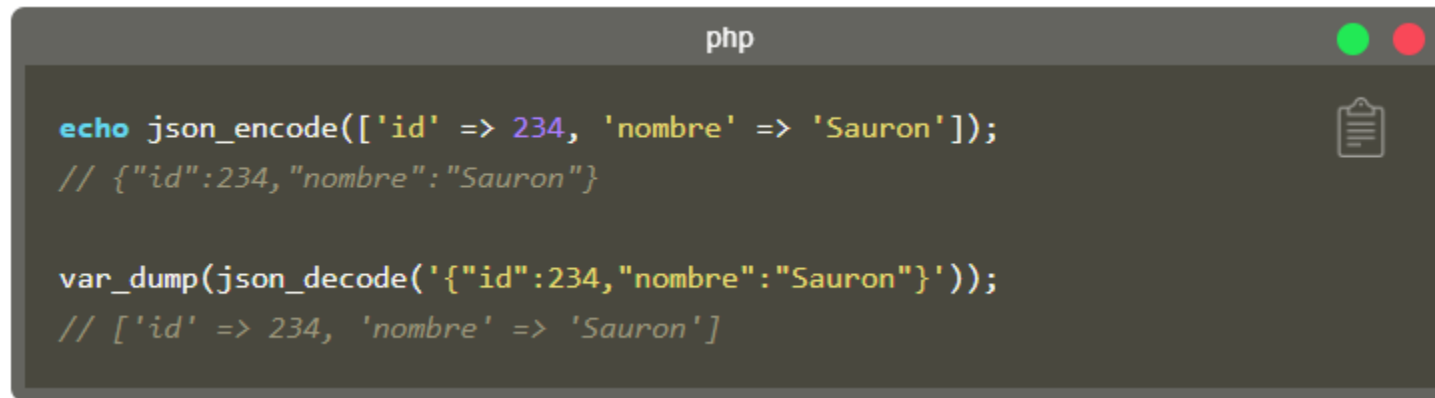
// ['precio', 'valoracion', 'fecha']
```

Formularios

- Campus ocultos
 - Es inseguro serializar un string que proviene de un cliente.
 - Se practica porque es cómodo y rápido para recuperar array u objetos, aunque lamentablemente debes evitarlo ya que es posible manipular el texto recibido para inyectar código malicioso en el backend.
 - En su lugar recibe un JSON, ya que sus campos son muy limitados, y construye manualmente todos los elementos.

Formularios

- Campus ocultos
 - Una última opción, la cual es actualmente un estándar cuando hablamos de una estructura de datos, es trabajar con JSON.



```
echo json_encode(['id' => 234, 'nombre' => 'Sauron']);  
// {"id":234,"nombre":"Sauron"}  
  
var_dump(json_decode('{"id":234,"nombre":"Sauron"}'));  
// ['id' => 234, 'nombre' => 'Sauron']
```

- Si utilizamos el ejemplo anterior quedaría tal que así.

Formularios

- Campos ocultos
 - Cada situación posee una mejor solución.
- La experiencia te guiará cual te conviene utilizar.

```
php

<?php
$filtros = ['precio', 'valoracion', 'fecha'];
?>

<html>
    <body>
        <form>
            <input type="hidden" name="filtros" value='<?= json_encode($filtros); ?>'>

            <input type="submit">
        </form>
    </body>
</html>
```

```
php

$filtroEnJSON = isset($_REQUEST['filtros']) ? $_REQUEST['filtros'] : '';
$filtro = json_decode($filtroEnJSON);
var_dump($filtro);

// ['precio', 'valoracion', 'fecha']
```

Formularios

- Actividad 18 – Listado de películas
 - Realiza los siguientes pasos:
 - Crea un input y un botón de submit.
 - Rellena el campo con el nombre de una película.
 - Cuando se pulse debe guardar el contenido en un array.
 - Imprime el resultado en una tabla.

Formularios

- Actividad 19 – Notas de los alumnos
 - Realiza los siguientes pasos:
 - Realiza un array o diccionario con unos alumnos y sus respectivas notas.
 - Marta: 7,8 Luis: 5 Lorena: 6,9 ...
 - Muestra las notas de una forma ordenada.

Alumno	Nota
Marta	7,8
Luis	5

- Da la posibilidad de añadir nuevos alumnos.
 - Pro:
 - Muestra la media en la parte inferior.

Formularios

- Actividad 20 – Montaña rusa
 - Para entrar en la atracción vamos a construir un validador que minimice las víctimas mortales. Para ello usaremos varios requisitos.
 - Debe superar una altura de 120cm.
 - Debe tener una edad superior a 16 años.
 - ¿Rechaza llevarnos a juicio por daños y perjuicios de un mal mantenimiento?
 - En caso de ser todo válido le daremos el ticket.
 - Pro:
 - Genera un ticket con su nombre y un número único. Ejemplo: “Alfonso, ticket 00034”.

Formularios

- Actividad 21 – Formulario de padre
 - Realiza los siguientes pasos:
 - Pide el nombre.
 - Pide el sexo.
 - Pide el número de hijos.
 - Muestra la siguiente frase dependiendo de los datos anteriores:
 - El señor Pepe tiene 1 hijo.
 - El señor Pepe tiene 4 hijos.
 - La señora Sonia tiene 1 hijo.
 - La señora Sonia no tiene hijos.

Funciones

- Cuando utilizamos en múltiples ocasiones un mismo fragmento de código debemos usar funciones (functions).
- Una herramienta para encapsular y ejecutar un mismo código.



```
// Declarar
function nombre_de_funcion(tipo_de_parametro $parametros): tipo_return
{
    ...
    return ...;
}

// Llamar
nombre_de_funcion($parametros);
```

Funciones

- En el ejemplo inferior tenemos una función clásica, con sintáxis moderna, donde es declarada y ejecutada seguido de un echo para ver el resultado. Lo único que hace es devolver un texto cuando es llamado.



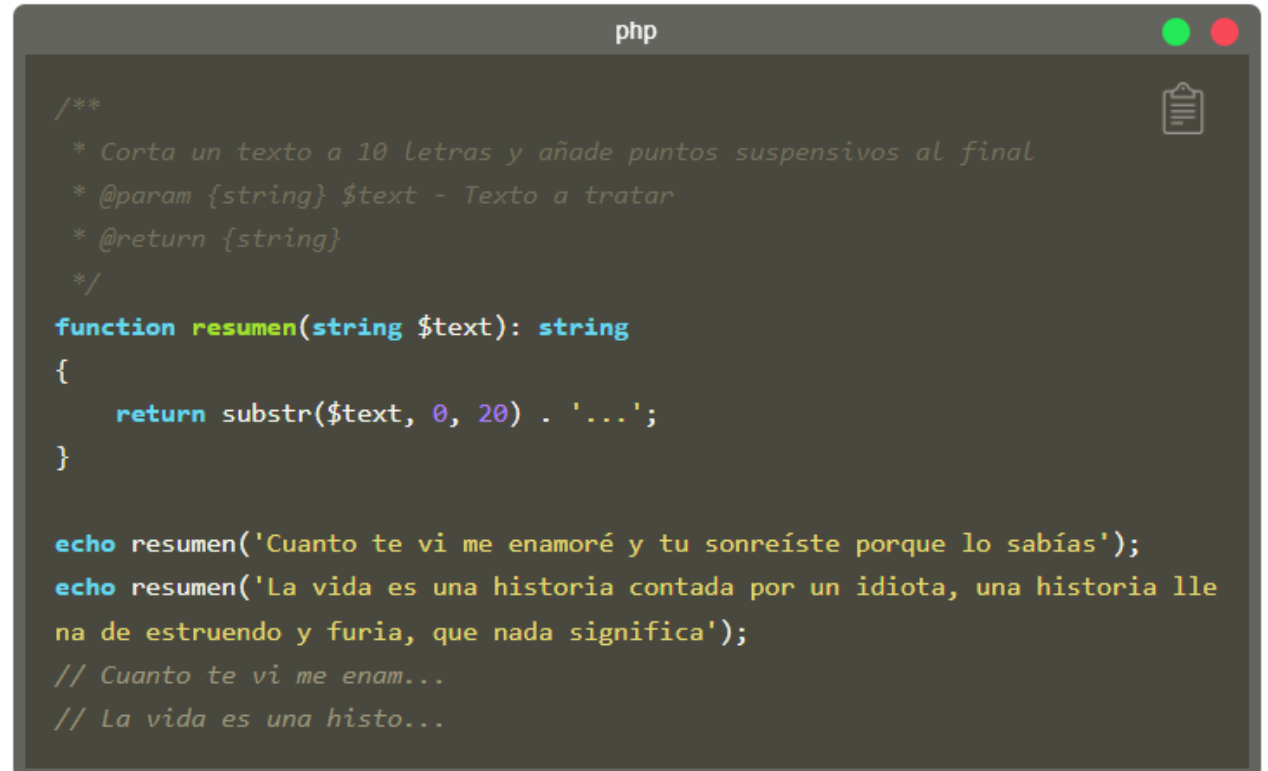
```
/**
 * Función con educación
 * @return {string}
 */
function saludar_ahora(): string
{
    return 'Hola, soy una función';
}
echo saludar_ahora();
// Hola, soy una función
```

Funciones

- La receta para hacer una función rica y con fundamento:
 - Las 4 primeras líneas es el formato de comentarios. Una función, por mucha prisa que tengas, debe estar comentada con el formato del ejemplo.
 - La palabra **function** está reservada. A continuación, el **nombre**, que debe estar en minúsculas con guiones bajos en lugar de espacios. Después unos **paréntesis** con los argumentos. Si no tiene, se dejan vacíos, pero siempre presentes. Luego **dos puntos**. Por último, el tipo de **valor resultante**.
 - Llaves para envolver tu código {}.
 - Y dentro la palabra reservada return seguido del valor a devolver.

Funciones

- Parámetros
 - Nuestras funciones serán más interesantes si les damos algunos parámetros. Al darle variables podemos usar siempre el mismo código, pero con algunas variaciones.

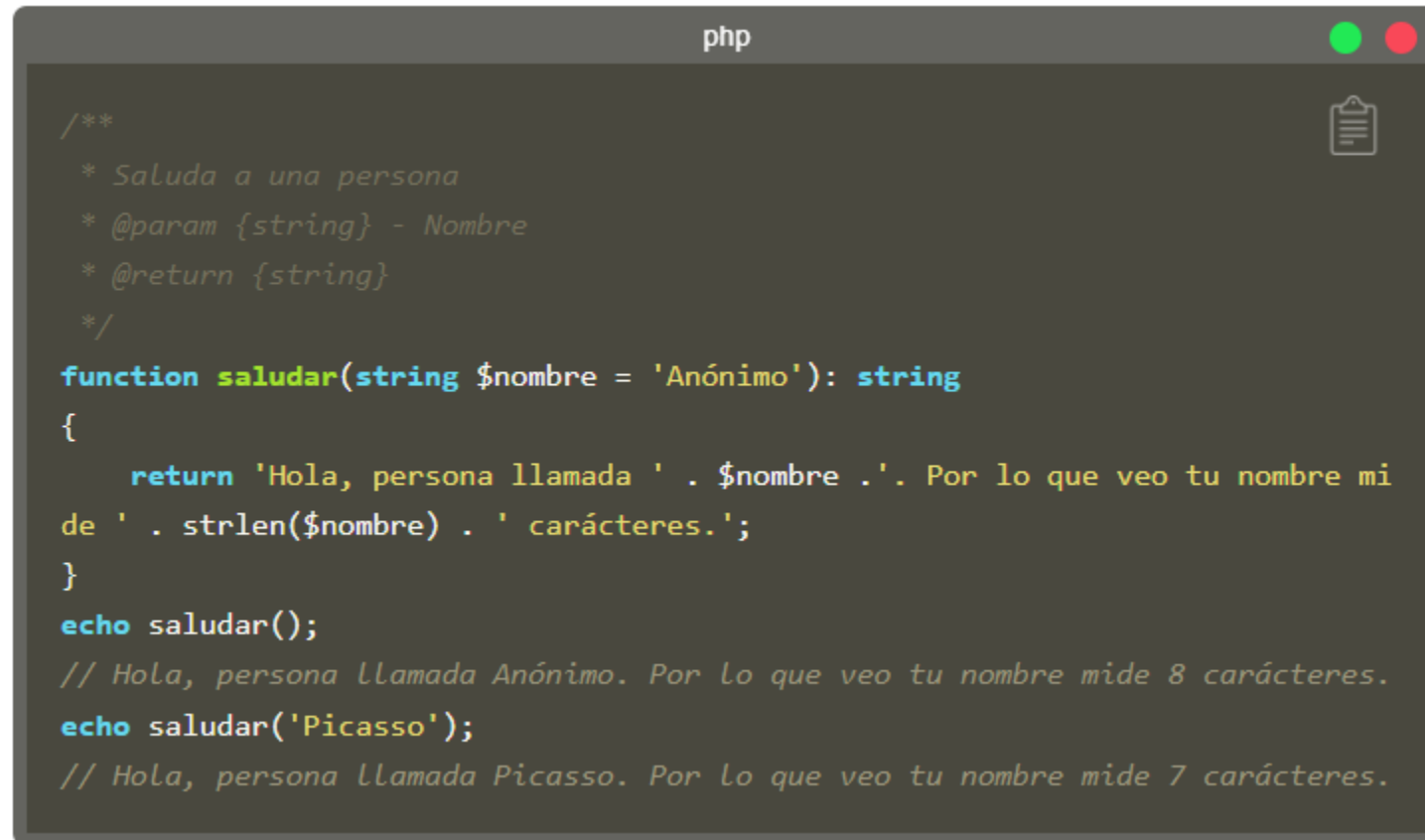


```
/**
 * Corta un texto a 10 letras y añade puntos suspensivos al final
 * @param {string} $text - Texto a tratar
 * @return {string}
 */
function resumen(string $text): string
{
    return substr($text, 0, 20) . '...';
}

echo resumen('Cuanto te vi me enamoré y tu sonreíste porque lo sabías');
echo resumen('La vida es una historia contada por un idiota, una historia llena de estruendo y furia, que nada significa');
// Cuanto te vi me enam...
// La vida es una histo...
```

Funciones

- Parámetros
 - Nuestros parámetros de entrada pueden tener un valor por defecto.



```
php

/**
 * Saluda a una persona
 * @param {string} - Nombre
 * @return {string}
 */
function saludar(string $nombre = 'Anónimo'): string
{
    return 'Hola, persona llamada ' . $nombre . '. Por lo que veo tu nombre mide ' . strlen($nombre) . ' caracteres.';
}

echo saludar();
// Hola, persona llamada Anónimo. Por lo que veo tu nombre mide 8 caracteres.

echo saludar('Picasso');
// Hola, persona llamada Picasso. Por lo que veo tu nombre mide 7 caracteres.
```

Funciones

- Parámetros
 - Y, por supuesto, podemos añadir varios parámetros.

```
php
/**
 * Saluda a una persona
 * @param {string} - Nombre
 * @param {string} - Profesión
 * @return {string}
 */
function saludar(string $nombre = 'Anónimo', string $profesion = 'ninguna'):
string
{
    return 'Hola, persona llamada ' . $nombre . '. Por lo que veo tu nombre mide ' . strlen($nombre) . ' caracteres. De profesión ' . $profesion . '.';
}
echo saludar();
// Hola, persona llamada Anónimo. Por lo que veo tu nombre mide 8 caracteres.
De profesión ninguna.
echo saludar('Espartaco');
// Hola, persona llamada Espartaco. Por lo que veo tu nombre mide 9 caracteres.
De profesión ninguna.
echo saludar('Picasso', 'pintor');
// Hola, persona llamada Picasso. Por lo que veo tu nombre mide 7 caracteres.
De profesión pintor.
```

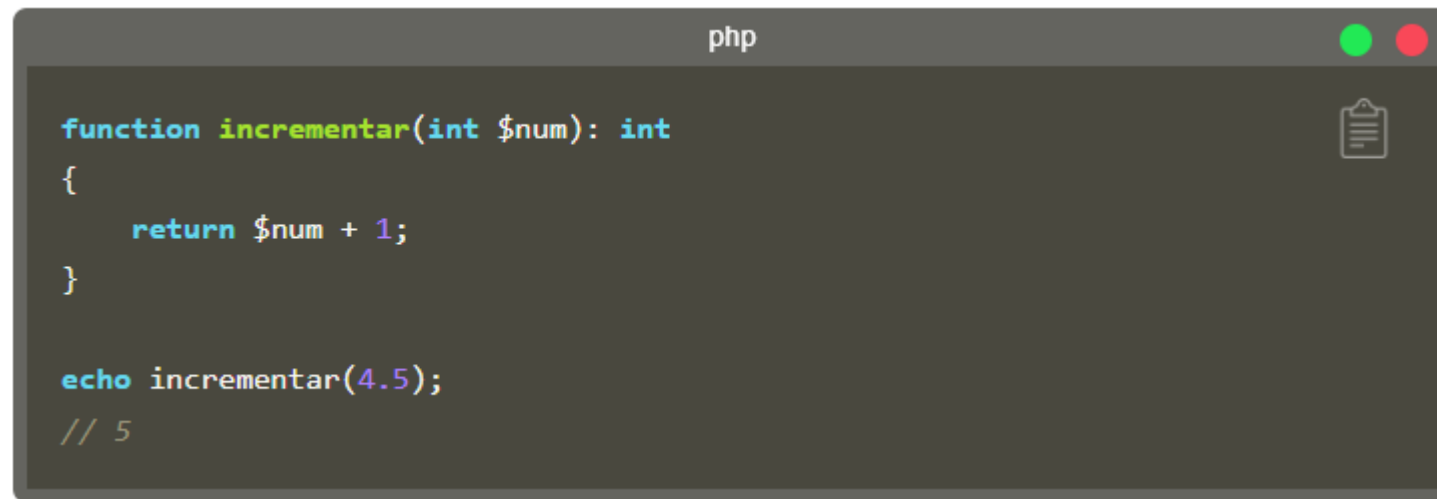

Funciones

- Tipos admitidos

Tipo	Descripción	Versión mínima
string	El parámetro debe ser un string.	PHP 7.0.0
int	El parámetro debe ser un valor de tipo integer.	PHP 7.0.0
float	El parámetro debe ser un número de tipo float.	PHP 7.0.0
bool	El parámetro debe ser un valor de tipo boolean.	PHP 7.0.0
array	El parámetro debe ser un array.	PHP 5.1.0
callable	El parámetro debe ser un callable válido.	PHP 5.4.0
self	El parámetro debe ser una instanceof de la misma clase donde está definido el método. Esto solamente se puede utilizar en clases y métodos de instancia.	PHP 5.0.0
nombre de clase/interfaz	El parámetro debe ser una instanceof del nombre de la clase o interfaz dada.	PHP 5.0.0

Funciones

- Tipos admitidos
 - De forma automática PHP arreglará las incompatibilidades de tipos.



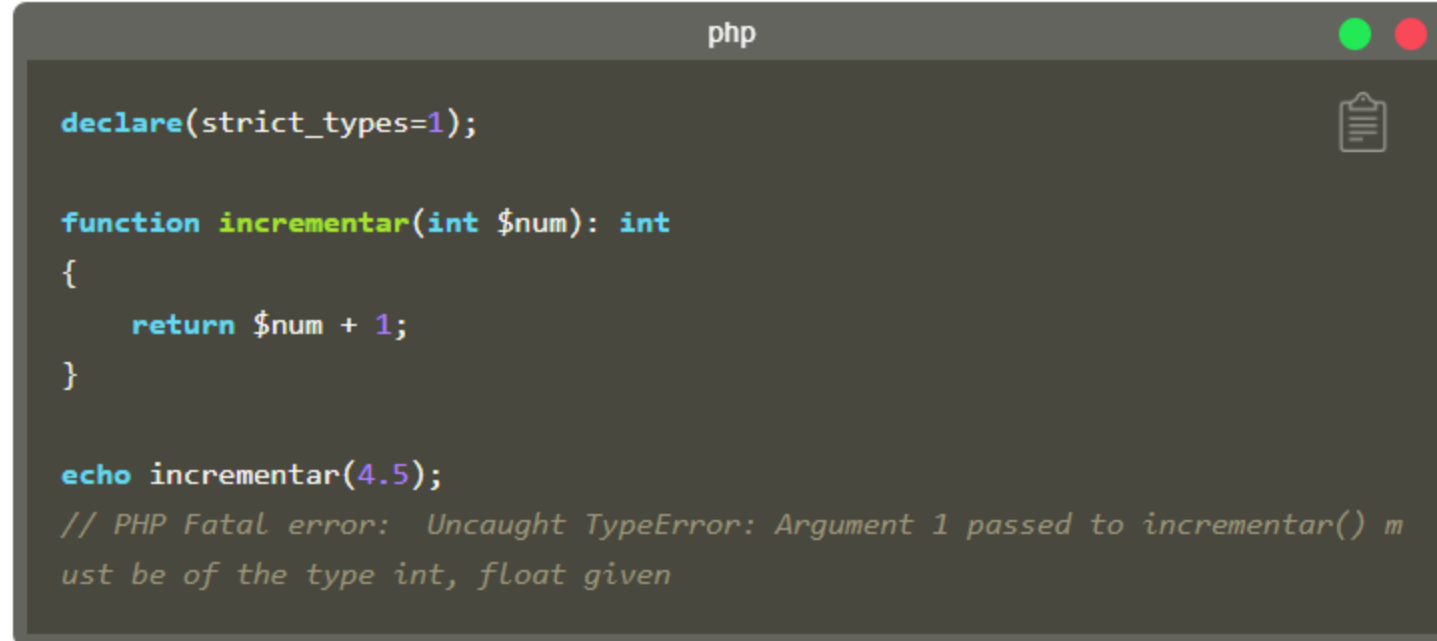
```
function incrementar(int $num): int
{
    return $num + 1;
}

echo incrementar(4.5);
// 5
```

The image shows a code editor window titled 'php' with a dark background. It contains PHP code that defines a function 'incrementar' which takes an integer parameter '\$num' and returns an integer. The function body simply returns '\$num + 1'. Below the function definition, there is a call to 'echo incrementar(4.5);' followed by a comment '// 5'. The code is color-coded: 'function' is blue, 'incrementar' is green, 'int' is blue, '\$num' is purple, 'return' is blue, '\$num + 1' is purple, 'echo' is blue, and '4.5' is purple. The comment '// 5' is in a lighter color. A clipboard icon is visible in the top right corner of the editor window.

Funciones

- Tipos admitidos
 - Pero si quieres ser estricto deberás desactivar esta ayuda. En otras palabras, que si encuentra algún problema de tipos muestre un error y no aplique una mágica solución.

A screenshot of a terminal window titled 'php' with standard macOS window controls (green, yellow, red buttons). The terminal displays PHP code with syntax highlighting. The code defines a function 'incrementar' that takes an integer and returns an integer. It then attempts to call this function with the float value 4.5. The output shows a fatal error: 'Uncaught TypeError: Argument 1 passed to incrementar() must be of the type int, float given'.

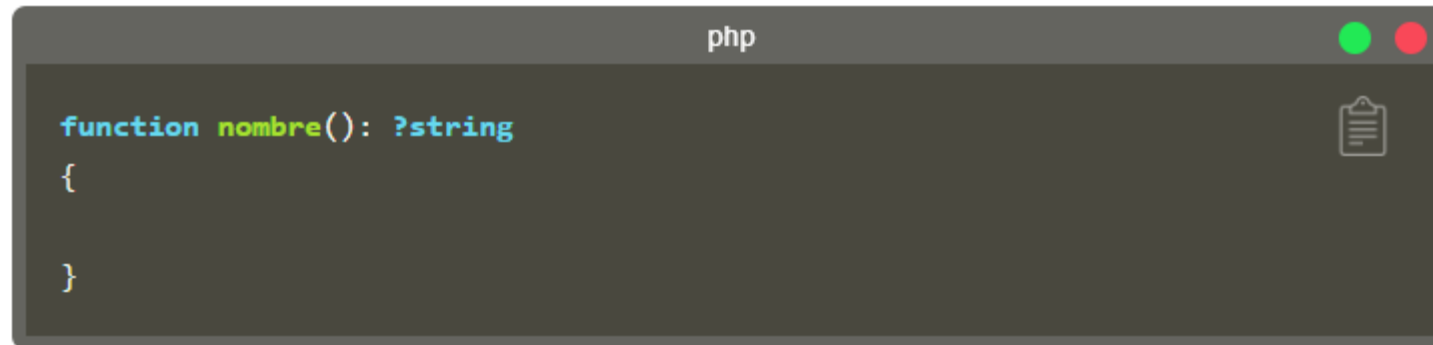
```
declare(strict_types=1);

function incrementar(int $num): int
{
    return $num + 1;
}

echo incrementar(4.5);
// PHP Fatal error:  Uncaught TypeError: Argument 1 passed to incrementar() must be of the type int, float given
```

Funciones

- Return con tipos alternativos
 - A partir de la versión 7.1 de PHP disponemos de la posibilidad de indicar si un return devuelve un tipo concreto o un null. Para ello solo habrá que añadir un interrogante en su inicio.



```
function nombre(): ?string
{

}
```

The image shows a code editor window titled 'php' with a dark background. It contains a PHP function definition: `function nombre(): ?string` followed by an opening curly brace `{` and a closing curly brace `}`. The text is color-coded: `function` is blue, `nombre()` is green, `:` is blue, `?string` is blue, and the braces are white. A clipboard icon is visible in the top right corner of the editor area.

- En el siguiente ejemplo podremos devolver un string o un null. Dependiendo de si el ganador esta entre los 3 primeros o no.

Funciones

- Return con tipos alternativos

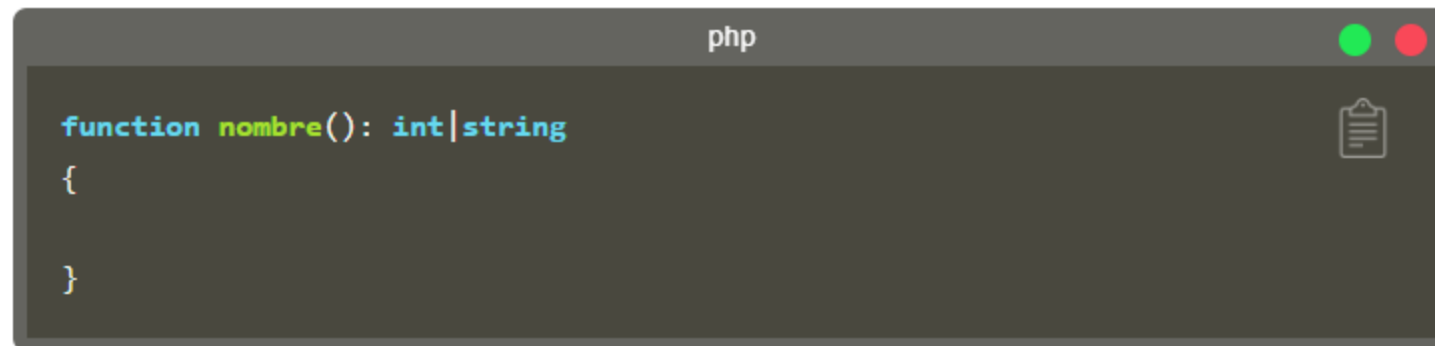
```
php
/**
 * Método que indica el tipo de metal que debe tener una medalla a partir del
 resultado
 * @param {int} $posicion - Posición
 * @return {string|null} - Tipo de metal
 */
function tipoDeMedalla(int $posicion): ?string
{
    switch ($posicion) {
        case 1:
            return 'Oro';
        case 2:
            return 'Plata';
        case 3:
            return 'Bronce';
        default:
            return null;
    }
}

echo tipoDeMedalla(2);
// Plata

echo tipoDeMedalla(34);
// null
```

Funciones

- Return con tipos alternativos
 - Tal vez fue añadido esta característica por lo práctico que resulta al realizar testing.
 - Y de la versión PHP 8 se enriquece más las posibilidades ya que es posible indicar 2 tipos diferentes.



```
function nombre(): int|string
{

}
```

The image shows a code editor window titled 'php' with a dark background. It contains a PHP function definition: `function nombre(): int|string` followed by an opening curly brace, a blank line, and a closing curly brace. A clipboard icon is visible in the top right corner of the editor area.

Funciones

- Return con tipos alternativos
 - Nos puede ayudar en casos como, por ejemplo, dar un resultado alternativo o de seguridad.

```
php

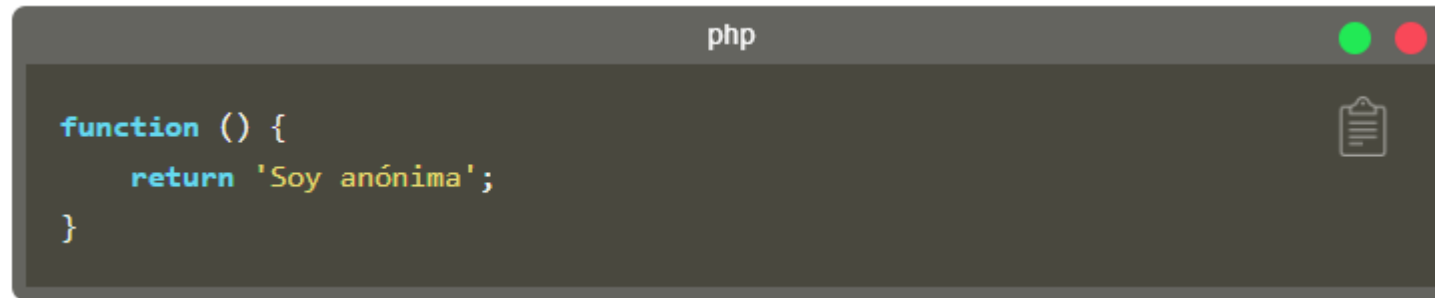
/**
 * Método que duplica un número en positivo
 * @param {int} $numero - Número a duplicar
 * @return {float|string} - Resultado o mensaje de ayuda
 */
function duplicarPositivo(float $numero): float|string
{
    if ($numero > 0) {
        return $numero * 2;
    } else {
        return 'No puedes usar número en negativo o cero';
    }
}

echo duplicarPositivo(12.1);
// 24.2

echo duplicarPositivo(-45);
// 'No puedes usar número en negativo o cero'
```

Funciones

- Anónimas
 - Las funciones anónimas son cerradas y pueden ser declaradas sin ningún nombre. Son obligatorias cuando tengamos que pasar una función como un parámetro de otra.

A code editor window with a dark background and a title bar that says 'php'. The window contains a PHP code snippet for an anonymous function. The code is:

```
function () {  
    return 'Soy anónima';  
}
```

 The text is color-coded: 'function' is blue, '()' is light blue, '{' is light blue, 'return' is blue, 'Soy anónima' is yellow, and ';' is light blue. There is a clipboard icon in the top right corner of the code area.

Funciones

- Anónimas
 - En el siguiente ejemplo incrementamos en 1 cada número del array.

```
php


$numeros = [10, 20, 30, 40];
$numerosIncrementados = array_map(function ($numero) {
    return $numero + 1;
}, $numeros);

var_dump($numerosIncrementados);

/*
array(4) {
    [0]=>
    int(11)
    [1]=>
    int(21)
    [2]=>
    int(31)
    [3]=>
    int(41)
}
*/
```

Funciones

- Usar variables externas
 - Si vas a usar variables que están presentes en tu código, puedes enriquecer el contenido de la función usando use.

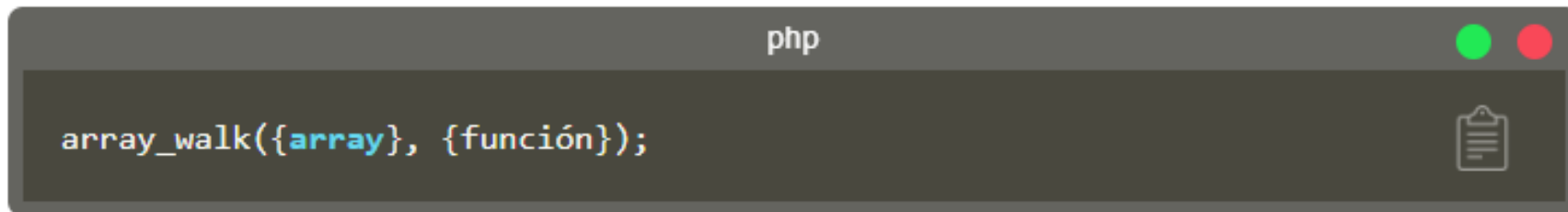


```
$tienda = 'pescadería';

function () use ($tienda) {
    return "Estoy en la $tienda";
}
```

Funciones

- Paradigma Funcional
 - Las funciones esenciales para iterar y gestionar un array son: `array_walk`, `array_filter`, `array_map` y `array_reduce`.
- `array_walk` (Iterar)
 - Recorre un array, similar a un `foreach`.



```
array_walk({array}, {función});
```

The image shows a code editor window with a dark background. The title bar at the top is labeled 'php' and has standard macOS window control buttons (red, yellow, green) on the right. The main area contains the PHP function signature `array_walk({array}, {función});` in a light-colored monospace font. A small clipboard icon is visible in the bottom right corner of the editor area.

Funciones

- array_walk (Iterar)
 - En este ejemplo vamos a imprimir todas las ciudades.

```
<?php

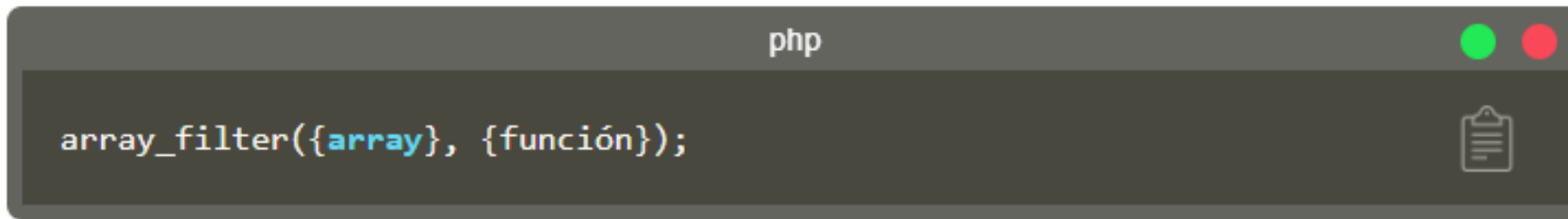
// Diccionario
$apartamentos = [
    [
        'precio/noche' => 40,
        'ciudad' => 'Valencia',
        'wifi' => True,
        'pagina web' => 'https://hotel.com'
    ],
    [
        'precio/noche' => 87,
        'ciudad' => 'Calpe',
        'wifi' => True,
        'pagina web' => 'https://calpe.com'
    ],
    [
        'precio/noche' => 67,
        'ciudad' => 'Valencia',
        'wifi' => False,
        'pagina web' => 'https://denia.com'
    ],
    [
        'precio/noche' => 105,
        'ciudad' => 'Benidorm',
        'wifi' => False,
        'pagina web' => 'https://benidorm.com'
    ]
];

array_walk($apartamentos, function ($apartamento, $posicion) {
    echo $apartamento['ciudad'] . PHP_EOL;
});

/*
Valencia
Calpe
Valencia
Benidorm
*/
```

Funciones

- array_filter (Filtrar)
 - Obtenemos un array, otro más pequeño.



```
array_filter({array}, {función});
```

A code editor window with a dark gray background and a title bar labeled 'php'. The window contains a single line of PHP code: `array_filter({array}, {función});`. The word `array` is highlighted in blue. To the right of the code, there is a small icon of a clipboard with a list, indicating a copy or paste action.

Funciones

- array_filter (Filtrar)
 - En este ejemplo filtraremos \$apartamentos para quedarnos con los que están en Valencia.

```
<?php

// Diccionario
$apartamentos = [
    [
        'precio/noche' => 40,
        'ciudad' => 'Valencia',
        'wifi' => True,
        'pagina web' => 'https://hotel.com'
    ],
    [
        'precio/noche' => 87,
        'ciudad' => 'Calpe',
        'wifi' => True,
        'pagina web' => 'https://calpe.com'
    ],
    [
        'precio/noche' => 67,
        'ciudad' => 'Valencia',
        'wifi' => False,
        'pagina web' => 'https://denia.com'
    ],
    [
        'precio/noche' => 105,
        'ciudad' => 'Benidorm',
        'wifi' => False,
        'pagina web' => 'https://benidorm.com'
    ]
];
```

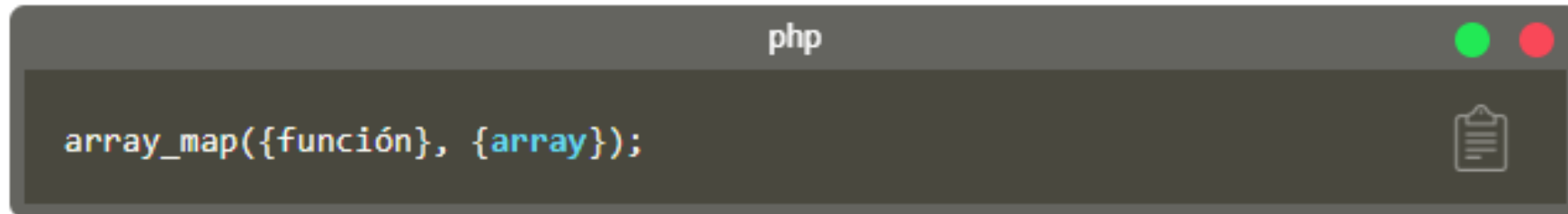
```
$todosLosApartamentosValencia = array_filter($apartamentos, function ($apartamento) {
    return $apartamento['ciudad'] === 'Valencia';
});

var_dump($todosLosApartamentosValencia);

/*
array(2) {
    [0]=>
    array(4) {
        ["precio/noche"]=>
        int(40)
        ["ciudad"]=>
        string(8) "Valencia"
        ["wifi"]=>
        bool(true)
        ["pagina web"]=>
        string(17) "https://hotel.com"
    }
    [2]=>
    array(4) {
        ["precio/noche"]=>
        int(67)
        ["ciudad"]=>
        string(8) "Valencia"
        ["wifi"]=>
        bool(false)
        ["pagina web"]=>
        string(17) "https://denia.com"
    }
}
*/
```

Funciones

- array_map (Modificar)
 - Transforma el contenido de un array, pero mantiene el número de elementos.



```
array_map({función}, {array});
```

A code editor window with a dark gray background and a title bar labeled 'php'. The window contains a single line of PHP code: `array_map({función}, {array});`. The word 'función' is in orange and 'array' is in blue. On the right side of the window, there are three colored circles (green, yellow, red) and a clipboard icon.

- En este ejemplo vamos a reducir el precio por noche en 1.

Funciones

- array_map (Modificar)

```
php
<?php

// Diccionario
$apartamentos = [
    [
        'precio/noche' => 40,
        'ciudad' => 'Valencia',
        'wifi' => True,
        'pagina web' => 'https://hotel.com'
    ],
    [
        'precio/noche' => 87,
        'ciudad' => 'Calpe',
        'wifi' => True,
        'pagina web' => 'https://calpe.com'
    ],
    [
        'precio/noche' => 67,
        'ciudad' => 'Valencia',
        'wifi' => False,
        'pagina web' => 'https://denia.com'
    ],
    [
        'precio/noche' => 105,
        'ciudad' => 'Benidorm',
        'wifi' => False,
        'pagina web' => 'https://benidorm.com'
    ]
];
```

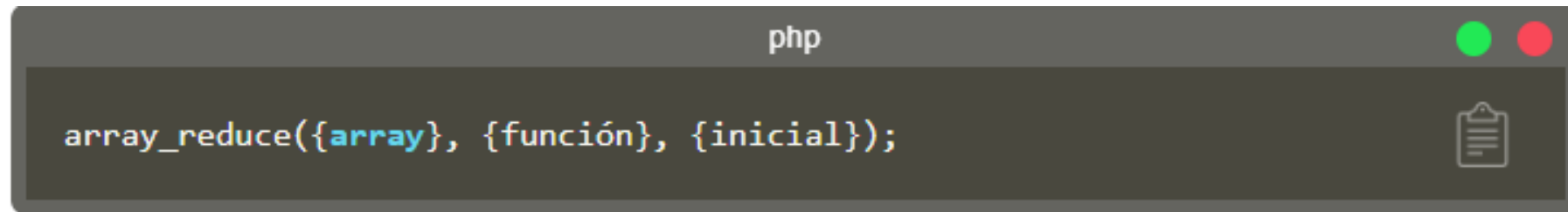
```
$apartamentosMasBaratos = array_map(function ($apartamento) {
    return array_merge($apartamento, ['precio/noche' => $apartamento['precio/
noche'] - 1]);
}, $apartamentos);

var_dump($apartamentosMasBaratos);
/*
array(4) {
    [0]=>
    array(4) {
        ["precio/noche"]=>
        int(39)
        ["ciudad"]=>
        string(8) "Valencia"
        ["wifi"]=>
        bool(true)
        ["pagina web"]=>
        string(17) "https://hotel.com"
    }
    [1]=>
```

```
[1]=>
array(4) {
    ["precio/noche"]=>
    int(86)
    ["ciudad"]=>
    string(5) "Calpe"
    ["wifi"]=>
    bool(true)
    ["pagina web"]=>
    string(17) "https://calpe.com"
}
[2]=>
array(4) {
    ["precio/noche"]=>
    int(66)
    ["ciudad"]=>
    string(8) "Valencia"
    ["wifi"]=>
    bool(false)
    ["pagina web"]=>
    string(17) "https://denia.com"
}
[3]=>
array(4) {
    ["precio/noche"]=>
    int(104)
    ["ciudad"]=>
    string(8) "Benidorm"
    ["wifi"]=>
    bool(false)
    ["pagina web"]=>
    string(20) "https://benidorm.com"
}
}
*/
```


Funciones

- array_reduce (Calcular)
 - Obtiene un resultado a partir de un array.



```
array_reduce({array}, {función}, {inicial});
```

The image shows a code editor window with a dark gray background. The title bar at the top is labeled 'php' and has two window control buttons (green and red) on the right. The main area of the editor displays the PHP function signature `array_reduce({array}, {función}, {inicial});` in a light gray monospace font. A small clipboard icon is visible in the bottom right corner of the editor area.

Funciones

- array_reduce (Calcular)
 - En este ejemplo vamos a calcular cual es la media del precio por noche.

```
<?php

// Diccionario
$apartamentos = [
    [
        'precio/noche' => 40,
        'ciudad' => 'Valencia',
        'wifi' => True,
        'pagina web' => 'https://hotel.com'
    ],
    [
        'precio/noche' => 87,
        'ciudad' => 'Calpe',
        'wifi' => True,
        'pagina web' => 'https://calpe.com'
    ],
    [
        'precio/noche' => 67,
        'ciudad' => 'Valencia',
        'wifi' => False,
        'pagina web' => 'https://denia.com'
    ],
    [
        'precio/noche' => 105,
        'ciudad' => 'Benidorm',
        'wifi' => False,
        'pagina web' => 'https://benidorm.com'
    ]
];

$media = array_reduce($apartamentos, function ($acumulador, $apartamento) {
    return $apartamento['precio/noche'] + $acumulador;
}, 0) / count($apartamentos);

echo $media;

// 74.75
```

Formularios

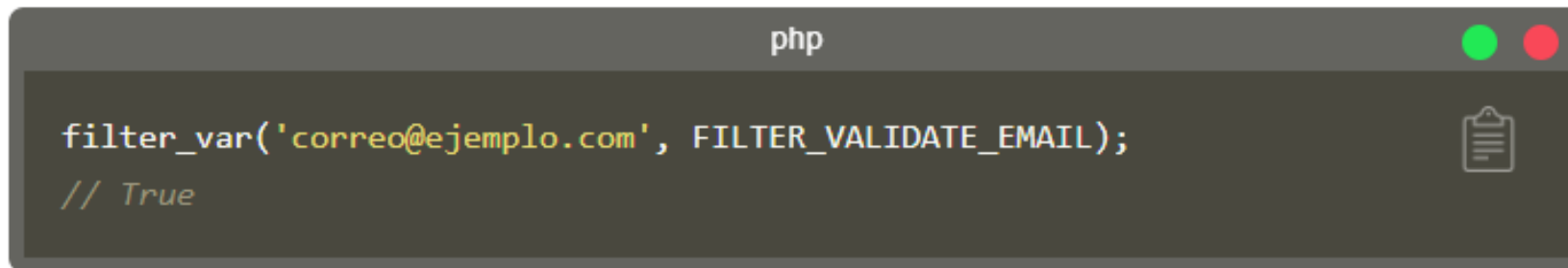
- Una de las tareas más laboriosas es validar una información que llega de un formulario.
- Es peligroso meter cualquier cosa que llegue a nuestra Base de Datos. El usuario es torpe o malintencionado.
 - Introducir e-mails sin sentidos.
 - Letras en lugar de números (por ejemplo, al pedir la edad).
 - Dejar campos vacíos cuando deben ser obligatorios.
 - Dar malos formatos. (por ejemplo, en un número de teléfono).
 - Una longitud muy corta o muy larga de un texto.
 - Código mal intencionado.
 - Y largo etcétera.

Formularios

- No existe una única manera de validar, cada programador tiene su método. Pero lo que siempre hay que realizar son unos pasos estrictos:
 - Enviar datos desde nuestro formulario.
 - Recoger los datos.
 - Validar cada campo.
 - Mostrar al usuario los errores con un mensaje.
 - Si existen errores, mantenerse en la página.
 - Si no existen errores, generar la acción que busques e informar al usuario del éxito.

Formularios

- Aunque valides en Javascript (frontend) debemos validar con PHP (backend). Los datos pueden ser alterados desde el navegador. ¡Nunca te fíes del usuario!
- Nativamente PHP nos proporciona una función llamada `filter_var`.



```
php

filter_var('correo@ejemplo.com', FILTER_VALIDATE_EMAIL);
// True
```

Formularios

- Aunque valides en Javascript (frontend) debemos validar con PHP (backend). Los datos pueden ser alterados desde el navegador. ¡Nunca te fíes del usuario!
- Nativamente PHP nos proporciona una función llamada `filter_var`.
- A continuación, puedes ver analizar un ejemplo completo y real donde se ha validado cada campo y se informa al usuario en caso de encontrarse cualquier problema.

Formularios

- A continuación, puedes ver y analizar un ejemplo completo y real donde se ha validado cada campo y se informa al usuario en caso de encontrarse cualquier problema.

```
php
<html>
  <body>
    <?php
      //=====
      // PROCESAR FORMULARIO
      //=====
      // Comprobamos si nos llega los datos por POST
      if ($_SERVER['REQUEST_METHOD'] == 'POST') {
        //-----
        // Funciones Para Validar
        //-----

        /**
         * Método que valida si un texto no esta vacío
         * @param {string} - Texto a validar
         * @return {boolean}
         */
        function validar_requerido(string $texto): bool
        {
          return !(trim($texto) == '');
        }
      }
    }
  }
</body>
</html>
```

Formularios

```
/**
 * Método que valida si es un número entero
 * @param {string} - Número a validar
 * @return {bool}
 */
function validar_entero(string $numero): bool
{
    return filter_var($numero, FILTER_VALIDATE_INT);
}

/**
 * Método que valida si el texto tiene un formato válido de E
-Mail
 * @param {string} - Email
 * @return {bool}
 */
function validar_email(string $texto): bool
{
    return filter_var($texto, FILTER_VALIDATE_EMAIL);
}
```


Formularios

```
//-----  
// Variables  
//-----  
$errores = [];  
$nombre = isset($_REQUEST['nombre']) ? $_REQUEST['nombre'] :  
null;  
  
$edad = isset($_REQUEST['edad']) ? $_REQUEST['edad'] : null;  
$email = isset($_REQUEST['email']) ? $_REQUEST['email'] : nul  
1;  
  
//-----  
// Validaciones  
//-----  
// Nombre  
if (!validar_requerido($nombre)) {  
    $errores[] = 'El campo Nombre es obligatorio.';  
}  
// Edad  
if (!validar_entero($edad)) {  
    $errores[] = 'El campo de Edad debe ser un número.';  
}  
// Email  
if (!validar_email($email)) {  
    $errores[] = 'El campo de Email tiene un formato no válid  
o.';  
}
```

Formularios

```
<!-- Botón submit -->
<input type="submit" value="Enviar">
</p>
</form>
</body>
</html>
```

```
//-----
// Lógica
//-----
if (!isset($errores)) {
    // Enviamos el correo
}

}

?>
<!-- Mostramos errores por HTML -->
<?php if (isset($errores)): ?>
<ul class="errores">
    <?php foreach ($errores as $error): ?>
        <li><?= $error ?></li>;
    <?php endforeach; ?>
</ul>
<?php endif; ?>
<!-- Formulario -->
<form method="post">
    <p>
        <!-- Campo nombre -->
        <input type="text" name="nombre" placeholder="Nombre">
    </p>
    <p>
        <!-- Campo edad -->
        <input type="text" name="edad" placeholder="Edad">
    </p>
    <p>
        <!-- Campo Email -->
        <input type="text" name="email" placeholder="Email">
    </p>
    <p>
```

Formularios

- Actividad 22 – Calculadora de newsletter
 - Vamos a realizar un sistema que nos calcule el precio de un servicio de newsletter. Dependiendo del número de emails que enviemos costará un precio u otro. A continuación, puedes ver una tabla.

De	A	Precio
0	2000	0 €
2001	10000	0.7 € unidad
10001	Infinito	0.2 € unidad

- Realiza los siguientes pasos:
 - Añade un campo para indicar el número de emails a enviar. Comprueba que es un número.
 - Añade una opción para indicar si quieres un seguro por cada mensaje, lo cual tendrá un recargo por mensaje de 0.1 €.
 - Al pulsar en submit muestra el precio total.

Formularios

- Actividad 23 – Reserva de apartamentos
 - El objetivo será crear diversos buscadores para encontrar nuestro apartamento ideal.
 - Guarda en un diccionario varios datos con la siguiente estructura.
 - Precio/noche. Comprueba que es un número.
 - Ciudad. Comprueba que es un texto.
 - Wifi. Comprueba que existe.
 - Página web. Comprueba que es un dominio válido.
 - Por ejemplo. ->
 - Realiza un formulario diferente por cada campo. Imprime los resultados de una manera bonita y humana.
 - Pro:
 - Calcula el precio medio de los resultado (puedes usar `array_reduce()`).

```
php
$apartamentos =[
    [
        'precio/noche' => 37,
        'ciudad' => 'Valencia',
        'wifi' => True,
        'pagina web' => 'https://hotel.com'
    ],
    [
        'precio/noche' => 87,
        'ciudad' => 'Madrid',
        'wifi' => False,
        'pagina web' => 'https://motel.es'
    ],
    ...
];
```

Ficheros

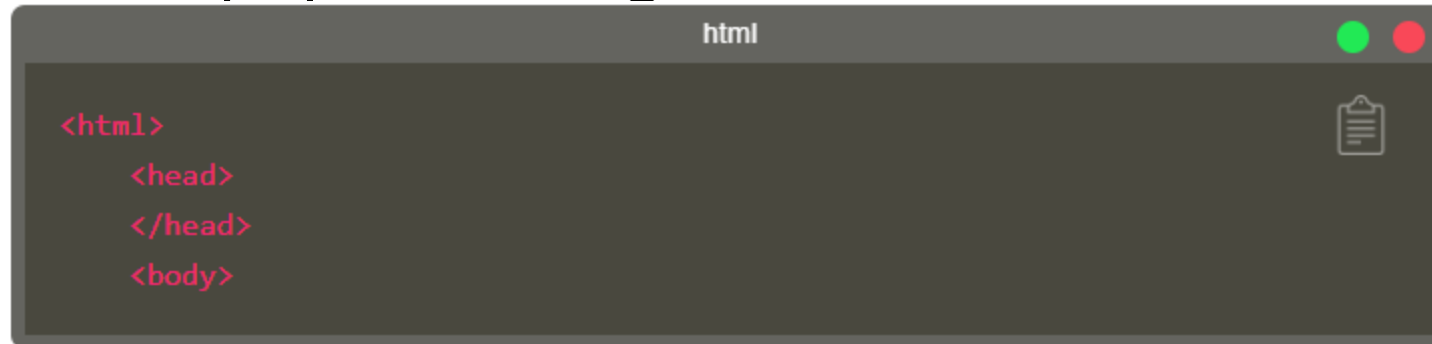
- Es posible importar o llamar el código desde otro archivo con PHP.
- De esta manera conseguiremos dividir nuestras páginas dinámicas en fragmentos más pequeños y fáciles de gestionar evitando el famoso código Spaghetti (escribir todo nuestro código en un solo fichero con un interminable número de líneas).
- Para esta buena práctica disponemos en nuestro bat-cinturón de hasta 4 herramientas diferentes.

Ficheros

Función	Descripción	Caso de error	Ejemplo
include "	Incluye el fichero en cada ocasión.	Da una advertencia, pero continua la ejecución.	include 'tu_archivo.php'
include_once "	Incluye el fichero en una ocasión.	Da una advertencia, pero continua la ejecución.	include_once 'tu_archivo.php'
require "	Incluye el fichero en cada ocasión.	Para la ejecución (Error fatal).	require 'tu_archivo.php'
require_once "	Incluye el fichero en una ocasión.	Para la ejecución (Error fatal).	require_once 'tu_archivo.php'

Ficheros

- Veamos con un ejemplo cómo funciona. Voy a tener un archivo llamado header.php con el siguiente contenido.

A code editor window titled 'html' with a dark background. It contains the following HTML code in red text: <html>, <head>, </head>, and <body>. A clipboard icon is visible in the top right corner of the editor area.

```
<html>
  <head>
</head>
  <body>
```

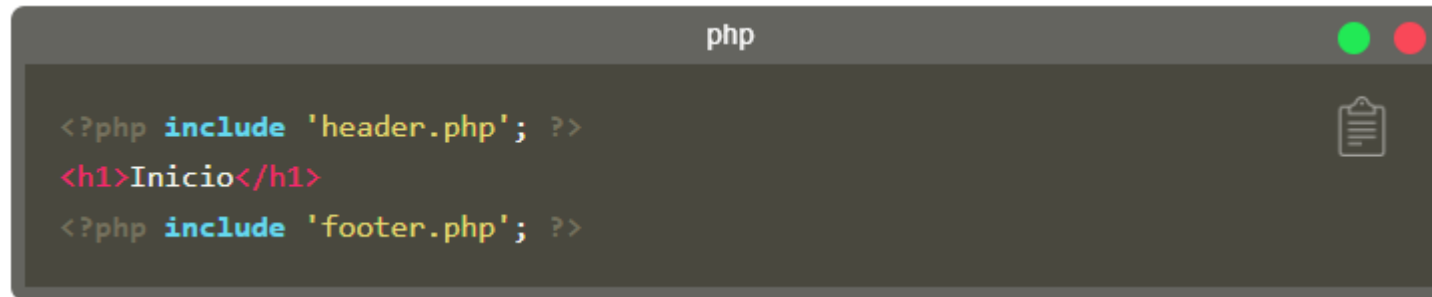
- Otro con el nombre footer.php.

A code editor window titled 'html' with a dark background. It contains the following HTML code in red text: <footer>Soy yo</footer>, </body>, and </html>. A clipboard icon is visible in the top right corner of the editor area.

```
    <footer>Soy yo</footer>
  </body>
</html>
```

Ficheros

- Ahora creo un fichero nuevo.



```
<?php include 'header.php'; ?>
<h1>Inicio</h1>
<?php include 'footer.php'; ?>
```

- Me daría el siguiente HTML.

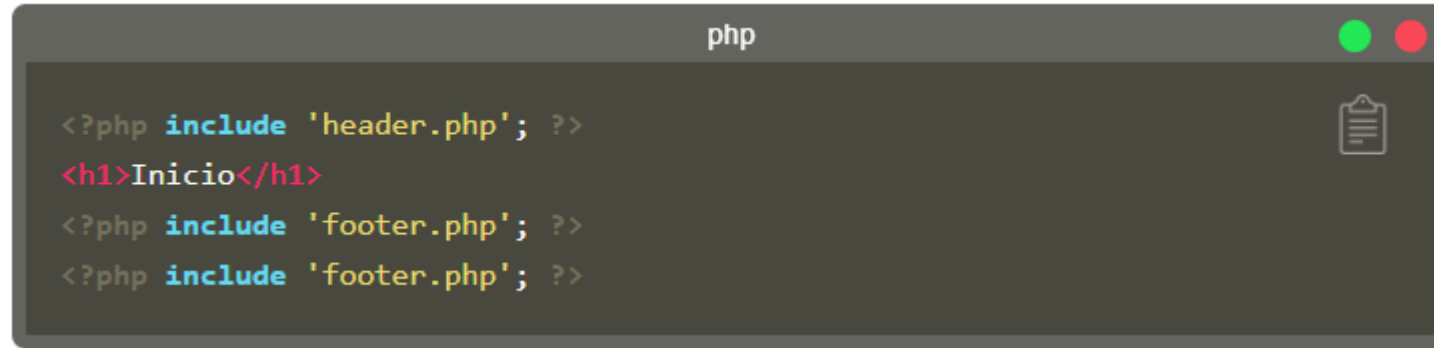


```
<html>
  <head>
  </head>
  <body>
    <h1>Inicio</h1>
    <footer>Soy yo</footer>
  </body>
</html>
```


Ficheros

- Potente, ¿verdad? Evitaremos repartir partes de nuestro código que sean muy reiterativas.
- En esta ocasión voy a repetir el include de footer.
- Ahora creo un fichero nuevo.

Ficheros



```
<?php include 'header.php'; ?>
<h1>Inicio</h1>
<?php include 'footer.php'; ?>
<?php include 'footer.php'; ?>
```

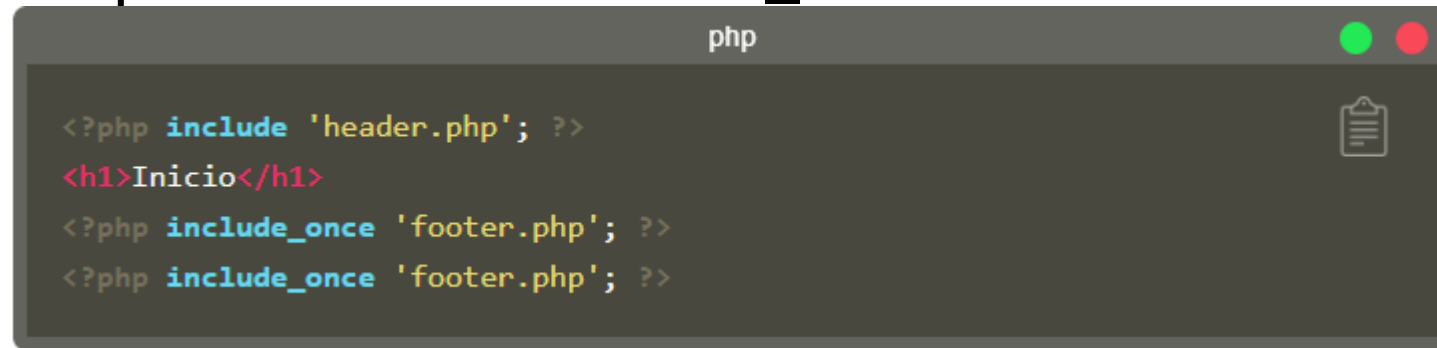
- Lo que me generaría un pie duplicado.



```
<html>
  <head>
  </head>
  <body>
    <h1>Inicio</h1>
    <footer>Soy yo</footer>
  </body>
</html>
  <footer>Soy yo</footer>
</body>
</html>
```

Ficheros

- Lo podemos prevenir con `include_once`.



```
<?php include 'header.php'; ?>
<h1>Inicio</h1>
<?php include_once 'footer.php'; ?>
<?php include_once 'footer.php'; ?>
```

- Si ya ha sido llamado, lo ignora.



```
<html>
  <head>
  </head>
  <body>
    <h1>Inicio</h1>
    <footer>Soy yo</footer>
  </body>
</html>
```

Ficheros

- Subir un archivo
 - Un fichero es un elemento en binario que no es ni un número o ni un texto: imagen, video, música, doc, iso...
 - Nosotros somos incapaces de leerlo a no ser que tengamos un ojo biónico y un chip en el cerebro. Si careces de estos dos requisitos solo podrás subirlo, por medio de un formulario, y almacenarlo en una carpeta.
 - Se debe tratar de una forma especial. Necesitaremos usar siempre el method POST y añadir enctype="multipart/form-data". Por último, usar el input de tipo archivo (file).

Ficheros

- Subir un archivo

```
html
<!-- Formulario -->
<form method="post" enctype="multipart/form-data">
  <p>
    <!-- Campo imagen -->
    <input type="file" name="fichero_usuario">
  </p>
  <p>
    <!-- Botón submit -->
    <input type="submit" value="Enviar">
  </p>
</form>
```

- Cuando nuestro formulario sea enviado el archivo estará almacenado en una variable llamada `$_FILES`. La cual es un array con toda la información que vas a necesitar.

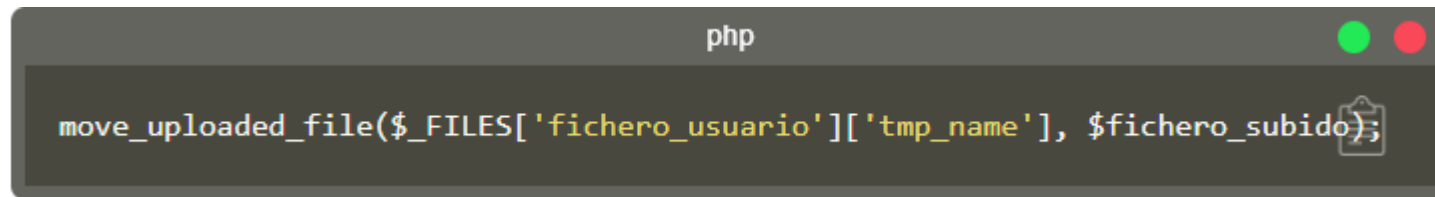
Ficheros

- Subir un archivo

Nombre	Ejemplo de contenido	Descripción
<code>\$_FILES['fichero_usuario']['name']</code>	<code>'foto_en_la_playa.jpg'</code>	Nombre del archivo
<code>\$_FILES['fichero_usuario']['type']</code>	<code>'image/png'</code>	MIME (formato del archivo)
<code>\$_FILES['fichero_usuario']['size']</code>	3232424	Tamaño en bytes (5MB -> 5 x 1024 x 1024 bytes)
<code>\$_FILES['fichero_usuario']['error']</code>	0	Código de error. El 0 es que todo ha ido bien, los otros puedes encontrarlos aquí
<code>\$_FILES['fichero_usuario']['tmp_name']</code>	213	Nombre temporal

Ficheros

- Subir un archivo
 - Ahora solo tendremos que moverlo de la carpeta temporal a la definitiva, usando el método `move_uploaded_file()`.



```
move_uploaded_file($_FILES['fichero_usuario']['tmp_name'], $fichero_subido);
```

- Aquí puedes ver un ejemplo completo.

Ficheros

- Subir un archivo

```
<!-- Formulario -->
<form method="post" enctype="multipart/form-data">
  <p>
    <!-- Campo imagen -->
    <input type="file" name="fichero_usuario">
  </p>
  <p>
    <!-- Botón submit -->
    <input type="submit" value="Enviar">
  </p>
</form>
</body>
</html>
```

```
<html>
  <body>
    <?php
      //=====
      // PROCESAR IMAGEN
      //=====

      // Comprobamos si nos llega los datos por POST
      if ($_SERVER['REQUEST_METHOD'] == 'POST') {
        // Definir directorio donde se guardará
        $dir_subida = './subidos/';
        // Definir la ruta final del archivo
        $fichero_subido = $dir_subida . basename($_FILES['fichero_usuario']['name']);

        // Mueve el archivo de la carpeta temporal a la ruta definida
        if (move_uploaded_file($_FILES['fichero_usuario']['tmp_name'], $fichero_subido)) {
          // Mensaje de confirmación donde todo ha ido bien
          echo '<p>Se subió perfectamente.</p>';
          // Muestra la imagen que acaba de ser subida
          echo '<p>
        </p>';
        } else {
          // Mensaje de error: ¿Límite de tamaño? ¿Ataque?
          echo '<p>¡Ups! Algo ha pasado.</p>';
        }
      }
    >>
```


Ficheros

- Multiarchivo (subir varios archivos)
 - Es posible subir varios archivos bajo el mismo nombre. Solo habrá que añadir unos corchetes ([]) después del name como si fuera un array.

```
html
<!-- Formulario -->
<form method="post" enctype="multipart/form-data">
  <p>
    <!-- Campos de imágenes -->
    <input type="file" name="imagen[]">
    <input type="file" name="imagen[]">
    <input type="file" name="imagen[]">
  </p>
  <p>
    <!-- Botón submit -->
    <input type="submit" value="Enviar">
  </p>
</form>
```

Ficheros

- Multiarchivo (subir varios archivos)
 - Al recibir los datos tendremos que iterar la variable, igual que un array.
 - Es recomendable comprobar en cada caso que el archivo se ha subido correctamente.

```
php
// Comprobamos si nos llega los datos por POST
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    // Iteramos todos los archivos
    foreach ($_FILES["imagen"]["error"] as $posicion => $error) {
        // Comprobamos si se ha subido correctamente
        if ($error == UPLOAD_ERR_OK) {
            // Definir directorio donde se guardará
            $dir_subida = './subidos/';
            // Definir la ruta final del archivo
            $fichero_subido = $dir_subida . basename($_FILES['imagen']['name'][$posicion]);
            // Mueve el archivo de la carpeta temporal a la ruta definida
            if (move_uploaded_file($_FILES['imagen']['tmp_name'][$posicion], $fichero_subido)) {
                // Mensaje de confirmación donde todo ha ido bien
                echo '<p>Se subió perfectamente' . $_FILES['imagen']['name'][$posicion] . '</p>';
                // Muestra la imagen que acaba de ser subida
                echo '<p></p>';
            } else {
                // Mensaje de error: ¿Límite de tamaño? ¿Ataque?
                echo '<p>¡Ups! Algo ha pasado.</p>';
            }
        }
    }
}
```

Ficheros

```
        // Mensaje de error: ¿Límite de tamaño? ¿Ataque?
        echo '<p>¡Ups! Algo ha pasado.</p>';
    }
}
}
}
?>
<!-- Formulario -->
<form method="post" enctype="multipart/form-data">
    <p>
        <!-- Campos de imágenes -->
        <input type="file" name="imagen[]">
        <input type="file" name="imagen[]">
        <input type="file" name="imagen[]">
    </p>
    <p>
        <!-- Botón submit -->
        <input type="submit" value="Enviar">
    </p>
</form>
</body>
</html>
```

```
<html>
<body>
    <?php
        //=====

        // PROCESAR IMAGENES
        //=====

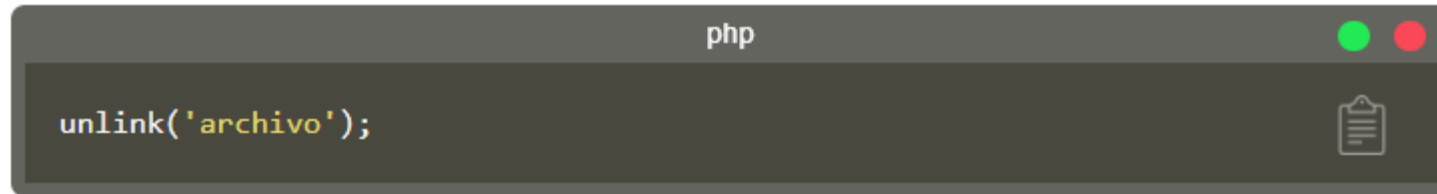
        // Comprobamos si nos llega los datos por POST
        if ($_SERVER['REQUEST_METHOD'] == 'POST') {
            // Iteramos todos los archivos
            foreach ($_FILES["imagen"]["error"] as $posicion => $error) {
                // Comprobamos si se ha subido correctamente
                if ($error == UPLOAD_ERR_OK) {
                    // Definir directorio donde se guardará
                    $dir_subida = './subidos/';
                    // Definir la ruta final del archivo
                    $fichero_subido = $dir_subida . basename($_FILES['imagen'][$posicion]);

                    // Mueve el archivo de la carpeta temporal a la ruta

                    if (move_uploaded_file($_FILES['imagen']['tmp_name'][$posicion], $fichero_subido)) {
                        // Mensaje de confirmación donde todo ha ido bien
                        echo '<p>Se subió perfectamente' . $_FILES['imagen'][$posicion] . '</p>';
                        // Muestra la imagen que acaba de ser subida
                        echo '<p>  
  <p>  
    <!-- Campos de imágenes -->  
    <input type="file" name="imagen[]">  
    <input type="file" name="imagen[]">  
    <input type="file" name="imagen[]">  
  </p>  
  <p>  
    <!-- Botón submit -->  
    <input type="submit" value="Enviar">  
  </p>  
</form>  
</body>  
</html>
```

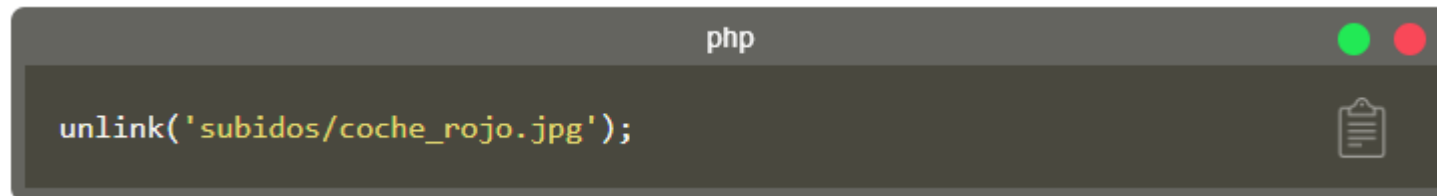
Ficheros

- Borrar archivos
 - Para eliminar un archivo debemos usar el método unlink.



A code editor window titled 'php' with a dark background. It contains the PHP code `unlink('archivo');`. The word 'archivo' is highlighted in yellow. On the right side of the editor, there is a clipboard icon.

- Tan solo hay que dar la ruta que deseamos.



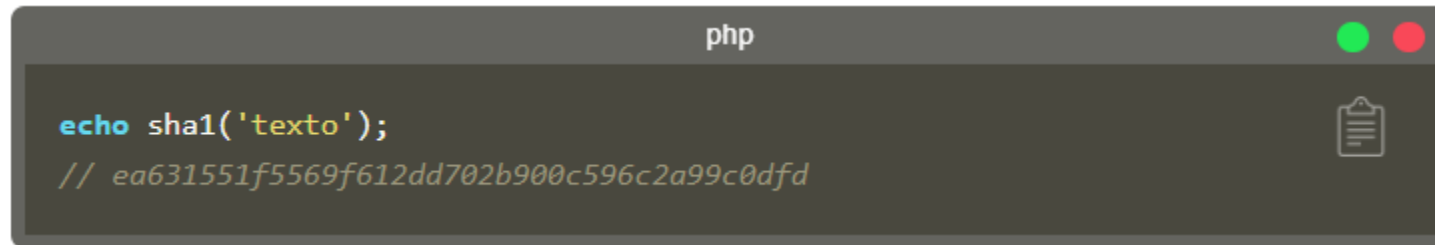
A code editor window titled 'php' with a dark background. It contains the PHP code `unlink('subidos/coche_rojo.jpg');`. The path 'subidos/coche_rojo.jpg' is highlighted in yellow. On the right side of the editor, there is a clipboard icon.

Ficheros

- Evitar que se sobrescriba
 - ¿Qué pasa si subimos dos archivos con el mismo nombre? Pues que el anterior desaparecería, se sobrescribiría por tener el mismo nombre y guardarse en el mismo lugar.
 - Debemos garantizar que el archivo posee un nombre único.
 - Un truco para solucionarlo generando un hash, o una secuencia alfanumérica única por cada archivo que sustituya al nombre. Un algoritmo muy popular es SHA-1.

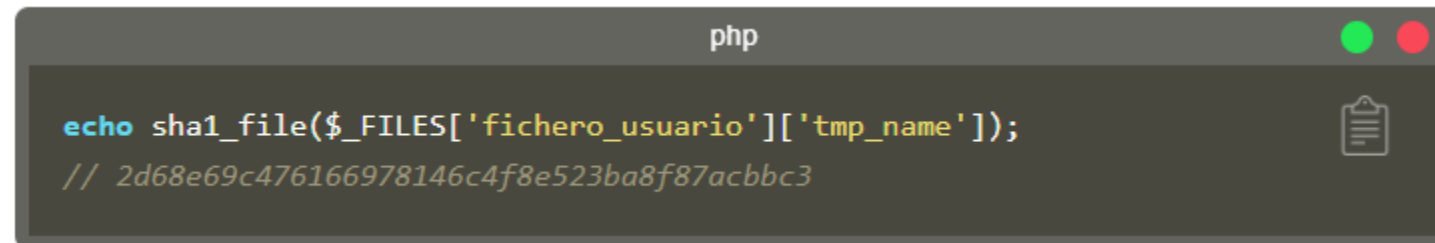
Ficheros

- Evitar que se sobrescriba
 - Si quisiéramos obtener un hash de un texto, deberíamos usar sha1().



```
echo sha1('texto');  
// ea631551f5569f612dd702b900c596c2a99c0dfd
```

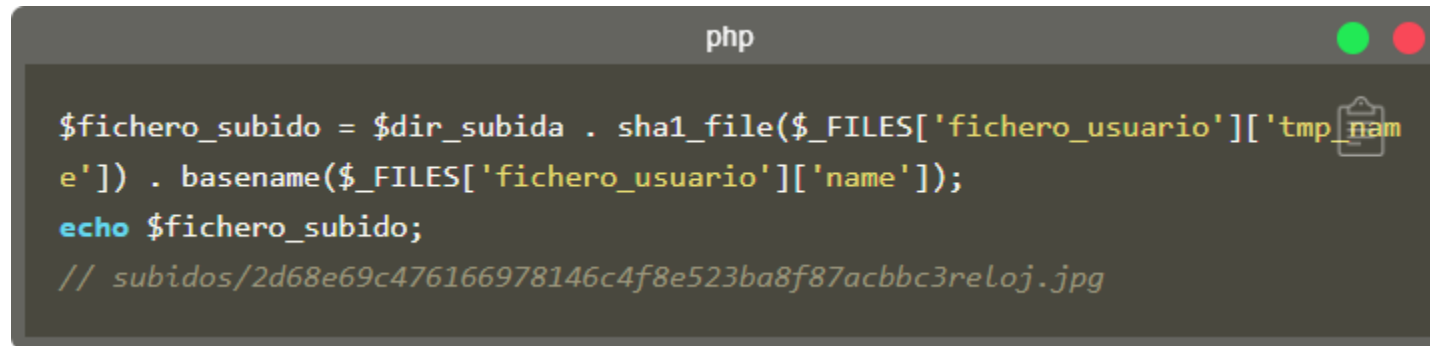
- Para archivos disponemos de una función determinada llamada sha1_file().



```
echo sha1_file($_FILES['fichero_usuario']['tmp_name']);  
// 2d68e69c476166978146c4f8e523ba8f87acbbc3
```

Ficheros

- Evitar que se sobrescriba
 - Si tuviéramos un archivo llamado reloj.jpg.

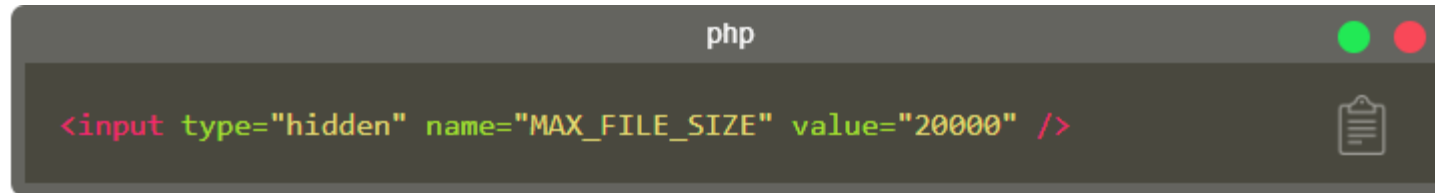


```
php
$archivo_subido = $dir_subida . sha1_file($_FILES['archivo_usuario']['tmp_name']) . basename($_FILES['archivo_usuario']['name']);
echo $archivo_subido;
// subidos/2d68e69c476166978146c4f8e523ba8f87acbbc3reloj.jpg
```

- Por muchos archivos reloj.jpg que suban nunca se sobrescribirán, a no ser que a nivel binario sean exactamente iguales (lo cual tampoco sería un problema porque sería el mismo fichero).

Ficheros

- Tamaño máximo
 - Si quieres limitar el tamaño de todos los archivos puedes hacerlo añadiendo un input especial.

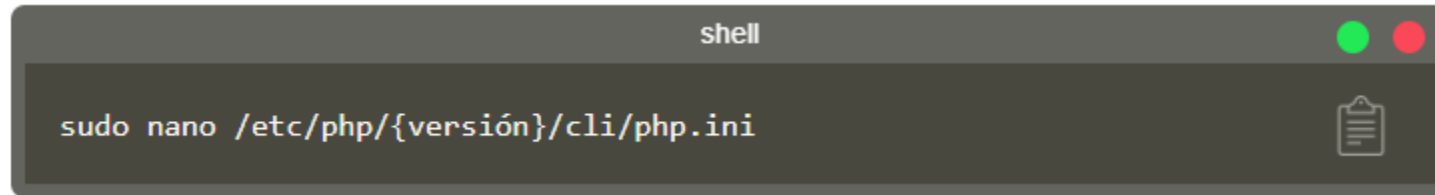
A code editor window with a dark gray background and a title bar that says 'php'. The window has standard macOS window controls (red, yellow, green buttons) in the top right corner. The main area of the window displays the following HTML code: `<input type="hidden" name="MAX_FILE_SIZE" value="20000" />`. The code is color-coded: '<input' is red, 'type="hidden"' is green, 'name="MAX_FILE_SIZE"' is yellow, and 'value="20000" />' is red. A small clipboard icon is visible in the bottom right corner of the code area.

```
<input type="hidden" name="MAX_FILE_SIZE" value="20000" />
```

- El value se mide en bytes.
- Si un archivo supera nuestra frontera dará un error, pero nunca llegará a subirse.
- Otra forma de cambiarlo, en este caso permanente, es modificando unas variables de PHP en su archivo de configuración.

Ficheros

- Tamaño máximo

A terminal window titled 'shell' with standard macOS window controls (green, yellow, red buttons). The command 'sudo nano /etc/php/{versión}/cli/php.ini' is entered in the terminal. A clipboard icon is visible on the right side of the terminal window.

```
shell  
sudo nano /etc/php/{versión}/cli/php.ini
```

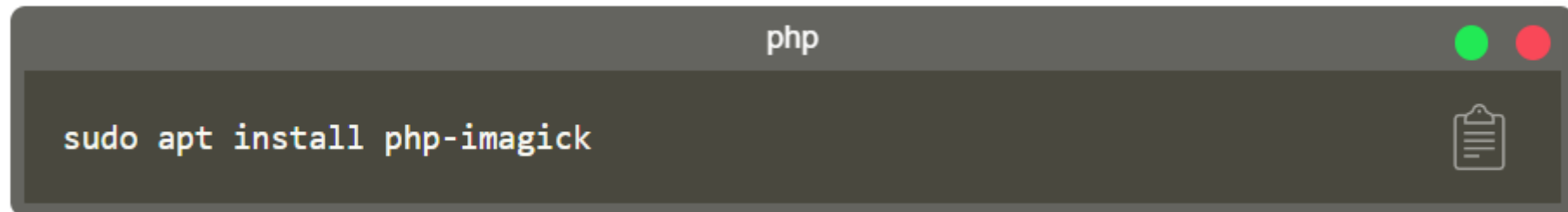
- Edita las siguientes variables si quieres limitarlo a 100Mb.
 - upload_max_filesize=100Mb
 - post_max_size=100Mb
- Igualmente valida el tamaño con PHP, es fácil manipular el límite dentro del navegador. Recuerda: nunca te fíes del usuario.

Ficheros

- Procesamiento de imágenes
 - PHP no esta solamente limitado a la generación de HTML y mover archivos, también puede procesar imágenes.
 - Existen una multitud de posibilidades.
 - Redimensionar (utilizado para crear miniaturas).
 - Recortar.
 - Aplicar filtros de color.
 - Crear imágenes (como suena).
 - Añadir marcas de agua.
 - Cambiar de formato.

Ficheros

- Procesamiento de imágenes
 - Para crear una miniatura podrías hacerlo usando la librería nativa Imagick.
 - Primero tendremos que instalarla en el sistema. Con Ubuntu o Debian es muy sencillo.

A terminal window with a dark gray background. The title bar at the top is dark gray with the text 'php' in white. On the right side of the title bar are three window control buttons: a green circle, a red circle, and a yellow circle. The main area of the terminal is dark gray and contains the text 'sudo apt install php-imagick' in a light gray monospace font. In the bottom right corner of the terminal area, there is a small white icon of a clipboard with a document on it.

```
php
```

```
sudo apt install php-imagick
```

Ficheros

- Procesamiento de imágenes
 - Ahora ya puedes trabajar con ella.
 - En el siguiente ejemplo se captura imagen.jpg y se redimensiona a 100px de ancho. Por último se guarda con el nombre de miniatura.jpg.

```
php

$imagen = new Imagick('imagen.jpg');

// Si se proporciona 0 como parámetro de ancho o alto,
// se mantiene la proporción de aspecto
$imagen->thumbnailImage(100, 0);

// La guarda
file_put_contents('miniatura.jpg', $imagen);
```

Ficheros

- Procesamiento de imágenes
 - Dispone de diversas herramientas para manipular formatos tan conocidos como: JPEG, GIF, PNG y WebP (entre otros).

Ficheros

- Procesamiento de imágenes
 - Aquí tienen un ejemplo completo que recoge todos los casos anteriores.
 - Valida que se ha adjuntado una imagen en el formulario.
 - Valida que sea una imagen en jpg o png.
 - Valida que no supere cierto tamaño. En este caso 2Mb.
 - Crea una miniatura. En este caso de 100px de ancho.
 - Cambia el nombre aleatorio para evitar posibles conflictos con otros archivos.
 - Muestra la miniatura en el HTML.

Ficheros

- Procesamiento de imágenes

```
<?php

//=====
// VARIABLES
//=====

$errorAvatar = 0;
$avatar = null;
$rutaThumbnail = null;

// Definir directorio donde se guardará
define('PATH_AVATAR', './subidos/');
define('PATH_AVATAR_THUMBNAIL', './subidos/thumbnails/');

// Tamaño max avatar: 2 Mb
define('MAX_SIZE_AVATAR_MB', 2);
define('MAX_SIZE_AVATAR', MAX_SIZE_AVATAR_MB * 1024 * 1024);
// En /etc/php/7.4/cli/php.ini edita las siguientes variables
// upload_max_filesize=100Mb
// post_max_size=100Mb

// Anchura miniatura
define('WIDTH_THUMBNAIL', 100);
```


Ficheros

- Procesamiento imágenes

```
php
<?php

//=====
// PROCESAR FORMULARIO
//=====

// Comprobamos si nos llega los datos por POST
if ($_SERVER['REQUEST_METHOD'] == 'POST' && isset($_FILES)) {

    //-----
    // Recoger avatar
    //-----

    // Verifica si existe el directorio, y en caso contrario lo crea
    if (!is_dir(PATH_AVATAR_THUMBNAIL)) {
        mkdir(PATH_AVATAR_THUMBNAIL, 0775, true);
    }

    // Definir la ruta final del archivo
    $nombreFoto = sha1_file($_FILES['avatar']['tmp_name']) . basename($_FILES
['avatar']['name']);
    $ficheroSubido = PATH_AVATAR . $nombreFoto;
```

Ficheros

- Procesamiento imágenes

```
php
//=====
//-----
//  Control de errores
//-----

// Tamaño máximo
if ($_FILES['avatar']['size'] > MAX_SIZE_AVATAR) {
    $errorAvatar = 1;
}

// Solo JPG y PNG
if ($_FILES['avatar']['type'] !== 'image/png' && $_FILES['avatar']['type'] !== 'image/jpeg') {
    $errorAvatar = 2;
}

// Obligatorio
if ($_FILES['avatar']['size'] === 0) {
    $errorAvatar = 3;
}
```

Ficheros

- Procesamiento imágenes

```
//-----  
//  Procesar imagen  
//-----  
  
if ($errorAvatar === 0) {  
    if (move_uploaded_file($_FILES['avatar']['tmp_name'], $ficheroSubido)) {  
        // Mueve el archivo de la carpeta temporal a la ruta definida  
        $avatar = $ficheroSubido;  
  
        // Creamos una miniatura  
        // No olvides instalarlo con: sudo apt install php-imagick  
        $imagen = new Imagick($avatar);  
  
        // Si se proporciona 0 como parámetro de ancho o alto,  
        // se mantiene la proporción de aspecto  
        $imagen->thumbnailImage(WIDTH_THUMBNAIL, 0);  
        $rutaThumbnail = PATH_AVATAR_THUMBNAIL . $nombreFoto;  
        file_put_contents($rutaThumbnail, $imagen);  
    }  
}  
}
```

Ficheros

- Procesamiento imágenes

```
//-----  
//  Procesar imagen  
//-----  
<!doctype html>  
<html lang="es">  
<head>  
    <meta charset="UTF-8">  
    <meta name="viewport"  
        content="width=device-width, user-scalable=no, initial-scale=1.0, m  
aximum-scale=1.0, minimum-scale=1.0">  
    <meta http-equiv="X-UA-Compatible" content="ie=edge">  
    <title>Perfil</title>  
</head>  
<body>  
    <?php if (isset($rutaThumbnail)): ?>  
        ">  
    <?php endif; ?>  
    <form method="post" enctype="multipart/form-data">  
        <p>  
            <label>  
                Foto:  
                <input type="file" name="avatar">  
            </label>  
        </p>  
    </form>  
</body>  
</html>
```

Subid

ida

Ficheros

- Procesamiento imágenes

```
//-----  
// Procesar imagen  
//-----  
<!doctype html>  
<html lang="es">  
  <?php if ($errorAvatar === 1): ?>  
    <p style="color: red">  
      Tamaño demasiado grande, intente que tenga menos de <?= MAX_SIZE_  
AVATAR_MB ?>Mb  
    </p>  
    <?php elseif ($errorAvatar === 2): ?>  
    <p style="color: red">  
      Solo admitido imagenes en JPG o PNG.  
    </p>  
    <?php elseif ($errorAvatar === 3): ?>  
    <p style="color: red">  
      Debes incluir una imagen  
    </p>  
    <?php endif; ?>  
    <p>  
      <input type="submit" value="Guardar">  
    </p>  
  </form>  
</body>  
</html>
```

Subid

m ida

TH

Formularios

- Actividad 24 – Perfil
 - Construye un formulario donde se pida la siguiente información: apodo, edad e imagen de perfil.
 - Al enviar muestra la información en un formato similar a Twitter o una red social. La imagen debe estar presente.

Formularios

- Actividad 25 – Papelería online
 - Crea un formulario para subir un producto a una tienda: número de serie, nombre, precio e imagen.
 - Crea una página de identificación antes de entrar en los productos: nombre, contraseña 1 y contraseña 2.
 - Debe coincidir ambas contraseñas, además del nombre y contraseña con unas variables que tengamos almacenadas.

Formularios

- Actividad 26 – Optimizando imágenes de perfil
 - Crea un formulario para subir un imagen. Al ser subida debes:
 - Redimensionarla a una anchura de 400 pixeles.
 - Mostrarla debajo del formulario.
 - Pro:
 - Convertirla en blanco y negro.
 - Crea un crop de la imagen para que sea cuadrada sin deformarla (400x400).