

# Introducción

Lenguaje PHP

# ¿Qué es PHP?

- PHP (acrónimo recursivo de PHP: Hypertext Preprocessor) es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Ejemplo</title>
  </head>
  <body>

    <?php
      echo "¡Hola, soy un script de PHP!";
    ?>

  </body>
</html>
```

# ¿Qué es PHP?

- Lo que distingue a PHP de algo del lado del cliente como Javascript es que el código es ejecutado en el servidor, generando HTML y enviándolo al cliente.
- El cliente recibirá el resultado de ejecutar el script, aunque no se sabrá el código subyacente que era.
- El servidor web puede ser configurado incluso para que procese todos los ficheros HTML con PHP, por lo que no hay manera de que los usuarios puedan saber qué se tiene debajo de la manga.

# ¿Qué puede hacer PHP?

- PHP está enfocado principalmente a la programación de scripts del lado del servidor, por lo que se puede hacer cualquier cosa que pueda hacer otro programa CGI, como recopilar datos de formularios, generar páginas con contenidos dinámicos, o enviar y recibir cookies.

# ¿Qué es CGI?

- CGI significa Common Gateway Interface ("Interfaz de Entrada Común"), o lo que es lo mismo, Application Programming Interface.
- CGI no es ningún lenguaje de programación, sino una API de servidor web. Se trata de un sistema de comunicación que le dice al servidor web cómo enviar y recibir datos de una aplicación de servidor a un cliente.
- Esto permite a los servidores usar aplicaciones de servidor para realizar funciones concretas que añaden mayor interactividad a los sitios web, como formularios, acceso a bases de datos, login de usuarios, chats, etc.

# ¿Qué puede hacer PHP?

- Scripts del lado del servidor. Este es el campo más tradicional y el foco principal. Son necesarias tres cosas para que esto funcione: el analizador de PHP (módulo CGI o servidor), un servidor web y un navegador web. Es necesario ejecutar el servidor con una instalación de PHP conectada. Se puede acceder al resultado del programa de PHP con un navegador, viendo la página de PHP a través del servidor.

# ¿Qué puede hacer PHP?

- Scripts desde la línea de comandos. Se puede crear un script de PHP y ejecutarlo sin necesidad de un servidor o navegador. Solamente es necesario el analizador de PHP para utilizarlo de esta manera.
- Este tipo de uso es ideal para scripts que se ejecuten con regularidad empleando cron (en unix o Linux) o el Planificador de tareas (en Windows). Estos scripts también pueden usarse para tareas simples de procesamiento de texto.



# ¿Qué puede hacer PHP?

- Escribir aplicaciones de escritorio. Probablemente PHP no sea el lenguaje más apropiado para crear aplicaciones de escritorio con una interfaz gráfica de usuario, pero si se conoce bien PHP, y se quisiera utilizar algunas características avanzadas de PHP en aplicaciones del lado del cliente, se puede utilizar PHP-GTK para escribir dichos programas. También es posible de esta manera escribir aplicaciones independientes de una plataforma.

# ¿Qué puede hacer PHP?

- Con PHP no se está limitado a generar HTML. Entre las capacidades de PHP se incluyen la creación de imágenes, ficheros PDF e incluso películas Flash (usando libswf y Ming) generadas sobre la marcha.
- También se puede generar fácilmente cualquier tipo de texto, como XHTML y cualquier otro tipo de fichero XML. PHP puede autogenerar estos ficheros y guardarlos en el sistema de ficheros en vez de imprimirlos en pantalla, creando una caché en el lado del servidor para contenido dinámico

# Primeros Pasos

Ejemplos


# phpinfo()

```
info.php
You, 16 minutes ago | 1 author (You)
1 <?php
2     phpinfo();
3 ?> You, 16 minutes ago • Example 1
```

← → ↻ ⓘ localhost/examples-PHP/info.php

🔗 ☆ ⚙️ 📄 🔍 ☰

PHP Version 8.1.4



|   |  |
|---|--|
| System                                  | Windows NT KRONOS 10.0 build 19045 (Windows 10) AMD64  |
| Build Date                              | Mar 16 2022 09:30:20   |
| Build System                            | Microsoft Windows Server 2019 Datacenter [10.0.17763]  |
| Compiler                                | Visual C++ 2019  |
| Architecture                            | x64  |
| Configure Command                       | cmd /c "cd . & php configure --enable-snapshot-build --enable-debug-pack --with-pdo-oci=.\..\..\..\instantclient\sdk,shared --with-oci8-19=.\..\..\..\instantclient\sdk,shared --enable-object-out-dir=.\obj --enable-com-dotnet=shared --without-analyzer --with-pgo" |
| Server API                              | Apache 2.0 Handler   |
| Virtual Directory Support               | enabled  |
| Configuration File (php.ini) Path       | no value   |
| Loaded Configuration File               | C:\xampp\php\php.ini   |
| Scan this dir for additional .ini files | (none)   |
| Additional .ini files parsed            | (none)   |
| PHP API                                 | 20210902   |

### Palabras reservadas de PHP

|  |                                     |  |                                   |                                   |  |
|--|-------------------------------------|--|-----------------------------------|-----------------------------------|--|
| <u><a href="#">__halt_compiler()</a></u> | <u><a href="#">abstract</a></u>     | <u><a href="#">and</a></u>                         | <u><a href="#">array()</a></u>    | <u><a href="#">as</a></u>         |  |
| <u><a href="#">break</a></u>             | <u><a href="#">callable</a></u>     | <u><a href="#">case</a></u>                        | <u><a href="#">catch</a></u>      | <u><a href="#">class</a></u>      |  |
| <u><a href="#">clone</a></u>             | <u><a href="#">const</a></u>        | <u><a href="#">continue</a></u>                    | <u><a href="#">declare</a></u>    | <u><a href="#">default</a></u>    |  |
| <u><a href="#">die()</a></u>             | <u><a href="#">do</a></u>           | <u><a href="#">echo</a></u>                        | <u><a href="#">else</a></u>       | <u><a href="#">elseif</a></u>     |  |
| <u><a href="#">empty()</a></u>           | <u><a href="#">enddeclare</a></u>   | <u><a href="#">endfor</a></u>                      | <u><a href="#">endforeach</a></u> | <u><a href="#">endif</a></u>      |  |
| <u><a href="#">endswitch</a></u>         | <u><a href="#">endwhile</a></u>     | <u><a href="#">eval()</a></u>                      | <u><a href="#">exit()</a></u>     | <u><a href="#">extends</a></u>    |  |
| <u><a href="#">final</a></u>             | <u><a href="#">finally</a></u>      | <u><a href="#">fn</a></u> (a partir de PHP 7.4)    | <u><a href="#">for</a></u>        | <u><a href="#">foreach</a></u>    |  |
| <u><a href="#">function</a></u>          | <u><a href="#">global</a></u>       | <u><a href="#">goto</a></u>                        | <u><a href="#">if</a></u>         | <u><a href="#">implements</a></u> |  |
| <u><a href="#">include</a></u>           | <u><a href="#">include_once</a></u> | <u><a href="#">instanceof</a></u>                  | <u><a href="#">insteadof</a></u>  | <u><a href="#">interface</a></u>  |  |
| <u><a href="#">isset()</a></u>           | <u><a href="#">list()</a></u>       | <u><a href="#">match</a></u> (a partir de PHP 8.0) | <u><a href="#">namespace</a></u>  | <u><a href="#">new</a></u>        |  |
| <u><a href="#">or</a></u>                | <u><a href="#">print</a></u>        | <u><a href="#">private</a></u>                     | <u><a href="#">protected</a></u>  | <u><a href="#">public</a></u>     |  |
| <u><a href="#">require</a></u>           | <u><a href="#">require_once</a></u> | <u><a href="#">return</a></u>                      | <u><a href="#">static</a></u>     | <u><a href="#">switch</a></u>     |  |
| <u><a href="#">throw</a></u>             | <u><a href="#">trait</a></u>        | <u><a href="#">try</a></u>                         | <u><a href="#">unset()</a></u>    | <u><a href="#">use</a></u>        |  |
| <u><a href="#">var</a></u>               | <u><a href="#">while</a></u>        | <u><a href="#">xor</a></u>                         | <u><a href="#">yield</a></u>      | <u><a href="#">yield from</a></u> |  |

### Constantes en tiempo de compilación

|                                      |                                  |                                 |                                     |                                 |                                   |
|--------------------------------------|----------------------------------|---------------------------------|-------------------------------------|---------------------------------|-----------------------------------|
| <u><a href="#">__CLASS__</a></u>     | <u><a href="#">__DIR__</a></u>   | <u><a href="#">__FILE__</a></u> | <u><a href="#">__FUNCTION__</a></u> | <u><a href="#">__LINE__</a></u> | <u><a href="#">__METHOD__</a></u> |
| <u><a href="#">__NAMESPACE__</a></u> | <u><a href="#">__TRAIT__</a></u> |                                 |                                     |                                 |                                   |

# Variables

- En PHP las variables se representan con un signo de dólar seguido por el nombre de la variable. El nombre de la variable es sensible a minúsculas y mayúsculas.

```
<?php
$var = 'Roberto';
$Var = 'Juan';
echo "$var, $Var";      // imprime "Roberto, Juan"

$4site = 'aun no';      // inválido; comienza con un número
$_4site = 'aun no';     // válido; comienza con un carácter de subrayado
$täyte = 'mansikka';    // válido; 'ä' es ASCII (Extendido) 228
?>
```

# Variables

- PHP también ofrece otra forma de asignar valores a las variables: asignar por referencia. Esto significa que la nueva variable simplemente referencia (en otras palabras, "se convierte en un alias de" ó "apunta a") la variable original. Los cambios a la nueva variable afectan a la original, y viceversa.

```
<?php
$foo = 'Bob';           // Asigna el valor 'Bob' a $foo
$bar = &$foo;           // Referenciar $foo vía $bar.
$bar = "Mi nombre es $bar"; // Modifica $bar...
echo $bar;
echo $foo;              // $foo también se modifica.
?>
```

# Variables

- válidos:
  - \$nombre\_usuario
  - \$precio\_producto
  - \$cantidad\_articulos
  - \$fecha\_registro
- no válidos:
  - \$1numero (comienza con un número)
  - \$nombre-usuario (contiene un caracter no permitido)
  - \$#precio (contiene un caracter no permitido)
  - \$cantidad articulos (contiene un espacio)



# Tipos de datos

- Escalares:
  - boolean - Puede ser true (verdadero) o false (falso).
  - integer
  - float (número de punto flotante, también conocido como double)
  - string

# Tipos de datos

- Compuestos:

- array
- object
- callable
- iterable

# Tipos de datos

- Especiales:
  - resource: identificador que hace referencia a un recurso externo.
  - null: valor que indica ausencia de valor.
- El tipo de una variable usualmente no lo declara el programador; al contrario, es decidido en tiempo de ejecución por PHP dependiendo del contexto en el que se emplea dicha variable.

# Comentarios

```
<?php
```

```
    //Esto es un comentario de una línea
```

```
    /* Esto
```

```
    es un comentario
```

```
    de varias líneas */
```

```
?>
```

# Mucho echo, pero y qué es ?

- echo — Muestra una o más cadenas
- echo no es realmente una función (es una construcción del lenguaje), por lo que no se requiere el uso de paréntesis con él. echo no se comporta como una función, es decir no siempre se puede usar en el contexto de una función. Además, si se quiere pasar más de un parámetro a echo, éstos no deben estar entre paréntesis.

# echo en acción

```
<?php
echo "Hola mundo";

echo "Esto abarca
multiple líneas. Los saltos de línea también
se mostrarán";

echo "Esto abarca\nmúltiples líneas. Los saltos de línea también\nse mostrarán.";

echo "Para escapar caracteres se hace \"así\".";

// Se pueden usar variables dentro de una sentencia echo
$foo = "foobar";
$bar = "barbaz";

echo "foo es $foo"; // foo es foobar

// También se pueden usar arrays
$baz = array("valor" => "foo");

echo "Esto es {$baz['valor']} !"; // Esto es foo !

// Si se utilizan comillas simples, se mostrará el nombre de la variable, no su valor
echo 'foo es $foo'; // foo es $foo
```

# echo en acción

```
// Si no usan otros caracteres, se puede utilizar echo para mostrar el valor de las variables.
echo $foo;           // foobar
echo $foo,$bar;      // foobarbarbaz

// Las cadenas pueden ser pasadas individualmente como varios argumentos o
// concatenadas como un único argumento
echo 'Esta ', 'cadena ', 'está ', 'hecha ', 'con múltiple parámetros.', chr(10);
echo 'Esta ' . 'cadena ' . 'está ' . 'hecha ' . 'con concatenación.' . "\n";

echo <<<END
Aquí se utiliza la sintaxis de "here document" para mostrar
múltiples líneas con interpolación de $variable. Nótese
que el finalizador de here document debe aparecer en una
línea con solamente un punto y coma. ¡Nada de espacios extra!
END;

// Ya que echo no se comporta como una función, el siguiente código no es válido.
($variable) ? echo 'true' : echo 'false';

// Sin embargo, el siguiente código funcionará:
($variable) ? print 'true' : print 'false'; // print también es una construcción, pero
// se comporta como una función, por lo que
// puede usarse en este contexto.
echo $variable ? 'true': 'false'; // dando la vuelta a la declaración
?>
```

# Otras opciones

- `print` - Mostrar una cadena
- `printf()` - Imprimir una cadena con formato
- `flush()` - Vaciar el búfer de salida del sistema



# Arrays ... modo formal

- Un array en PHP es en realidad un mapa ordenado. Un mapa es un tipo de datos que asocia valores con claves.

```
array(  
  clave => valor,  
  clave2 => valor2,  
  clave3 => valor3,  
  ...  
)
```

# Arrays ... aquí todo bien

```
<?php
$array = array(
    "foo" => "bar",
    "bar" => "foo",
);

// a partir de PHP 5.4
$array = [
    "foo" => "bar",
    "bar" => "foo",
];
?>
```

```
<?php
$array = array(
    1     => "a",
    "1"   => "b",
    1.5   => "c",
    true  => "d",
);
var_dump($array);
?>
```

# Arrays ... 3 doritos después

```
<?php
$array = array("foo", "bar", "hello", "world");
var_dump($array);
?>
```

```
<?php
$array = array(
    "foo" => "bar",
    42    => 24,
    "multi" => array(
        "dimensional" => array(
            "array" => "foo"
        )
    )
);

var_dump($array["foo"]);
var_dump($array[42]);
var_dump($array["multi"]["dimensional"]["array"]);
?>
```

```
<?php
$array = array(
    "a",
    "b",
    6 => "c",
    "d",
);
var_dump($array);
?>
```

```
<?php
function getArray() {
    return array(1, 2, 3);
}

// en PHP 5.4
$secondElement = getArray()[1];

// anteriormente
$tmp = getArray();
$secondElement = $tmp[1];

// o
list(, $secondElement) = getArray();
?>
```

# Duda dudosa

```
<?php

// Se realiza la conversión y luego se comprueba si es igual o no, todos darán 'true'
var_dump(7 == "a");

// 3 == 3 -> true
var_dump("3" == "03");

// 10 == 10 -> true
var_dump("10" == "1e1");

// 100 == 100 -> true
var_dump(100 == "1e2");

// En este caso la conversión no se realiza, por ende es más rápido
// igual o no
var_dump(0 === "a");

// "3" === "03" -> false
var_dump("3" === "03");

// "10" === "1e1" -> false
var_dump("10" === "1e1");

// 100 == "1e2" -> false
var_dump(100 === "1e2");

?>
```

# Objetos

```
<?php
class foo
{
    function hacer_algo()
    {
        echo "Haciendo algo.";
    }
}

$bar = new foo;
$bar->hacer_algo();
?>
```

# Operadores Aritméticos

| Ejemplo    | Nombre         | Resultado   |
|------------|----------------|---|
| +\$a       | Identidad      | Conversión de \$a a int o float según el caso.                          |
| -\$a       | Negación       | Opuesto de \$a.   |
| \$a + \$b  | Adición        | Suma de \$a y \$b.  |
| \$a - \$b  | Sustracción    | Diferencia de \$a y \$b.  |
| \$a * \$b  | Multiplicación | Producto de \$a y \$b.  |
| \$a / \$b  | División       | Cociente de \$a y \$b.  |
| \$a % \$b  | Módulo         | Resto de \$a dividido por \$b.  |
| \$a ** \$b | Exponenciación | Resultado de elevar \$a a la potencia \$bésima. Introducido en PHP 5.6. |

# Referencias

- *Concepto y funcionamiento de CGI.* (n.d.). Com.Es. Retrieved October 3, 2023, from <https://diego.com.es/concepto-y-funcionamiento-de-cgi>
- *Conceptos básicos.* (n.d.). Php.net. Retrieved October 3, 2023, from <https://www.php.net/manual/es/getting-started.php>
- *Estructuras de control en PHP.* (n.d.). Com.Es. Retrieved October 6, 2023, from <https://diego.com.es/estructuras-de-control-en-php>

# Repositorio GITHUB

- <https://github.com/kralos-7/examples-PHP.git>