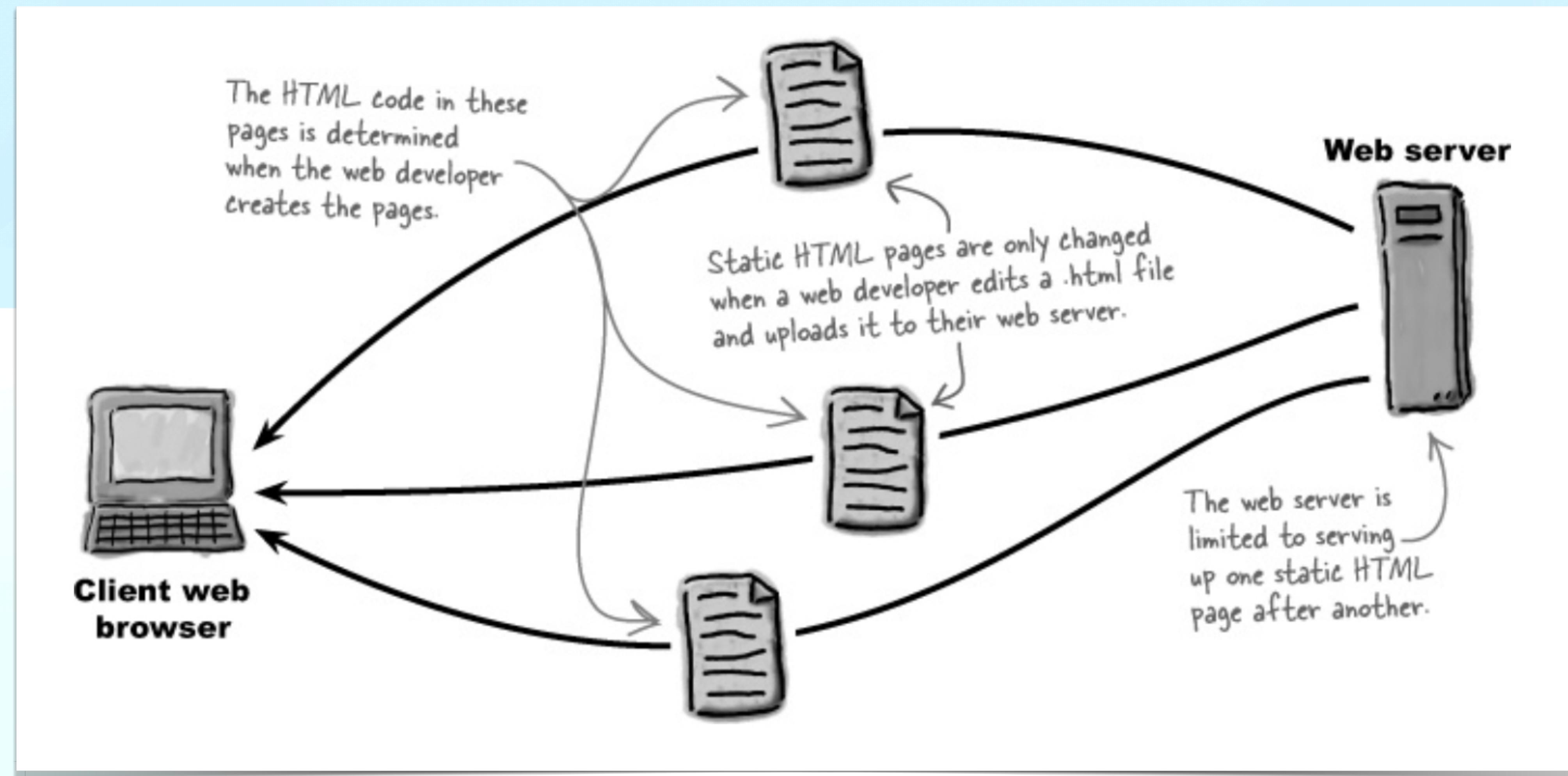


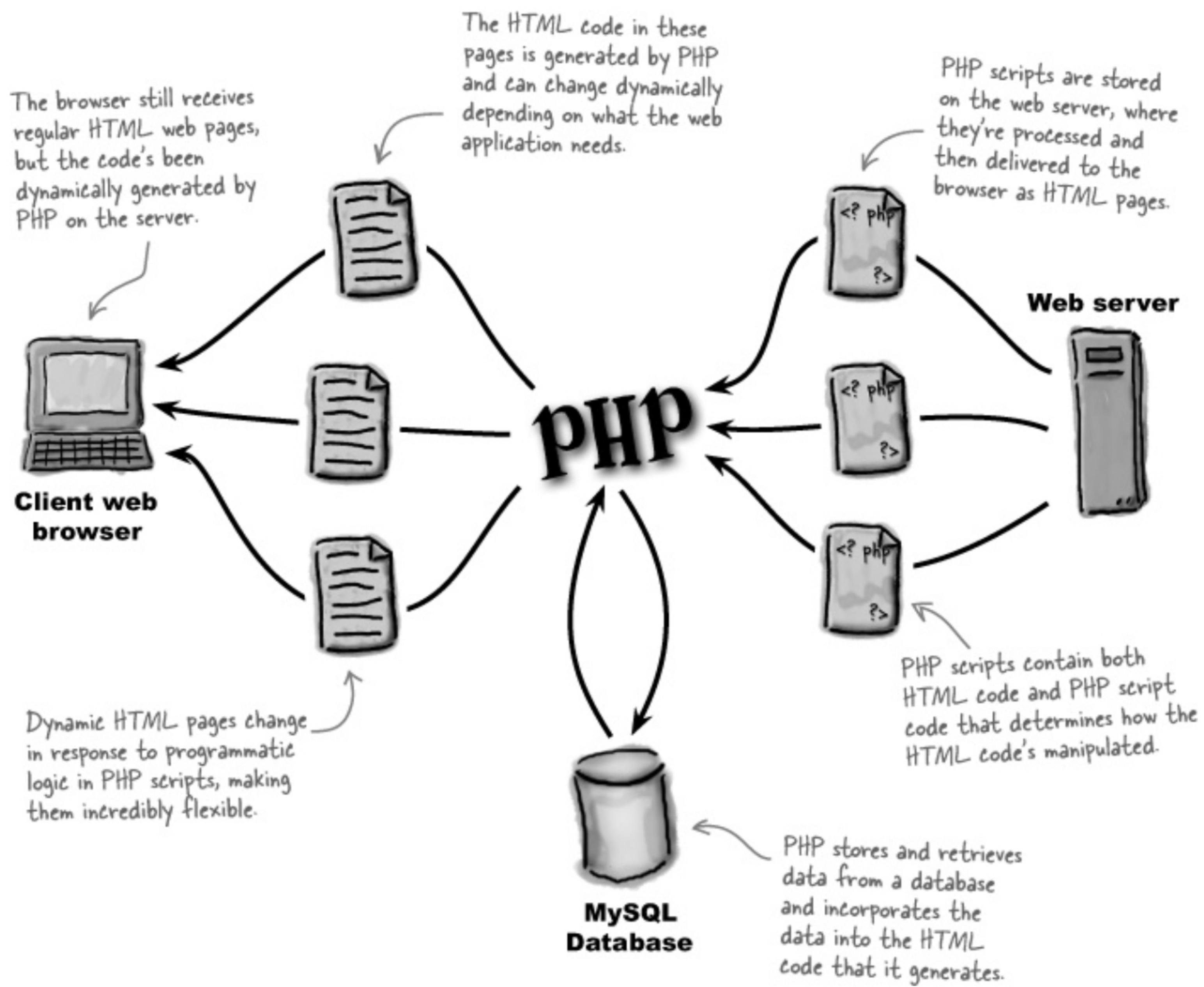
# **UT1 - Programación en entorno servidor. Introducción a PHP.**

# Programación en entorno servidor



# Programación en entorno servidor





# Full stack development =

## Front-end + Back-end

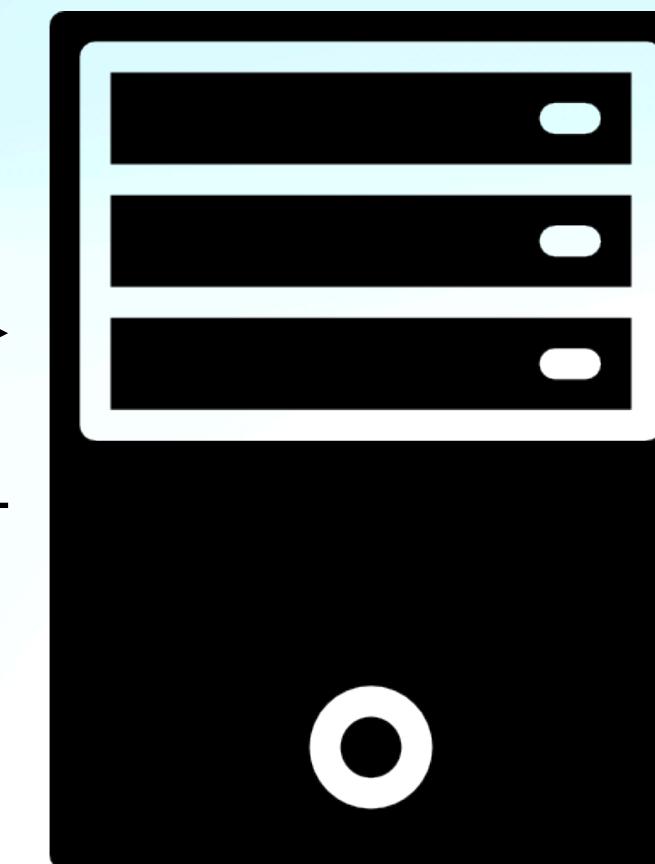


# Programación en entorno servidor

**Cliente**



**Servidor de aplicaciones**



HTTP  
↔

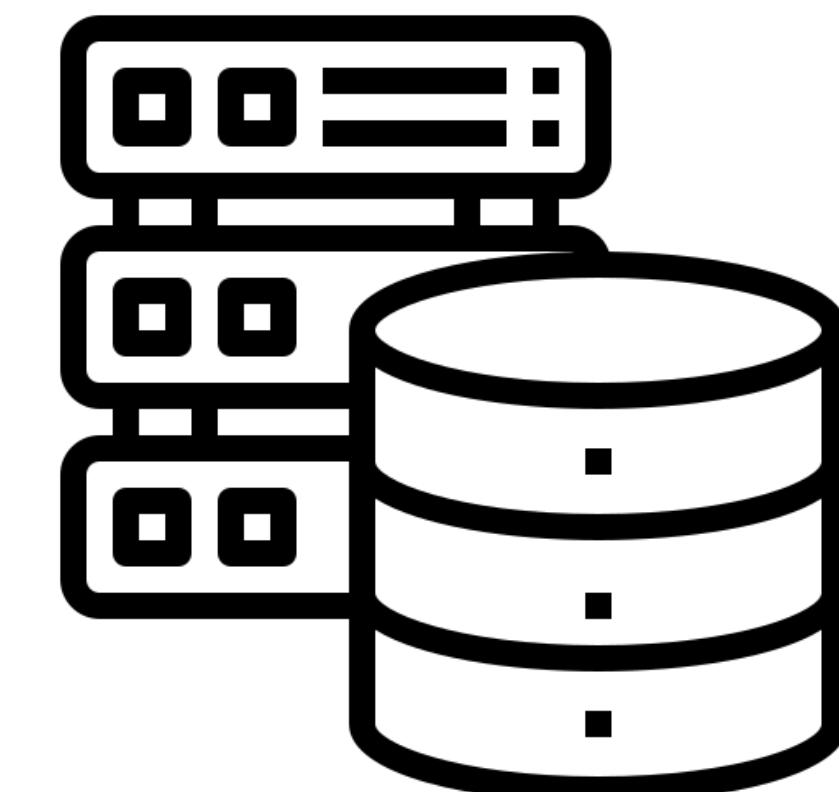
- Apache
- IIS
- NGINX
- Tomcat
- ...

**Aplicación**



- Acceso público
- Acceso privado

**Servidor de base de datos**



- MySQL
- MariaDB
- SQL Server
- Oracle
- PostgreSQL
- ...

# Programación en entorno servidor

## Ventajas de las aplicaciones web

- No requieren de software específico.
- Menor coste de implantación/mantenimiento.
- Acceso inmediato a la última versión.
- Pocos requisitos hardware para el usuario.
- Movilidad.
- Independencia de plataforma / sistema operativo.

# Programación en entorno servidor

The screenshot shows the Ticketmaster website interface. At the top, there are language options (ES) and a search bar. Below the header, the ticketmaster logo is followed by navigation links: Música, Arte y Teatro, Festivales, Deportes, Familia y Otros. A large red arrow points from the word 'Otros' towards the search bar. To the right of the search bar is a button labeled 'Accede/Regíst...'. The main content area features a banner for the 'Gran Canaria SUM Festival' with the date '30 SEP Y 1 OCT 2022'. Below the banner, the text 'Otros - Música' is displayed, followed by the heading 'Entradas para Gran Canaria SUM Festival'. At the bottom of the page, there are links for 'EVENTOS' and 'INFO'.

ES ES

**ticketmaster**® Música Arte y Teatro Festivales Deportes Familia y Otros

Artista, evento o recinto

Tarjeta Regalo Ibiza Ayuda Guía Festivales

Accede/Regíst...

Inicio / Música / Otros - Música / Gran Canaria SUM Festival Entradas

Gran Canaria  
**SUM**  
FESTIVAL  
LAS PALMAS DE  
GRAN CANARIA  
30 SEP Y 1 OCT  
2022

Otros - Música

**Entradas para Gran Canaria SUM Festival**

EVENTOS INFO

# Lado del cliente



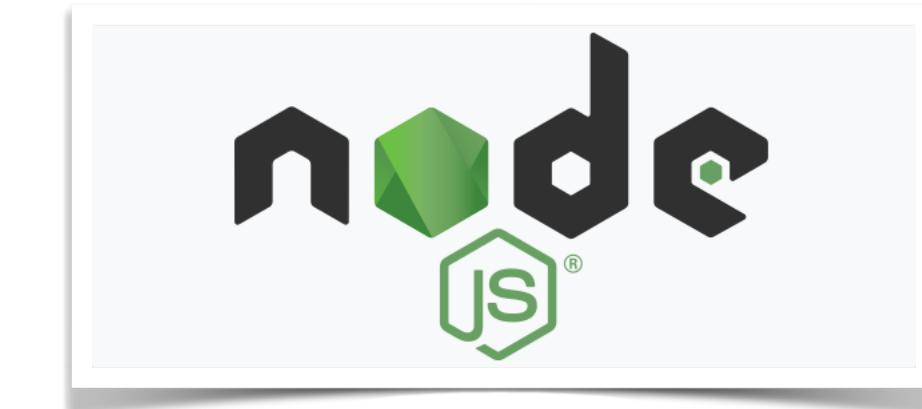
# Servidor



# Base de datos



# Frameworks



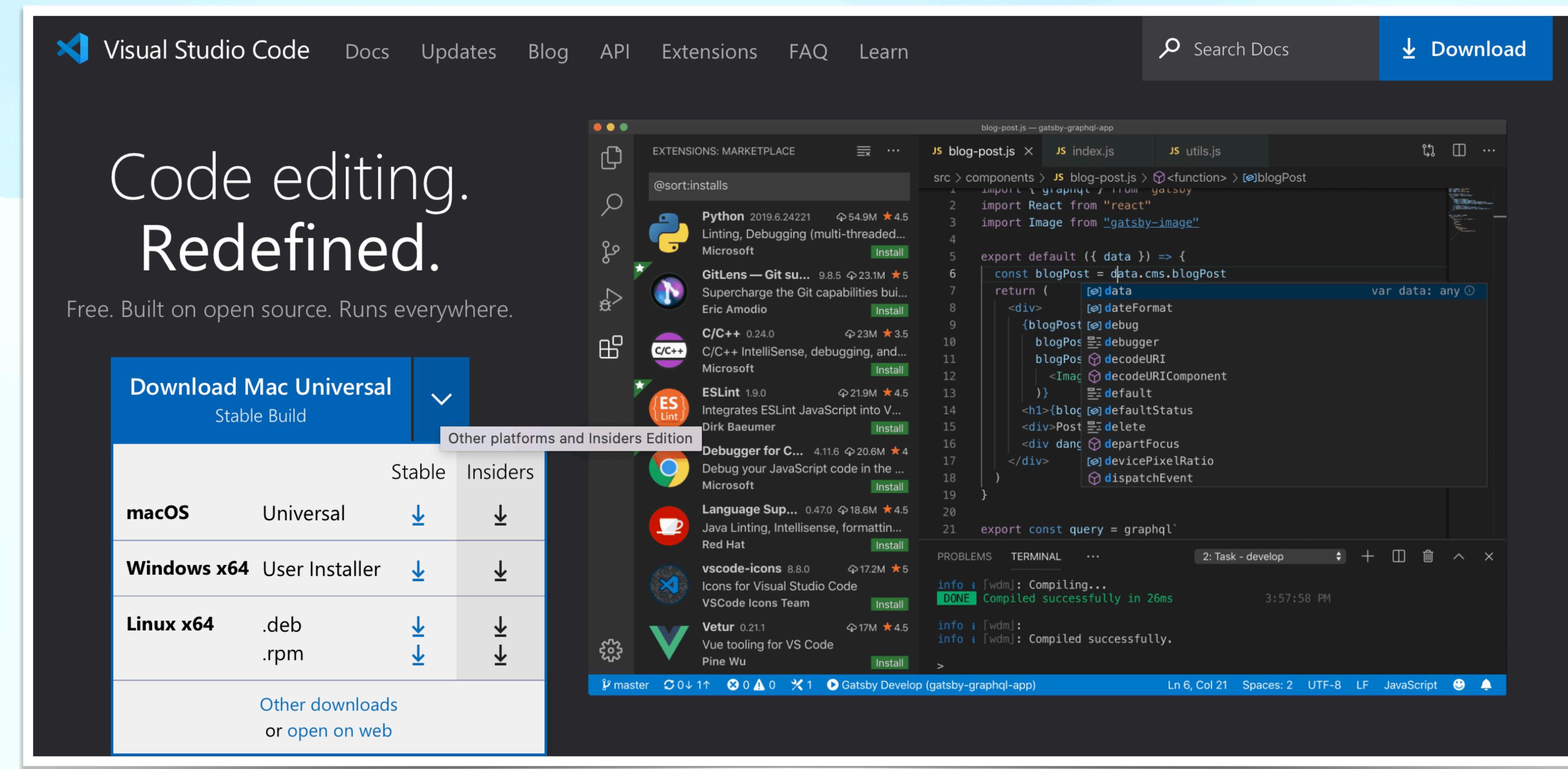
express



# Programación en entorno servidor

## Preparando el entorno

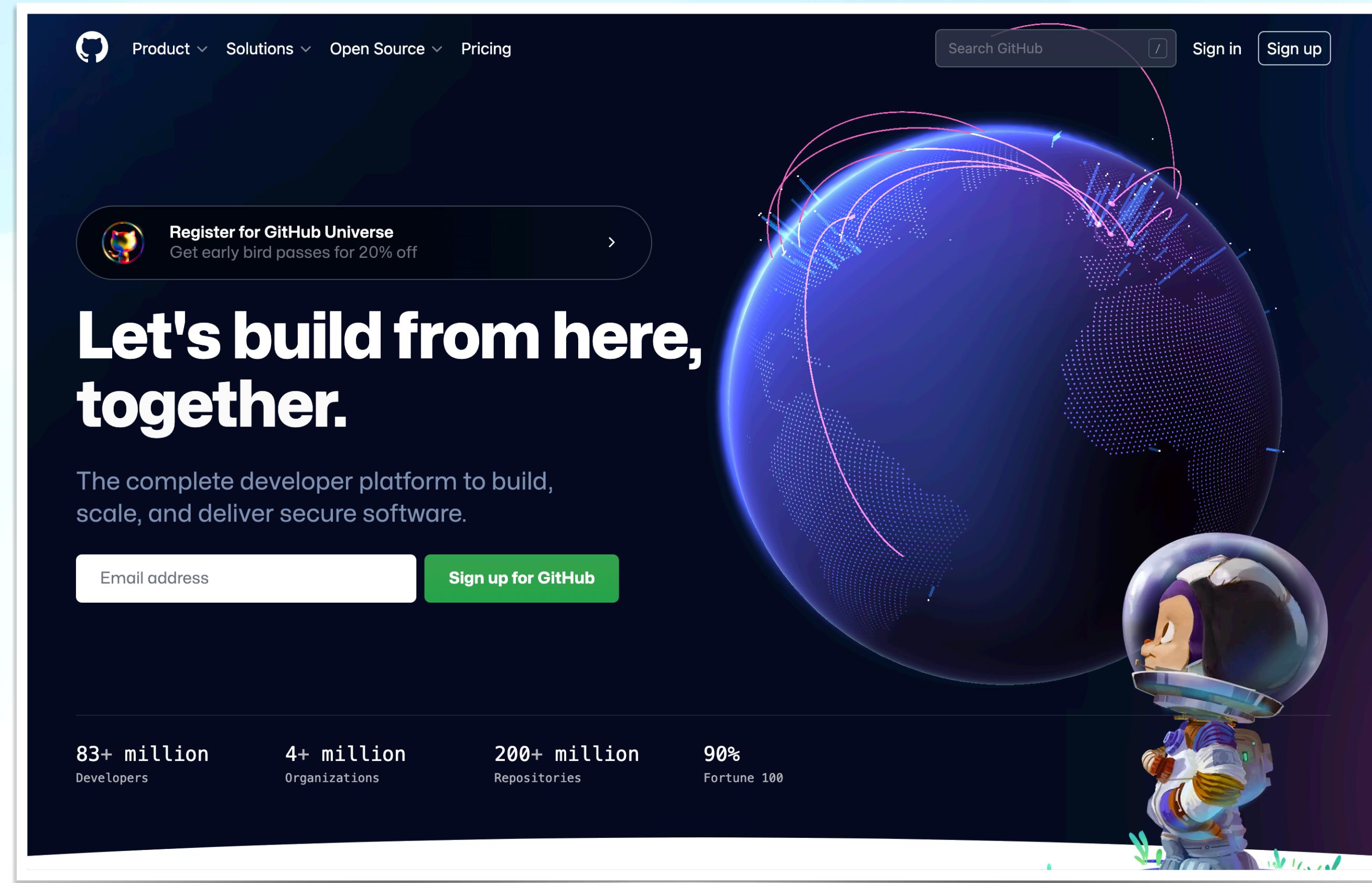
- Editor de código Visual Studio Code <https://code.visualstudio.com>



# Programación en entorno servidor

## Preparando el entorno

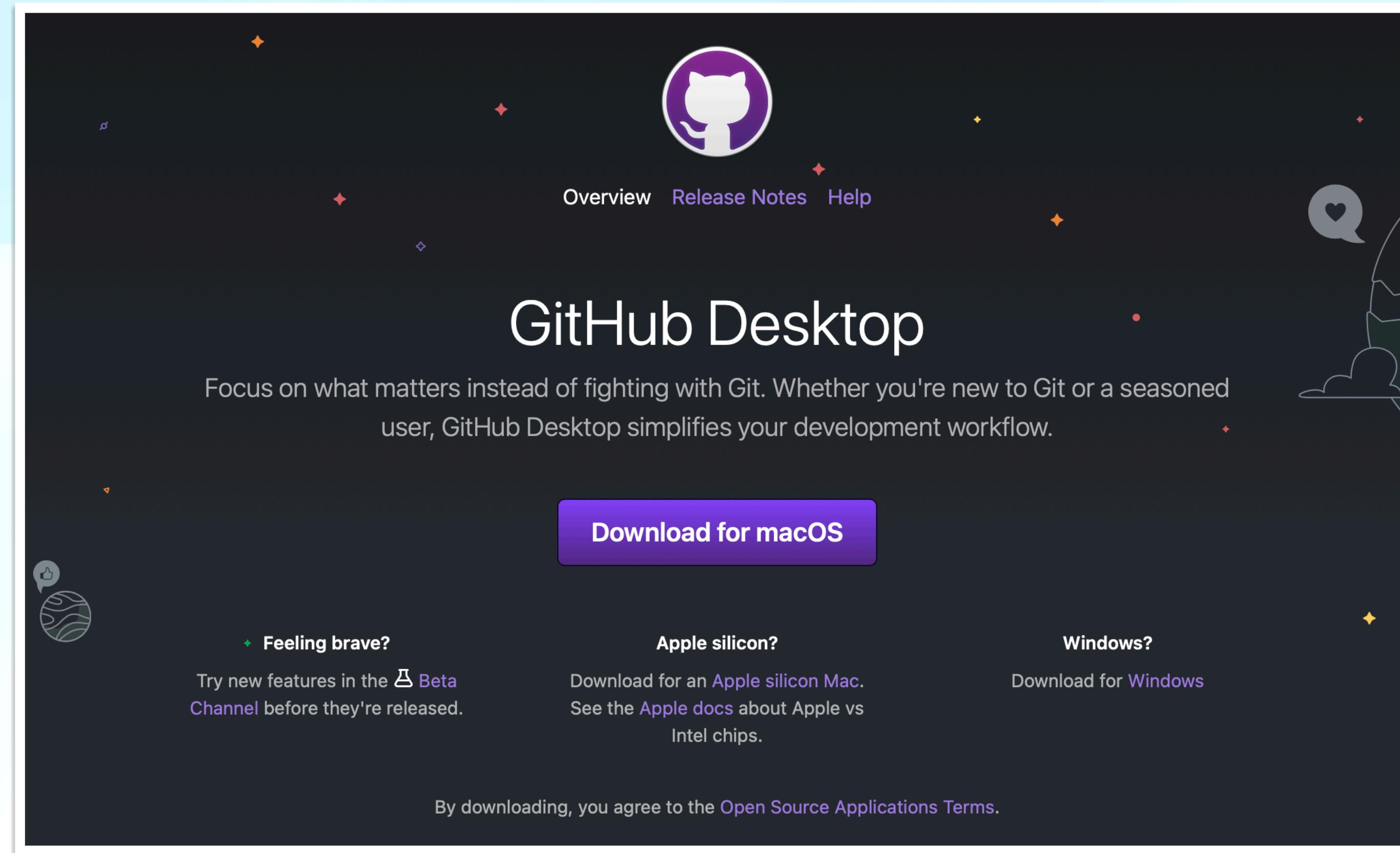
- Registrarnos en GitHub <https://github.com>



# Programación en entorno servidor

## Preparando el entorno

- Instalar el cliente de GitHub <https://desktop.github.com>



# Programación en entorno servidor

## Preparando el entorno

- Instalar Git <https://git-scm.com/download/win>

The screenshot shows the official Git website (<https://git-scm.com/>) with a focus on the 'Downloads' section for Windows. The page features the Git logo and the tagline '--distributed-even-if-your-workflow-isnt'. A search bar is at the top right. On the left, there's a sidebar with links for 'About', 'Documentation', 'Downloads' (which is highlighted in red), and 'Community'. Below the sidebar is a note about the 'Pro Git book'. The main content area is titled 'Download for Windows' and contains links to various download options: 'Standalone Installer', '32-bit Git for Windows Setup.', '64-bit Git for Windows Setup.', 'Portable ("thumbdrive edition")', '32-bit Git for Windows Portable.', '64-bit Git for Windows Portable.', and 'Using winget tool'.

git --distributed-even-if-your-workflow-isnt

Search entire site...

About

Documentation

Downloads

GUI Clients

Logos

Community

The entire [Pro Git book](#) written by Scott Chacon and Ben Straub is available to [read online for free](#). Dead tree versions are available on [Amazon.com](#).

## Download for Windows

[Click here to download](#) the latest (2.37.3) 32-bit version of Git for Windows. This is the most recent [maintained build](#). It was released [11 days ago](#), on 2022-08-30.

**Other Git for Windows downloads**

[Standalone Installer](#)

[32-bit Git for Windows Setup.](#)

[64-bit Git for Windows Setup.](#)

[Portable \("thumbdrive edition"\)](#)

[32-bit Git for Windows Portable.](#)

[64-bit Git for Windows Portable.](#)

[Using winget tool](#)

# Programación en entorno servidor

## Preparando el entorno

- Instalar XAMPP (Servidor web y BBDD) <https://www.apachefriends.org>

The screenshot shows the Apache Friends website with the XAMPP page selected. The top navigation bar includes links for Apache Friends, Descargar, Complementos, Alojamiento, Comunidad, Acerca de, a search bar, and a language switch to Spanish (ES). The main content area features the XAMPP logo (an orange square with a white play button icon) and the text "XAMPP Apache + MariaDB + PHP + Perl". Below this, a section titled "¿Qué es XAMPP?" provides a brief description of what XAMPP is and how it can be used. To the right of this text is a large image of the XAMPP application icon. At the bottom, there are four download links: "Descargar" (Windows, Linux, OS X) and "Pulsa aquí para otras versiones".

Apache Friends Descargar Complementos Alojamiento Comunidad Acerca de Buscar.. Buscar ES

## XAMPP Apache + MariaDB + PHP + Perl

### ¿Qué es XAMPP?

XAMPP es el entorno más popular de desarrollo con PHP

XAMPP es una distribución de Apache completamente gratuita y fácil de instalar que contiene MariaDB, PHP y Perl. El paquete de instalación de XAMPP ha sido diseñado para ser increíblemente fácil de instalar y usar.



**XAMPP**

**Descargar**  
Pulsa aquí para otras versiones

XAMPP para Windows  
8.1.6 (PHP 8.1.6)

XAMPP para Linux  
8.1.6 (PHP 8.1.6)

XAMPP para OS X  
8.1.6 (PHP 8.1.6)

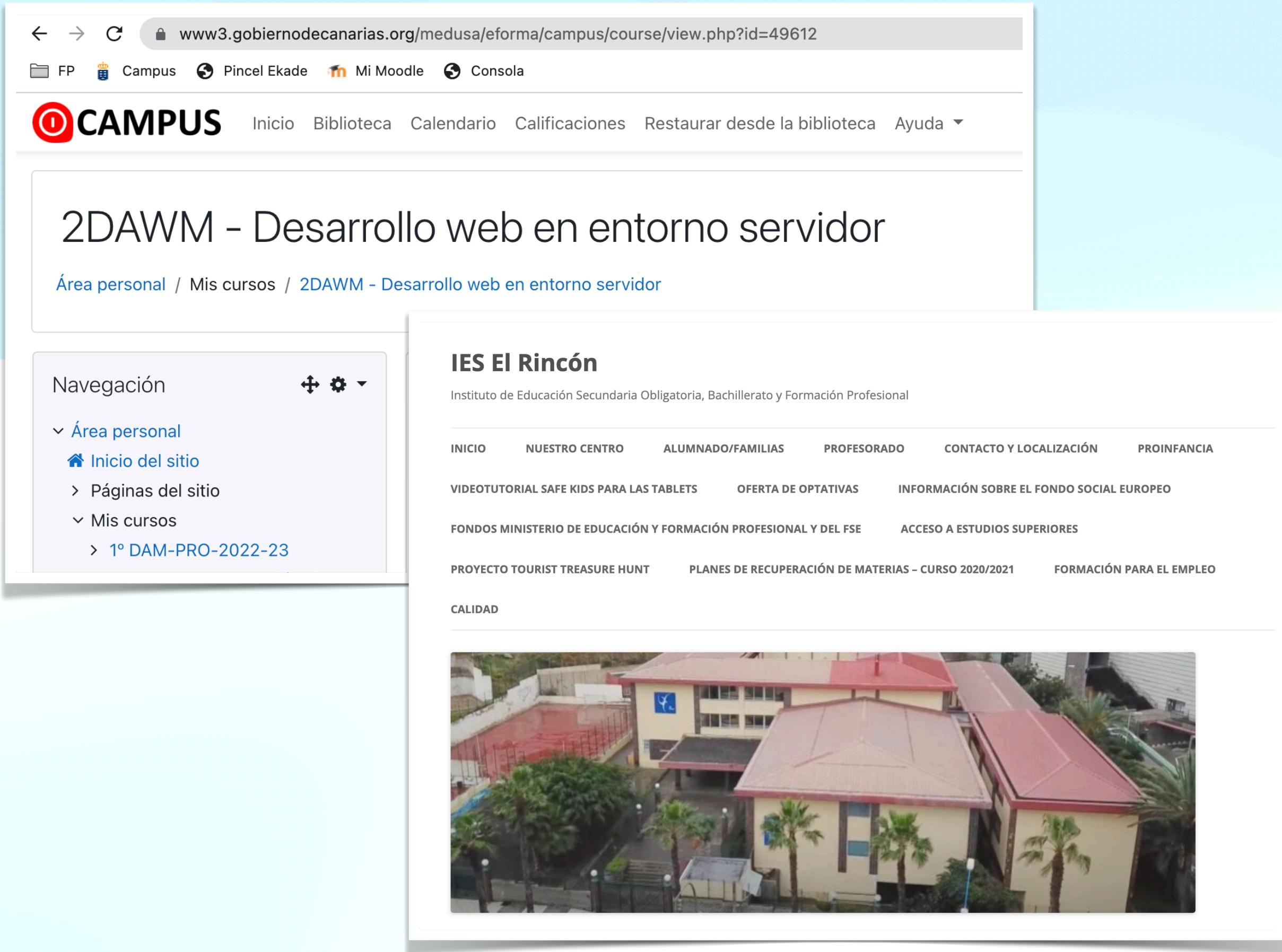
# Introducción a PHP

- PHP es un lenguaje de programación de código abierto, interpretado, sencillo pero potente orientado al desarrollo web.
- Las aplicaciones escritas en PHP se pueden ejecutar en los servidores web más extendidos (Apache, IIS, ...) y en los SSOO principales.
- Soporta un gran número de bases de datos (MySQL, PostgreSQL, Oracle, Sybase, MariaDB, MongoDB, etc).
- Soporta programación orientada a objetos

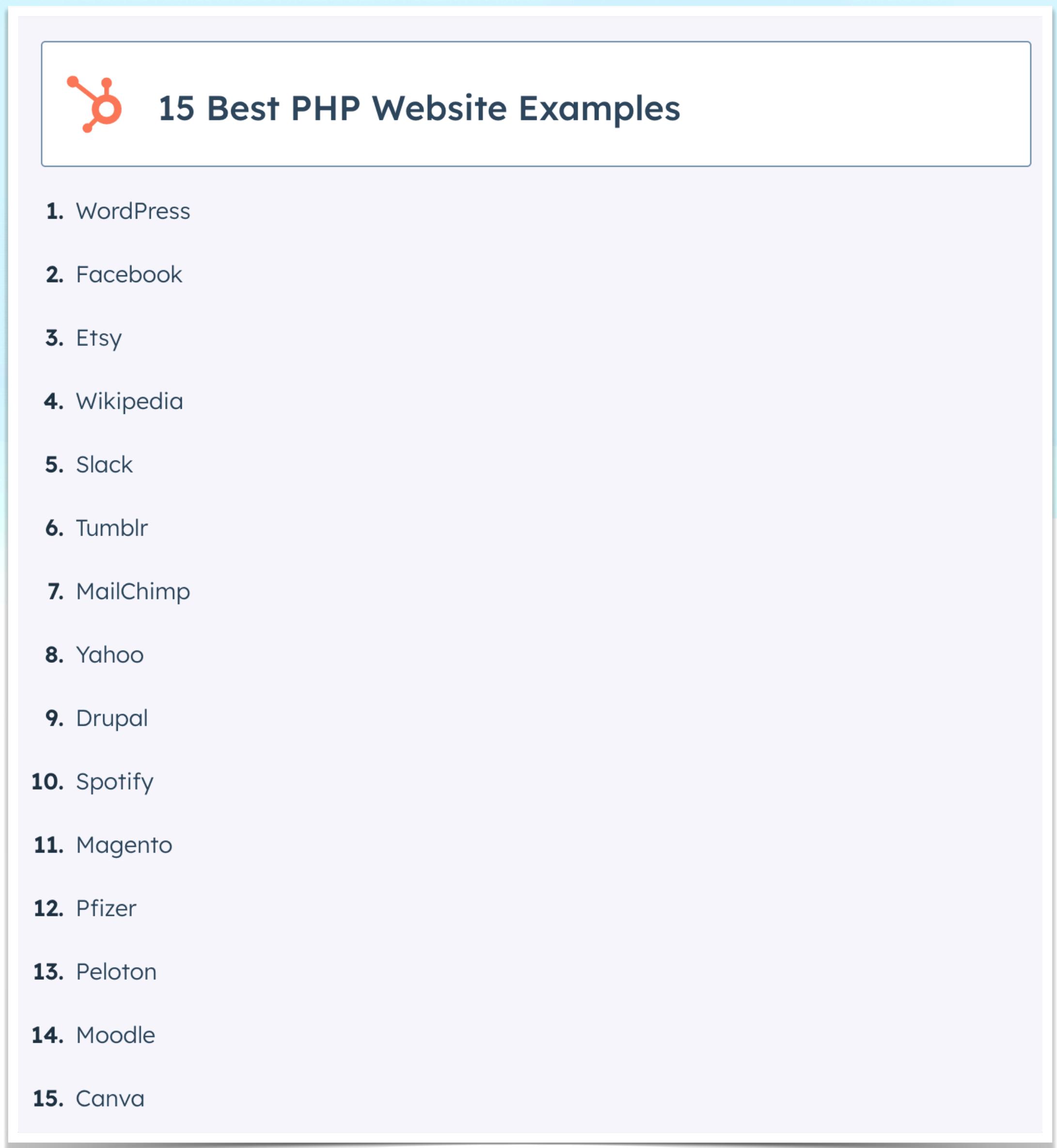
# Introducción a PHP

- ¿Dónde se usa PHP?
  - **Scripts del lado de servidor:** es necesario un analizador PHP, un servidor web y un navegador web.
  - Scripts desde la línea de comandos: se pueden programar scripts PHP para ser ejecutados desde la línea de comandos sin necesidad de servidor web.
  - Aplicaciones de escritorio

# Introducción a PHP



A screenshot of a web browser displaying a PHP-based application. The URL in the address bar is [www3.gobiernodecanarias.org/medusa/eforma/campus/course/view.php?id=49612](http://www3.gobiernodecanarias.org/medusa/eforma/campus/course/view.php?id=49612). The page title is "2DAWM - Desarrollo web en entorno servidor". The header includes links for "FP", "Campus", "Pincel Ekade", "Mi Moodle", and "Consola". Below the header, there's a logo for "CAMPUS" and a navigation menu with links to "Inicio", "Biblioteca", "Calendario", "Calificaciones", "Restaurar desde la biblioteca", and "Ayuda". A breadcrumb trail shows "Área personal / Mis cursos / 2DAWM - Desarrollo web en entorno servidor". On the left, a sidebar titled "Navegación" lists "Área personal" with "Inicio del sitio", "Páginas del sitio", "Mis cursos", and "1º DAM-PRO-2022-23". The main content area features the heading "IES El Rincón" and a photograph of a building with a red roof and yellow walls.



**15 Best PHP Website Examples**

1. WordPress
2. Facebook
3. Etsy
4. Wikipedia
5. Slack
6. Tumblr
7. MailChimp
8. Yahoo
9. Drupal
10. Spotify
11. Magento
12. Pfizer
13. Peloton
14. Moodle
15. Canva

# Introducción a PHP

- Para programar scripts PHP en el lado del servidor necesitamos un servidor web con soporte para PHP.
- De esta forma, todos los ficheros con la extensión .php serán tratados por el módulo PHP correspondiente, que los interpretará y ejecutar bajo petición del cliente.
- No es necesario compilar los ficheros .php.
- Si un fichero contiene solamente código de PHP, es preferible omitir la etiqueta de cierre de PHP al final del mismo.

# Introducción a PHP

```
<html>
  <head>
    <title>Prueba de PHP</title>
  </head>
  <body>
    <?php echo '<p>Hola Mundo</p>' ; ?>
  </body>
</html>
```

# Introducción a PHP

## Sintaxis

- El elemento básico en PHP es el bloque.
- Un bloque PHP está delimitado por “<?php” y “?>”.

```
<?php  
    sentencial;  
    sentenciaN;  
? >
```

# Introducción a PHP

## Uso de mayúsculas

- Los nombres de las clases y funciones definidas por el usuario, así como las palabras reservadas del lenguaje (echo, while, class, etc.) no son sensibles a mayúsculas.

```
<?php
    echo("Hola mundo");
    ECho("Hola mundo");
    $i=1;
    WHile($i<5) {
        $i++;
        echo($i);
    }
?>
```

- Sin embargo las variables sí lo son. Por lo tanto, las variables \$name, \$NAME y \$NaMe son variables diferentes.

# Introducción a PHP

## Variables y tipos de datos

- PHP es un lenguaje no fuertemente tipado.
- No es necesario indicar el tipo de dato al declarar una variable.
- De hecho, las variables no se declaran. Se crean cuando se les asigna un valor. El tipo de dato depende del valor con que se inicializan.
- Esto puede dar lugar a código de baja calidad y generar errores difíciles de detectar.

# Introducción a PHP

## Declaración de variables

- Los identificadores de variables van siempre precedidos por el carácter “\$”.

```
$variable = valor;
```

- Es posible cambiar el tipo de dato de una variable asignándole un valor de otro tipo de dato.
- Una variable “X” puede almacenar el nombre de otra variable “Z”. Podemos acceder al valor de “Z” a través del valor de “X” .

```
$x = 5;  
$y = "x";  
echo "Valor de la variable $y: ".$y;
```

# Introducción a PHP

## Declaración de variables

- Comprueba a través de un script donde incluyas el siguiente código cómo varían los tipos de las variables. Utiliza el método `gettype($variable)` para comprobar el tipo de una variable.

```
1 <?php
2 /* declaración de variables */
3 $entero = 4; // tipo integer
4 $numero = 4.5; // tipo coma flotante
5 $cadena = "cadena"; // tipo cadena de caracteres
6 $bool = TRUE; //tipo booleano
7 /* cambio de tipo de una variable */
8 $a = 5; // entero
```

# Introducción a PHP

## Variables no inicializadas

- Si se intenta utilizar una variable antes de asignarle un valor, se genera una advertencia, pero no se interrumpe el script.
- Si una variable no inicializada aparece en una expresión, dicha expresión se calcula tomando el valor por defecto para ese dato.
- Prueba lo anterior a través de este script:

```
1 <?php
2 $var1 = 100;
3 $var3 = 100 + $var2; //
4 echo "$var3 <br>"; //
5 $var3 = 100 * $var2; /
6 echo "$var3 <br>"; //
```

# Introducción a PHP

## Valor nulo

- El valor nulo está disponible a través de la palabra reservada **NULL**.
- Se puede comprobar si una variable tiene valor nulo mediante `is_null()`
- Representa una variable que no tiene valor.
  - `$variable = "valor";`
  - `$variable = null; // Variable sin valor`
  - `$variable = NULL; // Ídem`
  - `$variable = NULL; // Ídem`

# Introducción a PHP

## Constantes

- Para definir constantes se usa la función define(), que recibe el nombre de la constante y el valor que queremos darle.
  - `define("LIMITE", 1000);`
  - `echo LIMITE;`
- Se suele utilizar mayúsculas para denominar las constantes.

# Introducción a PHP

## Tipo numérico

- Para representar números enteros se usa el tipo de dato “integer”.
- Para los números reales se utiliza “float”.
- La conversión entre integer y float es automática. Si se recibe un float cuando se espera un integer, se trunca.
- Se pueden utilizar los operadores (int) o (float) para realizar conversiones.

# Introducción a PHP

## Cadenas

- El tipo de dato “string” permite almacenar cadenas de caracteres.
- Se pueden usar comillas simples o dobles para delimitar cadenas.
- Si se usan comillas dobles, las variables que aparezcan en la cadena se sustituirán por su valor. Esto evita tener que hacer concatenaciones.
  - `echo ("Este es el valor de la $variable");`
  - `echo ('Este es el valor de la '. $variable);`

# Introducción a PHP

## Booleanos

- Solo pueden tomar valores TRUE o FALSE.
- Este es el tipo de dato que se obtiene, entre otros casos, como resultado de los operadores de comparación y se utiliza en sentencias condicionales y bucles.
- Si se espera un valor booleano y se recibe otro tipo de dato se aplican las siguientes reglas.

Tipo	Valor como boolean
integer	Si es 0 se toma FALSE, en otro caso como TRUE
float	Si es 0.0 se toma FALSE, en otro caso como TRUE
string	Si es una cadena vacía o "0", se toma como FALSE, en otro caso como TRUE
variables no inicializadas	FALSE
null	FALSE
array	Si no tiene elementos se toma FALSE, en otro caso como TRUE

# Introducción a PHP

## Ámbito de las variables

### Local

- Una variable declarada en una función es local a la función. Solo es visible al código de la función. Las variables definidas fuera de una función no son accesibles dentro de ella.

```
<?php  
  
    function updatecounter()  
    {  
        $counter++;  
        echo "<br>El contador dentro es $counter";  
    }  
    $counter = 10;  
    updatecounter();  
    echo "<br>El contador fuera es $counter";  
?>
```

# Introducción a PHP

## Ámbito de las variables

### Global

- Las variables declaradas fuera de una función son globales. Pueden ser accedidas desde cualquier parte del programa, PERO no dentro de las funciones, salvo que se use la palabra reservada `global`.

```
function updateCounter()
{
    global $counter;
    $counter++;
}

$counter = 10;
updateCounter();
echo $counter;
11
```

# Introducción a PHP

## Ámbito de las variables

### Variables estáticas

- Una variable estática conserva su valor entre llamadas a una función pero es visible solo dentro de esa función.
- Se declaran con la palabra reservada `static`.

```
function updateCounter()
{
    static $counter = 0;
    $counter++;

    echo "Static counter is now {$counter}<br/>";
}

$counter = 10;
updateCounter();
updateCounter();

echo "Global counter is {$counter}";
Static counter is now 1
Static counter is now 2
Global counter is 10
```

# Introducción a PHP

## Ámbito de las variables

### Parámetros de funciones

- Los parámetros de las funciones son locales.
- Solo están disponibles dentro de las funciones.

# Introducción a PHP

## Variables predefinidas

- En PHP hay muchas variables predefinidas disponibles.
- Contienen información sobre el servidor, datos enviados por el cliente o variables de entorno. Dentro de ellas existe un grupo de ellas denominadas “superglobales” que están disponibles en cualquier ámbito.

Nombre	Descripción
\$GLOBALS	Variables globales definidas en la aplicación
\$_SERVER	Información sobre el servidor
\$_GET	Parámetros enviados con el método GET (en la URL)
\$_POST	Parámetros enviados con el método POST (formularios)
\$_FILES	Ficheros subidos al servidor
\$_COOKIE	Cookies enviadas por el cliente
\$_SESSION	Información de sesión
\$_REQUEST	Contiene la información de \$_GET, \$_POST y \$_COOKIE
\$_ENV	Variables de entorno

# Introducción a PHP

## Comentarios

```
<?php
    //comentario de línea
    $a = 0; // comentario de línea
    /* comentario
       de bloque
    */
    # comentario de una sola línea
```

# Introducción a PHP

## Operadores

- <https://www.php.net/manual/es/language.operators.php>

### Operadores

#### Tabla de contenidos

- [Precedencia de operadores](#)
- [Operadores aritméticos](#)
- [Operadores de asignación](#)
- [Operadores bit a bit](#)
- [Operadores de comparación](#)
- [Operadores de control de errores](#)
- [Operadores de ejecución](#)
- [Operadores de incremento/decremento](#)
- [Operadores lógicos](#)
- [Operadores para strings](#)
- [Operadores para arrays](#)
- [Operadores de tipo](#)

# Introducción a PHP

## Estructuras de control

- <https://www.php.net/manual/es/language.control-structures.php>

### Estructuras de Control

#### Tabla de contenidos

- [Introducción](#)
- [if](#)
- [else](#)
- [elseif/else if](#)
- [Sintaxis alternativa de estructuras de control](#)
- [while](#)
- [do-while](#)
- [for](#)
- [foreach](#)
- [break](#)
- [continue](#)
- [switch](#)

# Introducción a PHP

## Arrays

- Los arrays en PHP son una estructura muy flexible y potente.
- Los elementos de un array se identifican por una clave, que puede ser un entero o una cadena.
- Los elementos guardan un orden dentro del array. Este orden está determinado por el orden que se utilizó al declarar el array o al añadir nuevos.

```
$arr = [  
    clave1 => valor1,  
    ...  
    claven => valorn  
];
```

```
$var = array(  
    clave1 => valor1,  
    ...  
    claven => valorn  
);
```

# Introducción a PHP

## Arrays

- Se puede acceder a cada elemento del array por su clave usando los corchetes, como es habitual en otros lenguajes:
  - `$arr[clave] = valor;`
- El siguiente bucle se ejecuta una vez por cada elemento del array \$arr. En cada iteración del bucle, \$valor almacena el elemento correspondiente.

```
foreach($arr as $valor){  
    ...  
}
```

# Introducción a PHP

## Arrays

- También se puede utilizar especificando una variable para la clave:
  - `foreach ($arr as $clave => $valor) {...}`
- Se puede mostrar el contenido de un array, a efectos de depuración, mediante `print_r($array)`
- Es posible declarar un array sin usar claves. En ese caso se asignan como claves por defecto enteros consecutivos empezando por 0.
  - `$arr1 = array(10, 20, 30, 40)`

# Introducción a PHP

## Funciones predefinidas

- PHP cuenta con una gran cantidad de funciones predefinidas para las tareas más habituales.
- La mejor referencia la podemos encontrar en
  - <https://www.php.net/manual/es/funcref.php>

# Introducción a PHP

## Funciones predefinidas

### Funciones de variables

isset(\$var)	TRUE si la variable está inicializada y no es NULL
is_null(\$var)	TRUE si la variable es NULL
empty(\$var)	TRUE si la variable no está inicializada o su valor es FALSE
is_int(\$var), is_float(\$var), is_bool(\$var), is_array(\$var)	Para comprobar el tipo de dato de \$var
intval(\$var), floatval(\$var), boolvar(\$var), strval(\$var)	Para obtener el valor de \$var como otro tipo de dato

# Introducción a PHP

## Funciones predefinidas

### Funciones de cadenas

strlen(\$cad)	Devuelve la longitud de \$cad
explode(\$cad, \$token)	Parte una cadena utilizando \$token como separador. Devuelve un array de cadenas
implode(\$token, \$array)	Crea una cadena larga a partir de un array de cadenas, entre cadena y cadena se introduce \$token
strcmp(\$cad1, \$cad2)	Compara las dos cadenas. Devuelve 0 si son iguales, -1 si \$cad1 es menor y 1 si \$cad1 es mayor
strtolower(\$cad), strtoupper(\$cad)	Devuelven \$cad en mayúsculas o minúsculas, respectivamente
str(\$cad1, \$cad2)	Busca la primera ocurrencia de \$cad2 en \$cad1. Si no aparece devuelve FALSE, si aparece devuelve \$cad1 desde donde comienza la ocurrencia

# Introducción a PHP

## Funciones predefinidas

### Funciones de arrays

ksort(\$arr), krsort(\$arr)	Ordena el <i>array</i> por clave en orden ascendente o descendente
sort(\$arr), rsort(\$arr)	Ordena el <i>array</i> por valor en orden ascendente o descendente
array_values(\$arr)	Devuelve los valores de \$arr
array_keys(\$arr)	Devuelve las claves de \$arr
array_key_exists(\$arr, \$cla)	Devuelve verdadero si algún elemento de \$arr tiene clave \$cla
count(\$arr)	Devuelve el número de elementos del <i>array</i>

# Introducción a PHP

## Funciones definidas por el usuario

- La sintaxis para definir una función es la siguiente:

```
<?php  
function foo($arg_1, $arg_2, /* ..., */ $arg_n)  
{  
    echo "Función de ejemplo.\n";  
    return $valor_devuelto;  
}  
?>
```

# Introducción a PHP

## Funciones definidas por el usuario

- Las funciones pueden tener o no argumentos.
- Los argumentos son variables locales a la función y solo existen mientras se esté ejecutando.
- No es necesario declarar el tipo de dato que devuelve la función. De hecho, es posible que una función devuelva tipos de datos diferentes según el caso.

# Introducción a PHP

## Funciones definidas por el usuario

- Ejemplo de función:

```
<?php
    function suma($a, $b)    {
        return $a + $b;
    }
    echo suma(4,8).'  
' ;
    $var1 = 35;
    $var2 = 5;
    $var3 = suma($var1, $var2);
    echo $var3.'  
' ;
```

# Introducción a PHP

## Funciones definidas por el usuario

- Es posible especificar valores por defecto para los argumentos. Si al llamar a la función no se usa el argumento, se toma el valor por defecto.

```
<?php
    function saludar($nombre = 'usuario'){
        echo "Hola $nombre <br>";
    }
    saludar();
    saludar("Ana");
```

La salida será:

Hola usuario  
Hola Ana

# Introducción a PHP

## Incluyendo otros ficheros en nuestro PHP

- Podemos incluir código y HTML desde otro fichero a través de “require” y de “include”.
- Se puede utilizar en bucles y en estructuras de control condicionales.
- Los ficheros deben estar alojados en los directorios especificados en el parámetro “include\_path” (php.ini). Su valor se puede sobreescribir mediante la función “set\_include\_path()”.
- Si no se localizan los ficheros en el “include\_path”, se intentarán localizar en el directorio de ejecución del script.
- La diferencia entre “require” e “include” es...

```
<?php include "header.html"; ?>  
content  
<?php include "footer.html"; ?>
```

# Introducción a PHP

## Incluyendo otros ficheros en nuestro PHP

```
<?php include "header.html"; ?>  
content  
<?php include "footer.html"; ?>
```

```
require "codelib.php";  
mysub(); // defined in codelib.php
```

# Introducción a PHP

## Incluyendo otros ficheros en nuestro PHP

- Si el parámetro “allow\_url\_fopen” está habilitado en el fichero “php.ini” se pueden incluir ficheros a partir de urls del tipo “http://“, “https://“ o “ftp://“.

```
include "http://www.example.com/codelib.php";
```

- Podemos utilizar “require\_once” o “include\_once” para evitar que la inclusión por error del mismo fichero en más de una ocasión genere problemas de redefinición de funciones, múltiples cabeceras HTML, etc.
- En este caso, se cargan los ficheros la primera vez que se invocan y se ignoran las siguientes.

# Introducción a PHP

## Incluyendo otros ficheros en nuestro PHP

- El código alojado en un fichero incluido se importa en el ámbito que tiene efecto en donde la sentencia “include” se encuentre.
- Si el “include” o “require” están en una función, las variables que están en los ficheros incluidos tendrán el ámbito de la función donde se cargaron.

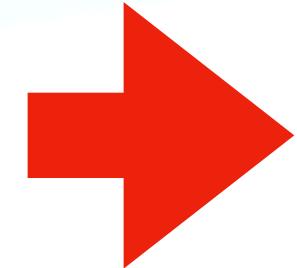
# Introducción a PHP

## Manejo de formularios

```
<html>
<body>

<form action="welcome.php" method="post">
Name: <input type="text" name="name"><br>
E-mail: <input type="text" name="email"><br>
<input type="submit">
</form>

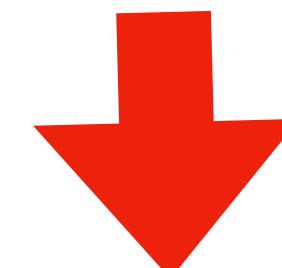
</body>
</html>
```



```
<html>
<body>

Welcome <?php echo $_POST["name"]; ?><br>
Your email address is: <?php echo $_POST["email"]; ?>

</body>
</html>
```



Welcome John  
Your email address is john.doe@example.com

# Introducción a PHP

## Manejo de formularios

- Tanto los métodos POST como GET crean un array formado por pares clave->valor donde las claves son los nombres de los campos del formulario y los valores contienen los datos registrados por el usuario.
- POST y GET se manejan mediante los arrays `$_GET` y `$_POST`. Ambas son variables “superglobales”, es decir, que están accesibles SIEMPRE: en cualquier ámbito, desde cualquier función, clase o fichero sin hacer nada especial.

# Introducción a PHP

## Manejo de formularios

- Método GET
  - La información enviada a través de un formulario mediante GET está disponible para todos los usuarios (los campos y sus valores se visualizan en la URL).
  - Tiene un límite respecto a la cantidad de información que permite enviar (cerca de 2000 caracteres).
  - Permite guardar la URL de envío en los favoritos, lo cuál puede ser útil a veces.
  - Se puede usar para el envío de datos no sensibles.

# Introducción a PHP

## Manejo de formularios

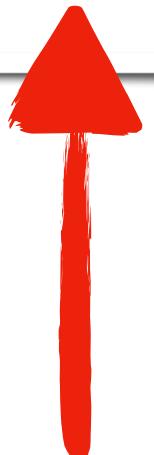
- Método POST
  - La información que se envía mediante POST no se muestra en la URL (se envía en el cuerpo de la petición HTTP).
  - No tiene límites respecto a la cantidad de información a enviar.
  - Permite el envío de ficheros.
  - Es el método más usado.

# Introducción a PHP

## Manejo de formularios

- Campos que permiten enviar múltiples valores
  - Si vamos a recibir un campo que puede tener múltiples valores (por ejemplo, un checkbox) tenemos que declararlo de la siguiente forma para que podamos acceder mediante un array a todos sus valores:

```
<input type="checkbox" name="vehicle[]" value="Bike">
<input type="checkbox" name="vehicle[]" value="Car">
<input type="checkbox" name="vehicle[]" value="Boat">
```



# Introducción a PHP

## Manejo de formularios - Envío de ficheros

- El servidor debe tener habilitada, en el fichero “php.ini”, la directiva “file\_uploads”.

```
file_uploads = On
```

- El formulario HTML debe tener:
  - El atributo “enctype” de la etiqueta “form” con el siguiente valor:  
`enctype="multipart/form-data"`
  - Un campo de tipo “file”.

```
<input type="file" name="fileToUpload" id="fileToUpload">
```

# Introducción a PHP

## Manejo de formularios - Envío de ficheros

```
<!DOCTYPE html>
<html>
<body>

<form action="recibearchivo.php" method="post" enctype="multipart/form-data">
    Select image to upload:
    <input type="file" name="fichero">
    <input type="submit" value="Upload Image" name="submit">
</form>

</body>
</html>
```

# Introducción a PHP

## Manejo de formularios - Envío de ficheros

```
<?php

$fileUploadDir = "/usr/local/var/www/dsw/ficheros/";

if (move_uploaded_file($_FILES["fichero"]["tmp_name"], $fileUploadDir.$_FILES["fichero"]["name"])) {
    echo "El fichero " . $_FILES["fichero"]["name"] . " se ha subido correctamente.";
} else {
    echo "Ha ocurrido un error al subir el fichero";
}

?>
```

# Introducción a PHP

## Manejo de formularios - Envío de múltiples ficheros

```
<form action="recibevariosarchivos.php" method="post" enctype="multipart/form-data">
    <p>Ficheros:<br>
    <input type="file" name="ficheros[]" />
    <br><input type="file" name="ficheros[]" />
    <br><input type="file" name="ficheros[]" />
    <br><input type="submit" value="Enviar" />
</p>
</form>
```

# Introducción a PHP

## Manejo de formularios - Envío de múltiples ficheros

```
Array
(
    [ficheros] => Array
        (
            [name] => Array
                (
                    [0] => mensajes.txt
                    [1] => PideTextoMuestra.java
                    [2] => basic (1).ics
                )

            [type] => Array
                (
                    [0] => text/plain
                    [1] => application/octet-stream
                    [2] => text/calendar
                )

            [tmp_name] => Array
                (
                    [0] => /private/var/tmp/phpLWZqx0
                    [1] => /private/var/tmp/phpSKPeQk
                    [2] => /private/var/tmp/phpIgQc6A
                )

            [error] => Array
                (
                    [0] => 0
                    [1] => 0
                    [2] => 0
                )

            [size] => Array
                (
                    [0] => 32
                    [1] => 362
                    [2] => 1791130
                )
        )
)
```

# Introducción a PHP

## Manejo de ficheros

- La función “**readfile()**” lee el fichero especificado y muestra su contenido.
- Devuelve el número de bytes leídos.

```
<?php  
readfile("texto.txt");  
?>
```

# Introducción a PHP

## Manejo de ficheros

- La función “**fopen()**” abre un fichero en el modo especificado.
- Devuelve un manejador de fichero si tiene éxito, o false si se produjo un error.

```
$manejador = fopen("fichero.txt", "r");
```

- “**fread()**” lee un fichero abierto previamente.
- El primer parámetro debe contener el manejador del fichero, y el segundo parámetro el máximo número de bytes a leer.

```
fread($gestor, filesize("fichero.txt"));
```

# Introducción a PHP

## Manejo de ficheros

### Una lista de los modos posibles de fopen() usando mode

mode	Descripción
'r'	Apertura para sólo lectura; coloca el puntero al fichero al principio del fichero.
'r+'	Apertura para lectura y escritura; coloca el puntero al fichero al principio del fichero.
'w'	Apertura para sólo escritura; coloca el puntero al fichero al principio del fichero y trunca el fichero a longitud cero. Si el fichero no existe se intenta crear.
'w+'	Apertura para lectura y escritura; coloca el puntero al fichero al principio del fichero y trunca el fichero a longitud cero. Si el fichero no existe se intenta crear.
'a'	Apertura para sólo escritura; coloca el puntero del fichero al final del mismo. Si el fichero no existe, se intenta crear. En este modo, <a href="#">fseek()</a> solamente afecta a la posición de lectura; las lecturas siempre son pospuestas.
'a+'	Apertura para lectura y escritura; coloca el puntero del fichero al final del mismo. Si el fichero no existe, se intenta crear. En este modo, <a href="#">fseek()</a> no tiene efecto, las escrituras siempre son pospuestas.

# Introducción a PHP

## Manejo de ficheros

- “**fclose()**” se utiliza para cerrar un fichero abierto.

```
fclose($manejador);
```

- “**fgets()**” nos permite leer una línea de un fichero abierto.
- “**feof()**” comprueba si se ha llegado al final del fichero.

```
<?php  
$myfile = fopen("webdictionary.txt", "r") or die("Unable to open file!");  
// Output one line until end-of-file  
while(!feof($myfile)) {  
    echo fgets($myfile) . "<br>";  
}  
fclose($myfile);  
?>
```

# Introducción a PHP

## Manejo de ficheros

- “**fwrite()**” se utiliza para escribir en un fichero.
- El primer parámetro contiene el manejador del fichero y el segundo parámetro la string que se va a escribir.

```
<?php
myfile = fopen("newfile.txt", "w") or die("Unable to open file!");
$txt = "John Doe\n";
fwrite($myfile, $txt);
$txt = "Jane Doe\n";
fwrite($myfile, $txt);
fclose($myfile);
?>
```