

# Desarrollo Web en Entorno Servidor

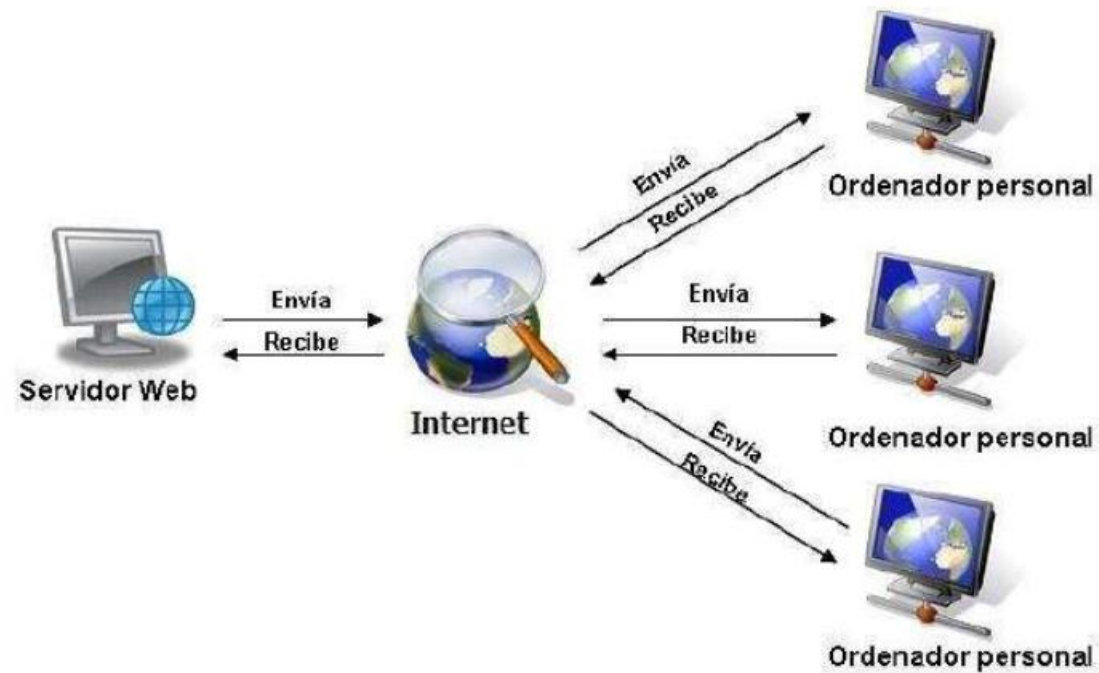
---

Programación en PHP

José A. Lara

# Funcionamiento de Internet

## FUNCIONAMIENTO DE INTERNET





# Funcionamiento de Internet

---

- En primer lugar el usuario abre un navegador y escribe en la URL (Uniform Resource Locator) `http://www.cesurformacion.com`
- El navegador descompone dicha URL en varias partes, como el protocolo que va a ser el HTTP, el servidor, el puerto y la ruta o recurso. En nuestro caso el servidor que está solicitando el usuario es `www.cesurformacion.com`
- Existe otro protocolo que convertirá esta petición a una IP, puesto que todos los dispositivos con conexión a Internet tienen una identificación única (no es del todo así, pero podemos ahora mismo explicarlo así).
- Ese servidor, tendrá instalado un sistema operativo y un servidor que recogerá la petición del cliente, la sabrá entender y analizar y devolverá una respuesta.
- Por último esta respuesta recorre el camino inverso hasta llegar al usuario, al navegador, el cual mostrará el resultado

# Protocolos

---

- Según la norma del grupo de trabajo referente en Internet (IETF), un **protocolo de comunicaciones** es un sistema de reglas que permiten que dos o más entidades de un sistema de comunicación se comuniquen entre ellas para transmitir información por medio de cualquier tipo de variación de una magnitud física.



# Protocolos

---

- Se trata de las reglas o el estándar que define la sintaxis, semántica y sincronización de la comunicación, así como también los posibles métodos de recuperación de errores. Los protocolos pueden ser implementados por hardware, por software, o por una combinación de ambos.
- Un protocolo, al fin y al cabo, podríamos entenderlo como ese idioma que aprendemos, con sus reglas y con su léxico, y que nos permite comunicarnos con diferentes personas.

# Protocolo HTTP

---

- El protocolo de transferencia de hipertexto (HTTP) es uno de los protocolos de aplicación más omnipresentes y ampliamente adoptados en Internet: es el lenguaje común entre clientes y servidores, lo que permite la web moderna. Es el protocolo de comunicación que permite las transferencias de información en la World Wide Web.
- Este protocolo, como muchos protocolos y mecanismos que intervienen en la interacción dentro de Internet es un protocolo de tipo Cliente-Servidor (concepto que también introduciremos en esta unidad) y que con una sintaxis pequeña permite comunicar a un navegador con un servidor de aplicaciones web para devolver resultados como las conocidas “páginas web”.



# Protocolo HTTP

---

- Una de las ventajas o desventajas, dependiendo del punto de vista que estemos trabajando, es que el HTTP como bien indica su definición (Text), es un protocolo textual que nos permite visualizar tanto lo que el navegador está enviando o como la respuesta del servidor (lo veremos más adelante con muchos ejemplos). Por lo tanto no hará falta ningún desencriptador ni ningún mecanismo adicional para que cualquiera, que sepa de la sintaxis de HTTP, pueda interpretar qué está ocurriendo con el mismo.

# Seguridad del protocolo HTTP

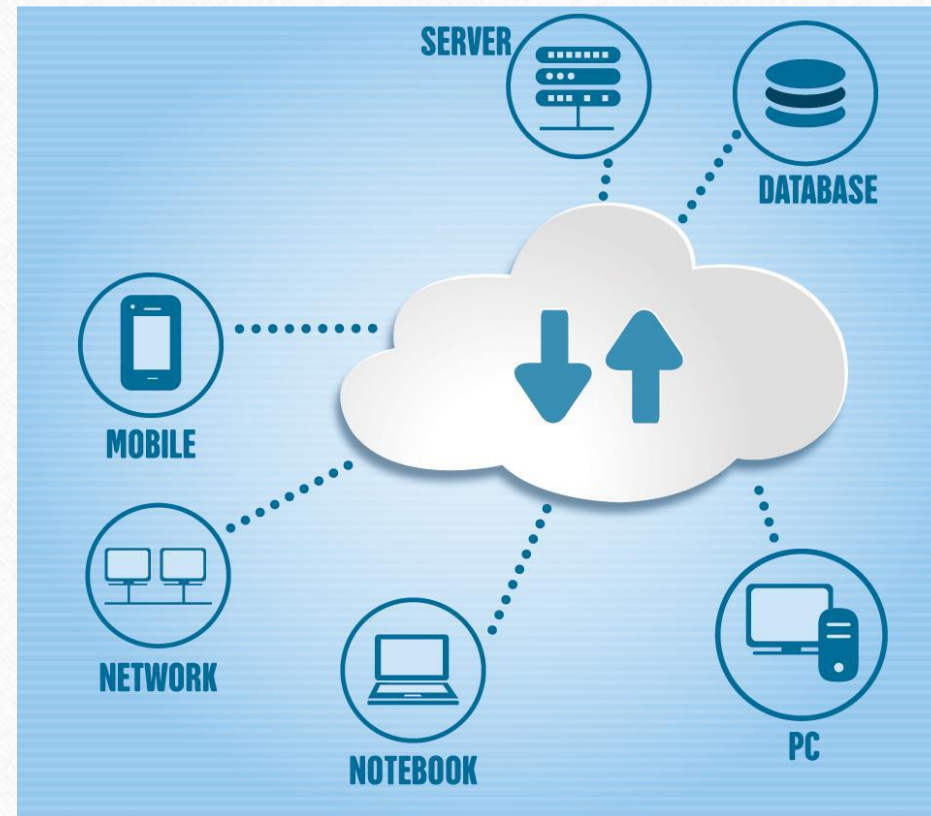
---

- El protocolo es **muy inseguro**, ya que cualquiera puede visualizar y capturar los mensajes que estamos transmitiendo entre nuestra máquina y la remota.
- Imaginemos que estamos realizando una compra online, y como proceso habitual de esa compra nos soliciten introducir la tarjeta de crédito, ese número de tarjeta, que no es ni más ni menos que un campo tipo texto sería visible para cualquier atacante que interceptase la comunicación. Es en este punto donde se utiliza el protocolo HTTPS, protocolo que no vamos a explicar en detalle, y que proporciona un recubrimiento **seguro** de nuestras comunicaciones.



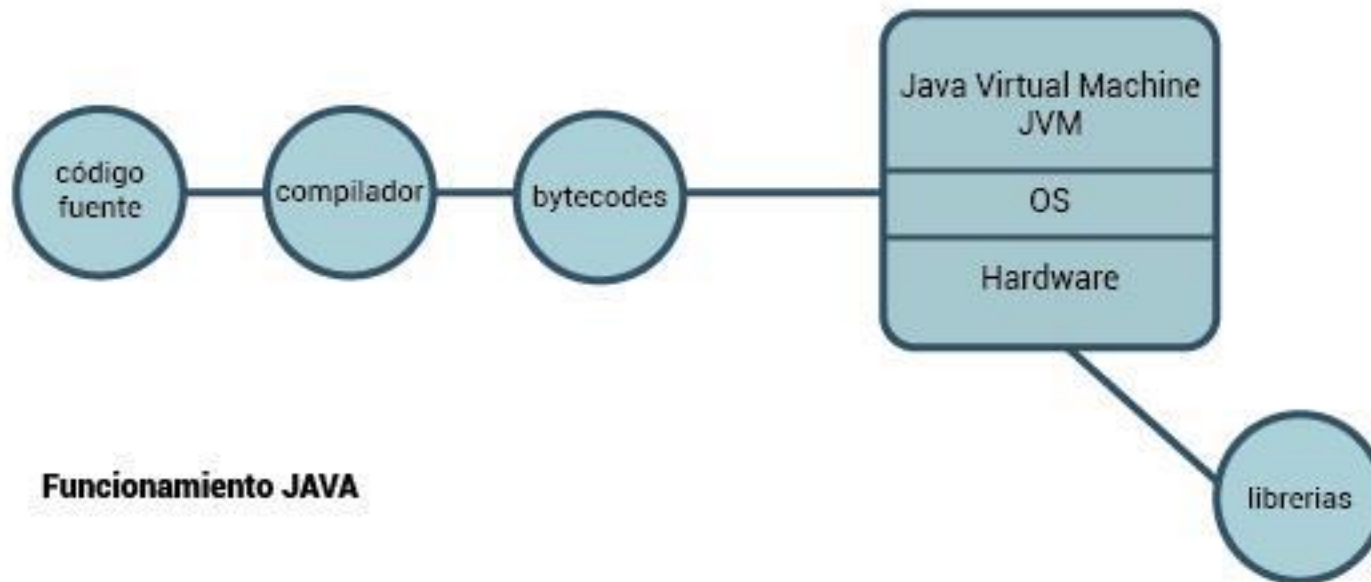
# Cliente - Servidor

---



# Lenguajes de programación

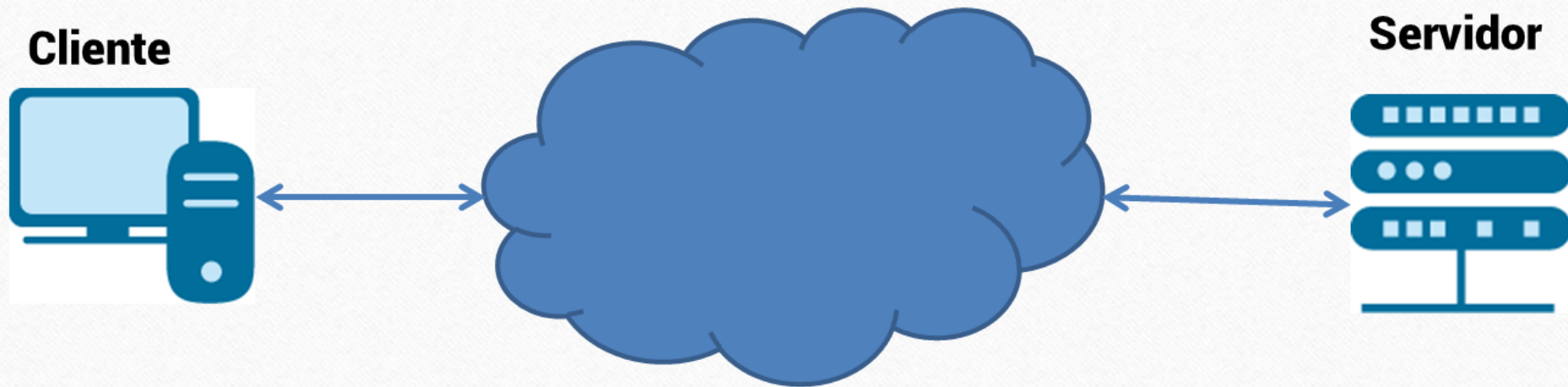
- Compilado frente a interpretado



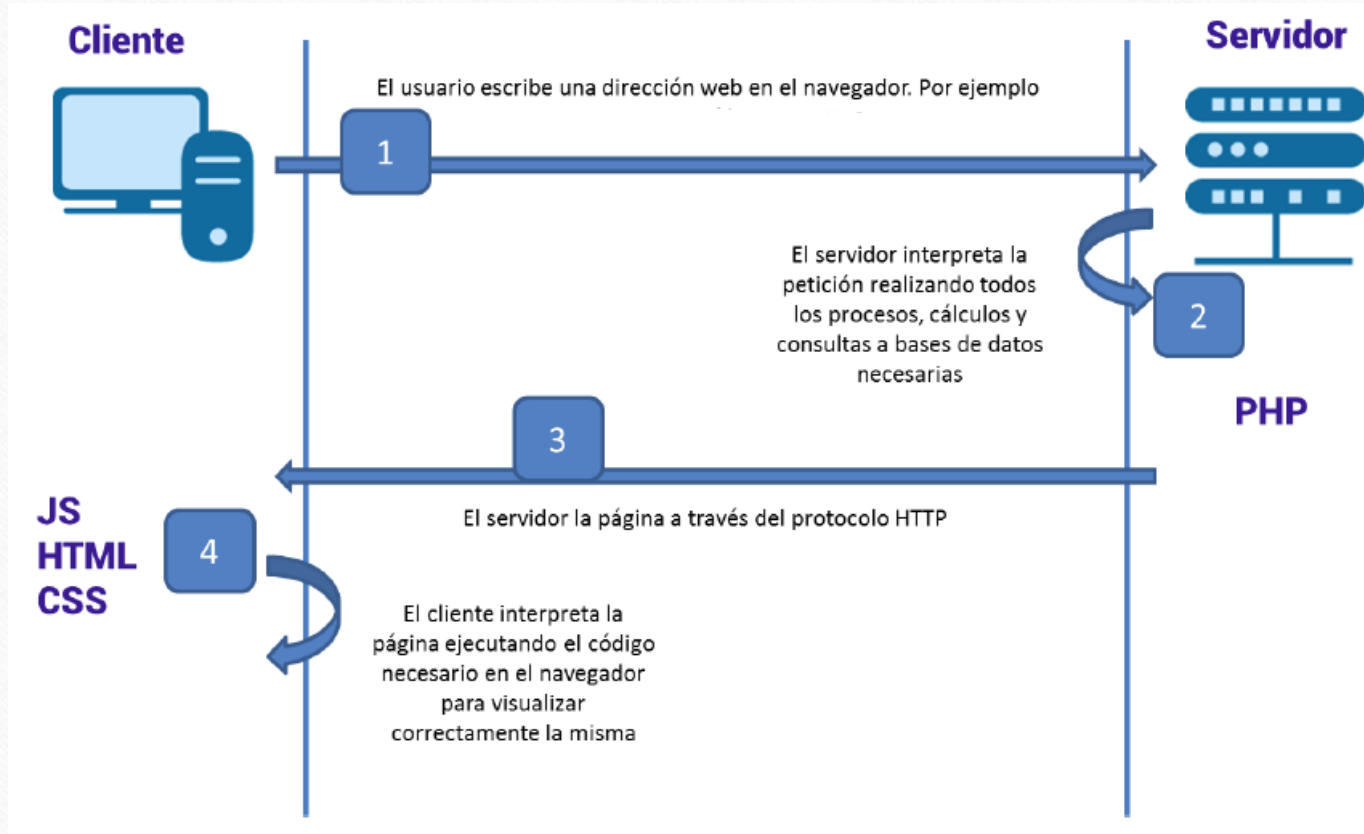


# Lenguajes cliente – Lenguajes servidor

---



# Lenguajes cliente – Lenguajes servidor





# PHP

---

- Según la documentación de fabricante, PHP, acrónimo de "PHP: Hypertext Preprocessor", es un lenguaje de 'scripting' de propósito general y de código abierto que está especialmente pensado para el desarrollo web y que puede ser embebido en páginas HTML. Su sintaxis recurre a C, Java y Perl, siendo así sencillo de aprender. El objetivo principal de este lenguaje es permitir a los desarrolladores web escribir dinámica y rápidamente páginas web generadas; aunque se puede hacer mucho más con PHP. Fue creado originalmente por Rasmus Lerdorf en el año 1995 como solución a una problemática que encontró en el análisis de información a partir de código C.

# Versiones de PHP

---

- PHP ha sido un lenguaje de programación que se ha caracterizado por la estabilidad de sus versiones, al contrario de muchos lenguajes actuales donde el cambio de versión se puede producir en menos de 12 meses. La segunda característica podemos destacar su intento de “compatibilidad hacia atrás” que ha provocado en muchas ocasiones no avanzar en la mejora del lenguaje.
- Hasta su versión 4, el lenguaje se construía de una forma más o menos caótica. Será a partir de esta versión donde el lenguaje alcanza otra dimensión más profesional.



# Versiones de PHP

---

- Será en la versión 5 donde el lenguaje incorpora orientación a objetos y será entonces cuando se comienza a pensar en soluciones más grandes, flexibles y estructuradas. Será entonces cuando muchos frameworks comienzan a surgir teniendo como referencia el lenguaje de programación PHP.
- La versión 6 nunca llegó a utilizarse realmente, ya que el desarrollo sobre la problemática Unicode resultó más compleja de lo que en un principio se planificó. Por lo que PHP pasó directamente a su versión 7 que es la que actualmente utilizamos.

# Frameworks

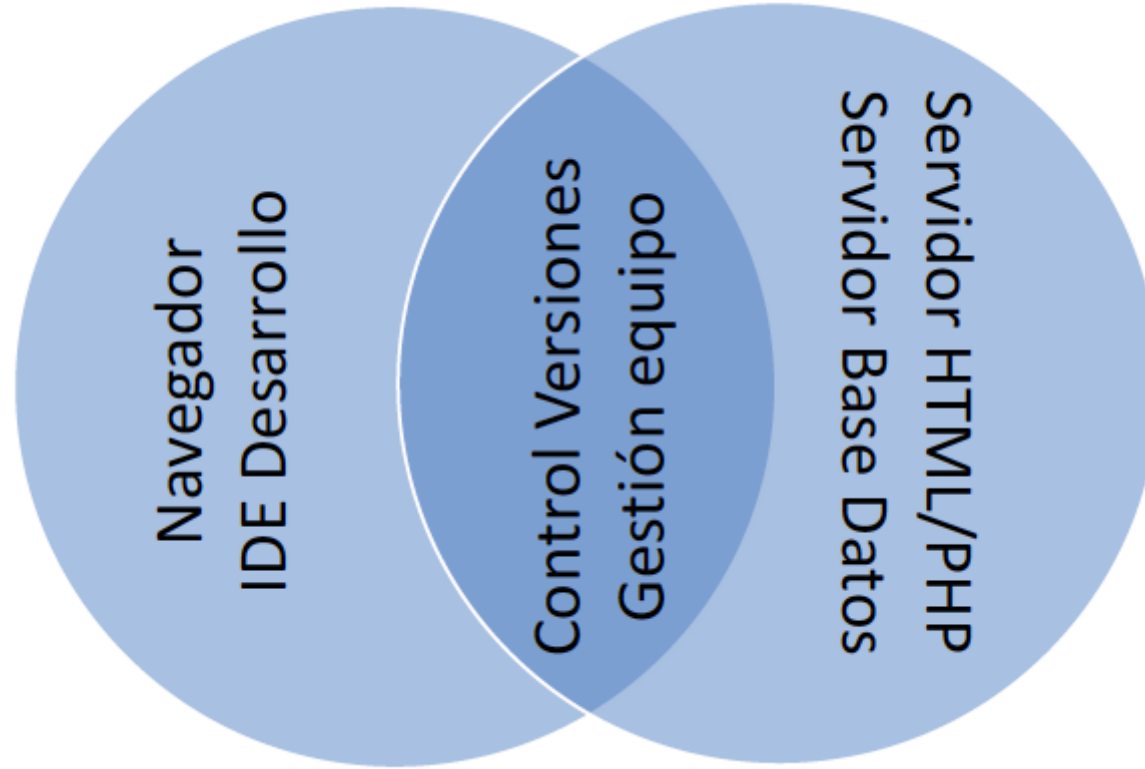
---

- Uno de los grandes éxitos actuales de los lenguajes de programación son los frameworks de desarrollo que nos permiten configurar determinados problemas ya que están ya desarrollados, como es el trabajo con Bases de Datos, la Seguridad, los Usuarios.



# Entorno de trabajo

---



# Entorno de trabajo

---

- **Control de versiones**, herramientas que nos permitirán no solo mantener nuestro código salvado y guardado, sino que además nos permitirá realizar un seguimiento de las diferentes versiones, actualizaciones y planificar y llevar a cabo ramas de trabajo.
- Gestión de equipos, la gestión del trabajo, planificación de las tareas, gestión de los recursos y gestión del tiempo son herramientas muy muy necesarias hoy en día para llevar a cabo nuestra tarea de desarrollo.



# Cliente

---

## Navegador web

- Componente software que se utiliza en el cliente y que permite acceder al contenido ofrecido por los servidores de Internet sin la necesidad de que el usuario instale un nuevo programa.
- En este módulo, Desarrollo de Aplicaciones Web en entorno Servidor, el Navegador es un elemento final, una herramienta en la que visualizar el resultado de nuestro código programado. Es necesario e imprescindible, pero no va a ser el objetivo como lo es en el desarrollo de cliente.
- ¿Qué navegador debo tener instalado?

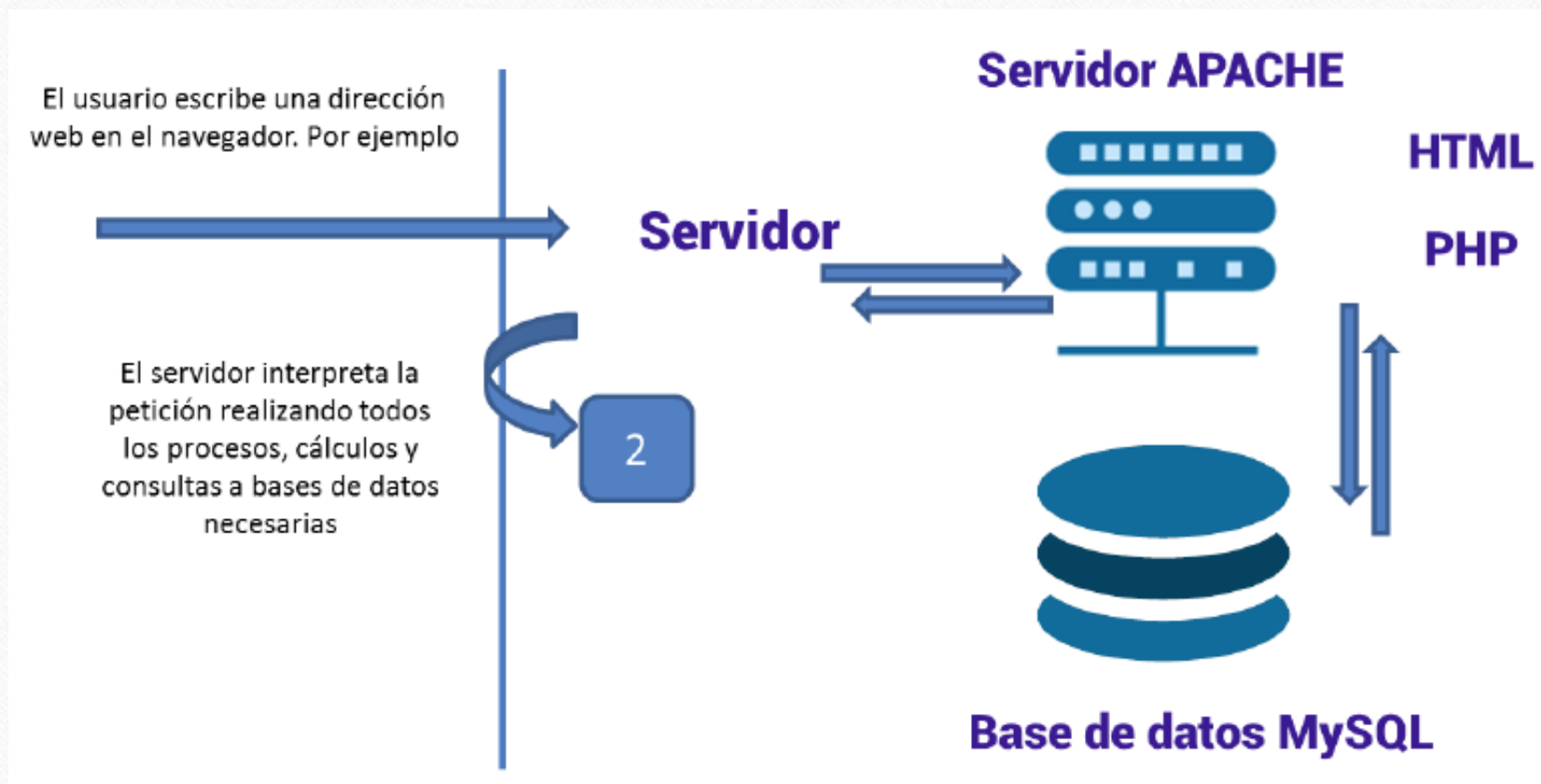
# IDE de desarrollo

---

- El mundo de los editores o IDE ha evolucionado de una forma tan rápida y evolucionada como el resto de tecnología relacionada con el desarrollo.
- Puede ser que todo comenzara con Sublime hace unos años: ligero, flexible y “hackeable”.
- Por último Atom.io de GitHub, que al igual Sublime es ligero, flexible y hackeable.



# Servidor



# Tipos de instalaciones

---

- Nos centraremos en los sistemas, paquetes y elementos que de una forma rápida y sencilla nos preparen un entorno de desarrollo para poder desarrollar.
- Los tipos de instalaciones que nos podemos encontrar son:
  - Sistemas en la nube: (PaaS)
  - Instalaciones locales: (SaaS)



# Tipos de instalaciones

---

- Platform as a Service (PaaS), nos permite tener una serie de recursos, normalmente virtualizados, como espacio en disco duro, memoria RAM, procesador, ... A partir de estos recursos la instalación requiere de un Sistema Operativo y el resto de elementos que vayamos a utilizar para la instalar tanto el servidor Apache como el MySQL.

# Tipos de instalaciones

---

- Software as a Service (SaaS), en este no solo se nos ofrece una serie de recursos hardware en la nube sino una serie de software en la nube, en nuestro caso los dos servidores necesarios. Este sistema es super interesante para nosotros como desarrolladores ya que nos despreocupamos de la instalación y mantenimiento del sistema y nos centramos en lo que realmente nos importa, el desarrollo software.

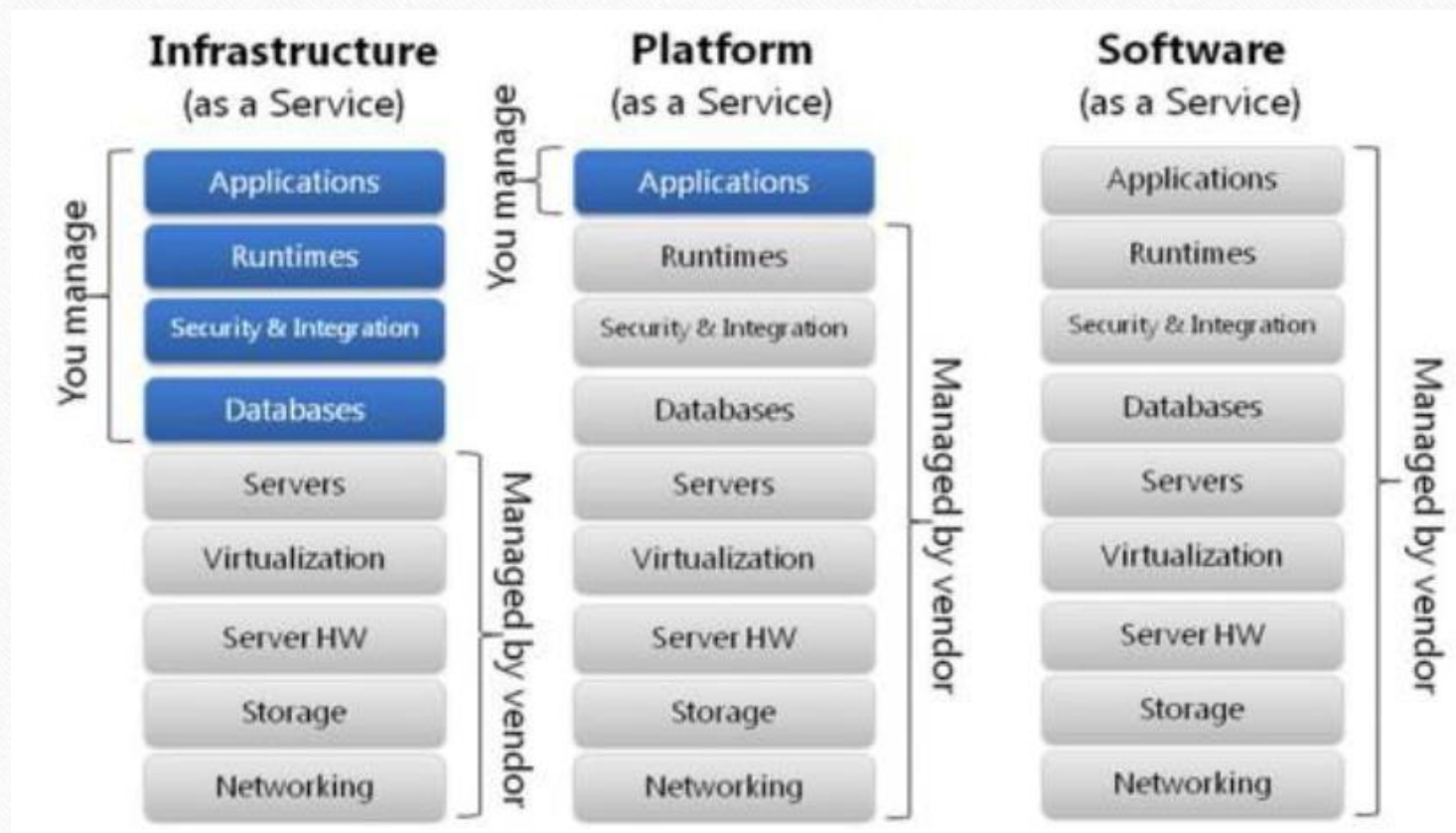


# Tipos de instalaciones

---

- Instalaciones locales mediante software empaquetado como XAMPP, WAMPServer, LAMP, MAMP, ... En este caso, y mediante una instalación lo más desatendida posible, un software nos guía a la instalación de todos los servidores y software necesarios para pasar al desarrollo en local.

# Tipos de instalaciones



# Sistemas locales

---

Ventajas	Desventajas
<ul style="list-style-type: none"><li>• Fácil de instalar y configurar.</li><li>• Total control del sistema.</li><li>• Fácil de mantener y realizar backups.</li></ul>	<ul style="list-style-type: none"><li>• Complicado el trabajo colaborativo</li><li>• Sistema de debug, pero no se puede utilizar como sistema en producción.</li></ul>



# Instalación de nuestros servidores locales

---

- XAMPP (Mac, Windows y Linux)  **XAMPP** Apache + MariaDB + PHP + Perl

<https://www.apachefriends.org/es/index.html>

- MAMP (Mac y Windows)

<https://www.mamp.info/en/windows/>



- WampServer (solo Windows)

<https://www.wampserver.com/en/>



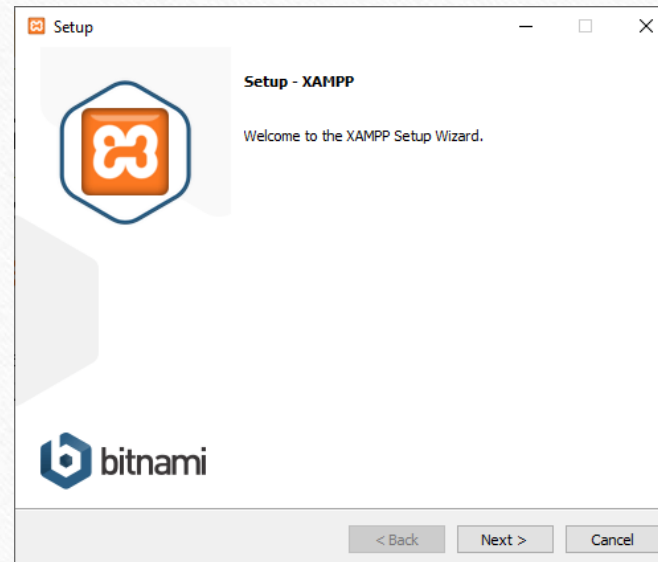
# Instalación de nuestros servidores locales

---

- XAMPP (Mac, Windows y Linux)

 **XAMPP** Apache + MariaDB + PHP + Perl

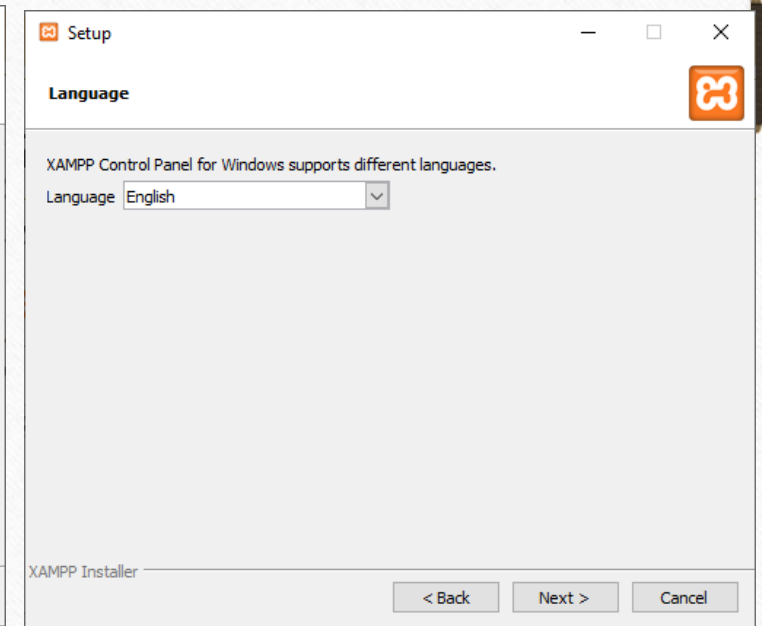
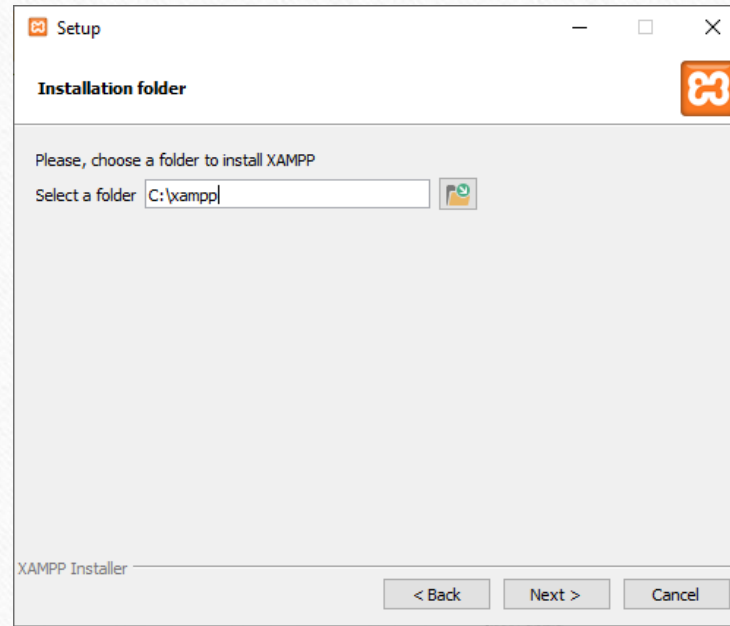
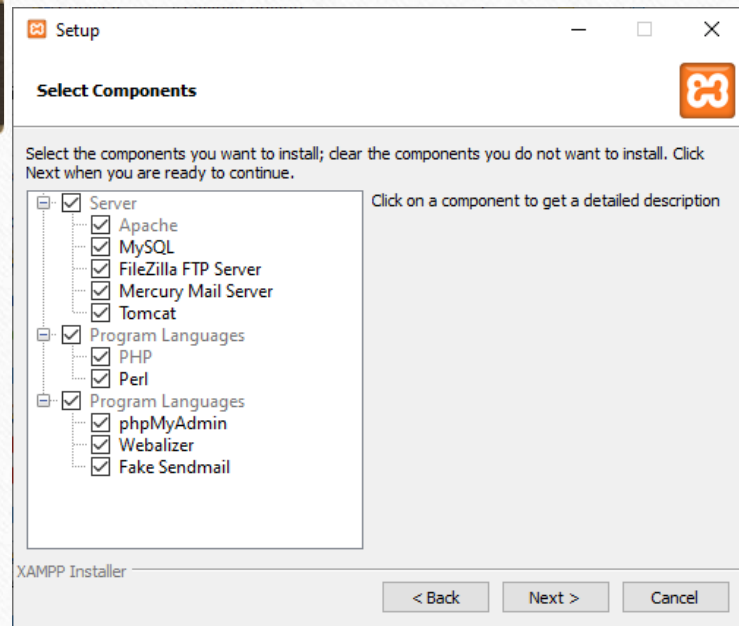
- La instalación de XAMPP es bastante intuitiva. Solamente pulsaremos en los botones de Siguiente



# Instalación de nuestros servidores locales

- XAMPP (Mac, Windows y Linux)

 **XAMPP** Apache + MariaDB + PHP + Perl

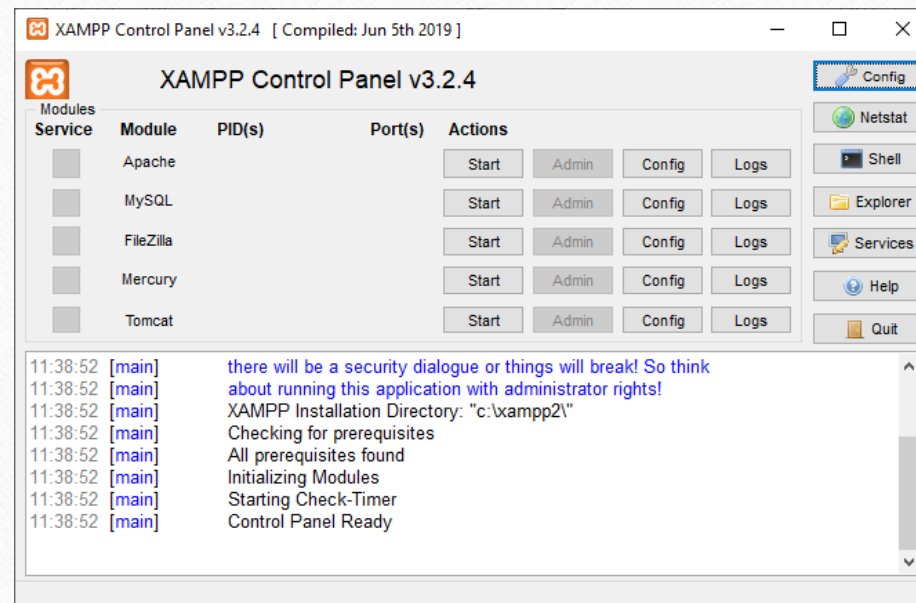




# Instalación de XAMPP

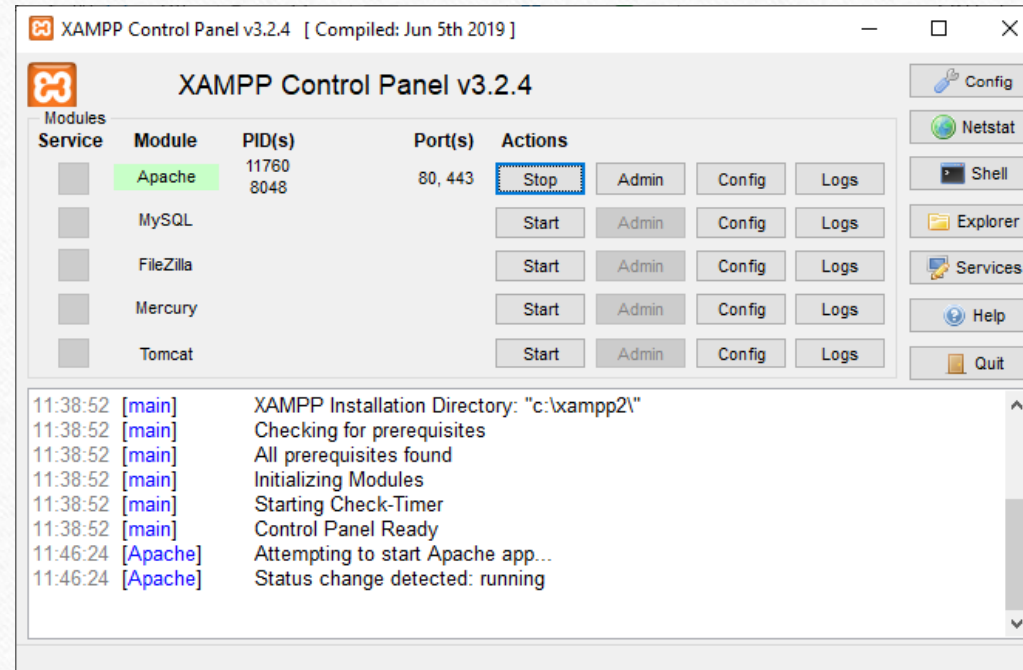
## Inicialización de los servicios: Apache, MySQL

- Una vez instalado XAMPP, para iniciar los servicios accedemos al panel de control de la aplicación



# Instalación de XAMPP

## Inicialización de los servicios: Apache, MySQL



# PHPInfo - PHPMyAdmin

- Si pulsamos sobre el botón **Admin** en el servicio Apache, nos mostrará la siguiente página web:




The screenshot shows the XAMPP control panel interface. At the top is a dark blue navigation bar with links: Apache Friends, Applications, FAQs, HOW-TO Guides, PHPInfo, and phpMyAdmin. Below this is a header section with the XAMPP logo (an orange square with a white 'X') and the text 'XAMPP Apache + MariaDB + PHP + Perl'. The main content area has a heading 'Welcome to XAMPP for Windows 7.4.9' followed by a paragraph: 'You have successfully installed XAMPP on this system! Now you can start using Apache, MariaDB, PHP and other components. You can find more info in the [FAQs](#) section or check the [HOW-TO Guides](#) for getting started with PHP applications.' Below this is another paragraph: 'XAMPP is meant only for development purposes. It has certain configuration settings that make it easy to develop locally but that are insecure if you want to have your installation accessible to others. If you want have your XAMPP accessible from the internet, make sure you understand the implications and you checked the [FAQs](#) to learn how to protect your site. Alternatively you can use [WAMP](#), [MAMP](#) or [LAMP](#) which are similar packages which are more suitable for production.' Then, a line of text says 'Start the XAMPP Control Panel to check the server status.' Below that is a section titled 'Community' with the text 'XAMPP has been around for more than 10 years – there is a huge community behind it. You can get involved by joining our Forums,'.



# PHPInfo - PHPMyAdmin

- Pulsamos sobre PHPInfo:

PHP Version 7.4.9	
	
System	Windows NT DESKTOP-TDA64PS 10.0 build 19041 (Windows 10) AMD64
Build Date	Aug 4 2020 11:45:36
Compiler	Visual C++ 2017
Architecture	x64
Configure Command	cscript /nologo /e:jscript configure.js "--enable-snapshot-build" "--enable-debug-pack" "--with-pdo-oci=c:\php-snap-build\deps_aux\oracle\x64\instantclient_12_1\sdk,shared" "--with-oci8-12c=c:\php-snap-build\deps_aux\oracle\x64\instantclient_12_1\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	C:\WINDOWS
Loaded Configuration File	C:\xampp2\php\php.ini
Scan this dir for additional .ini files	(none)
Additional .ini files parsed	(none)
PHP API	20190902
PHP Extension	20190902
Zend Extension	320190902
Zend Extension Build	API320190902,TS,VC15
PHP Extension Build	API20190902,TS,VC15
Debug Build	no
Thread Safety	enabled
Thread API	Windows Threads
Zend Signal Handling	disabled
Zend Memory Manager	enabled
Zend Multibyte Support	provided by mbstring

# PHPInfo - PHPMyAdmin












- Si hemos inicializado el servicio de MySQL y pulsamos sobre PHPMyAdmin:

The screenshot displays the PHPMyAdmin web interface. On the left, a sidebar shows a tree view of databases: 'Nueva', 'information\_schema', 'mysql', 'performance\_schema', 'phpmyadmin', and 'test'. The main content area is divided into several panels. The top panel, 'Configuraciones generales', shows the 'Server connection collation' set to 'utf8mb4\_unicode\_ci'. Below it, the 'Configuraciones de apariencia' panel shows the 'Idioma - Language' set to 'Español - Spanish' and the 'Tema' set to 'pmahomme'. On the right, the 'Servidor de base de datos' panel lists server details: 'Servidor: 127.0.0.1 via TCP/IP', 'Tipo de servidor: MariaDB', 'Conexión del servidor: No se está utilizando SSL', 'Versión del servidor: 10.4.14-MariaDB - mariadb.org binary distribution', 'Versión del protocolo: 10', 'Usuario: root@localhost', and 'Conjunto de caracteres del servidor: UTF-8 Unicode (utf8mb4)'. The 'Servidor web' panel lists: 'Apache/2.4.46 (Win64) OpenSSL/1.1.1g PHP/7.4.9', 'Versión del cliente de base de datos: libmysql - mysqlnd 7.4.9', 'extensión PHP: mysqli, curl, mbstring', and 'Versión de PHP: 7.4.9'. The 'phpMyAdmin' panel at the bottom right lists: 'Acerca de esta versión: 5.0.2 (actualizada)', 'Documentación', 'Página oficial de phpMyAdmin', 'Contribuir', and 'Obtener ayuda'.

# Carpeta htdocs

---

- Un archivo en PHP es en realidad una página construida en HTML a la que se le han añadido trozos (scripts) escritos en PHP. Al crear un archivo en PHP lo guardaremos con la extensión .php. La carpeta "htdocs" será una de nuestras principales herramientas de trabajo. Es en esta carpeta donde guardaremos todos los archivos y sitios que hagamos utilizando PHP.

 anonymous	23/10/2019 17:11
 apache	23/10/2019 17:11
 cgi-bin	23/10/2019 17:15
 contrib	23/10/2019 17:11
 FileZillaFTP	23/10/2019 17:15
 htdocs	15/07/2020 11:28
 img	23/10/2019 17:11
 install	23/10/2019 17:15
 licenses	23/10/2019 17:11
 locale	23/10/2019 17:11
 mailoutout	23/10/2019 17:11



# Editores de código

---

- Sublime Text (Mac, Windows y Linux)

<https://www.sublimetext.com/>

- Notepad++ (Windows)

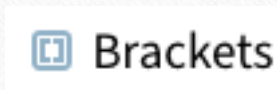
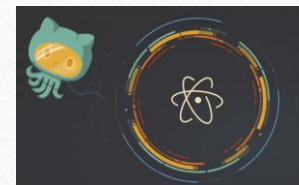
<https://notepad-plus-plus.org/>

- Atom (Mac, Windows y Linux)

<https://atom.io/>

- Adobe Brackets (Mac, Windows y Linux)

<http://brackets.io/index.html>



# Hola mundo

---

En Sublime Text -> (Ctrl + Shift + P) -> Package Control: Install Package

(<http://www.gitmedio.com/gitmedio/11-plugins-imprescindibles-para-sublime-text-3/>)

(<https://blog.pleets.org/article/16-plugins-de-sublime-text>)

```
<?php
```

```
    echo "Hola mundo";
```

```
?>
```

# Comentarios

---

```
<?php
```

```
//Comentario
```

```
# Comentario
```

```
/*
```

```
    Comentario
```

```
*/
```

```
echo "Hola Mundo";
```

```
?>
```

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
    <meta charset="UTF-8">
```

```
    <title>Document</title>
```

```
</head>
```

```
<body>
```

```
    <h1>
```

```
        <?php
```

```
            echo "Hola
```

```
Mundo";
```

```
        ?>
```

```
    </h1>
```

```
</body>
```



# Variables

---

```
<?php
    $nombre = "Jose";
    $otro = 'Ose';
    $numero = 7;
    $numero_decimal = 7.7;
    $verdadero = false;
    echo 'Hola, $nombre';
    echo "Hola, $nombre";
    echo 'Hola, ' . $nombre;
    echo gettype($nombre);
?>
```

# Constantes

---

```
<?php  
define('PI', 3.14);  
echo PI;  
?>
```

# Sentencia if

---

```
<?php
$edad=16;
$nombre='Pepe';
if($edad>=18 && $nombre == 'Pepe'){
    echo '<h1>Mayor de edad</h1>';
} else if ($edad<18 || $nombre != 'Pepe'){
    echo '<h1>Menor de edad</h1>';
} else {
    echo '<h1>No puede acceder</h1>';
}
?>
```



# Operadores

---

- **Aritméticos**

+ Suma  
- Resta  
\* Multiplicación  
/ División

- **Asignación**

=  
+=  
-=  
\*=  
/=

- **Comparación**

==  
===  
!=, <>  
!=  
>  
<  
>=  
<=

- **Lógicos**

and, &&  
or, ||  
xor  
!

- **Incremento**

++\$x  
\$x++  
--\$x  
\$x--

- **De cadenas**

.  
.=

# Estructura switch

---

```
<?php
$mes='Febrero';
switch($mes){
    case 'Enero':
        echo 'Feliz año nuevo';
        break;
    case 'Diciembre':
        echo 'Feliz Navidad';
        break;
    default:
        echo 'Saludos';
        break;
}
?>
```

# Shorthand if

---

```
<?php
    $edad=20;
    $edad= (isset($edad)?$edad:'El usuario no ha escrito la edad');
    echo 'Edad: '.$edad;
?>
```



# Bucle while

---

```
<?php
    $x=20;
    while($x>=1){
        echo $x . '<br>';
        $x--;
    }
?>
```

# Bucle for

---

```
<?php
    for ($i=1; $i<=10; $i++){
        echo $i." - Hola <br/>";
    }
?>
```

# Bucle do-while

---

```
<?php
    $i=1;
    do{
        echo $i. '<br/>';
        $i++;
    } while($i<=10);
?>
```



# Break y Continue

```
<?php
    $paises = array(
        'España', 'Portugal', 'Francia', 'Italia',
        'Alemania', 'Reino Unido', 'Irlanda'
    );

    foreach($paises as $pais){
        echo $pais . '<br/>';
        if ($pais=='España'){
            break;
        }
    }
?>
```

```
<?php
    $paises = array(
        'España', 'Portugal', 'Francia', 'Italia',
        'Alemania', 'Reino Unido', 'Irlanda'
    );

    foreach($paises as $pais){
        if ($pais=='España'){
            continue;
        }
        echo $pais . '<br/>';
    }
?>
```

# Ejercicios

---

1. Hacer un programa que sume dos variables que almacenan dos números distintos.
2. hacer un programa que muestre en pantalla información de PHP con la función `phpinfo()`. Muestre la información centrada horizontalmente en la pantalla.
3. Mostrar en pantalla una tabla de 10 por 10 con los números del 1 al 100. Colorear las filas alternando gris y blanco. Además, el tamaño será una constante: `define(TAM, 10)`.
4. Crear un select en HTML que tenga los números del 1 al 10 como opciones. El número de valores del select (10) estará definido en una constante.

# Ejercicios

---

5. Mostrar las tablas de multiplicar del 1 al 10. Utilizar el bucle for.
6. Crearemos una tabla de valores de seno y coseno de -1 a 1 en incrementos de 0.1. Los valores negativos que resulten los queremos mostrar en rojo, y los valores positivos en azul.
7. Mostrar el mayor valor de 3 variables numéricas (Ejemplo de mensaje: La variable \$a tiene el valor mayor: 10).
8. Mostrar las tablas de multiplicar del 1 al 10 utilizando el bucle while.



# Arrays

---

```
<?php
    $semana = array('Lunes','Martes','Miercoles','Jueves','Viernes','Sabado','Domingo');
    echo $semana[1] . '<br/>';
    $semana[7] = 'Jose';
    echo $semana[7] . '<br/>';
    $otro = ['Lunes','Martes','Miercoles','Jueves','Viernes','Sabado','Domingo'];
    $otromas = array('Lunes','Martes','Miercoles','Jueves','Viernes','Sabado','Domingo', array('Mes','Año'));
    echo $otromas[7][0];
?>
```

# Arrays asociativos

---

```
<?php
    $alex = array('telefono' => '687987438', 'edad' => 25, 'apellido' => 'Pérez', 'pais' => 'España');
    echo $alex['telefono'] . "<br/>";
    echo $alex['edad'] . "<br/>";
    echo $alex['apellido'] . "<br/>";
    echo $alex['pais'] . "<br/>";
?>
```

# Arrays multidimensionales

---

```
<?php
    $amigos = array(
        array('Alejandro',20),
        array('Pepe',25),
        array('Maria', 23)
    );
    echo $amigos[0][0];
?>
```



# Más sobre Arrays

---

```
<?php
$meses = array(
    'Enero', 'Febrero', 'Marzo', 'Abril',
    'Mayo', 'Junio', 'Julio', 'Agosto',
    'Septiembre', 'Octubre', 'Noviembre', 'Diciembre'
);
// echo count($meses);
$ultimo_mes=count($meses)-1;
echo $meses[$ultimo_mes];
?>
```

# Recorrer Arrays

```
<?php
    $meses = array(
        'Enero', 'Febrero', 'Marzo', 'Abril',
        'Mayo', 'Junio', 'Julio', 'Agosto',
        'Septiembre', 'Octubre', 'Noviembre', 'Diciembre'
    );
?>
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Meses del año</title>
</head>
<body>
    <h1>Meses del año</h1>
    <ul>
        <?php
            foreach ($meses as $mes) {
                echo '<li>' . $mes . '</li>';
            }
        ?>
    </ul>
</body>
</html>
```

# Ordenar arrays

```
<?php
    $meses = array(
        'Enero', 'Febrero', 'Marzo', 'Abril',
        'Mayo', 'Junio', 'Julio', 'Agosto',
        'Septiembre', 'Octubre', 'Noviembre', 'Diciembre'
    );
    //sort($meses);
    //rsort($meses);
?>
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Meses del año</title>
</head>
<body>
    <h1>Meses del año</h1>
    <ul>
        <?php
            foreach ($meses as $mes) {
                echo '<li>' . $mes . '</li>';
            }
        ?>
    </ul>
</body>
</html>
```



# Funciones sobre arrays

---

- count

```
<?php
    $a[0] = 1;
    $a[1] = 3;
    $a[2] = 5;
    $resultado = count($a);
    echo $resultado;
    // $resultado == 3
?>
```

# Funciones sobre arrays

---

- asort

```
<?php
    $fruits = array("d" => "lemon", "a" => "orange", "b" => "banana", "c"
        => "apple");
    asort($fruits);
    foreach ($fruits as $key => $val) {
        echo "$key = $val\n";
    } /* EL RESULTADO ES c = apple b = banana d = lemon a = orange */
?>
```

# Funciones sobre arrays

---

- shuffle

```
<?php
    $números = range(1, 20);
    shuffle($números);
    foreach ($números as $número) {
        echo "$número ";
    }
?>
```



# Funciones sobre arrays

---

- array\_merge

```
<?php
$array1 = array("color" => "red", 2, 4);
$array2 = array("a", "b", "color" => "green", "shape" => "trapezoid"
, 4);
$resultado = array_merge($array1, $array2);
print_r($resultado);

/* EL RESULTADO ES Array ( [color] => green [0] => 2 [1] => 4 [2] =>
a [3] => b [shape] => trapezoid [4] => 4
) */

?>
```

# Funciones para visualizar arrays

---

- OPCIÓN 1: Utilizando estructuras de control tanto *for* para arrays indexados con integer o indexados automáticamente, como utilizando *foreach*, que nos sirve para cualquier tipo de arrays. Esta opción nos permite presentar arrays tanto en modo *debug* como en producción.

# Funciones para visualizar arrays

---

- OPCIÓN 2: Utilizando las funciones *print\_r* o *var\_dump*, ambas opciones nos muestran por pantalla de forma rápida un array de cualquier tipo, pero sólo lo podemos utilizar en modo debug.

```
<?php
    $a = array(1, 2, array("a", "b", "c"));
    var_dump($a);
    /* EL RESULTADO ES array(3) { [0]=> int(1) [1]=>int(2) [2]=>
    array(3) { [0]=> string(1) "a" [1]=> string(1) "b" [2]=> string(1)
    "c" } } */
?>
```



# Funciones para visualizar arrays

---

- **echo**
  - Muestra una o más cadenas separadas por comas
  - No tiene un valor de retorno
  - Ejemplo:

```
echo "Cadena 1", "Cadena 2";
```

# Funciones para visualizar arrays

---

- **print**
  - Muestra solo una simple cadena
  - Devuelve 1, por lo cual puede ser usada en una expresión
  - Ejemplo:

```
print "Hello";  
if($expresion && print "Cadena"){  
    // Hacer algo  
}
```

# Funciones para visualizar arrays

---

- **print\_r()**
  - Muestra una representación más entendible de un solo valor cualquiera
  - No solo acepta cadenas, también acepta arrays y objetos formateándolos para ser visualizados de una forma más entendible
  - Puede devolver su valor de salida como un valor de retorno si le pasa true como su segundo argumento
  - Útil para la depuración



# Funciones para visualizar arrays

---

- **print\_r()**

- Ejemplo: 

```
$a = array ('a' => 'manzana', 'b' => 'banana', 'c' => array ('x', 'y', 'z'));  
print_r ($a);
```

- Devolverá:

```
Array  
(  
    [a] => manzana  
    [b] => banana  
    [c] => Array  
        (  
            [0] => x  
            [1] => y  
            [2] => z  
        )  
)
```

# Funciones para visualizar arrays

---

- **var\_dump()**
  - Muestra una representación más entendible de un valor cualquiera o más separados por comas
  - No solo acepta cadenas, también acepta arrays y objetos formateándolos para ser visualizados de una forma más entendible
  - Usa un formato diferente al anterior print\_r(), por ejemplo, también muestra el tipo del valor
  - Útil para la depuración
  - No tiene un valor de retorno

# Funciones para visualizar arrays

---

- **var\_dump()**

- Ejemplo:

```
$a = array(1, 2, array("a", "b", "c"));  
var_dump($a);
```

- Devolverá:

```
array(3) {  
    [0]=>  
    int(1)  
    [1]=>  
    int(2)  
    [2]=>  
    array(3) {  
        [0]=>  
        string(1) "a"  
        [1]=>  
        string(1) "b"  
        [2]=>  
        string(1) "c"  
    }  
}
```



# Funciones para visualizar arrays

---

- **var\_export()**
  - Muestra una representación más entendible y ejecutable de un valor cualquiera
  - No solo acepta cadenas, también acepta arrays y objetos formateándolos para ser visualizados de una forma más entendible
  - Usa un formato de salida diferente de `var_dump()` y `print_r()`, la salida es un código de PHP válido
  - Útil para la depuración
  - Puede devolver su valor de salida como un valor de retorno si le pasa `true` como su segundo argumento

# Funciones para visualizar arrays

---

- **var\_export()**

- Ejemplo:

```
class A {  
    public $var;  
}
```

```
$a = new A;  
$a->var = 5;
```

```
var_export($a);
```

Devolverá:

```
A::__set_state(array(  
    'var' => 5,  
))
```

# Funciones para visualizar arrays

---

```
<?php
    $texto='Jose';
    $numero=10;
    $numero2='5';
    $arreglo = array('Jose', 'Pepe', 'Juan', 'Maria');
    $array_asoc = array('nombre' => 'Carlos', 'edad' => 20);
    $booleano = false;

    print_r($numero);
?>
```



# Ejercicios

---

1. Almacenar en un vector los 10 primeros números pares. Imprimir cada uno en una línea.
2. Escriba un programa que muestre una tirada de dado al azar y escriba en letras el valor obtenido. (Usar `rand(num_min, num_max)`).
3. Escriba un programa que muestre una tirada de un número de dados al azar entre 2 y 7 (dados) e indique los valores.
4. Escriba un programa que llene con números aleatorios (entre el -5 y el 5) una matriz de  $5 * 6$  y que imprima cuántos de los números almacenados son ceros, cuántos positivos y cuántos negativos.

# Ejercicios

Queremos almacenar en una matriz el número de alumnos con el que cuenta una academia, ordenados en función del nivel y del idioma que se estudia. Tendremos 3 filas que representarán al Nivel básico, medio y de perfeccionamiento y 4 columnas en las que figurarán los idiomas (0 = Inglés, 1 = Francés, 2 = Alemán y 3 = Ruso). Se pide realizar la declaración de la matriz y asignarle los valores indicados en la siguiente imagen a cada elemento de las siguientes maneras (crea un archivo PHP por cada una de estas maneras):

1. Con una sintaxis basada exclusivamente en índices, y mostrar por pantalla los alumnos que existen en cada nivel e idioma.
2. Con una sintaxis basada en el uso anidado de la palabra array, y mostrar por pantalla los alumnos que existen en cada nivel e idioma.
3. Con una sintaxis que combine el uso de array y el uso de índices, y mostrar por pantalla los alumnos que existen en cada nivel e idioma.

1	14	8	3
6	19	7	2
3	13	4	1



# Ejercicios

---

- Crear un array asociativo con la siguiente estructura:

Juan => [altura=>175,pelo=>rubio,ojos=>azules]

María=>[altura=>168,pelo=>castaña,ojos=>marrones]

Pedro=>[altura=>180,pelo=>castaño,ojos=>verdes]

Mostrar por pantalla:

la altura de Juan

los ojos de María

el pelo de Pedro



# Ejercicios

---

Generar en PHP un array bidimensional de 6x4 tal que cada fila contenga los sucesivos múltiplos de 3, desde el 3 en adelante. Mostrar el array en una tabla HTML.

Crear un array de 6x6 con números enteros en PHP, de forma que muestre por pantalla el array en forma de tabla HTML y el número de la fila cuya suma sea mayor que las demás.