

# Bienvenidos

# Ajax / fetch / axios

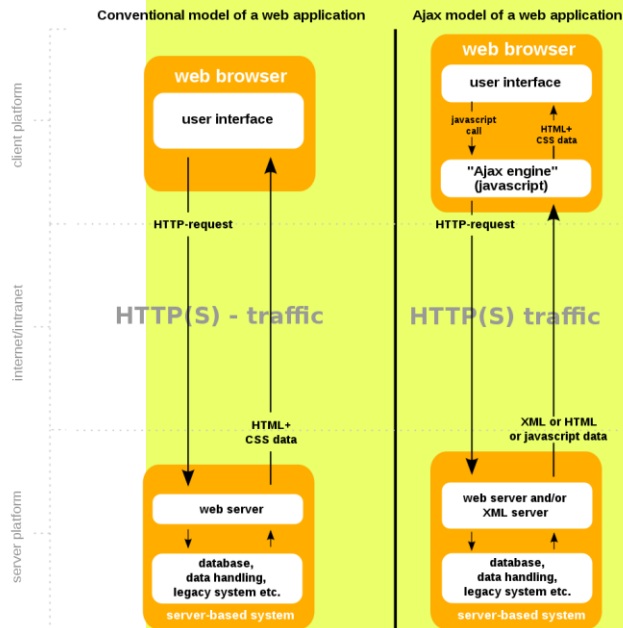
# Ajax, Fetch y Axios

Ajax es una técnica de programación utilizada para construir páginas web más complejas y dinámicas, utilizando una tecnología conocida como XMLHttpRequest. Las siglas de Ajax responden a Asynchronous JavaScript and XML, o Javascript y XML Asíncrono.

Ajax permite modificar partes específicas del DOM de una página HTML sin la necesidad de refrescar la página entera. También permite trabajar asincrónicamente, lo que significa que el código seguirá corriendo mientras esa parte de la página de tu sitio web intenta recargarse.

Fetch es una API moderna que proporciona una forma más fácil y flexible de realizar solicitudes HTTP que Ajax.

Axios es una biblioteca de cliente HTTP para JavaScript que se basa en promesas para enviar peticiones HTTP.



**Clase 03.** Programación Server-Side  
Introducción a PHP

# Aplicaciones Móviles Y Cloud Computing

# Temario

02

## Programación Asíncrona

- ✓ Programación asíncrona en Javascript
- ✓ Ajax, Fetch, Axios

03

## Programación Server Side Principios de PHP

- ✓ [Programación Server Side](#)
- ✓ [Principios de PHP](#)

04

## Aplicaciones web con PHP

- ✓ Aplicaciones web PHP
- ✓ Sesiones / JWT



# Objetivos de la clase

- Comprender el concepto de programación server side
- Conocer aspectos básicos del protocolo HTTP
- Conocer las bases de PHP



# Programación Server Side

# HTTP

HTTP (Hyper Text Transfer Protocol) refiere a un **protocolo**, el cual es un conjunto de reglas que permite la comunicación entre dos o más sistemas. Gracias a este protocolo, las computadoras saben comunicarse entre sí y permiten comunicarse con servidores para obtención de datos. Lo podemos ver en todas las páginas que visitamos.



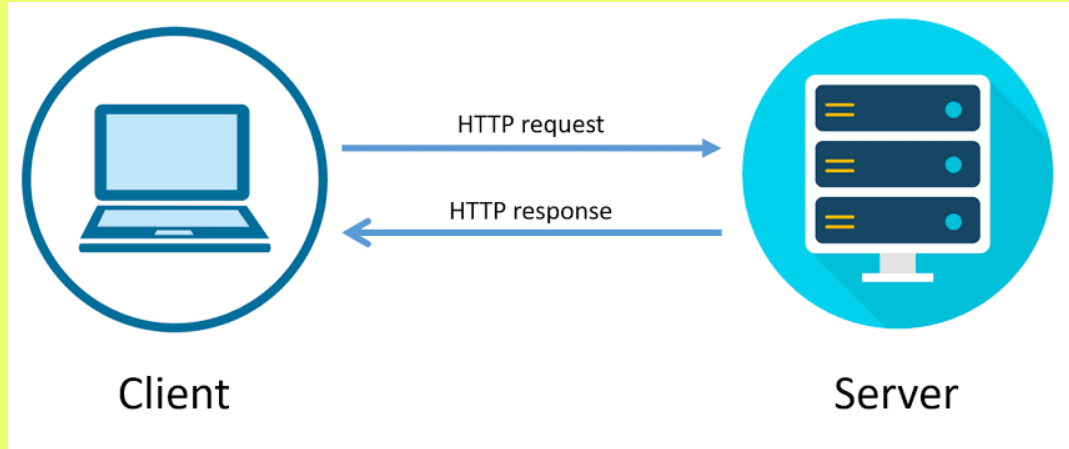
Universidad  
Provincial del Sudoeste  
*Promoviendo el Desarrollo Armónico de la Región*



# HTTP

## ¿Cómo funciona?

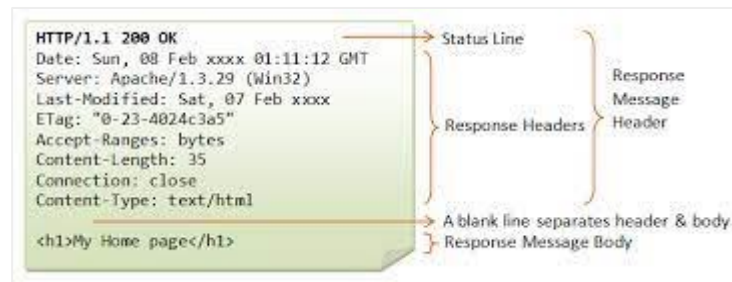
Se basa en un modelo de **petición - respuesta**, de manera que el **cliente** tiene que hacer una petición de información, y entonces **el servidor** se encarga de responder con dicha información.



# ¿Qué tipo de información pedimos?

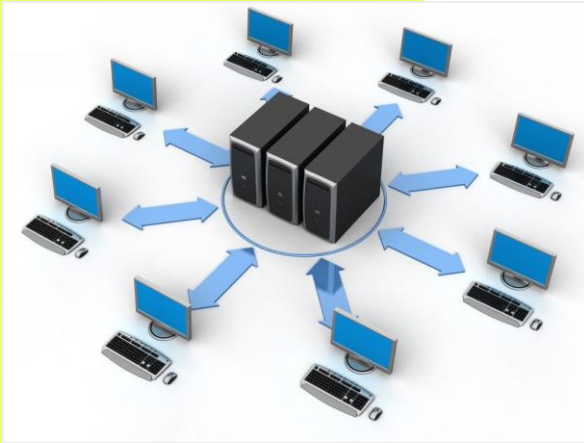
Al hacer una petición, estamos solicitando al servidor ciertos recursos. Estos pueden ser:

- ✓ Un dato, como un nombre, una fecha, una edad...
- ✓ Información más compleja, como una imagen o un vídeo.
- ✓ Un archivo para descargar
- ✓ ¡Incluso una página web completa!



Ejemplo de la respuesta de un servidor, devolviendo una etiqueta h1

# Maneja múltiples peticiones



Cuando programamos nuestro servidor, lo hacemos para escuchar peticiones. La pregunta es: **¿de quién escuchar las peticiones?**

¡Bajo su **configuración por defecto**, un servidor puede escuchar múltiples peticiones de múltiples clientes al mismo tiempo!





# ¿Y si el servidor se apaga?

Sabemos que un programa se inicia, hace sus operaciones, y luego finaliza.

¿Qué tiene de diferente el servidor? ¿Cómo puede recibir peticiones en diferentes períodos de tiempo sin finalizar? 🤖



# ¡Importante!

El cliente siempre es quien hace las peticiones (**requests**) y el servidor siempre será quien hace las respuestas (**responses**) . Cuando hacemos frontend, somos clientes. Cuando desarrollamos backend, nos encargamos de las respuestas.

# Configurando un servidor

```
const http = require('http');

const server = http.createServer((request,response)=>{
  response.end("¡Mi primer hola mundo desde backend!")
})

server.listen(8080,()=>{
  console.log("Listening on port 8080")
})
```



# Servidor con Express js

# ¿Qué es Express js?

Express js es un framework minimalista que permitirá desarrollar servidores más complejos.

Express nos facilitará:

- ✓ Utilizar diferentes rutas para las peticiones.
- ✓ Mejorar la estructura de nuestro proyecto.
- ✓ Manejar funcionalidades más complejas y utilización de middlewares.





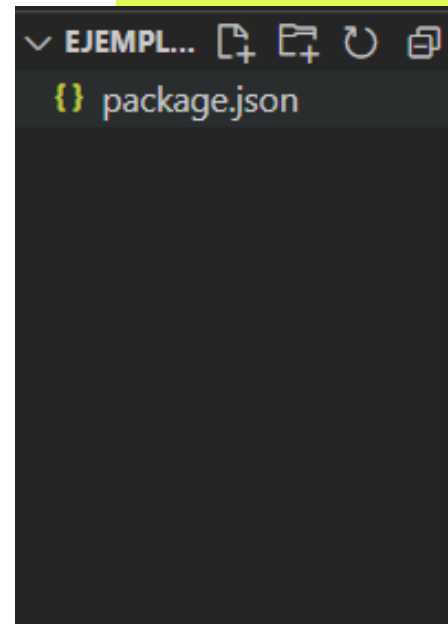
# Preparación de un proyecto de Express js



# Paso 1: npm init -y

Express no es nativo de nodejs, por lo tanto, necesitaremos primero contar con un **package.json** para gestionar las dependencias a instalar.

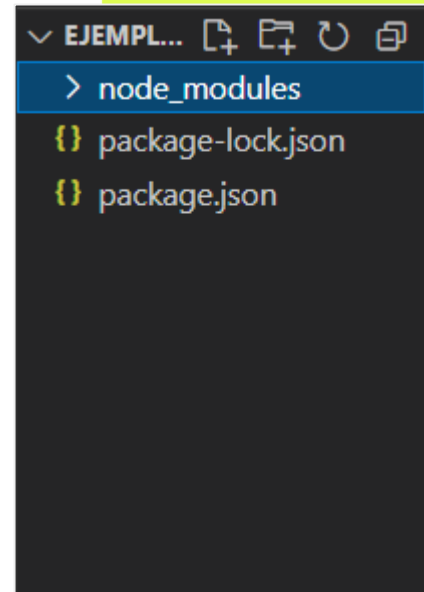
Una vez que tenemos package.json en nuestra carpeta, podemos continuar instalando dependencias.



# Paso 2: npm install express

Procedemos a instalar de manera local express js. Al ejecutar este comando, notaremos cómo se genera una carpeta `node_modules`, que es donde se encuentra almacenado express

A partir de este punto, ya contamos con la estructura elemental instalada. El resto es más “flexible”.



# Paso 3: Estructurar el proyecto

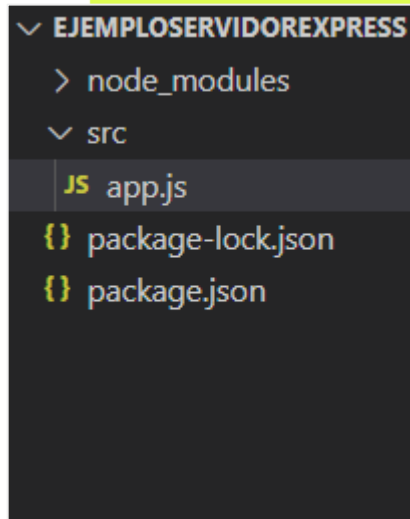
Se recomienda tener una carpeta src, donde vivirá todo nuestro código, dentro de la cual crearemos un archivo con el nombre "app.js"

Finalmente, el archivo app.js ya puede importar la dependencia instalada de **express js**, ya sea por commonjs:

```
const express = require('express');
```

o bien por module (recordar colocar el type:"module" en package.json):

```
import express from 'express';
```





# Break

¡Unos minutos y volvemos!



Universidad  
Provincial del Sudoeste  
*Promoviendo el Desarrollo Armónico de la Región*

# Conceptos básicos de PHP

# ¿Qué es PHP?

PHP es un lenguaje de programación interpretado del lado del servidor, y de uso general, que se adapta especialmente al desarrollo web. Fue creado a mediados de la década del 90.

PHP se ejecuta en el servidor y genera una respuesta que se envía al cliente a través de la red.

PHP es compatible con una amplia variedad de bases de datos y se integra fácilmente con otros lenguajes de programación y tecnologías web.

Tiene una gran comunidad de desarrollo, de la que se puede obtener soporte fácilmente.

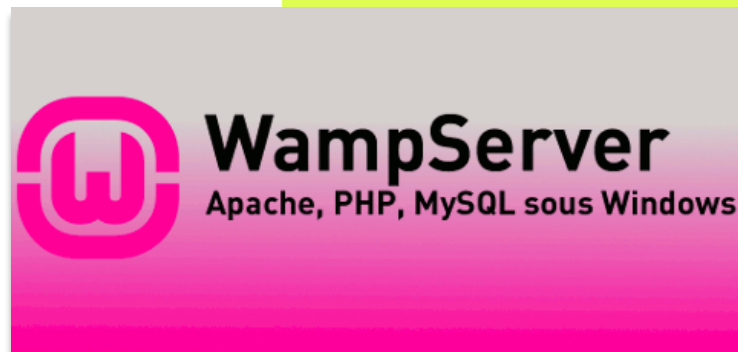


# ¿Qué necesito para correr PHP?

Al ser PHP un lenguaje interpretado, es necesario contar con un entorno que permita la ejecución de los scripts que escribamos en este lenguaje. Esto es, un servidor que ejecute PHP y en el cual vivirán nuestros aplicativos, para su poder ejecutar.

Nosotros utilizaremos [WampServer](#), instalado de manera local.

En caso de que tengas Linux o Mac, puedes optar por [XAMPP](#). En Linux existe la alternativa de Lamp, que puede instalarse con el comando “sudo apt install lamp-server^”



Universidad  
Provincial del Sudoeste  
*Promoviendo el Desarrollo Armónico de la Región*



# Bases de PHP



Universidad  
Provincial del Sudoeste  
*Promoviendo el Desarrollo Armónico de la Región*

# Generalidades PHP

- ✓ Etiquetas: `<?php ... ?>`
- ✓ Impresión en pantalla: `echo`
- ✓ Existe un atajo para `echo`: `<?= "texto..." ?>`
- ✓ Concatenación de strings: se realiza con `.` (con punto)
- ✓ Las instrucciones deben cerrarse con `;`
- ✓ Los comentarios en el código se realizan con `//` (para comentar una línea) o `/* ... */` para comentarios multilinea



# Tipos de datos y Variables

- ✓ En PHP las variables se definen con un nombre, al que le antepone el signo \$. Por ej: \$nombre= "Diego";
- ✓ Para conocer el tipo de datos de una variable, podemos utilizar la instrucción `gettype($variable)`;
- ✓ Existen: integer, double, string, boolean, null (primitivos), y por otro lado arrays, objetos (variables compuestas)
- ✓ Para debug se utiliza `var_dump($variable)`; Funciona en cierta forma, como un `console.log` de JavaScript.
- ✓ Una constante se define con la instrucción `define('nombre', 'valor')`;
- ✓ Existen constantes predefinidas. Su nombre comienza con `PHP_`



# Variables Superglobales

- ✓ `$_SERVER['SERVER_ADDR']`; Sirve para visualizar características o configuraciones del servidor
- ✓ `$_GET["nombre"]`; Sirve para recibir parámetros enviados desde el cliente, vía método get (query params)
- ✓ `$_POST["email"]`; Sirve para recibir parámetros enviados desde el cliente (con un formulario, por ejemplo), vía método post



# Estructuras de control

- ✓ PHP utiliza una sintaxis similar a la de JavaScript para sus estructuras de control: if, while, switch, for, etc.

```
ej02 > php index.php > ...
```

```

1 nombre='Diego';
2
3 if(nombre=='Diego'){
4     echo "<h4>Hola, $nombre...!!! Bienvenido</h4>";
5 }
6
7
8 $numero=0;
9
10 while($numero<=10){
11     echo "<p>$numero</p>";
12     $numero++;
13 }
14
15
16 echo "<p>";
17 for($numero=1; $numero<=10; $numero++){
18     echo "$numero, ";
19 }
20 echo "</p>";

```



# Funciones en PHP

- ✓ Presentan, también, una sintaxis similar a la de JavaScript:

```
ej02 >  index.php > ...  
24  
25 <?php  
26  
27 $nombre='Mariela';  
28 ✓ function saludar($persona){  
29     echo "Hola, $persona. ¿Cómo estás?";  
30 }  
31  
32 saludar($nombre);  
33  
34 ?>
```



# Funciones predefinidas

- ✓ `var_dump($variable);` Sirve para el contenido de una variable, o dato
- ✓ `date('d-m-y');` Muestra la fecha, en el formato indicado como argumento. Hay muchas funciones relacionadas con fechas. Con solo escribir `date`, y si tenemos la ayuda del editor activa, podremos confirmarlo
- ✓ `time();` Permite sacar la fecha en formato timestamp
- ✓ `rand($min, $max);` Sirve para generar números aleatorios
- ✓ `sqrt($variable);` Calcula la raíz cuadrada de `$variable`
- ✓ `pi();` Muestra el valor de PI (3,1416)
- ✓ `round($numero, $decimales);` Redondea `$numero`, según los decimales indicados



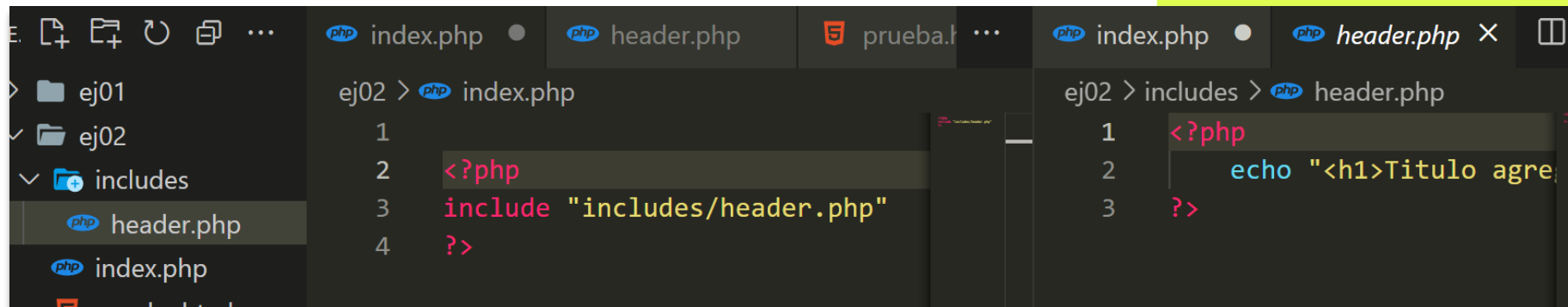
# Funciones predefinidas

- ✓ `get_type($variable);` muestra el tipo de datos de la variable indicada como argumento
- ✓ Funciones “`is_`”: permiten comprobar si una variable es de algún tipo, o tiene alguna característica. Hay muchas funciones `is_`
- ✓ `isset($variable);` nos permite comprobar la existencia de `$variable` en nuestro script
- ✓ `unset($variable);` elimina la variable que pasamos como argumento.
- ✓ `empty($variable);` muestra si una variable tiene contenido. Devuelve true o false
- ✓ `strlen($texto);` devuelve la cantidad de letras de la variable `$texto`
- ✓ Funciones para cadenas de caracteres: comienzan con `str`. Por ejemplo: `strpos($cadena, “algo”)`. O la antes citada `strlen`. Además, `strtolower`, `strtoupper`, `str_replace`, etc.





# Modularizando: include



The screenshot shows a code editor with two tabs: 'index.php' and 'header.php'. The 'index.php' tab is active, showing the following code:

```
1  
2 <?php  
3 include "includes/header.php"  
4 ?>
```

The 'header.php' tab is also visible, showing the following code:

```
1 <?php  
2 echo "<h1>Titulo agre  
3 ?>
```


The file explorer on the left shows the directory structure: 'ej01', 'ej02', and 'includes' (containing 'header.php' and 'index.php').



# Arreglos con PHP



# Arrays en PHP

ej02 >  index.php > ...

```
118
119 <?php
120     $heroes=array('Batman', 'Superman', 'Hulk', 'Black-Widow');
121     var_dump($heroes);
122     array_push($heroes, 'Ironman', 'Thor', 'the Wasp', 'Gamora', 'Wonder Woman');
123     var_dump($heroes);
124
125
126     $villanos=['Magneto', 'Thanos', 'Loki', 'Joker'];
127     var_dump($villanos);
128     ?>
```

C:\wamp64\www\UPSO\clase03\ej02\index.php:121:

**array** (size=4)

```
0 => string 'Batman' (length=6)
1 => string 'Superman' (length=8)
2 => string 'Hulk' (length=4)
3 => string 'Black-Widow' (length=11)
```

C:\wamp64\www\UPSO\clase03\ej02\index.php:127:

**array** (size=4)

```
0 => string 'Magneto' (length=7)
1 => string 'Thanos' (length=6)
2 => string 'Loki' (length=4)
3 => string 'Joker' (length=5)
```



Universidad  
Provincial del Sudoeste  
*Promoviendo el Desarrollo Armónico de la Región*

# Arrays en PHP

```
echo "<ul>";
for($i=1; $i<count($heroes); $i++){
    echo "<li>$heroes[$i]</li>";
}
echo "</ul>";

echo "<hr>";
foreach ($heroes as $indice => $heroe) {
    $orden=$indice+1;
    echo "$orden) $heroe<br>";
}
```

- Superman
- Hulk
- Black-Widow
- Ironman
- Thor
- the Wasp
- Gamora
- Wonder Woman

- 1) Batman
- 2) Superman
- 3) Hulk
- 4) Black-Widow
- 5) Ironman
- 6) Thor
- 7) the Wasp
- 8) Gamora
- 9) Wonder Woman



# Arrays asociativos

```
148  <?php
149
150  $persona=[
151      "nombre"=>"Diego",
152      "apellido"=>"Peretti",
153      "edad"=>61,
154      "nacionalidad"=>"argentino"
155  ];
156
157  var_dump($persona);
158
159  ?>
160
161  <?= $persona["apellido"] ?>
```

C:\wamp64\www\UPSO\clase03\ej02\index.php:157:

```
array (size=4)
  'nombre' => string 'Diego' (length=5)
  'apellido' => string 'Peretti' (length=7)
  'edad' => int 61
  'nacionalidad' => string 'argentino' (length=9)
```

Peretti



Universidad  
Provincial del Sudoeste  
*Promoviendo el Desarrollo Armónico de la Región*

# ¿Preguntas?



Universidad  
Provincial del Sudoeste  
*Promoviendo el Desarrollo Armónico de la Región*

# Muchas gracias.



Universidad  
Provincial del Sudoeste  
*Promoviendo el Desarrollo Armónico de la Región*