

DESARROLLO WEB

PROGRAMACION FRONT-END

“Todo lo que el usuario puede ver en una pagina web y su interacción con ella”



ANGULAR es un framework front-end para aplicaciones web desarrollado en TypeScript, de código abierto, mantenido por Google, que se utiliza para crear y mantener aplicaciones web de una sola página (SPA). Su objetivo es aumentar las aplicaciones basadas en navegador con capacidad de Modelo Vista Controlador (MVC), en un esfuerzo para hacer que el desarrollo y las pruebas sean más fáciles.

Una **Single-Page Application (SPA)** es un tipo de aplicación web que ejecuta todo su contenido en una sola página.

Funciona cargando el contenido **HTML**, **CSS** y **JavaScript** por completo al abrir la web. Al ir pasando de una sección a otra, solo necesita cargar el contenido nuevo de forma dinámica si este lo requiere, pero no hace falta cargar la página por completo. Esto mejora los tiempos de respuesta y agiliza mucho la navegación, favoreciendo así a la **experiencia de usuario**.

Para entender como trabaja una **SPA** (Single Page Application) **una analogía** puede ayudar, pero haciendo **una comparación** con otro estilo de hacer desarrollo web llamado MPA (multiple page application)

🏠 **MPA (Multi Page Application) = Ir al cine**

- Cada vez que quieres ver una película diferente, tienes que:
 - Salir de la sala
 - Volver a comprar boleto
 - Hacer fila
 - Entrar a otra sala
 - Esperar que inicie la función



Cada acción implica **una nueva carga completa**, como cuando el navegador recarga toda la página.

📺 SPA (Single Page Application) = Ver películas en Netflix

- Estás en tu sillón.
- Quieres cambiar de película o serie → haces clic y listo.
- El video nuevo aparece sin moverte ni recargar nada.
- Solo cambia el contenido de la pantalla, pero **sigues en la misma app**.

Solo se actualiza lo necesario (como los datos en formato JSON), y el **resto permanece igual**, ofreciendo una experiencia más rápida y fluida.



Un proyecto de Angular se programa utilizando:



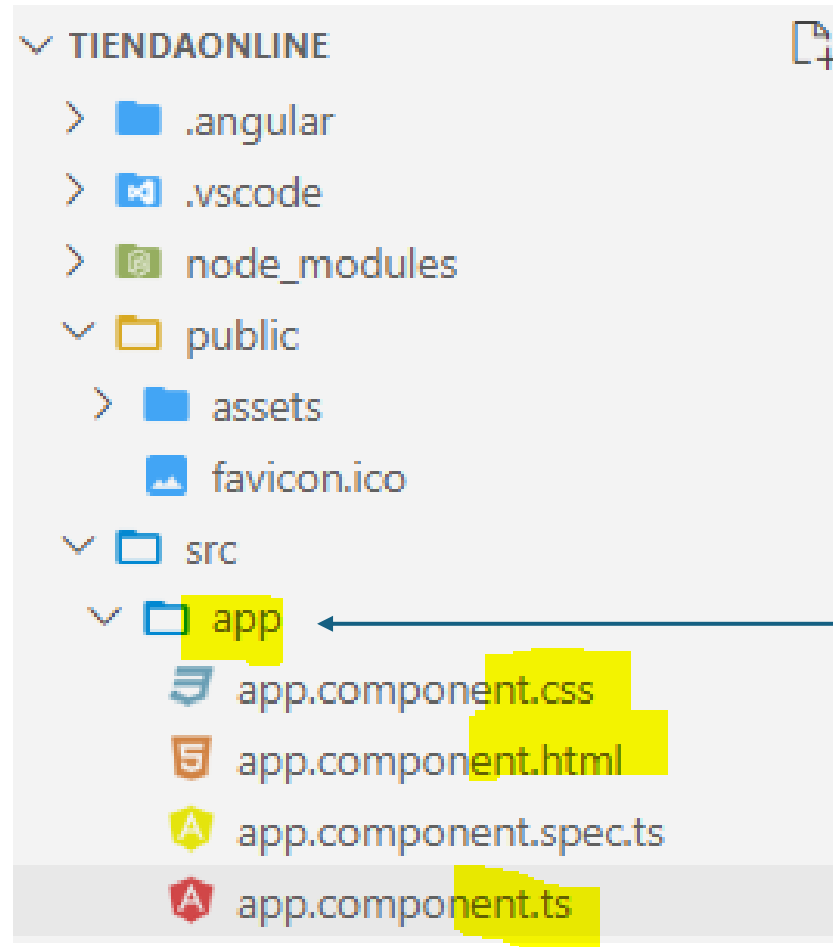
¿Qué necesito para programar en Angular?

- 1** Instalar NodeJS, para tener acceso a npm (gestor de paquetes de javascript), por ejemplo ejecutar un npm install, npm install Bootstrap, etc.
- 2** Instalar Angular CLI (Command Line Interface) para tener acceso a comandos para crear el proyecto, componentes, levantar el servidor, etc
- 3** Instalar un editor web (visual estudio code)

Un **componente** en Angular es un bloque de código **re-utilizable**, que consta básicamente de 3 archivos: un **CSS**, un **HTML** (también conocido como plantilla o en inglés, template) y un **TypeScript** (en adelante, TS).



La carpeta **app** con la que viene Angular **por defecto es un componente**
incluye los 3 archivos.



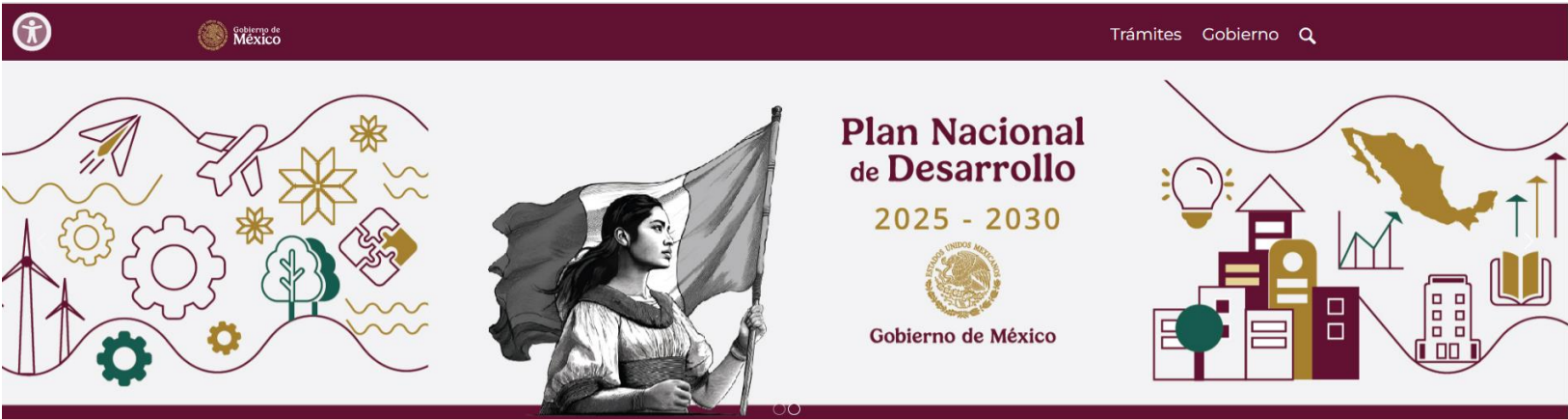
app.component.ts

src > app > app.component.ts > ...

```
1  import { Component } from '@angular/core';
2  import { RouterOutlet } from '@angular/router';
3  import { ListarEmpleadosComponent } from '../componentes/lista';
4
5  @Component({
6    selector: 'app-root',
7    imports: [RouterOutlet, ListarEmpleadosComponent],
8    templateUrl: './app.component.html',
9    styleUrls: ['./app.component.css']
10 })
11 export class AppComponent {
12   title = 'repasoAngular';
13 }
14
```

Un componente es una clase de Typescript con el decorador @Component.

Un **componente** controla un espacio de la pantalla, que se denomina vista.



The screenshot shows the top section of the Mexican Government website. It features a dark blue header with the logo of the Government of Mexico and navigation links for 'Trámites' and 'Gobierno'. Below the header is a large banner for the 'Plan Nacional de Desarrollo 2025 - 2030' featuring a woman holding the Mexican flag and various icons representing development sectors like energy, education, and infrastructure.

← componente

← componente

Lo más buscado:

CURP

Acta de nacimiento

Recibo de luz


Pasaporte

Cédula profesional

Precio de gasolina

← componente


Trámites por categoría



IDENTIDAD, PASAPORTE Y MIGRACIÓN

Acta de nacimiento
CURP
Cartilla Militar
Pasaporte


Ver todos >



EDUCACIÓN

Cédula Profesional
Duplicado de Cédula
Registro UNADM
Inscripciones AEFCM


Ver todos >



ENERGÍA

Solicitud de servicio
Recibo
Fallas en el servicio
Precio de gasolina

Ver todos >



TRABAJO

Busca empleo
Consulta de NSS
Saldo de préstamo ISSSTE
Semanas cotizadas IMSS

Ver todos >

→ componente

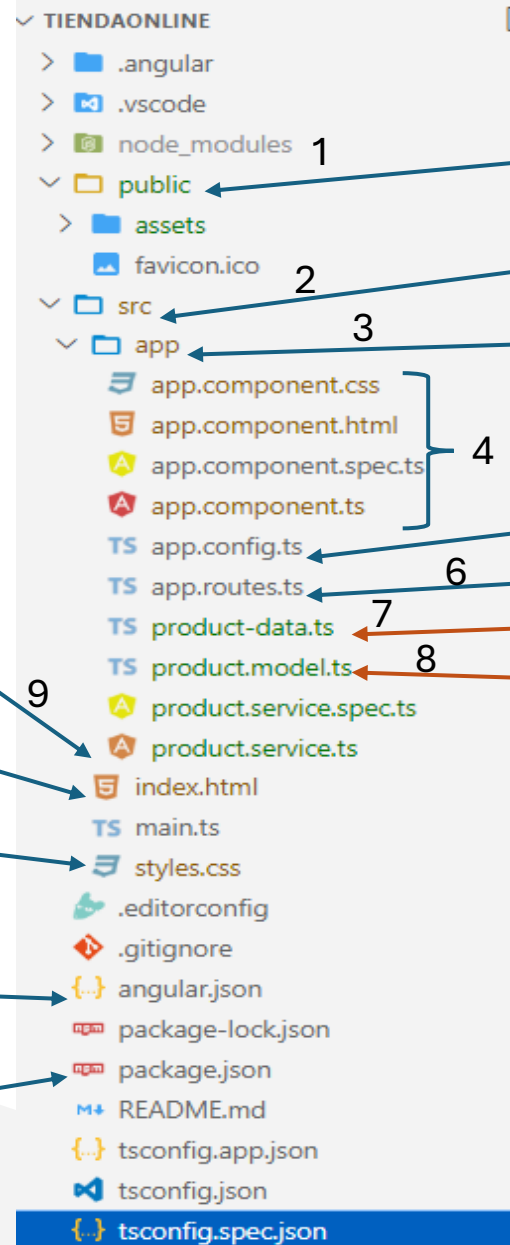
Por lo tanto ...

LOS COMPONENTES

PIEZAS DE UI

PUEDEN REUTILIZARSE Y
COMBINARSE

Estructura de un proyecto en Angular



¿Qué es una interface en TypeScript (y Angular)?

Una **interface** es como un **contrato** que define la forma que debe tener un objeto. Le dice a Angular (y al programador) qué **propiedades** debe tener ese objeto y de qué **tipo** deben ser.

¿Para qué sirve la interface en Angular?

- Ayuda a **organizar mejor los datos**.
- Hace que el código sea más **seguro y predecible**.
- Mejora el **autocompletado y la detección de errores** en tu editor (como VS Code).
- Se usa mucho en **servicios** cuando recibes o mandas datos a una API, o en formularios, componentes, etc.

TS personal.ts M X

src > app > models > TS personal.ts > ...

```
1  export interface Personal {  
2  
3      nombre:string;  
4      edad:number;  
5      sueldo:number;  
6      antiguedad:number;  
7      foto:string;  
8      id:number;  
9  }
```

Servicios en Angular

Un servicio en Angular provee datos al componente que los solicite.

Estos datos pueden venir de un API, de localStorage e incluso de un almacenamiento de datos estáticos como un array de objetos previamente definido.

Proporciona métodos para que los componentes tengan acceso a dichos datos.

```
empleados.service.ts M X
src > app > empleados.service.ts > ...
1  import { Injectable } from '@angular/core';
2  import plantilla from '../data/plantilla.json';
3  import { Personal } from '../models/personal';
4
5  @Injectable({
6    providedIn: 'root'
7  })
8  export class EmpleadosService {
9
10     empleados:Personal[]=plantilla;
11
12     constructor() { }
13
14     ngOnInit(): void { }
15
16     getAll():Personal[]{
17       return this.empleados;
18     }
19
20     getById(){ }
21     create(){ }
22     update(){ }
23     delete(){ }
24
25   } //fin de la clase
```

¿Qué es la inyección de dependencia en un servicio?

La inyección de dependencia es un patrón de diseño que utiliza angular para crear una instancia de la clase del servicio y ponerla a disposición de los componentes que la incluyan en su constructor.

```
listar-empleados.component.ts X
src > app > componentes > listar-empleados > listar-empleados.component.ts > Li
1  import { Component } from '@angular/core';
2  import { EmpleadosService } from '../empleados.service';
3  import { Personal } from '../models/personal';
4  import { CommonModule } from '@angular/common';
5
6  @Component({
7    selector: 'app-listar-empleados',
8    imports: [CommonModule],
9    templateUrl: './listar-empleados.component.html',
10   styleUrls: ['./listar-empleados.component.css']
11 })
12 export class ListarEmpleadosComponent {
13
14   misEmpleados!: Personal[];
15
16   constructor(private empleadosService: EmpleadosService){
17
18   }
19
20   ngOnInit(): void {
21     this.misEmpleados=this.empleadosService.getAll();
22     console.log(this.empleadosService);
23   }
24 }
25
26
```

Esto es inyectar una dependencia

Practica Empleados

https://drive.google.com/file/d/1nNFvPC_tBfTZ15BWMEZPBYvbUlkyquLL/view?usp=sharing

- **Tarea personal:**

- Tema del proyecto de tarea: Catálogo de Doctores.
- En el siguiente link puedes tomar un array de doctores.
- <https://drive.google.com/file/d/1eTsP8Tqt2rNwnhVS8tlUIEau7Ft6XnDs/view>
- Genera el modelo(interface), el servicio, componente listar-doctores para mostrar datos en consola
- Crear repositorio local y subir a GitHub