

Aprendiendo Angular

From Scratch

Sandra Ivette

July 2024

Introducción

- ▶ ¿Qué es Angular?
- ▶ ¿Por qué aprender Angular?
- ▶ Historia y evolución de Angular.

¿Qué es Angular?

- ▶ Es un Framework Frontend de JavaScript, Ayuda a construir interfaces web interactivas y modernas.
- ▶ Es una colección de herramientas y funciones que rodean al Framework, tal como una CLI, gestor de proyectos, Herramientas para Debugging, Plugins para IDE, entre otras herramientas.

¿Por qué aprender Angular?

- ▶ Simplifican el proceso de creación de interfaces conforme un proyecto vaya creciendo.
- ▶ Escribes Código Declarativo
- ▶ Separación de conceptos vía Componentes
- ▶ principios y conceptos OOP
- ▶ Usas TypeScript de forma nativa

Historia y evolución de Angular.

- ▶ Es un framework que sigue evolucionando e innovando, aunque lo hace de una manera muy estable y compatible con versiones anteriores.
- ▶ El equipo de Angular tienen una política de actualización de versiones cada 6 meses.
- ▶ Las actualizaciones son incrementales, esto significa que Angular 2 es la base "Vanilla" y sus posteriores versiones hacen mejoras "opcionales" a la versión base.

Instalación de Angular

- ▶ Requisitos previos: Node.js y npm.
- ▶ Instalación de Angular CLI.
- ▶ Creación de un nuevo proyecto.
- ▶ Estructura del proyecto Angular.

Requisitos previos: Node.js y npm.

- ▶ Node.js es un entorno de ejecución para JavaScript.
- ▶ Permite ejecutar código JavaScript en el servidor.
- ▶ En esta presentación, aprenderemos a instalar Node.js en Windows.

Paso 1: Descargar Node.js

- ▶ Visita el sitio web oficial de Node.js: <https://nodejs.org/>
- ▶ Haz clic en el botón de descarga para Windows.



Run JavaScript Everywhere

Node.js® is a free, open-source, cross-platform JavaScript runtime environment that lets developers create servers, web apps, command line tools and scripts.

[Download Node.js \(LTS\)](#)

Downloads Node.js v20.15.0¹ with long-term support. Node.js can also be installed via [package managers](#).

Want new features sooner? Get **Node.js v22.4.0¹** instead.

```
1 // server.mjs
2 import { createServer } from 'node:http';
3
4 const server = createServer((req, res) => {
5   res.writeHead(200, { 'Content-Type': 'text/plain'
6   res.end('Hello World!\n');
7 });
8
9 // starts a simple http server locally on port 3000
10 server.listen(3000, '127.0.0.1', () => {
11   console.log('Listening on 127.0.0.1:3000');
12 });
13
14 // run with 'node server.mjs'
```

JavaScript [Copy to clipboard](#)

Learn more what Node.js is able to offer with our [Learning materials](#).

Trademark Policy Privacy Policy Code of Conduct Security Policy

© OpenJS Foundation

Paso 2: Ejecutar el Instalador

- ▶ Una vez descargado el archivo, haz doble clic para ejecutar el instalador.
- ▶ Se abrirá el asistente de instalación de Node.js.
- ▶ Lee el acuerdo de licencia.
- ▶ Haz clic en "Next".
- ▶ Elige la carpeta donde se instalará Node.js.
- ▶ Haz clic en "Next".
- ▶ Elige los componentes a instalar.
- ▶ Deja las opciones predeterminadas y haz clic en "Next".
- ▶ Revisa la configuración y haz clic en "Install".
- ▶ Una vez finalizada la instalación, haz clic en "Finish".
- ▶ Node.js ya está instalado en tu sistema.

Verificar Instalación

- ▶ Abre una terminal (CMD o PowerShell).
- ▶ Escribe 'node -v' y 'npm -v' para verificar la instalación.
- ▶ Deberías ver las versiones de Node.js y npm instaladas.

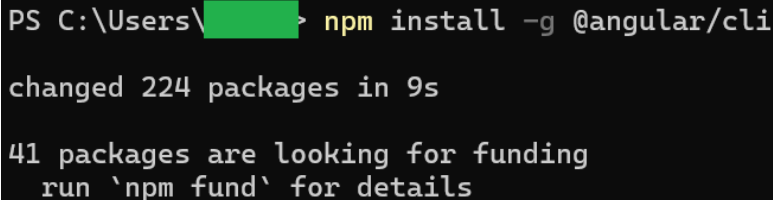
```
PS C:\Users\[redacted]> node -v
v20.15.0
PS C:\Users\[redacted]> npm -v
10.7.0
```

Paso 3: Instalar Angular CLI

- ▶ Angular CLI es una herramienta de línea de comandos para inicializar, desarrollar, y mantener aplicaciones Angular.
- ▶ En la terminal, ejecuta el siguiente comando:

Comando

```
npm install -g @angular/cli
```

A screenshot of a Windows command prompt terminal. The prompt shows the user is in the directory 'C:\Users\[redacted]'. The command 'npm install -g @angular/cli' has been entered and executed. The output shows that 224 packages were changed in 9 seconds, and 41 packages are looking for funding, with a suggestion to run 'npm fund' for details.

```
PS C:\Users\[redacted] > npm install -g @angular/cli
changed 224 packages in 9s
41 packages are looking for funding
  run `npm fund` for details
```

La bandera "-g" indica que se está instalando este paquete a nivel Global.

Paso 4: Crear un Nuevo Proyecto Angular

- ▶ Navega a la carpeta donde deseas crear el proyecto.
- ▶ Ejecuta el comando:

Comando

`ng new my-app`

- ▶ La convención indica que se escriba el nombre del proyecto en minúsculas y separado con guiones (Kebab case)
- ▶ Sigue las instrucciones en pantalla para configurar el nuevo proyecto.

```
PS C:\Users\... \Escritorio> ng new my-app

Would you like to share pseudonymous usage data about this project with the Angular Team
at Google under Google's Privacy Policy at https://policies.google.com/privacy. For more
details and how to change this setting, see https://angular.dev/cli/analytics.

Yes

Thank you for sharing pseudonymous usage data. Should you change your mind, the following
command will disable this feature entirely:

  ng analytics disable --global

Global setting: enabled
Local setting: No local workspace configuration file.
Effective status: enabled
? Which stylesheet format would you like to use? CSS [ https://developer.mozilla.org/docs/Web/CSS
]
? Do you want to enable Server-Side Rendering (SSR) and Static Site Generation (SSG/Prerendering)? Yes
```

Paso 4: Crear un Nuevo Proyecto Angular

- ▶ En caso de que tenga un error por falta de permisos, ejecuta el comando:

Comando

```
Set-ExecutionPolicy -Scope CurrentUser  
-ExecutionPolicy Unrestricted
```

```
PS C:\Users\██████████\Escritorio> ng new my-app  
ng : File C:\Users\██████████\AppData\Roaming\npm\ng.ps1 cannot be  
loaded because running scripts is disabled on this system. For  
more information, see about_Execution_Policies at  
https://go.microsoft.com/fwlink/?LinkID=135170.  
At line:1 char:1  
+ ng new my-app  
+ ~~~  
+ CategoryInfo          : SecurityError: (:) [], PSSecurityException  
+ FullyQualifiedErrorId : UnauthorizedAccess  
PS C:\Users\██████████\Escritorio> Set-ExecutionPolicy -Scope CurrentUser -ExecutionPolicy Unrestricted
```

Paso 5: Ejecutar la Aplicación Angular

- ▶ Navega a la carpeta del proyecto:
- ▶ Ejecuta el siguiente comando para iniciar el servidor de desarrollo:

Comando

`ng serve`

- ▶ Abre tu navegador y ve a `http://localhost:4200/` para ver tu aplicación.

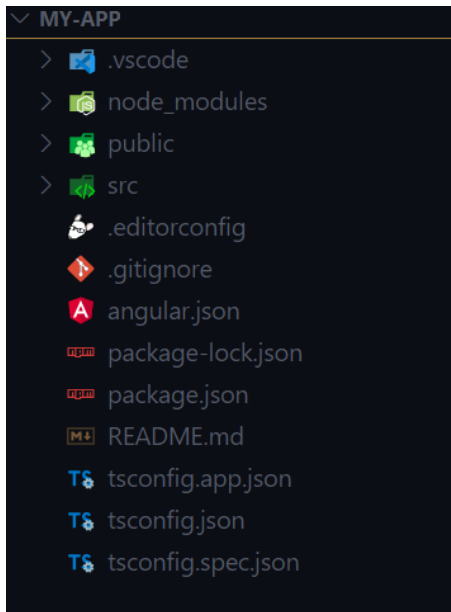


Hello, my-app

Congratulations! Your app is running. 🎉

[Explore the Docs](#)[Learn with Tutorials](#)[CLI Docs](#)[Angular Language Service](#)[Angular DevTools](#)

Estructura de un Proyecto en Angular



Estructura de un Proyecto en Angular

- ▶ README.md - Descripción del proyecto y primera vista dentro de un repositorio, además de detalles necesarios para levantar los servicios.
- ▶ tsconfig.* - toda la configuración de TypeScript se hace mediante estos ficheros.
- ▶ package.json - indica todas las dependencias de Angular que se tienen en este proyecto.
- ▶ package-lock.json - fichero que se actualiza automáticamente con las versiones utilizadas en el proyecto.
- ▶ angular.json - describe toda la arquitectura del Proyecto, así como la configuración de este mismo.
- ▶ .gitignore - indica directorios y ficheros a ignorar en caso de ser guardado en un gestor de versiones como git.

Estructura de los directorios de la aplicacion

- ▶ /src - Carpeta Principal donde se encuentran los archivos de código del proyecto.
- ▶ /src/app - Carpeta donde se encuentran los componentes Principales de la aplicación.
- ▶ /public - Todo fichero que se encuentre en esta carpeta será de fácil acceso mediante una URL relativa.
- ▶ /node_modules - Todas las dependencias de Angular se guardan en este fichero.
- ▶ /.vscode - configuración del editor de texto a usar (Visual Studio Code en este ejemplo).

Estructura de la carpeta /src

- ▶ /src/style.css - Estilos globales de la aplicación.
- ▶ /src/main.ts - Es el punto de entrada de la aplicación.
- ▶ /src/index.html - Es el fichero que inicializa la aplicación.
- ▶ /src/app - Carpeta donde se encuentran los componentes Principales de la aplicación.

Estructura de la carpeta /src/app

```
src/  
  app/  
    core/  
    shared/  
    components/  
    services/  
    models/  
    app.module.ts  
    app.component.ts  
    app.component.html  
    app.component.css  
    app.component.spec.ts
```

Estructura de la carpeta /src/app

- ▶ `app.module.ts` - Este archivo es el módulo principal de la aplicación. Define los componentes, directivas, pipes y servicios que pertenecen a este módulo y los que deben ser importados para que la aplicación funcione.
- ▶ `app.component.ts` - Define el componente principal de la aplicación. Incluye la lógica de negocio y las propiedades necesarias para la interfaz de usuario.
- ▶ `app.component.html` - La plantilla HTML del componente principal. Define la estructura visual de la interfaz de usuario.
- ▶ `app.component.css` - Archivo de estilos CSS del componente principal. Define la apariencia visual del componente.
- ▶ `app.component.spec.ts` - Archivo de pruebas unitarias para el componente principal.

Estructura de la carpeta /src/app

- ▶ /core - Contiene servicios y otras piezas reutilizables que solo se deben cargar una vez en toda la aplicación, como guardias, interceptores y servicios globales.
- ▶ /shared - Contiene componentes, directivas y pipes reutilizables en toda la aplicación. También puede incluir módulos compartidos.
- ▶ /components - Agrupa los componentes específicos de la aplicación. Cada componente suele tener su propia carpeta con sus archivos ".ts", ".html", ".css" y ".spec.ts".
- ▶ /services - Contiene los servicios que manejan la lógica de negocio y la comunicación con APIs externas.
- ▶ /models - Define las interfaces y clases que representan datos utilizados en la aplicación.

Componentes en Angular

- ▶ ¿Qué es un componente?
- ▶ Creación de componentes.
- ▶ Estructura de un componente.
- ▶ Comunicación entre componentes.

¿Qué es un componente?

Un componente representa una parte de la interfaz de usuario de la aplicación. Cada componente en Angular está compuesto por cuatro elementos principales:

- ▶ Clase (TypeScript): Define la lógica y el comportamiento del componente. Se declara como una clase TypeScript y contiene propiedades y métodos que manejan los datos y las interacciones del usuario.
- ▶ Decorador (@Component): Especifica metadatos sobre el componente, como su selector, plantilla y estilos.
- ▶ Plantilla (HTML): Define la estructura de la interfaz de usuario del componente. La plantilla puede incluir enlaces de datos, directivas y otros componentes.
- ▶ Estilos (CSS/SCSS): Define la apariencia visual del componente.

Estructura de un componente

- ▶ Clase TypeScript (ExampleComponent): Define una propiedad "title" y un método "onClick" que actualiza el valor de "title".
- ▶ Decorador @Component:
 - ▶ selector: Define el nombre del elemento HTML que representa este componente (<app-example>).
 - ▶ templateUrl: Apunta al archivo HTML que contiene la plantilla de este componente.
 - ▶ styleUrls: Apunta al archivo CSS que contiene los estilos de este componente.
- ▶ Plantilla (HTML):
 - ▶ Usa la interpolación ({{title}}) para mostrar el valor de la propiedad title.

Estructura de un componente

```
@Component({ // Decorador @Component
  selector: 'app-root', //Nombre del Componente para llamarlo en html
  standalone: true, //Indicador Standalone
  imports: [RouterOutlet, ExampleComponent], // Imports de Componentes Anidados
  templateUrl: './app.component.html', //Vista HTML
  styleUrls: ['./app.component.css'] // Estilos CSS
})

//Declaracion de la Clase, se incluye el comportamiento del Componente
export class AppComponent {
  title = 'my-app';
}
```

Creacion de un Componente

- Puedes crear un componente en Angular utilizando el CLI de Angular. Por ejemplo, para crear un componente llamado "example", ejecutarías:

Comando

```
ng generate component example
```

- Esto generará los archivos necesarios para el componente (example.component.ts, example.component.html, example.component.css, y example.component.spec.ts) y actualizará el módulo Angular (app.module.ts) para incluir el nuevo componente.

Uso de un Componente

- ▶ Para usar un componente en otro componente o en una plantilla, simplemente incluyes su selector en el HTML. Por ejemplo, para usar el componente `ExampleComponent` en el componente raíz (`app.component.html`), agregarías:

Comando

```
<app-example></app-example>
```

- ▶ Los componentes en Angular son bloques de construcción esenciales para desarrollar aplicaciones web modulares y mantenibles. Cada componente encapsula una parte de la interfaz de usuario y su lógica asociada, lo que facilita la reutilización y el mantenimiento del código.

Directivas y Pipes

- ▶ ¿Qué son las directivas?
- ▶ Tipos de directivas.
- ▶ Creación y uso de directivas personalizadas.
- ▶ ¿Qué son los pipes?
- ▶ Uso de pipes incorporados y personalizados.

Directivas y Pipes

- ▶ Las plantillas de Angular son dinámicas, cuando Angular las renderiza, transforma el DOM de acuerdo a las instrucciones dadas por las directivas.
- ▶ Cada Directiva que usamos tiene un nombre, y determina donde puede ser usada, sea en un elemento, atributo, componente o clase.
- ▶ Las directivas añaden comportamiento a los elementos de una aplicación Angular. (Extienden la funcionalidad del HTML)
- ▶ Existen 3 tipos:
 - ▶ Directivas de atributo
 - ▶ Directivas estructurales
 - ▶ Componentes

Directivas de Atributos

- ▶ Las directivas de atributos alteran la apariencia o el comportamiento de un elemento del DOM y son usados como atributos de los elementos.
- ▶ Ejemplo de directivas de atributo:
 - ▶ `ngModel`: Implementa binding.
 - ▶ `ngClass`: Permite añadir o eliminar varias clases.
 - ▶ `ngStyle`: Permite asignar estilos inline.

Directivas de Atributos

- ▶ Para crear una directiva de atributo, desde Angular CLI se utiliza el comando:

Comando

```
ng generate directive <nombre>
```

```
ng g d <nombre>
```

- ▶ Esto crea un archivo `src/app/<nombre>.directive.ts` y su correspondiente archivo de test

Directivas Estructurales

- ▶ Alteran la estructura del DOM, añadiendo, eliminando y manipulando los elementos a los que están unidos.
- ▶ Ejemplo de directivas estructurales:
 - ▶ `*ngIf`: Nos permite incluir condicionales de lógica.
 - ▶ Evaluar sentencias.
 - ▶ Hacer comparaciones.
 - ▶ Mostrar/ocultar secciones de código.
 - ▶ `*ngFor`: Directiva para iterar, permite ejecutar bucles. Evaluar de acuerdo a una condición n veces.

Directivas Estructurales

- ▶ Para generar una directiva estructural se utiliza el comando:

Comando

`ng generate directive <nombre>`

- ▶ Importar los elementos necesarios y agregar la lógica de acuerdo al tipo de directiva que se desee crear.

Directivas y Pipes

- ▶ Los pipes sirven para realizar transformaciones de los datos, a la hora de mostrarlos en los templates de los componentes.
- ▶ Utilizar pipes para transformar cadenas, valores de monedas, fechas, y otros tipos de datos.
- ▶ Los pipes son simples funciones para usar en los templates que aceptan una entrada y regresan un valor transformado.
- ▶ Son muy útiles, ya que se pueden utilizar en toda una aplicación después de declararla una única vez.

Directivas y Pipes

Angular tiene incorporados algunos pipes de uso habitual.

- ▶ DatePipe: Formatea una fecha de acuerdo a la regla local.
- ▶ UpperCasePipe: Transforma texto a todo Mayúsculas.
- ▶ LowerCasePipe: Transforma texto a todo Minusculas.
- ▶ CurrencyPipe: Convierte un número a una moneda.
- ▶ DecimalPipe: Convierte un número a una cadena decimal.
- ▶ PercentPipe: Convierte un número a una cadena de porcentaje.

Es posible encadenar pipes para que la salida de una se convierta en la entrada de otra.

Servicios y Dependencias

- ▶ ¿Qué son los servicios?
- ▶ Inyección de dependencias.
- ▶ Creación y uso de servicios.

Ruteo en Angular

- ▶ Configuración de rutas.
- ▶ Navegación entre vistas.
- ▶ Rutas anidadas y parámetros de ruta.

Trabajo con Formularios

- ▶ Formularios reactivos y por template.
- ▶ Validación de formularios.
- ▶ Manejo de datos de formularios.

Consumo de APIs

- ▶ Uso de HttpClient.
- ▶ Realización de peticiones HTTP.
- ▶ Manejo de respuestas y errores.