













Angular 17

Contenido del curso

-  Tema 1. Primera aplicación
-  Tema 2. Introducción a TypeScript
-  Tema 3. Arquitectura de Angular
-  Tema 4. Plantillas y data binding
-  Tema 5. Componentes

-  Tema 6. Rutas
-  Tema 7. Formularios
-  Tema 8. Servicios
-  Tema 9. Aplicaciones tradicionales
-  Tema 10. Directivas y pipes

Requisitos y contenido general

Requisitos

Java, C# o conocimientos de POO

Javascript

HTML /CSS

Cliente / servidor

Periodo de cambio



standalone



signals



@plantillas



@NgModule, *ngIf, *ngFor

Sólo HTML Y CSS



Sin Bootstrap



Sin Material






Sin SCSS / SASS



Angular 17

Tema 1. Primera Aplicación

Objetivos

-  Aprender los conceptos básicos de Angular y su arquitectura
-  Familiarizase con el entorno de desarrollo y ejecución
-  Entender la estructura de un proyecto

Contenidos

- Qué es Angular
- Herramientas de desarrollo y ejecución
 - Node.js / npm
 - Angular CLI (ng)
 - Visual Studio Code
- Creación de un proyecto
- Estructura de un proyecto
- Componentes, plantillas y data binding (introducción)

Qué es Angular



Angular es un framework de Javascript para la creación de Aplicaciones de Página Única (SPA).

También es la plataforma de desarrollo que usaremos para aplicar el framework



Node.js y npm. Entorno de ejecución de JavaScript y Gestor de paquetes



TypeScript. Superconjunto de JavaScript



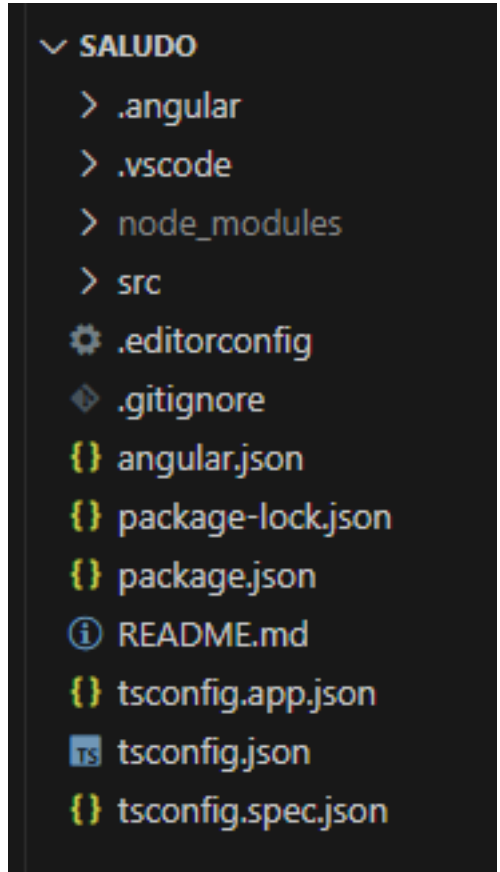
Angular CLI (ng)



Visual Studio Code. Editor de código fuente

Estructura de un proyecto (1)

.angular	Ficheros temporales para acelerar la ejecución
.vscode	Configuración de Visual Studio Code
node_modules	Carpeta de descarga de npm (todas las bibliotecas)
src	Todo el código fuente. La carpeta de trabajo
.editorconfig	Configuración del editor
.gitignore	Configuración de GIT
angular.json	Configuración de Angular: Página inicial, CSS por defecto, fichero main, etc.
package.json	Fichero de configuración de npm
package-lock.json	Fichero de configuración auxiliar de npm (uso interno)
tsconfig.json	Configuración del compilador de TypeScript
tsconfig.app.json	Configuración del compilador de TypeScript para subaplicaciones (poco usado)
tsconfig.spec.json	Configuración de las pruebas unitarias



Estructura de un proyecto (2)

index.html	La página HTML de nuestra aplicación SAP
style.css	La hoja de estilos por defecto de la aplicación
assets	Carpeta para recursos estáticos (imágenes, archivos PDF, etc.)
favicon.ico	Icono por defecto de la aplicación (pestañas del navegador)
main.ts	La función inicial de la aplicación
app.config.ts	Fichero auxiliar de main para configuración (generalmente servicios)
app	La carpeta donde escribiremos todo el código fuente
app.component.xxx	Componente inicial creado por el asistente

```
▼ src
  ▼ app
    # app.component.css
    <> app.component.html
    TS app.component.ts
    TS app.config.ts
  > assets
  ★ favicon.ico
  <> index.html
  TS main.ts
  # styles.css
```

¿Qué hemos aprendido?



- Qué es Angular
- Cómo crear un proyecto y su estructura general
- Manejo de Angular CLI (ng) y Visual Studio Code
- Introducción a los componentes

Resumen de comandos

npm

npm install [-g] [paquete]

npm uninstall paquete

npm -version

ng

ng version

ng help

ng new nombre_proyecto

 --routing false

 --skip-tests

ng generate component nombre_componente / ng g c

ng serve [-o]

Ejercicio 01

Crea el proyecto «ejercicio01» de la misma forma que hemos creado el proyecto de ejemplo, sin rutas ni clases para pruebas unitarias.

Añade dos componentes que se dibujen en el componente principal, de izquierda a derecha. Usa tablas, float, flex... lo que te resulte más cómodo.

En el primer componente define un texto y un botón, que al pulsarlo lo pase a mayúsculas o minúsculas alternativamente. Los métodos de la clase «string» son los de siempre: «toUpperCase()» y «toLowerCase()».

El segundo componente tendrá un contador similar al que hemos visto en clase, pero con tres botones: incrementar, decrementar y poner a cero. Quiero que lo añadas **dos veces** al componente principal, para que compruebes que cada componente añadido es un objeto independiente.

Aplica estilos para que tenga un aspecto aceptable. Si creas estilos comunes a todos los componentes (tipo de fuente, tamaño) defínelos en «style.css»

Ten cuidado con las tildes y las eñes en los identificadores. Aunque Angular trabaja en unicode, a TypeScript le sientan fatal los caracteres extraños.