



Desarrollo y Programación de
Aplicaciones Avanzadas con Angular

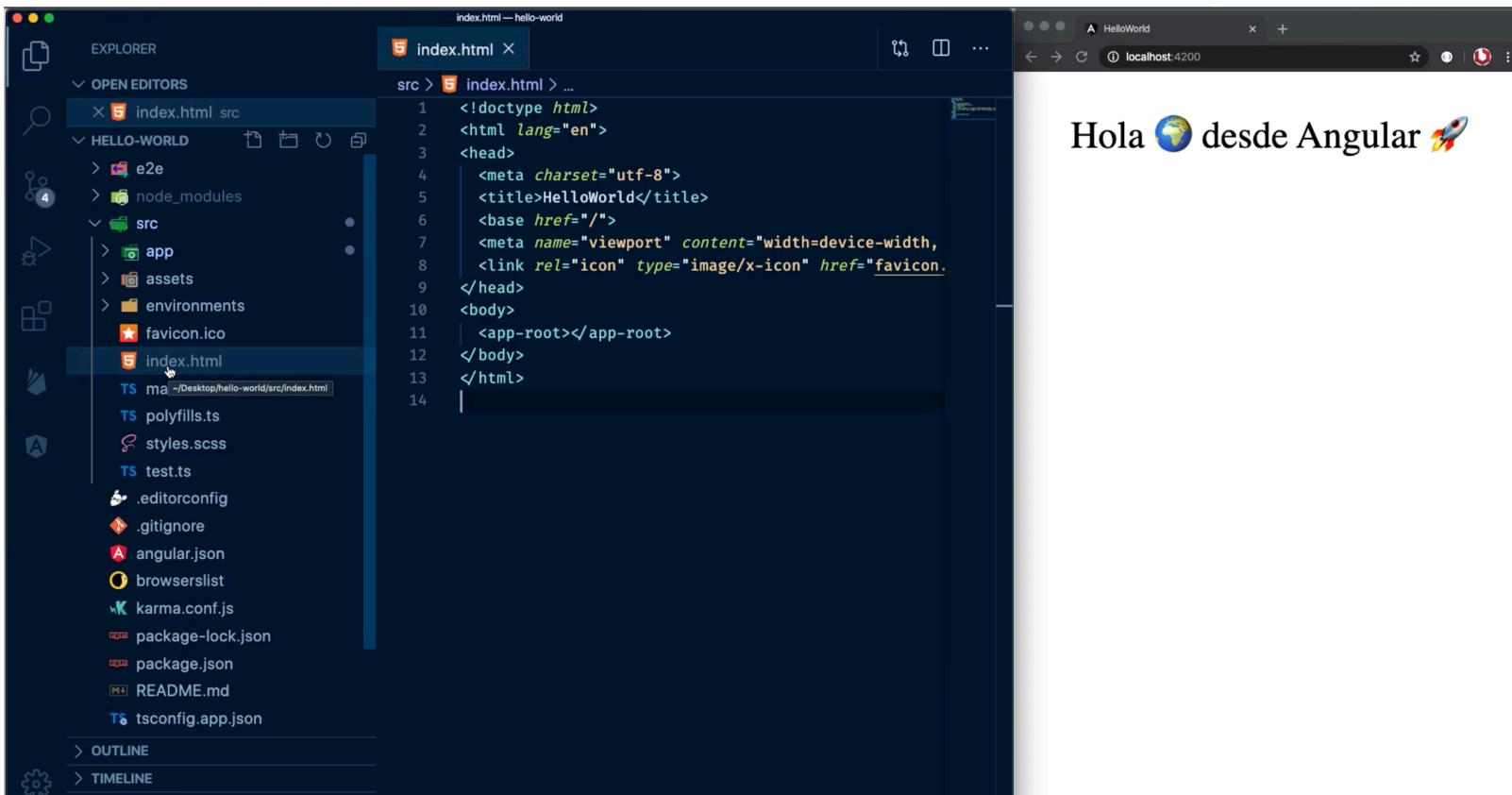
Formación online



Módulo 3. AppModule & AppComponent

Módulo 3. AppModule & AppComponent

index.html



The image shows a code editor interface with two main panes. The left pane is the 'EXPLORER' view, displaying the project structure of a 'HELLO-WORLD' application. It includes files like 'e2e', 'node_modules', 'src' (containing 'app', 'assets', 'environments', 'favicon.ico', and 'index.html'), and various configuration files ('polyfills.ts', 'styles.scss', 'test.ts', '.editorconfig', '.gitignore', 'angular.json', 'browserslist', 'karma.conf.js', 'package-lock.json', 'package.json', 'README.md', 'tsconfig.app.json'). The right pane is the 'index.html — hello-world' editor, showing the HTML code for the application's entry point. The browser preview window on the right shows the rendered output: 'Hola  desde Angular .

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>HelloWorld</title>
  <base href="/">
  <meta name="viewport" content="width=device-width,>
  <link rel="icon" type="image/x-icon" href="favicon.>
</head>
<body>
  <app-root></app-root>
</body>
</html>
```

Módulo 3. AppModule & AppComponent

¿Qué es un módulo?

Un NgModule (módulo) es una clase Typescript marcada con el decorador @NgModule

Las propias librerías de Angular son módulos, como FormsModule, HttpClientModule o RouterModule. Muchas librerías externas también son hechas y compatibles con NgModules, como Angular Material, Ionic o AngularFire. Angular sigue una arquitectura modular y los módulos lo hacen excesivamente fácil

Toda aplicación Angular tiene al menos 1 módulo, el módulo root. Se empaqueta (bootstrap) este módulo para poder arrancar la app.

Módulo 3. AppModule & AppComponent

AppModule

Módulo 3. AppModule & AppComponent

¿Qué es un módulo?

Array con todos los componentes declarados en el módulo

Array con todos los módulos necesarios para ejecutar sus componentes

```
src > app > A app.module.ts > ...
1 import { BrowserModule } from '@angular/platform-browser';
2 import { NgModule } from '@angular/core';
3
4 import { AppComponent } from './app.component';
5
6 @NgModule({
7   declarations: [
8     AppComponent
9   ],
10  imports: [
11    BrowserModule
12  ],
13  providers: [],
14  bootstrap: [AppComponent]
15 })
16 export class AppModule { }
```

Nombre de la clase que representa el módulo

Módulo 3. AppModule & AppComponent

¿Qué puede hacer un módulo?

Declarar que componentes, directivas, pipes y servicios... pertenecen a un módulo

Hacer públicas algunas o todas sus declaraciones de componentes, directivas, pipes...
de tal forma que pueda ser recusado por otros módulos

Importar otros módulos propios de nuestra aplicación

Módulo 3. AppModule & AppComponent

¿Qué es un componente?

Un componente es la pieza o bloque de código más básico de una aplicación Angular, y todo entorna en base a componentes. Es una clase Typescript con el decorador @Component

Un componente debe pertenecer a un NgModule (módulo) y debe estar declarado en sólo y exclusivamente uno

Un componente se basa en la parte lógica (.ts) la vista (.html) y los estilos (.css o .scss)
aunque el único realmente imprescindible es el .ts

Módulo 3. AppModule & AppComponent

AppComponent

Módulo 3. AppModule & AppComponent

¿Qué es un componente?

Metadatos -> @Component decorator
(describen aspectos del componente)

Contexto del componente

```
1 import { Component } from '@angular/core';
2
3 {
4   @Component({
5     selector: 'app-root',
6     templateUrl: './app.component.html',
7     styleUrls: ['./app.component.scss']
8   })
9   export class AppComponent {
10     // Aquí se escribe el código referente al componente
11 }
```

Módulo 3. AppModule & AppComponent

Lifecycle hooks

Todos los componentes en Angular tienen ciclos de vida (lifecycle hook) los cuales podemos sobre-escribir para darles un comportamiento extra. Es recomendable implementar la interfaz que representa el lifecycle hook junto con la implementación del método.

Los 3 mas importantes/usados son **OnInit**, **OnDestroy**, **OnChanges** (aunque existen algunos mas)

OnInit: Se ejecuta 1 sola vez cuando el componente es creado y toda su información/código html ha sido cargado. Se usa para tareas al iniciar el componente.

OnDestroy: Se ejecuta 1 sola vez cuando el componente es destruido. Se usa para liberar memoria, eventos...

OnChanges: Se ejecuta múltiples veces, cada vez que alguna propiedad vinculada (data-bind) cambia (cuando un @Input recibe un valor seguidamente se ejecuta este hook). También se lanza antes que onInit a causa del Change-Detector. Recibe un parámetro que nos indica qué ha cambiado.

NOTA: Los componentes pueden tener un constructor, pero no es recomendado usar ninguna lógica ahí más que inyectar servicios. Para lógica, OnInit.

Módulo 3. AppModule & AppComponent

Lifecycle hooks

```
import { Component, OnInit, OnDestroy, OnChanges, SimpleChanges } from '@angular/core';

@Component({
  selector: 'app-prueba',
  templateUrl: './prueba.component.html',
  styleUrls: ['./prueba.component.scss']
})
export class PruebaComponent implements OnInit, OnDestroy, OnChanges {

  constructor() {
    console.log('DENTRO DE constructor'); NO ESTÁ LISTO EL COMPONENTE AÚN...
  }

  ngOnInit(): void {
    console.log('DENTRO DE ngOnInit'); ESTÁ LISTO EL COMPONENTE!
  }

  ngOnDestroy(): void {
    console.log('DENTRO DE ngOnDestroy'); HACER LIMPIEZA ANTES DE DESTRUIR EL COMPONENTE
  }

  ngOnChanges(changes: SimpleChanges): void {
    console.log('DENTRO DE ngOnChanges', changes); SE LANZA ANTES QUE ONINIT Y LUEGO, CADA
    VEZ QUE DATA-BIND CAMBIE (EJ: INPUTS)
  }
}
```

Módulo 3. AppModule & AppComponent

Creación de Módulos / Componentes

- Para crear un módulo:
 - **\$ ng m nombre-modulo**
- Para crear un componente utilizamos el comando:
 - **\$ ng g c nombre-componente**
 - La ventaja de utilizar el comando de creación de componentes es que **declara** automáticamente el componente dentro del módulo correspondiente.

<https://angular.io/guide/feature-modules>

Módulo 3. AppModule & AppComponent



Angular Material

<https://material.angular.io/>

LABORATORIO: Creación de componente

Haciendo uso de Angular Material (<https://material.angular.io/>):

Se pide crear un componente de 'toolbar' para mostrar una barra de navegación superior (app bar) en nuestra aplicación web. De momento el HTML del componente solo debe incluir un text "Toolbar". Debemos asegurarnos que sea incluido ese componente dentro del app-root component.