*Clase 02. COMPONENTES Y ELEMENTOS DE UN PROYECTO ANGULAR*

ANÁLISIS DE ESTRUCTURA

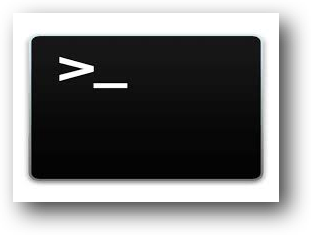
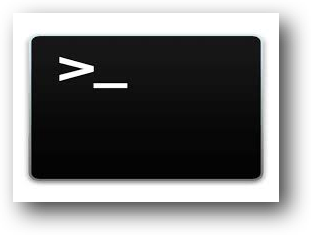
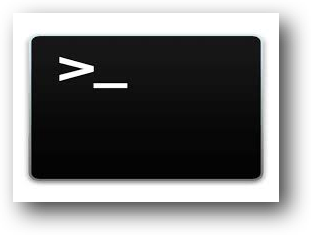
Si bien un proyecto Angular puede ser creado partiendo de un simple archivo index.html o una estructura básica de página web, hoy nadie empezaría un proyecto en Angular sin partir de una estructura de archivos, carpetas y herramientas configuradas y optimizadas para la tarea 📁.

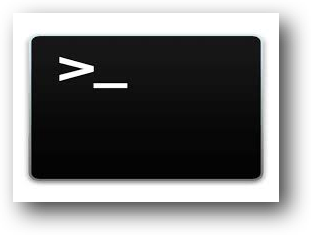
La tarea del armado de una estructura y configuración de herramientas se lleva a cabo mediante el uso de la herramienta Angular CLI propuesta por el mismo Framework 😉.

ANÁLISIS DE ESTRUCTURA

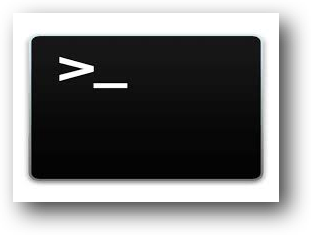
RECORDEMOS

¿CÓMO CREAR UN PROYECTO?

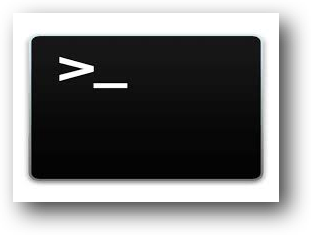
1. Verifica tu instalación  
    ng version
2. Crea tu primer proyecto  
    ng new my-project
3. Accede a la carpeta del proyecto  
   cd my-project
4. Ejecución del servidor

ng serve

Ejecuta nuestro entorno de desarrollo dejando habilitado el puerto donde corre nuestra aplicación por defecto.

ng serve -o

Hace lo mismo que el comando ng serve pero corre un navegador en el puerto por defecto - Recomendado🚀.

 ng serve -o --port=3500

Con este comando podemos elegir el puerto en el que se va a ejecutar nuestro proyecto.

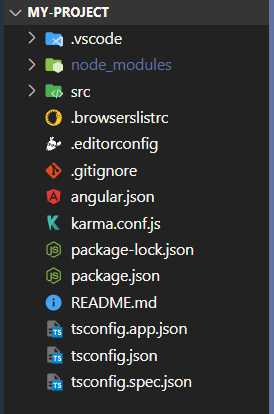
**ANGULAR BOILERPLATE**

Con la ejecución de creación de un proyecto Angular mediante el comando ng new de ANGULAR-CLI, se genera una estructura plantilla del tipo boilerplate. Este tipo de estructura no es más que un proyecto inicial con las configuraciones y herramientas necesarias de desarrollo 🙌.

Entre estas configuraciones se incluye una plantilla básica de archivos y carpetas que estaremos analizando a continuación.

**ESTRUCTURA**

**ARCHIVOS DE CONFIGURACIÓN**

gitignore

.browserslistrc

.editorconfig

angular.json

karma.config

package-lock.json

package.json

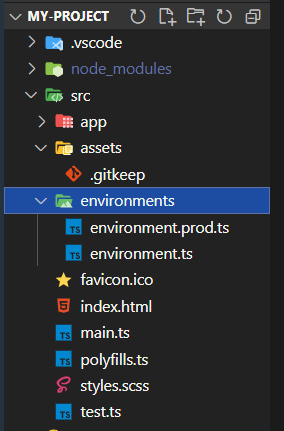
README.md

tsconfig.app.json

tsconfig.json

tsconfig.spec.json

**CARPETA SRC**

Carpeta assets

Carpeta environments

favicon.ico

index.html

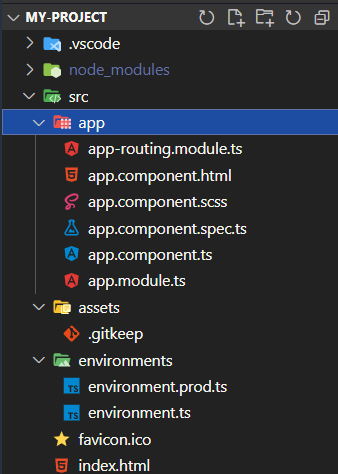
main.ts

polyfill.ts

styles.scss

test.ts

**CARPETA SRC/APP**

app-routing.module.ts

app.component.html

app.component.scss

app.component.spec.ts

app.component.ts

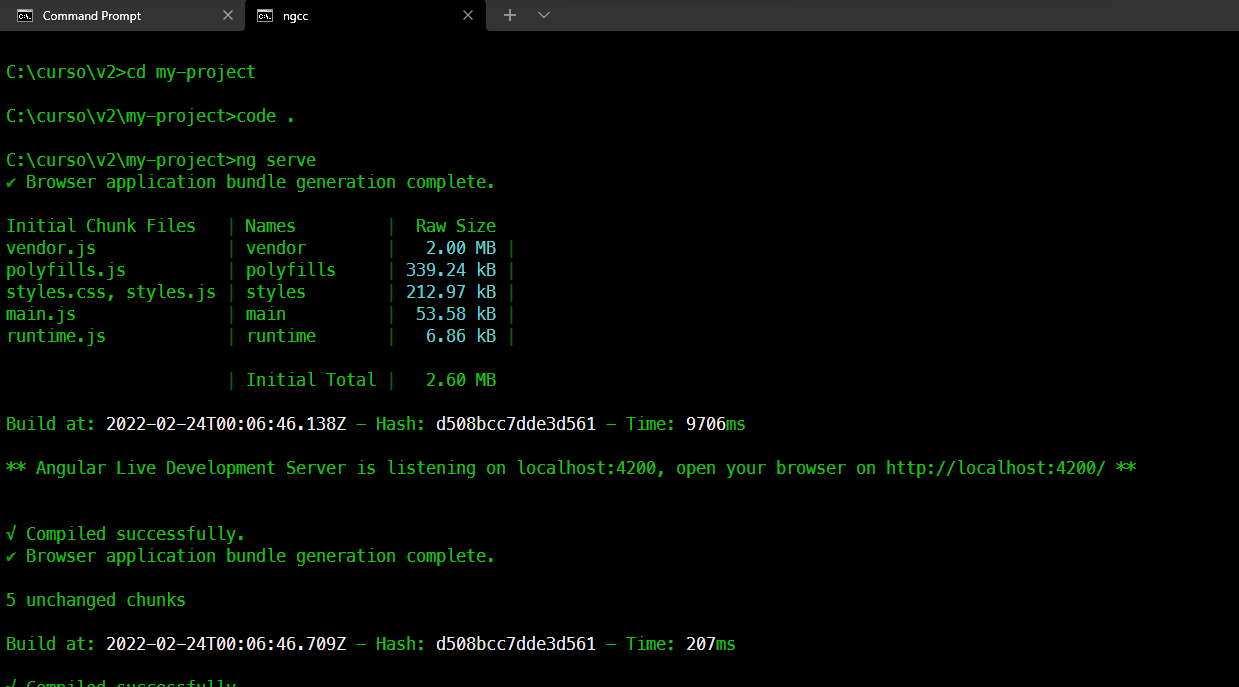
App.module.ts

Esta es la estructura inicial. En una app real encontraremos muchos más elementos.

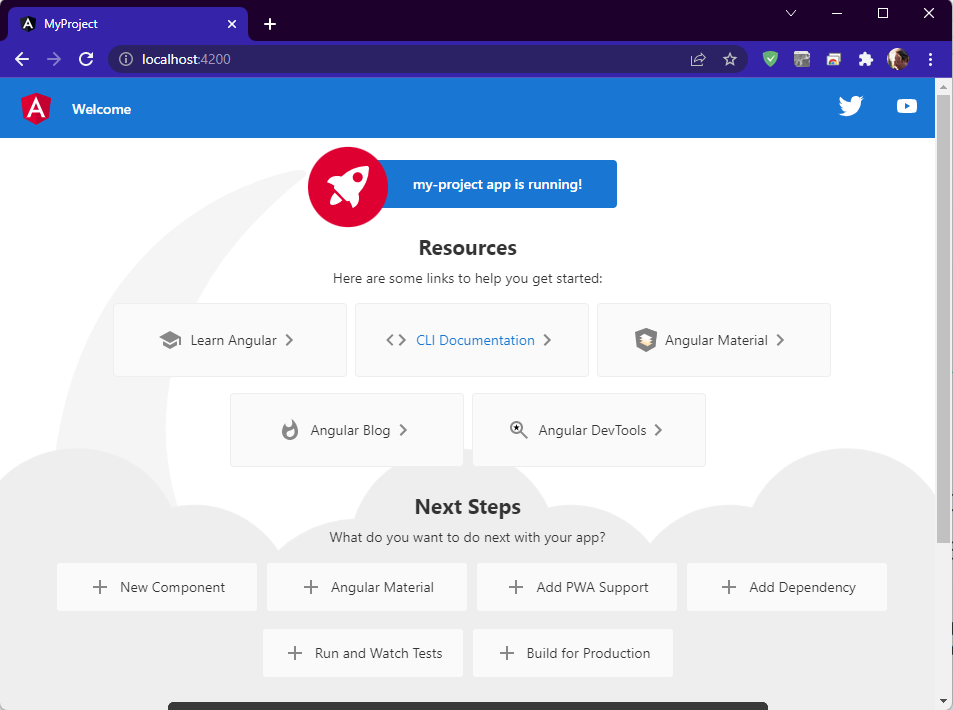
CICLO DE EDICIÓN, EJECUCIÓN Y DEPURACIÓN

CICLO

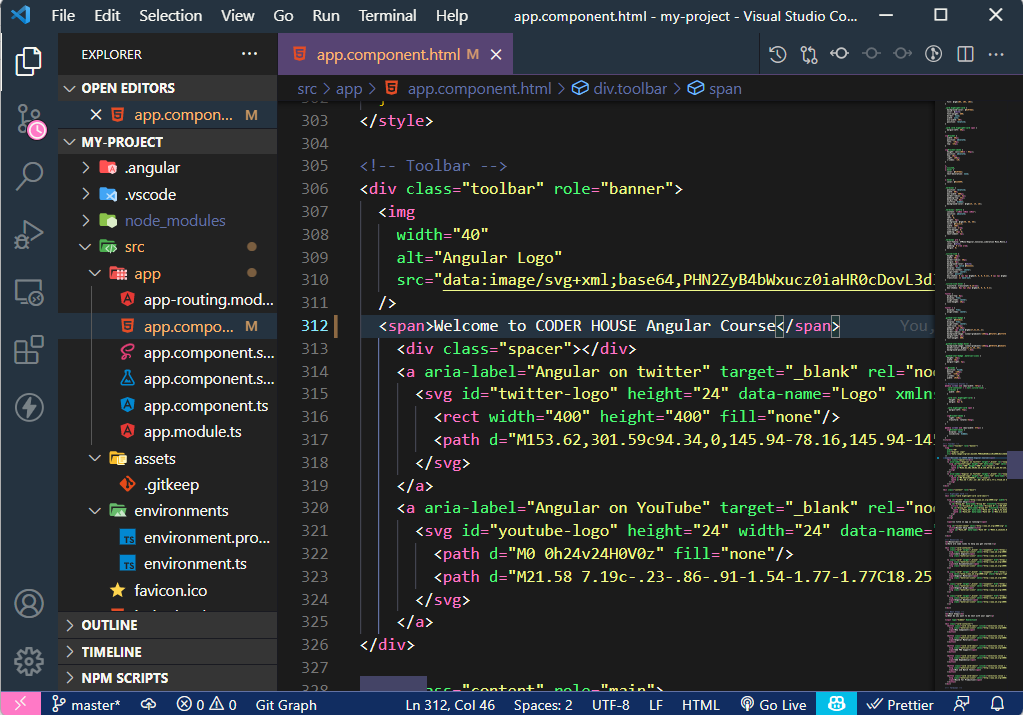
1. Ejecución del servidor “Angular”



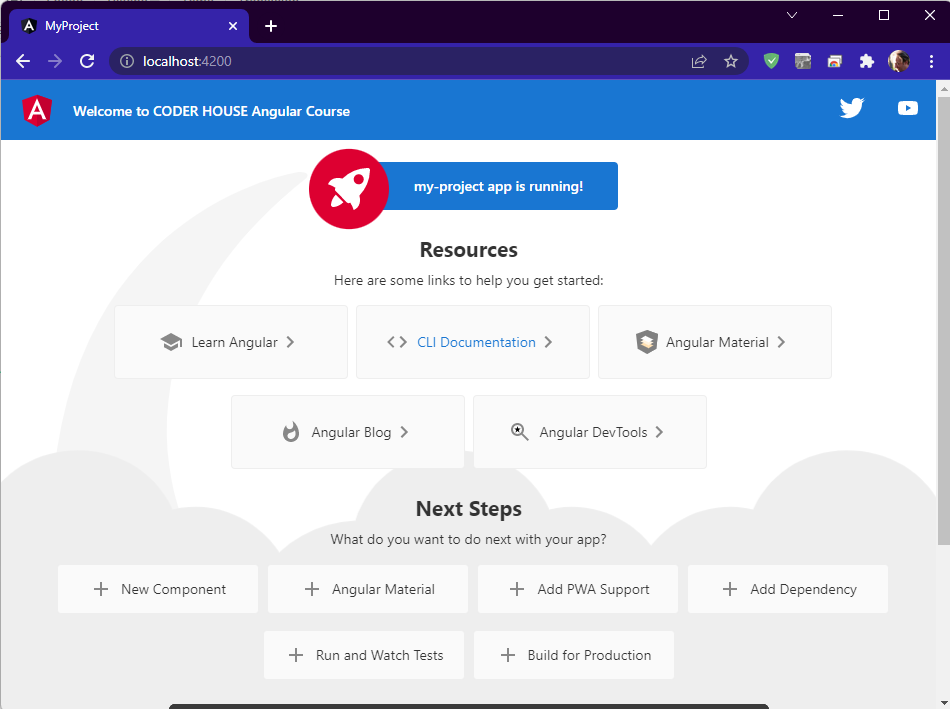
1. Navegar a la app: localhost: 4200



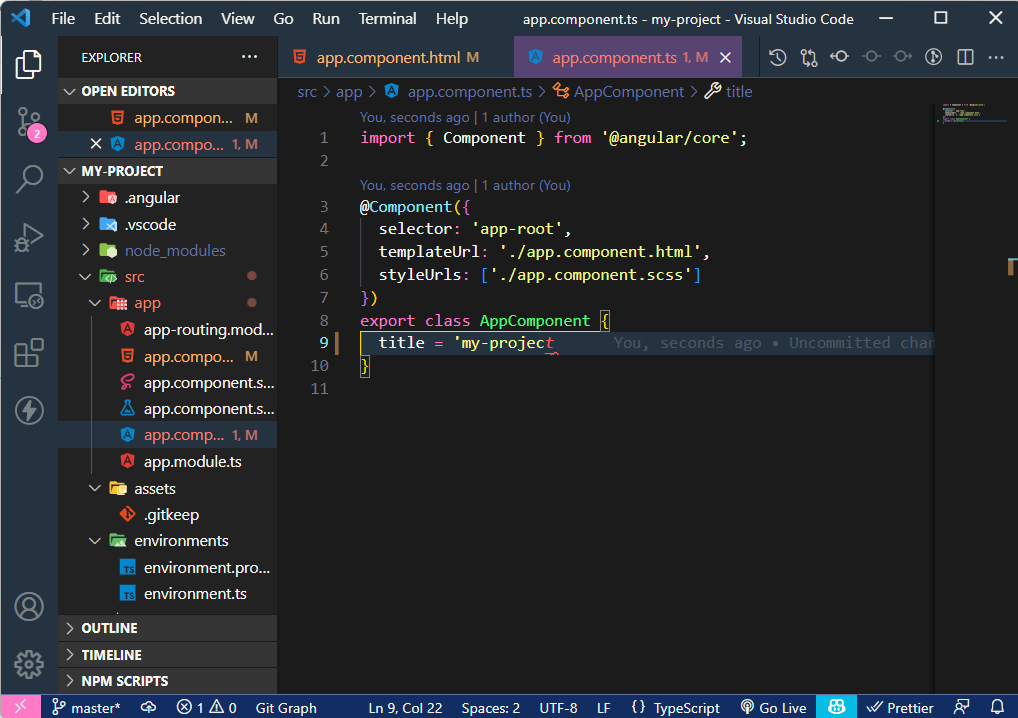
1. Editar y salvar el código



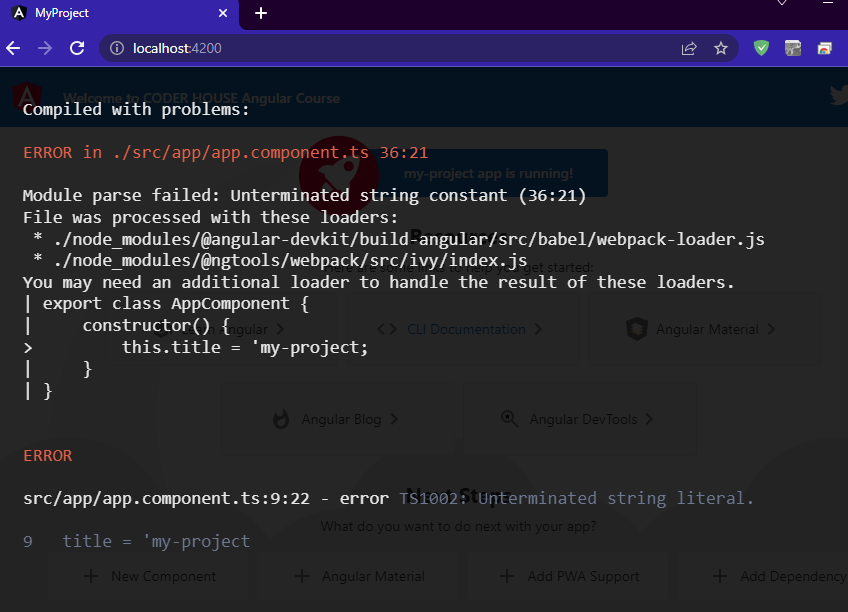
1. Refresh automático en el navegador



1. Generamos un error y salvamos



1. Error en el navegador



COMPONENTES

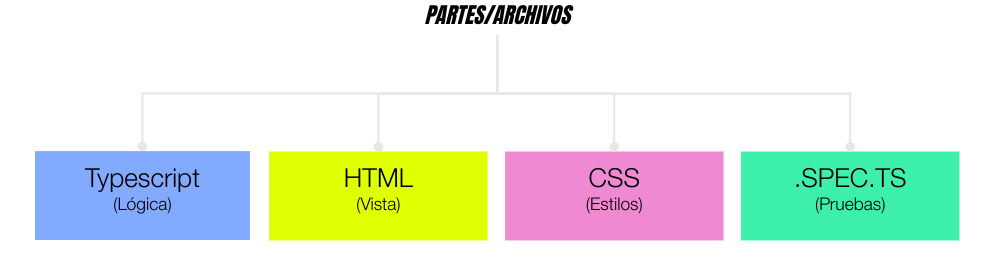
¿QUÉ ES UN COMPONENTE? Un componente representa una porción (o toda) de la aplicación y está contenido dentro de un módulo. Cada componente define una clase que contiene datos y lógica y está asociado con una plantilla HTML y CSS.

Se puede decir que un componente está orientado a la experiencia del usuario 💻.

Como podemos ver aquí, nuestra aplicación es un componente que vive en la única página HTML de nuestra aplicación 🤯:



**COMPONENTE BÁSICO**

  
⚠ Nota importante: Un componente puede tener uno o más archivos, este es un factor que dependerá de la necesidad de tu proyecto.

**Typescript (Lógica)  
APP COMPONENT TS**  
Esta parte del componente encapsula la lógica de la aplicación y referencia a los archivos de vista y estilo de este mismo 👉

*import* { Component } *from* '@angular/core';

@*Component*({

selector: 'app-root',

templateUrl: './app.component.html',

styleUrls: ['./app.component.css'],

})

*export* class AppComponent {

title = 'Mi tienda de Regalos';

}  
**HTML (Vista)  
APP COMPONENT HTML**

Esta parte del componente referencia estructura de la vista, básicamente cómo van a estar organizados los datos que queremos mostrar 💻. Se vincula a los estilos por las definiciones establecidas en el archivo de lógica:  
<h1 *class*="titulo">Bienvenidos</h1>

<h2 *class*="subtitulo">{{ title }}</h2>

**CSS (Estilos)  
APP COMPONENT SPEC.TS**

Este archivo se encarga de darle el formato visual al componente. Se encuentra diferenciado en el archivo de lógica 👇  
.*titulo* {

color: red;

}

.*subtitulo* {

color: yellow;

}

**.SPEC.TS (Pruebas)**

**APP COMPONENT SPEC.TS**  
Este archivo se encarga de las pruebas del componente 🙌. Es muy común a la hora de construir un componente. Sin embargo, aún no lo vamos a utilizar. Por ello, no lo crearemos en esta instancia.

import { TestBed } from '@angular/core/testing';

import { RouterTestingModule } from '@angular/router/testing';

import { AppComponent } from './app.component';

describe('AppComponent', () => {

beforeEach(async () => {

await TestBed.configureTestingModule({

imports: [ RouterTestingModule],

declarations: [AppComponent],

}).compileComponents();

});

it('should create the app', () => {

const fixture = TestBed.createComponent(AppComponent);

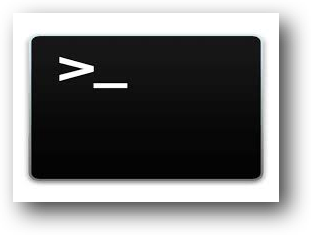
const app = fixture.componentInstance;

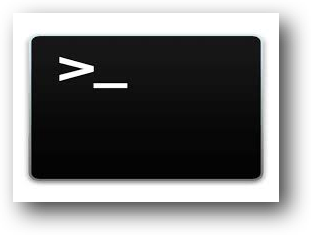
expect(app).toBeTruthy();

});

});

Componentes. Ahora bien, ¿cómo los creamos? 🤔

* Desde Angular.CLI  
  ng generate component <name> [options]

ng g c <name> [options]

* Generación Manual

Para la generación manual se crea una carpeta con los siguientes archivos:

component\_name

name.component.html

name.component.css

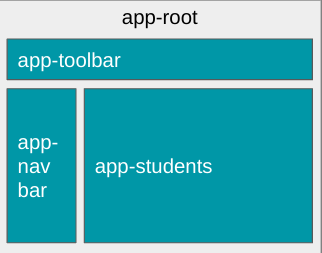
name.component.spec.ts

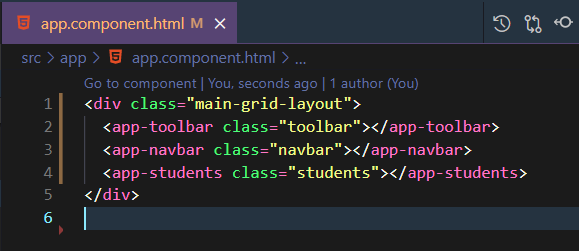
name.component.ts

COMPONENTIZACIÓN DE UNA APLICACIÓN

Como vimos, nuestra aplicación es un componente: app-root. También es conocido como componente principal o componente raíz 🌳. Dentro de él vamos a ir agregando los demás componentes.

De esta manera, se va generando una jerarquía de componentes padres-hijo dentro de nuestra aplicación.





Resumen de lo visto en clase hoy:

* ANGULAR CLI. Estructura y componentes.
* Ciclo de ejecución y edición.