DWC

(Desarrollo Web en entorno cliente)

Índice

1.- ¿Qué es Angular?

Angular es un framework JavaScript, construido sobre TypeScript de código abierto desarrollado por Google para crear aplicaciones web SPA (Single Page Application). Es una de las plataformas más populares para el desarrollo web frontend.

Angular ofrece una serie de ventajas, como:

- **Modularidad:** Angular se basa en una arquitectura modular que facilita la organización y el mantenimiento del código.
- **Enrutamiento:** Angular proporciona un sistema de enrutamiento robusto para navegar entre diferentes secciones de la aplicación.
- **Data binding:** Angular facilita la conexión entre la interfaz de usuario y los datos de la aplicación mediante el data binding.
- **Directivas:** Angular proporciona un conjunto de directivas que permiten modificar el comportamiento de los elementos HTML.
- **Inyección de dependencias:** Angular facilita la inyección de dependencias en los componentes y servicios.
- **Testing:** Angular proporciona herramientas para facilitar las pruebas de las aplicaciones.

Como plataforma, Angular incluye:

- Un marco basado en componentes para crear aplicaciones web escalables
- Una colección de bibliotecas bien integradas que cubren una amplia variedad de funciones, incluido el enrutamiento, la administración de formularios, la comunicación cliente-servidor y más.
- Un conjunto de herramientas de desarrollo para ayudarle a desarrollar, crear, probar y actualizar su código

Con Angular, estamos aprovechando una plataforma que puede escalar desde proyectos de un solo desarrollador hasta aplicaciones de nivel empresarial. Lo mejor de todo es que el ecosistema Angular consta de un grupo diverso de más de 1,7 millones de desarrolladores, autores de bibliotecas y creadores de contenido.

2.- Alternativas a Angular

Dentro del desarrollo en la parte del cliente, destacan principalmente Angular, React y Vue, cada uno tiene unas características diferentes.

React es una biblioteca Javascript de código abierto diseñada para crear interfaces de usuario con el objetivo de facilitar el desarrollo de aplicaciones en una sola página. Es mantenido por Facebook y la comunidad de software libre.

- **Ventajas:** Es una biblioteca JavaScript muy popular para construir interfaces de usuario. Es conocida por su flexibilidad, rendimiento y vasto ecosistema de librerías y herramientas. Utiliza un DOM virtual para actualizaciones eficientes y su arquitectura basada en componentes promueve la reutilización del código.
- **Desventajas:** Está enfocado en los componentes de la interfaz de usuario, por lo que necesitaremos librerías adicionales para el enrutamiento, la administración de estado y otras funcionalidades.

Vue.js es un framework de JavaScript de código abierto para la construcción de interfaces de usuario y aplicaciones de una sola página. Fue creado por Evan You, y es mantenido por él y por el resto de los miembros activos del equipo central que provienen de diversas empresas como Netlify y Netguru

- **Ventajas:** Es conocido por su facilidad de uso, curva de aprendizaje suave y equilibrio entre características y flexibilidad. Incorpora conceptos tanto de Angular como de React, ofreciendo una estructura familiar para los desarrolladores de Angular y un enfoque simplificado para los nuevos en frameworks. Vue proporciona enlace de datos bidireccional y un DOM virtual para actualizaciones eficientes.
- **Desventajas:** Si bien el ecosistema está creciendo, puede que no sea tan extenso como el de React o Angular. Vue.js podría no ser ideal para aplicaciones empresariales a gran escala debido a su enfoque en la simplicidad.

3.- Instalación de Angular

Para instalar Angular en nuestro sistema local, necesitamos tener instalado una versión de **Node.js**. Algunas versiones de Angular requieren una versión mínima, para comprobar nuestra versión instalada utilizar **node -v**
Angular depende de paquetes npm para muchas características y funciones Al instalar Node.js en nuestro equipo deberíamos tener disponible el gestor de paquetes npm, para comprobarlo utilizar **npm -v**

Para poder utilizar Angular deberemos instalar el **CLI (Command Line Interface)** de Angular.
El CLI de Angular es una herramienta fundamental para el desarrollo de aplicaciones Angular. Se trata de una interfaz que nos permite interactuar con Angular desde la terminal y automatizar varias tareas comunes.

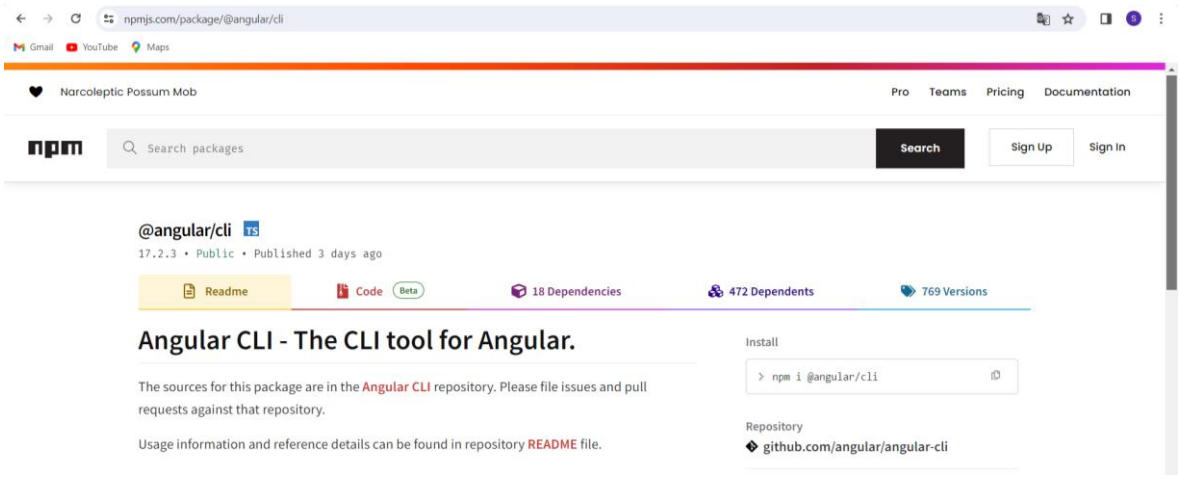
Algunas de las funciones principales del CLI son:

- **Inicializar proyectos:** Podemos crear nuevos proyectos de Angular desde cero con una estructura de carpetas y archivos predefinida.
- **Generar componentes, servicios y otros elementos:** El CLI nos permite generar rápidamente componentes, servicios, módulos y otros elementos que necesitemos en nuestra aplicación.
- **Desarrollar localmente:** El CLI incluye un servidor de desarrollo integrado que nos permite ejecutar tu aplicación Angular localmente para probarla y depurarla.
- **Compilar y optimizar la aplicación:** Podemos utilizar el CLI para compilar nuestra aplicación Angular en un formato optimizado para producción.
- **Ejecutar pruebas unitarias y end-to-end:** El CLI nos ayuda a ejecutar las pruebas unitarias y end-to-end de nuestra aplicación.

Para instalar el CLI actual de angular deberemos ejecutar en la consola:

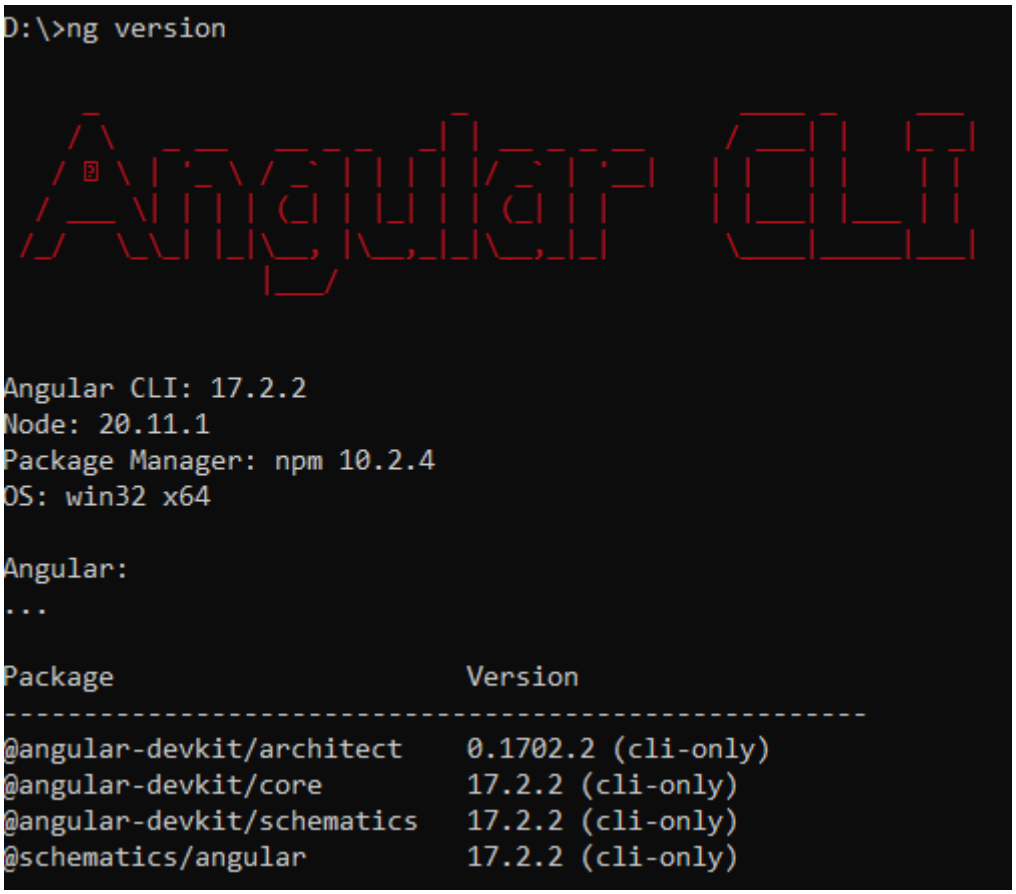
```
npm install -g @angular/cli
```

Si quisiéramos instalar una versión anterior a la actual, deberemos de instalar su correspondiente paquete npm, para poder comprobar la versiones disponibles podemos visitar la página https://www.npmjs.com/package/@angular/cli



Desde aquí podremos obtener la versión adecuada

Una vez instalado el CLI de angular podemos comprobar su versión con el comando **ng versión**



4.- Creación y ejecución de un proyecto

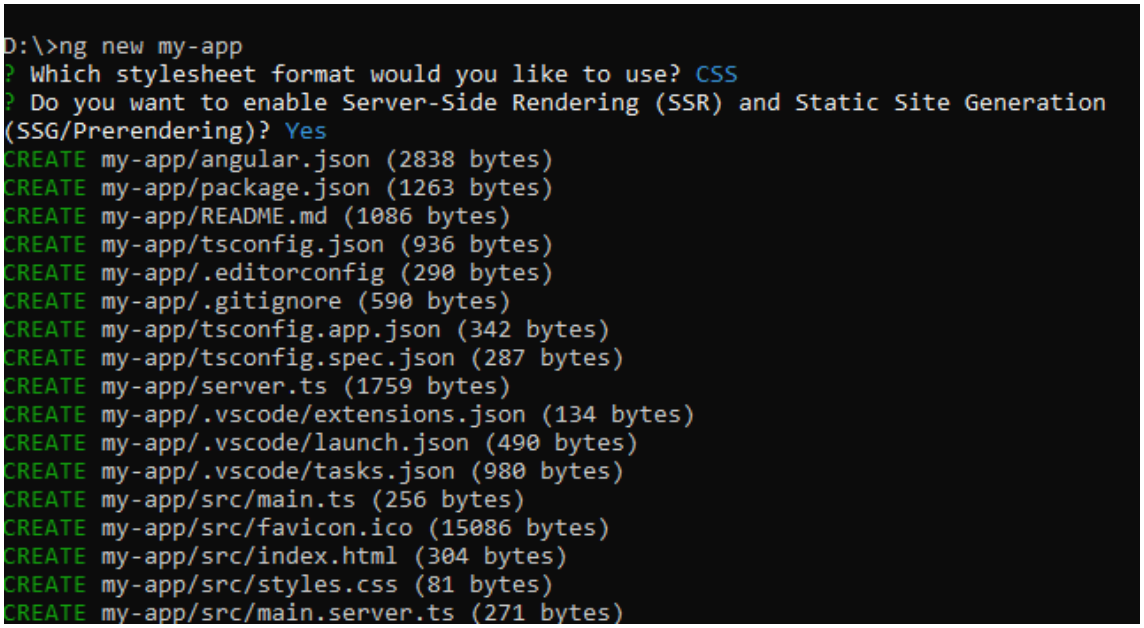
Una vez que ya tenemos instalado el CLI de Angular ya podemos crear nuestro primer proyecto. Angular CLI se utiliza desde la consola y tiene todos los comandos necesarios para poder realizar cualquier acción necesaria en Angular. Para hacer referencia al CLI utilizaremos **ng** y el comando que necesitemos.

Para crear un proyecto nuevo deberemos utilizar el comando **new** seguido del nombre del espacio de trabajo, es decir el directorio donde se creará nuestro proyecto

Para crear un proyecto en Angular llamado my-app deberemos ejecutar:

```
ng new my-app
```

Esto lo que hará es crear la estructura básica de un proyecto de Angular e instalar los paquetes y las dependencias necesarias. Durante este proceso nos suele preguntar por las características de nuestro proyecto, dependiendo de la versión de Angular pueden variar las preguntas que nos haga.



Angular CLI incluye un servidor para que podamos crear y servir nuestra aplicación localmente. Para poder ejecutar nuestro proyecto deberemos:

1. Situarnos en la carpeta del espacio de trabajo, como my-app.
2. Ejecutar el siguiente comando:

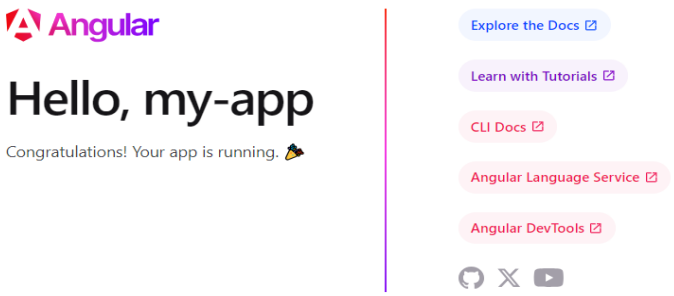
```
cd my-app
ng serve --open
```

El comando **ng serve inicia el servidor, observa sus archivos y reconstruye la aplicación** a medida que realiza cambios en esos archivos.
La opción **--open** (o simplemente -o) abre automáticamente el navegador http://localhost:4200/ y ejecuta el proyecto

También podemos ejecutar el proyecto con el comando npm start de esta manera estamos llamando al script del package.json y el resultado es el mismo.

```
"scripts": {
    "ng": "ng",
    "start": "ng serve",
    "build": "ng build",
    "watch": "ng build --watch --configuration development",
    "test": "ng test",
    "serve:ssr:my-app": "node dist/my-app/server/server.mjs"
  }
```

En el navegador se nos cargara el proyecto creado. Según la versión la pantalla será diferente, pero siempre suele ser una página web que explica las bases del framework y donde encontrar información sobre Angular.



5.- Live reload

Angular CLI instala y configura un conjunto de herramientas que nos harán la vida más fácil. Entre otras, destaca la capacidad de **recargar la aplicación en caliente** en cuanto guardamos nuestro trabajo como programador.

Para probarlo sólo tenemos que dejar arrancada la aplicación y **cambiar un fichero de html o código y comprobar el resultado** en el navegador.

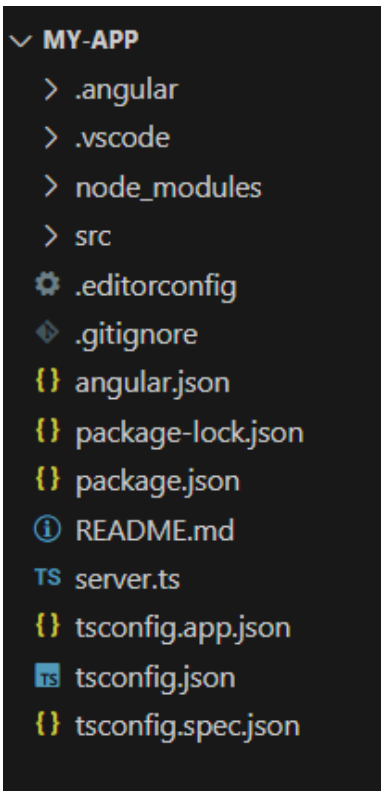
Vamos a modificar el fichero app.component.html, para ello lo abrimos y eliminamos todo lo que hay. Una vez eliminado el contenido que se había generado al crear el proyecto escribimos nuestro html. Si guardamos podremos comprobar cómo en el navegador **se habrá actualizado automáticamente**.

Toda **esta magia depende de una cadena de comandos** que lanzan distintas herramientas previamente instaladas y configuradas por el CLI. Entre ellas cabe mencionar a ***WebPack***, un coloso que afortunadamente viene instalado y preparado para funcionar de manera transparente.

6.- Scaffolding en Angular

El **scaffolding** en Angular hace referencia a la estructura de carpetas que se crea al generar un proyecto nuevo con ng new. Esto nos permite crear rápidamente la estructura básica de un proyecto y nos ayuda a ahorrar tiempo y esfuerzo al generar automáticamente los archivos y componentes necesarios para empezar a trabajar.

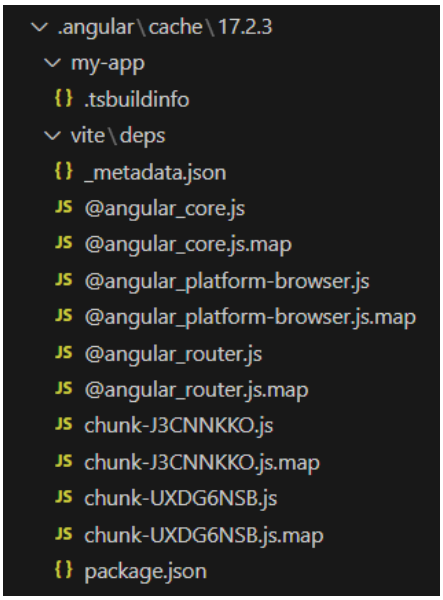
6.1.- Raíz del proyecto



En esta carpeta podemos encontrar las subcarpetas .angular, .vscode, node_modules y src. También podemos encontrar :

- **angular.json:** Configuración de Angular.
- **package.json:** Dependencias utilizadas
- **package-lock.json**: Versiones dependencias instaladas por npm
- **README.md**: Información básica de uso
- **server.ts**: Está relacionado con el renderizado del lado del servidor (SSR). Sólo aparece si hemos seleccionado SSR durante la creación del proyecto
- **tsconfig.json:** Configuración de TypeScript
- **tsconfig.app.json**: Configuración TypeScript para ese proyecto
- **tsconfig.spec.json:** Configuración pruebas unitarias

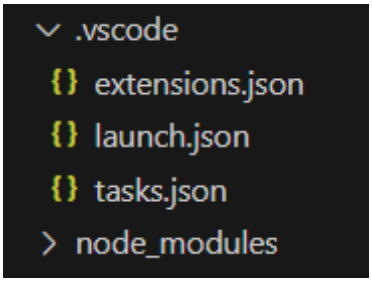
6.2.- Carpeta .angular



En esta carpeta podemos encontrar:

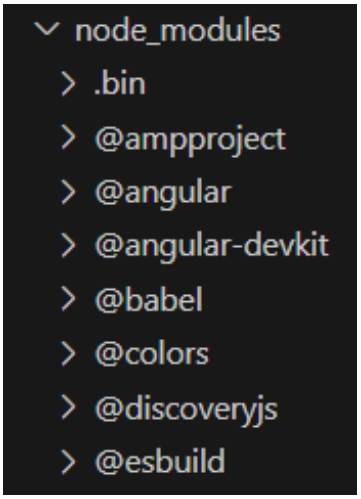
- **my-app:** esta carpeta contiene el fichero tsbuildinfo, que es un archivo generado por el compilador TypeScript en determinadas situaciones durante el proceso de compilación incremental.
- **vite:** No es una carpeta propia de Angular. Vite es una herramienta de construcción frontend conocida por su experiencia de desarrollo rápido.

6.3.- Carpeta .vscode



Esta carpeta contiene archivos de configuración específicos de VSCode que se utilizan para personalizar la experiencia de desarrollo dentro del editor.

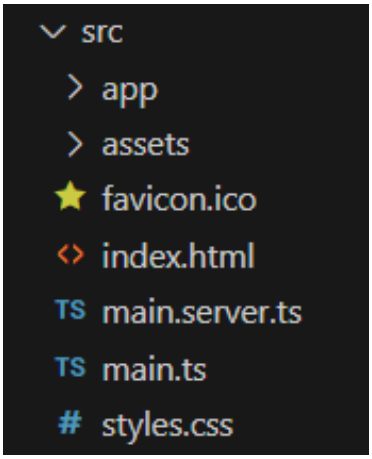
6.4.- Carpeta node_modules



Es un componente esencial para la gestión de dependencias en proyectos Node.js. Nos permite aprovechar librerías externas y facilita la organización y mantenimiento de nuestro código.

La carpeta node_modules suele ser bastante grande y contener muchos archivos. Se recomienda ignorarla en nuestro sistema de control de versiones (como Git) ya que su contenido se puede regenerar al instalar las dependencias nuevamente.

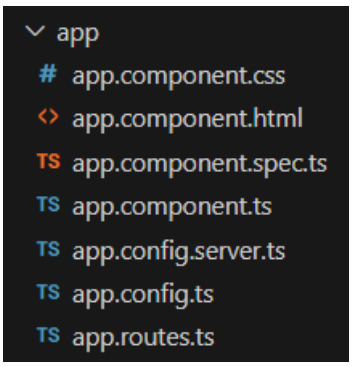
6.5.- Carpeta src



Es una de las más importantes en un proyecto, es donde se encuentra la mayor parte del código. Incluye las subcarpetas app y assets.

- **fav.icon:** Icono de la aplicación
- **index.html:** El fichero de inicio de la aplicación
- **main.server.ts:** Fichero ts para SSR
- **main.ts:** Bootstraping del proyecto
- **styles.css:** Estilos globales del proyecto

6.5.1- Carpeta app



Es un directorio crucial que contiene la mayor parte del código fuente de la aplicación. Es donde se define la estructura, la lógica y el comportamiento de la aplicación.
Es el directorio en el cual vamos a estar trabajando en el desarrollo de nuestra aplicación.
Es donde se definen los componentes, inicialmente se crea un componente llamado **app.component**. Para cada componente se generan 4 archivos:

- **.css:** Contiene el css de ese componente
- **.html:** Contiene el html del componente
- **.ts:** Contiene la lógica del componente, código en TypeScript
- **.spec.ts:** Contiene la lógica para las pruebas de test

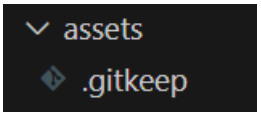
Además, incluye los archivos:

- **app.config.server.ts**: Usado para el SSR (importado en main.server.ts)
- **app.config.ts:** Usado para el BootStraping (importado en main.server.ts)
- **app.routes.ts**: Fichero utilizado para el enrutado de los componentes

A partir de la versión 17 desaparece el app.module.ts. Angular a partir de esta versión recomienda el **uso de componentes standalone** y elimina el app.module.ts como modulo principal de la aplicación. Ahora las dependencias de uso de componentes y módulos se realiza directamente sobre cada componente.

El CLI de Angular nos crea cada elemento de Angular (Componentes, Modelos, Directivas, Servicios, Guardas, Pipes, Módulos…) por defecto en esta carpeta, con lo cual generalmente se suele crear una carpeta para cada tipo de elemento, de esta manera la organización del proyecto es mucho más clara.

6.5.2- Carpeta assets



Es un directorio especial que almacena **archivos estáticos** que nuestra aplicación utiliza.

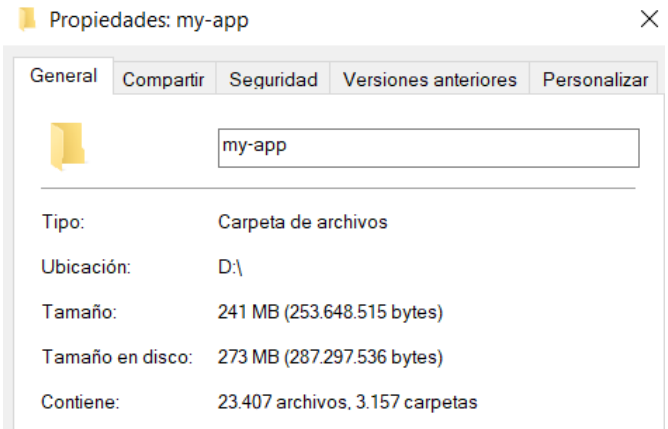
Estos archivos no forman parte del código fuente de la aplicación y no se procesan por Angular durante la compilación.

En este directorio se suelen guardar archivos de tipo:

- **Imágenes (PNG, JPG, GIF, SVG, etc.):** Logos, iconos, fondos de pantalla y otras imágenes utilizadas en la interfaz de usuario.
- **Fuentes (TTF, OTF, WOFF, etc.):** Fuentes personalizadas que queramos usar en nuestra aplicación.
- **Archivos de audio (MP3, OGG, WAV, etc.):** Sonidos y música de fondo.
- **Archivos de vídeo (MP4, WebM, etc.):** Vídeos utilizados en la aplicación**.**
- **Archivos de datos (JSON, CSV, etc.):** Datos que nuestra aplicación necesita para funcionar, pero que no se va a modificar por la propia aplicación.

7.- Modo producción

Un proyecto creado en Angular sin hacer nada ya ocupa un tamaño considerable de espacio y un numero de archivos muy grande.

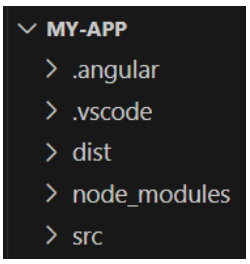


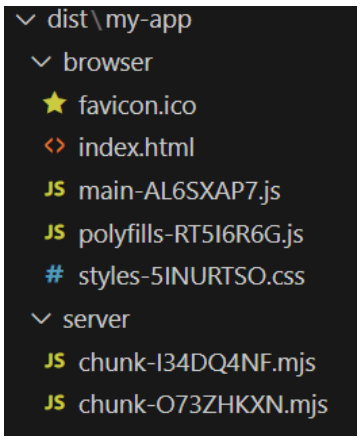
Angular nos permite trabajar en local en un **modo de desarrollo** y posteriormente cuando el proyecto esté finalizado generar los archivos necesarios para le ejecución del proyecto en un servidor Web. Esto es el **modo producción**.

Antes de implementar la aplicación web, Angular proporciona una forma de verificar el comportamiento de la aplicación web con la ayuda de algunos comandos CLI. Por lo general, el comando **ng serve** se usa para compilar, observar y servir la aplicación desde la memoria local. Pero para la implementación, el comportamiento de la aplicación se ve ejecutando el comando **ng build**.

ng serve	ng build
El comando ng serve sirve para desarrollos rápidos, locales e iterativos y también para compilar, observar y servir la aplicación desde un servidor de desarrollo CLI local.	El comando ng build sirve para compilar las aplicaciones e implementar los elementos de compilación.
El comando no genera una carpeta de salida.	La carpeta de salida es **dist/.**
El servicio ng crea elementos desde la memoria para una experiencia de desarrollo más rápida.	El comando ng build genera archivos de salida solo una vez y no los sirve.

ng build compila la aplicación Angular en un directorio de salida llamado **dist/** en la ruta de salida dada.





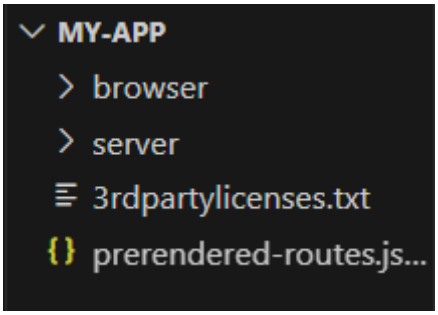
Este comando debe ejecutarse desde el directorio de trabajo. El creador de aplicaciones en Angular usa la herramienta de creación de paquetes web, con

opciones de configuración especificadas en el archivo de configuración del espacio de trabajo (angular.json) o con una configuración alternativa con nombre.

La carpeta dist es la carpeta de compilación que contiene todos los archivos y carpetas que se pueden alojar en el servidor. La carpeta dist contiene el código compilado de su aplicación angular en formato JavaScript y también los archivos HTML y CSS necesarios.

```
D:\my-app>ng build
- Building...
Browser bundles
Initial chunk files     | Names               |  Raw size | Estimated transfer size
main-AL6SXAP7.js        | main                | 180.92 kB |               49.55 kB
polyfills-RT5I6R6G.js   | polyfills           |  33.10 kB |               10.72 kB
styles-5INURTSO.css     | styles              |   0 bytes |                0 bytes

                        | Initial total       | 214.03 kB |               60.27 kB


Server bundles
Initial chunk files     | Names               |  Raw size
server.mjs              | server              |   1.05 MB |
chunk-I34DQ4NF.mjs      | -                   | 495.63 kB |
polyfills.server.mjs    | polyfills.server    | 261.67 kB |
chunk-VVCT4QZE.mjs      | -                   |   2.48 kB |
render-utils.server.mjs | render-utils.server |   1.41 kB |
chunk-ODB4TVJZ.mjs      | -                   | 521 bytes |
main.server.mjs         | main.server         | 149 bytes |

Lazy chunk files        | Names               |  Raw size
chunk-O73ZHKXN.mjs      | xhr2                |  11.80 kB |
Prerendered 1 static route.
Output location: D:\my-app\dist\my-app

Application bundle generation complete. [14.234 seconds]
```

Ahora ya tenemos los archivos necesarios para alojar en un servidor, para comprobarlo deberemos de abrir el proyecto con un servidor Web y abrir el archivo index.html de la carpeta browser



Podemos hacerlo con Live Server que viene incluido en VS Code