# Setup

Para desarrollar una aplicación usando Angular en un sistema local, necesitamos configurar un entorno de desarrollo que incluya la instalación de:

* Node.js (min version required 10.13.0) and npm (min version required 6.11.0)
* Angular CLI
* Visual Studio Code

Instalar Node.js and Visual Studio Code desde sus páginas oficiales.

Pasos para instalar Angular CLI

* Angular CLI puede ser instalado usando node package manager con el comando siguiente: *npm install -g @angular/cli*
* Probar la correcta instalación con el siguiente comando: *ng v*

Angular CLI es una herramienta de interfaz de línea de comandos para construir aplicaciones Angular. Hace que el desarrollo de aplicaciones sea más rápido y fácil de mantener.

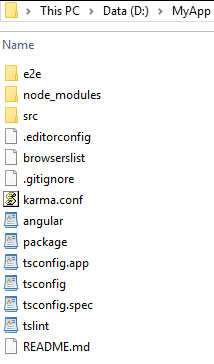
Con CLI, podemos crear proyectos, agregarle archivos y realizar tareas de desarrollo como pruebas, agrupación e implementación de aplicaciones.

* npm install -g @angular/cli: Installs Angular CLI globally
* ng new <project name>: Creates a new Angular application
* ng serve --open: Builds and runs the application on lite-server and launches a browser
* ng generate <name>: Creates class, component, directive, interface, module, pipe and service
* ng build: Builds the application

## Creando una aplicación:

1. Cree una aplicación con el nombre 'MyApp' usando el siguiente comando cli: *ng new MyApp*
2. El comando anterior mostrará dos preguntas. La primera pregunta es para crear un archivo routing module (app-routing.module.ts) que se explicará más adelante.
3. La siguiente pregunta es para seleccionar la hoja de estilo a usar en la aplicación. Seleccione CSS.

Podremos ver un ejemplo de la estructura de la aplicación en la imagen projectStructure.



* e2e /: esta carpeta contiene todas las pruebas End to End (e2e) de la aplicación escrita en Jasmine y ejecutada por el Protractor.
* node\_modules /: Node.js crea esta carpeta y coloca todos los módulos npm instalados como se enumeran en package.json.
* src /: todos los archivos relacionados con la aplicación se almacenarán en su interior.
* angular.json: archivo de configuración para Angular CLI donde establecemos varios valores predeterminados y también configuramos qué archivos se incluirán durante la compilación del proyecto.
* package.json: este es un archivo de configuración de nodo que contiene todas las dependencias necesarias para Angular.
* tsconfig.json: este es el archivo de configuración de Typescript donde podemos configurar las opciones del compilador.
* tslint.json: este archivo contiene reglas de linting preferidas por la guía de estilo angular.

Una vez se tiene la aplicación se debe usar "*npm install*" para instalar node\_modules.

Luego para ejecutar la aplicación se utiliza: *ng serve --open*.

# Components & Modules

## Concepts

¿Por qué componentes en angular?

Un componente es el componente básico de una aplicación Angular.

Los componentes enfatizan la separación de problemas y cada parte de la aplicación angular puede escribirse independientemente una de la otra

Los componentes son reutilizables.

Por ejemplo, observe en la aplicación mCart que el componente superior es el componente mCart (AppComponent) que consta de componentes secundarios llamados componente de bienvenida, componente de inicio de sesión, etc.

Revisar:

* app.component.ts
* app.component.html
* app.component.css

Ya hemos visto cómo crear un componente, ahora exploremos los módulos.

* Los módulos en Angular se utilizan para organizar la aplicación. Las aplicaciones Angular son una colección de módulos.
* Un módulo en Angular es una clase con el decorator @NgModule agregado. Los metadatos de @NgModule contendrán las declaraciones de componentes, pipes, directivas, servicios que se utilizarán en toda la aplicación.
* Cada aplicación Angular debe tener un módulo raíz que se carga primero para iniciar la aplicación.
* También podemos tener submódulos que deben configurarse en el módulo raíz.

### Root Module

Revisar:

* app.module.ts localizado dentro del folder app.
* main.ts localizado dentro del folder src.

### Cargando el root component en la página HTML.

Revisar:

* index.html bajo el folder src.

Para ejecutar la aplicación: *ng serve --open*

Para ejecutar la aplicación con otro puerto: *ng serve --open --port 3000*

### Creando un component:

***ng generate component hello***: Este comando generará un folder dentro de app con un archivo ts, html y css.

* En app.module.ts debe agrearse el archivo ts nuevo en bootstrap si se desea colocar como raíz.
* En main.ts debe cambiarse la referencia en body al nuevo html (<app-hello>).

## Template

El lenguaje predeterminado para templates en Angular es HTML. Los templates representan la vista cuyo objetivo es mostrar data y modificarla cuando sea que un evento ocurra.

¿Por qué templates?

Los templates nos permiten separar la capa de visualización del framework. Podremos cambiar esta vista sin romper la aplicación.

Podemos definir un template en dos formas:

* Inline template: Dentro del mismo de la misma clase componente. Dentro del decorator @Component, en el property template (en lugar de templateUrl) podremos agregar el HTML.

@Component({

  selector: 'app-root',

  template: `

        <h1> Welcome </h1>

        <h2> Course Name: {{ courseName }}</h2>

    `,

  styleUrls: ['./app.component.css']

})

* External template: es el predeterminado en Angular

### Elementos de Template

* HTML
* Interpolation: La interpolación es una de las formas de enlace de datos (data binding) donde podemos acceder a los datos del componente en un template. Para la interpolación, utilizamos llaves dobles {{}}.
* Template Expressions: El texto dentro de {{}} es llamado template expresion: {{ expression }}. Angular primero evalúa la expresión y devuelve el resultado como una cadena. El alcance de un template expression es una instancia de componente. Eso significa que, si escribimos {{ courseName }}, courseName debería ser propiedad del componente al que está vinculada este template.
* Template Statements: son las declaraciones que responden a un evento de usuario:

(event) = statement

Por ejemplo (click) = "changeName()"

Esto es llamado event binding. En Angular todos los eventos deben ir en ().

Ejemplo en hello.component.

## Change Detection

Ahora exploremos cómo Angular detecta los cambios y los actualiza en la aplicación en los lugares respectivos. Las aplicaciones Angular se ejecutan más rápido que las aplicaciones Angular 1.x debido a su mecanismo mejorado de detección de cambios.

Entonces, ¿qué es el mecanismo de detección de cambios y cómo ayuda a ejecutar aplicaciones angulares tan rápido?

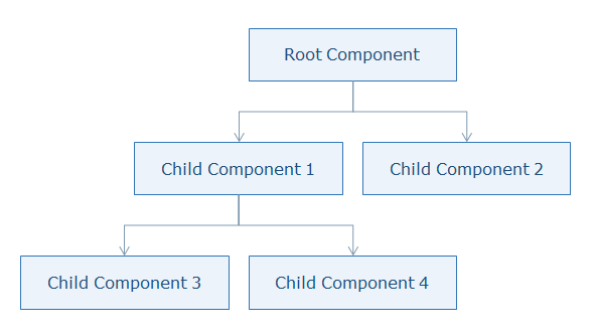
* La detección de cambios es un proceso en Angular que mantiene las vistas sincronizadas con los modelos.
* En Angular, el flujo es unidireccional de arriba hacia abajo en un árbol de componentes. Un cambio en una aplicación web puede ser causado por eventos, llamadas Ajax y temporizadores que son asíncronos.

¿Quién informa a Angular sobre los cambios?

* Zones informa a Angular sobre los cambios en la aplicación. Detecta automáticamente todas las acciones asincrónicas en tiempo de ejecución en la aplicación.

Ahora veamos qué hace Angular una vez que se detecta un cambio:

* Angular ejecuta un algoritmo detector de cambios en cada componente de arriba a abajo en el árbol de componentes. Este algoritmo detector de cambios se genera automáticamente en tiempo de ejecución, lo que verificará y actualizará los cambios en los lugares apropiados del árbol de componentes.
* Angular es muy rápido pese a que pasa por todos los componentes de arriba a abajo para cada evento, ya que genera código amigable para VM. Debido a esto, Angular puede realizar cientos de miles de cheques en pocos milisegundos.



# Use Directives

# Implement data binding

# Apply pipes

# Create nested components and share data between them

# Add validations to the forms

# Create services

# Implement routing