
ANGULAR 15

# CONTENIDO

- Angular.
- Instalación.
- Commandos.
- Estructura (Scaffolding).
- Components.
- Routing.
- Directives.
- Interfaces.
- Pipes.
- Guards.
- Data Binding.
- Comunicación entre componentes.
- Manipulación del DOM.
- Services.
- Interceptors.
- async-await.
- RxJS.

# Interfaces

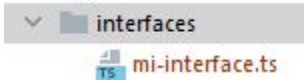
Una interface define la estructura de un objeto. Este especifíca las propiedades y sus tipos, pero no provee una implementación.

### Comando

```
ng generate interface interface-name [options]
```

### Resultado

```
├── .../
└──────── mi-interface.ts
```




### Definición

```
export interface MiInterface {
      id: string;
      numero: number;
      fecha: Date;
      estado: string;
      descripcion: string | null;
}
```

# Interfaces

## Uso

```
//variables
myVar: MiInterface;
myVar: MiInterface[];


//asignaciones
this.myVar = {
                id: '1a2b3c',
                numero: 123,
                fecha: new Date(),
                estado: 'activo',
                descripcion: null
            }

this.myVar = datos;

 //funciones
getSomething(something: MiInterface) { ... }
```

# Pipes

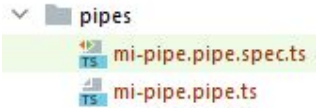
Un pipe es una tubería o canal que se denota con el carácter **|**. Una de las ventajas de utilizarlas es que se pueden usar en toda la aplicación.

### Comando

```
ng generate pipe pipe-name [options]
```

### Resultado

```
├── .../
├──────── mi-pipe.pipe.spect.ts
└──────── mi-pipe.pipe.ts
```




```
...
export class MiPipePipe implements PipeTransform {

      transform(value: unknown, ...args: unknown[]): unknown {
            return null;
      }

}
```

# Pipes

## Flujo de información


## Uso

```
<p> {{ nombre | lowercase }} </p>
```


```
<p> {{ nombre | uppercase }} </p>
```


```
<p> {{ importe | currency:'ARS': 'symbol' : '4.2-2' }} </p>
```

# Pipes

## Definición

```
...
export class MiPipePipe implements PipeTransform {

      transform(nombre: string): string {
            return 'El nombre es: ' + nombre;
      }

}
```

# Guards

Los Guards son servicios que permiten controlar el acceso a ciertas partes de una aplicación Angular.

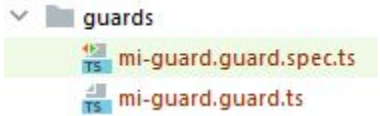
## Comando

```
ng generate guard guard-name [options]
```

```
? Which type of guard would you like to create? (Press <space> to select, <a> to toggle all, <i> to invert selection, and <enter> to proceed)
>(*) CanActivate
 ( ) CanActivateChild
 ( ) CanDeactivate
 ( ) CanLoad
 ( ) CanMatch
```

## Resultado

```
├── .../
├──────── mi-guard.guard.spect.ts
└──────── mi-guard.guard.ts
```



# Guards

**Resultado**

```
...
export class MiGuardGuard implements CanActivate {

      canActivate(route: ActivatedRouteSnapshot,state: RouterStateSnapshot):
            Observable<boolean | UrlTree> | Promise<boolean | UrlTree> | boolean | UrlTree {

            return true;
      }

}
```

# Guards

## Definición

```
...
export class MiGuardGuard implements CanActivate {

      canActivate(route: ActivatedRouteSnapshot,state: RouterStateSnapshot):
            Observable<boolean | UrlTree> | Promise<boolean | UrlTree> | boolean | UrlTree {

            let url: string = state.url;
            return this.checkLogin(url);
      }

      checkLogin(url: string): boolean {
            if (sessionStorage.getItem('TOKEN')) {
                  return true;
            }

            return false;
      }
}
```