**Angular Notes**

* **Angular Command Line**

**Angular install**

npm install -g @angular/cli

ng new <project name>

cd <project name>

**run the files**

ng serve

**Add bootstrap files**

npm install bootstrap@4.6.0

**Add model class**

ng g class yourclass –type=model -–skip-tests

**Add ng lint**

ng add @angular-eslint/schematics

**Add ng pwa**

ng add @angular/pwa@12.0.0

**Add ngrx**

ng add @ngrx/store

**Add ngrx effects**

ng add @ngrx/effects

**Add devtools**

npm install --save-dev @ngrx/store-devtools

**Generate files**

ng g c <Component name>

ng g c -is <Component name> -–skip-tests(inline style)

ng g m <Module name> --routing

ng g cl <Class name>

ng g s <service name>

ng g p <pipe name>

ng g d <directive name>

ng g i <interface name>

**Rxjs update**

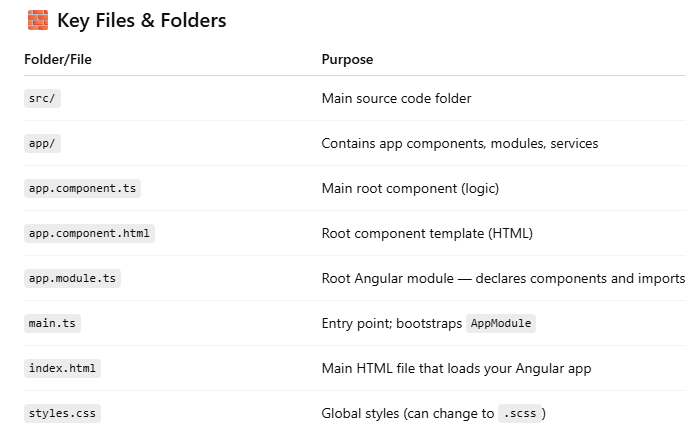
npm i rxjs@latest

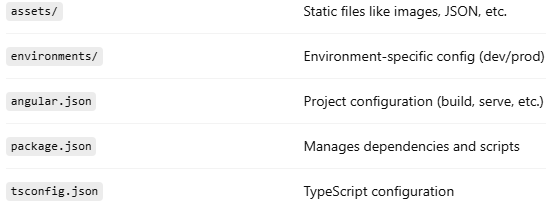
**guard**

ng g guard <guard name>

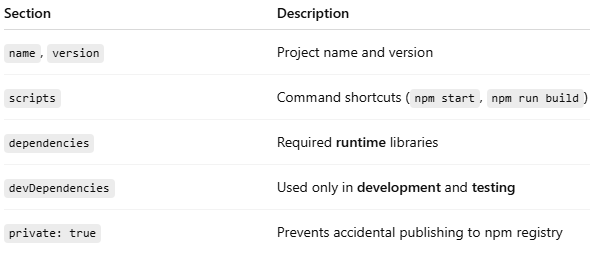
* **Angular Structure**



****

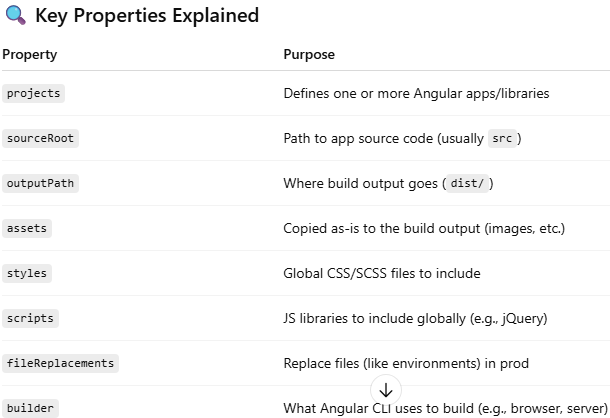
****

* **Anguar Package JSON**

****

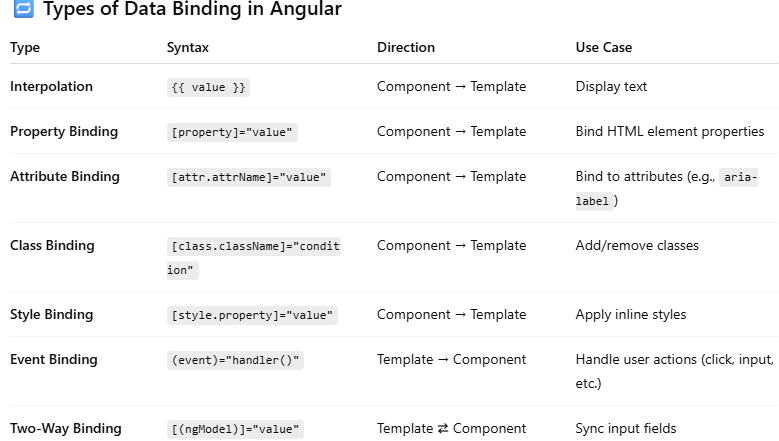
****

* **Angular Angular Json**

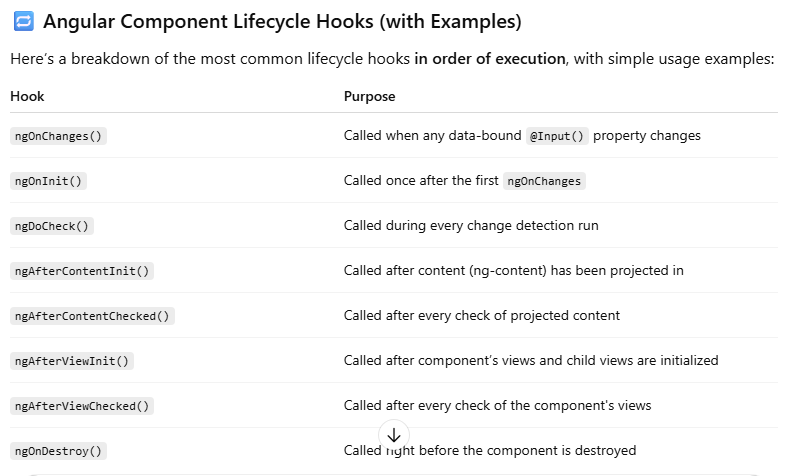
****

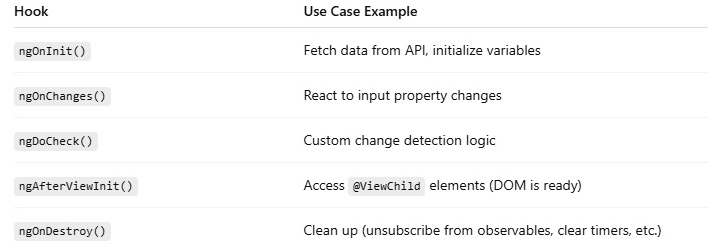
**** ****

* **Angular Databinding**

****

* **Angular Lifecycle**

****



* **Angular Component**

@Component({

selector: 'app-hello', // Used as use Element/Attribute/Class

templateUrl: './hello.component.html', // HTML template

template: `<h1>Hello, {{ name }}!</h1>` //Define html tag using backticks

styleUrls: ['./hello.component.css'] // CSS styles

styles: [` //Define all css using backticks

h1 {

color: green;

font-size: 24px;

}

`]

})



* **App Module**

@NgModule({

declarations: [

AppComponent, // declare all components, pipes, directives here

HelloComponent

],

// import necessary Angular modules like HttpClientModule, any module not using in lazyload etc

imports: [

BrowserModule,

FormsModule,

AppRoutingModule,

FormsModule,

ReactiveFormsModule,

HttpClientModule,

//StoreModule.forRoot(appReducer),

//StoreDevtoolsModule.instrument({ logOnly: environment.production })

//StoreModule.forRoot({counter: counterReducer}),

],

providers: [ // services (DI)

{

provide: HTTP\_INTERCEPTORS,

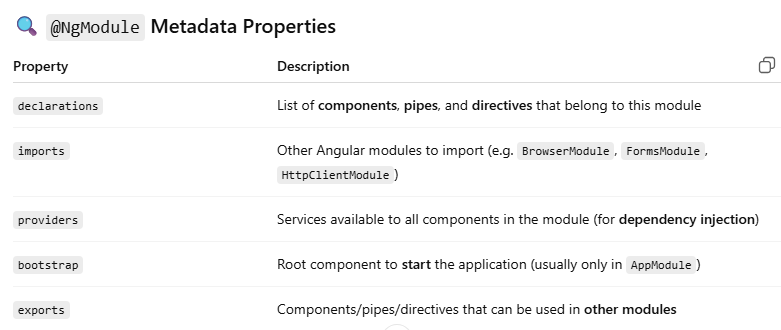
useClass: AuthInterceptor,

multi: true // important for supporting multiple interceptors

}

],

bootstrap: [AppComponent] // root component to bootstrap

})

* **Angular Service**

In **Angular**, a **service** is a class that contains **reusable business logic**, **data access**, or **shared state** that can be injected into components or other services using **dependency injection (DI)**.

import { HttpClient } from '@angular/common/http';

import { Injectable } from '@angular/core';

import { Observable } from 'rxjs';

@Injectable({ providedIn: 'root' })

//All logic Inside Class

export class DataService {

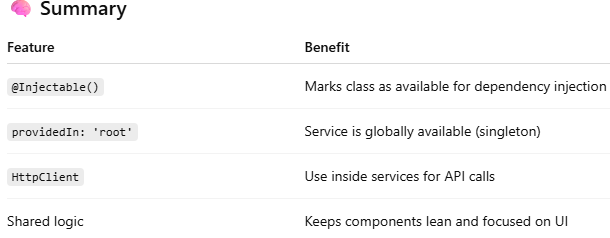
constructor(private http: HttpClient) {}

getUsers(): Observable<any> {

return this.http.get('https://jsonplaceholder.typicode.com/users');

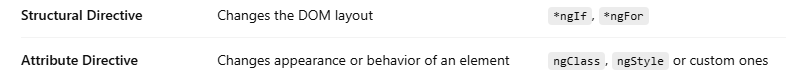
}

}

****

* **Angular Directive**

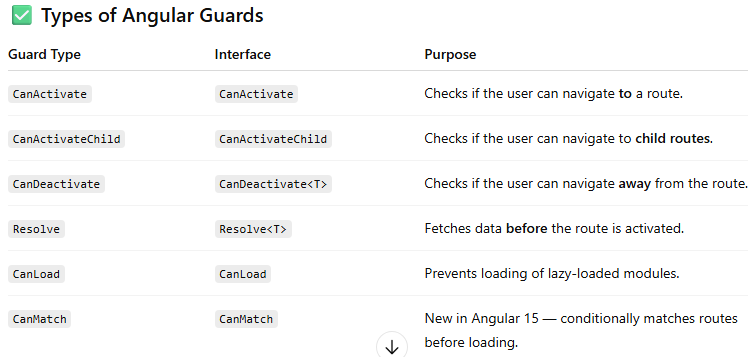
In **Angular**, a **directive** is a class that adds behavior or modifies the structure or appearance of elements in the DOM.

****

**Custom Directive**

* **Guard**

In **Angular**, **Guards** are used to control navigation — whether a user can access or leave a particular route. Guards implement specific interfaces and are typically used for **authentication**, **authorization**, **data fetching**, or **preventing unsaved changes from being lost**.

****

* **Router Navigate**

organisation/wl/project-detail?id=74412095&caf=64721797&projectId=64721766&clearence=3

**How implements in route navigate**

constructor(private router: Router) {}

this.router.navigate(

['organisation', 'wl', 'project-detail'],

{

queryParams: {

id: 74412095,

caf: 64721797,

projectId: 64721766,

clearence: 3

}

}

);

**OR**

this.router.navigate(

['organisation/wl/project-detail'],

{

queryParams: {

id: 74412095,

caf: 64721797,

projectId: 64721766,

clearence: 3

}

}

);

**what use state**

In Angular's Router.navigate, the state property is used to **pass data to the target route without using query parameters or route parameters**.

this.router.navigate(['/wlc-raise-eds', id], {

state: { name: 'Test', role: 'developer' }

});

**Target Component**

import { Router } from '@angular/router';

constructor(private router: Router) {

const data = this.router.getCurrentNavigation()?.extras.state;

console.log('Data received:', data);

}

**Below URL Implements**

wlc-agenda-summary/WL%2FAGENDA%2FNBWL%2F103588%2F2025/edit

this.router.navigate(['/wlc-agenda-summary', agendaId, 'edit']);

* **ActivatedRoute**

import { ActivatedRoute } from '@angular/router';

constructor(private route: ActivatedRoute) {

        this.route.params.subscribe((params) => {

            this.page = params['page'];

            this.agendaMomId = params['id'];

        });

}

**ActivatedRoute Interface looks like:**

interface ActivatedRoute {

snapshot: ActivatedRouteSnapshot

url: Observable<UrlSegment[]>

params: Observable

queryParams: Observable

fragment: Observable

data: Observable

outlet: string

component: Type | string | null

routeConfig: Route | null

root: ActivatedRoute

parent: ActivatedRoute | null

firstChild: ActivatedRoute | null

children: ActivatedRoute[]

pathFromRoot: ActivatedRoute[]

paramMap: Observable

queryParamMap: Observable

toString(): string

}



const routes: Routes = [

{

path: 'dashboard/:userId',

component: DashboardComponent,

data: { title: 'User Dashboard' },

children: [

{

path: 'reports',

component: ReportsComponent,

outlet: 'sidebar',

data: { title: 'User Reports' }

}

]

}

];

**Example URL:**

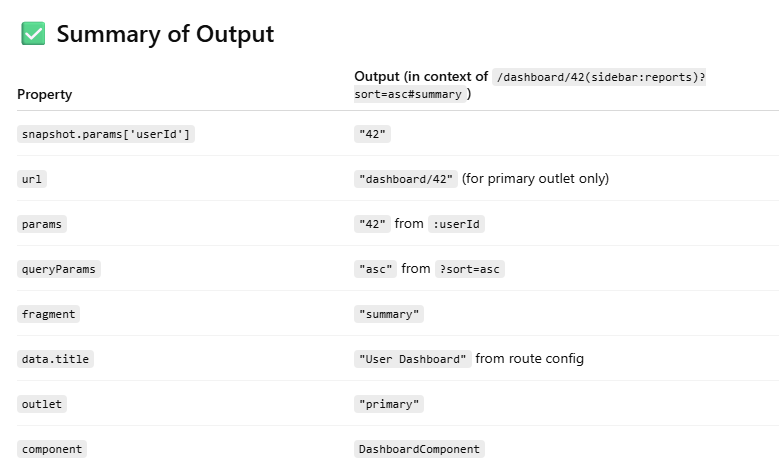
/dashboard/42(sidebar:reports)?sort=asc#summary





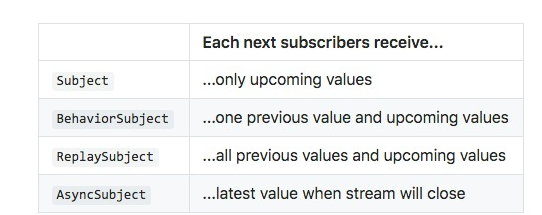


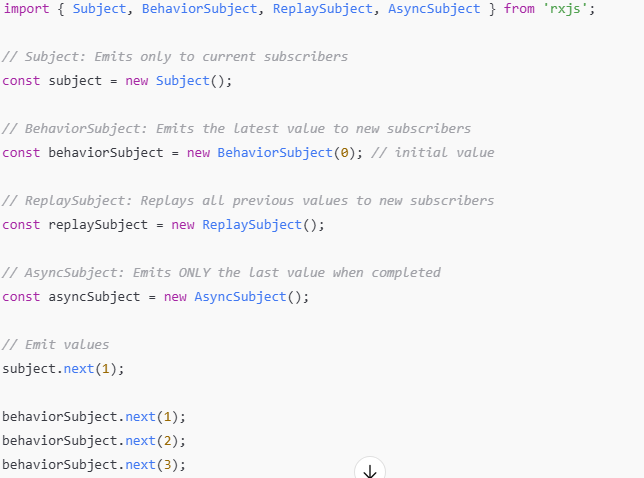


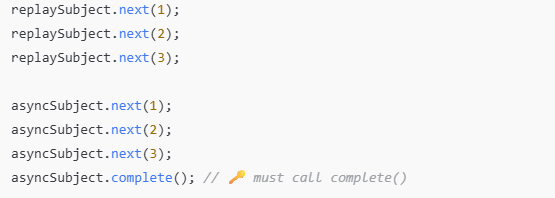




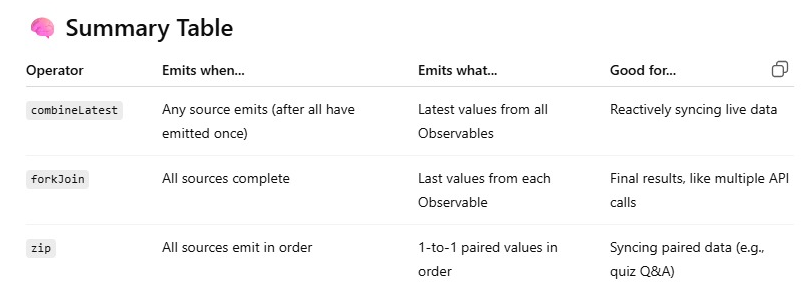
**RXJS Notes**

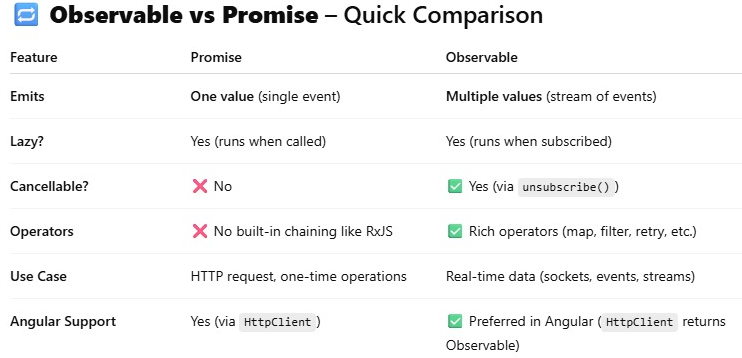
****

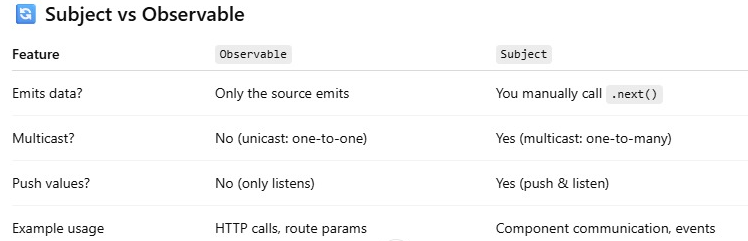




****

****

****

****

