**What is RxJS?**

RxJS is a **library for reactive programming using Observables**, to make it easier to compose asynchronous or callback-based code.

**🤝 RxJS + Angular**

Angular heavily uses RxJS under the hood. Some key areas where RxJS is used in Angular:

* **HTTP Requests**: HttpClient returns Observables.
* **Reactive Forms**: You can subscribe to form control value changes.
* **Routing**: Route parameters and events can be observed.
* **Event Handling**: You can observe user interactions.
* **State Management**: Libraries like NgRx and Akita use RxJS for state streams.

**🔧 Basic RxJS Concepts in Angular**

**1. Observable**

import { Observable } from 'rxjs';

const observable = new Observable(observer => {

observer.next('Hello from RxJS');

observer.complete();

});

observable.subscribe(value => console.log(value));

**2. Operators**

Operators help you transform or combine observables. Some popular ones:

* map
* filter
* switchMap
* debounceTime
* takeUntil

Example: Debouncing input

import { fromEvent } from 'rxjs';

import { debounceTime, map } from 'rxjs/operators';

fromEvent(inputElement, 'keyup')

.pipe(

debounceTime(300),

map(event => event.target.value)

)

.subscribe(value => console.log(value));

**3. Subjects**

A Subject is both an observable and an observer.

ts

CopyEdit

import { Subject } from 'rxjs';

const subject = new Subject();

subject.subscribe(data => console.log('Sub 1:', data));

subject.subscribe(data => console.log('Sub 2:', data));

subject.next('Hello Subscribers');

**4. switchMap Example in HTTP Call**

ts

CopyEdit

import { switchMap } from 'rxjs/operators';

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

this.searchControl.valueChanges.pipe(

debounceTime(300),

switchMap(value => this.http.get(`api/search?q=${value}`))

).subscribe(results => {

this.searchResults = results;

});

**📦 Common RxJS Use Cases in Angular**

| **Use Case** | **RxJS Feature** |
| --- | --- |
| HTTP calls | HttpClient + Observable |
| User input debounce | debounceTime, map |
| API cancel on new request | switchMap |
| Unsubscribe on destroy | takeUntil with Subject |
| Form changes | formControl.valueChanges |