1. formControl.valueChanges is Subject, if need initial value use pipe startWith

2. BehaviorSubject is readonly, bs.next(val) is not reassigning

3. BehaviorSubject always requires initial value (default)

4 pipe(…).subscribe(val => bs.next(val)) to avoid memory leak

5. If service used by one component, use @Injectable() and in component: providers: [Service]

In this case service is destroyed when component destroyed

6. pipes originating from events need to be ended with first() or takeUntilDestroyed(this.DestroyRef)

And then .subscribe(…). DestroyRef is injected and needed if pipe is in lc hooks (ngOnInit…)

7. for subscription best is to use |async in template: @if(obs$ | async; as obs){ }

8. Dynamic search 🡪 calling service:

formControl.valueChanges.pipe(

takeUntilDestroyed(),

filter(() => this.formControl.valid),

debounceTime(1000),

tap((searchVal: string) => …do SE here, like calling a service with searchVal…),

takeUntilDestroyed()

).subscribe()

9. constructor vs ngOnInit: 99% use ngOnInit (esp @Input, register external service call, reference props within template), use constructor if reacting to template actions, like valueChanges

10. declare interface for each service and implement it

11. result of http call is obs$:

obs$.pipe(

flatMap(data => data.postalCodes),

defaultEmpty(null) // or default value … if no result came from api

12. use constants like urls in environments/environment.ts

13. http req 1 -> req2: obs1$.pipe(switchMap((obs1Val) => http req2)

14. const x = signal(5) -- like useState(5)

const xx = computed(() => x \* 5) --- like useMemo

effect(() => …do some action referencing x…) --- like useCallback

x() to get value, x.set(8) to set value

15. if service holds result (as signal) of http call (obs$):

a. http -> obs$;

b. convert obs$ into Promise: firstValueFrom(obs$)

c. resultSignal.set(await promise) !! wrapped in async fun called from component, ex updateVal

d. component consumes resultSignal:

constructor(){

useEffect(() => this.normalVal = this.service.resultSignal() ) }

(see example in git weather-app-angular, signals branch, weatherService)