RxJS Operators and Pipelines

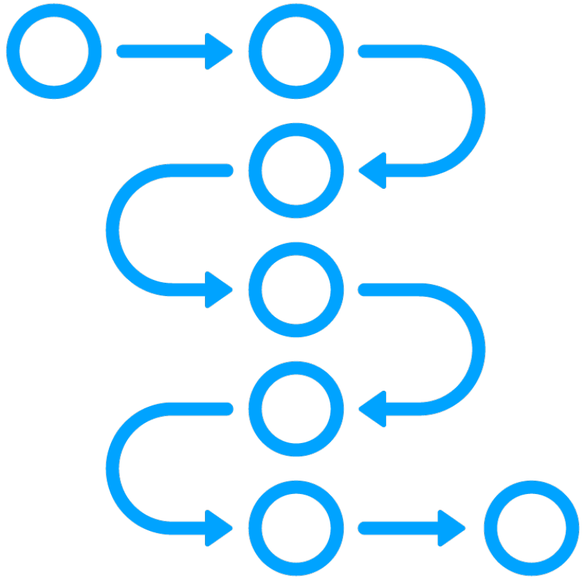
Deborah Kurata

Developer | Content Creator | MVP | GDE

@deborahkurata | https://www.youtube.com/@deborah_kurata

Two Types of RxJS Operators

Creation functions



Pipeable operators

RxJS Operators

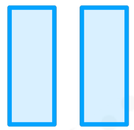
Transform

Catch and handle errors

Filter

Delay, timeout, ...

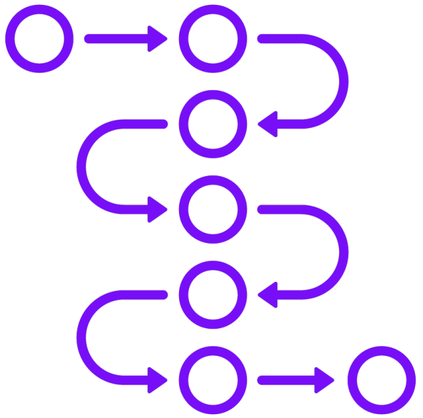
Combine, merge, ...



Throttle, debounce, ...

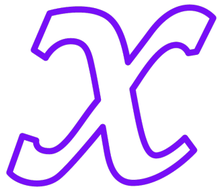
Operator Pipeline

```
of(2, 4, 6)
  .pipe(
    map(item => item * 2),
    tap(item => console.log(item))
  ).subscribe(item => console.log(item));
```

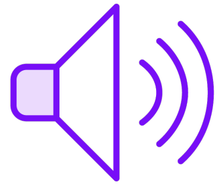


Coding Steps

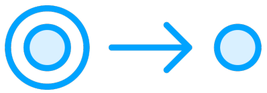
Write help message



Declare state



Create and subscribe to an observable



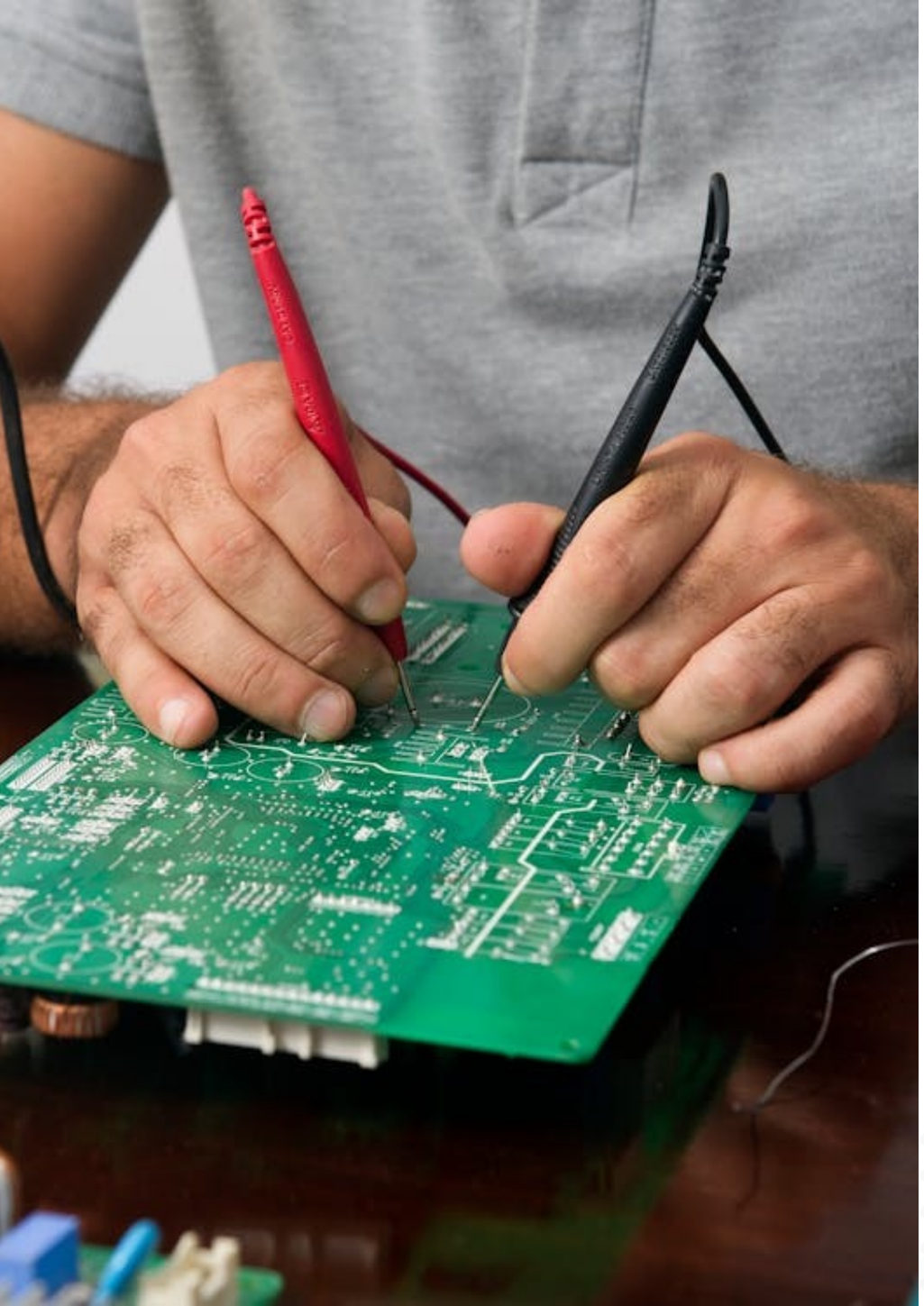
Transform event to key value

Filter the key values

Show/hide the help

RxJS Operator: map

```
of(2, 4, 6)
  .pipe(
    map(item => item * 2)
  )
  .subscribe(x => console.log(x));
```



RxJS Operator: `tap`

Taps into the emissions without affecting the item

```
tap(item => console.log(item))
```

For each item emitted in, the same item is emitted out

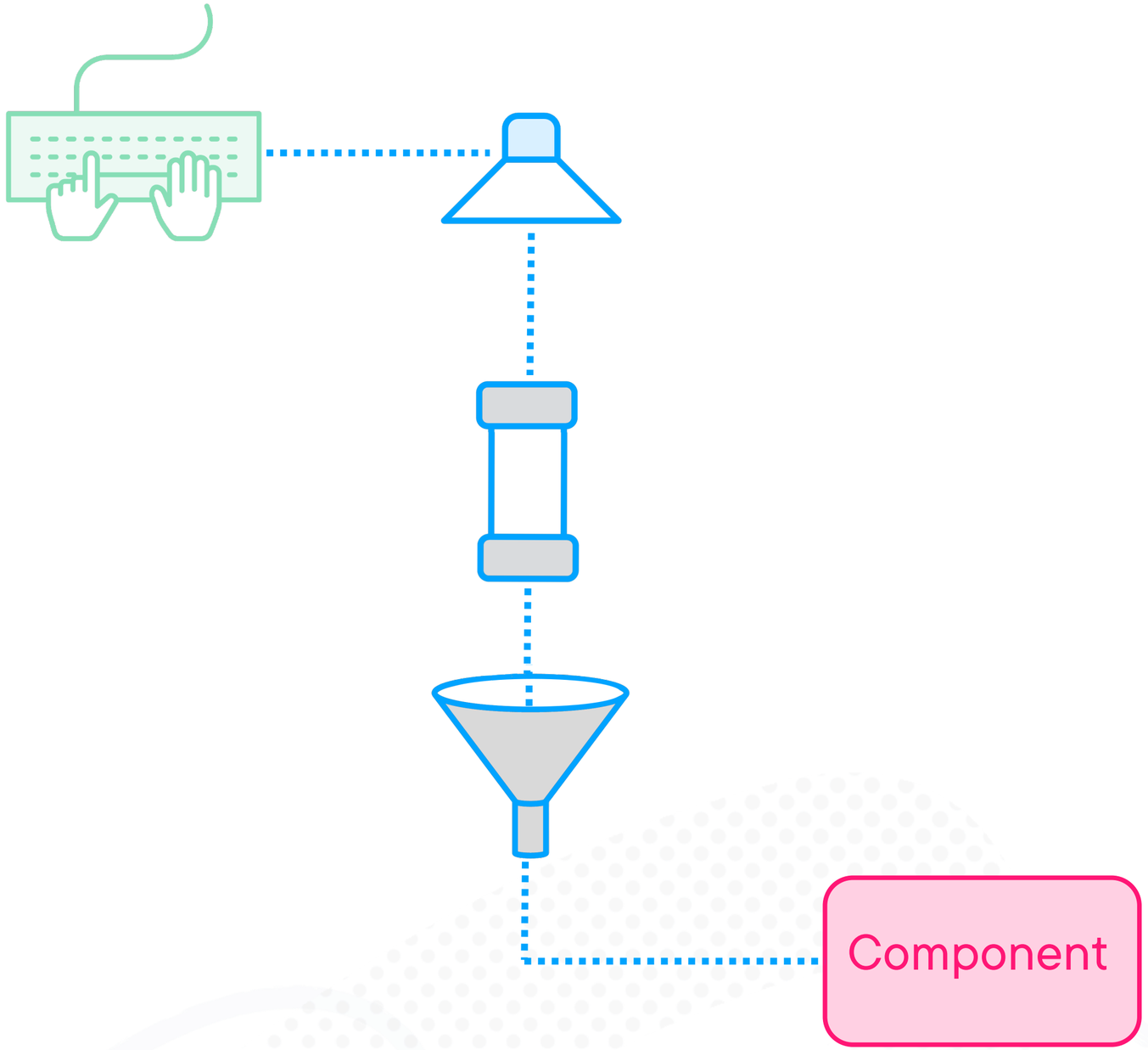
Used for

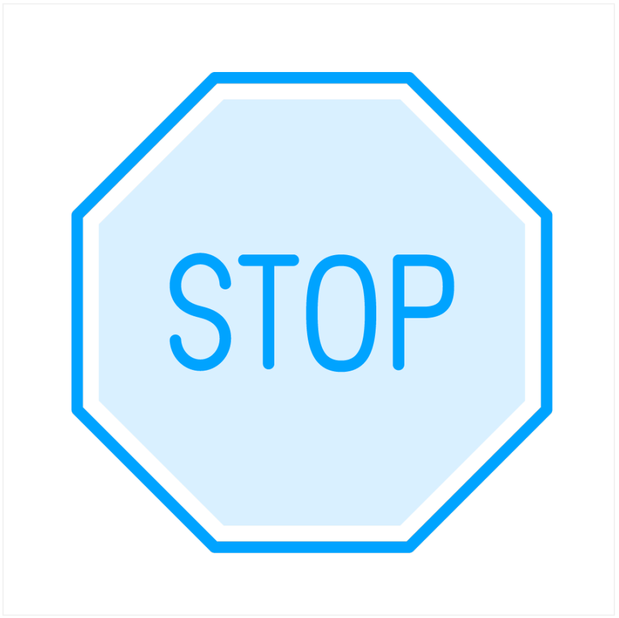
- Debugging
- Performing actions outside of the flow of data (side effects)

Subscribe to start receiving notifications

Unsubscribe to stop receiving notifications

Prevents memory leaks





Unsubscribing

`unsubscribe()` **method**

`takeUntilDestroyed()` **operator**

GitHub

https://github.com/DeborahK/angular-rxjs-ps-course

Beginning sample application files:

apm-begin

Final (completed) sample application files:

apm-end

File with clickable links to additional information:

MOREINFO.md