Functional Reactive Programming

Mastering Time

MINDERA
software craft

What is FRP?

- Using functional programming constructs to handle streams of values over time.
- A way to handle asynchronicity.
- An elegant solution for infinite streams.

MINDERA
software craft

What is FRP?

- Using functional programming constructs to handle streams of values over time.
- A way to handle asynchronicity.
- An elegant solution for infinite streams.

MINDERA
software craft

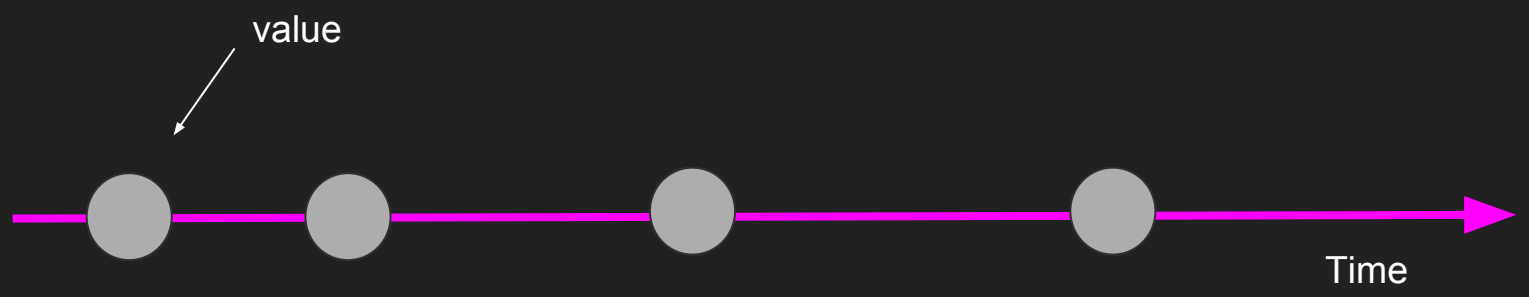
What is FRP?

- Using functional p ams of values over time.
- A way to handle a
- An elegant solutio

Observable.timer(

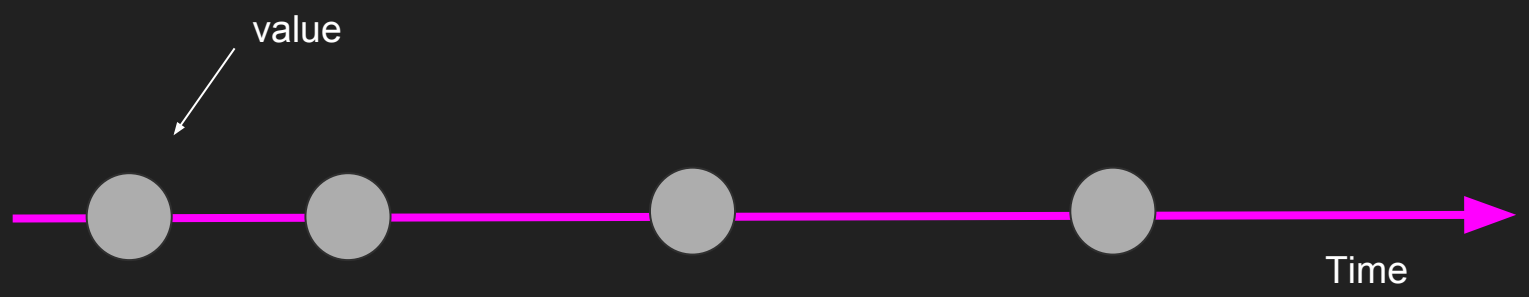
MINDERA
software craft

What’s a stream?



MINDERA
software craft

What’s a stream?



MINDERA
software craft

Observable

- A way to handle stream of values that occur over time.
- Push-based
- Lazy
- Concurrency model agnostic (synchronous, asynchronous)
- A combination of **Observer** and **Iterator** patterns and functional programming

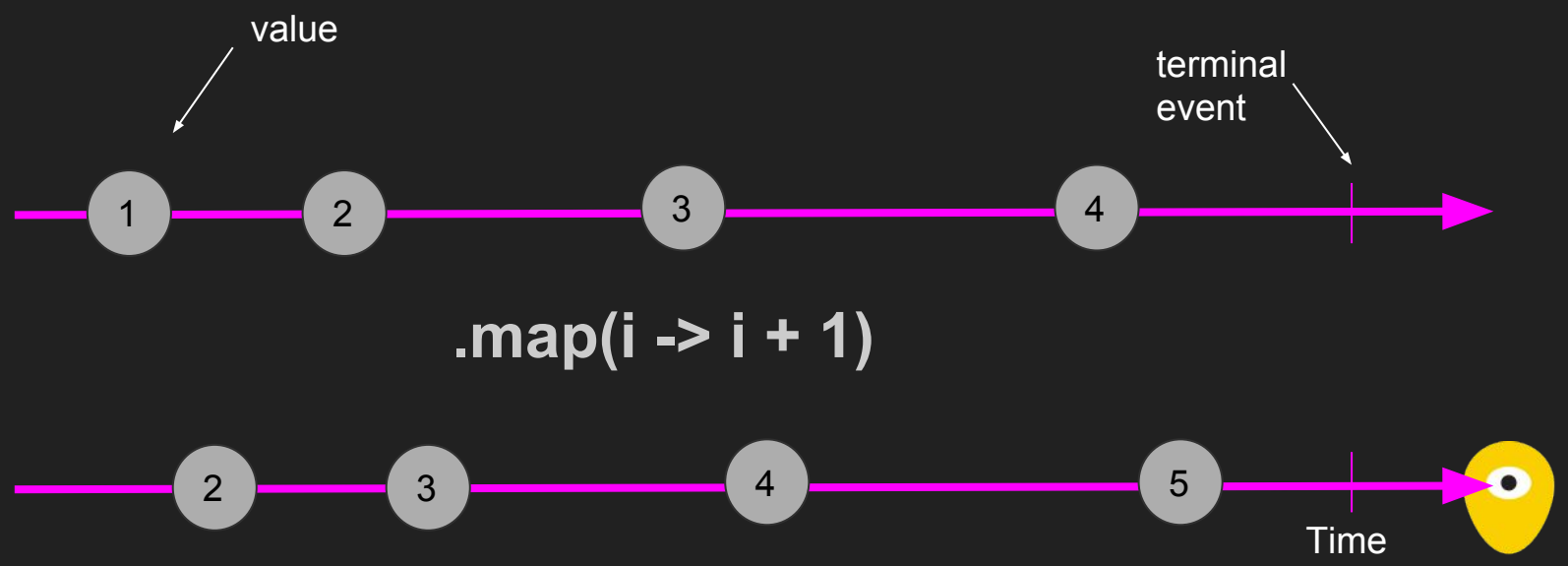
MINDERA
software craft

Operators and Transformations

- map
- flatMap
- filter
- reduce

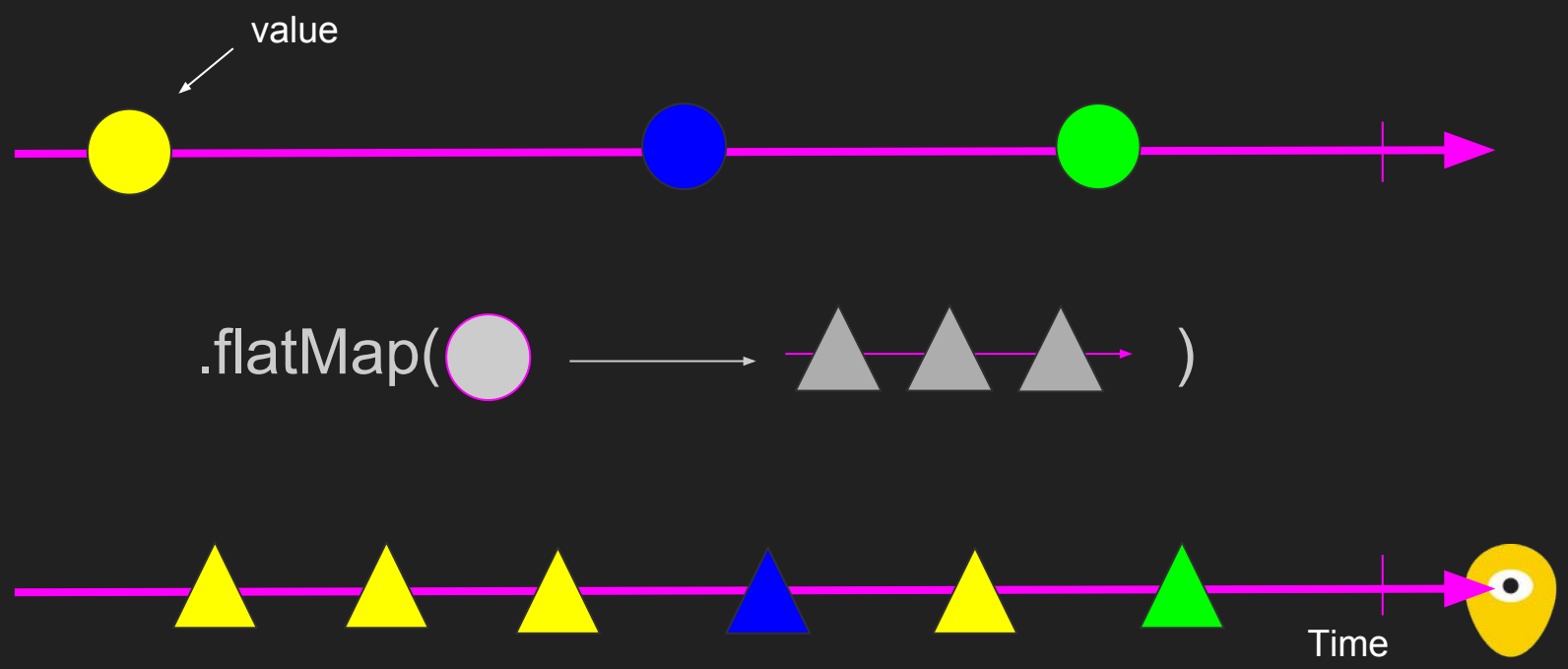
MINDERA
software craft

What’s a map over a stream?



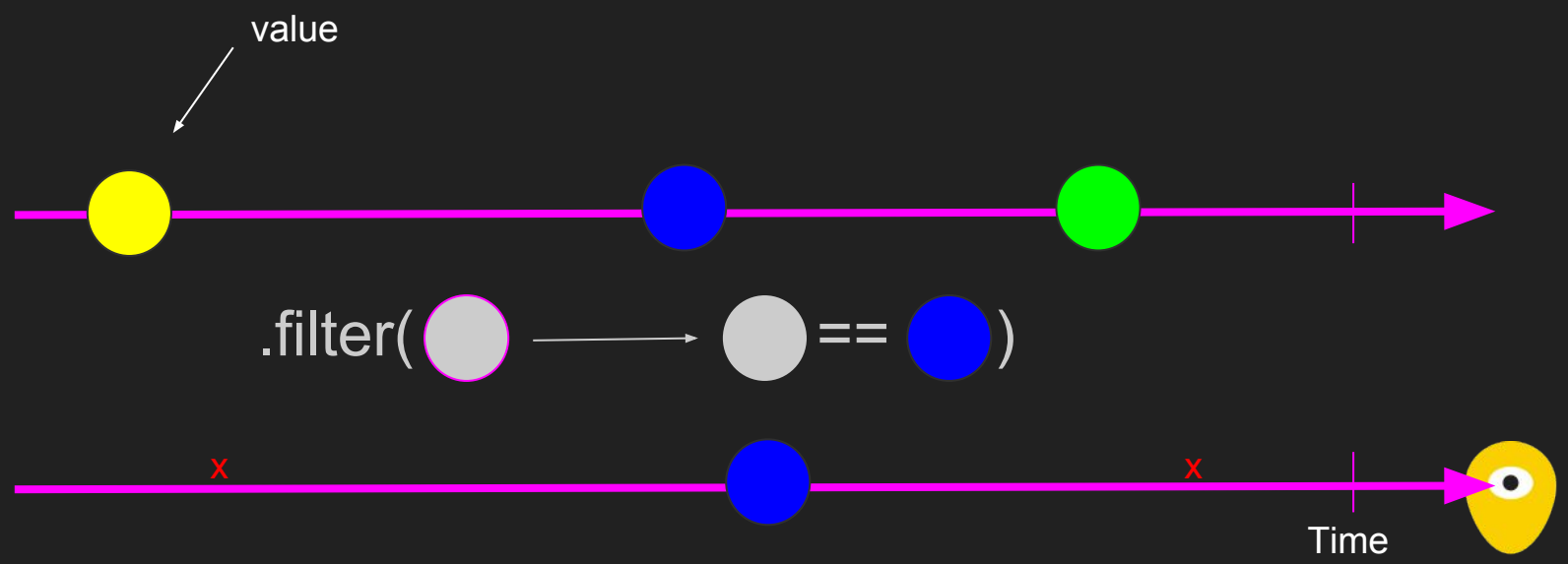
MINDERA
software craft

What’s a flatMap over a stream?



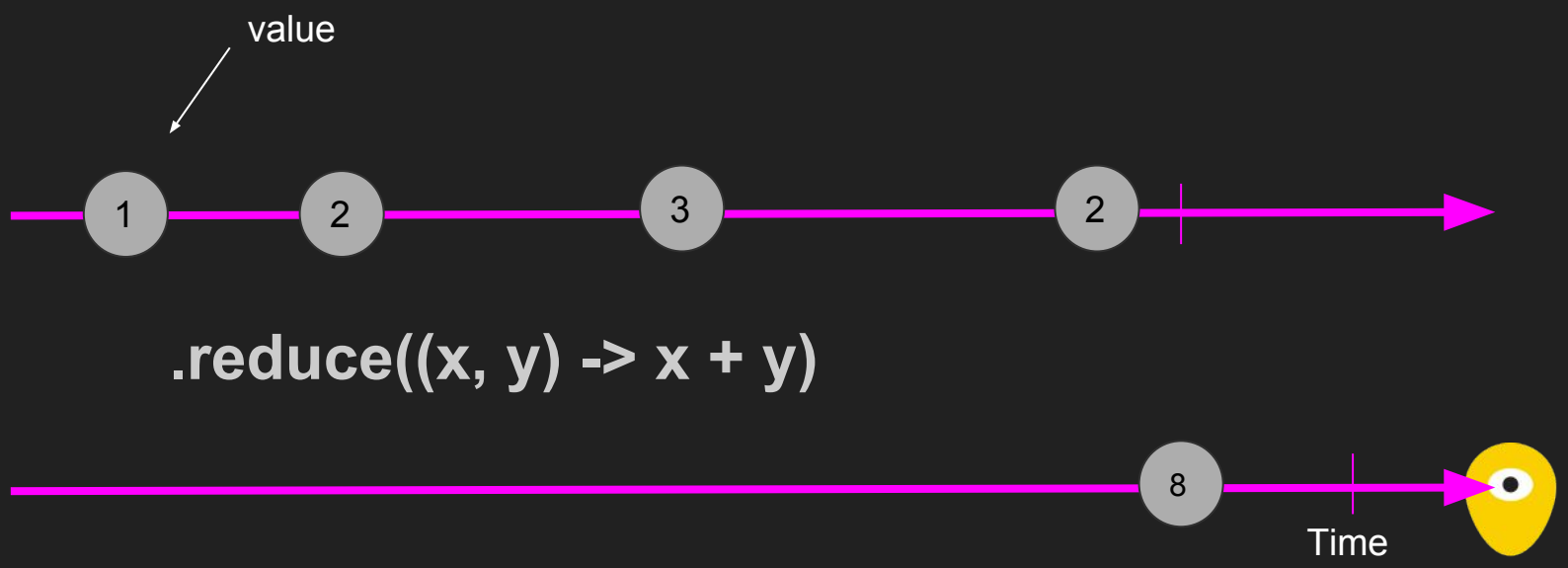
MINDERA
software craft

What’s a filter over a stream?



MINDERA
software craft

What’s a reduce over a stream?



MINDERA
software craft

Observable

Being lazy this does nothing:

```
myObservable
    .map(i -> i + 1);
```

Observable

Being lazy this does nothing:

```
myObservable
    .map(i -> i + 1);

myObservable
    .map(this::myMethod);
```

Observable

Being lazy this does nothing:

```
myObservable
    .map(i -> i + 1);

myObservable
    .map(this::myMethod);
```

You have to subscribe

MINDERA
software craft

Observer

Connects to Observable via subscription

Three types of events pushed by the Observable:

- onNext
- onError
- onCompleted

What’s a stream?

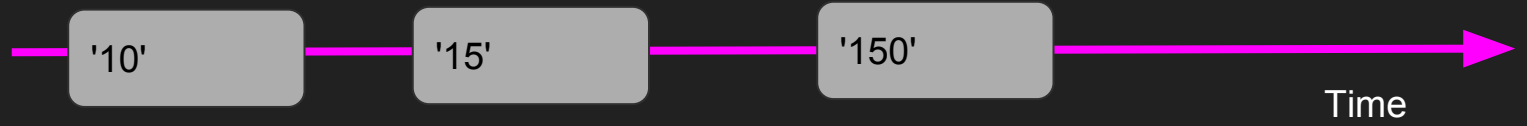
```
Rx.Observable.fromEvent(document.querySelector('input'), 'input')
  .map(ev => ev.target.value)
  .filter(n => n > 10)
  .map(n => n * 2)
  .subscribe(n => console.log(n)); // e.g. - 30, 300, ...
```

MINDERA

software craft

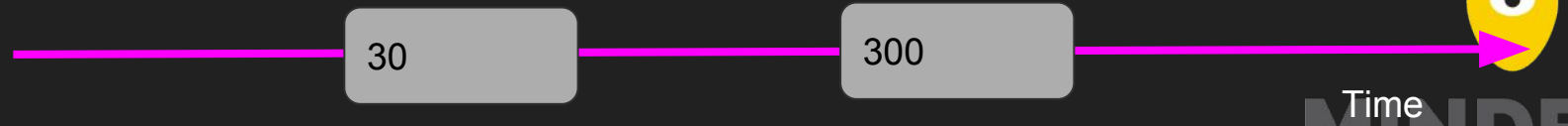
```
Rx.Observable.fromEvent(document.querySelector('input'), 'input')
```


```
.map(ev => ev.target.value)
```




```
.filter(n => n > 10)
```


```
.map(n => n * 2)
```




```
.subscribe(n => console.log(n));
```

Demo

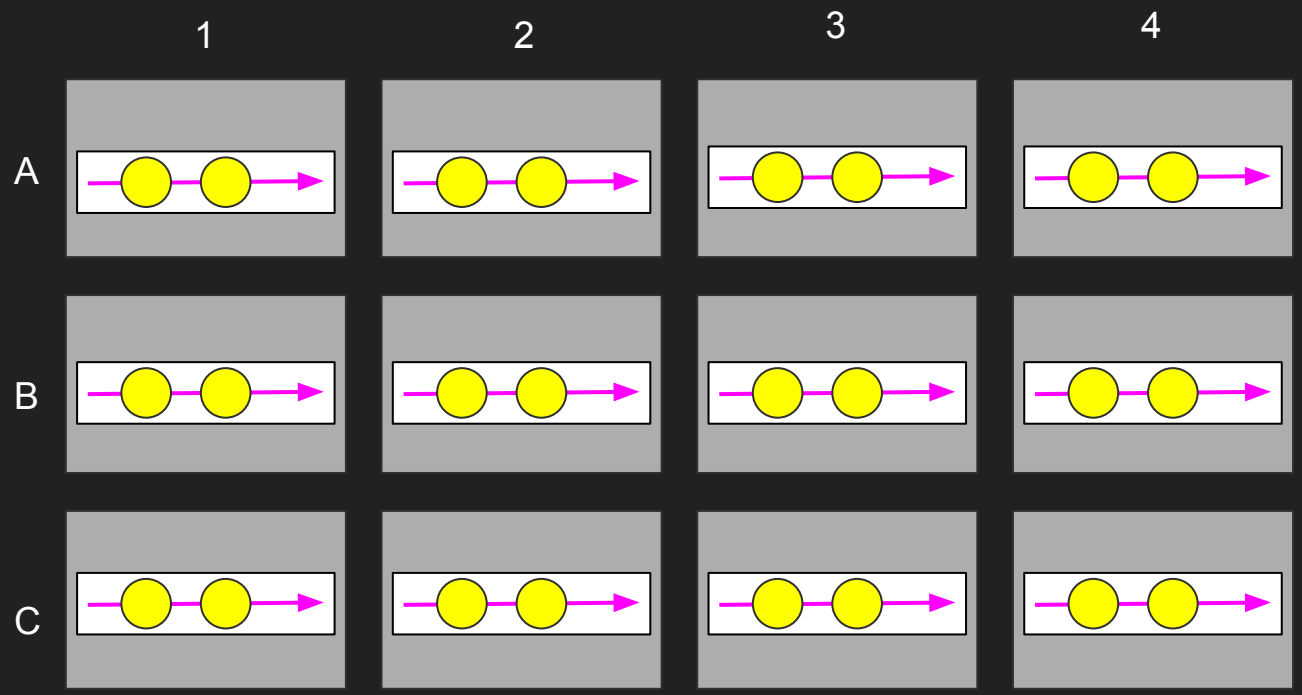
An excel sheet built with JS and RxJS, powered by an computation engine built on Java and RxJava.

MINDERA

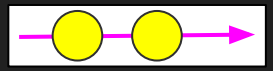
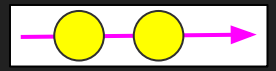
software craft

	1	2	3	4
A	1	2		
B		A1+A2		
C			B2 * 2	

MINDERA
software craft



MINDERA
software craft

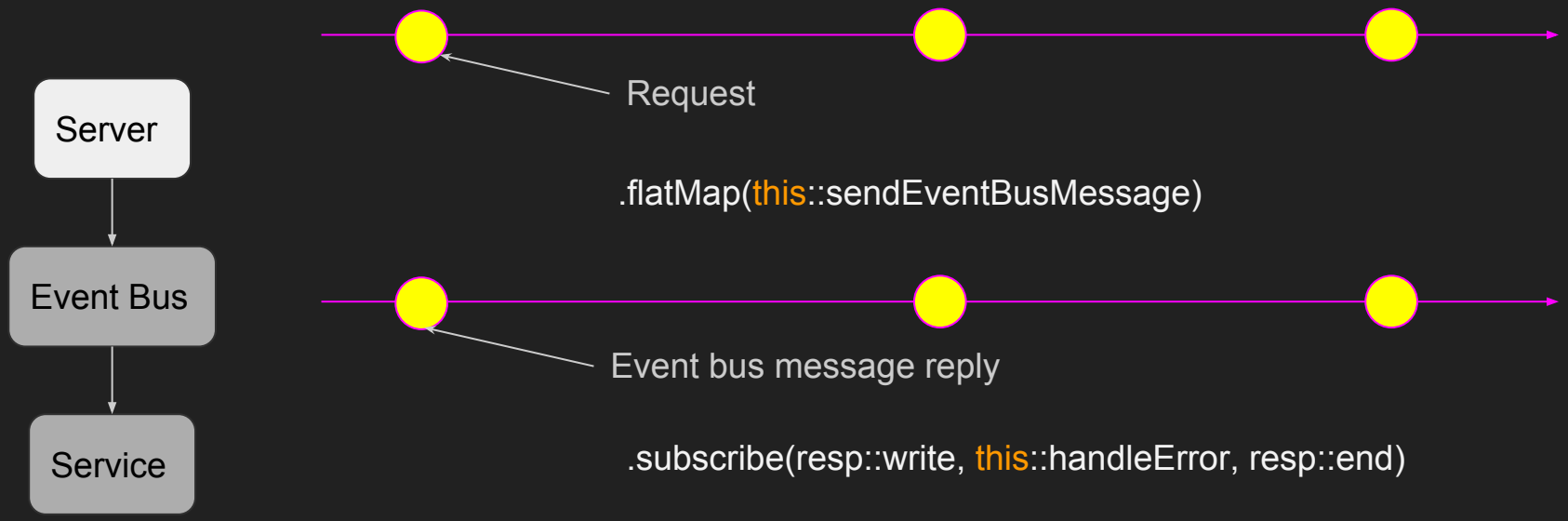
allChanges = merge( ,  , …)

allChanges = 

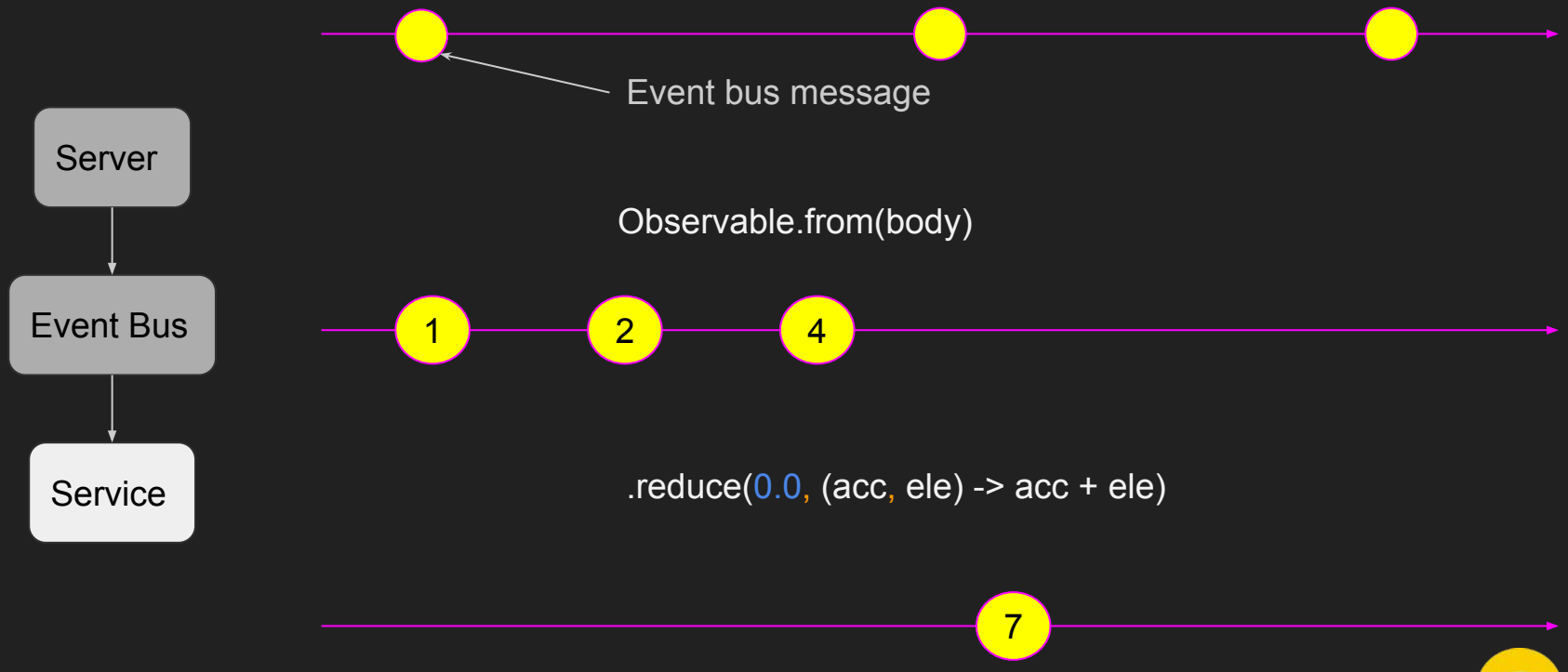
```
allChanges
        .filter(change => !change.computed)
        .flatMap(change => compute(change, allChanges)) //or mergeMap
        .subscribe(computed => {
                updateCell(computed.cellId, computed.value);
        });
```

MINDERA

software craft



MINDERA
software craft



Q&A

MINDERA

software craft

Thank you!

Mehul Irá
mehul.ira@mindera.com

Gabriel Pinto
gabriel.pinto@mindera.com

MINDERA
software craft