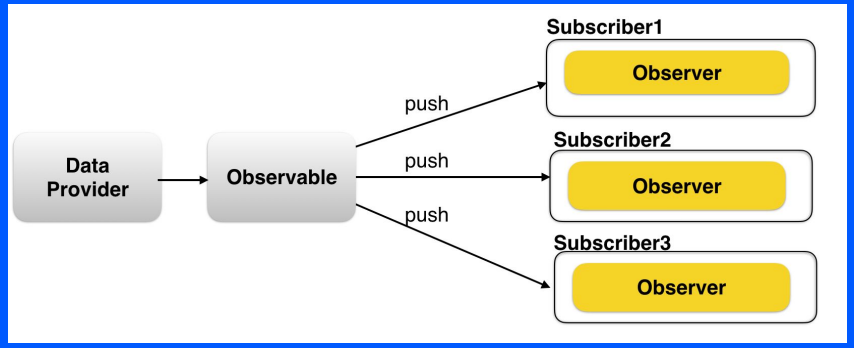
Jan Strapek
Front-end developer

RxJS operators and observable chain workshop

Recommended sites to use

- ➔ https://rxjs-dev.firebaseapp.com/
  - ◆ Official documents with latest features.
- ➔ https://www.learnrxjs.io/
  - ◆ From my point of view the best description of operators.
- ➔ https://rxmarbles.com/
  - ◆ If you want visual description of operator.

Observable



- Wrapper for data that can be subscribed to.
- Subscriber will be notified every time data changes.

Cold
(lazy)

```
const cold = new Observable( subscribe: o => o.next(Math.random()));

cold.subscribe(console.log);
cold.subscribe(console.log);
```

Result:

- ➔ 0.7171676764784061
- ➔ 0.4260696466189559 7

Hot
(multi)

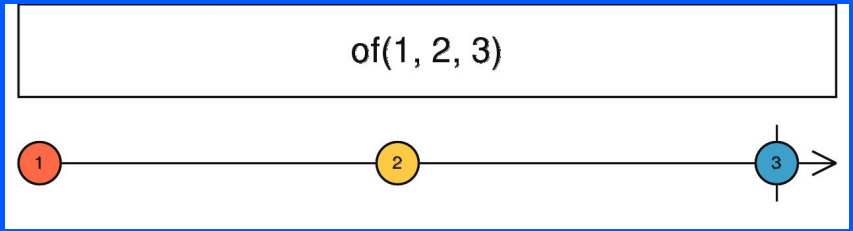
```
const hot = new Observable( subscribe: o => o.next(Math.random()))
  .pipe(shareReplay());

hot.subscribe(console.log);
hot.subscribe(console.log);
```

Result:

- ➔ 0.8267920466147021
- ➔ 0.8267920466147021

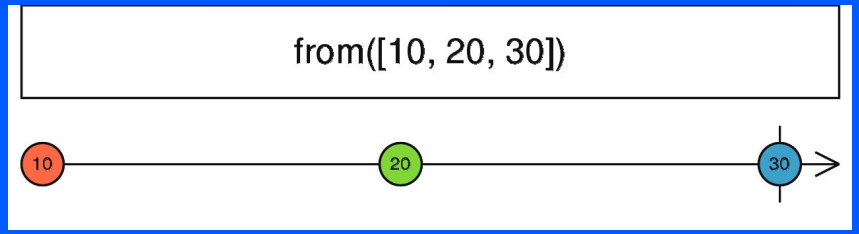
Creation Operators

of



➔ Converts the arguments to an observable sequence.

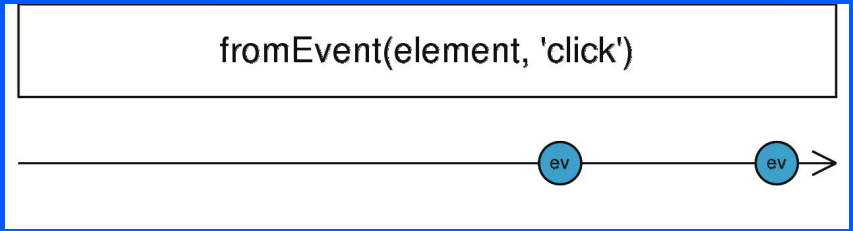
from



➔ Create Observable from Array, a Promise, an iterable object, an array-like object, or an Observable-like object.

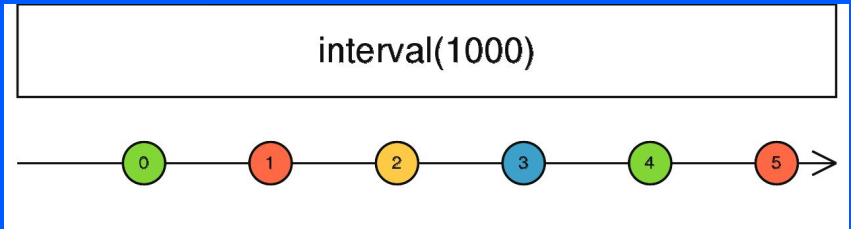
➔ Unlike ***of,*** it does flattering and emits each argument.

fromEvent



- Creates an Observable that emits events of specific type.
- Supported types: DOM (EventTarget, NodeList, HtmlCollection), Node.js EventEmitter,

interval



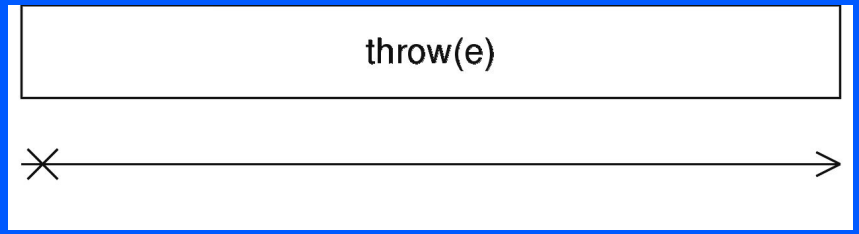
➔ Creates on Observable that emits sequential numbers every specified interval of time.

iif

```
let subscribeToFirst = true;
const firstOrSecond = iif(
  condition: () => subscribeToFirst,
  of( a: 'first'),
  of( a: 'second')
);
firstOrSecond.subscribe( next: value => console.log(value));
// Logs:
// 'first'

subscribeToFirst = false;
firstOrSecond.subscribe( next: value => console.log(value));
// Logs:
// 'second'
```

➔ Decides at subscription time which Observable will actually be subscribed.

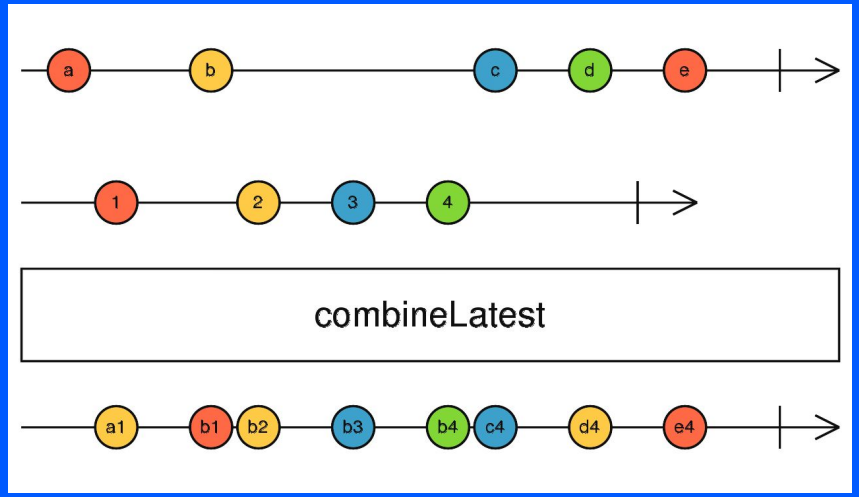
throwError



➔ Creates on Observable that emits no items to the Observer and immediately emits an error notification.

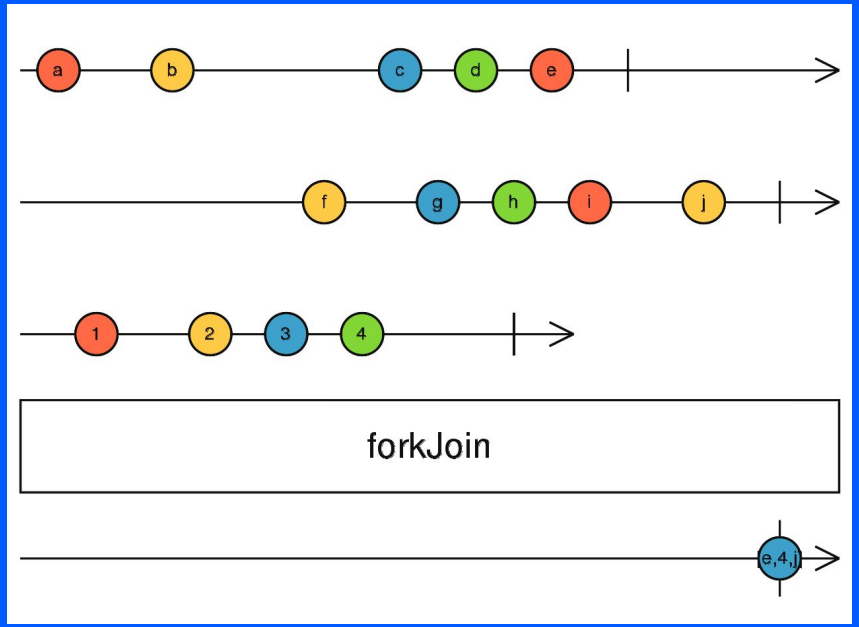
Join Creation Operators

combineLatest



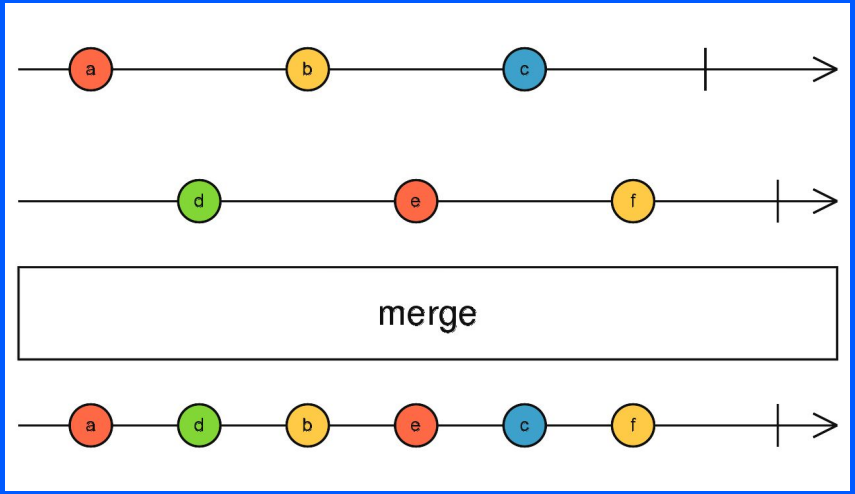
➔ Combines multiple Observables to create single Observable whose values are calculated from latest values of each of its input Observables.

forkJoin



- When all Observables complete, emit the last emitted value from each.
- It will emit values only once.

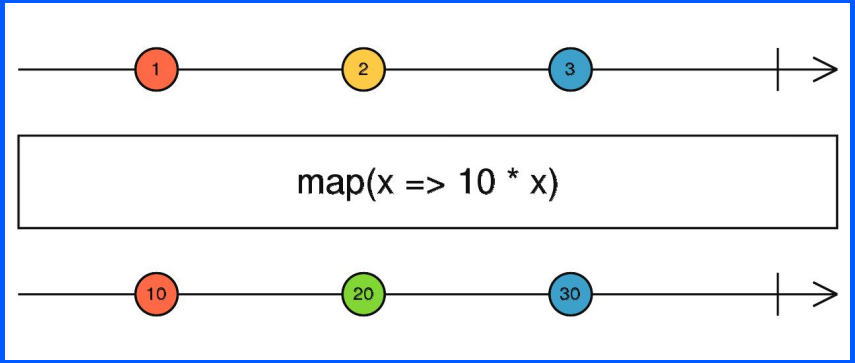
merge



➔ Creates an output Observable which concurrently emits all values from every given input.

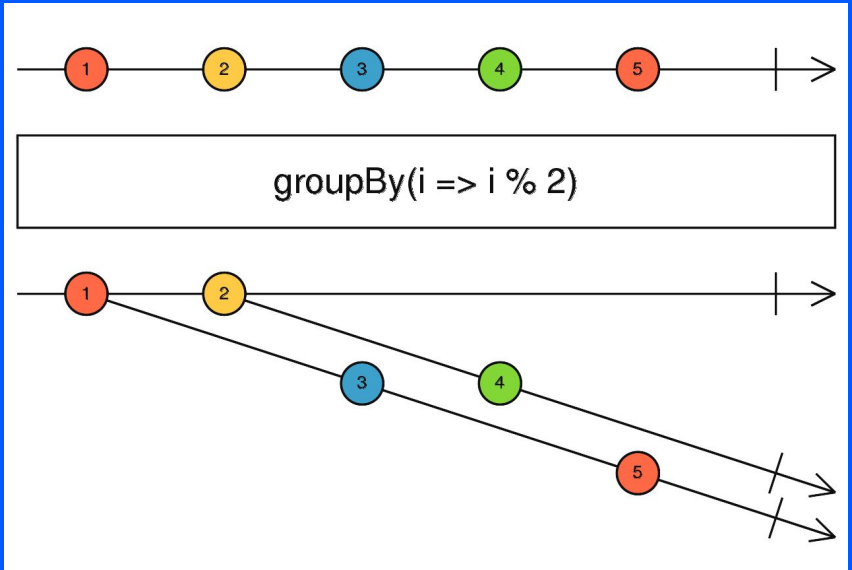
Transformation Operators

map



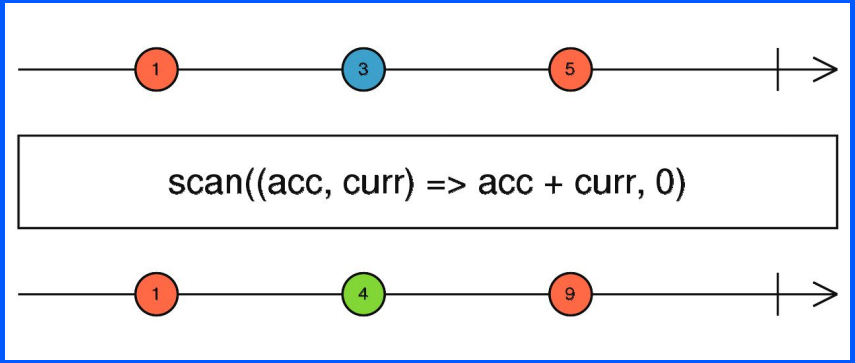
➔ Applies a given project function to each value emitted.

groupBy



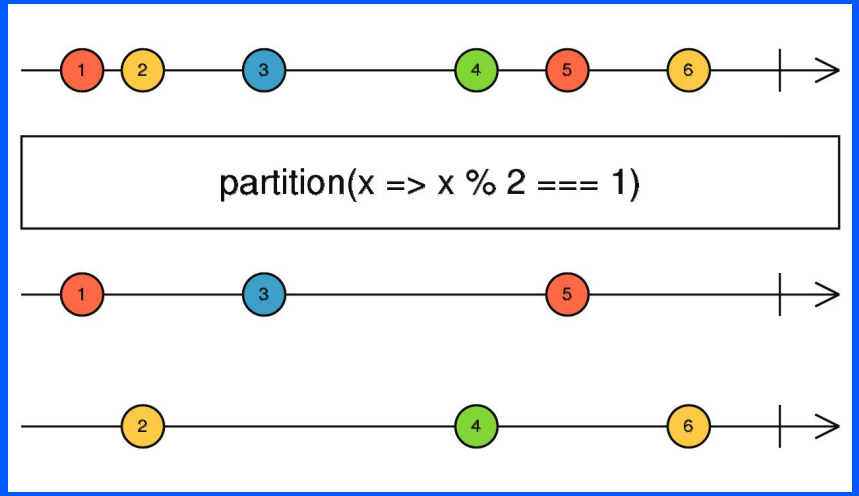
➔ Groups the items emitted by an Observable according to a specified criterion.

scan



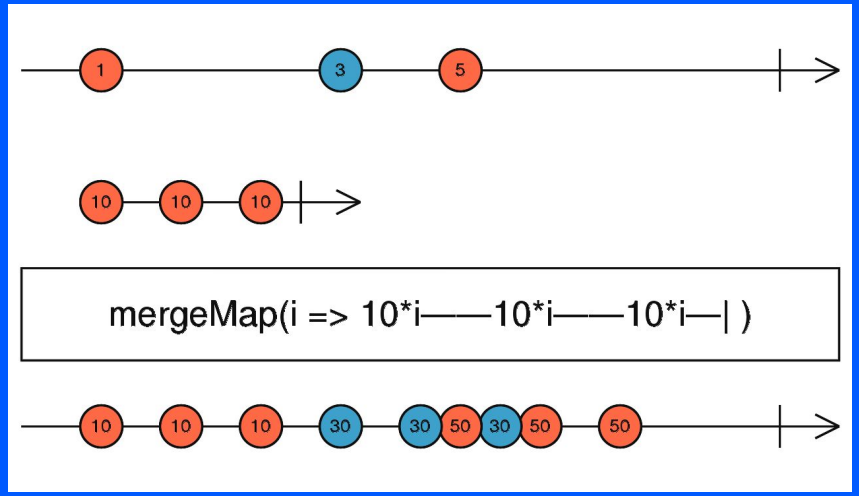
- Applies an accumulator function over the source, and returns each intermediate result.
- It's like ***reduce***, but emits the current accumulation.

partition



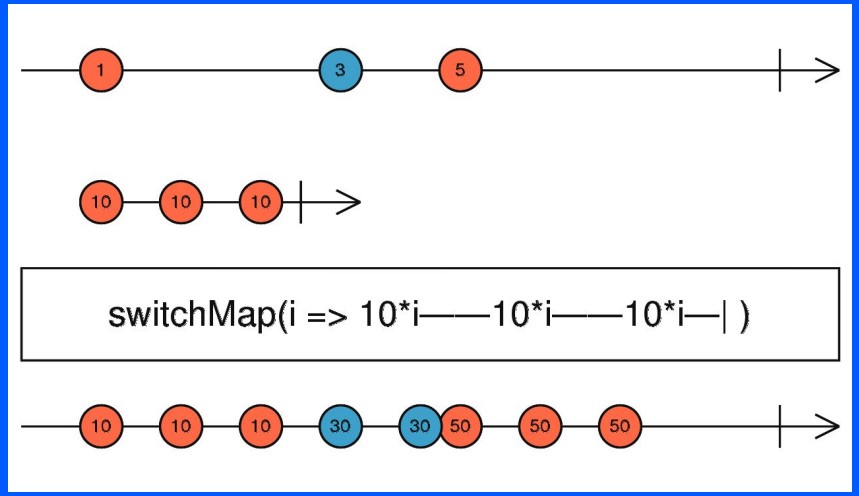
- Splits the source into two, one with values that satisfy a predicate, and another with values that don't.
- It's like ***filter***, but returns two Observables.

mergeMap



- Projects each source value to an Observable which is merged in the output Observable.
- It maintains all inner subscriptions.

switchMap



➔ Projects each source value to an Observable which is merged, emitting values only from the most recently projected Observable.

Exercise 1

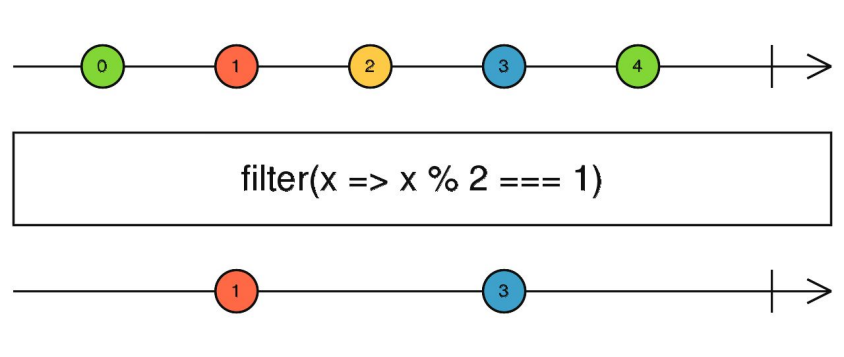
Work-space
https://stackblitz.com/edit/rxjs-type-ahead-pntuqt

Working program
https://rxjs-type-ahead-result-t7p3iy.stackblitz.io

Result
https://stackblitz.com/edit/rxjs-type-ahead-result-t7p3iy

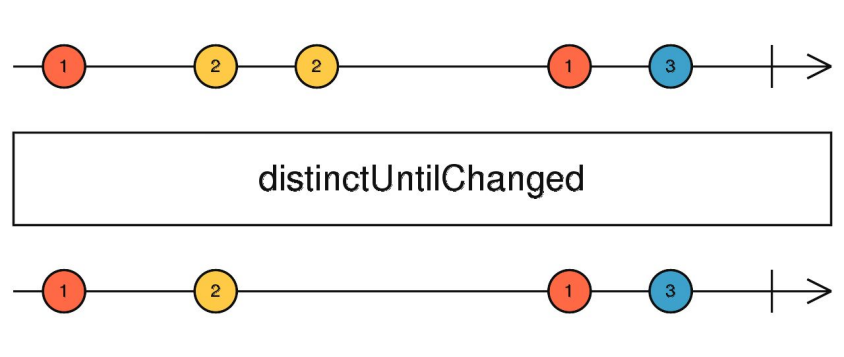
Filtering Operators

filter



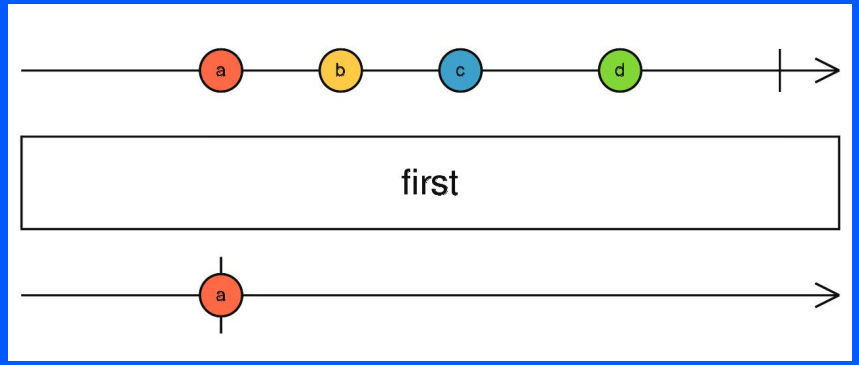
➔ Filter items emitted by the source by only emitting those that satisfy a specified predicate.

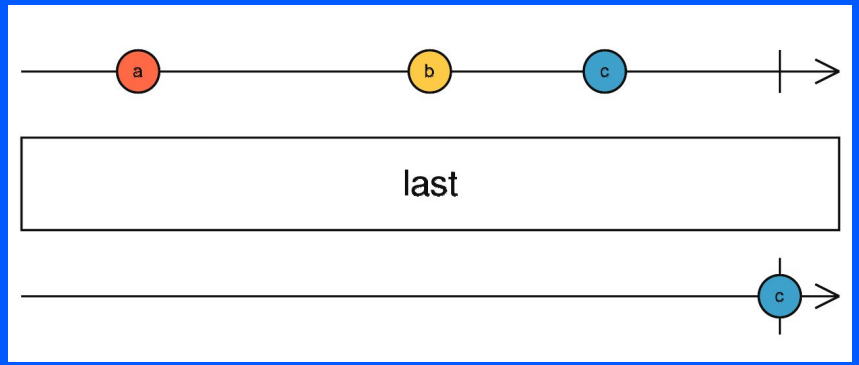
distinctUntilChanged



➔ Only emit when the current value is different that the last.

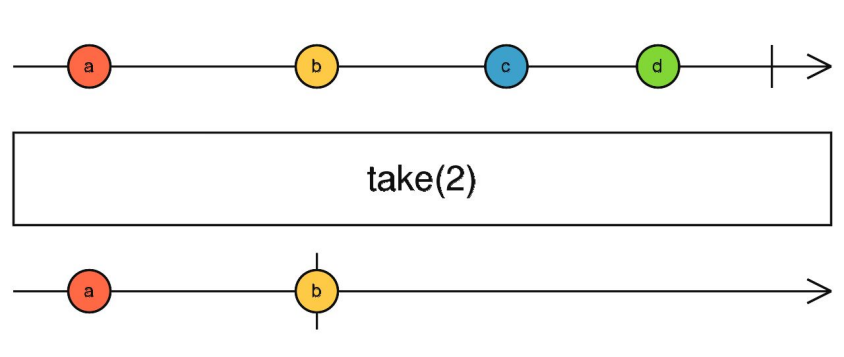
first/last





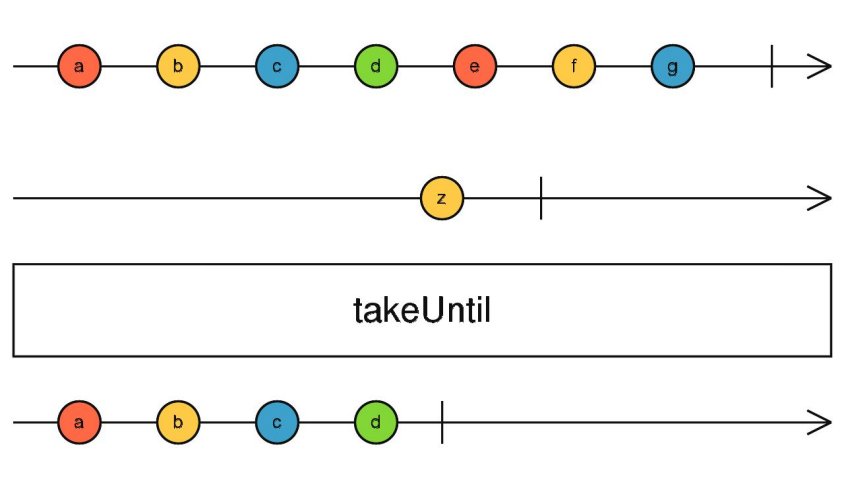
- Emits only the *first/last* value emitted by source Observable.
- It optionaly takes predicate function, in which case it will emit *first/last* value that satisfy the predicate.

take



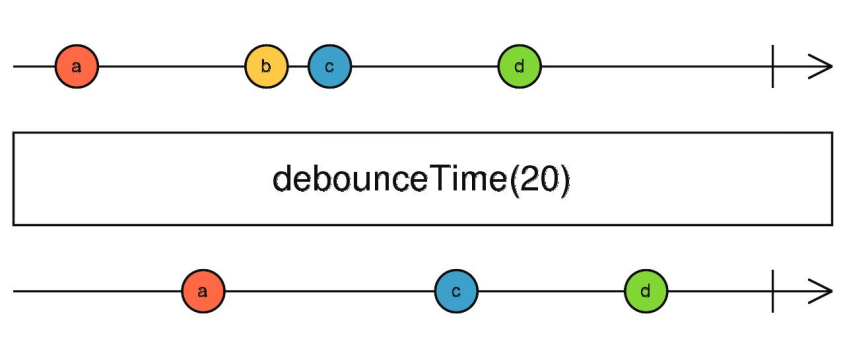
➔ Emit provided number of values before completing

takeUntil



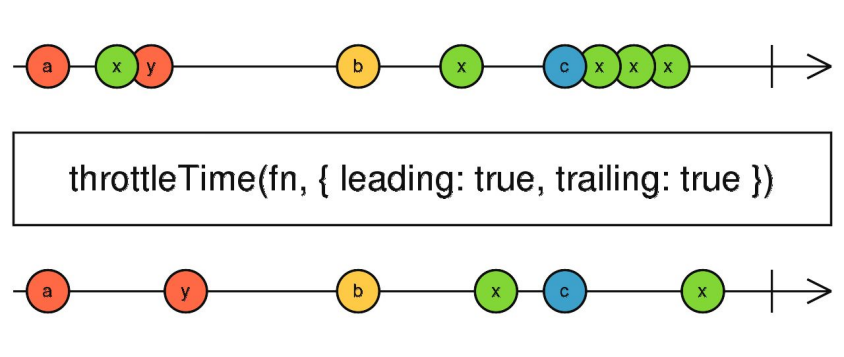
➔ Emit values until provided Observable emits.

debounceTime



➔ Discard emitted values that take less than the specified time between output.

throttleTime



➔ Emit value, then ignores subsequent values for specified duration.

Exercise 2

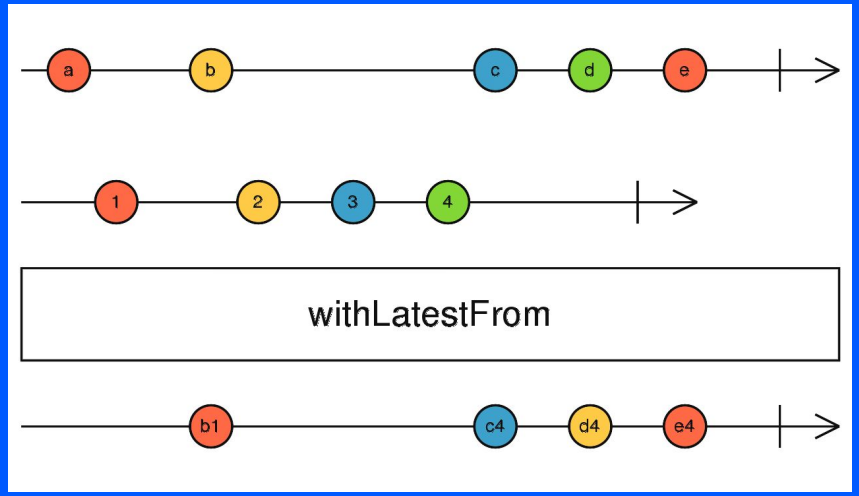
Work-space
https://stackblitz.com/edit/url-change-7wdzq7

Working program
https://url-change-result-asmx4k.stackblitz.io

Result
https://stackblitz.com/edit/url-change-result-asmx4k

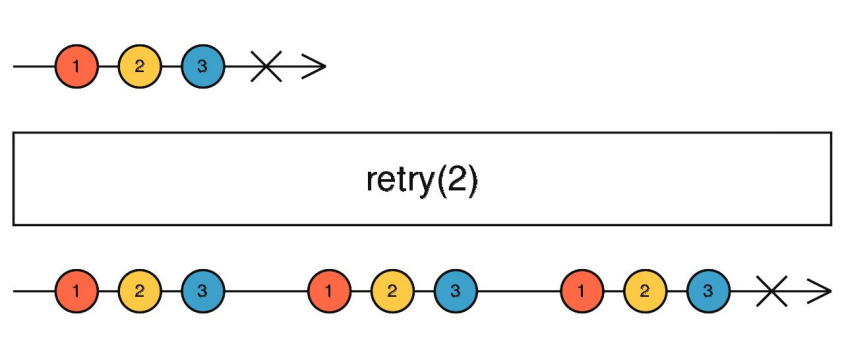
Other Operators

withLatestFrom



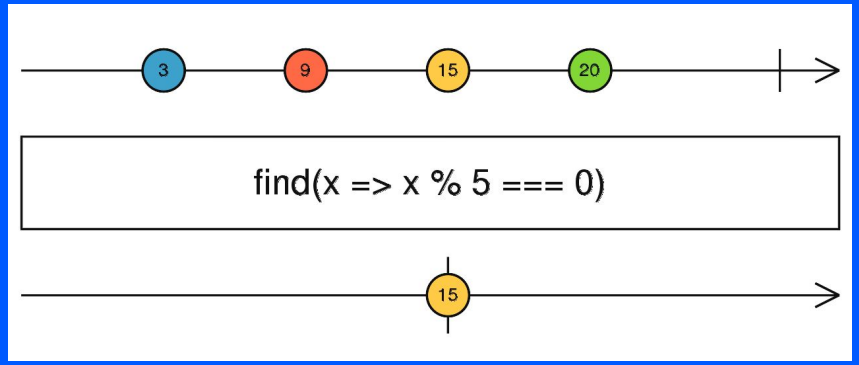
➔ Combines the source Observable with other Observable to create an Observable whose values are calculated from the latest values of each, only when the source emits.

retry



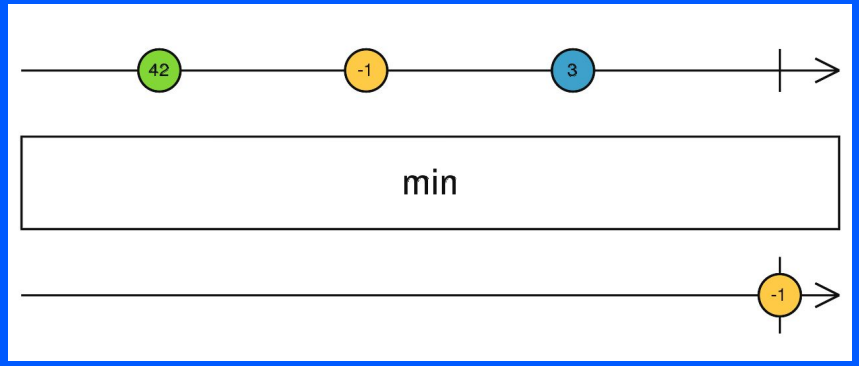
➔ Retry an Observable sequence a specific number of times should an error occur.

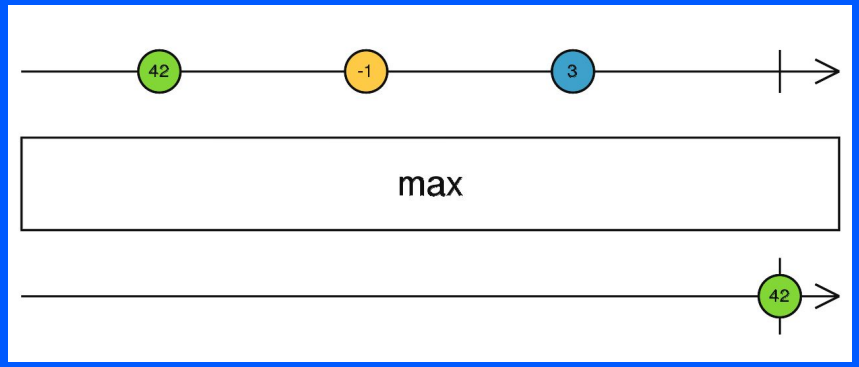
find



➔ Emits only the first value emitted by the source that meets the condition.

min/max





➔ The *min/max* operator operates on an Observable that emits **numbers,** and when source Observable completes it emits a single *min/max* value.

every

- If all values pass predicate before completion emit true, else false.

tap

- Perform a side effect for every emission on the source Observable, but return an Observable that is identical to the source.

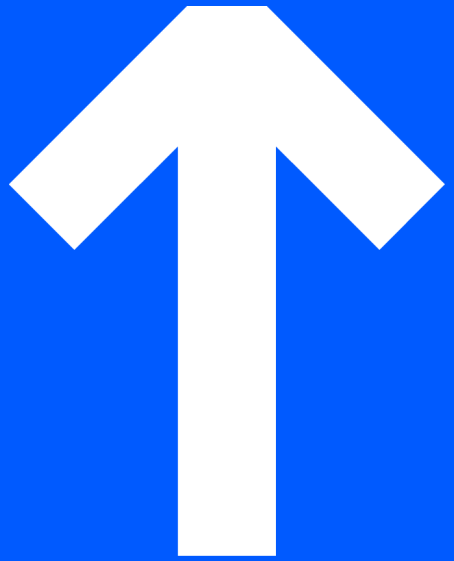
Exercise 3

Work-space
https://stackblitz.com/edit/rxjs-counter-8lhamr

Working program
https://rxjs-counter-result-uf7krb.stackblitz.io

Result
https://stackblitz.com/edit/rxjs-counter-result-uf7krb

up



Q&A

flow