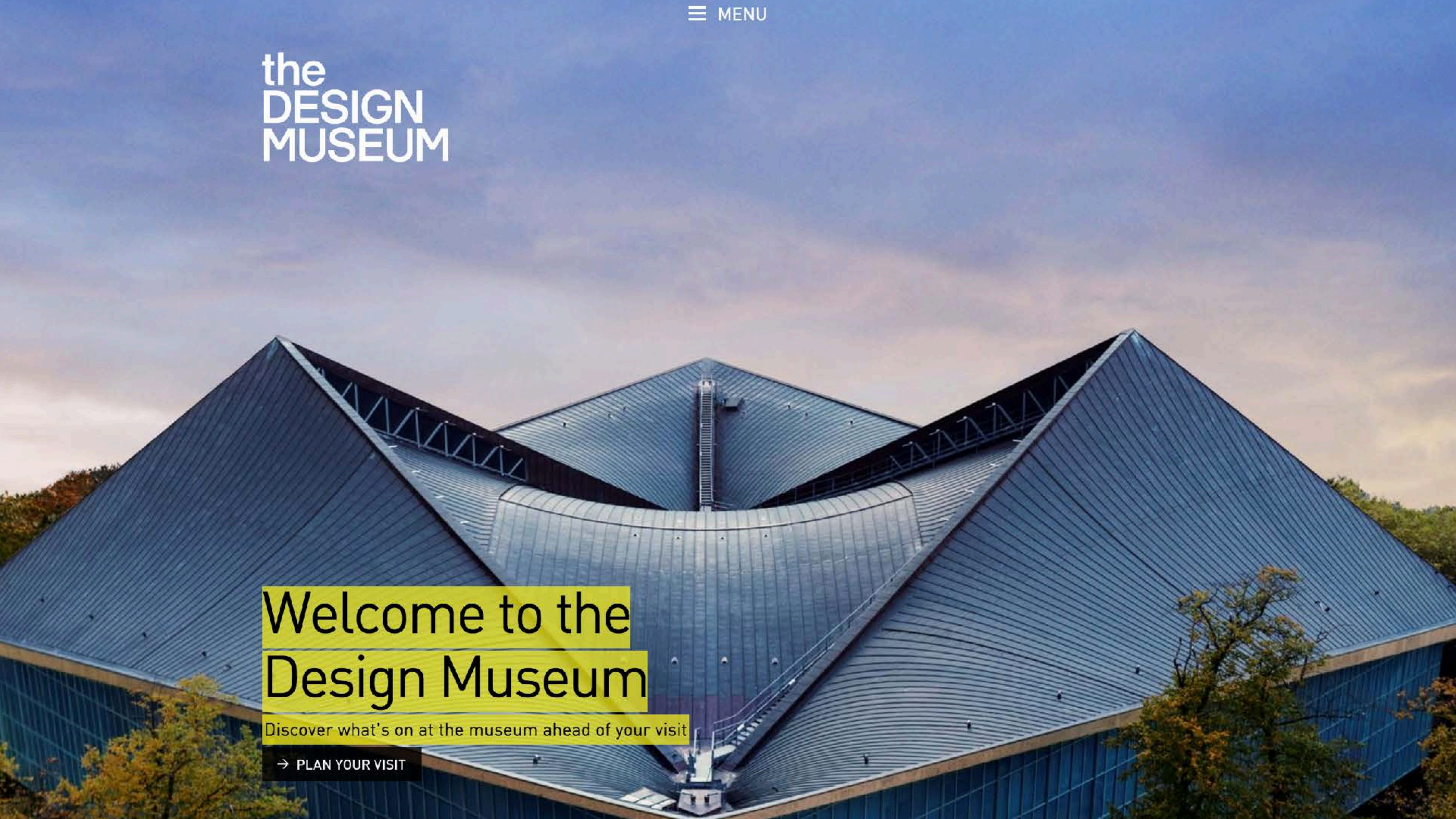
International
JavaScript
Conference

An in-depth look at debugging RxJS code

Mark van Straten
@markvanstraten
q42.com


Q
42


MENU
the
DESIGN
MUSEUM
Welcome to the
Design Museum
Discover what's on at the museum ahead of your visit
→ PLAN YOUR VISIT


Q
42
hue
PHILIPS


PHILIPS

Why not use callbacks?

TESTABILITY

Enter RXJS

scheduler not passed to transformation func causing test to run in real time
OOM zip() fast and slow emitting stream
no catch() breaking the stream
infinite retry() logic
switch of() and from()
toPromise() on never completing stream

BUGS EVERYWHERE

no defer() around promise, retr ails
OOM due to concatMap non-co ting Observables
OOM due to memory leak in Rx libra
switchMap().catch() returns Observable.empty(), stream completes
Promise executes without subscription (Eager)
Duplicate streams due to not using share()
Observable not completing due to wrongly coded completion logic
Error handling breaking main observable flow

subscribed twice to a cold stream
made a filter statement with 5000 subscribers
timer() arguments - start direct, once, repeat always?
mergeMap without concurrency arg
error crashing stream, forgot to add subscribe() logging
forgot to subscribe()
created an exception inside subscribe() logic
subscribe() statement inside other stream

WHERE DO THE BUGS HIDE?

VALUES
LIFECYCLE
TRANSFORMATIONS

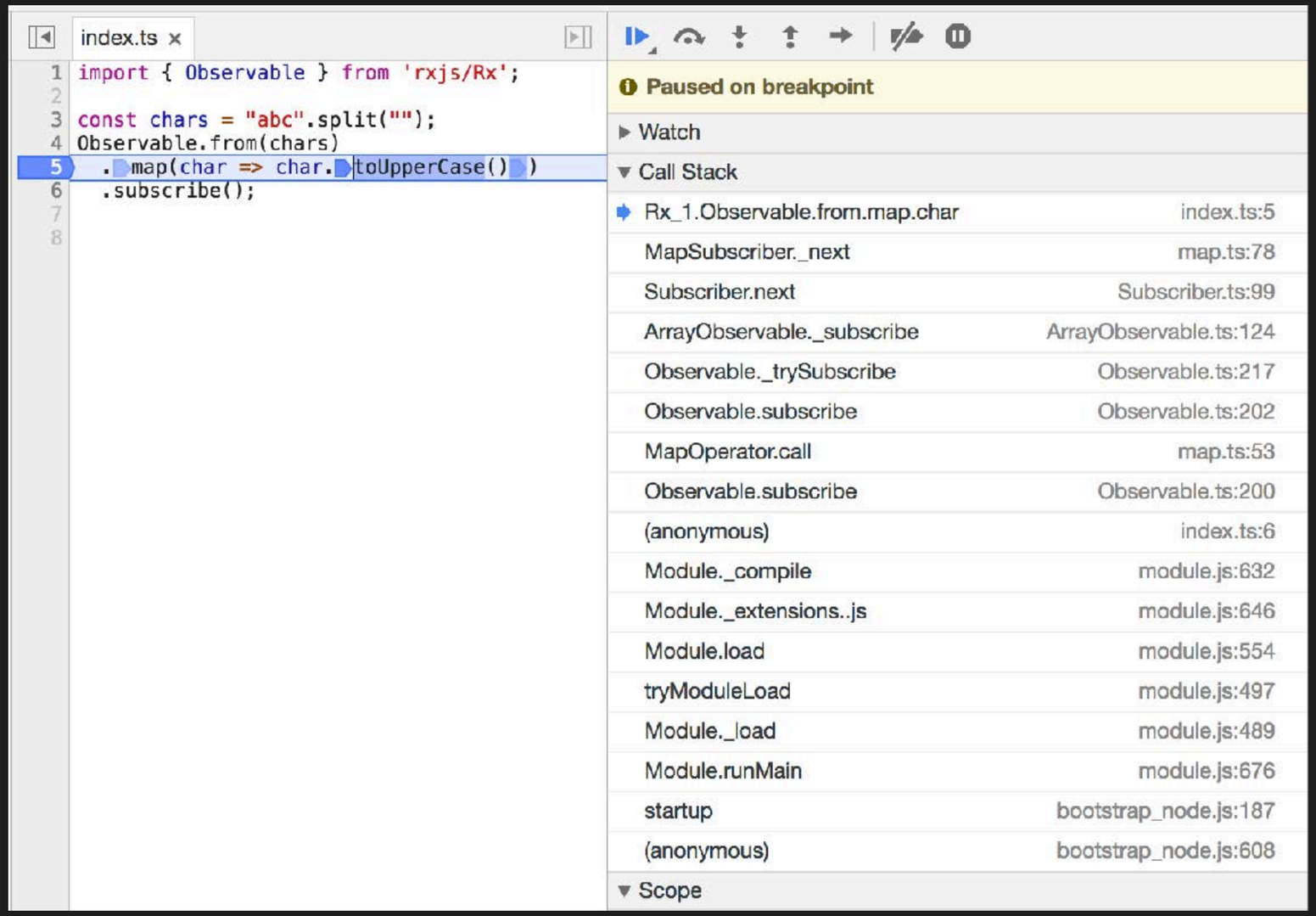
DEBUGGING VALUES

OUTPUT LOGGING

```
const chars = "abc".split("");
Observable.of(chars)
   .do(val ⟹ console.log('next: ' + val))
   .subscribe();
```

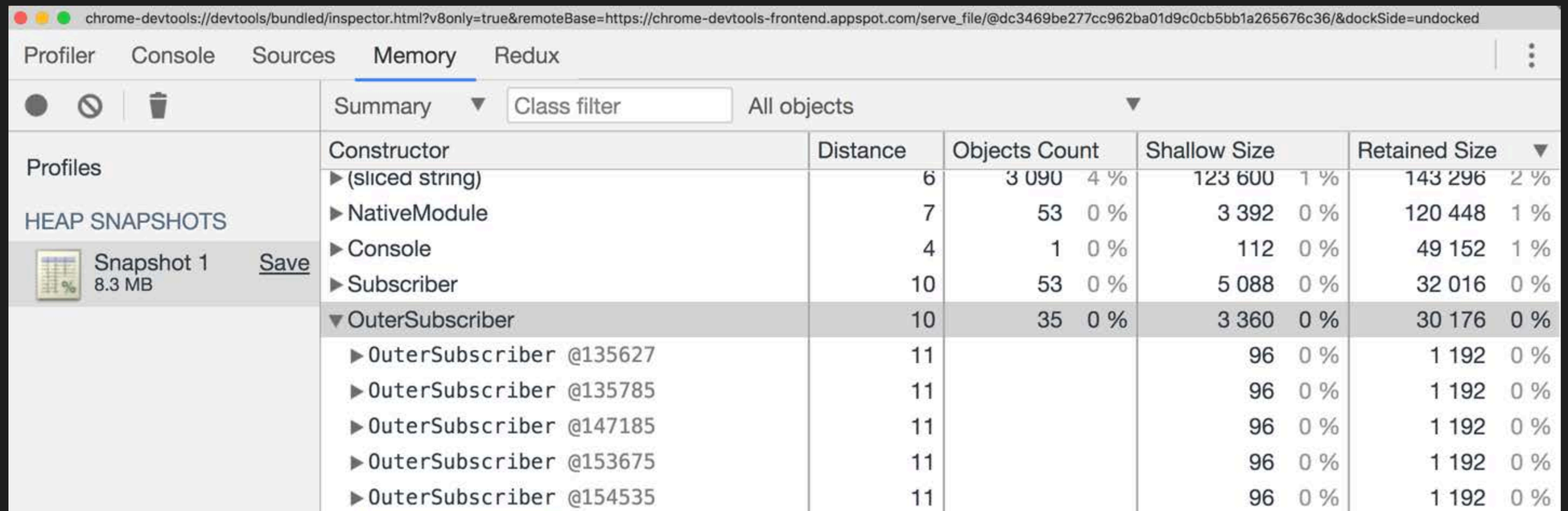
```
next: ["a","b","c"]
```

BREAKPOINTS



MEMORY DUMPS

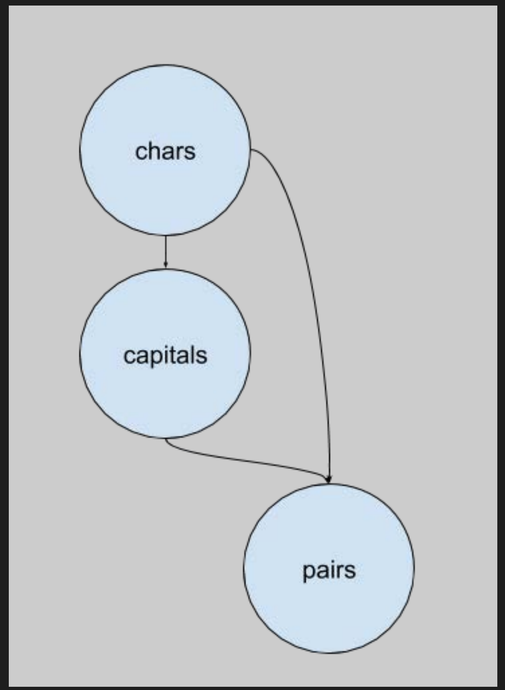
```
const chars = "abcdefghijk".split("");
Observable.from(chars)
  .scan((acc, curr) ⇒ curr + acc, "")
  .subscribe(console.log);
```



DEBUGGING LIFECYCLE

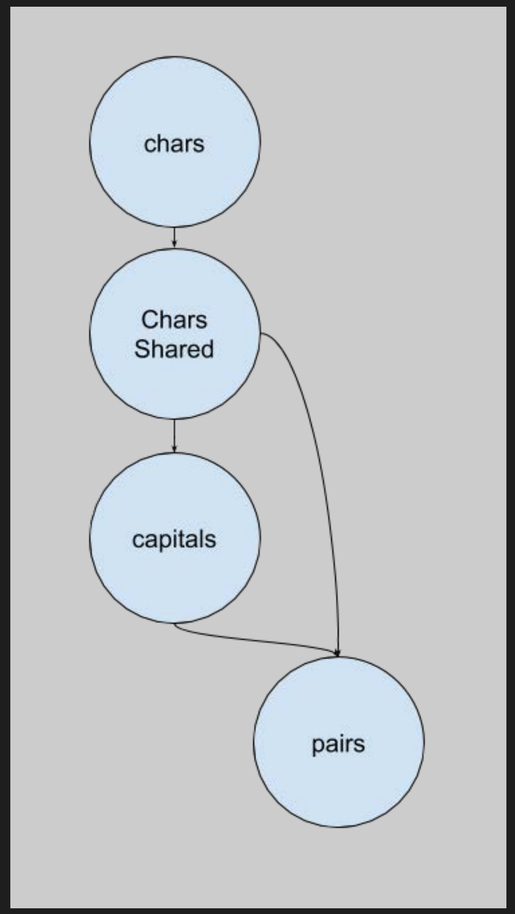
DRAWING DEPENDENCY GRAPHS

```
const chars = Observable
  .from("abc".split(""))
  .debug('chars-source-stream', false);
const capitals = chars
  .map(char ⇒ char.toUpperCase());
const pairs = chars
  .zip(capitals)
  .subscribe(console.log);
```



```
[chars-source-stream] subscribed
[chars-source-stream] completed
[chars-source-stream] unsubscribed
[chars-source-stream] subscribed
[ 'a', 'A' ]
[ 'b', 'B' ]
[ 'c', 'C' ]
[chars-source-stream] completed
[chars-source-stream] unsubscribed
```

```
const chars = Observable
  .from("abc".split(""))
  .debug('chars-source-stream', false);
const charsShared = charsCold
  .share();
```



```
[chars-source-stream] subscribed
[ 'a', 'A' ]
[ 'b', 'B' ]
[ 'c', 'C' ]
[chars-source-stream] completed
[chars-source-stream] unsubscribed
```

LIFECYCLE EVENTS

```
function debug<T> (this: Observable<T>, identifier: string): Observable<T> {
  return Observable.empty<any>()
    .do(_ => {}, undefined, () => console.log(`[${identifier}] subscribed`))
    .concat(
      this.do(
        (val) => console.log(`[${identifier}] nextVal: ${val}`),
        (err) => console.log(`[${identifier}] error: ${err.message}`),
        () => console.log(`[${identifier}] completed`)
      )
    )
    .finally(() => console.log(`[${identifier}] unsubscribed`));
};
```

Stream thrown error logs:

```
[1] subscribed
[1] unsubscribed
```

Stream completed

```
[2] subscribed
[2] completed
[2] unsubscribed
```

Unsubscribe before completion

```
[3] subscribed
[3] unsubscribed
```

DEBUGGING TRANSFORMATIONS

```
Observable.interval(100)
  .concatMap(i ⟹ Observable.of(i).delay(1000))
```

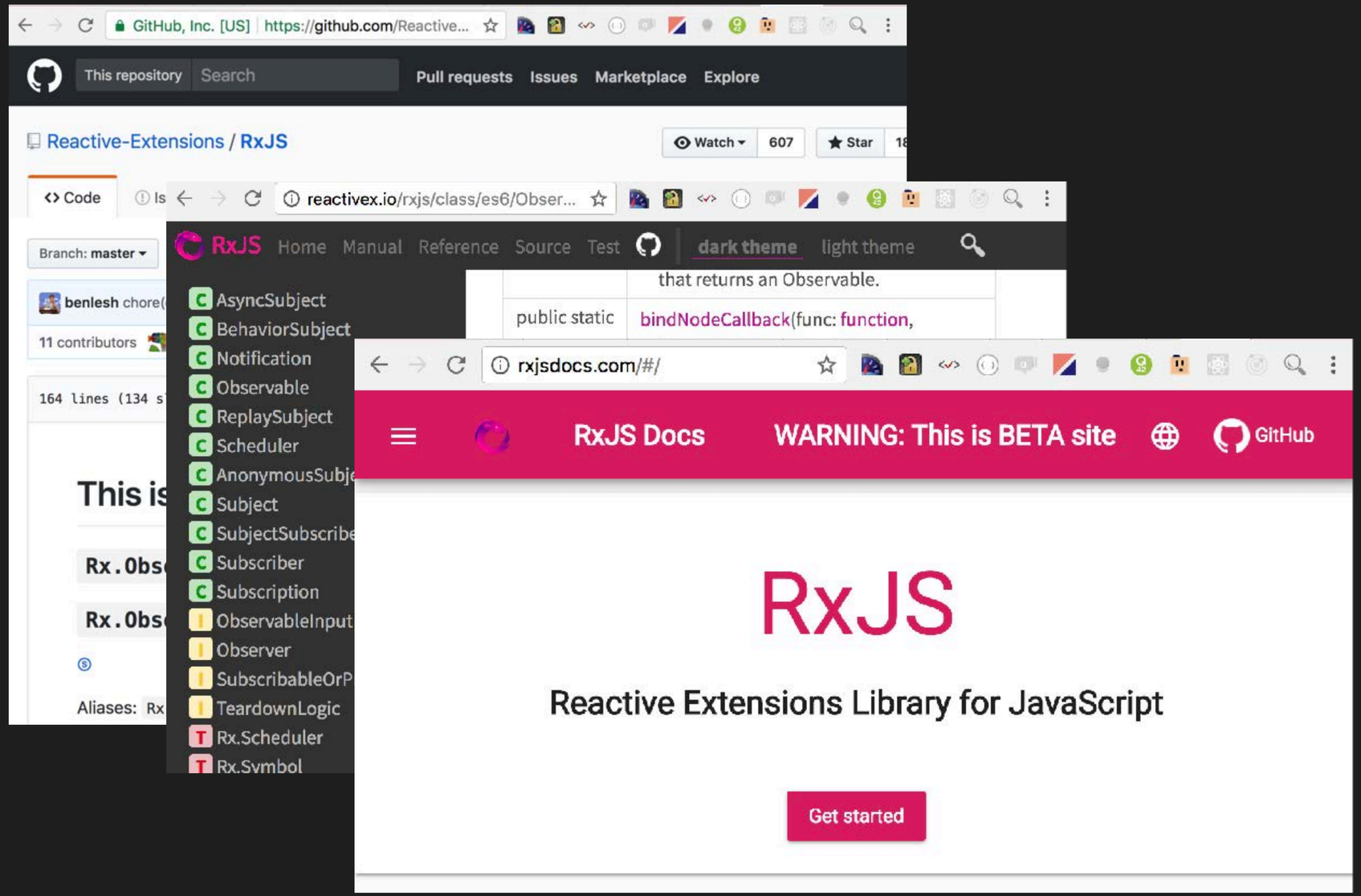
OOM

```
stockTicker() : Observable<StockPrice> {
   return Observable.interval(100)
      .concatMap(_ ⇒ fetchPricesFromSlowServer());
}
```

```
uploadImages(path: string): Observable<string> {
  return readDir(path, "*.png")
    .mergeMap(image ⟹ uploadImage(image));
}
```

429 Too Many Requests

INCORRECT OPERATOR


GitHub, Inc. [US] https://github.com/Reactive...
This repository
Search
Pull requests
Issues
Marketplace
Explore
Reactive-Extensions / RxJS
Watch
607
Star
Code
Branch: master
benlesh chore(
11 contributors
164 lines (134 s
This is
Rx.Obs
Rx.Obs
Aliases: Rx
reactivex.io/rxjs/class/es6/Obser...
RxJS
Home
Manual
Reference
Source
Test
dark theme
light theme
AsyncSubject
BehaviorSubject
Notification
Observable
ReplaySubject
Scheduler
AnonymousSubje
Subject
SubjectSubscribe
Subscriber
Subscription
ObservableInput
Observer
SubscribableOrP
TeardownLogic
Rx.Scheduler
Rx.Symbol
that returns an Observable.
public static
bindNodeCallback(func: function,
rxjsdocs.com/#/
RxJS Docs
WARNING: This is BETA site
GitHub
RxJS
Reactive Extensions Library for JavaScript
Get started

MARBLE DIAGRAMS

rxmarbles.com/#debounce

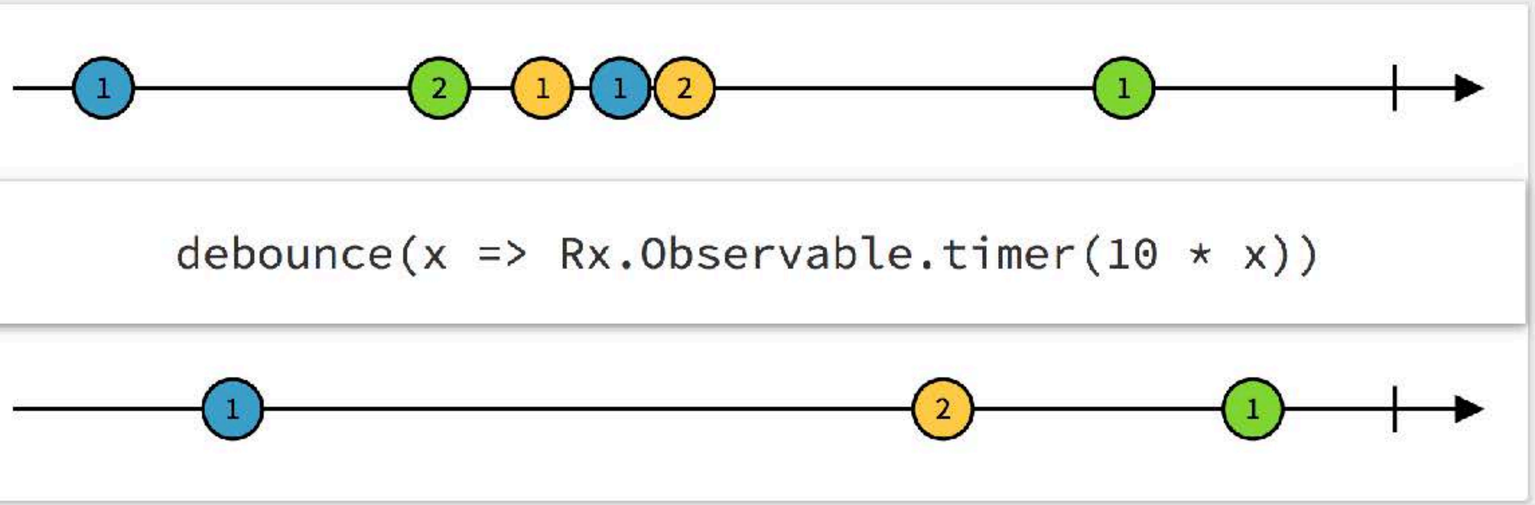
RxJS Marbles

Interactive diagrams of Rx Observables

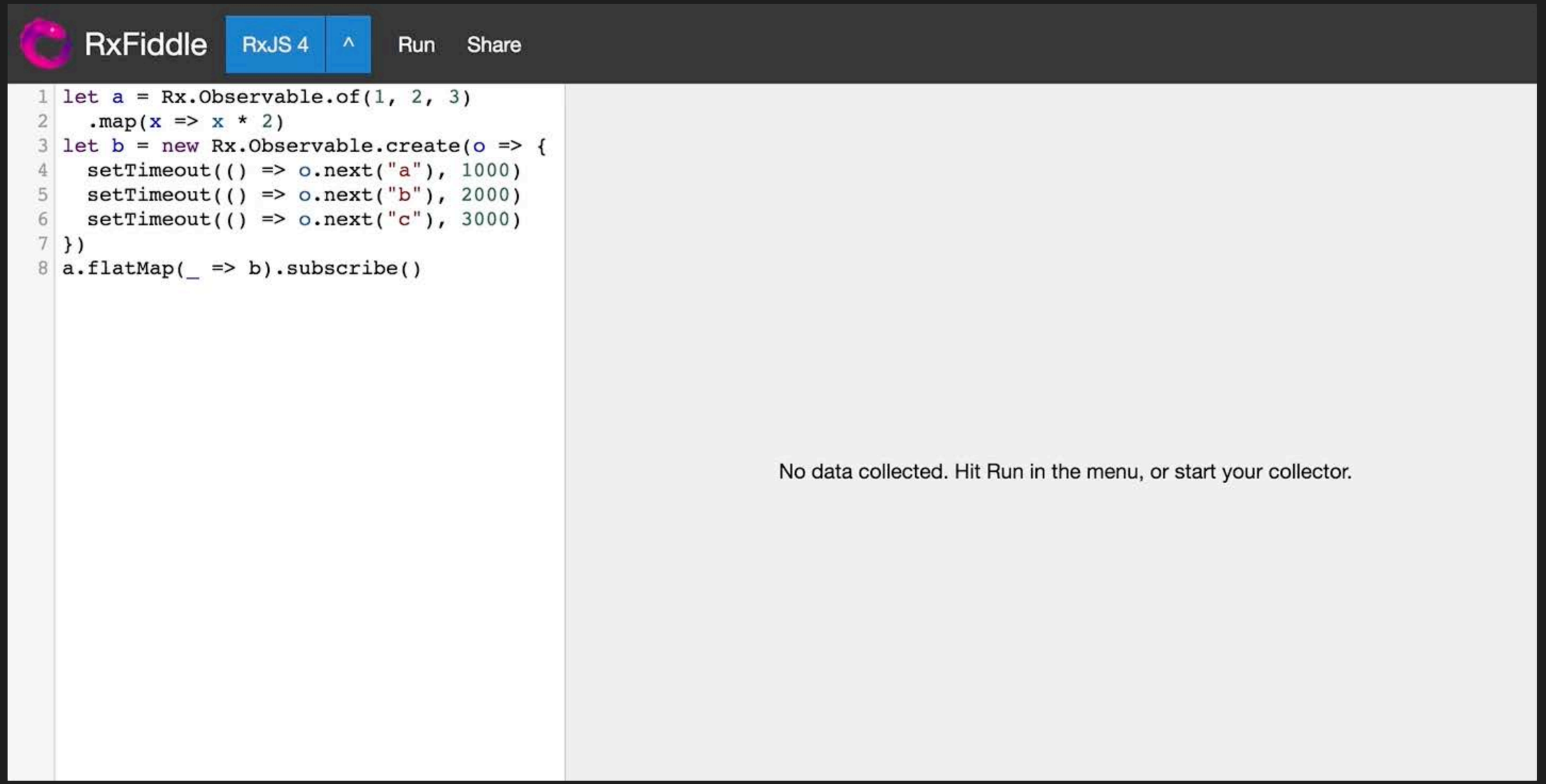
- defaultIfEmpty
- every
- sequenceEqual

COMBINATION OPERATORS

- combineLatest
- concat
- merge
- race
- startWith
- withLatestFrom
- zip



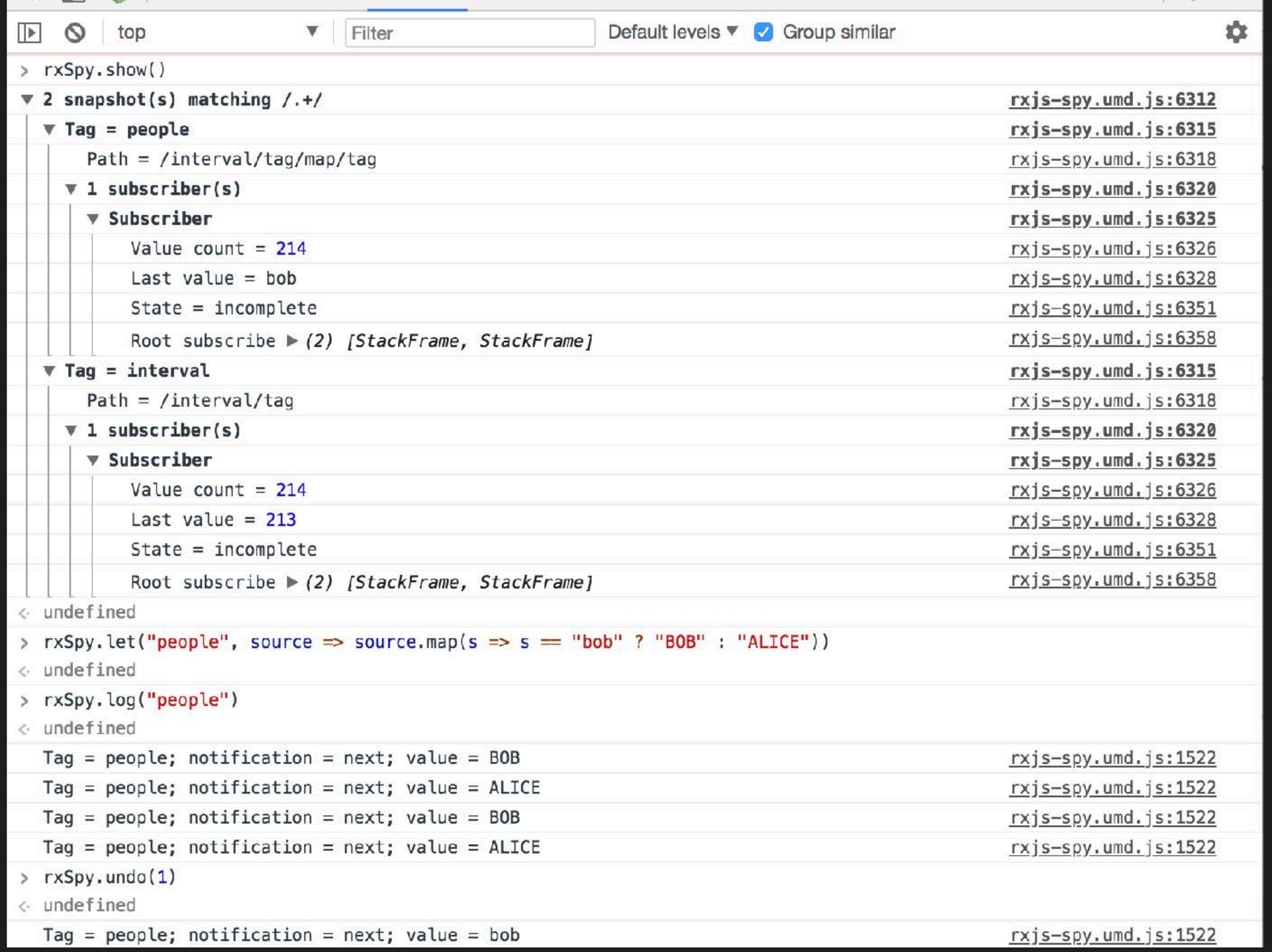
RXFIDDLE


RxFiddle
RxJS 4
^
Run
Share
1 let a = Rx.Observable.of(1, 2, 3)
2 .map(x => x * 2)
3 let b = new Rx.Observable.create(o => {
4 setTimeout(() => o.next("a"), 1000)
5 setTimeout(() => o.next("b"), 2000)
6 setTimeout(() => o.next("c"), 3000)
7 })
8 a.flatMap(_ => b).subscribe()
No data collected. Hit Run in the menu, or start your collector.

RXSPY

```
<script>
  var spy = RxSpy.create();
  (function () {
    var interval = new Rx.Observable
      .interval(2000)
      .tag("interval");

    var people = interval
      .map((value) ⇒ {
          const names = ["alice", "bob"];
          return names[value % names.length];
      })
      .tag("people")
      .subscribe();
  })();
</script>
```


top
Filter
Default levels
Group similar
> rxSpy.show()
2 snapshot(s) matching /.+/ rxjs-spy.umd.js:6312
Tag = people rxjs-spy.umd.js:6315
Path = /interval/tag/map/tag rxjs-spy.umd.js:6318
1 subscriber(s) rxjs-spy.umd.js:6320
Subscriber rxjs-spy.umd.js:6325
Value count = 214 rxjs-spy.umd.js:6326
Last value = bob rxjs-spy.umd.js:6328
State = incomplete rxjs-spy.umd.js:6351
Root subscribe ▶ (2) [StackFrame, StackFrame] rxjs-spy.umd.js:6358
Tag = interval rxjs-spy.umd.js:6315
Path = /interval/tag rxjs-spy.umd.js:6318
1 subscriber(s) rxjs-spy.umd.js:6320
Subscriber rxjs-spy.umd.js:6325
Value count = 214 rxjs-spy.umd.js:6326
Last value = 213 rxjs-spy.umd.js:6328
State = incomplete rxjs-spy.umd.js:6351
Root subscribe ▶ (2) [StackFrame, StackFrame] rxjs-spy.umd.js:6358
< undefined
> rxSpy.let("people", source => source.map(s => s == "bob" ? "BOB" : "ALICE"))
< undefined
> rxSpy.log("people")
< undefined
Tag = people; notification = next; value = BOB rxjs-spy.umd.js:1522
Tag = people; notification = next; value = ALICE rxjs-spy.umd.js:1522
Tag = people; notification = next; value = BOB rxjs-spy.umd.js:1522
Tag = people; notification = next; value = ALICE rxjs-spy.umd.js:1522
> rxSpy.undo(1)
< undefined
Tag = people; notification = next; value = bob rxjs-spy.umd.js:1522

```
// create repl for rxspy usage after app code
const rxspy = create();
import * as repl from 'repl';
const replServer = repl.start({
  prompt: "rx-spy debugger > ",
});
replServer.context.rxspy = rxspy;
```

```
rx-spy debugger > rxspy.log('people')
[Function]
rx-spy debugger > Tag = people; notification = next;
value = alice
Tag = people; notification = next; value = bob
Tag = people; notification = next; value = alice
Tag = people; notification = next; value = bob
```

HOW TO PREVENT THESE BUGS

TYPESCRIPT

KISS

```
return readdir(path)
   .filter(file ⇒ file.mimeType === "image/png")
   .filter(file ⇒ file.fileSize < 10 * 1024 )
   .catch((err) ⇒ {
      console.log("reading images failed: " +
err.message)
      return Observable.empty<string>();
   })
   .mergeMap(file ⇒ {
      return uploadImage(path)
         .retryWhen(err ⇒ err
            .delay(500)
            .take(3)
               .concat(
                  Observable.throw(new Error('upload
```

```
readImagesInDir(path)
  .mergeMap(image ⇒ uploadImageToCdn(image.path))
  .mergeMap(uploadRes ⇒ saveCdnUrlInDb(uploadRes))
```

DOCUMENT STREAMS

```
/**
 * persists to our local SQL instance where the
 * file with the given path has been uploaded
 * to our content delivery network
 *
 * side-effects:  mutates the database
 * hot/cold:      cold
 * lifecycle:     Error: Has built in retry logic with
 *                small backoff (500ms). After 3
 *                attempts will throw error.
 *                Completes: after url has been
 *                persisted
 *
 * @param uploadResult
 */
function saveCdnUrlInDb(uploadResult: { path: string,
```

TESTS

All files

92.42% Statements `805/871` **81.43%** Branches `228/280`

File	Statements
src	86.9%
src/lib	91.75%
src/lib/channels	86.84%
src/lib/errors	100%
src/lib/rx-adapters	92.31%
src/lib/rx-operators	98.51%
src/lib/serialization	98.46%
src/routes	100%

RECAP

Conventional debugging
techniques still apply

.do(), .debug() and dependency graphs are low barrier ways to gain insights in values and stream lifecycle

rxfiddle and rx-spy are very helpful
to learn how streams transform
values and debug these

But above all; Prevent bugs using typescript, code hygiene and tests

IS RXJS WORTH IT?

"I was taught that the way of progress was neither swift nor easy."

–Marie Curie


THANKS FOR
LISTENING
@markvanstraten
q42.com
Q
42

REFERENCES

- https://gist.github.com/crunchie84/f212e3674ea02480431716bebd214ea3 (debug operator)
- https://rxfiddle.net/
- https://cartant.github.io/rxjs-spy/
- https://medium.com/@benlesh/rxjs-dont-unsubscribe-6753ed4fda87
- https://tech.residebrokerage.com/debugging-node-js-memory-problems-d450787d9253