DoubleVerify

Reactive JavaScript

Tim De Pauw

👋 Hi!

Tim De Pauw
Engineering Manager
DoubleVerify Ghent

✉ tim.depauw@doubleverify.com
tmdpw
timdp

🌎 DoubleVerify, Inc.

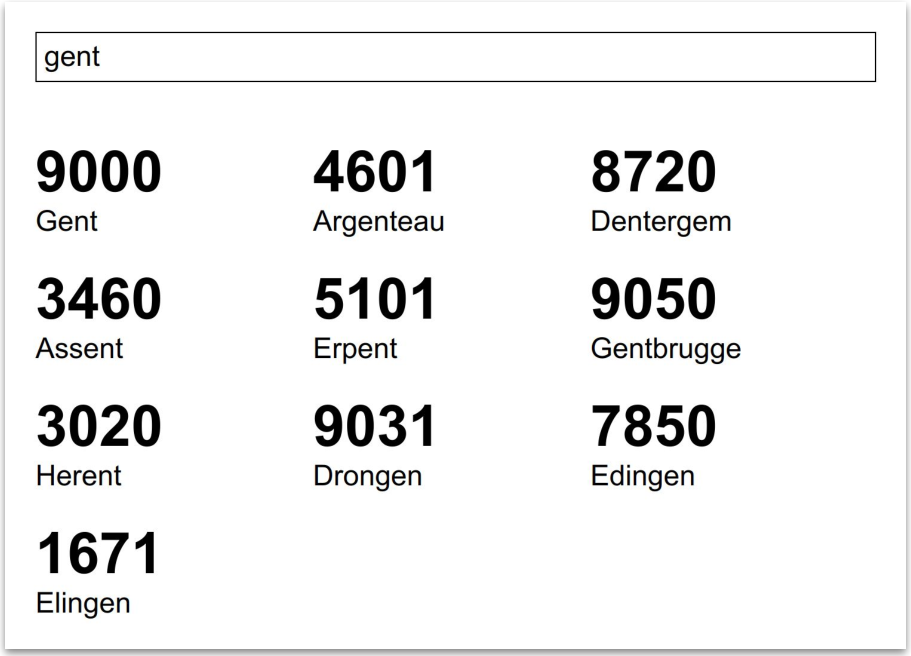
- Marketing measurement
- Authentic Impression®
  - Fully viewed
  - By a real person
  - In a brand-safe environment
- Let's build a better industry™
- Offices worldwide
- 450+ people

BRAND SAFE
VIEWABLE
DV AUTHENTIC IMPRESSION®
FRAUD FREE
IN GEO

🍟 DoubleVerify Ghent

- Two product lines:
  - Video measurement for advertisers
  - Yield optimization for publishers
- Truly full-stack JavaScript:
  - Lots of front end
  - Lots of back end
  - Lots of devops
- Small team with big plans

⚔️ Coding Challenge: ZIP Code Lookup



gent

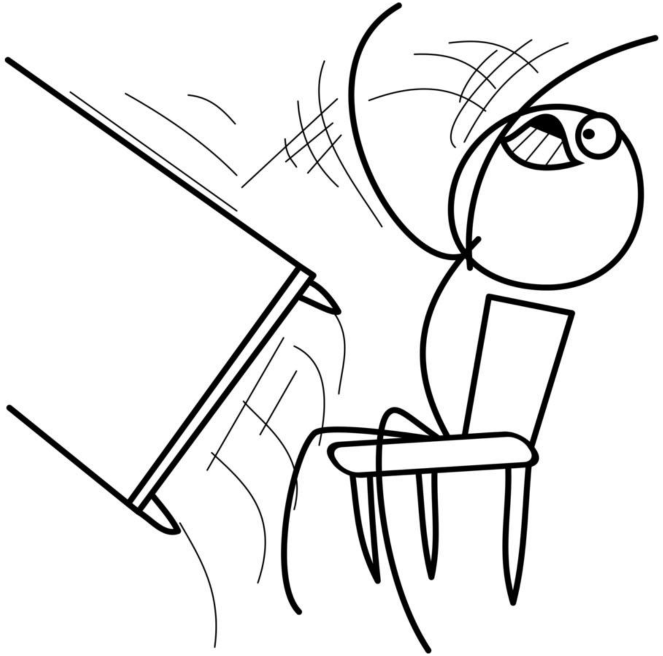
9000 Gent	**4601** Argenteau	**8720** Dentergem
3460 Assent	**5101** Erpent	**9050** Gentbrugge
3020 Herent	**9031** Drongen	**7850** Edingen
1671 Elingen		

👷 Let’s Build It!

How hard could it be?

🤒 Known Issues

- New request every keystroke
  - Server: high load
  - Client: concurrent requests/responses
- Empty input is special case
- Hard to test



✨ Reactive Programming

- It's like working with **lists** (or arrays) over **time**
- **Streams** (observables) and **operators**
- Wire streams up, then kick things off

JIWI'S MACHINES

✨ Reactive Programming

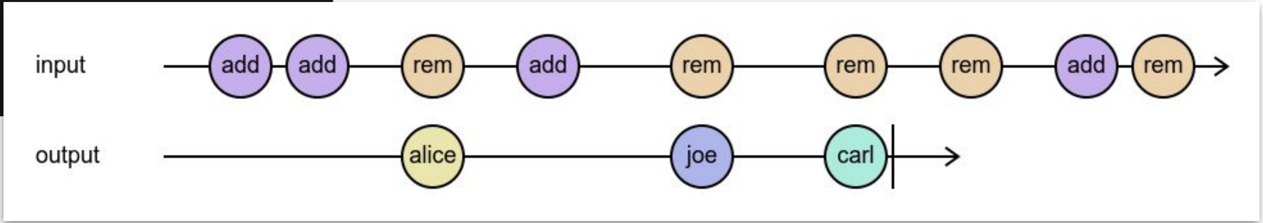
- It’s like working with **lists** (or arrays) over **time**
- **Streams** (observables) and **operators**
- Wire streams up, then kick things off
- Language-agnostic
- JavaScript: **RxJS**

🔎 JavaScript Array vs. RxJS Observable

```
const events = [
  { type: 'userAdded', name: 'alice' },
  { type: 'userAdded', name: 'bob' },
  { type: 'userRemoved', name: 'alice' },
  { type: 'userAdded', name: 'carl' },
  // etc.
]

const firstThreeRemovedUsers = events
  .filter(({ type }) => type === 'userRemoved')
  .map(({ name }) => name)
  .slice(0, 3)
```

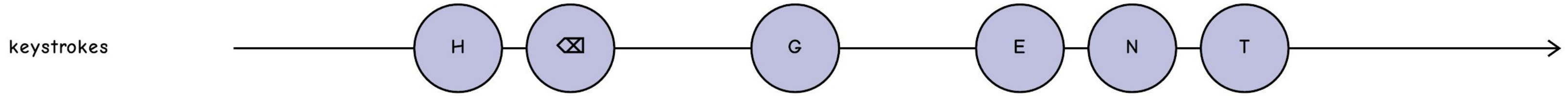
```
const firstThreeRemovedUsers$ = events$.pipe(
  filter(({ type }) => type === 'userRemoved'),
  map(({ name }) => name),
  take(3)
)
```



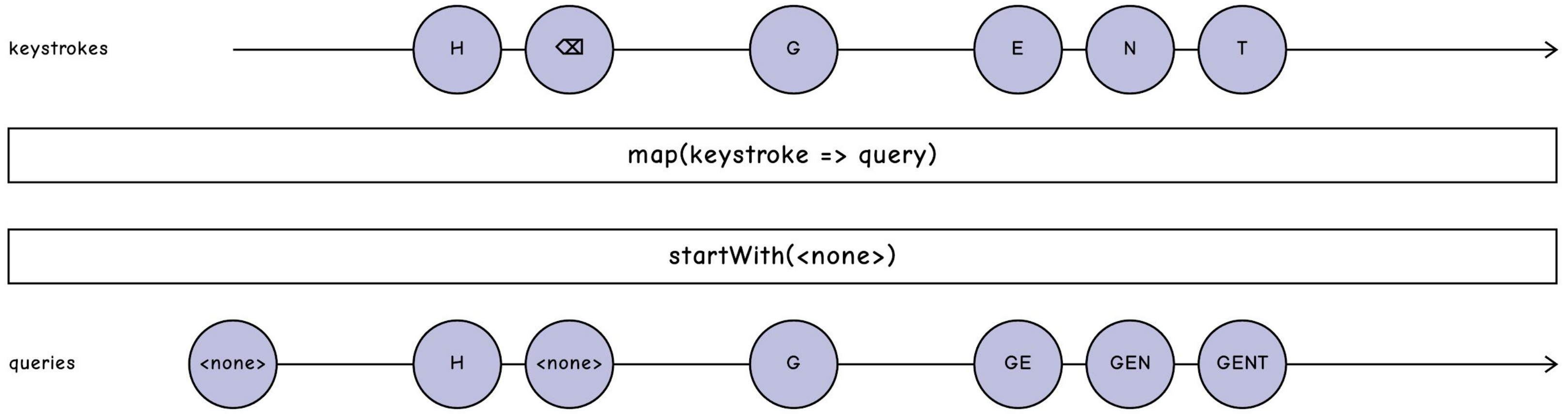
🤔 Let's Think Reactive(ly)!

What streams do we need?

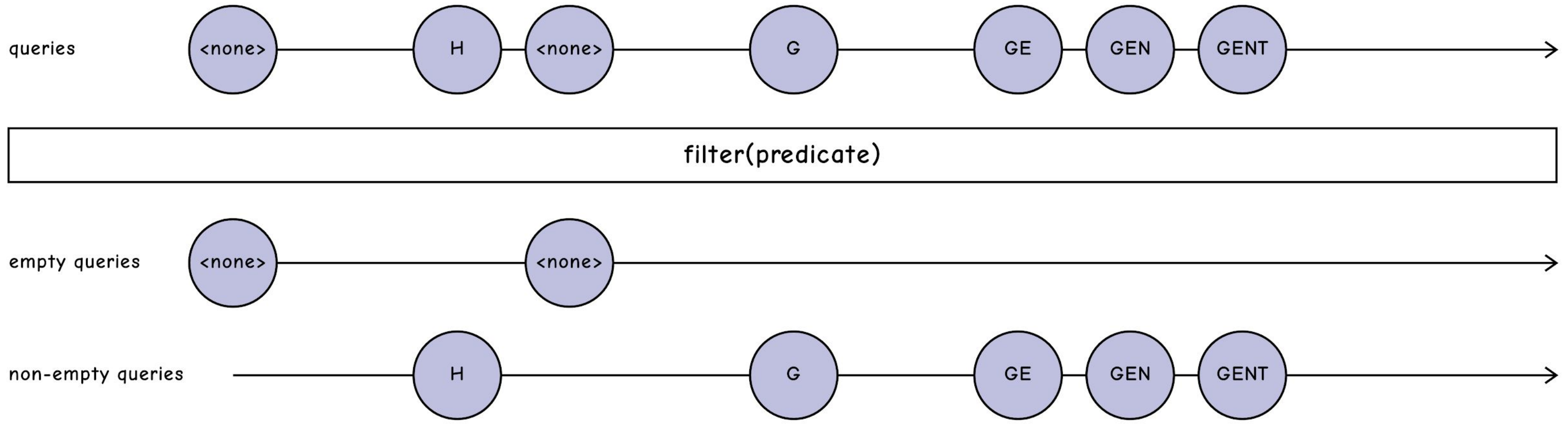
🌊 Handling User Input



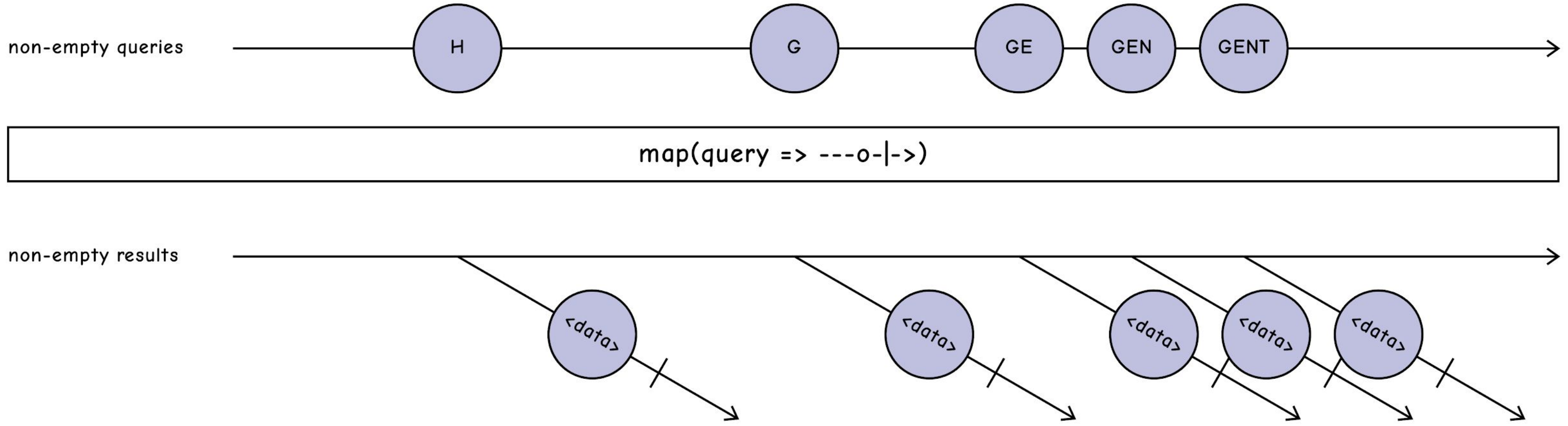
🌊 Transforming User Input to Queries



🌊 Splitting Queries

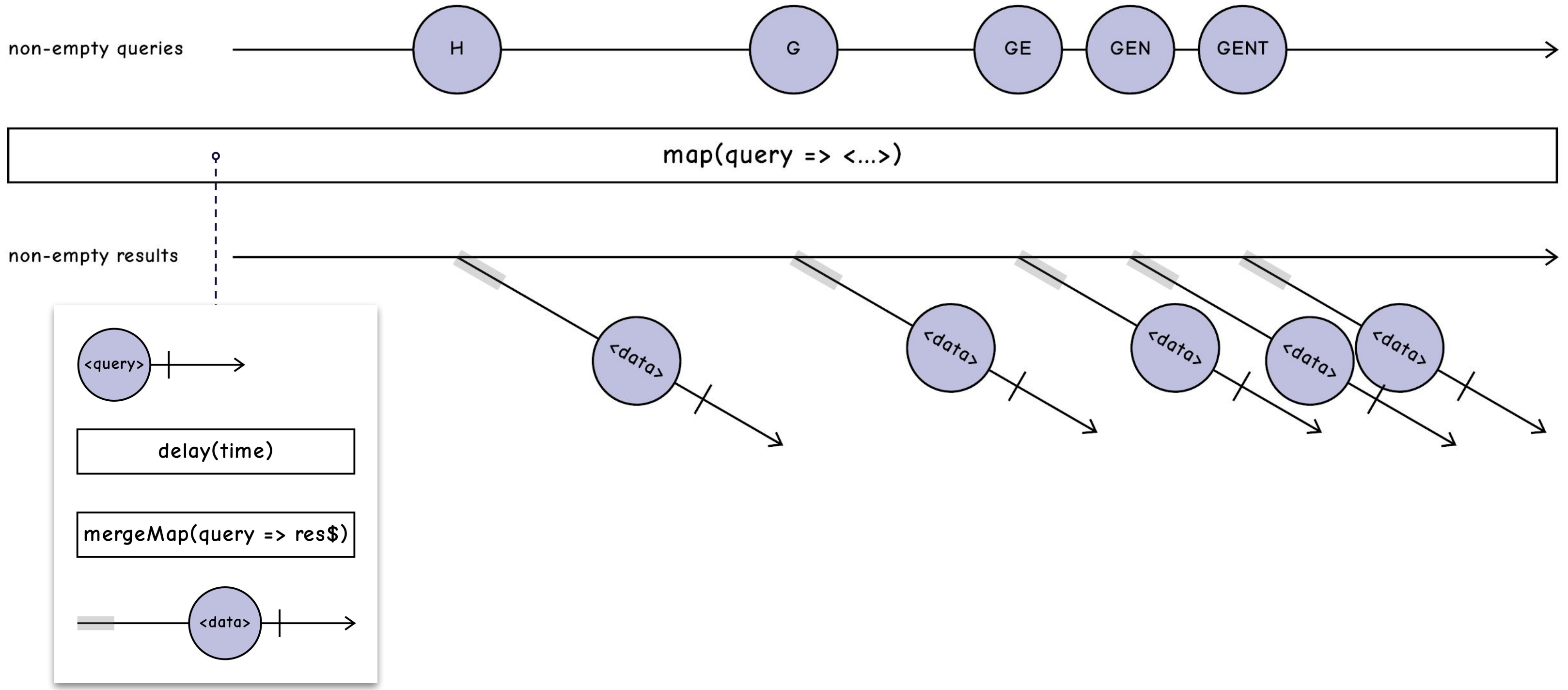


🌊 Loading Search Results

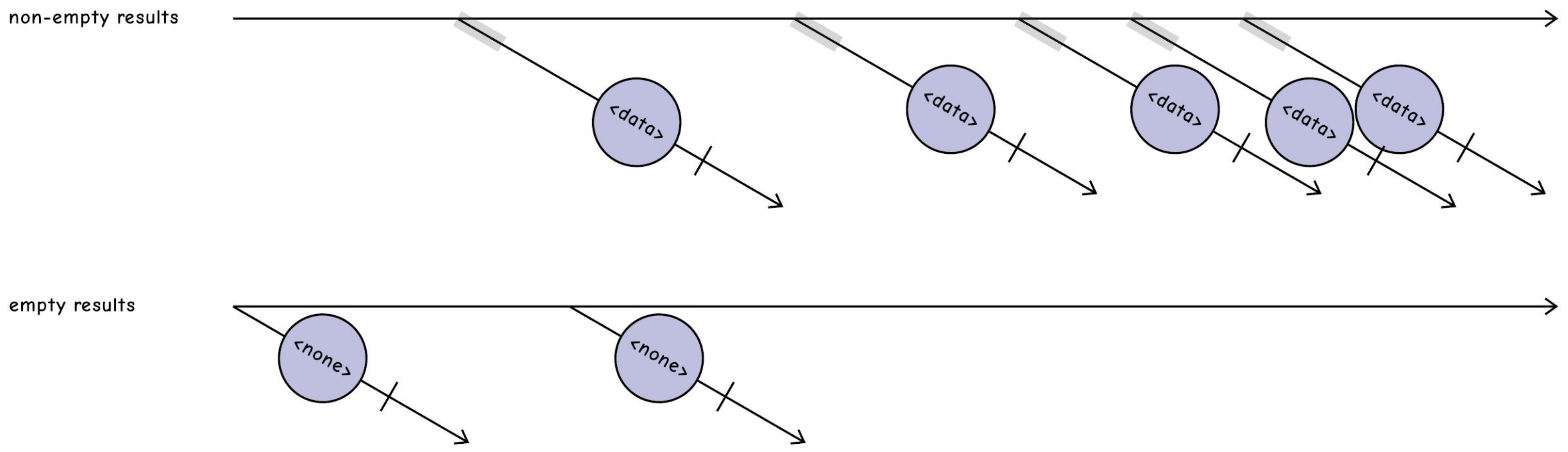


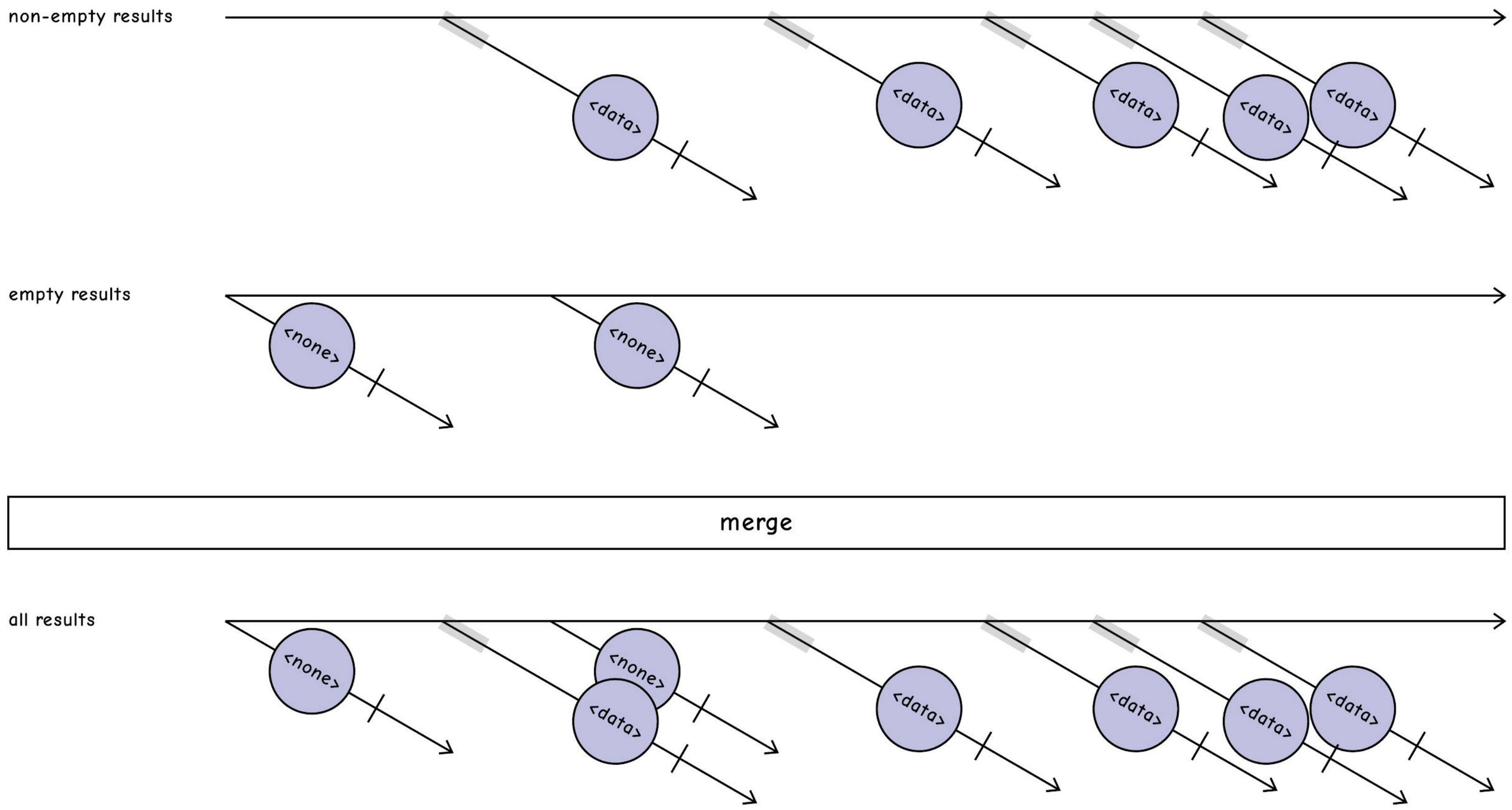
*This is a **higher-order observable***

🌊 Debouncing Searches



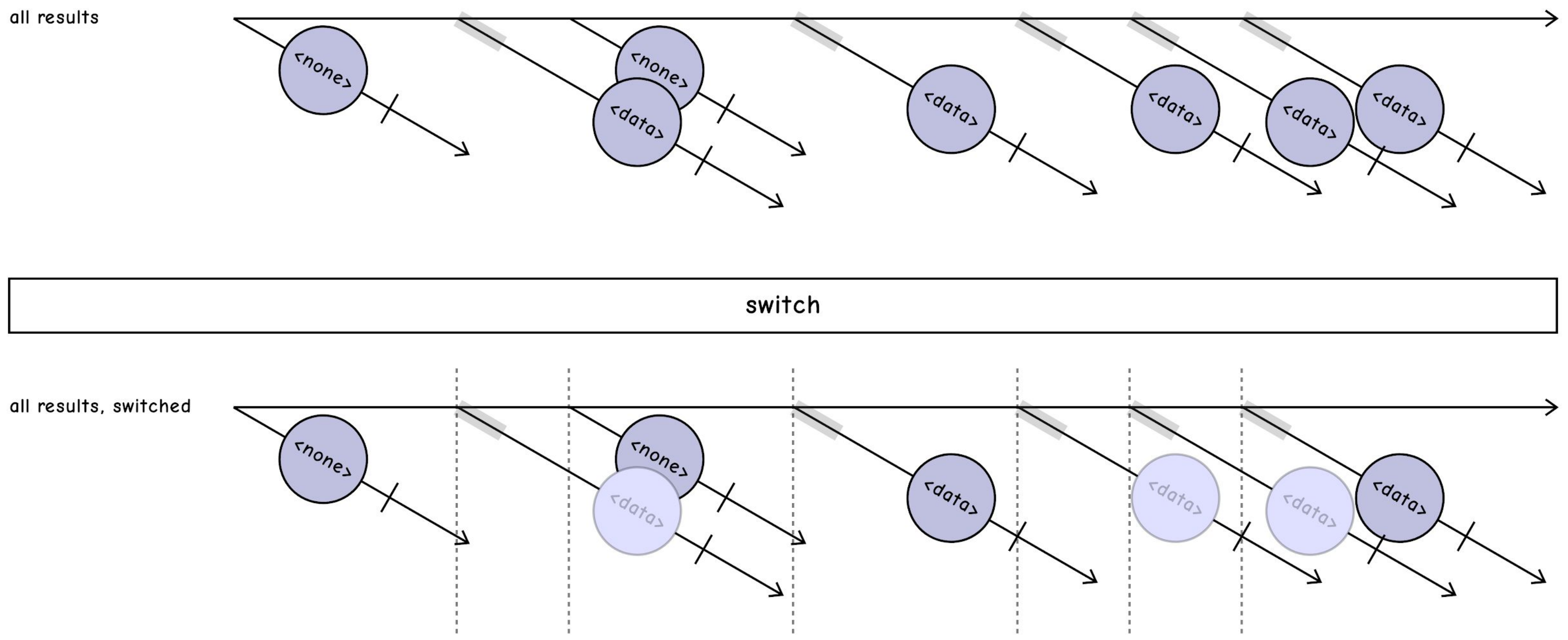
🌊 Unifying Result Streams

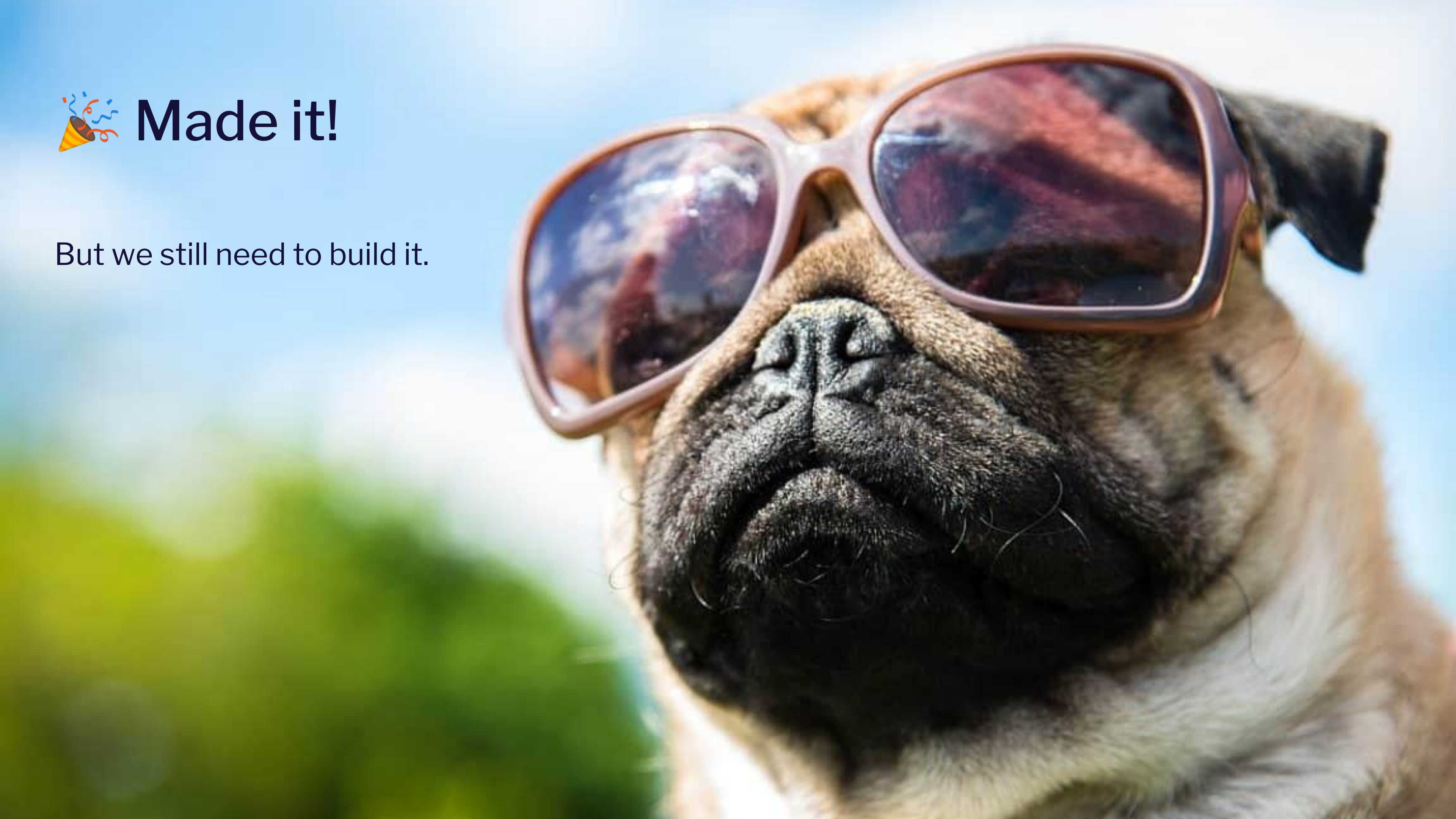




DoubleVerify

🌊 Canceling Ongoing Searches

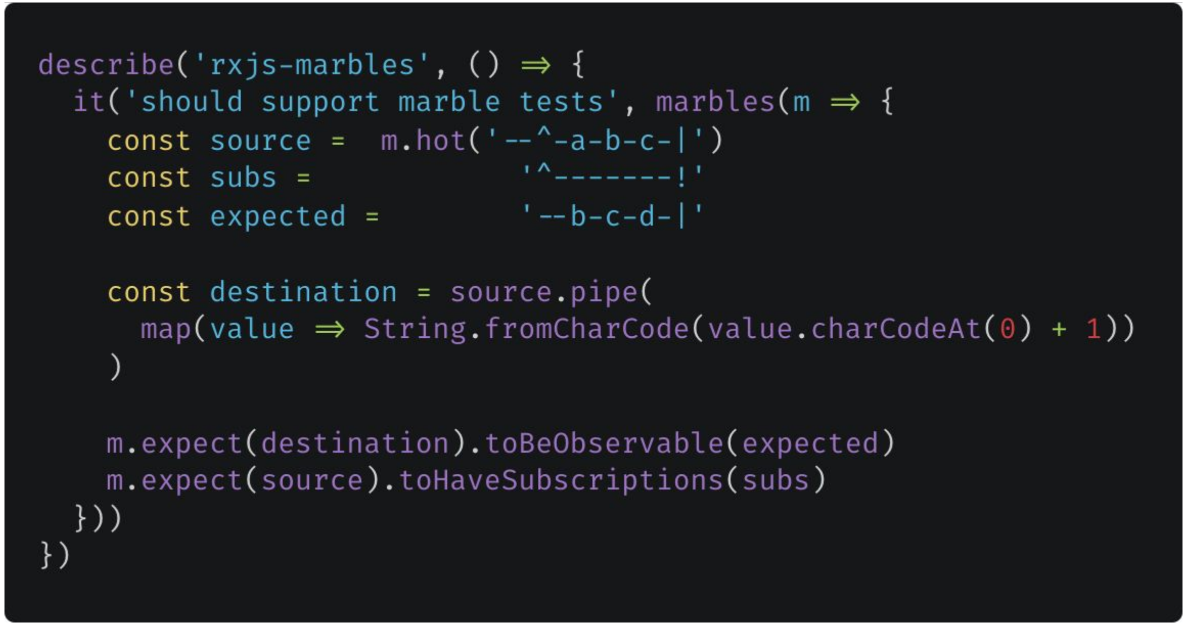




🎉 Made it!

But we still need to build it.

🐒 Testing Observables: rxjs-marbles



```
describe('rxjs-marbles', () => {
  it('should support marble tests', marbles(m => {
    const source =  m.hot('--^-a-b-c-|')
    const subs =            '^-------!'
    const expected =        '--b-c-d-|'

    const destination = source.pipe(
      map(value => String.fromCharCode(value.charCodeAt(0) + 1))
    )

    m.expect(destination).toBeObservable(expected)
    m.expect(source).toHaveSubscriptions(subs)
  }))
})
```

📚 Resources

- ‘The introduction to Reactive Programming you’ve been missing’ https://gist.github.com/staltz/868e7e9bc2a7b8c1f754
- Rx for your favorite language http://reactivex.io
- **RxJS documentation** (careful: raw API docs ahead) http://reactivex.io/rxjs
- Interactive playground: **RxJS Marbles** https://rxmarbles.com
- Testing framework: **rxjs-marbles** https://cartant.github.io/rxjs-marbles/
- React + Redux: **redux-observable** https://redux-observable.js.org
- Reactive Web server: **Marble.js** https://marblejs.com
- Marble diagram generator: **Swirly** https://swirly.now.sh

🙋‍♂️

Q&A

DoubleVerify

Thanks!

tim.depauw@doubleverify.com

doubleverify.gent