Angular Fundamentals
Module 4 – Observables

Peter Kassenaar
info@kassenaar.com

Async services with RxJS/Observables

Reactive programming
with asynchronous streams

Async Services

- Fetching static data: *synchronous* action
- Working via `HttpClient`: *asynchronous* action
- Angular 1: `Promises`
- Angular 2: `Observables`

Angular : ReactiveX library `RxJS`

ReactiveX Introduction Docs Languages Resources Community

Reactiv

An API for asynchronous pr
with observable streams

Choose your platform

ReactiveX Introduction Docs Languages Resources Community

Languages

- Java: RxJava
- JavaScript: RxJS
- C#: Rx.NET
- C#(Unity): UniRx
- Scala: RxScala
- Clojure: RxClojure
- C++: RxCpp
- Ruby: Rx.rb
- Python: RxPY
- Groovy: RxGroovy
- JRuby: RxJRuby
- Kotlin: RxKotlin
- Swift: RxSwift

ReactiveX for platforms and frameworks

- RxNetty
- RxAndroid
- RxCocoa

DOCUMENTATION
Observable
Operators
Single

LANGUAGES
RxJava
RxJS
Rx.NET

RESOURCES
Tutorials

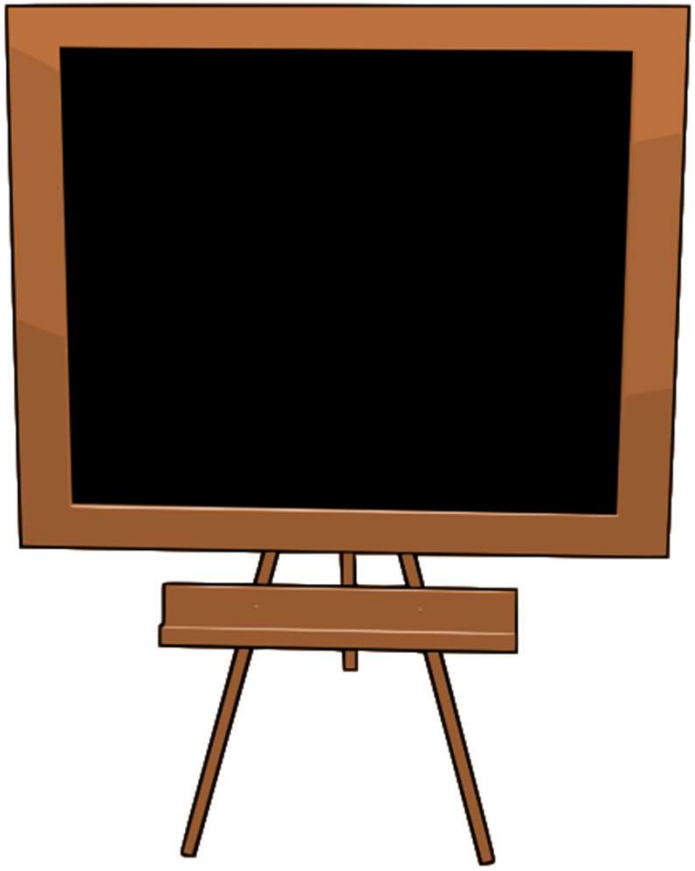
COMMUNITY
GitHub
Twitter
Others

http://reactivex.io/

The observable design pattern

"Understanding the observable stream"

Explanation



Why Observables?

"We can do much more with observables than with promises.

With observables, we have a whole bunch of operators to pull from, which let us customize our streams in nearly any way we want."

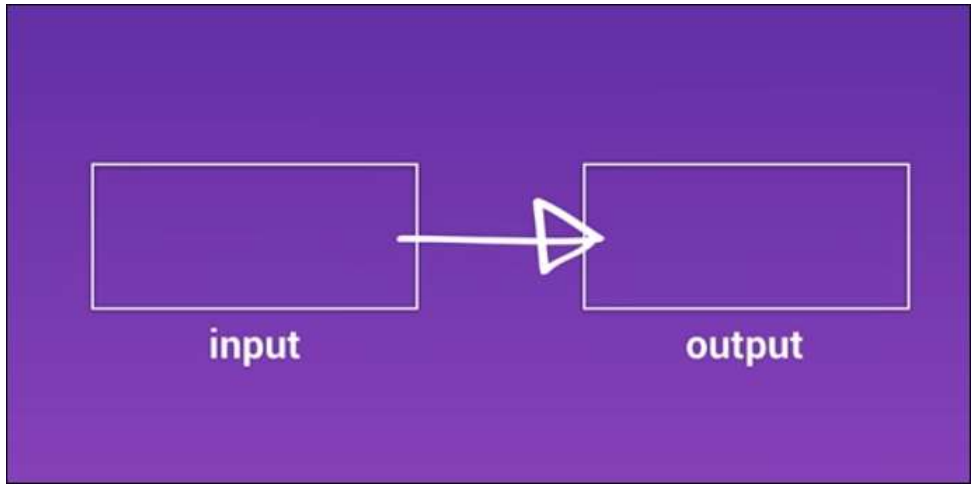
https://auth0.com/blog/2015/10/15/angular-2-series-part-3-using-http/

Observables and RxJs

- "Reactive Programming"
  - *"Reactive programming is programming with asynchronous data streams."*
  - https://gist.github.com/staltz/868e7e9bc2a7b8c1f754
- Observables have a lot of extra options, as opposed to Promises
  - Mapping
  - Filtering
  - Combining
  - Cancel
  - Retry
  - …
- Consequence: no more `.success()`, `.error()` and `.then()` chaining!

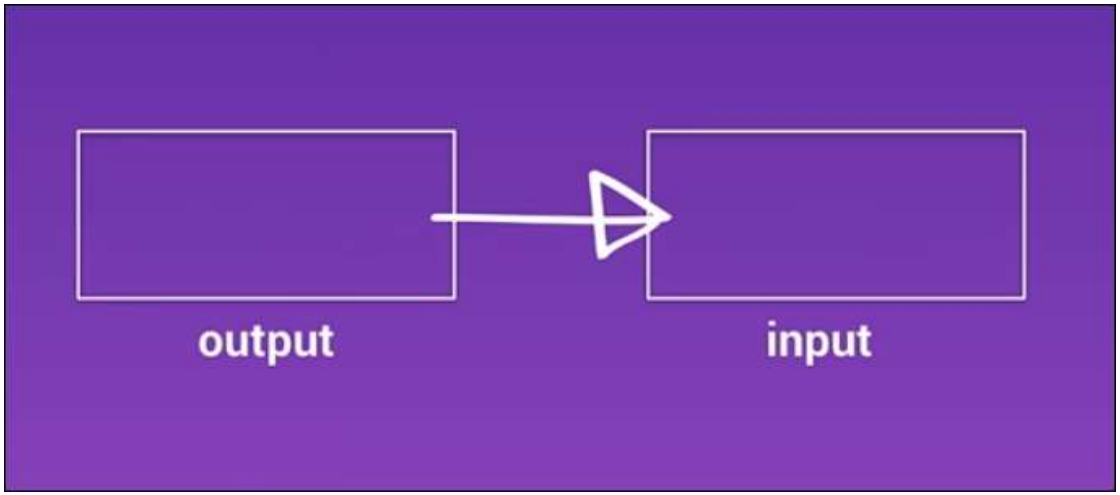
How do observables work

- First - The Observable Stream
- Later - all 10.000 operators…
- Traditionally:

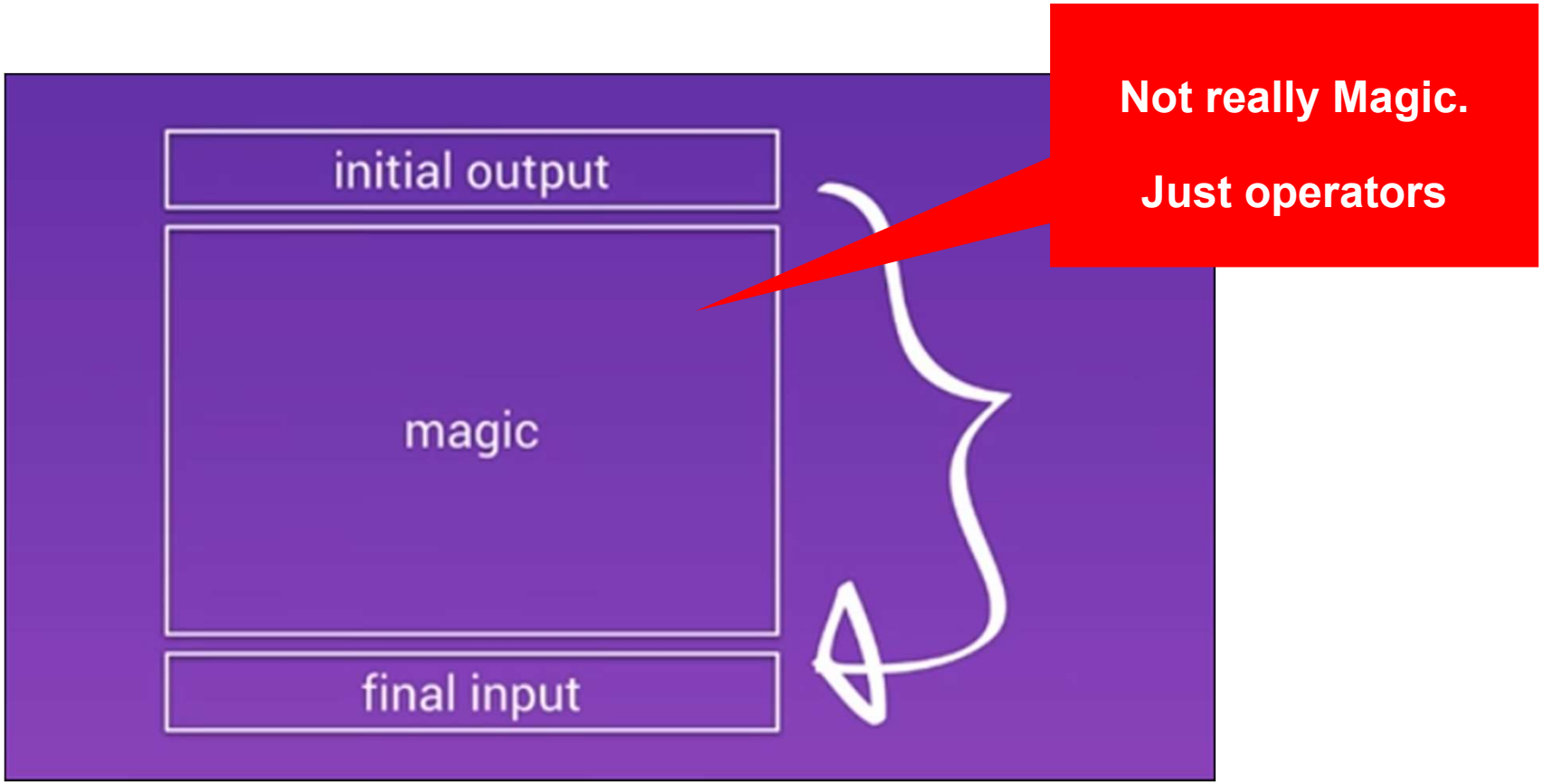


https://www.youtube.com/watch?v=5CTL7aqSvJU

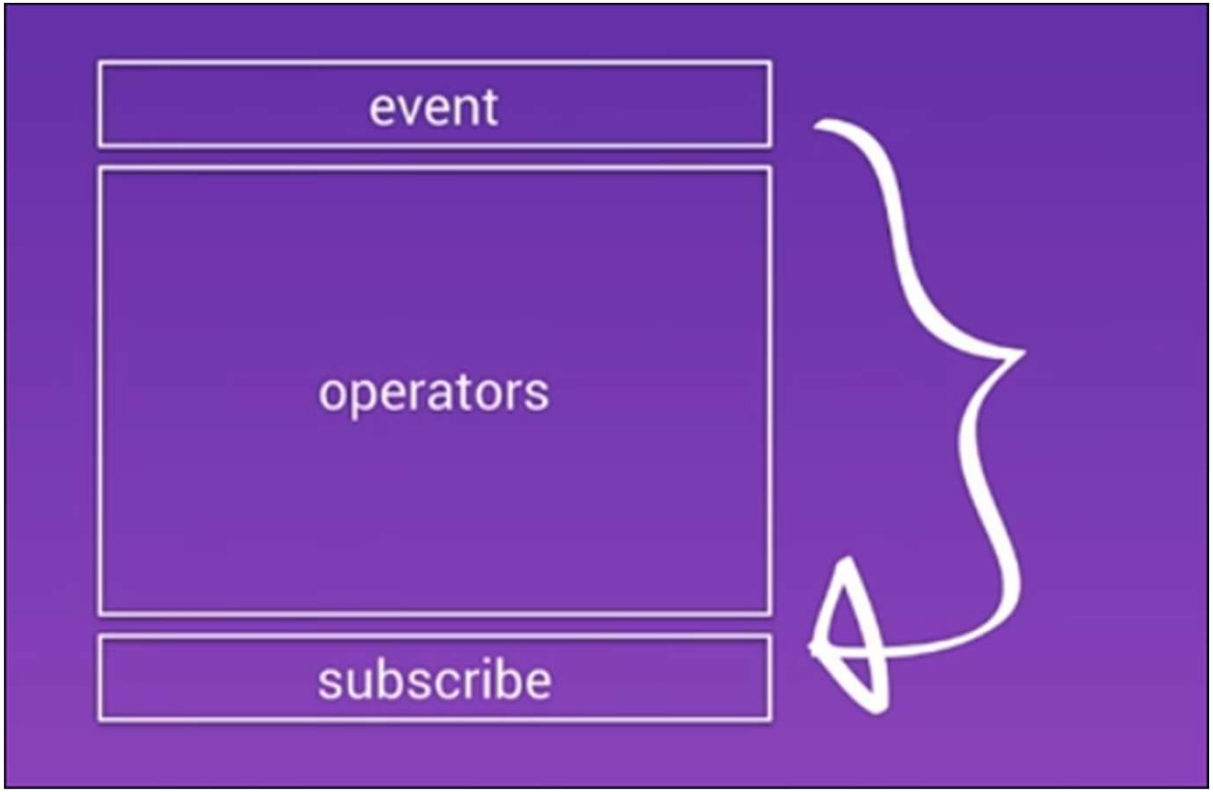
- With Observables -
  - a system, already outputting data,
  - Subscribe to that data
- "trade Output for Input"
- "Push vs. Pull"



"The observable sandwich"



“The Observable Sandwich"



In code, for http-call:

```
this.http.get<City[]>('assets/data/cities.json')
    .pipe(
        filter(…),
        map(…)
    )
    .subscribe((result) => {
      //... Do something
  });
```

Initial Output

Optional: operator(s)

Final Input

Import `HttpClientModule` in `@ngModule` – don't forget

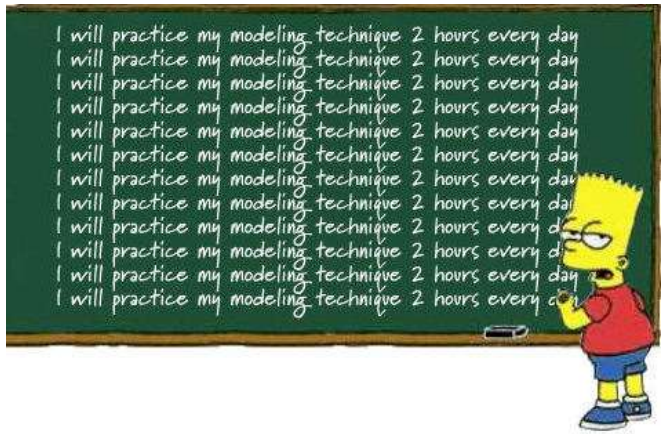
- *// Angular Modules*
  …
- **import** {HttpClientModule} **from '@angular/common/http'**;

```
// Module declaration
@NgModule({
    imports      : [BrowserModule, HttpClientModule],
    declarations: [AppComponent],
    bootstrap    : [AppComponent],
    providers    : [CityService] // DI voor service
})
export class AppModule {
}
```

Exercise

- See the example in `/201_services_http`
- Create your own `.json`-file and import it in your application.
- Exercise `5c), 5d)`

Exercise….



More on subscriptions

Using parameters inside the subscriber

Subscribe - only once per block!

- Part of RxJs
- Three parameters:
  - `success()`
  - `error()`
  - `complete()`

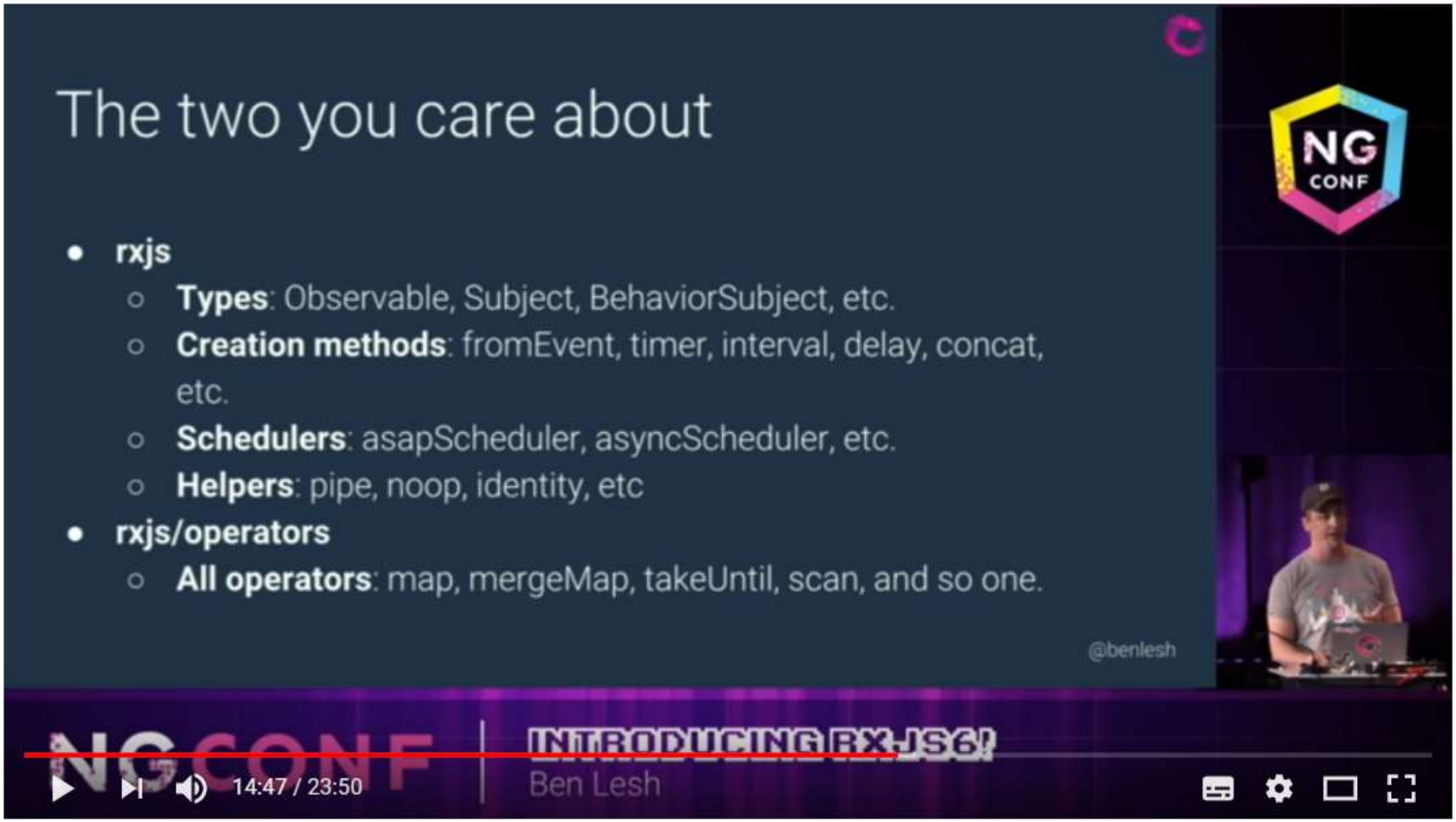
```
this.cityService.getCities()
   .subscribe(cityData => {
         this.cities = cityData
      },
      err => console.log(err),
      ()=> console.log('Getting cities complete...')
   )
```

Pipeable operators

- In RxJS 6.x and up: all operators inside `.pipe()` functie
- The parameters of pipe function are the operators!
- Comma-separate different operators.
  - Don't forget `import {…} from 'rxjs/operators';`

```
.pipe(
   delay(3000),
   retry(3),
   map(result => …),
   takeUntil(…condition…)
)
```

Background info: Ben Lesh on observables in RxJS 6.0



Introducing RxJS6! - Ben Lesh

https://www.youtube.com/watch?v=JCXZhe6KsxQ

https://www.learnrxjs.io/

learnrxjs.io/learn-rxjs/operators

Learn RxJS

Introduction

LEARN RXJS

Operators
- Combination
- Conditional
- Creation
- Error Handling
- Multicasting
- Filtering
- Transformation
- Utility
- Full Listing

Subjects

Recipes

Concepts

Powered by GitBook

Operators

CONTENTS

A complete list of RxJS operators with clear explanations, relevant resources, and executable examples.

Prefer a complete list in alphabetical order?

Contents (By Operator Type)

- Combination
  - combineAll
  - combineLatest ⭐
  - concat ⭐
  - concatAll
  - endWith
  - forkJoin
  - merge ⭐
  - mergeAll
  - pairwise
  - race
  - startWith ⭐
  - withLatestFrom ⭐
  - zip
- Conditional
  - defaultIfEmpty
  - every
  - iif
  - sequenceequal
- Creation

Using the `async` pipe

Automagically `.subscribe()` and `.unsubscribe()`

Async Pipe

- On `.subscribe()`, you actually need to `.unsubscribe()` to avoid memory leaks
  - Best practice
  - Not *really* necessary on HTTP-requests, but you do in other subscriptions.
- No more manually `.subscribe()` and `.unsubscribe()`:
  - Use Angular `async` pipe

- Component:

```
cities$: Observable<City[]>; // Now: Observable to Type

…
ngOnInit() {
   // Call service, returns an Observable
   this.cities$ = this.cityService.getCities()
}
```

- View:

```
<li *ngFor="let city of cities$ | async”>
```

https://blog.angularindepth.com/angular-question-rxjs-subscribe-vs-async-pipe-in-component-templates-c956c8c0c794
(Background information)

Working with Live API's

- MovieApp
- `examples\210-services-live`

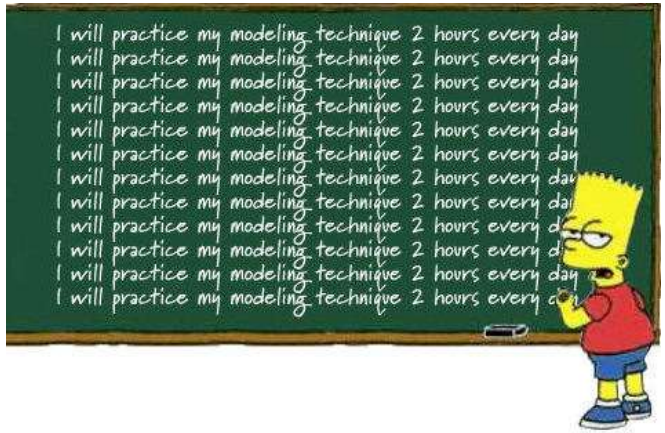
Example API's

- https://pokeapi.co/ - Pokemon API
- http://openweathermap.org/API (weather forecast)
- https://jsonplaceholder.typicode.com/ random users, posts, photos
- http://ergast.com/mrd/ - Ergast Motor (F1) API
- http://www.omdbapi.com/ - Open Movie Database
- http://swapi.co/ - Star Wars API
- See also `JavaScript APIs.txt` with other API's.

Exercise

- Pick one of your own projects, or see for examples:
  - `210-services-live`
- Create a small application using one of the API's in the file `JavaScript API's.txt`, using RxJS-calls, for example
  - Star Wars API
  - OpenWeatherMap API
  - …
- Exercise `5e)`

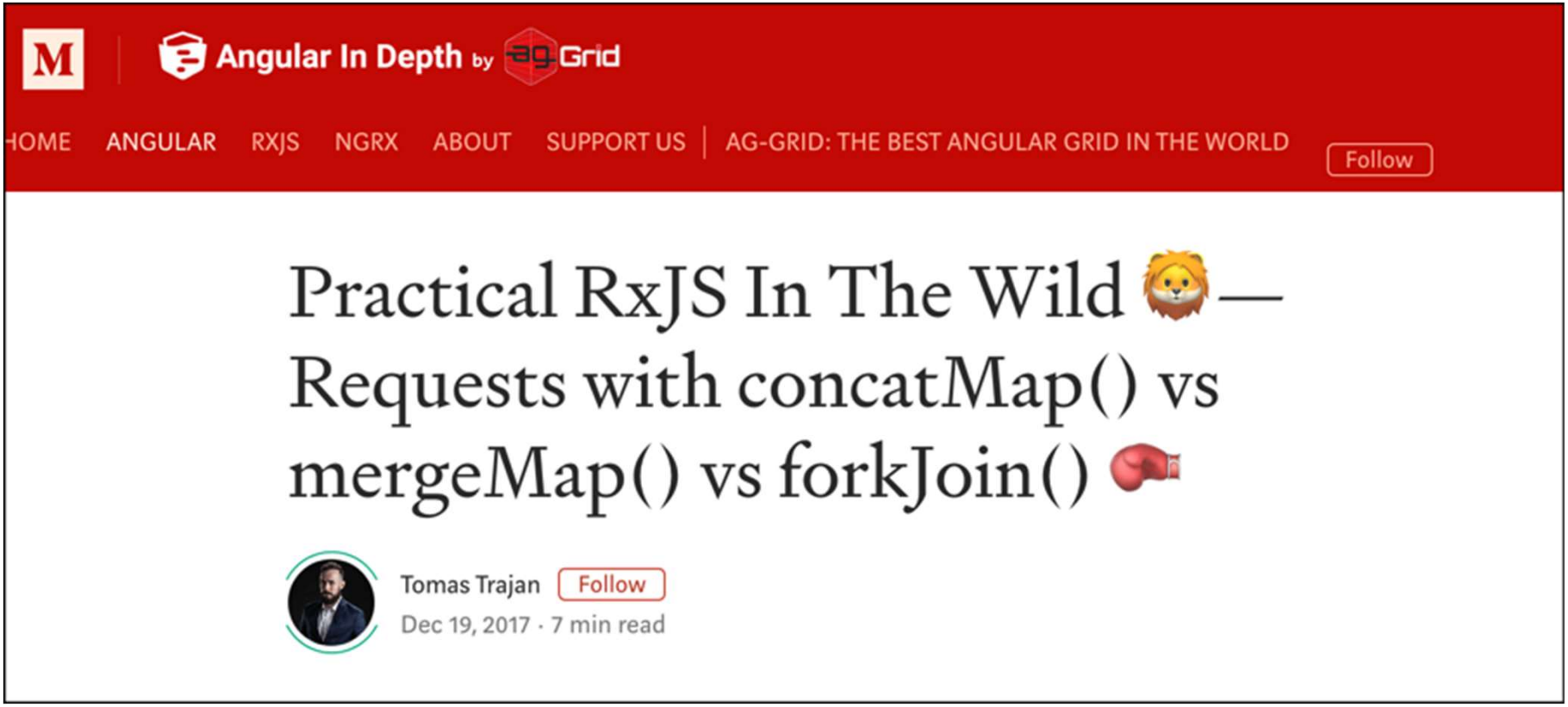
Exercise….



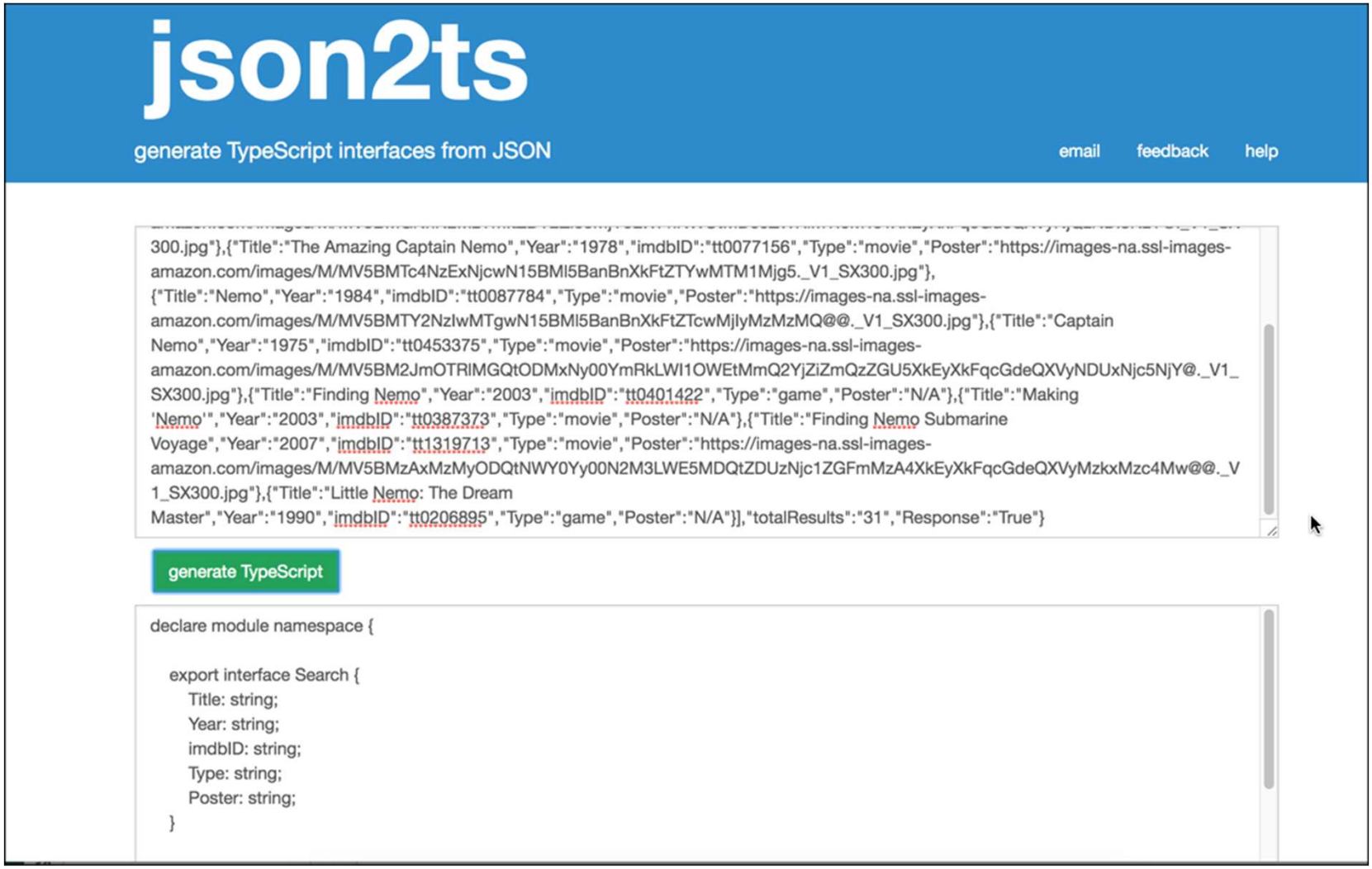
More info on observables

Fetching a collection of observables

https://blog.angularindepth.com/practical-rxjs-in-the-wild-requests-with-concatmap-vs-mergemap-vs-forkjoin-11e5b2efe293

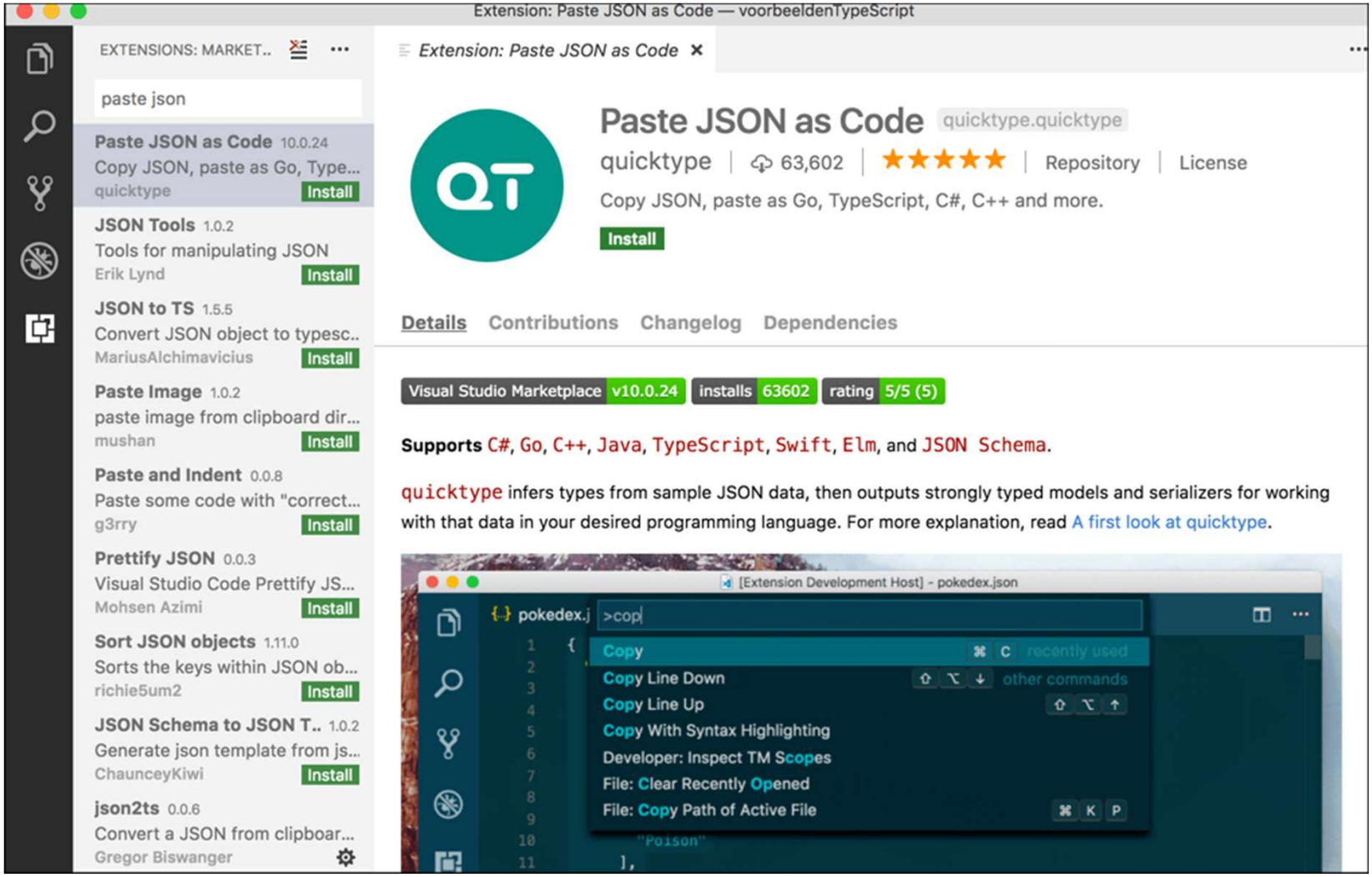


Online JSON to TypeScript converter



http://json2ts.com/

In VS Code? Use this extension!



https://marketplace.visualstudio.com/items?itemName=quicktype.quicktype

Data Mocking - Mockaroo

mockaroo realistic data generator

PRICING SIGN IN

Need some mock data to test your app?

Mockaroo lets you generate up to 1,000 rows of realistic test data in CSV, JSON, SQL, and Excel formats.

Need more data? Plans start at just $50/year.

Field Name	Type	Options
id	Row Number	blank: 0 % fx
first_name	First Name	blank: 0 % fx
last_name	Last Name	blank: 0 % fx
email	Email Address	blank: 0 % fx
gender	Gender	blank: 0 % fx
ip_address	IP Address v4	blank: 0 % fx

Add another field

Rows: 1000 Format: CSV Line Ending: Unix (LF) Include: header BOM

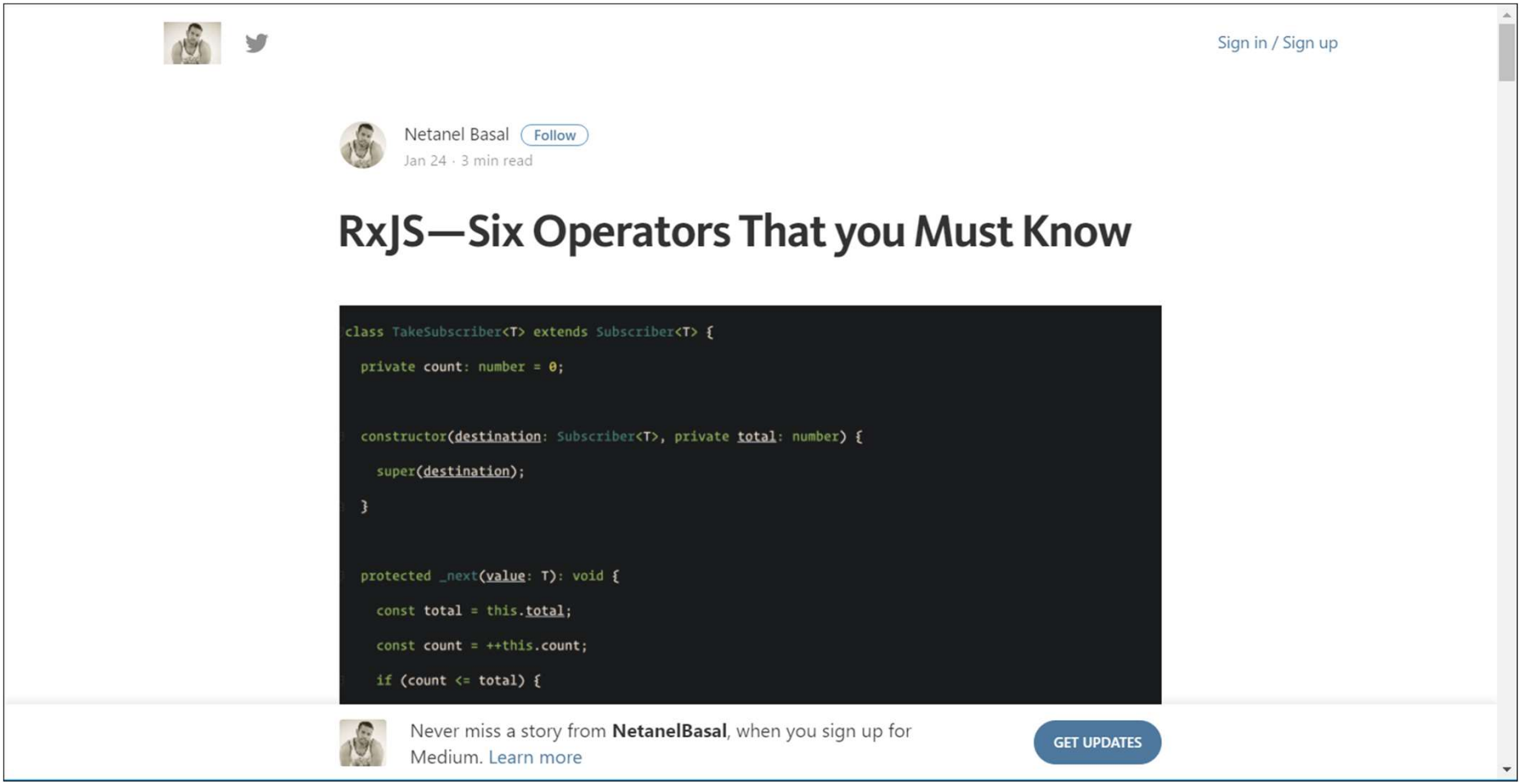
Download Data Preview More Want to save this for later? Sign up for free.

http://mockaroo.com/

Useful operators

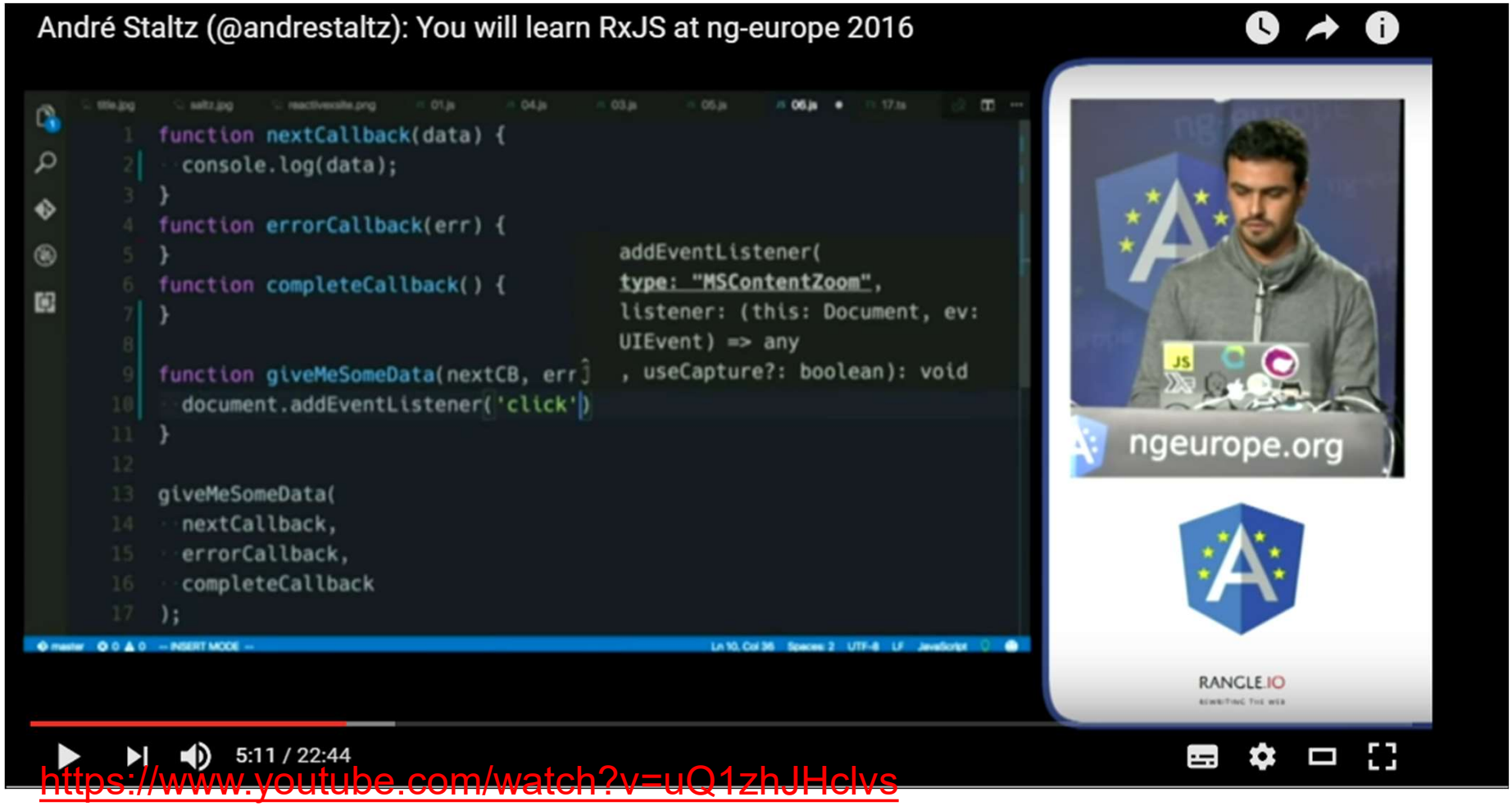
- RxJS operators are (mostly) just like Array operators
- Perform actions on a stream of objects
- Grouped by subject
  - Creation operators
  - Transforming
  - Filtering
  - Combining
  - Error Handling
  - Conditional and Boolean
  - Mathematical
  - …

6 Operators you must know



https://netbasal.com/rxjs-six-operators-that-you-must-know-5ed3b6e238a0#.11of73aox

Creating Observables from scratch - André Staltz



https://www.youtube.com/watch?v=uQ1zhJHclvs

GitHub Gist Search... All gists GitHub New gist

staltz / introrx.md
Last active an hour ago

Star 10,812 Fork 1203

Code Revisions 259 Stars 10812 Forks 1203

Embed <script src="https://gist. Download ZIP

The introduction to Reactive Programming you've been missing

introrx.md Raw

The introduction to Reactive Programming you've been missing

(by @andrestaltz)

This tutorial as a series of videos

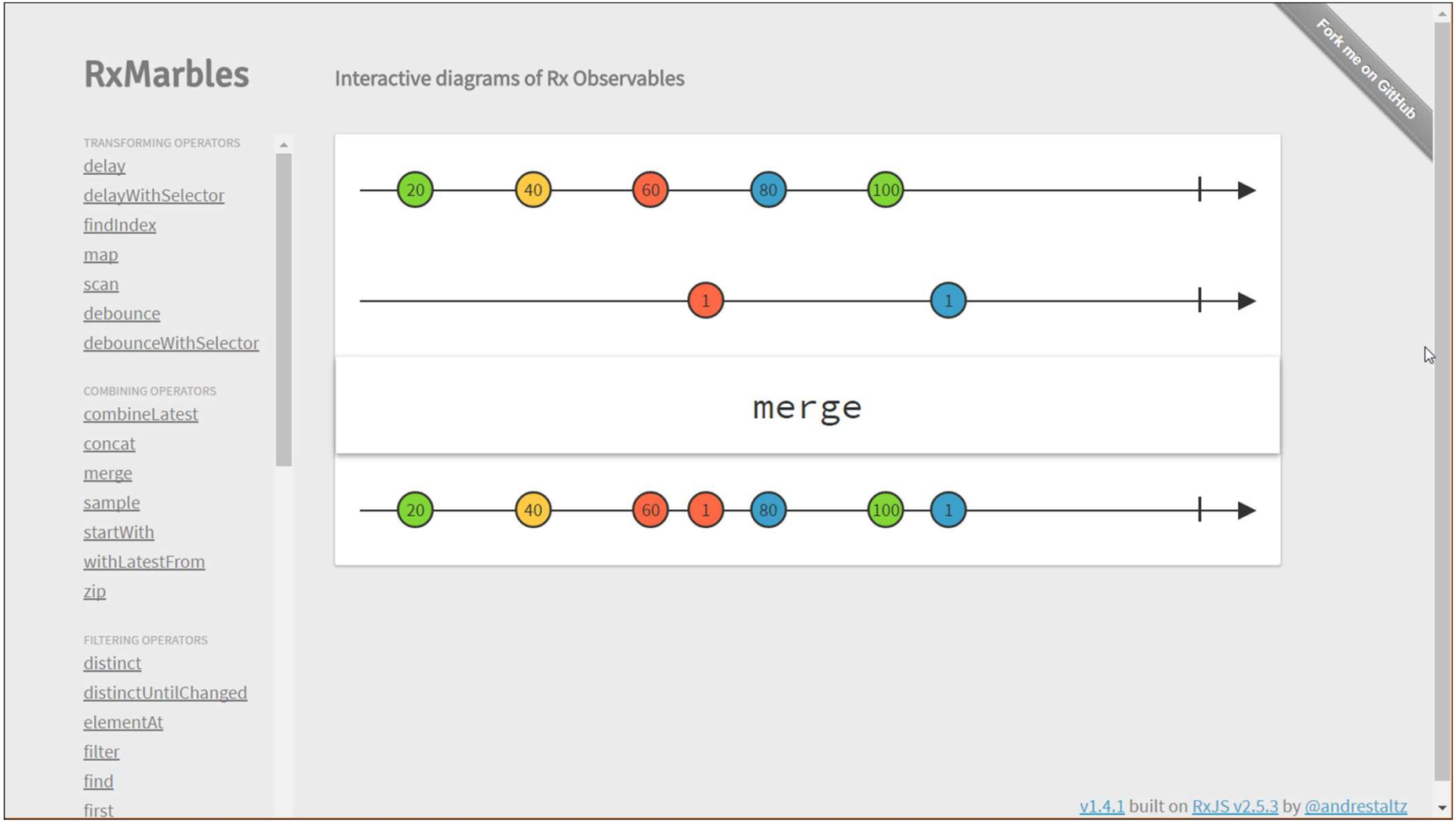
If you prefer to watch video tutorials with live-coding, then check out this series I recorded with the same contents as in this article: Egghead.io - Introduction to Reactive Programming.

So you're curious in learning this new thing called Reactive Programming, particularly its variant comprising of Rx, Bacon.js, RAC, and others.

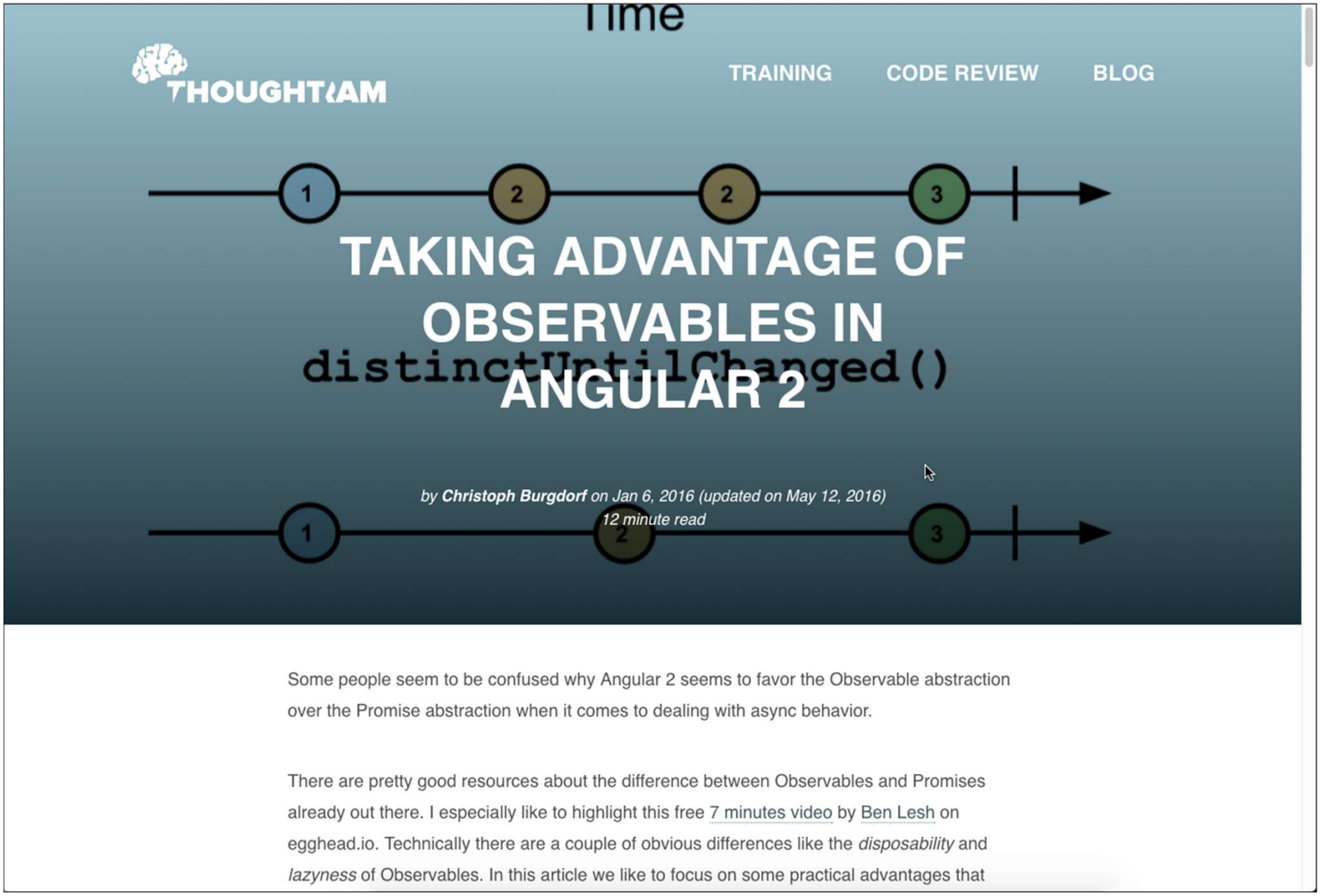
Learning it is hard, even harder by the lack of good material. When I started, I tried looking for tutorials. I found only a handful of practical guides, but they just scratched the surface and never tackled the challenge of building the whole architecture

https://gist.github.com/staltz/868e7e9bc2a7b8c1f754

RxMarbles



http://rxmarbles.com/



http://blog.thoughtram.io/angular/2016/01/06/taking-advantage-of-observables-in-angular2.html