IVAN JAKUS JAVACRO 2017

# WHAT ARE SOME TOOLS EVERY ANGULAR DEVELOPER SHOULD HAVE IN THEIR TOOLBOX?

ijakus@croz.net

## ABOUT ME

- Software developer @ CROZ
- 6+ years experience as a full stack developer
- Worked mainly with ExtJS before

- Angular since 2.0.0-alpha.39 (December 2015)
- Custom framework development + quite a few business logic modules

# WHAT ARE WE TALKING ABOUT?

- Angular
- Functional Reactive Programming
- Event sourcing
- Components / Widgets

# WHAT IS ANGULAR?

- development platform for building mobile and desktop web applications
- component-based web development
- allows developer to focus on actual work
- performance

---

- steep learning curve
- high-speed train

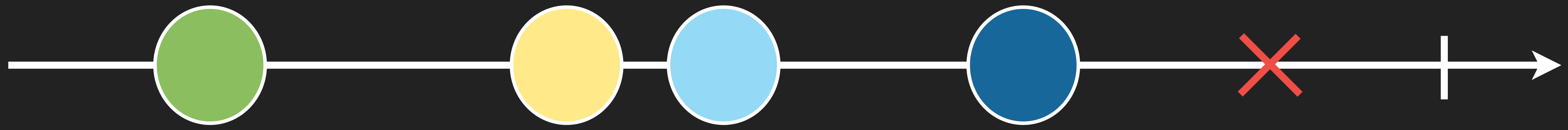
# HOW TO ANGULAR

```
npm install -g @angular/cli
ng new demo_project
cd demo_project
ng serve
```

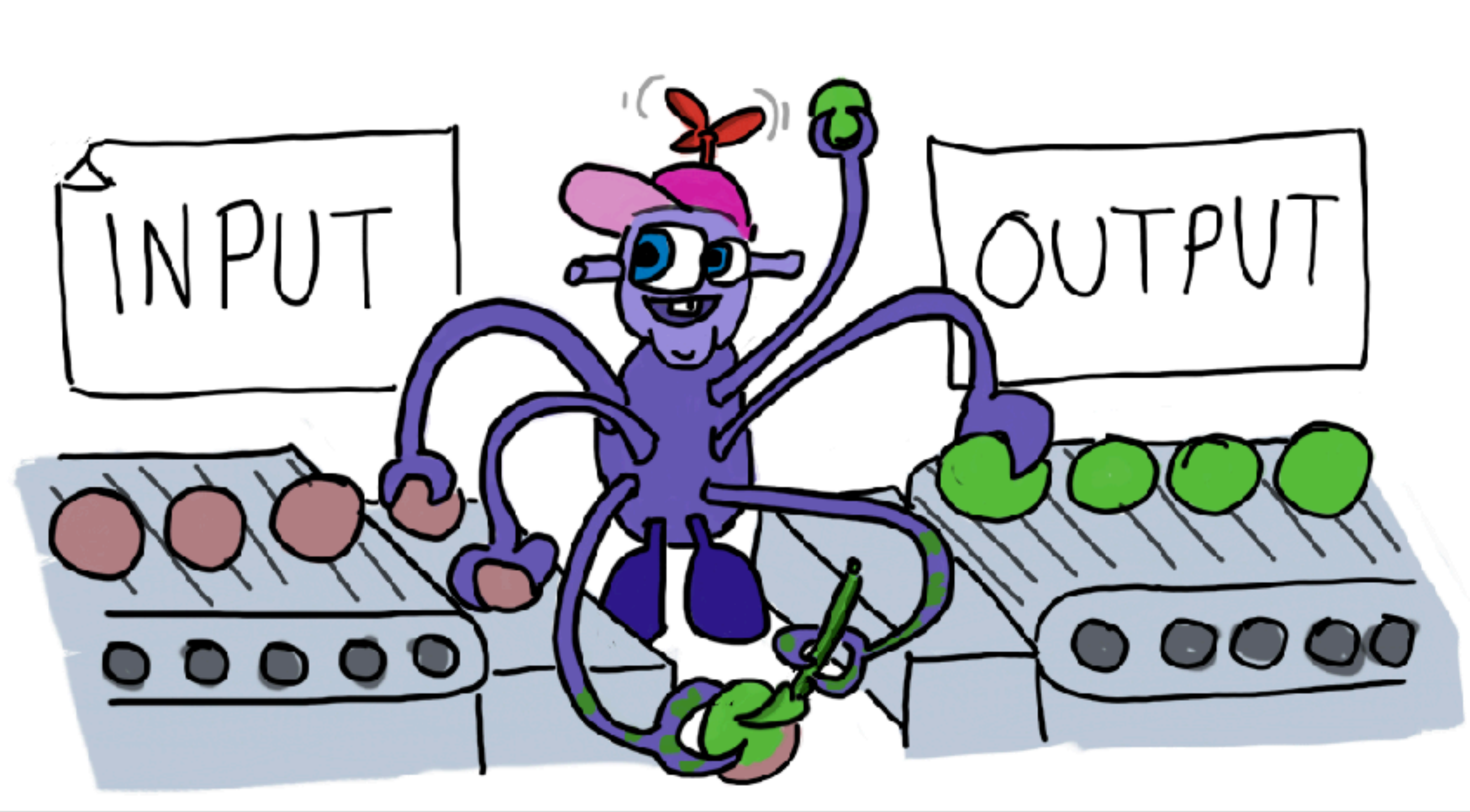
> FUNCTIONAL REACTIVE PROGRAMMING IS A PROGRAMMING PARADIGM FOR REACTIVE PROGRAMMING USING THE BUILDING BLOCKS OF FUNCTIONAL PROGRAMMING.

The Internets

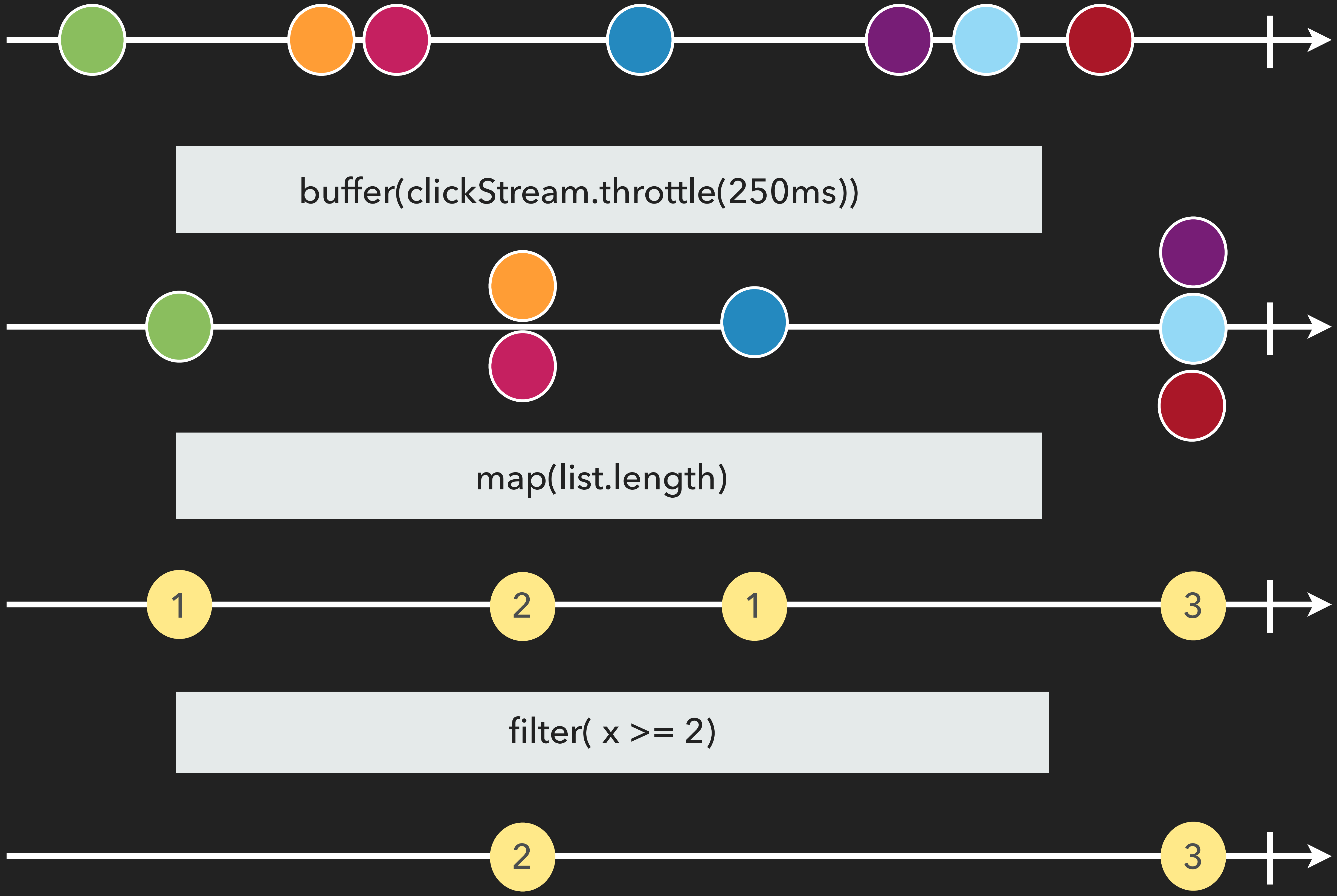
# REACTIVE PROGRAMMING



# FUNCTIONAL PROGRAMMING



# FUNCTIONAL REACTIVE PROGRAMMING



buffer(clickStream.throttle(250ms))

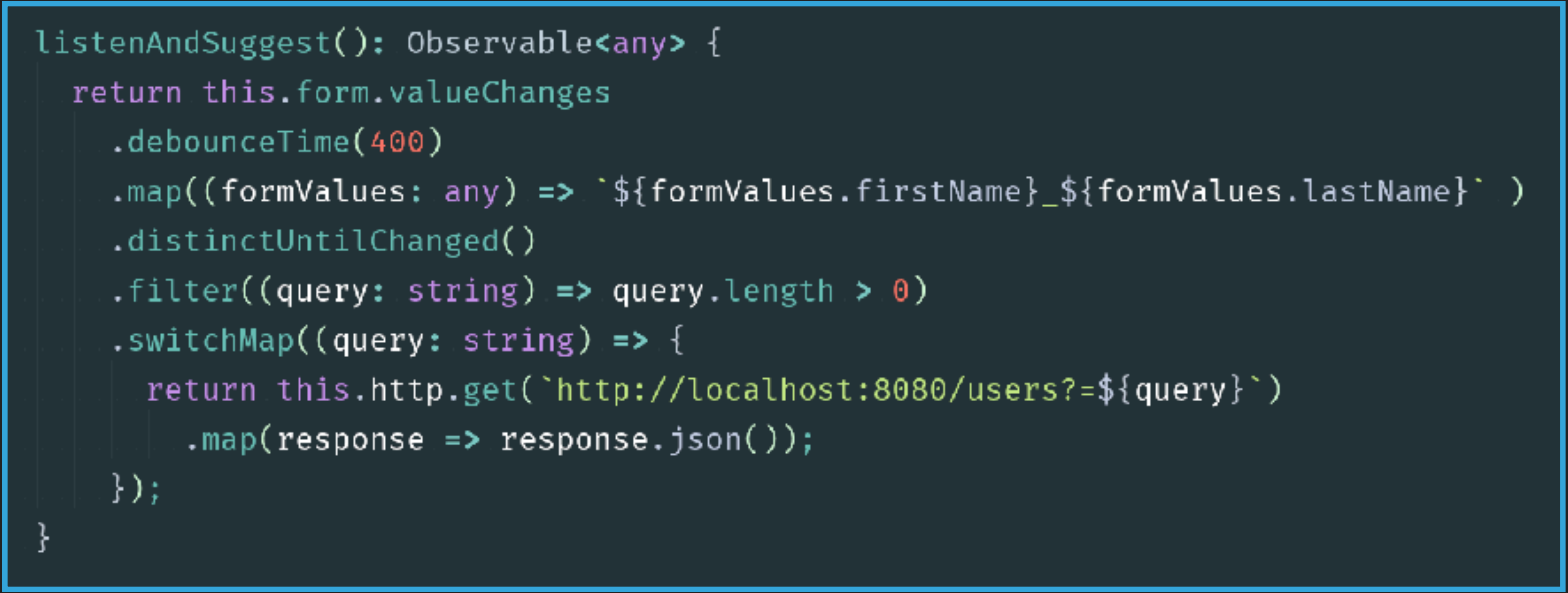
map(list.length)

filter( x >= 2)

# WHAT IS RXJS AND WHY DO I CARE?

- RxJS = ReactiveX library for JavaScript
- RxJS is one of building blocks of Angular
  - HTTP requests, Forms, Router, …
  - Built-in async pipe
- No need to break stream from server call to rendering
- Be modern and cool but with a painful start and loads of reading

# HOW DOES IT LOOK LIKE IN CODE?



```
listenAndSuggest(): Observable<any> {
  return this.form.valueChanges
    .debounceTime(400)
    .map((formValues: any) => `${formValues.firstName}_${formValues.lastName}` )
    .distinctUntilChanged()
    .filter((query: string) => query.length > 0)
    .switchMap((query: string) => {
      return this.http.get(`http://localhost:8080/users?=${query}`)
        .map(response => response.json());
    });
}
```

EVENT SOURCING

> EVENT SOURCING ENSURES THAT ALL CHANGES TO APPLICATION STATE ARE STORED AS A SEQUENCE OF EVENTS.

Martin Fowler

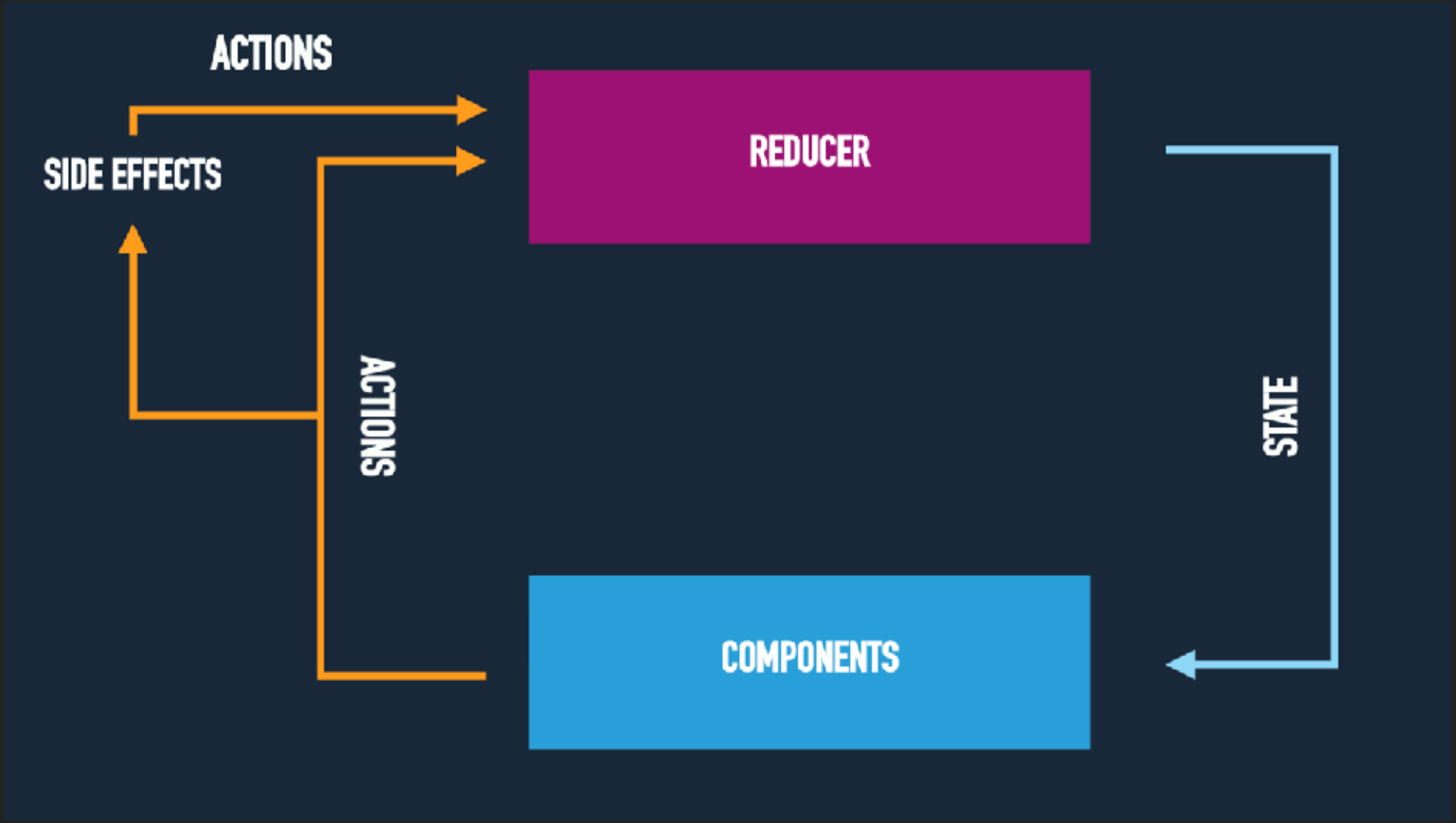
# WHAT IS NGRX

- Set of small reactive libraries
- Inspired by Redux and implemented with RxJS

## @ngrx/store

- Controlled state container
- State is a single immutable data structure
- Actions describe state changes

# EVENT SOURCING



## WHY USE NGRX

- Centralised Immutable State
- Performance benefits
- Testability

# NGRX LIBRARIES

- @ngrx/store
- @ngrx/effects
- @ngrx/router-store
- @ngrx/store-devtools
- @ngrx/db

# DO I NEED WIDGET SET?

- Do I have enough knowledge to build it myself?
- Do I have time to build it myself?
- Do I have time to compare all different versions of button / input / data table?
- Will selected widget become dead before production?
- Maybe I can go without widgets?

# ANGULAR MATERIAL

- Set of Material design components
- Seamlessly integrates with Angular
- Easy to use with straightforward API
- Customisable using themes
- Web & mobile + support for touch and screen readers
- Well tested with a big community and Google support

# TERADATA COVALENT

- UI platform on top of Material
- Follows principle of Angular Material
- Set of well tested, documented and configurable components
  - Layouts, Data table (atomic), Stepper, Expansion panel, File Input, Notifications, Chips, Pipes, Syntax highlighting, …
- NGX-Charts & NGX-Translate
- Mock API

# WHAT'S ELSE WORTH GOOGLING

- Augury
- @angular/mobile-toolkit
- @angular/language-service
- Codelyzer
- Yarn

# FINAL THOUGHTS

- Frontend development isn’t boring
- Event sourcing and reactive programming are not gimmicks but core building blocks in Angular ecosystem
- Try implementing something you usually only see at conferences
- It’s challenging, but definitely fun

THANK YOU FOR YOUR ATTENTION

# QUESTIONS