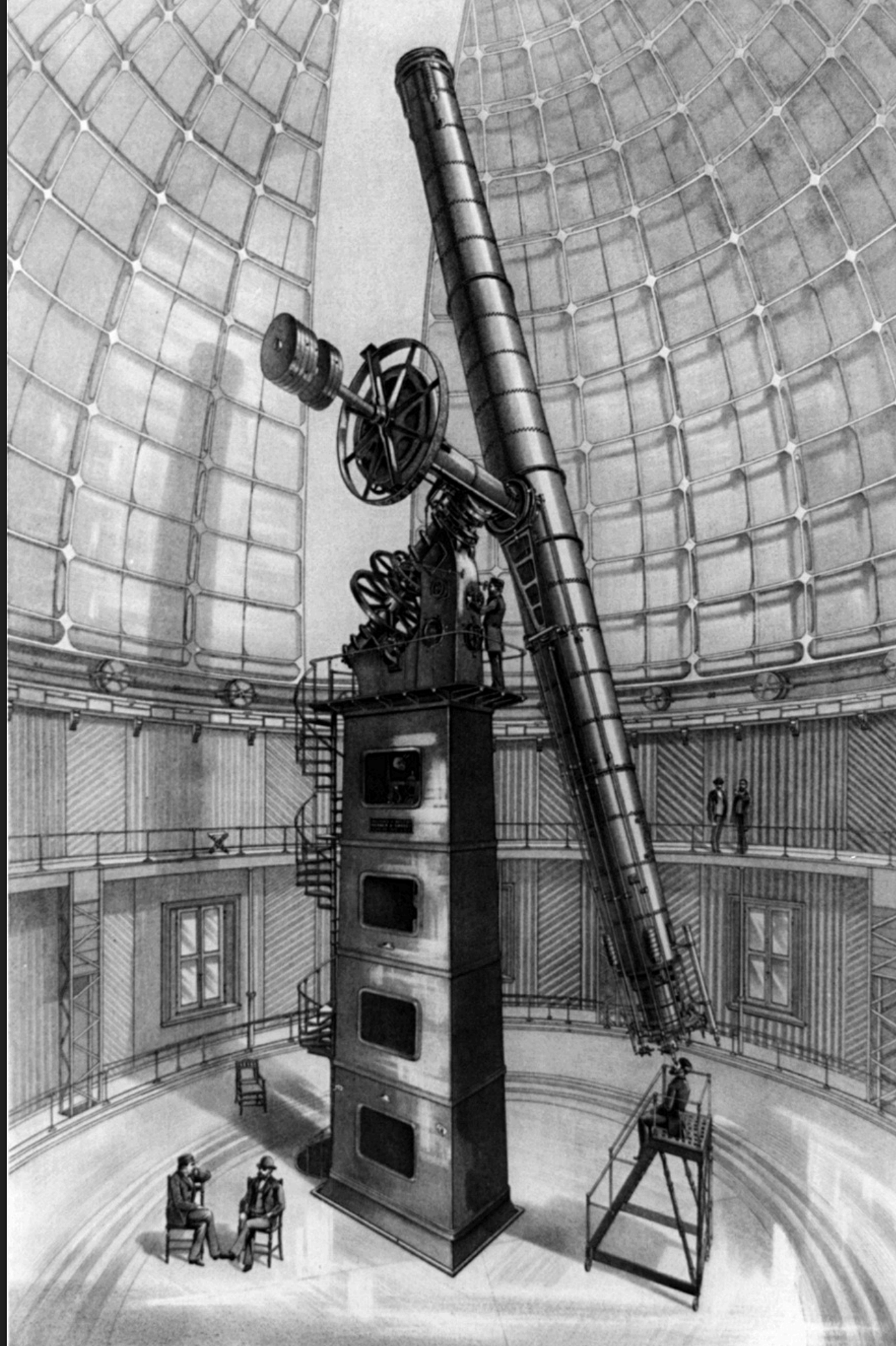
An Angular Point of Vue



Angular developers looking at all the Vue.js GitHub stars

```
nico:~ nico$ npx nicokoenig
npx: Installierte 39 in 1.921s

   Nico Koenig

        Work:  Freelance developer, consultant and trainer
     Twitter:  https://twitter.com/theNicoKoenig
      GitHub:  https://github.com/nicokoenig
    Codepen:  https://codepen.io/nicokoenig

         Web:  https://koenig.codes

        Card:  npx nicokoenig
```

· SINCE 2016 ·

Krautipsum

TRADITIONAL GERMAN FILLER TEXT

Aufgemerkt! Lude und Panzerkampfwagen ergötzen adrett Früchtchen. Der fidel Müßiggang katzbuckeln. Das geflissentlich Grüne Minna ergötzen. Flausen und Steckenpferd erquicken töricht Muckefuck. Das Sättigungsbeilage flanieren das feist Franzosenkrankheit. Der ausgemergelt Bürgermeisterstück

Angular, Vue and myself 👍

- Angular
  - Since December 2015
  - Projects from 5 to 40 developers
  - Workshops & Bootcamps
- Vue
  - Since May 2018
  - Project with 6 developers

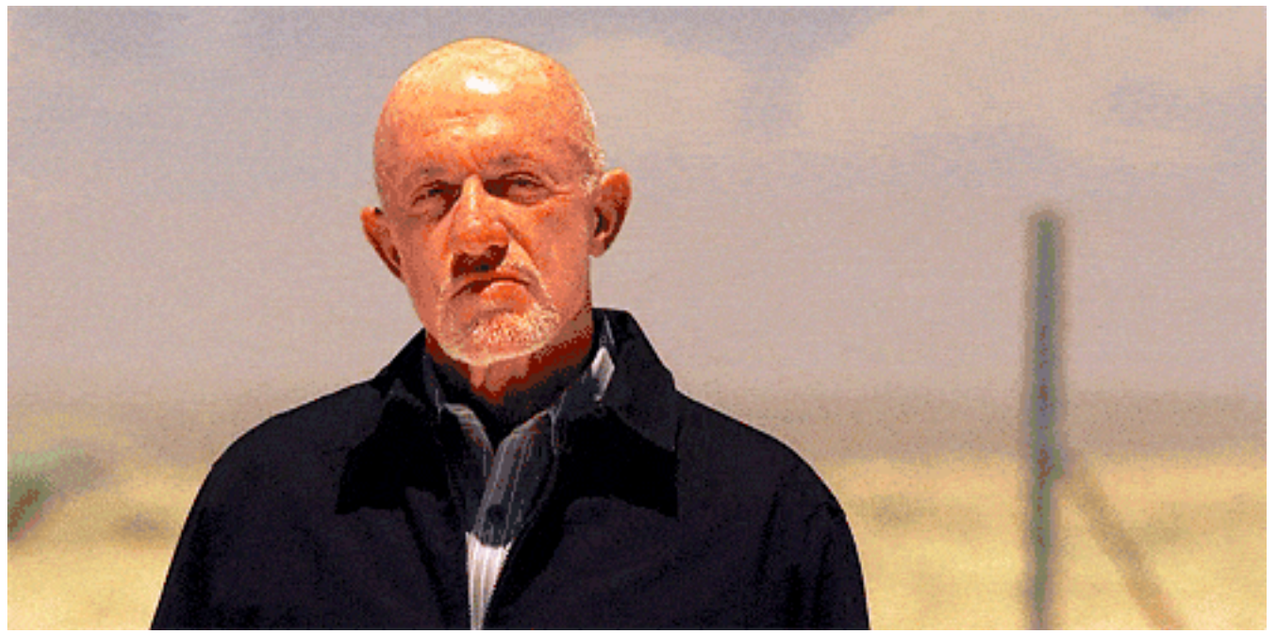
Ready to rumble? 🙄

🔗 http://www.hearthcards.net/

Ready to rumble? 🙄

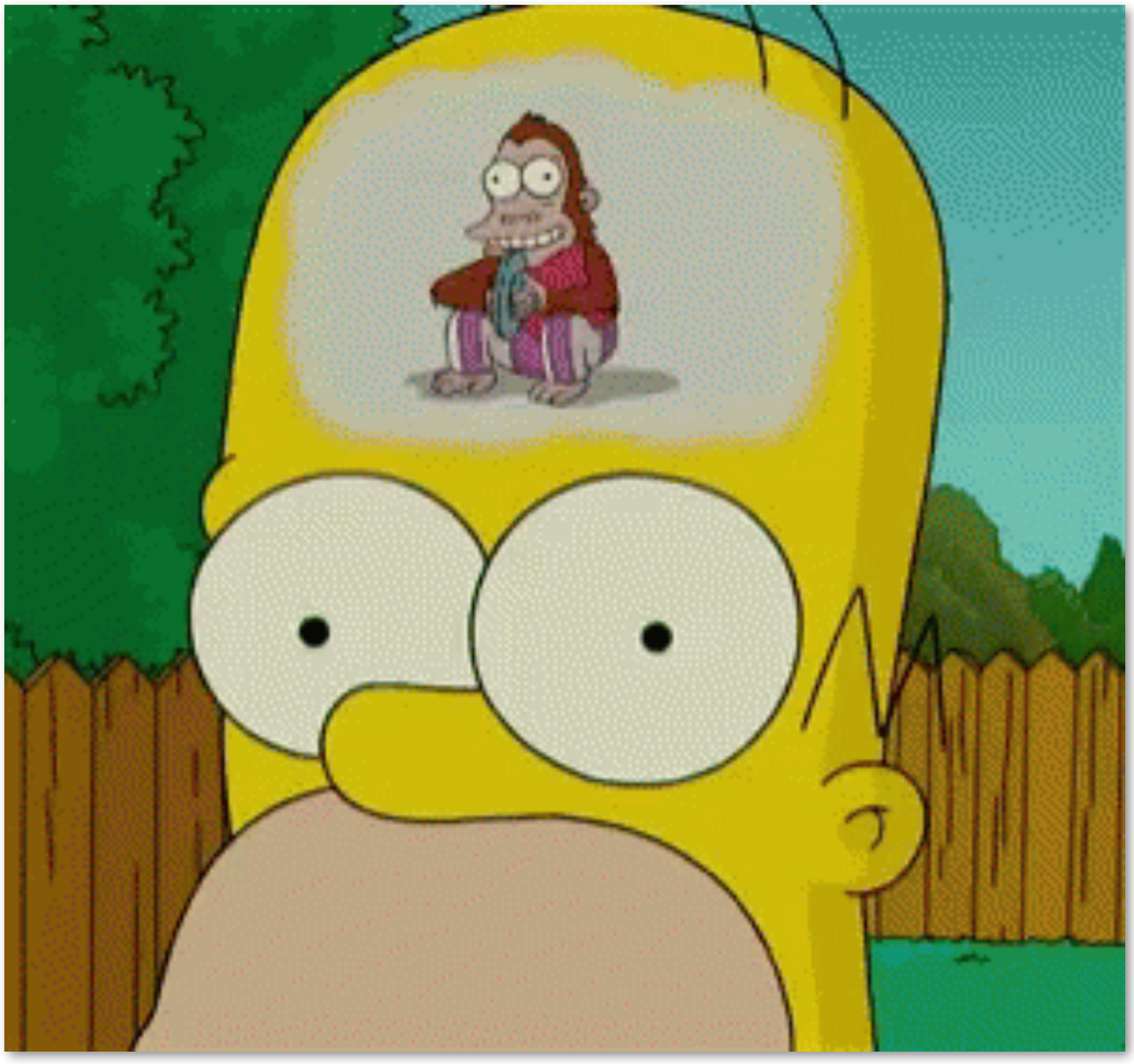
🔗 http://www.hearthcards.net/

Ready to rumble? 🙄

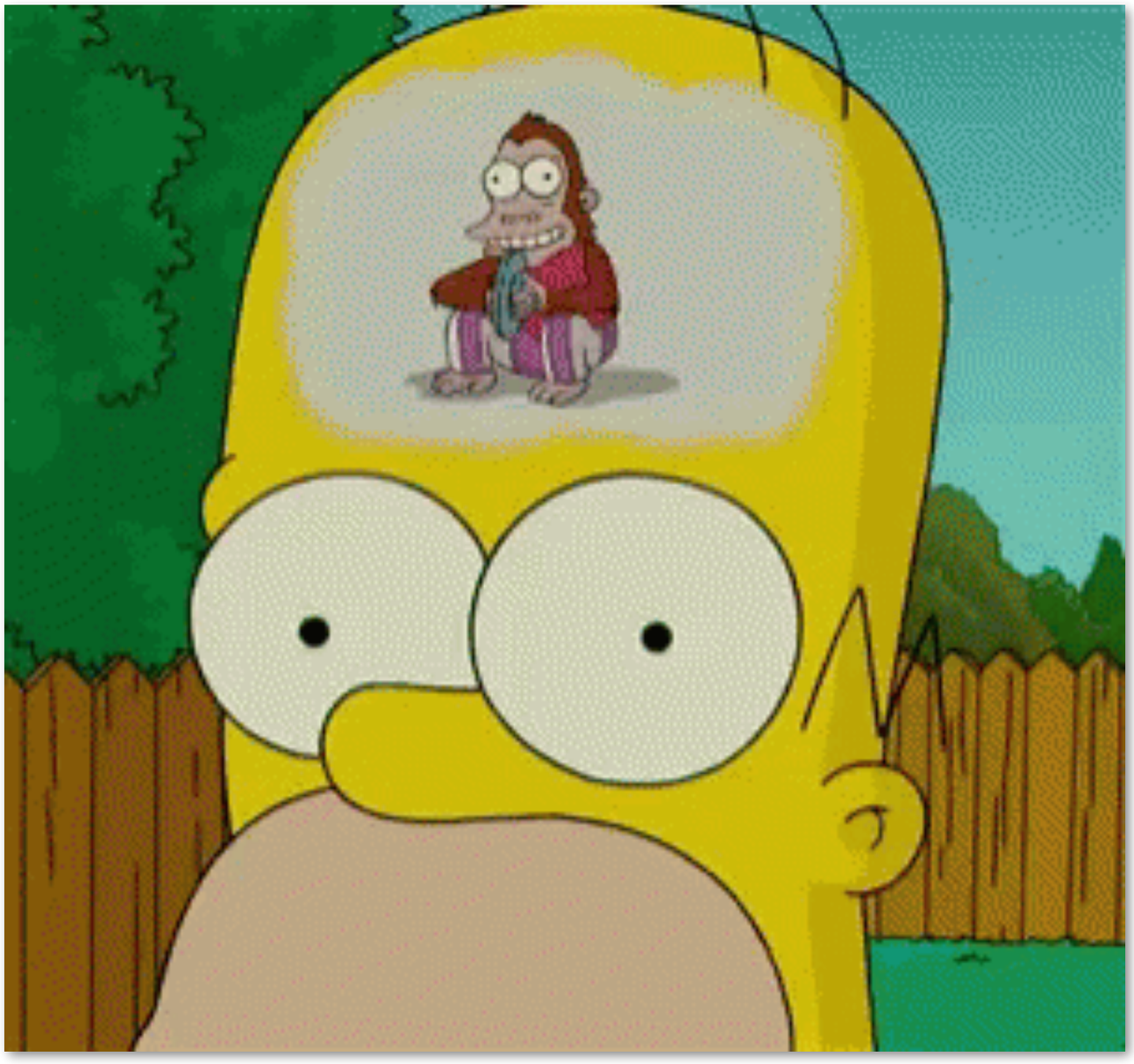


🔗 http://www.hearthcards.net/

Think Nico, think…



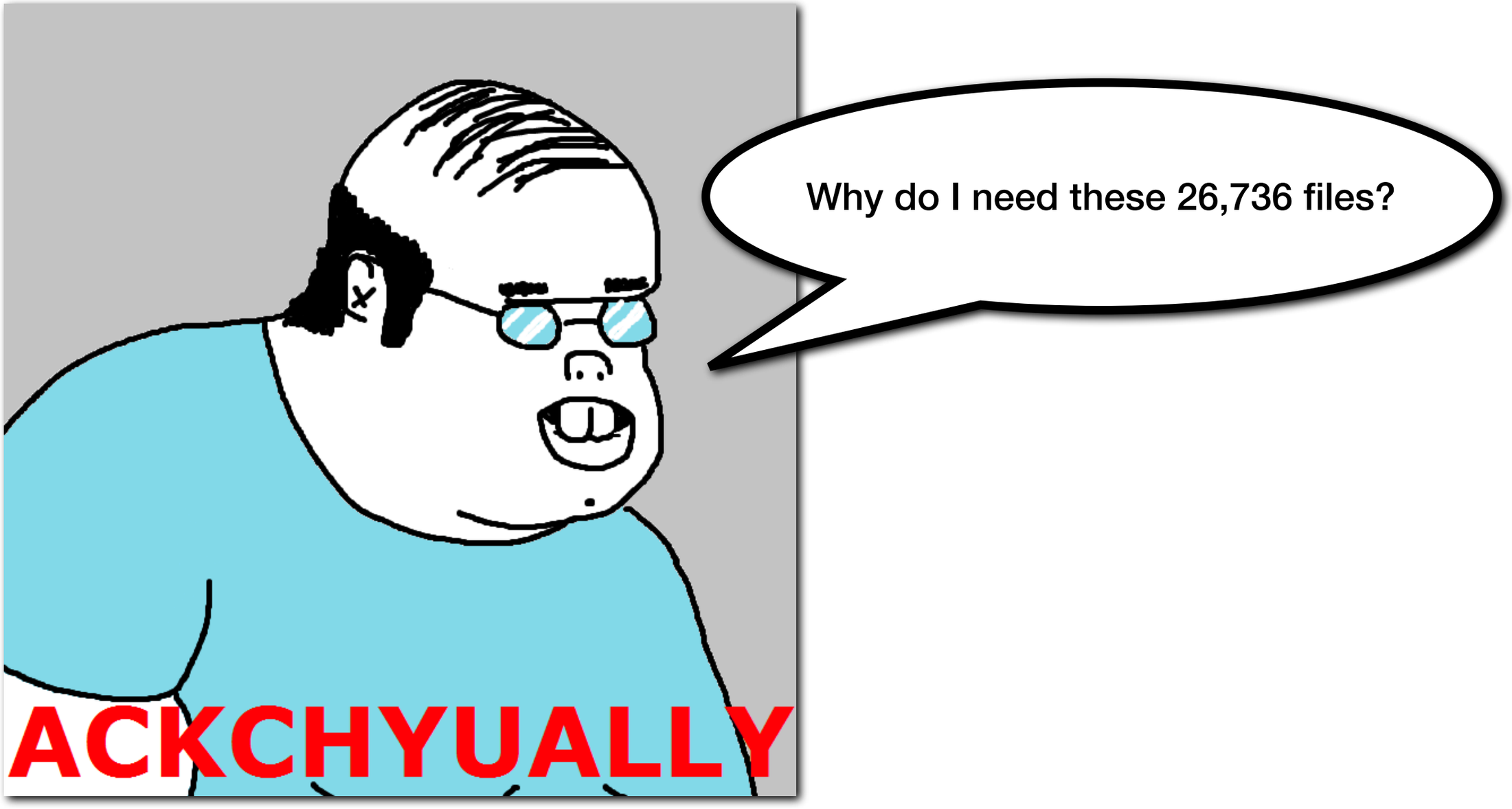
Think Nico, think…



Frameworks FTW 🚀

Evolution of front-end development 🦐🦎🦖🦄

Frameworks FTW 🚀



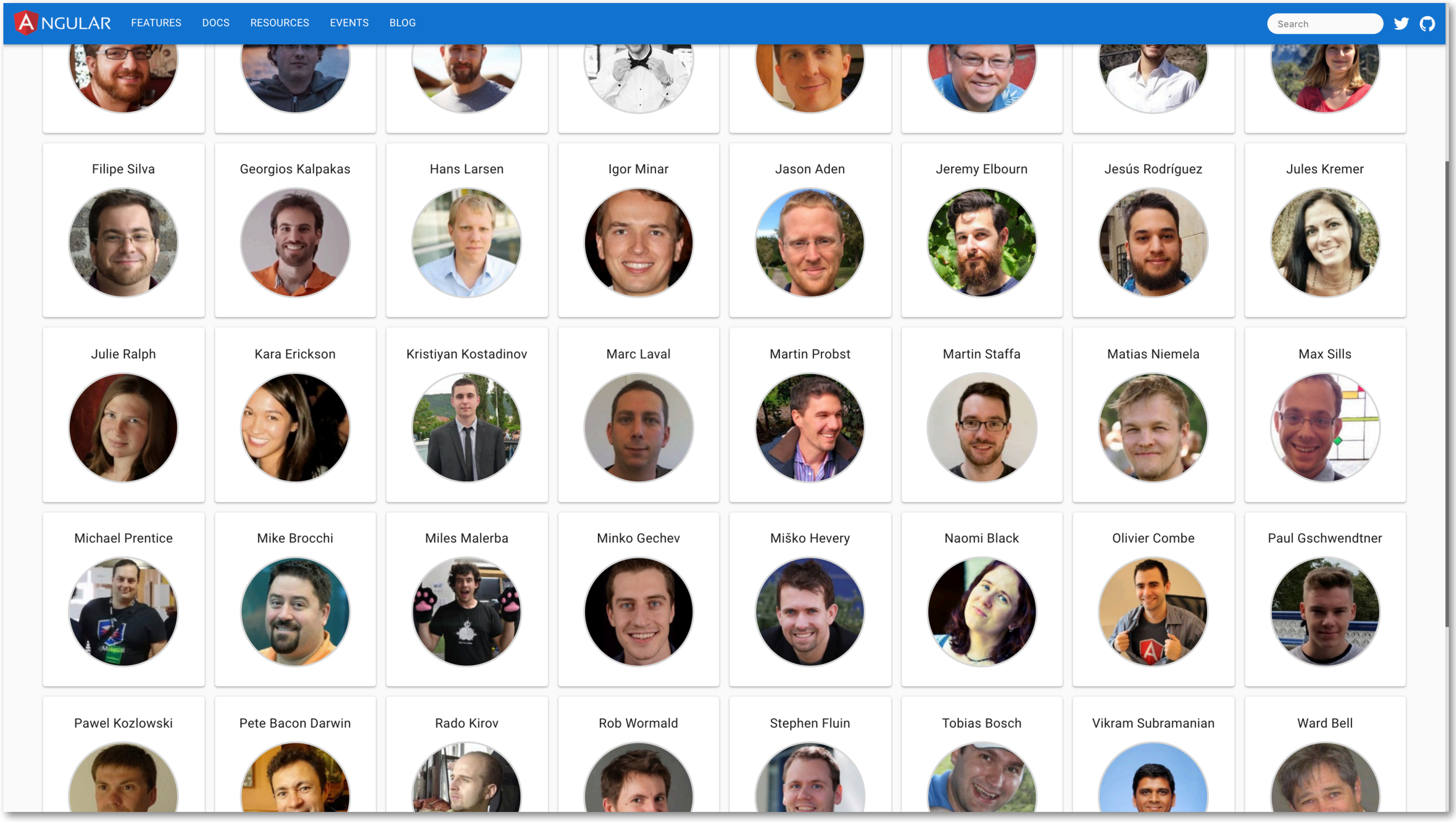
When the magic happened

Evan You publishing the first version of Vue.js on GitHub

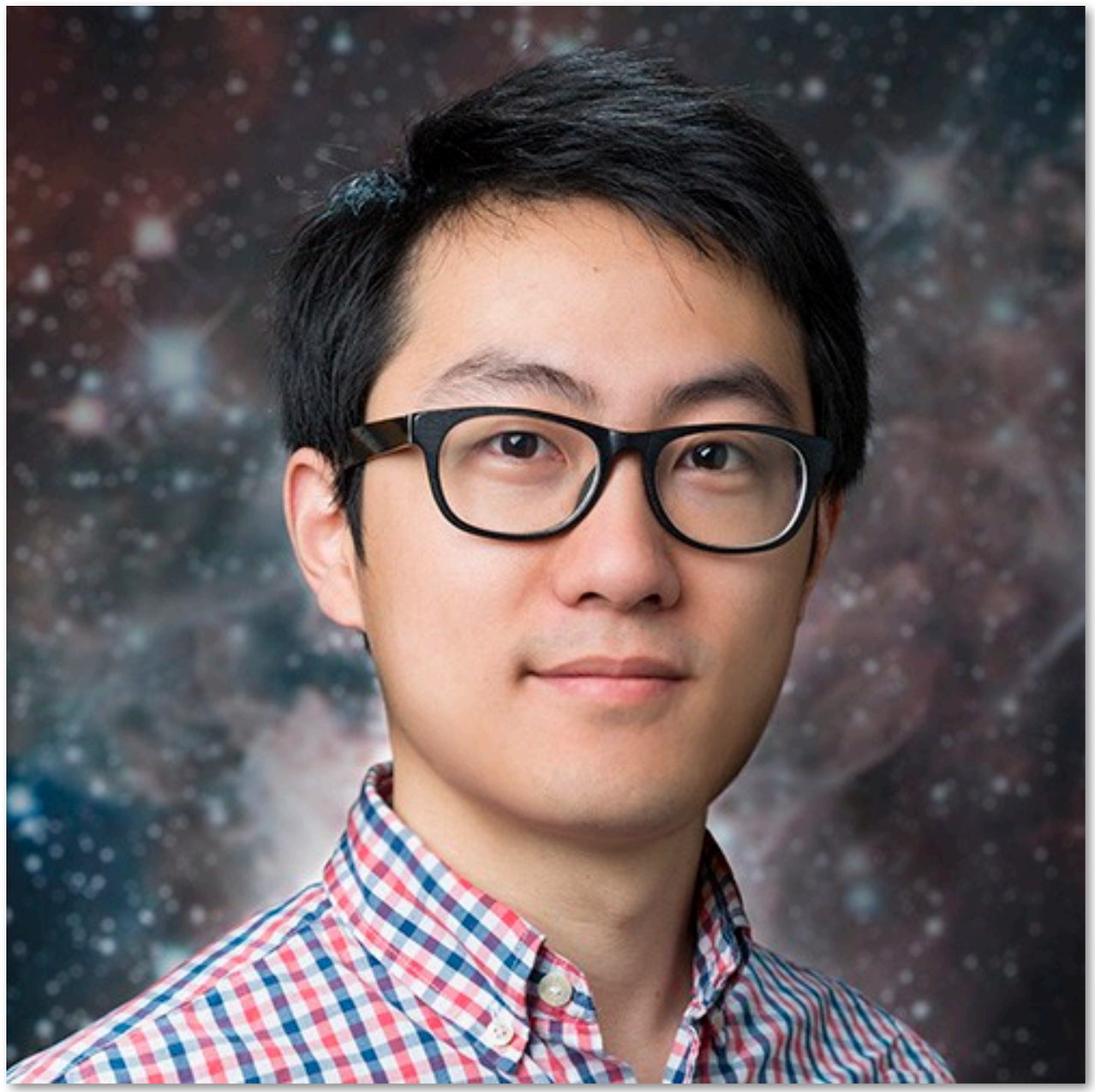
A brief history of angular

- AngularJS first release 2010
- Angular 2.0.0 September 2016
  Angular 4.0.0 March 2017
  Angular 5.0.0 November 2017
  Angular 6.0.0 May 2018
  Angular 7.0.0 October 2018
- Major release every 6 months

Meet the team



Evan You



A brief history of Vue

- Evan You used AngularJS at Google Creative Labs, but wanted to build something more lightweight

- Evan You built Vue.js in his free time

- Late 2013 - Private development of Vue.js
  Feb 2014 - Public Release als Open Source
  Oct 2015 - 1.0
  Feb 2016 - Full-time
  Sep 2016 - Version 2.0

- Vue 3.0 announced 2019 (maybe)

If a T-Rex eats Evan You, will Vue survive?

oneminutejs Follow

Apr 29, 2018 · 6 min read

🔗 https://medium.com/@oneminutejs/if-a-t-rex-eats-evan-you-will-vue-survive-d496a4b99872

vuejs / vue

Watch 5,773 | Star 126,887 | Fork 18,090

Code | Issues 169 | Pull requests 53 | Projects 2 | Wiki | Insights

- Pulse
- **Contributors**
- Community
- Commits
- Code frequency
- Dependency graph
- Network
- Forks

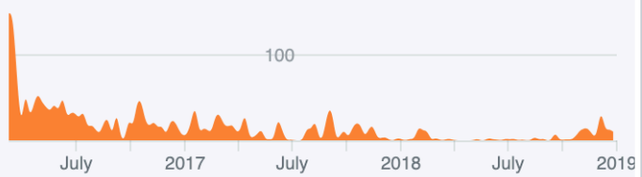
Apr 10, 2016 – Feb 2, 2019

Contributions: **Commits**

Contributions to dev, excluding merge commits

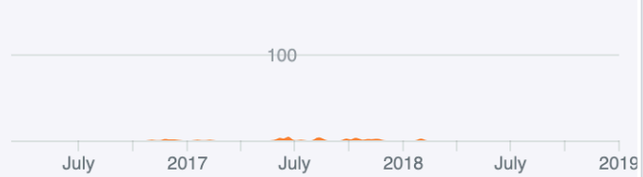
#1 yyx990803

2,210 commits 513,527 ++ 370,820 --



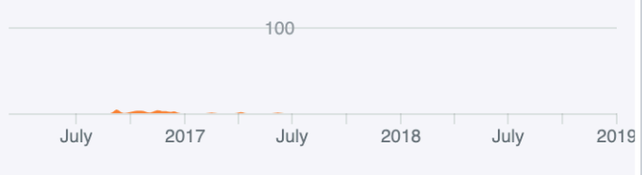
#2 Hanks10100

47 commits 7,245 ++ 3,540 --



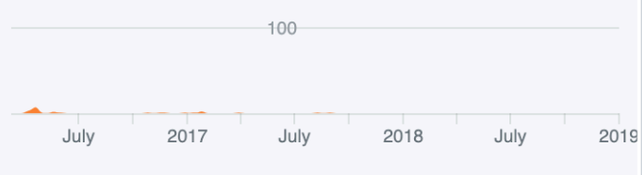
#3 defcc

37 commits 1,939 ++ 344 --



#4 kazupon

32 commits 3,663 ++ 130 --



#5 posva

#6 HerringtonDarkholme

🔗 https://github.com/vuejs/vue/graphs/contributors

Quote from Evan You

“I personally don’t want to be responsible for everything. I want the project to be able to evolve, even if someday I stop working on it. So part of my work right now is to make that happen”

🔗 https://blog.hackages.io/https-blog-hackages-io-evanyoubhack2017-cc5559806157

Meet the team

Vue.js

Learn · Ecosystem · Team · Support Vue · Translations

Meet the Team

Evan You

CORE FOCUS vuejs/* · vuejs-templates/*

Creator @ Vue.js

Jersey City, NJ, USA

中文 · English

patreon.com/evanyou

Jinjiang

CORE FOCUS cn.vuejs.org

ECOSYSTEM apache/incubator-weex

Alibaba

Hangzhou, China

中文 · English

Pine Wu

CORE FOCUS vetur

Engineer on VSCode @ Microsoft

中文 · English · 日本語

Damian Dulisz

CORE FOCUS news.vuejs.org

ECOSYSTEM shentao/vue-multiselect · shentao/vue-global-events

Consultant

Wrocław, Poland

Polski · English

Michał Sajnóg

CORE FOCUS eslint-plugin-vue · vue-devtools

ECOSYSTEM vue-computed-helpers · vue-content-placeholders

Senior Frontend Developer / Team Leader @ Netguru

Poznań, Poland

Polski · English

Linusborg

CORE FOCUS vuejs/* · vuejs-templates/* · vue-touch

ECOSYSTEM portal-vue

Mannheim, Germany

Deutsch · English

forum.vuejs.org

Chris Fritz

CORE FOCUS vuejs.org · vue-migration-helper

ECOSYSTEM vue-2.0-simple-routing-example · vue-ssr-demo-simple

Educator & Consultant

Lansing, MI, USA

English · Deutsch

patreon.com/chrisvuefritz

Katashin

CORE FOCUS vuex · vue-class-component

ECOSYSTEM vue-designer

Software Engineer @ ClassDo

Singapore

日本語 · English

GU Yiling

CORE FOCUS vue · cn.vuejs.org

ECOSYSTEM Justineo/vue-awesome · ecomfe/vue-echarts · ecomfe/veui

Senior web developer @ Baidu, inc.

Shanghai, China

中文 · English

Phan An

ECOSYSTEM vuequery · vue-google-signin-button

Munich, Germany

Tiếng Việt · English

vi.vuejs.org · phanan.net

Eduardo

CORE FOCUS vuefire · vue-router

ECOSYSTEM vuexfire · vue-mdc · vue-motion

Lead Instructor @ IronHack

Paris, France

Español · Français · English

codementor.io/posva

Sodatea

CORE FOCUS vue-cli · vue-loader

Hangzhou, China

中文 · English

Edd Yerburgh

CORE FOCUS vue-test-utils

ECOSYSTEM avoriaz

Full Stack Developer

London, UK

English

eddyerburgh.me

Darek Gusto Wędrychowski

Kraków, Poland

Polski · English

Rahul Kadyan

CORE FOCUS rollup-plugin-vue · vue-issue-helper

ECOSYSTEM keynote · bootstrap-for-vue · vue-interop

Software Engineer @ Myntra

Bangalore, India

हिंदी · English

znck.me · codementor.io/znck

Sarah Drasner

ECOSYSTEM intro-to-vue · vue-vscode-snippets · vue-sublime-snippets · nuxt-type · animating-vue-workshop · cda-locale · vue-weather-notifier

Senior Cloud Developer Advocate @ Microsoft

Kazupon

CORE FOCUS vuejs.org · jp.vuejs.org

ECOSYSTEM vue-i18n · vue-cli-plugin-i18n · vue-i18n-loader · vue-i18n-extensions

CTO & Full Stack Developer

Tokyo, Japan

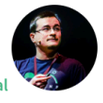
ULIVZ

CORE FOCUS vuepress

Senior Frontend Developer @ AntFinancial

Hangzhou, China

中文 · English



Guillaume Chau

CORE FOCUS vue-devtools · vue-cli · vue-curated

ECOSYSTEM vue-apollo · vue-meteor · vue-virtual-scroller · v-tooltip

Frontend Developer @ Livestorm

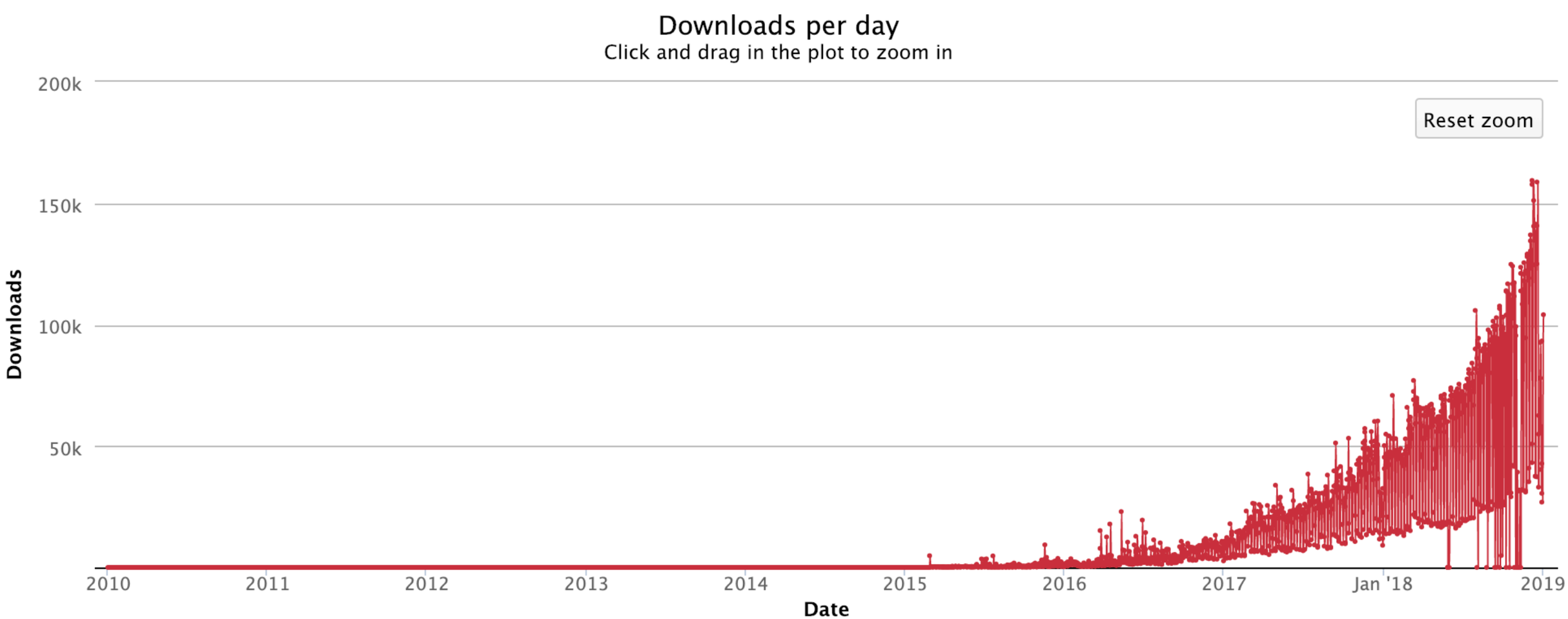
Lyon, France

Drop it like it’s hot? 🔥

Numbers

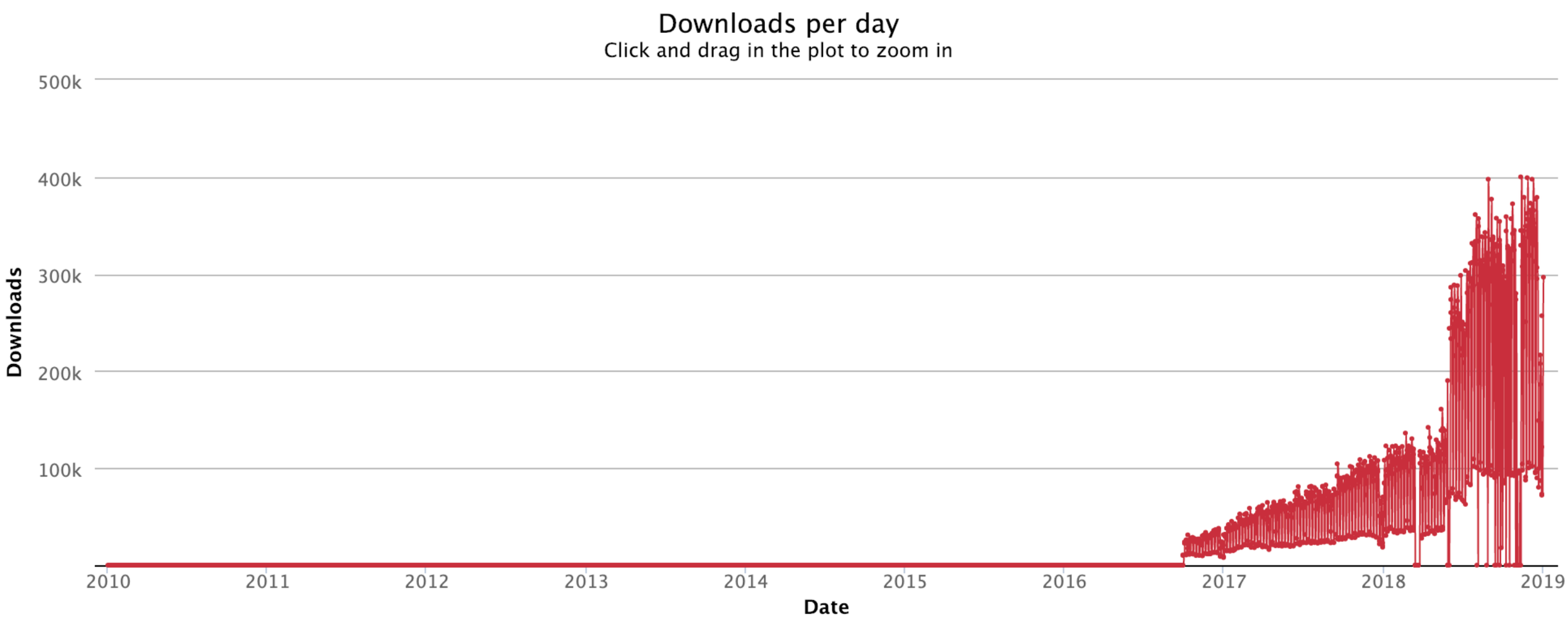
CTO checking out GitHub insights for front-end frameworks

NPM downloads of Vue: 34,492,921



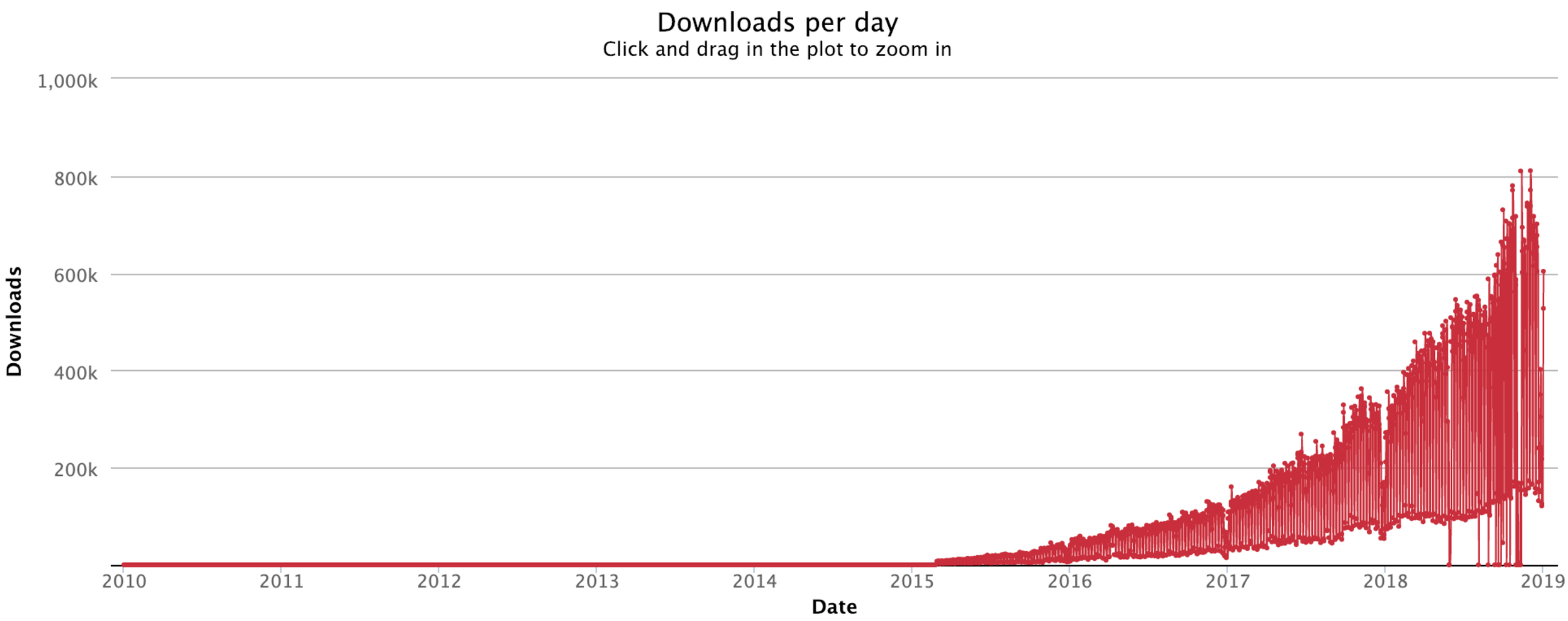
🔗 https://npm-stat.com/charts.html?package=vue&from=2010-01-03&to=2019-02-06

NPM downloads of Angular: 90,558,916



🔗 https://npm-stat.com/charts.html?package=%40angular%2Fcore&from=2010-01-03&to=2019-02-06

NPM downloads of React: 235,999,050



🔗 https://npm-stat.com/charts.html?package=react&from=2010-01-03&to=2019-02-06

NPM downloads of React: 235,999,050

🔗 https://npm-stat.com/charts.html?package=react&from=2010-01-03&to=2019-02-06

		
Year of release	2016	2014
NPM Downloads	90,000,000	34,000,000
Github Stars	44,000	125,000
Github Forks	11,000	18,000
Github Commits	12,800	2,876
Github Contributors	835	254

Project Setup

Junior developers assisting senior engineer at project setup

minimal setup

```html
<!DOCTYPE html>
<html>
<head>
    <script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
    <title>vuehoo!</title>
</head>
<body>
    <div id="app">{{ message }}</div>
    <script>
        var app = new Vue({
            el: '#app',
            data: { message: 'Ja Servus, Vue!' }
        })
    </script>
</body>
</html>
```

Vue CLI

Vue CLI 3

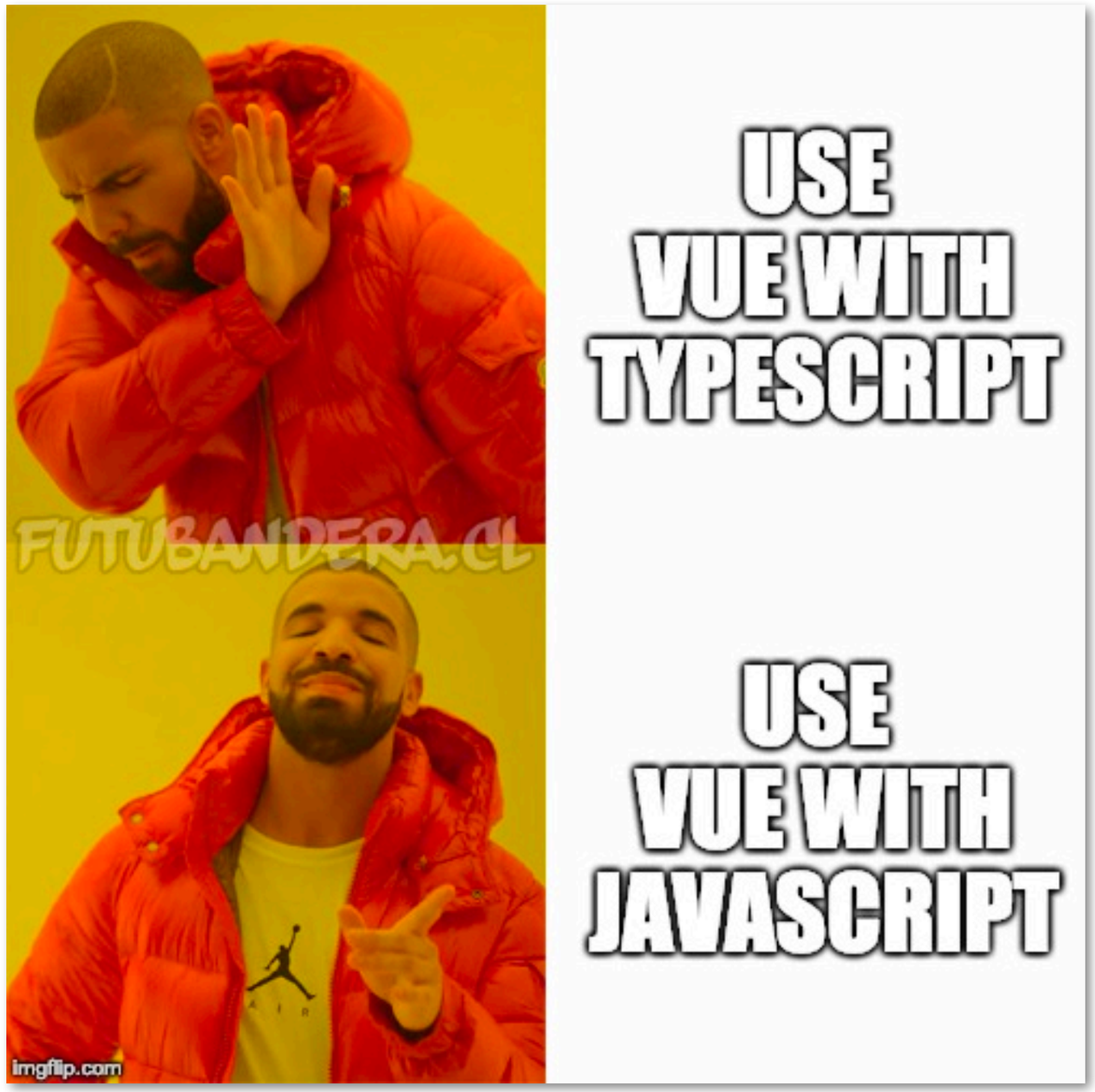
🛠️ Standard Tooling for Vue.js Development

Get Started →

Vue CLI

- Project setup
- Prototyping
- Plugin system
- Build (app, lib, web component)
- UI

TypeScript or JavaScript



Boilerplates

Vue Enterprise Boilerplate (by Chris Fritz)

README.md

Vue Enterprise Boilerplate

PASSED

> This is an ever-evolving, very opinionated architecture and dev environment for new Vue SPA projects using Vue CLI 3. Questions, feedback, and for now, even bikeshedding are welcome. 😄 If you'd like to increase the time I can spend on this project, as well as other Vue resources, **please consider becoming a sponsor on Patreon**. 🙏

Features

- **Thorough documentation**: Written with the same care as Vue's core docs to quickly train new team members and consolidate knowledge.
- **Guaranteed consistency**: Opinionated linting for Vue, JavaScript/JSON, SCSS, and Markdown, integrated into Visual Studio Code and run against staged files on pre-commit.
- **First-class tests**: Practice test-driven development with both unit and end-to-end tests. Unit tests with Jest live as first-class citizens alongside your source files, while Cypress provides reliable end-to-end tests in an intuitive GUI for development.
- **Speedy development**: Between configurable generators, handy aliases, and global base components, your productivity will skyrocket.

🔗 https://github.com/chrisvfritz/vue-enterprise-boilerplate

vue-starter

https://github.com/devCrossNet/vue-starter

README.md

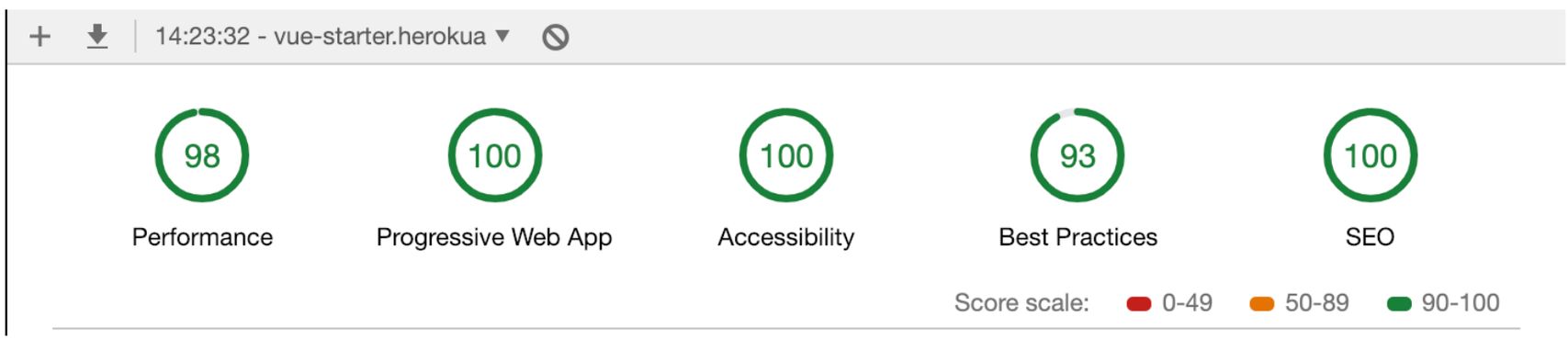
vue-starter v2.0.2 tweet slack

> The most complete boilerplate for production-ready PWAs. With focus on performance, development speed, and best practices

> 一个灵活的、可扩展的、自定的，已经准备好用于生产的渐进式网络应用样板，聚焦于性能、开发速度和最佳实践

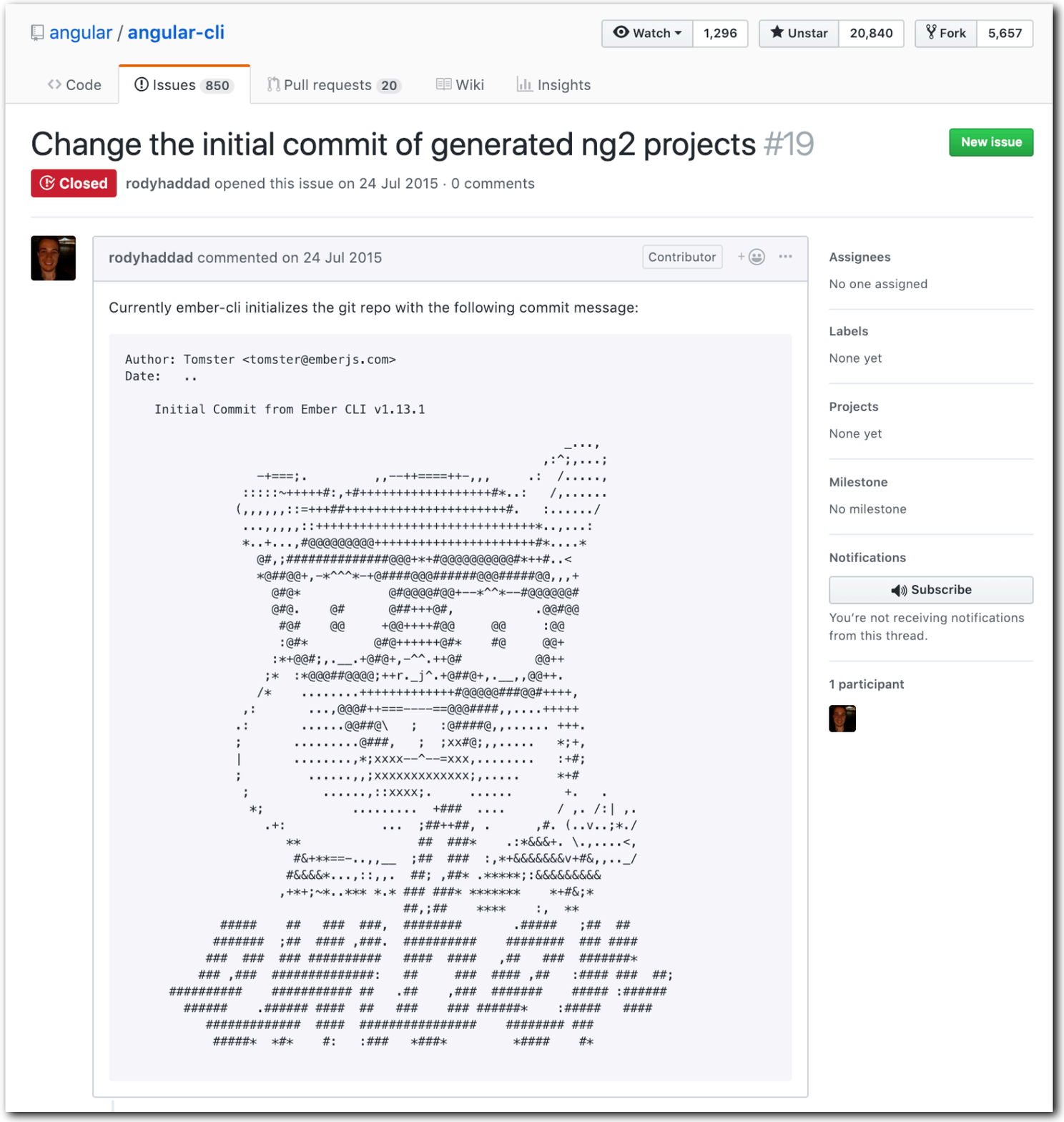
Maintained? yes | build passing | codecov 100% | maintainability A | dependencies up to date | dev dependencies up to date | issues 178 closed | release v2.0.2 | License MIT

Lighthouse Score



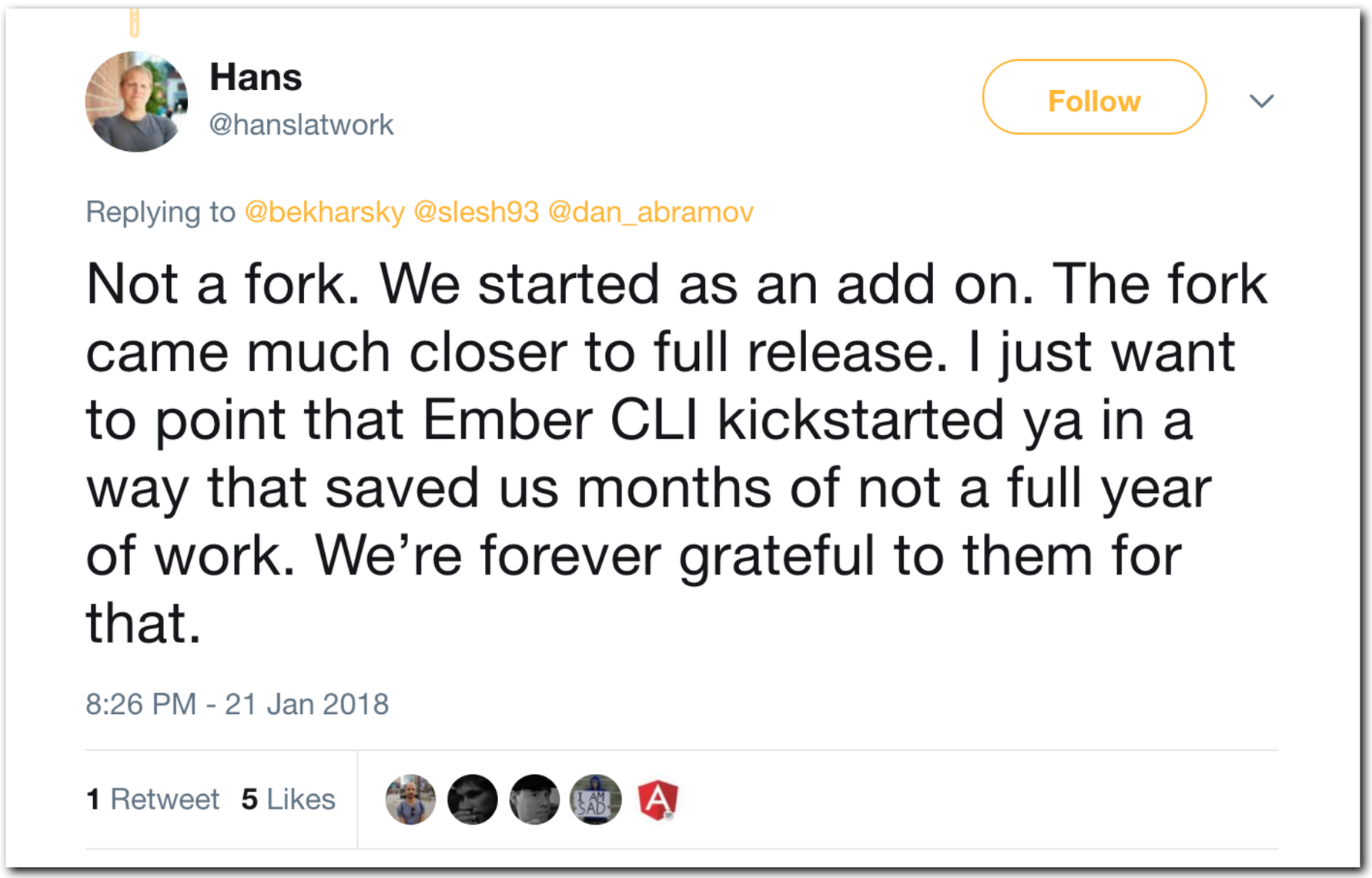
🔗 https://github.com/devCrossNet/vue-starter

Angular CLI issue #19



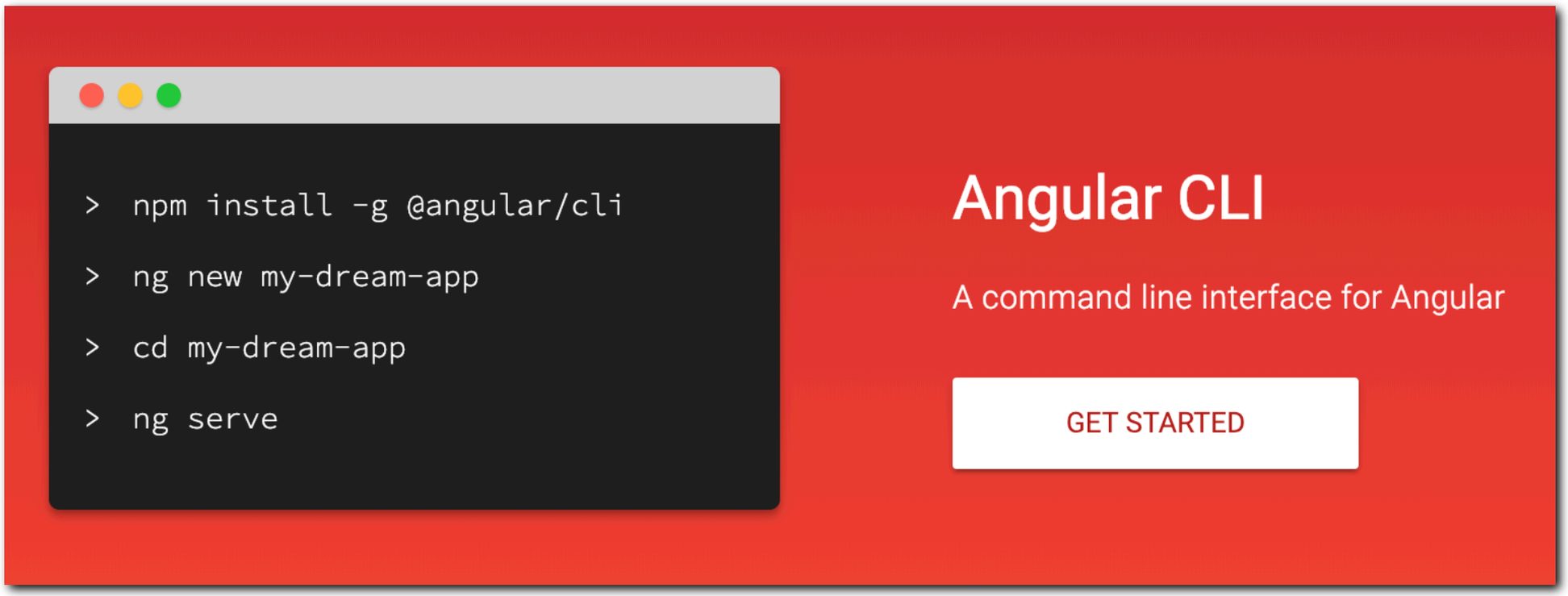
🔗 https://github.com/angular/angular-cli/issues/19

Ember CLI kickstarted Angular CLI



🔗 https://twitter.com/hanslatwork/status/955220131445010432

Angular CLI



🔗 https://cli.angular.io/

Angular CLI

- Project setup
- Scaffolding 🔥🔥🔥
- Schematics (= Plugins)
- Build

🔗 https://cli.angular.io/

Angular Console

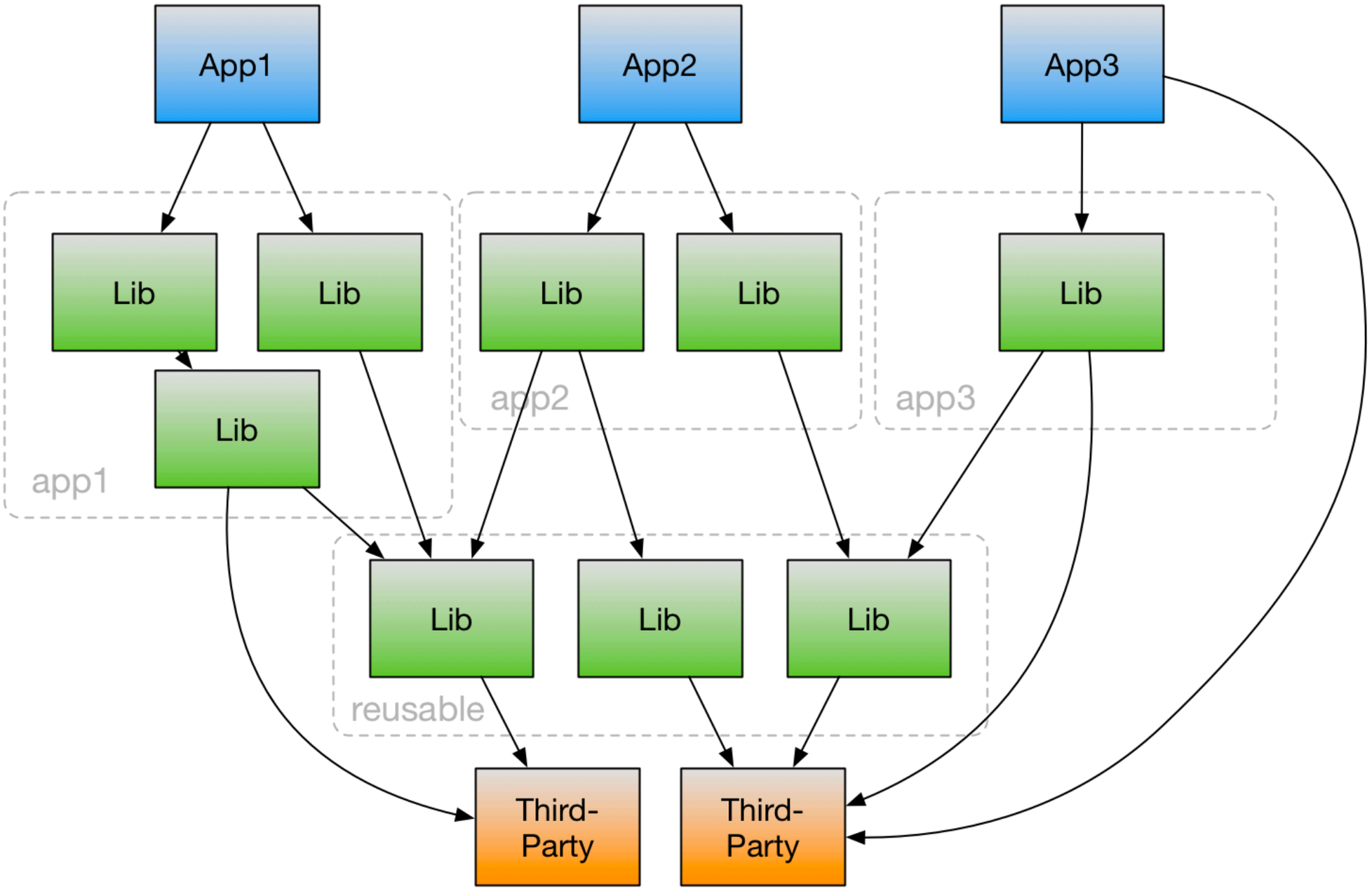
🔗 https://angularconsole.com/

Nrwl Extensions for Angular

An open source toolkit for enterprise Angular applications.

Nx is designed to help you create and build enterprise-grade Angular applications with proven project structure and patterns. We developed it based on our experience working at Google and helping the Fortune 500 build ambitious Angular applications.

NX



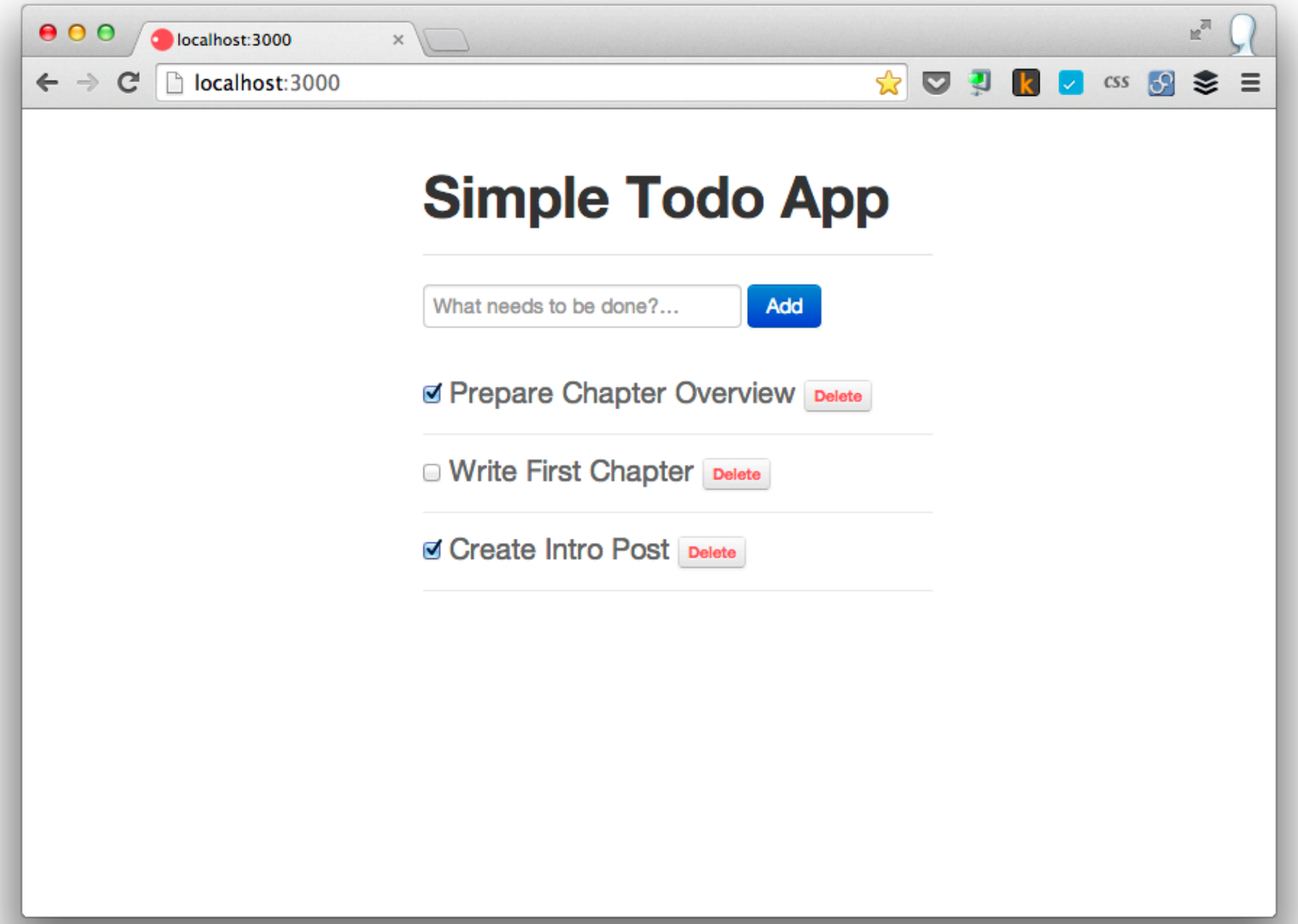
🔗 https://nrwl.io/nx/

🔗 https://blog.nrwl.io/12-things-to-help-large-organizations-do-angular-right-f261a798ad6b

Framework Pieces

NPM dependencies graph of the new project stack

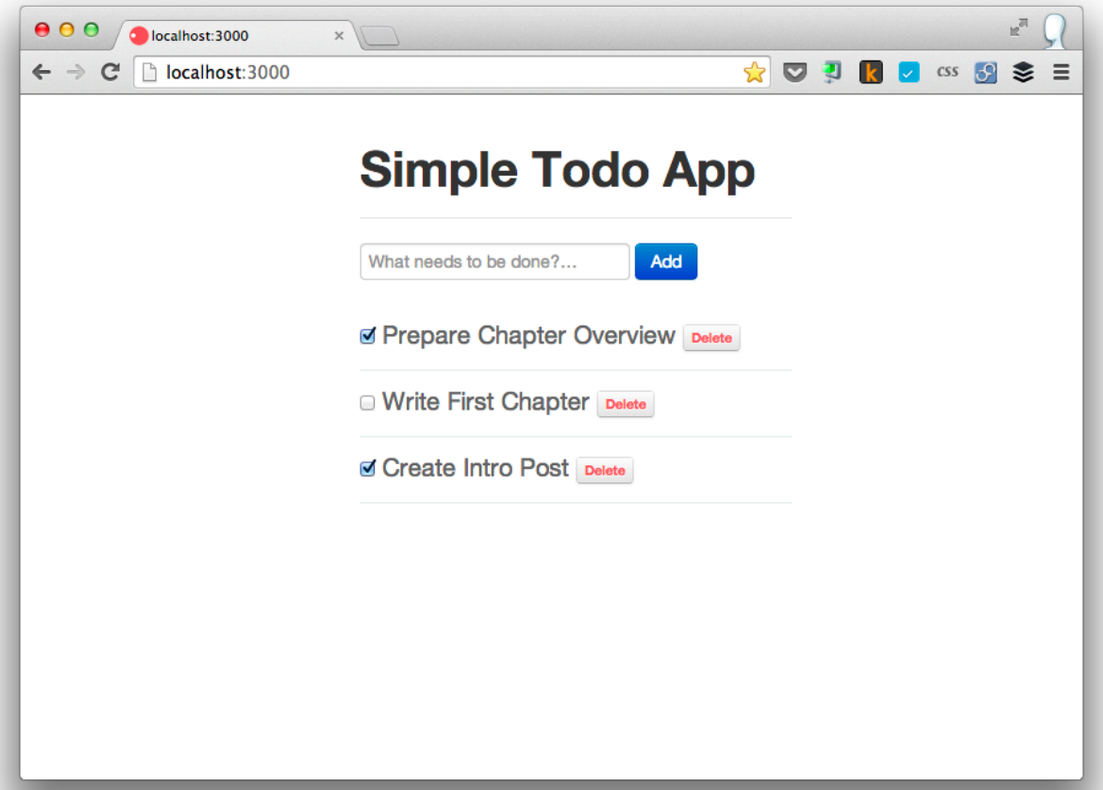
Imagine a todo app…

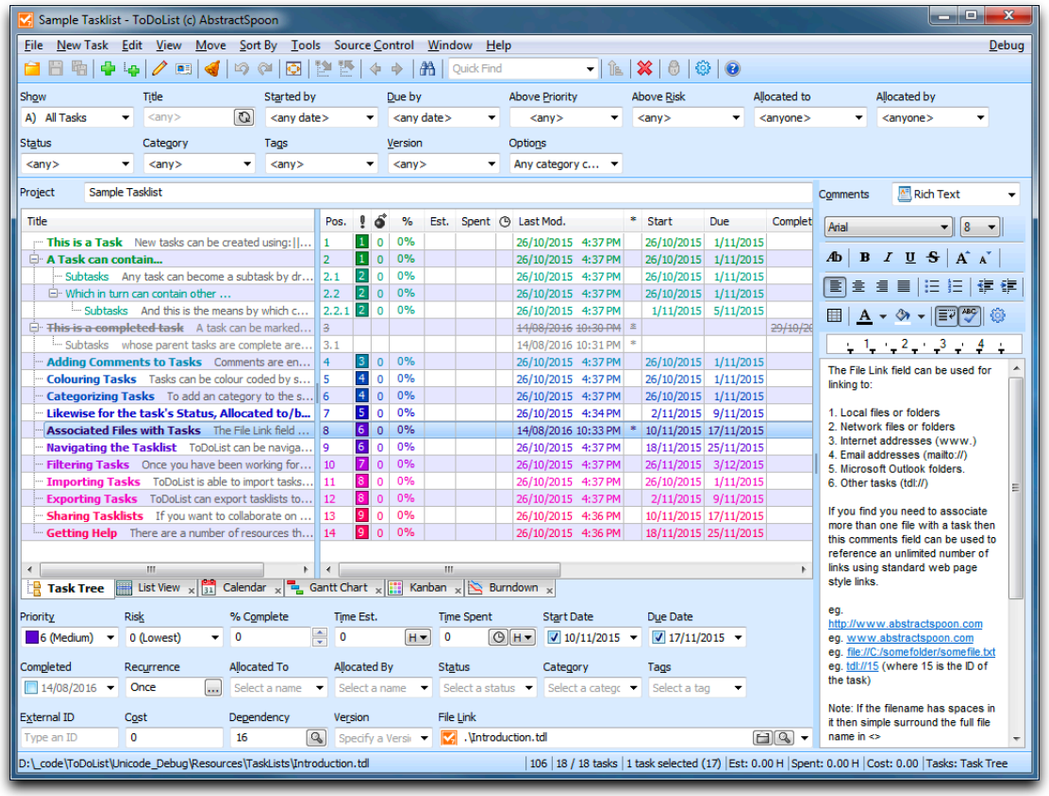


🔗 https://github.com/meteorhacks/meteor-todo-app

https://abstractspoon.weebly.com

Imagine a todo app…

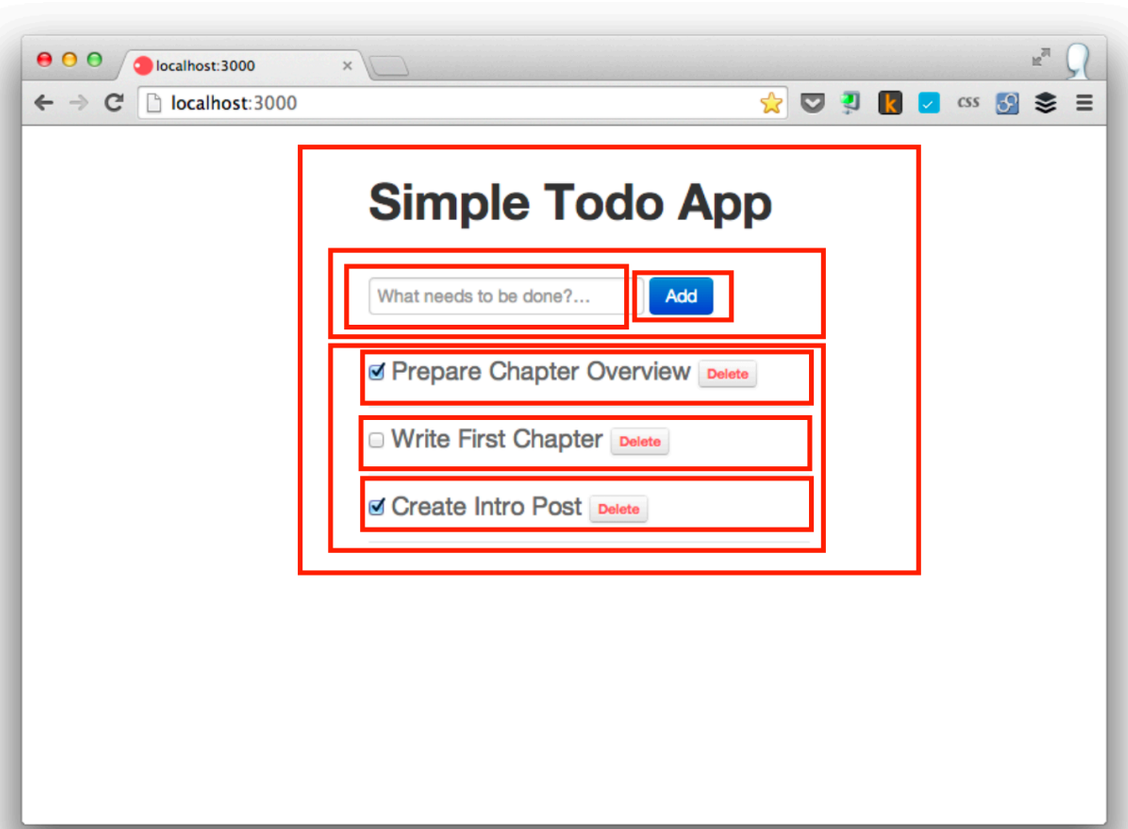




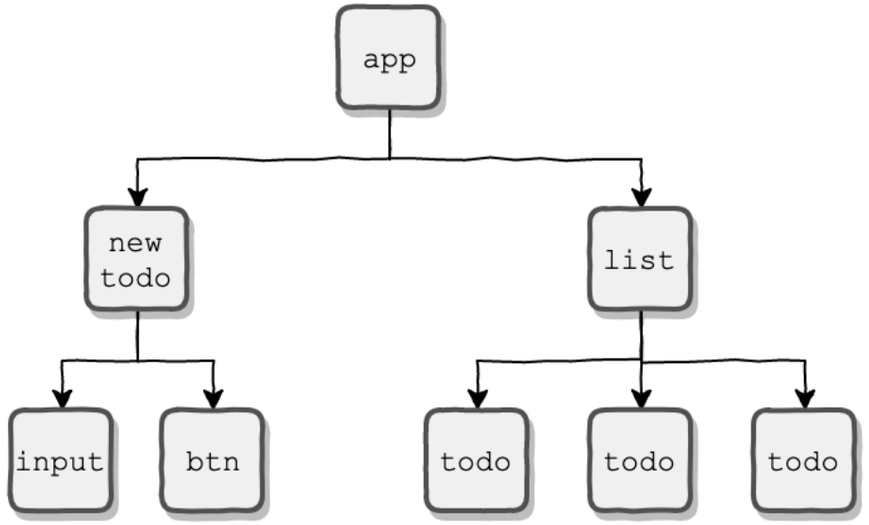
🔗 https://github.com/meteorhacks/meteor-todo-app

https://abstractspoon.weebly.com

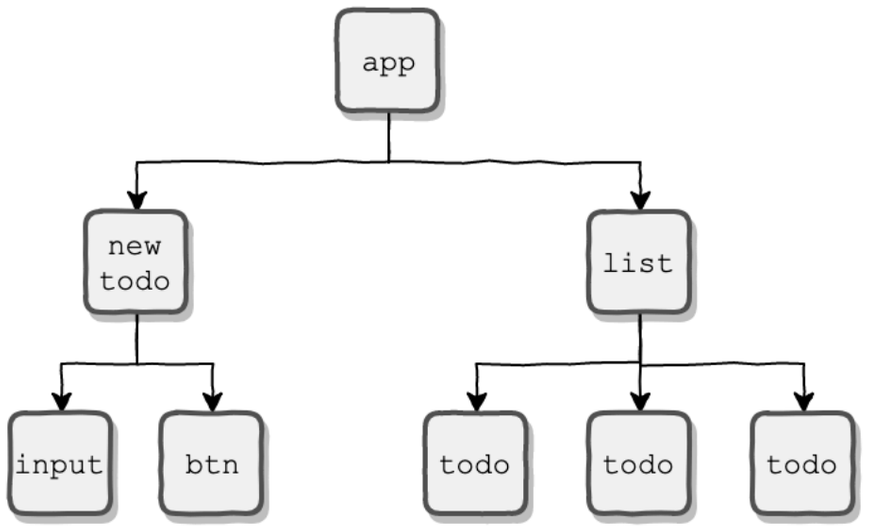
That’s just a bunch of components 😅



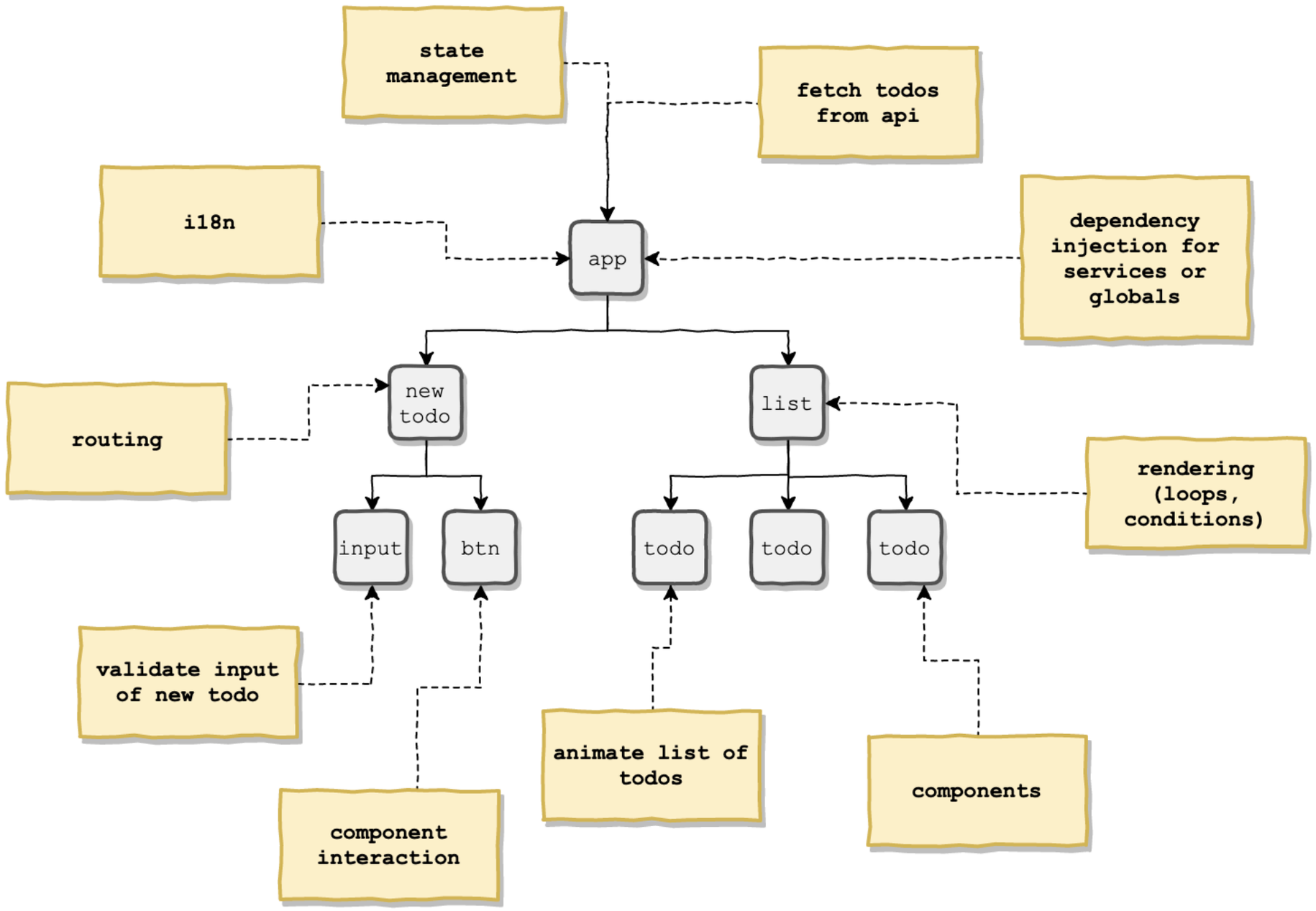
https://github.com/meteorhacks/meteor-todo-app



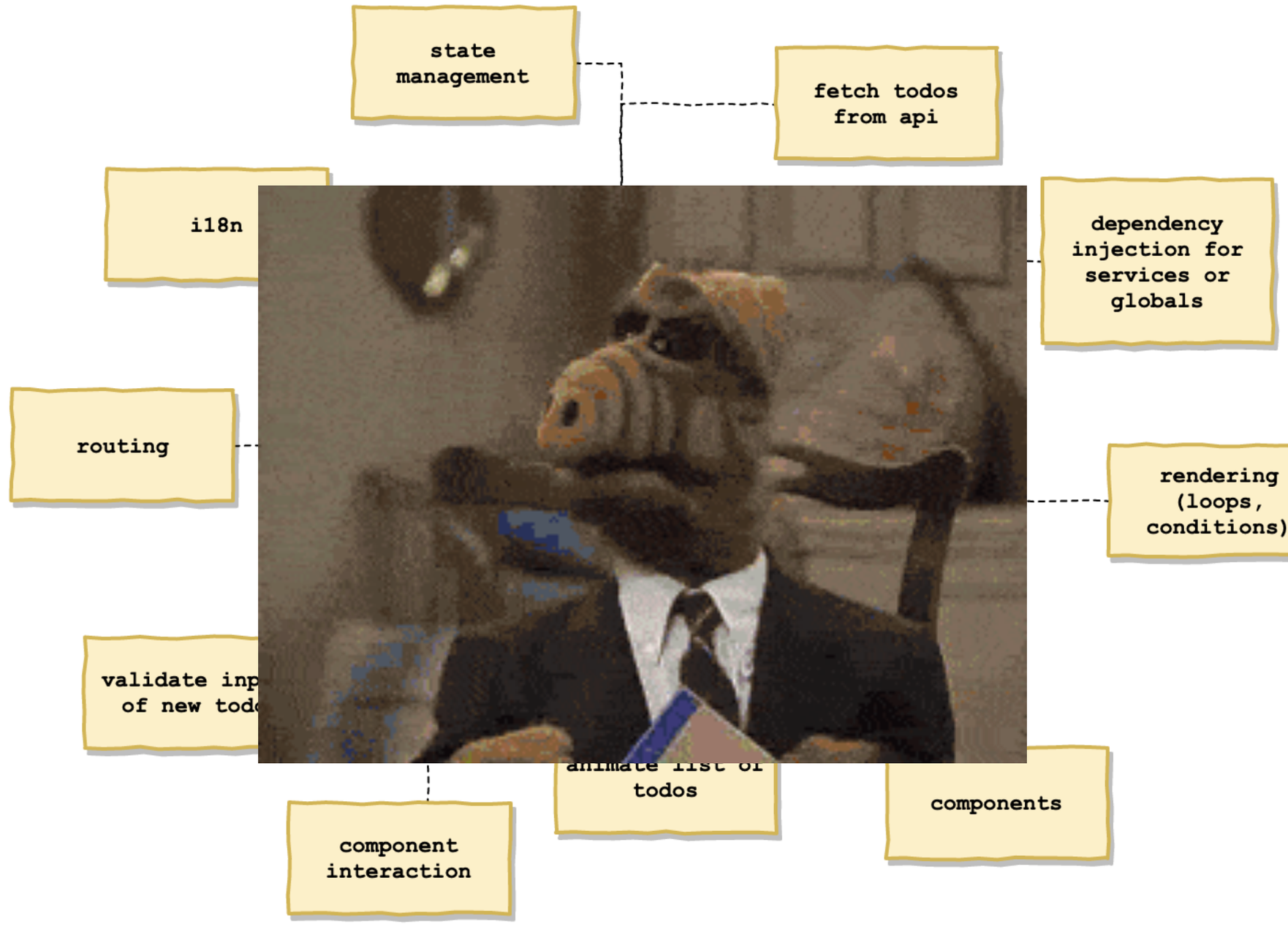
Oh wow 😰



Oh wow 😰


state management
fetch todos from api
i18n
app
dependency injection for services or globals
new todo
routing
input
btn
validate input of new todo
component interaction
list
rendering (loops, conditions)
todo
todo
todo
animate list of todos
components

Oh wow 😰


state management
fetch todos from api
i18n
dependency injection for services or globals
routing
rendering (loops, conditions)
validate inp
of new todo
animate list of todos
components
component interaction

Oh wow 😰


WHEN YOUR TODO APP NEEDS I18N, DEPENDENCY INJECTION, ROUTING, ANIMATIONS, STATE MANAGEMENT AND FORM VALIDATION
cabbygator
Modern problems require modern solutions
imgflip.com

✅ or ❌ Part of framework*

* package maintained by the frameworks core team
  & contained in the GitHub organization

📦 name of package(s)

📋 additional resource or list or whatever

Build components and render data

```
@Component({
  selector: 'todo-item',
  template: `<li>{{ todo }}</li>`,
})
export class TodoItemComponent {
    todo: string = 'learn angular';
}
```

✅ Part of angular
📦 @angular/core

```
<template>
  <li>{{ todo }}</li>
</template>

<script>
export default {
  data: function () {
    return {
      todo: 'learn vue'
    }
  }
}
</script>
```

✅ Part of vue
📦 vue

Build components and render data

```angular
@Component({
  selector: 'todo-item',
  template: `<li>{{ todo }}</li>`,
})
export class TodoItemComponent {
    todo: string = 'learn angular';
}
```

✅ Part of angular

📦 @angular/core

```vue
<template>
  <li>{{ todo }}</li>
</template>

<script>
export default {
  data: function () {
    return {
      todo: 'learn vue'
    }
  }
}
</script>
```

✅ Part of vue

📦 vue

Build components and render data

```angular
@Component({
  selector: 'todo-item',
  template: `<li>{{ todo }}</li>`,
})
export class TodoItemComponent {
    todo: string = 'learn angular';
}
```

✅ Part of angular

📦 @angular/core

```vue
<template>
  <li>{{ todo }}</li>
</template>

<script>
export default {
  data: function () {
    return {
      todo: 'learn vue'
    }
  }
}
</script>
```

✅ Part of vue

📦 vue

Build components and render data

```
@Component({
  selector: 'todo-item',
  template: `<li>{{ todo }}</li>`,
})
export class TodoItemComponent {
    todo: string = 'learn angular';
}
```

✅ Part of angular
📦 @angular/core

```
<template>
  <li>{{ todo }}</li>
</template>

<script>
export default {
  data: function () {
    return {
      todo: 'learn vue'
    }
  }
}
</script>
```

✅ Part of vue
📦 vue

Routing (aka mapping of components to urls)

```
const appRoutes: Routes = [
  { path: 'todos', component: TodoListComponent },
  { path: 'todos/:id', component: TodoComponent },
  { path: '', redirectTo: '/todos', pathMatch: 'full'
  },
  { path: '**', component: PageNotFoundComponent }
];

@NgModule({
  imports: [RouterModule.forRoot(appRoutes)],
  exports: [RouterModule]
})
export class AppRoutingModule {}
```

```
const routes = [
    { path: '/todos', component: TodoList },
    { path: '/todos/:id', component: Todo },
    { path: '/', redirect: { name: 'todos' }},
    { path: '*', component: PageNotFound }
]

const router = new VueRouter({
  routes // short for `routes: routes`
})
```

✅ Part of angular
📦 @angular/router

✅ Part of vue
📦 vue-router

Routing (aka mapping of components to urls)

```
const appRoutes: Routes = [
  { path: 'todos', component: TodoListComponent },
  { path: 'todos/:id', component: TodoComponent },
  { path: '', redirectTo: '/todos', pathMatch: 'full'
  },
  { path: '**', component: PageNotFoundComponent }
];

@NgModule({
  imports: [RouterModule.forRoot(appRoutes)],
  exports: [RouterModule]
})
export class AppRoutingModule {}
```

```
const routes = [
    { path: '/todos', component: TodoList },
    { path: '/todos/:id', component: Todo },
    { path: '/', redirect: { name: 'todos' }},
    { path: '*', component: PageNotFound }
]

const router = new VueRouter({
  routes // short for `routes: routes`
})
```

✅ Part of angular
📦 @angular/router

✅ Part of vue
📦 vue-router

Routing (aka mapping of components to urls)

```
const appRoutes: Routes = [
  { path: 'todos', component: TodoListComponent },
  { path: 'todos/:id', component: TodoComponent },
  { path: '', redirectTo: '/todos', pathMatch: 'full'
  },
  { path: '**', component: PageNotFoundComponent }
];

@NgModule({
  imports: [RouterModule.forRoot(appRoutes)],
  exports: [RouterModule]
})
export class AppRoutingModule {}
```

```
const routes = [
    { path: '/todos', component: TodoList },
    { path: '/todos/:id', component: Todo },
    { path: '/', redirect: { name: 'todos' }},
    { path: '*', component: PageNotFound }
]

const router = new VueRouter({
  routes // short for `routes: routes`
})
```

✅ Part of angular
📦 @angular/router

✅ Part of vue
📦 vue-router

Routing (aka mapping of components to urls)

```
const appRoutes: Routes = [
  { path: 'todos', component: TodoListComponent },
  { path: 'todos/:id', component: TodoComponent },
  { path: '', redirectTo: '/todos', pathMatch: 'full'
  },
  { path: '**', component: PageNotFoundComponent }
];

@NgModule({
  imports: [RouterModule.forRoot(appRoutes)],
  exports: [RouterModule]
})
export class AppRoutingModule {}
```

```
const routes = [
    { path: '/todos', component: TodoList },
    { path: '/todos/:id', component: Todo },
    { path: '/', redirect: { name: 'todos' }},
    { path: '*', component: PageNotFound }
]

const router = new VueRouter({
  routes // short for `routes: routes`
})
```

✅ Part of angular
📦 @angular/router

✅ Part of vue
📦 vue-router

Fetch data from backend

```ts
export class TodoListComponent implements OnInit {
    todos$: Observable<Todo[]>;

    constructor(private http:HttpClient) {
    }

    ngOnInit() {
        this.todos$ = this.http
            .get<Todo[]>('https://someUrl/todos');
    }
}
```

✅ Part of angular

📦 @angular/common

```vue
<script>
export default {
  data: function () {
    return {
      todos: []
    }
  },
  mounted () {
    axios
      .get('https://someUrl/todos')
      .then(response => (this.todos = response))
  }
}
</script>
```

❌ Not part of vue

📦 i.e. axios or vue-resource

📋 https://github.com/vuejs/awesome-vue#http-requests

Fetch data from backend

```ts
export class TodoListComponent implements OnInit {
    todos$: Observable<Todo[]>;

    constructor(private http:HttpClient) {
    }

    ngOnInit() {
        this.todos$ = this.http
            .get<Todo[]>('https://someUrl/todos');
    }
}
```

✅ Part of angular

📦 @angular/common

```vue
<script>
export default {
  data: function () {
    return {
      todos: []
    }
  },
  mounted () {
    axios
      .get('https://someUrl/todos')
      .then(response => (this.todos = response))
  }
}
</script>
```

❌ Not part of vue

📦 i.e. axios or vue-resource

📋 https://github.com/vuejs/awesome-vue#http-requests

Fetch data from backend

```ts
export class TodoListComponent implements OnInit {
    todos$: Observable<Todo[]>;

    constructor(private http:HttpClient) {
    }

    ngOnInit() {
        this.todos$ = this.http
            .get<Todo[]>('https://someUrl/todos');
    }
}
```

✅ Part of angular

📦 @angular/common

```vue
<script>
export default {
  data: function () {
    return {
      todos: []
    }
  },
  mounted () {
    axios
      .get('https://someUrl/todos')
      .then(response => (this.todos = response))
  }
}
</script>
```

❌ Not part of vue

📦 i.e. axios or vue-resource

📋 https://github.com/vuejs/awesome-vue#http-requests

Fetch data from backend

```ts
export class TodoListComponent implements OnInit {
    todos$: Observable<Todo[]>;

    constructor(private http:HttpClient) {
    }

    ngOnInit() {
        this.todos$ = this.http
            .get<Todo[]>('https://someUrl/todos');
    }
}
```

✅ Part of angular

📦 @angular/common

```vue
<script>
export default {
  data: function () {
    return {
      todos: []
    }
  },
  mounted () {
    axios
      .get('https://someUrl/todos')
      .then(response => (this.todos = response))
  }
}
</script>
```

❌ Not part of vue

📦 i.e. axios or vue-resource

📋 https://github.com/vuejs/awesome-vue#http-requests

Manage state

```
// Totally opinionated by myself
// b/c I like angular services

@Injectable()
export class TodoService {

  private todos: Todo[] = [];

  constructor() { }

  addTodo(newTodo: Todo) {
    this.todos.push(newTodo);
  }
}
```

✅ Part of angular, but depends on you
📦 @angular/core or @ngrx/platform

```
const store = new Vuex.Store({
  state: {
    todos: []
  },
  mutations: {
    addTodo (state, newTodo) {
      state.todos.push(newTodo)
    }
  }
})
```

✅ Part of vue
📦 vuex
📋 https://github.com/vuejs/vuex

Manage state

```
// Totally opinionated by myself
// b/c I like angular services

@Injectable()
export class TodoService {

  private todos: Todo[] = [];

  constructor() { }

  addTodo(newTodo: Todo) {
    this.todos.push(newTodo);
  }
}
```

✅ Part of angular, but depends on you

📦 @angular/core or @ngrx/platform

```
const store = new Vuex.Store({
  state: {
    todos: []
  },
  mutations: {
    addTodo (state, newTodo) {
      state.todos.push(newTodo)
    }
  }
})
```

✅ Part of vue

📦 vuex

📋 https://github.com/vuejs/vuex

Manage state

```
// Totally opinionated by myself
// b/c I like angular services

@Injectable()
export class TodoService {

  private todos: Todo[] = [];

  constructor() { }

  addTodo(newTodo: Todo) {
    this.todos.push(newTodo);
  }
}
```

✅ Part of angular, but depends on you

📦 @angular/core or @ngrx/platform

```
const store = new Vuex.Store({
  state: {
    todos: []
  },
  mutations: {
    addTodo (state, newTodo) {
      state.todos.push(newTodo)
    }
  }
})
```

✅ Part of vue

📦 vuex

📋 https://github.com/vuejs/vuex

Manage state

```
// Totally opinionated by myself
// b/c I like angular services

@Injectable()
export class TodoService {

  private todos: Todo[] = [];

  constructor() { }

  addTodo(newTodo: Todo) {
    this.todos.push(newTodo);
  }
}
```

✅ Part of angular, but depends on you
📦 @angular/core or @ngrx/platform

```
const store = new Vuex.Store({
  state: {
    todos: []
  },
  mutations: {
    addTodo (state, newTodo) {
      state.todos.push(newTodo)
    }
  }
})
```

✅ Part of vue
📦 vuex
📋 https://github.com/vuejs/vuex

Form handling and validation

```
<form>
  <input id="newTodo" name="newTodo"
         required minlength="4"
         [(ngModel)]="newTodo" #refTodo="ngModel">

  <div *ngIf="refTodo.invalid &&
              (refTodo.dirty || refTodo.touched)">
    <div *ngIf="refTodo.errors.required">
      Name is required.
    </div>
    <div *ngIf="refTodo.errors.minlength">
      Name must be at least 4 characters long.
    </div>
  </div>
</form>
```

✅ Part of angular

📦 @angular/core or @ngrx/platform

```
<!-- form input handling with vue -->
<input v-model="todo" placeholder="Get sh*t done, yo!">
<p>Todo is: {{ todo }}</p>
```

```
<!-- form validation with vee-validate -->
<input v-validate="'required|min:5'" name="todo" type="text">
<span>{{ errors.first('todo') }}</span>
```

✅ Form handling is part of vue

❌ Form validation is part of vue

📦 vue or vee-validate

📋 https://github.com/vuejs/awesome-vue#validation

Form handling and validation

```
<form>
  <input id="newTodo" name="newTodo"
         required minlength="4"
         [(ngModel)]="newTodo" #refTodo="ngModel">

  <div *ngIf="refTodo.invalid &&
              (refTodo.dirty || refTodo.touched)">
    <div *ngIf="refTodo.errors.required">
      Name is required.
    </div>
    <div *ngIf="refTodo.errors.minlength">
      Name must be at least 4 characters long.
    </div>
  </div>
</form>
```

✅ Part of angular

📦 @angular/core or @ngrx/platform

```
<!-- form input handling with vue -->
<input v-model="todo" placeholder="Get sh*t done, yo!">
<p>Todo is: {{ todo }}</p>
```

```
<!-- form validation with vee-validate -->
<input v-validate="'required|min:5'" name="todo" type="text">
<span>{{ errors.first('todo') }}</span>
```

✅ Form handling is part of vue

❌ Form validation is part of vue

📦 vue or vee-validate

📋 https://github.com/vuejs/awesome-vue#validation

Form handling and validation

```
<form>
  <input id="newTodo" name="newTodo"
         required minlength="4"
         [(ngModel)]="newTodo" #refTodo="ngModel">

  <div *ngIf="refTodo.invalid &&
              (refTodo.dirty || refTodo.touched)">
    <div *ngIf="refTodo.errors.required">
      Name is required.
    </div>
    <div *ngIf="refTodo.errors.minlength">
      Name must be at least 4 characters long.
    </div>
  </div>
</form>
```

✅ Part of angular

📦 @angular/core or @ngrx/platform

```
<!-- form input handling with vue -->
<input v-model="todo" placeholder="Get sh*t done, yo!">
<p>Todo is: {{ todo }}</p>
```

```
<!-- form validation with vee-validate -->
<input v-validate="'required|min:5'" name="todo" type="text">
<span>{{ errors.first('todo') }}</span>
```

✅ Form handling is part of vue

❌ Form validation is part of vue

📦 vue or vee-validate

📋 https://github.com/vuejs/awesome-vue#validation

Form handling and validation

```
<form>
  <input id="newTodo" name="newTodo"
         required minlength="4"
         [(ngModel)]="newTodo" #refTodo="ngModel">

  <div *ngIf="refTodo.invalid &&
              (refTodo.dirty || refTodo.touched)">
    <div *ngIf="refTodo.errors.required">
      Name is required.
    </div>
    <div *ngIf="refTodo.errors.minlength">
      Name must be at least 4 characters long.
    </div>
  </div>
</form>
```

✅ Part of angular

📦 @angular/core or @ngrx/platform

```
<!-- form input handling with vue -->
<input v-model="todo" placeholder="Get sh*t done, yo!">
<p>Todo is: {{ todo }}</p>
```

```
<!-- form validation with vee-validate -->
<input v-validate="'required|min:5'" name="todo" type="text">
<span>{{ errors.first('todo') }}</span>
```

✅ Form handling is part of vue

❌ Form validation is part of vue

📦 vue or vee-validate

📋 https://github.com/vuejs/awesome-vue#validation

Dependency Injection

```
// Service (or whatever) to be injected
@Injectable({
  providedIn: 'root',
})
export class TodoService {
  private todos = ['prepare slides', 'sleep']

  constructor() { }

  getTodos() {
    return this.todos;
  }
}


/*** MORE MAGIC HAPPENS HERE ***/


// Inject the injectable in component
export class TodoComponent {

  constructor(todoService: TodoService) {
    this.heroes = heroService.getTodos();
  }
}
```

✅ Part of angular

📦 @angular/core

```
// parent component providing 'foo'
var Provider = {
  provide: {
    foo: 'bar'
  },
  // ...
}

// child component injecting 'foo'
var Child = {
  inject: ['foo'],
  created () {
    console.log(this.foo) // => "bar"
  }
  // ...
}
```

✅ Part of vue

📦 vue

Dependency Injection

```
// Service (or whatever) to be injected
@Injectable({
  providedIn: 'root',
})
export class TodoService {
  private todos = ['prepare slides', 'sleep']

  constructor() { }

  getTodos() {
    return this.todos;
  }
}


/*** MORE MAGIC HAPPENS HERE ***/


// Inject the injectable in component
export class TodoComponent {

  constructor(todoService: TodoService) {
    this.heroes = heroService.getTodos();
  }
}
```

✅ Part of angular

📦 @angular/core

```
// parent component providing 'foo'
var Provider = {
  provide: {
    foo: 'bar'
  },
  // ...
}

// child component injecting 'foo'
var Child = {
  inject: ['foo'],
  created () {
    console.log(this.foo) // => "bar"
  }
  // ...
}
```

✅ Part of vue

📦 vue

Dependency Injection

```
// Service (or whatever) to be injected
@Injectable({
  providedIn: 'root',
})
export class TodoService {
  private todos = ['prepare slides', 'sleep']

  constructor() { }

  getTodos() {
    return this.todos;
  }
}


/*** MORE MAGIC HAPPENS HERE ***/


// Inject the injectable in component
export class TodoComponent {

  constructor(todoService: TodoService) {
    this.heroes = heroService.getTodos();
  }
}
```

✅ Part of angular

📦 @angular/core

```
// parent component providing 'foo'
var Provider = {
  provide: {
    foo: 'bar'
  },
  // ...
}

// child component injecting 'foo'
var Child = {
  inject: ['foo'],
  created () {
    console.log(this.foo) // => "bar"
  }
  // ...
}
```

✅ Part of vue

📦 vue

Dependency Injection

```
// Service (or whatever) to be injected
@Injectable({
  providedIn: 'root',
})
export class TodoService {
  private todos = ['prepare slides', 'sleep']

  constructor() { }

  getTodos() {
    return this.todos;
  }
}


/*** MORE MAGIC HAPPENS HERE ***/


// Inject the injectable in component
export class TodoComponent {

  constructor(todoService: TodoService) {
    this.heroes = heroService.getTodos();
  }
}
```

✅ Part of angular

📦 @angular/core

```
// parent component providing 'foo'
var Provider = {
  provide: {
    foo: 'bar'
  },
  // ...
}

// child component injecting 'foo'
var Child = {
  inject: ['foo'],
  created () {
    console.log(this.foo) // => "bar"
  }
  // ...
}
```

✅ Part of vue

📦 vue

i18n

```
{
    "HOME": {
        "TITLE": "hello {{value}}"
    }
}
```

```
<!-- i18n with ngx-translate -->
<h2>{{ 'HOME.TITLE' | translate }}</h2>
```

✅ Part of angular, but there are better alternatives

📦 @angular/core or @ngx-translate/core

```
const messages = {
  en: {
    title: 'TODO App',
  }
}
```

```
<h2>{{ $t('title') }}</h2>
```

❌ Not part of vue (but maintained by a core member)

📦 vue-i18n

📋 https://github.com/vuejs/awesome-vue#i18n

i18n

```
{
    "HOME": {
        "TITLE": "hello {{value}}"
    }
}
```

```
<!-- i18n with ngx-translate -->
<h2>{{ 'HOME.TITLE' | translate }}</h2>
```

✅ Part of angular, but there are better alternatives

📦 @angular/core or @ngx-translate/core

```
const messages = {
  en: {
    title: 'TODO App',
  }
}
```

```
<h2>{{ $t('title') }}</h2>
```

❌ Not part of vue (but maintained by a core member)

📦 vue-i18n

📋 https://github.com/vuejs/awesome-vue#i18n

i18n

```
{
    "HOME": {
        "TITLE": "hello {{value}}"
    }
}
```

```
<!-- i18n with ngx-translate -->
<h2>{{ 'HOME.TITLE' | translate }}</h2>
```

```
const messages = {
  en: {
    title: 'TODO App',
  }
}
```

```
<h2>{{ $t('title') }}</h2>
```

✅ Part of angular, but there are better alternatives

📦 @angular/core or @ngx-translate/core

❌ Not part of vue (but maintained by a core member)

📦 vue-i18n

📋 https://github.com/vuejs/awesome-vue#i18n

i18n

```
{
    "HOME": {
        "TITLE": "hello {{value}}"
    }
}
```

```
<!-- i18n with ngx-translate -->
<h2>{{ 'HOME.TITLE' | translate }}</h2>
```

✅ Part of angular, but there are better alternatives

📦 @angular/core or @ngx-translate/core

```
const messages = {
  en: {
    title: 'TODO App',
  }
}
```

```
<h2>{{ $t('title') }}</h2>
```

❌ Not part of vue (but maintained by a core member)

📦 vue-i18n

📋 https://github.com/vuejs/awesome-vue#i18n

Animations

```
@Component({
  selector: 'app-open-close',
  animations: [
    trigger('openClose', [
        state('open',
            style({ height: '200px', opacity: 1 })),
        state('closed',
            style({ height: '100px', opacity: 0.5 })),
      transition('open => closed', [ animate('1s') ]),
      transition('closed => open', [ animate('0.5s') ]),
    ]),
  ],
  template:`<div [@openClose]="isOpen ? 'open' : 'closed'">
   <p>The box is now {{ isOpen ? 'Open' : 'Closed' }}!</p>
   </div>`
})
export class OpenCloseComponent {
  isOpen = true;

  toggle() {
    this.isOpen = !this.isOpen;
  }
}
```

✅ Part of angular

📦 @angular/animations

```
<template>
  <div>
    <button v-on:click="show = !show">Toggle</button>
    <transition name="fade">
      <p v-if="show">hello</p>
    </transition>
  </div>
<template>

<script>
export default = {
  data: { show: true }
}
</script>

<style>
.fade-enter-active, .fade-leave-active {
  transition: opacity .5s;
}
.fade-enter, .fade-leave-to {
  opacity: 0;
}
</style>
```

✅ Part of vue (basic support)

📦 vue

Animations

```
@Component({
  selector: 'app-open-close',
  animations: [
    trigger('openClose', [
        state('open',
            style({ height: '200px', opacity: 1 })),
        state('closed',
            style({ height: '100px', opacity: 0.5 })),
      transition('open => closed', [ animate('1s') ]),
      transition('closed => open', [ animate('0.5s') ]),
    ]),
  ],
  template:`<div [@openClose]="isOpen ? 'open' : 'closed'">
   <p>The box is now {{ isOpen ? 'Open' : 'Closed' }}!</p>
   </div>`
})
export class OpenCloseComponent {
  isOpen = true;

  toggle() {
    this.isOpen = !this.isOpen;
  }
}
```

✅ Part of angular

📦 @angular/animations

```
<template>
  <div>
    <button v-on:click="show = !show">Toggle</button>
    <transition name="fade">
      <p v-if="show">hello</p>
    </transition>
  </div>
<template>

<script>
export default = {
  data: { show: true }
}
</script>

<style>
.fade-enter-active, .fade-leave-active {
  transition: opacity .5s;
}
.fade-enter, .fade-leave-to {
  opacity: 0;
}
</style>
```

✅ Part of vue (basic support)

📦 vue

Animations

```
@Component({
  selector: 'app-open-close',
  animations: [
    trigger('openClose', [
        state('open',
            style({ height: '200px', opacity: 1 })),
        state('closed',
            style({ height: '100px', opacity: 0.5 })),
      transition('open => closed', [ animate('1s') ]),
      transition('closed => open', [ animate('0.5s') ]),
    ]),
  ],
  template:`<div [@openClose]="isOpen ? 'open' : 'closed'">
   <p>The box is now {{ isOpen ? 'Open' : 'Closed' }}!</p>
   </div>`
})
export class OpenCloseComponent {
  isOpen = true;

  toggle() {
    this.isOpen = !this.isOpen;
  }
}
```

```
<template>
  <div>
    <button v-on:click="show = !show">Toggle</button>
    <transition name="fade">
      <p v-if="show">hello</p>
    </transition>
  </div>
<template>

<script>
export default = {
  data: { show: true }
}
</script>

<style>
.fade-enter-active, .fade-leave-active {
  transition: opacity .5s;
}
.fade-enter, .fade-leave-to {
  opacity: 0;
}
</style>
```

✅ Part of angular

📦 @angular/animations

✅ Part of vue (basic support)

📦 vue

Animations

```
@Component({
  selector: 'app-open-close',
  animations: [
    trigger('openClose', [
        state('open',
            style({ height: '200px', opacity: 1 })),
        state('closed',
            style({ height: '100px', opacity: 0.5 })),
      transition('open => closed', [ animate('1s') ]),
      transition('closed => open', [ animate('0.5s') ]),
    ]),
  ],
  template:`<div [@openClose]="isOpen ? 'open' : 'closed'">
   <p>The box is now {{ isOpen ? 'Open' : 'Closed' }}!</p>
   </div>`
})
export class OpenCloseComponent {
  isOpen = true;

  toggle() {
    this.isOpen = !this.isOpen;
  }
}
```

✅ Part of angular

📦 @angular/animations

```
<template>
  <div>
    <button v-on:click="show = !show">Toggle</button>
    <transition name="fade">
      <p v-if="show">hello</p>
    </transition>
  </div>
<template>

<script>
export default = {
  data: { show: true }
}
</script>

<style>
.fade-enter-active, .fade-leave-active {
  transition: opacity .5s;
}
.fade-enter, .fade-leave-to {
  opacity: 0;
}
</style>
```

✅ Part of vue (basic support)

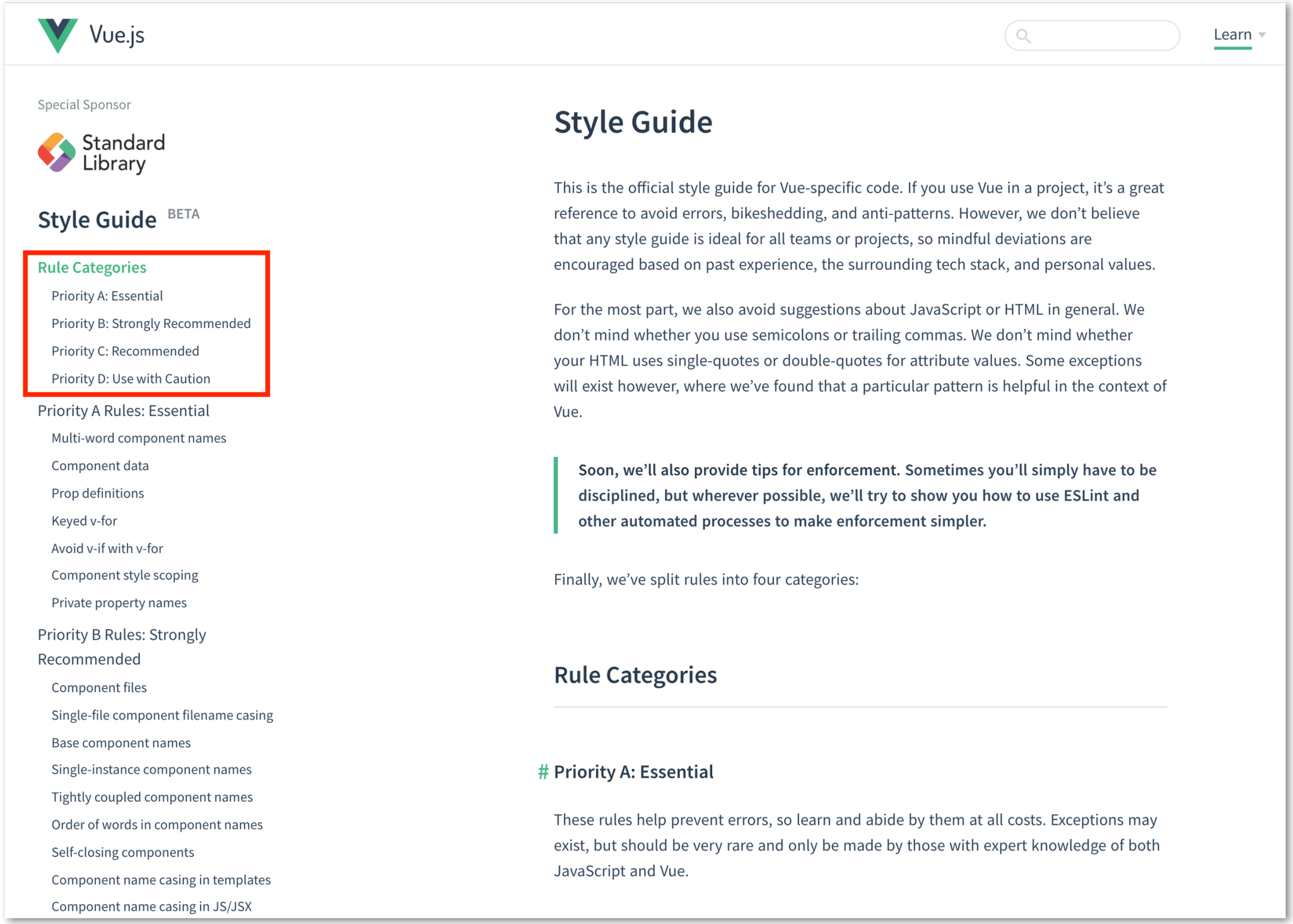
📦 vue

		
Core	✅	✅
Routing	✅	✅
HTTP	✅	❌
Form handling	✅	✅
Form validation	✅	❌
State handling	✅	✅
Dependency injection	✅	✅
i18n	✅	❌
Animations	✅	✅

Style Guide

Developers reading through the angular style guide

Vue Style Guide



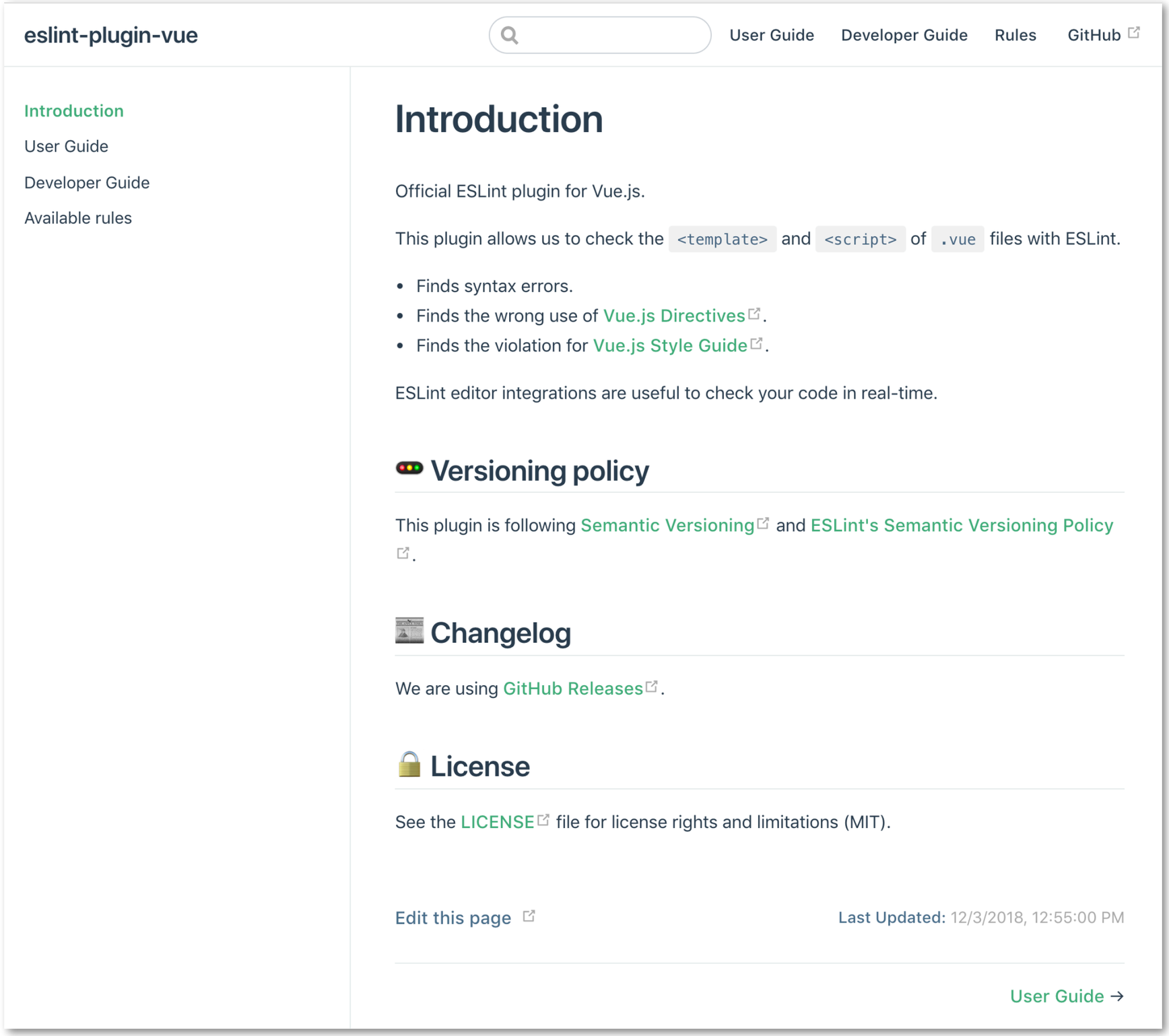
🔗 https://vuejs.org/v2/style-guide/

Vue Style Guide

„This is the official style guide for Vue-specific code. If you use Vue in a project, it’s a great reference to avoid errors, bikeshedding, and anti-patterns. **However, we don’t believe that any style guide is ideal for all teams or projects**, so mindful deviations are encouraged based on past experience, the surrounding tech stack, and personal values.“

🔗 https://vuejs.org/v2/style-guide/

eslint plugin



🔗 https://vuejs.github.io/eslint-plugin-vue/

Angular Style Guide

Style Guide

Looking for an opinionated guide to Angular syntax, conventions, and application structure? Step right in! This style guide presents preferred conventions and, as importantly, explains why.

Style vocabulary

Each guideline describes either a good or bad practice, and all have a consistent presentation.

The wording of each guideline indicates how strong the recommendation is.

Do is one that should always be followed. *Always* might be a bit too strong of a word. Guidelines that literally should always be followed are extremely rare. On the other hand, you need a really unusual case for breaking a *Do* guideline.

Consider guidelines should generally be followed. If you fully understand the meaning behind the guideline and have a good reason to deviate, then do so. Please strive to be consistent.

Avoid indicates something you should almost never do. Code examples to *avoid* have an unmistakable red header.

Why? gives reasons for following the previous recommendations.

🔗 https://angular.io/guide/styleguide#style-guide

Angular Style Guide format

Single responsibility

Apply the *single responsibility principle* (SRP) to all components, services, and other symbols. This helps make the app cleaner, easier to read and maintain, and more testable.

Rule of One

Style 01-01

Do define one thing, such as a service or component, per file.

Consider limiting files to 400 lines of code.

Why? One component per file makes it far easier to read, maintain, and avoid collisions with teams in source control.

Why? One component per file avoids hidden bugs that often arise when combining components in a file where they may share variables, create unwanted closures, or unwanted coupling with dependencies.

Why? A single component can be the default export for its file which facilitates lazy loading with the router.

The key is to make the code more reusable, easier to read, and less mistake prone.

The following *negative* example defines the `AppComponent`, bootstraps the app, defines the `Hero` model object, and loads heroes from the server all in the same file. *Don't do this*.

app/heroes/hero.component.ts

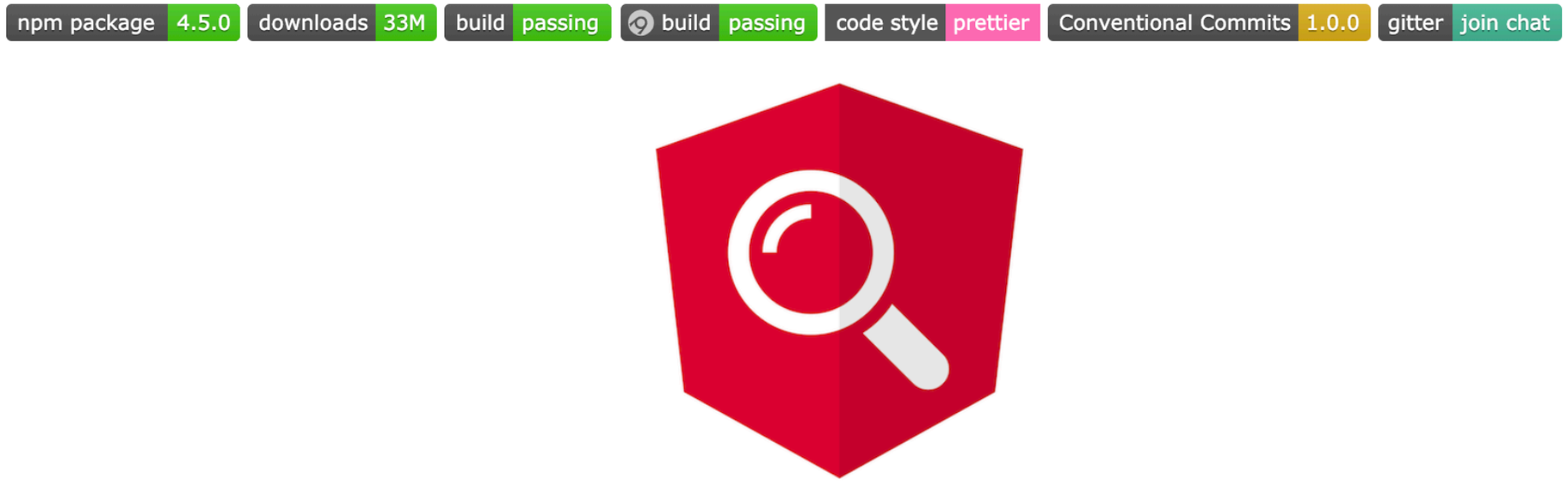
```
/* avoid */
import { Component, NgModule, OnInit } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { platformBrowserDynamic } from '@angular/platform-browser-dynamic';

```

🔗 https://angular.io/guide/styleguide#single-responsibility

Codelyzer (comes with Angular CLI)

README.md



codelyzer

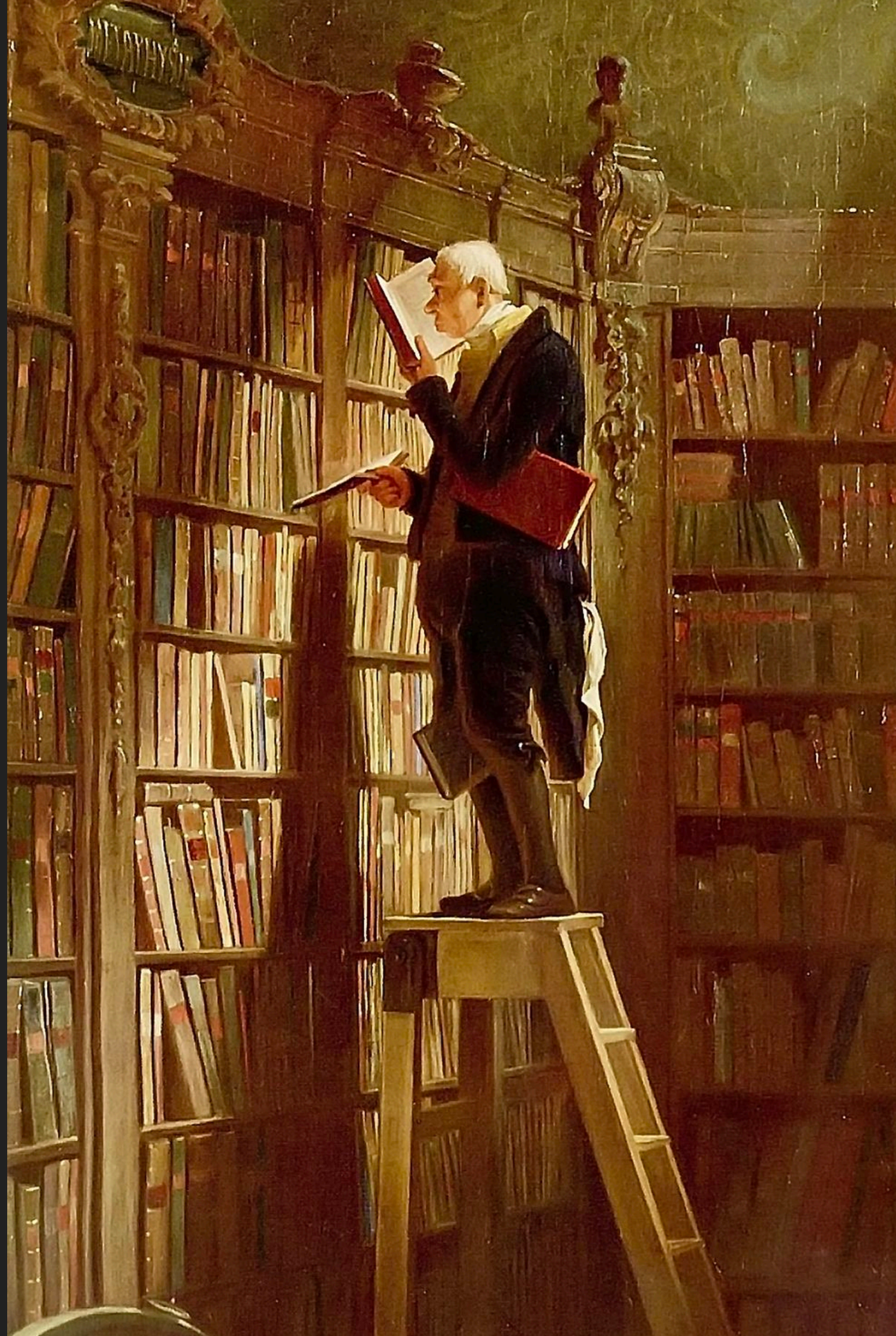
A set of tslint rules for static code analysis of Angular TypeScript projects.

You can run the static code analyzer over web apps, NativeScript, Ionic, etc.

Vote for your favorite feature here. For more details about the feature request process see this document

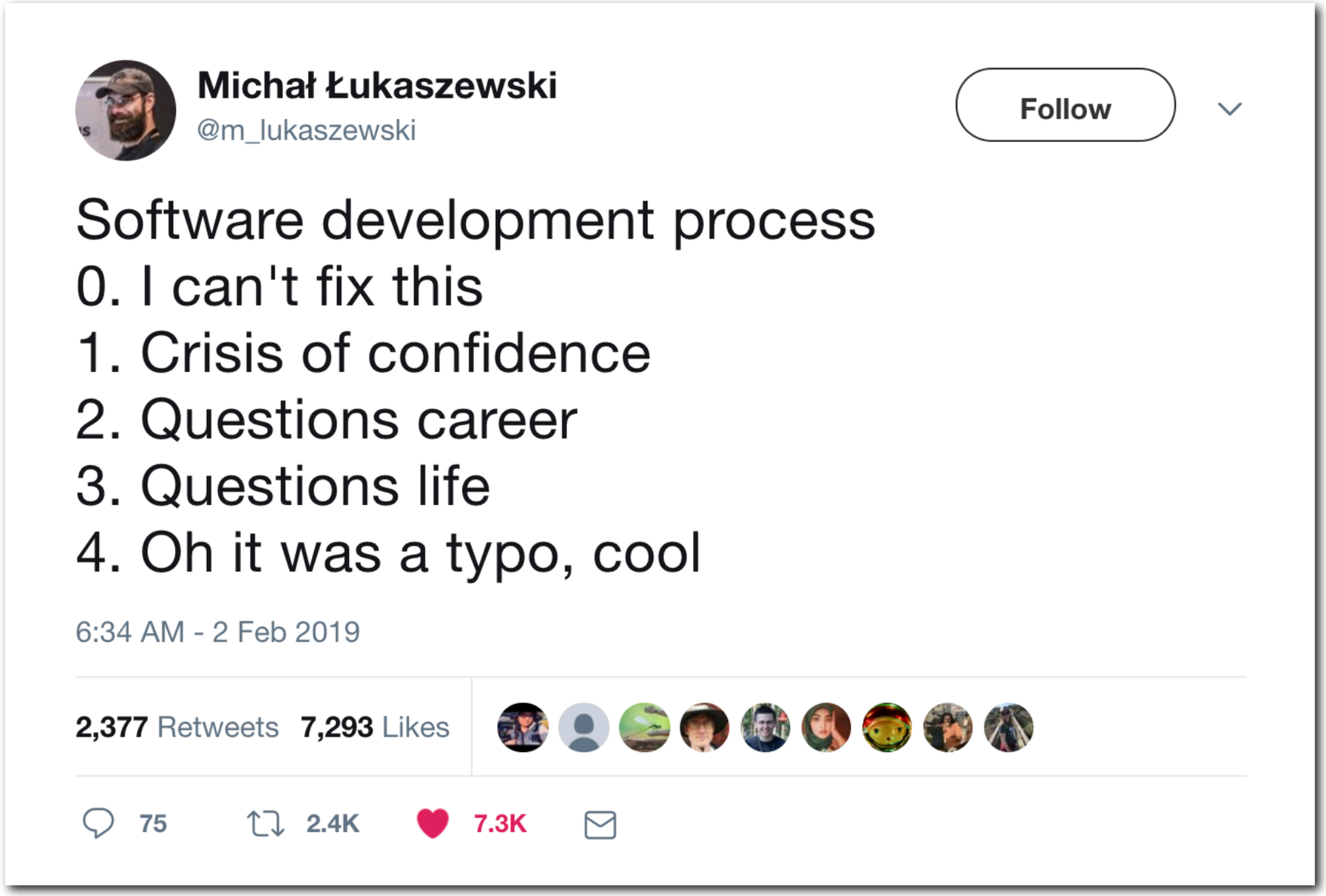
🔗 https://github.com/mgechev/codelyzer

Learning curve and docs



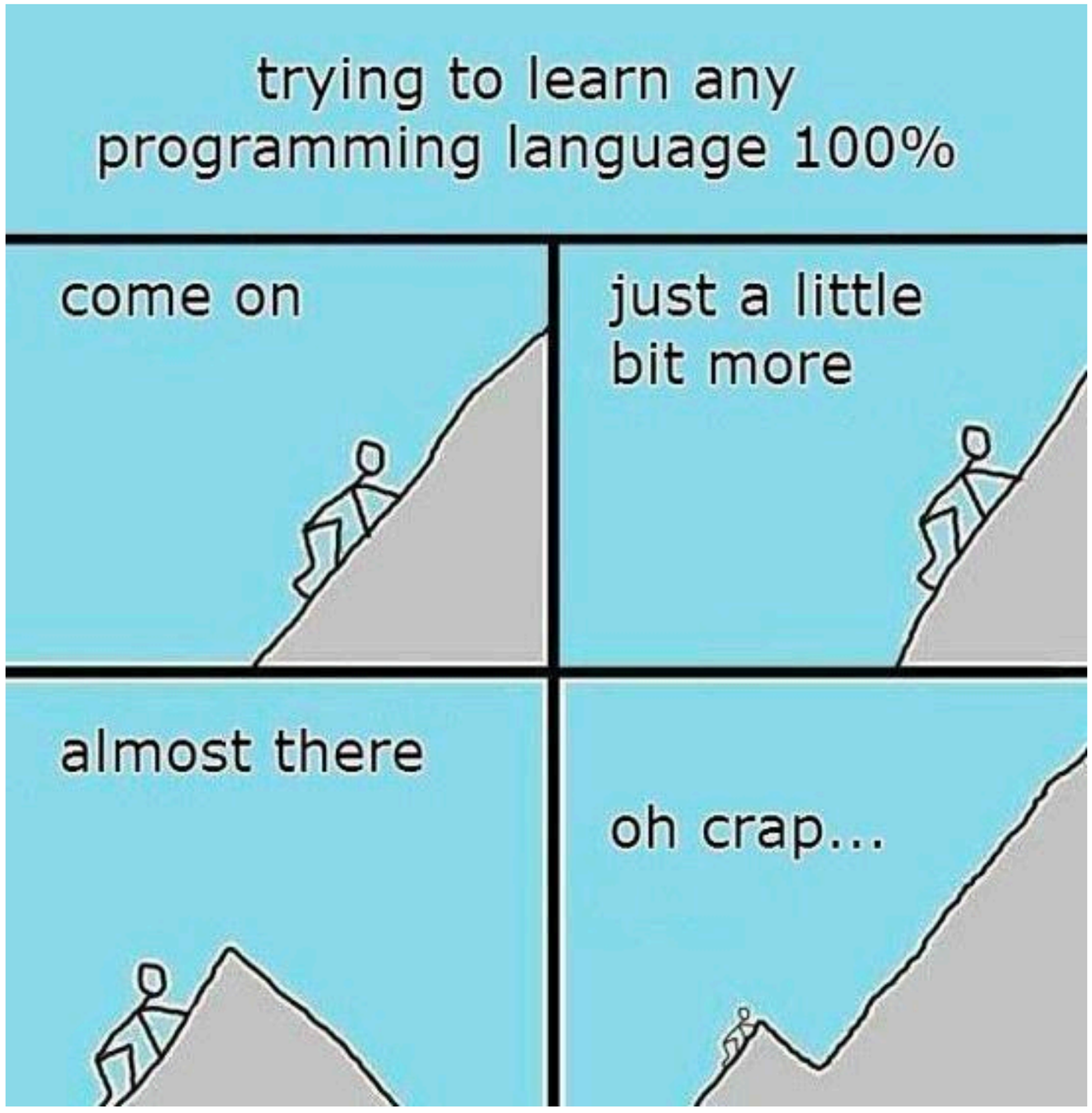
Senior developer looking for documentation of legacy feature

Sounds familiar? 😕🥺😢😭🙄



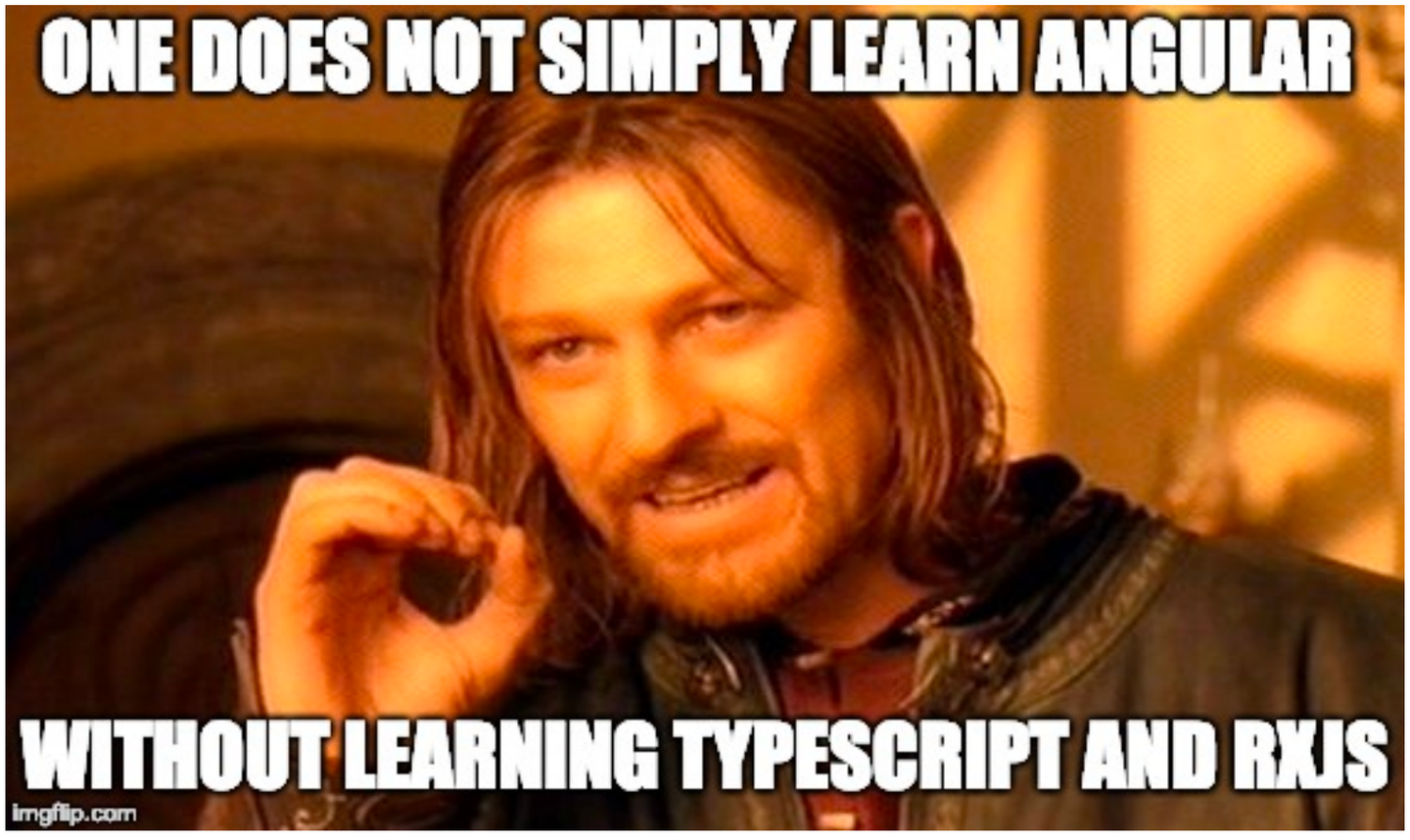
🔗 https://twitter.com/m_lukaszewski/status/1091630873315893248

Oh, that’s me.



🔗 https://www.reddit.com/r/ProgrammerHumor/comments/7x96ts/learning_a_new_programming_language/

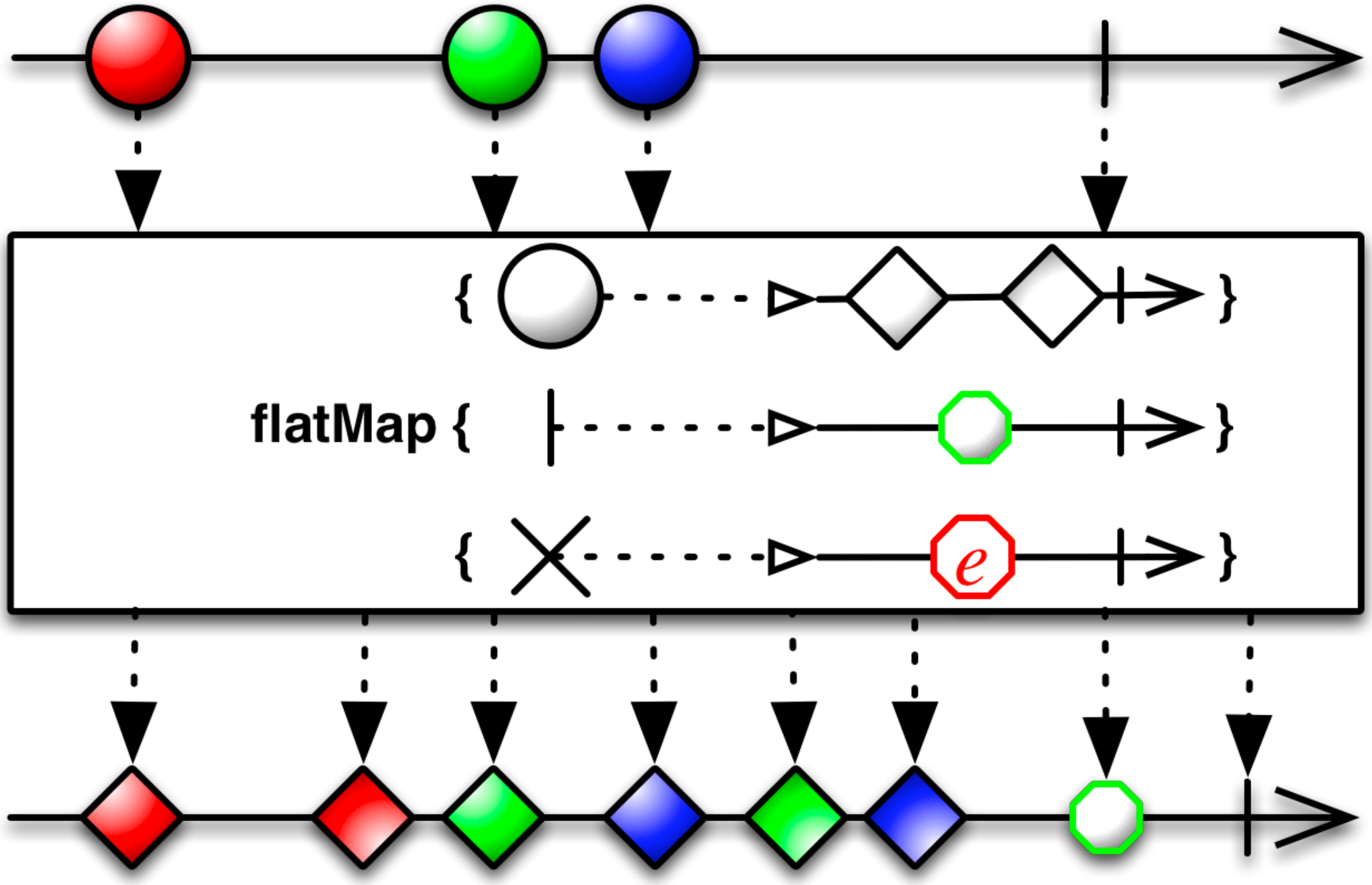
Thinks to consider when learning Angular



TS

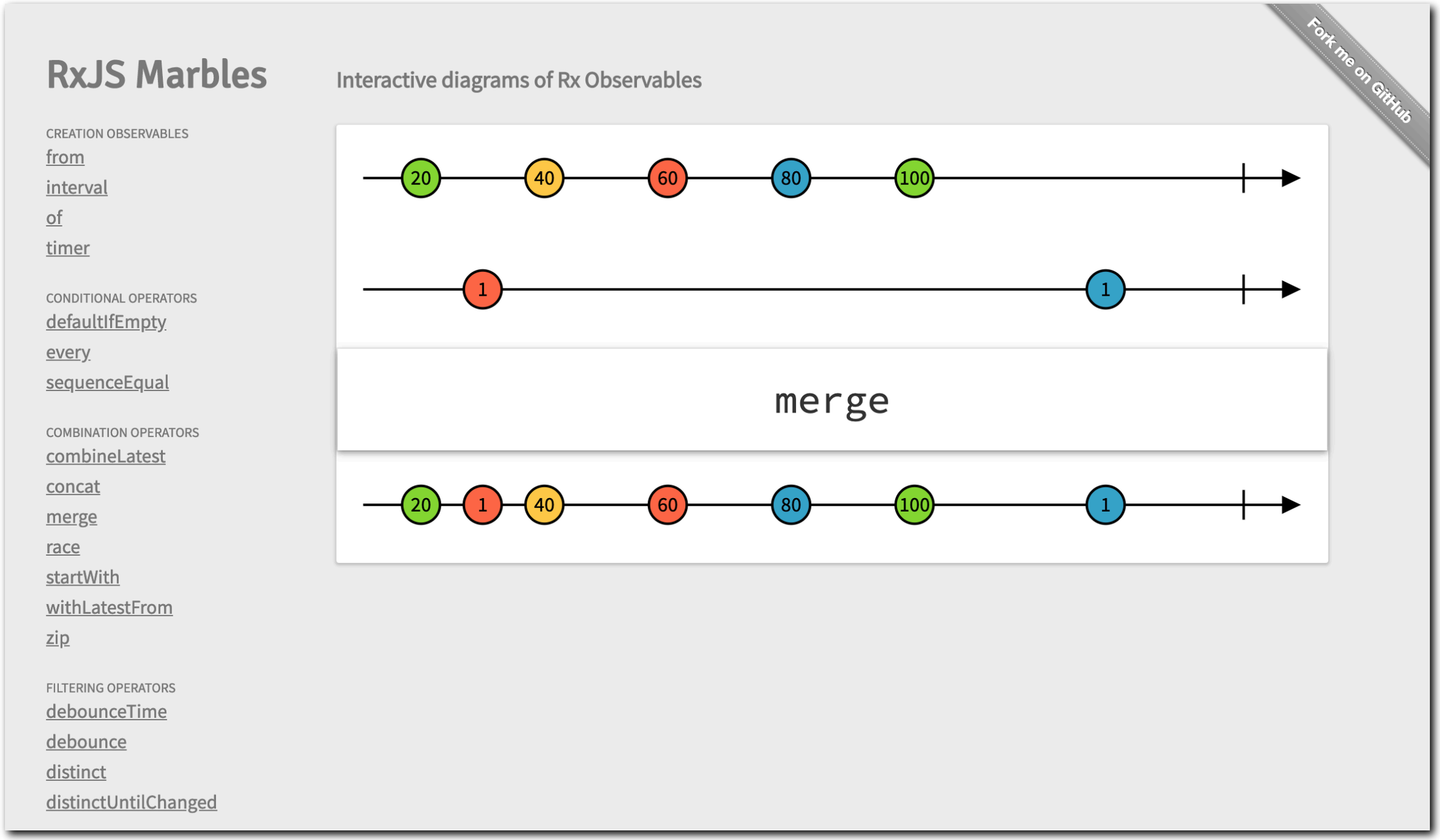
RxJS

RxJS 😵😵😵



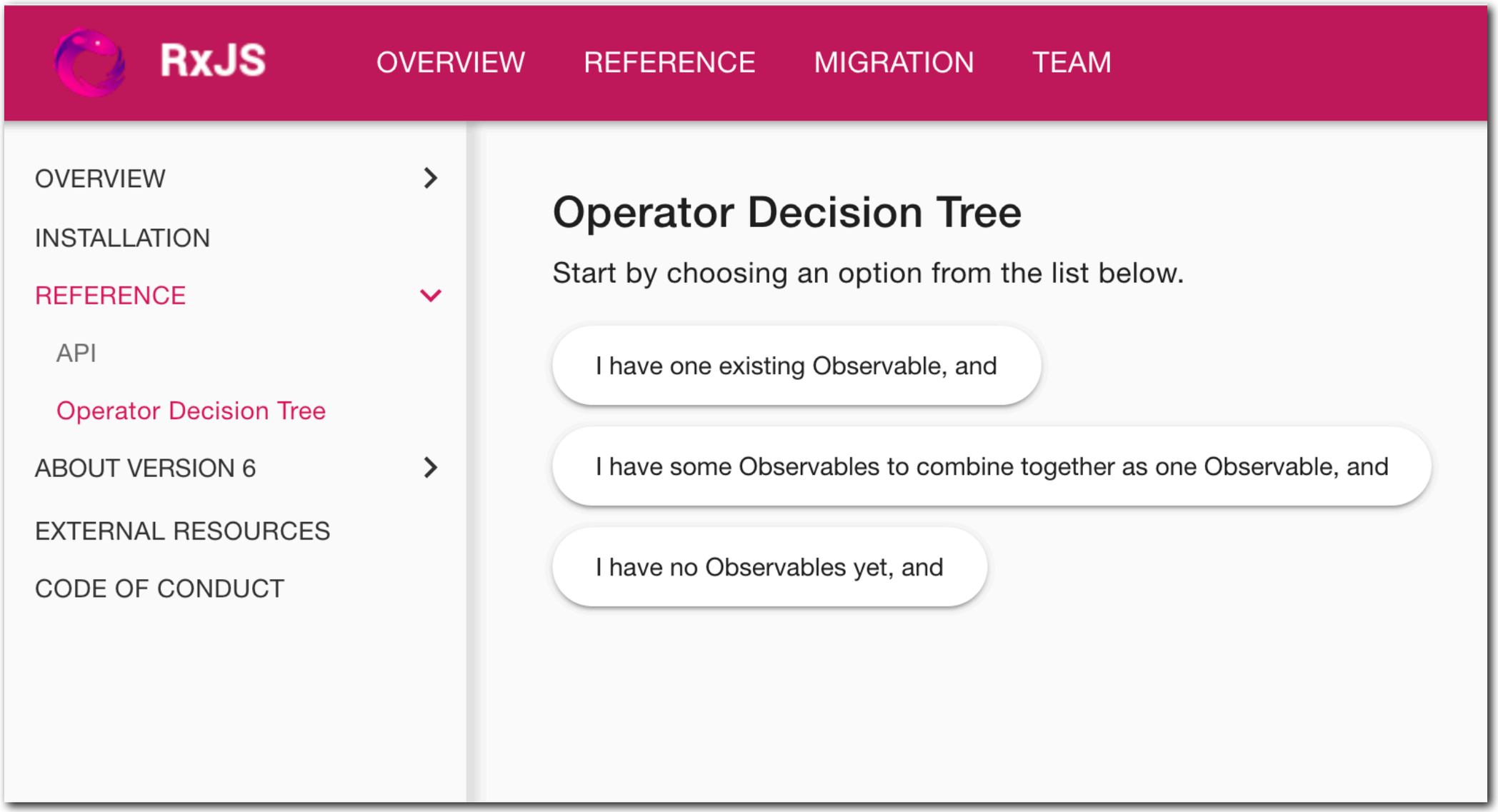
🔗 http://reactivex.io/documentation/operators/flatmap.html

RxJS Marbles FTW



🔗 https://rxmarbles.com/#merge

RxJS Operator Decision Tree

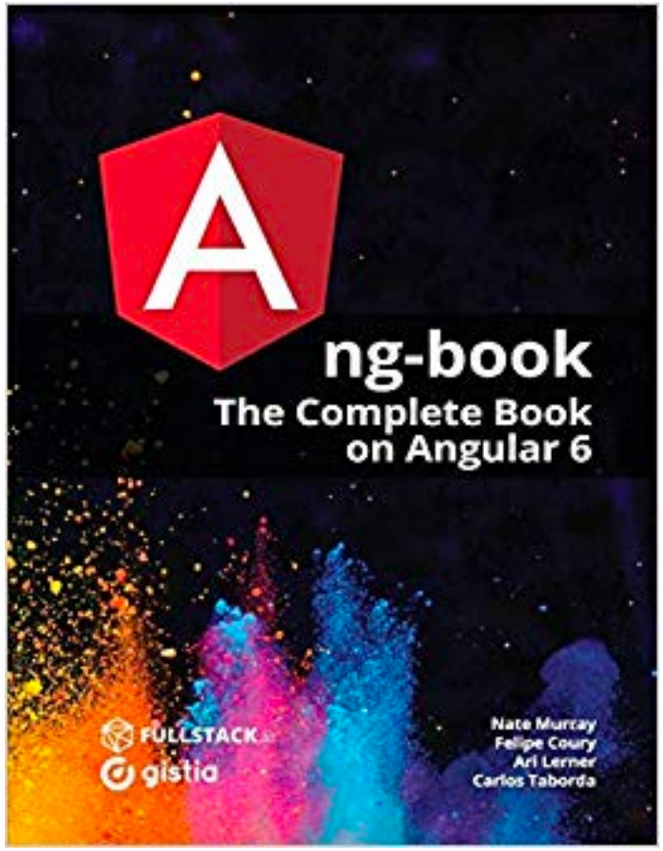


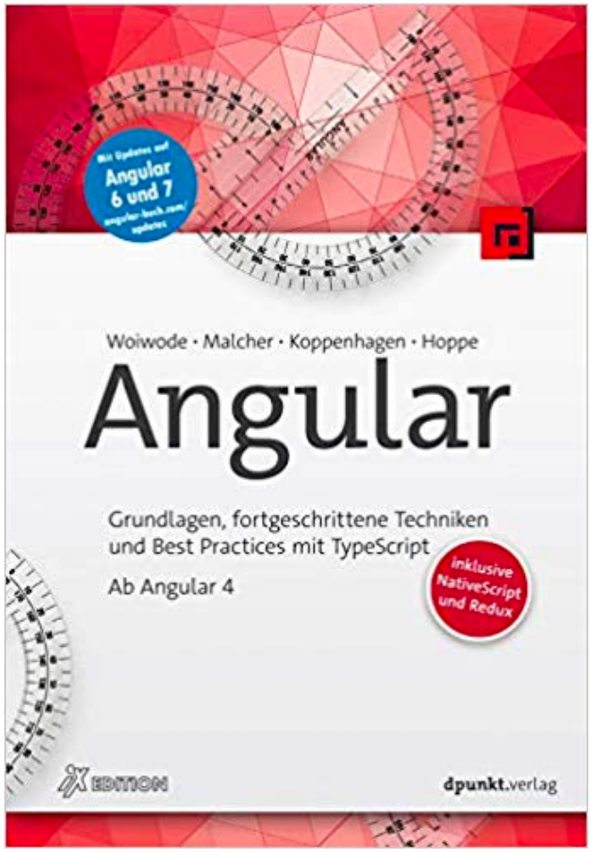
🔗 https://rxjs-dev.firebaseapp.com/operator-decision-tree

12-week web dev bootcamp graduation

🔗 http://classicprogrammerpaintings.com/post/145559207468/12-week-web-dev-bootcamp-graduation-edmund

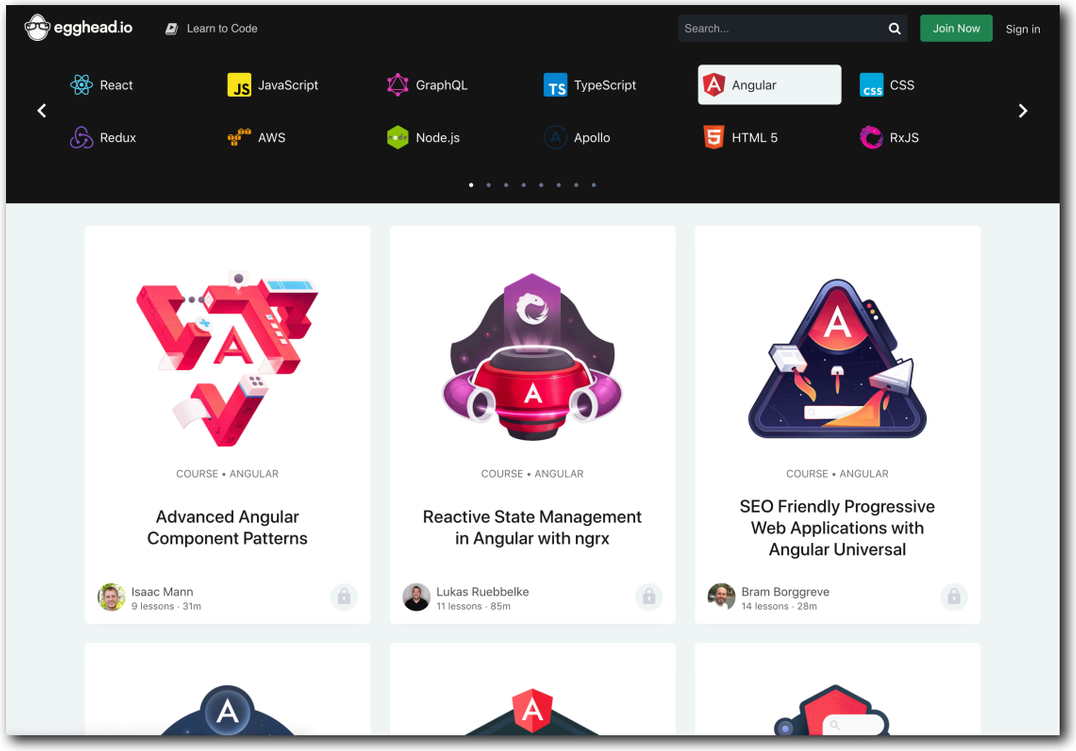
Books

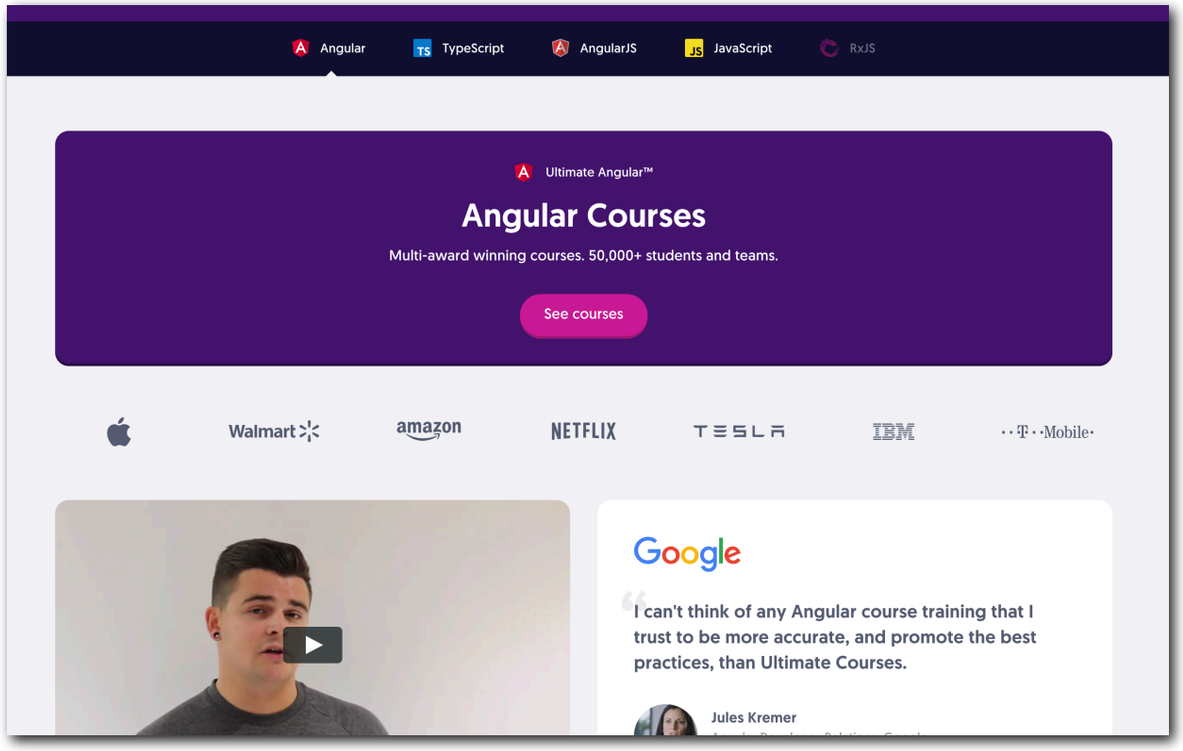

ng-book
The Complete Book
on Angular 6
FULLSTACK
gistia
Nate Murray
Felipe Coury
Ari Lerner
Carlos Taborda


Mit Updates auf
Angular
6 und 7
angular-buch.com
updates
Woiwode · Malcher · Koppenhagen · Hoppe
Angular
Grundlagen, fortgeschrittene Techniken
und Best Practices mit TypeScript
Ab Angular 4
inklusive
NativeScript
und Redux
iX EDITION
dpunkt.verlag


Inkl. Material Design
Templating,
Directives,
i18n, FormGr
RxJS, Depend
Injection,
WebSockets
Christoph Höller
Angular
Das umfassende Handbuch
Angular-Webapplikationen professionell entwickeln
Einführung, Praxis, TypeScript und RxJS
Formulare, Routing, HTTP-Anbindung, Animationen,
i18n, reaktive Anwendungen, Performance-Tuning
Für alle Versionen ab Angular 7
Rheinwerk
Computing

Online Courses





🔗 https://egghead.io/courses/for/angular

🔗 https://ultimatecourses.com/angular

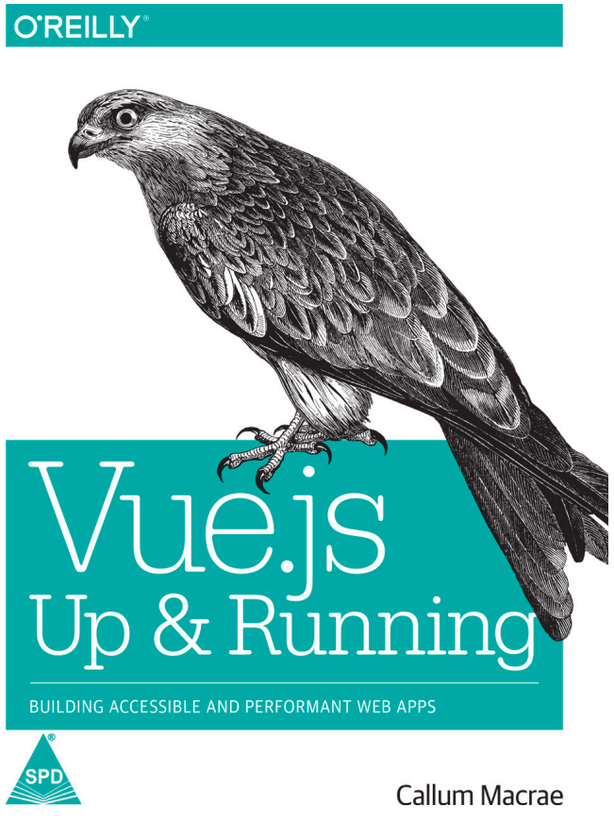
Thinks to consider when learning Vue

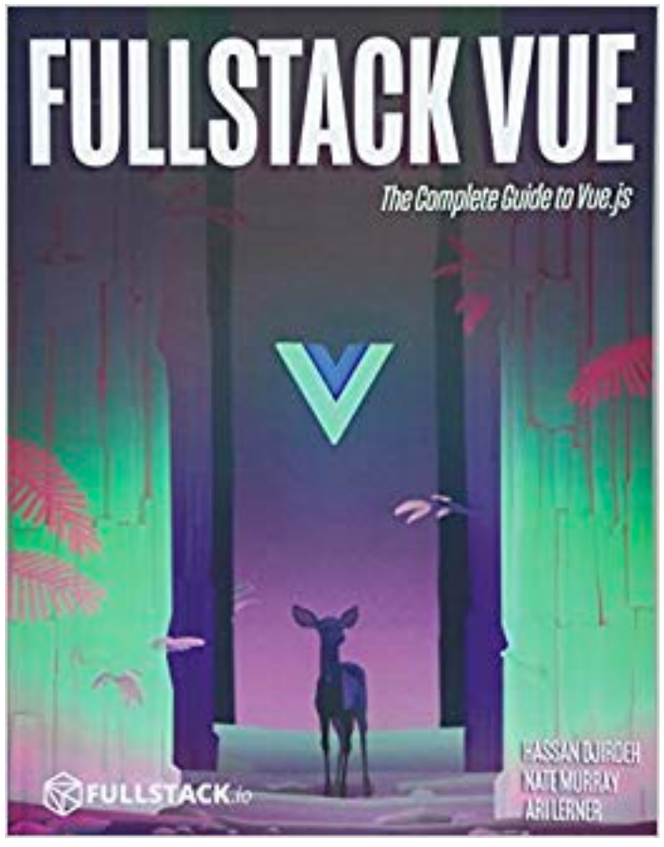
- No need for additional learning of other libraries or tooling (in the first place)

Vue resources

- https://vuejs.org/v2/guide/
- https://vuejs.org/v2/style-guide/
- https://vuejs.org/v2/cookbook/

Books


O'REILLY
Vue.js
Up & Running
BUILDING ACCESSIBLE AND PERFORMANT WEB APPS
SPD
Callum Macrae


FULLSTACK VUE
The Complete Guide to Vue.js
FULLSTACK.io
HASSAN DJIRDEH
NATE MURRAY
ARI LERNER

awesome-vue 🙌

vuejs / awesome-vue

Watch 2,046 | Star 42,062 | Fork 5,830

Code | Issues 60 | Pull requests 22 | Projects 0 | Insights

Branch: master | awesome-vue / README.md | Find file | Copy path

mya-ake feat: Add vue-slot-checker (#2660) e3ac514 14 days ago

1,000 contributors and others

Executable File | 2718 lines (2331 sloc) | 259 KB | Raw | Blame | History

🔗 https://github.com/vuejs/awesome-vue/

Quote from Evan You

„React kind of attracts a lot of functional oriented programmers. Vue attracts a lot of people who’d prefer the more classic HTML, CSS & JavaScript model of development. And Angular attracts a lot of people coming from a Java or C# enterprise background.“

🔗 https://blog.hackages.io/https-blog-hackages-io-evanyoubhack2017-cc5559806157

Too long, didn’t listen

André Staltz publishes „The introduction to Reactive Programming you've been missing“ on GitHub.

Vue in a nutshell 🐚

- Developed and maintained by the community
- Not opinionated
- Small core library for minimal setup (components, rendering, etc.)
- Large eco system of plugins and libs
- If you know HTML, CSS & JavaScript - you are ready to go
- Use VUE CLI for project setup, get inspiration from boilerplates to extend this setup

Angular in a nutshell 🐚

- All in one platform developed by google
- Totally opinionated
- Without learning TypeScript and RxJS, your Angular project will fail
- Regular releases (every ~6 months)
- Sophisticated patterns for large scale enterprise projects
- Scaffolding using the Angular CLI is pure joy