Vue.js:
The Progressive Framework

 vuejs.org

 @vuejs

 vuejs/vue

Evan You

@youyuxi @yyx990803

Currently: full-time open source!
Previously: Meteor, Google Creative Lab

Today

59,709 GitHub Stars

Top 10 All-Time

2nd most-starred JavaScript framework

~558k monthly NPM downloads

(excluding stats from mirrors in China)

Chrome DevTools Extension
~209k weekly active users

World-wide Commercial Usage

Community

314 GitHub Contributors

across the vuejs organization

Thriving Community Projects

Quasar Framework

iView

Muse-UI

Vux

Vuetify

Vue Material

Evolution

“Just a view layer library”

~~“Just a view layer library”~~

The Progressive Framework

Frameworks are designed to help us deal with complexity.

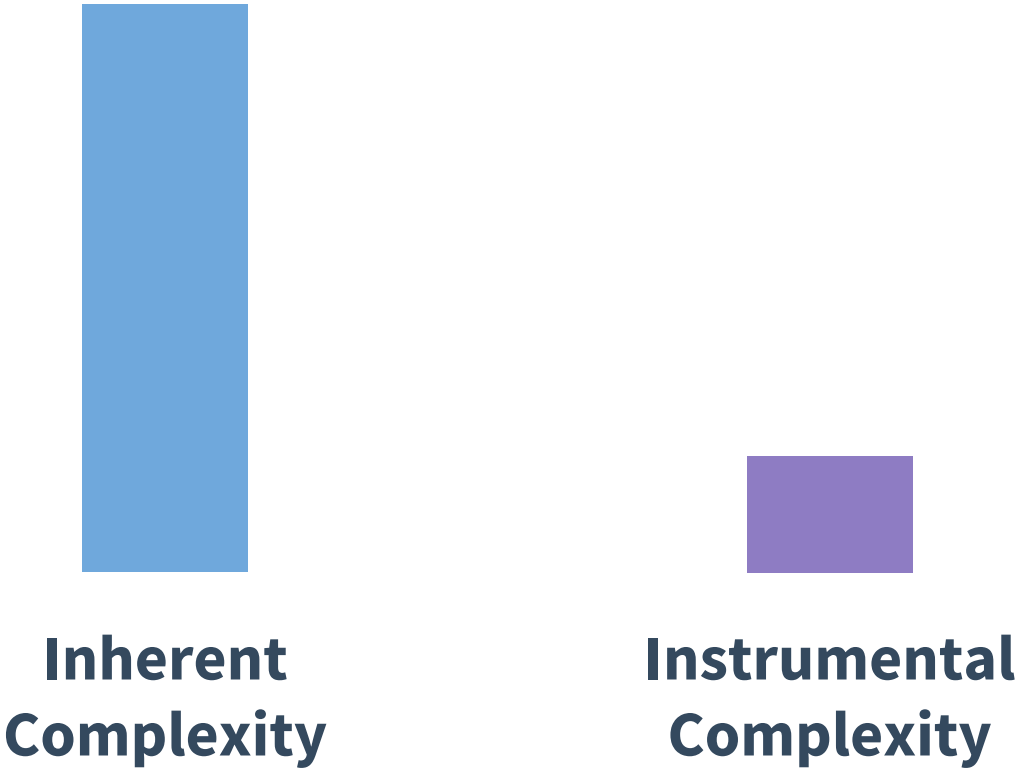
...but frameworks themselves can also introduce complexity of their own.

Application Complexity
vs.
Framework Complexity

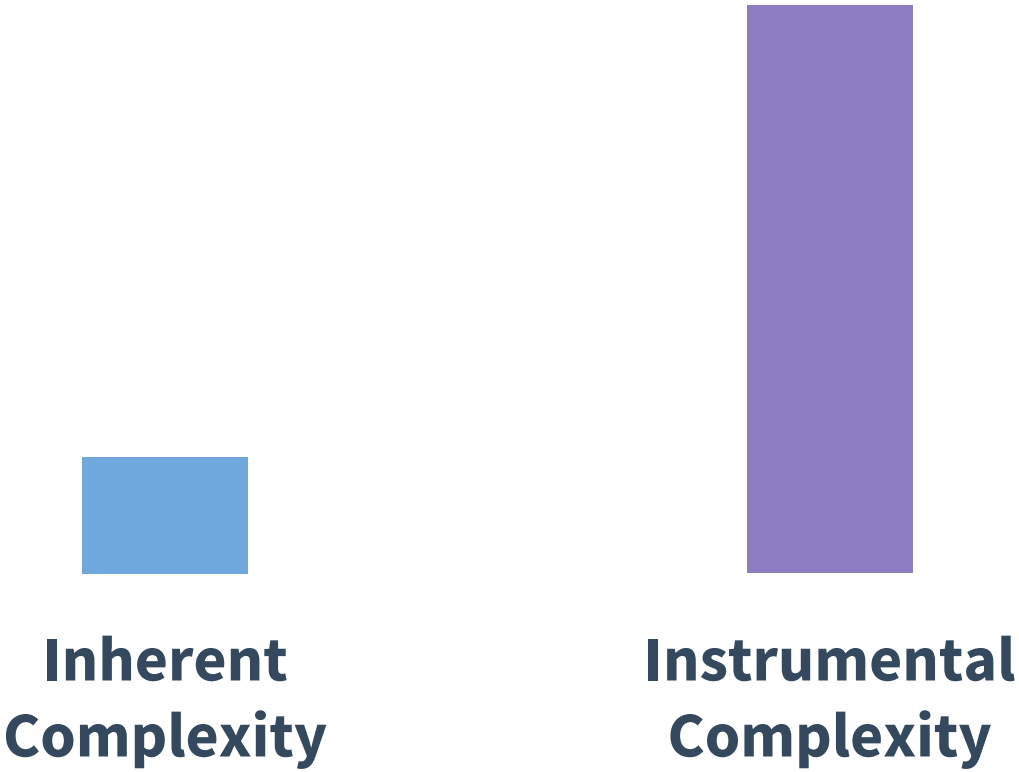
Inherent Complexity
vs.
Instrumental Complexity

Instrumental Complexity is the price we pay for the tools’ capacity in handling Inherent Complexity.

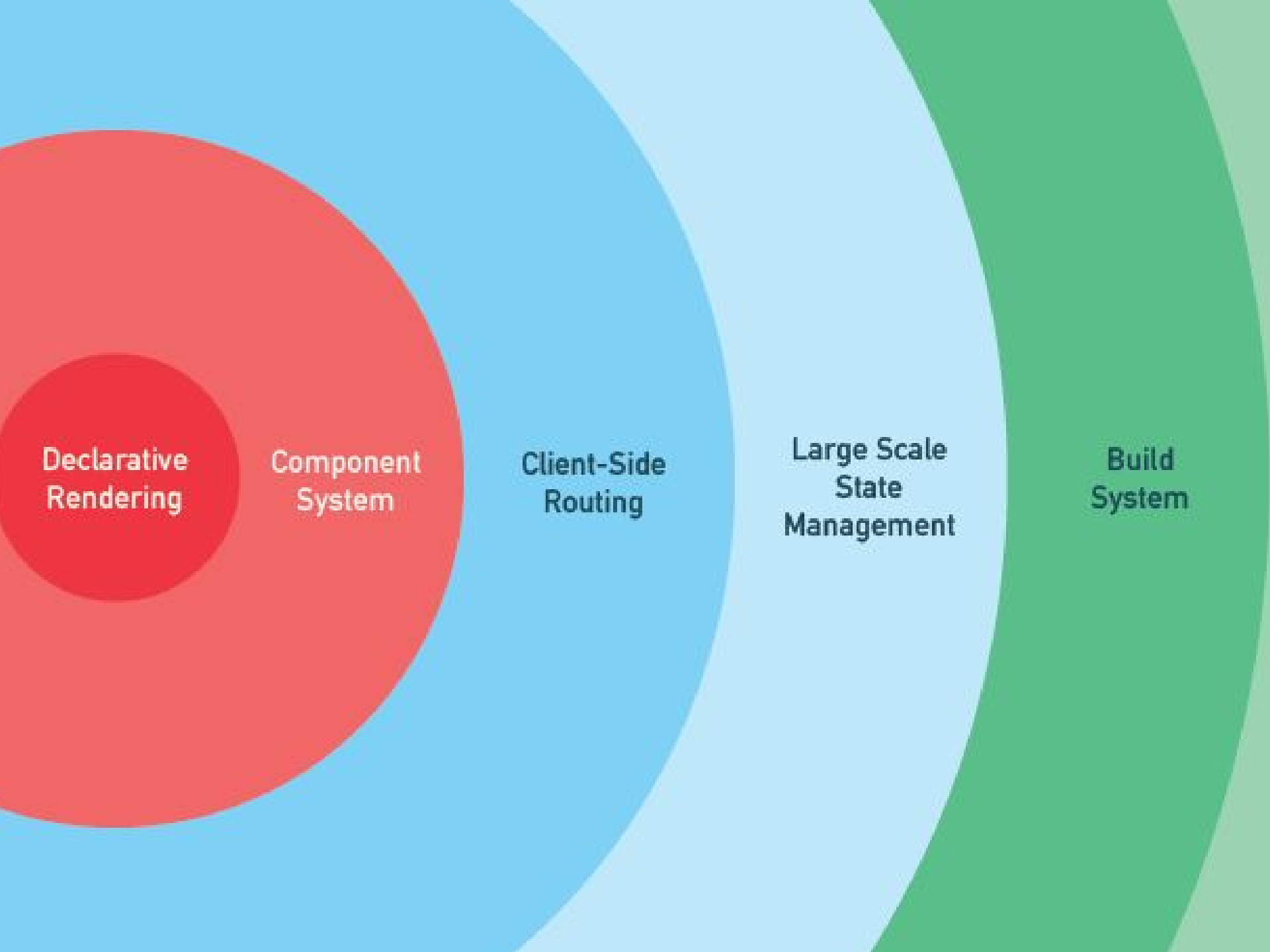
Insufficient



Overkill

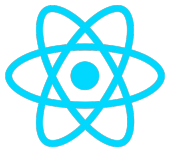


“Pick the right tool for the job”


Declarative Rendering
Component System
Client-Side Routing
Large Scale State Management
Build System

The Framework Spectrum

Templating Engines



React

Vue

Backbone

Angular

Ember

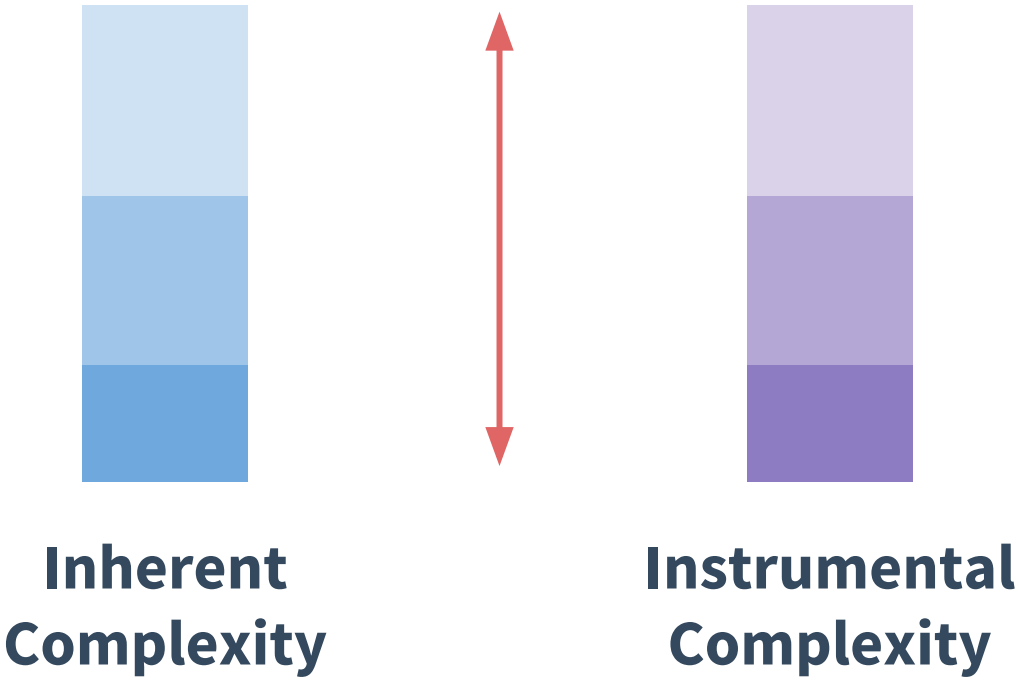
Meteor

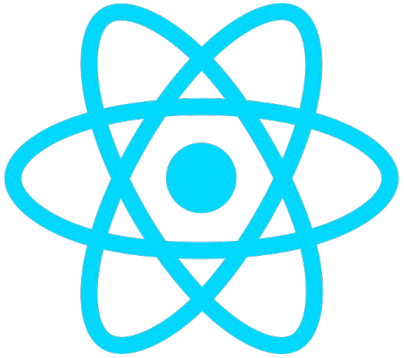
Less

More

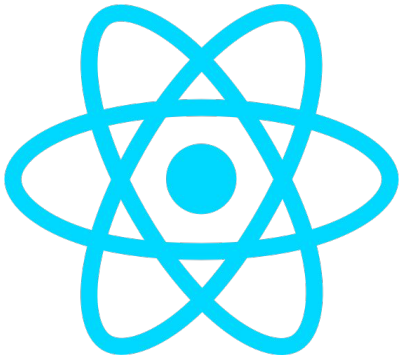

With supporting libraries
React+
Vue+
Templating Engines
React
Vue
Backbone
Angular
Ember
Meteor
Less
More

Scalability is Not a One-Way Street





View Layer Core
+
Optional Support Libraries



View Layer Core

+

Optional Support Libraries?

Eric Clemmons Follow
Creator of React Resolver, Genesis/Evolution for WordPress. Purveyor of a better Developer Experience…
Dec 26, 2015 · 4 min read

Javascript Fatigue

A few days ago, I met up with a friend & peer over coffee.

> *Saul: "How's it going?"*
>
> *Me: "Fatigued."*
>
> *Saul: "Family?"*
>
> *Me: "No, Javascript."*

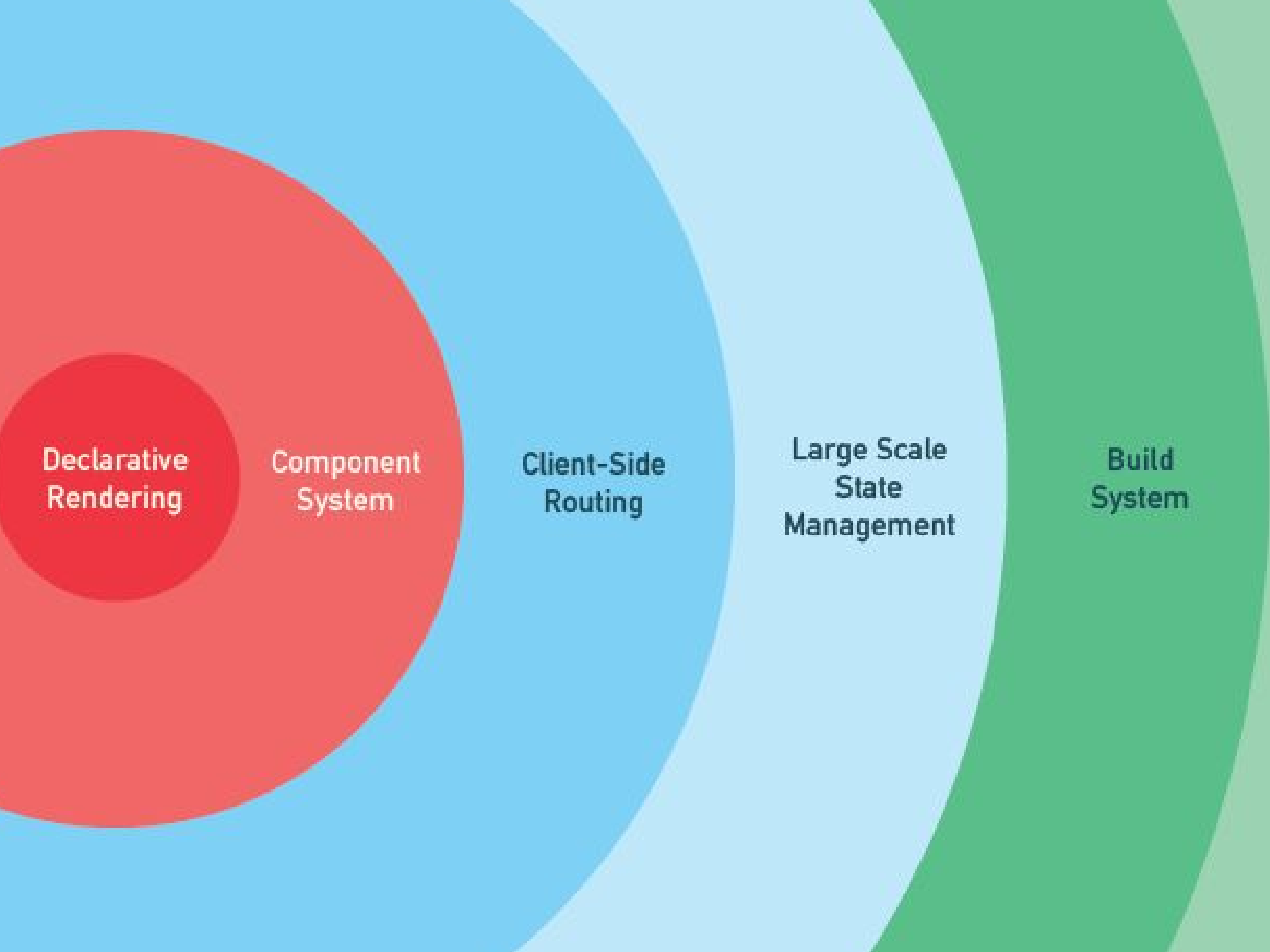
The Progressive Framework

pro·gres·sive

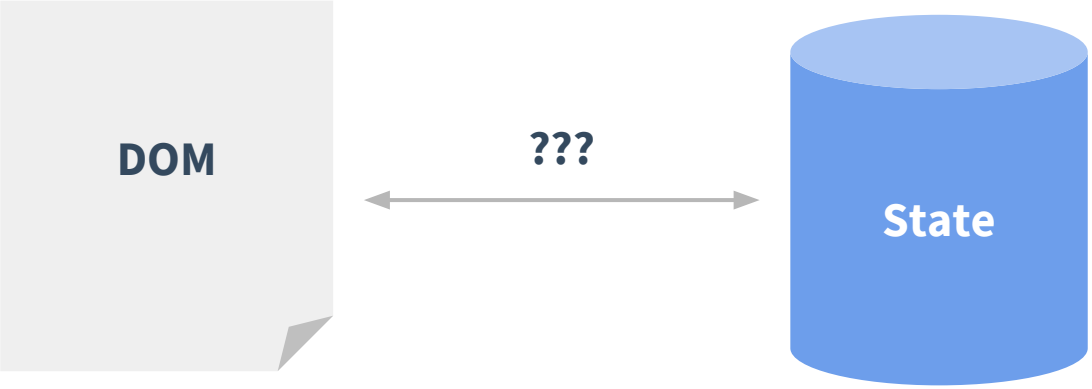
/prəˈgresiv/

adjective

1. happening or developing gradually or in stages; proceeding step by step.
   "a progressive decline in popularity"
   synonyms: continuing, continuous, increasing, growing, developing, ongoing, accelerating, escalating; More


Declarative Rendering
Component System
Client-Side Routing
Large Scale State Management
Build System

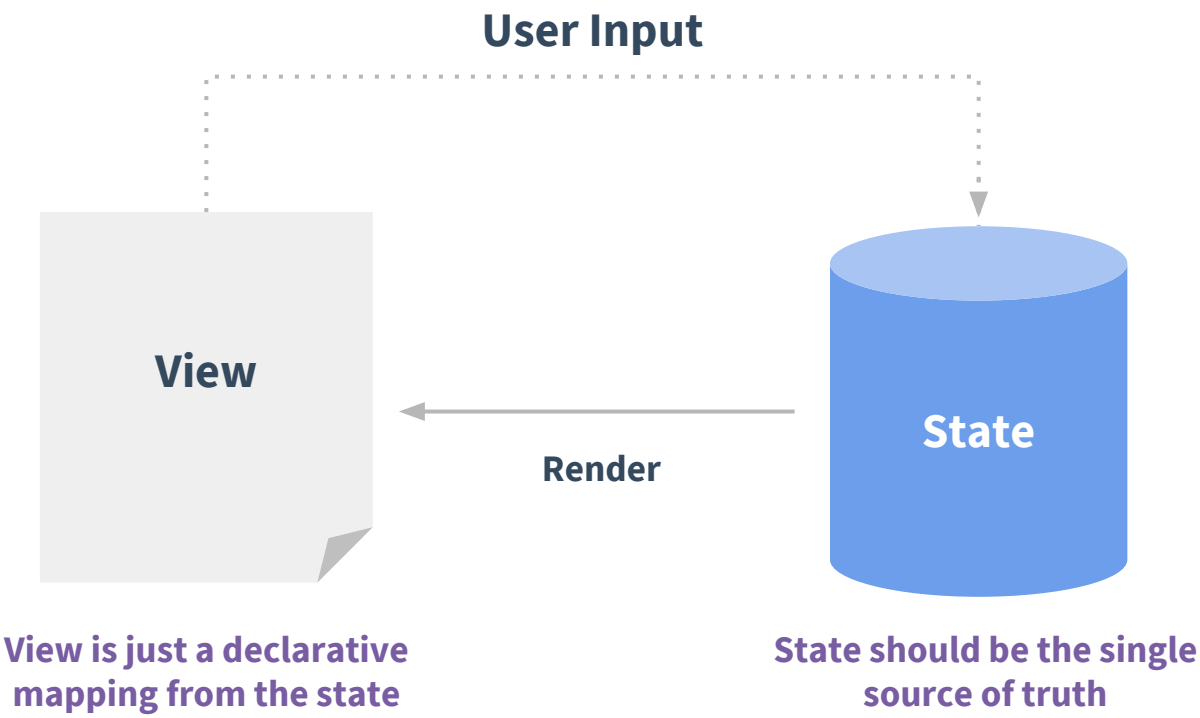
Declarative Rendering


DOM
???
State

Problems with the DOM

- Re-rendering entire chunks of DOM is expensive and disruptive
- Imperatively keeping the DOM in sync with the state is tedious and error-prone

Declarative & Reactive Rendering



Todo List Demo

(what else can we build?)

```
<div id="app">
  <ul>
    <li
      v-for="todo in todos"
      v-on:click="todo.done = !todo.done">
      {{ todo.title }} ({{ todo.done ? 'done' : 'nah' }})
    </li>
  </ul>
</div>
```

Hello World in a plain HTML file

```html
<script src="https://unpkg.com/vue/dist/vue.js"></script>

<div id="app">
  <p>{{ message }}</p>
</div>
```

```vue
new Vue({
  el: '#app',
  data: {
    message: 'Hello Vue.js!'
  }
})
```

Templates

vs.

JSX

vs.

Render Functions

“JavaScript in HTML”

vs.

“HTML in JavaScript”

vs.

“Just JavaScript”

Templates

Pros:

- Can be enhanced from plain HTML
- Better reflect visual/semantic structure
- Designer friendly

Cons:

- Not as flexible as a real programming language
- It’s all strings. Difficult to take full advantage of JS static analysis tools.

JSX / Render Functions

Pros:

- Full flexibility of JavaScript
- Can fully leverage existing JavaScript tooling

Cons:

- Too much flexibility leads to
  - Harder to scan visual/semantic structure of content
  - stylistic bikeshedding

Evan You
@youyuxi

HN thread about Vue -> Obligatory comment “I like React because JSX” -> thread turns into JSX formatting bikeshedding. me: facepalm

RETWEETS **20** LIKES **191**

10:29 PM - 7 Feb 2017

13 20 191

But more importantly:

- Both are ways to declaratively map state to desired render output
- Their pros and cons are actually complementary to each other
- Developer background and mindset affect our effectiveness with a certain programming model.

This is why Vue 2 supports both.

Template -> [Parser] -> AST -> [Codegen] -> Render Function

vue-template-explorer.now.sh

vue-template-explorer

Evan

Secure https://vue-template-explorer.now.sh/#%3Cdiv%20id%3D%...

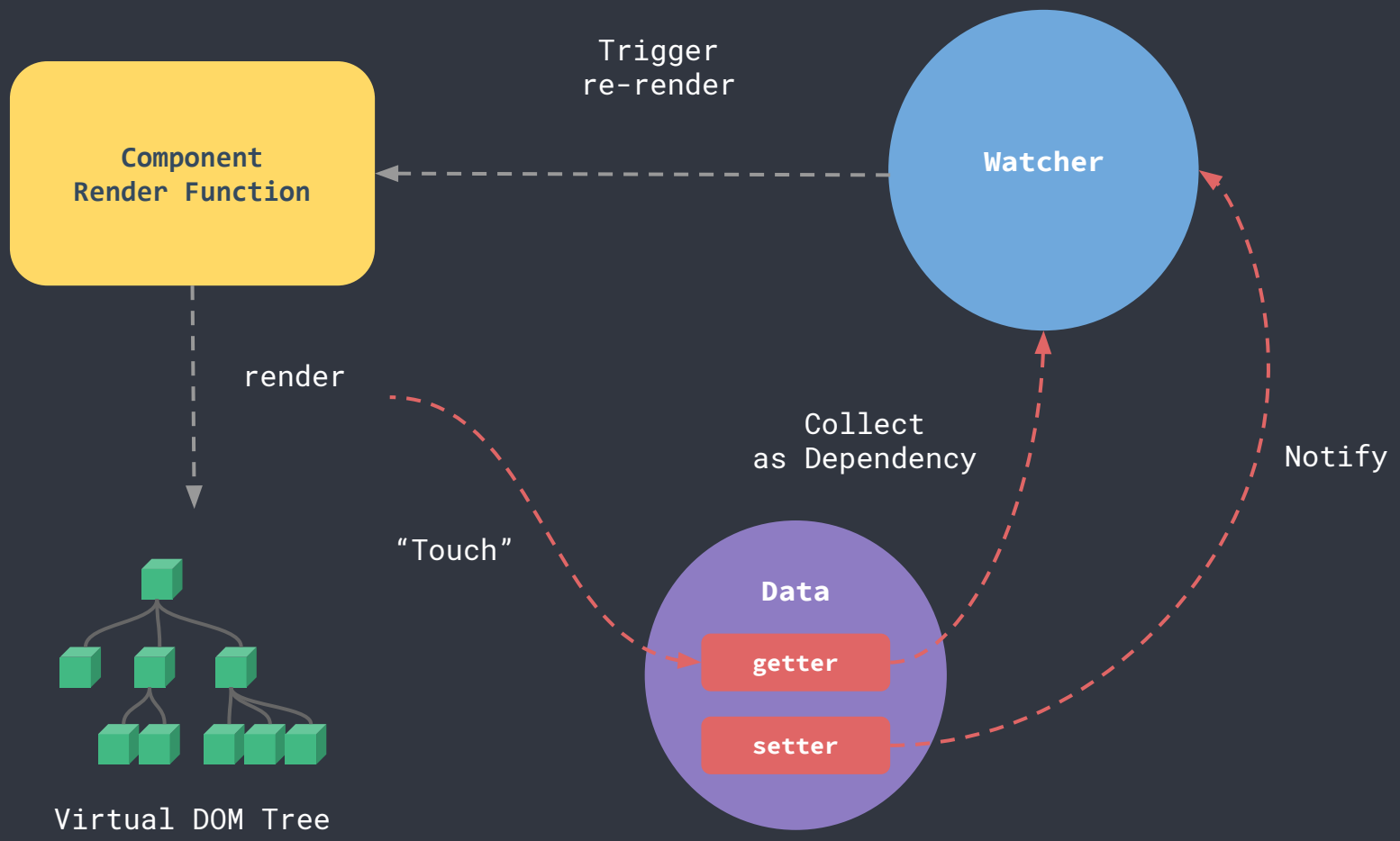
Vue Template Explorer (Vue version: 2.1.10)

Strip with?

```
1 <div id="app">{{ msg }}</div>
```

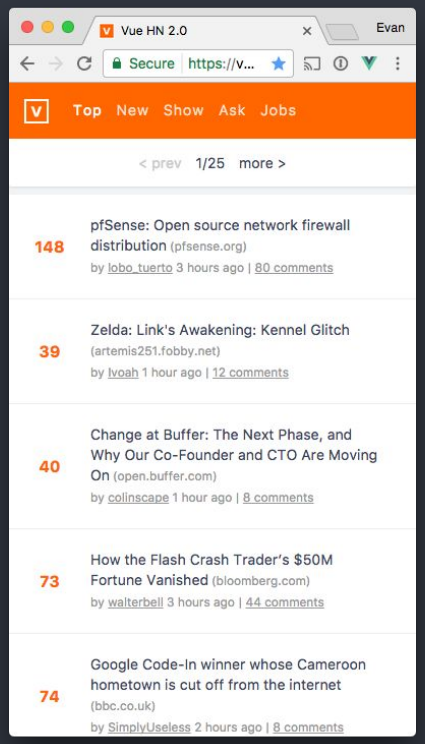
```
function render() {
  with(this) {
    return _c('div', {
      attrs: {
        "id": "app"
      }
    }, [_v(_s(msg))])
  }
}
```

```
Template
 |
[ Base Compiler ]
 |
Render Function (using `with`)
 |
[ Post Compiler ]
 |
Render Function (`with` stripped)
```

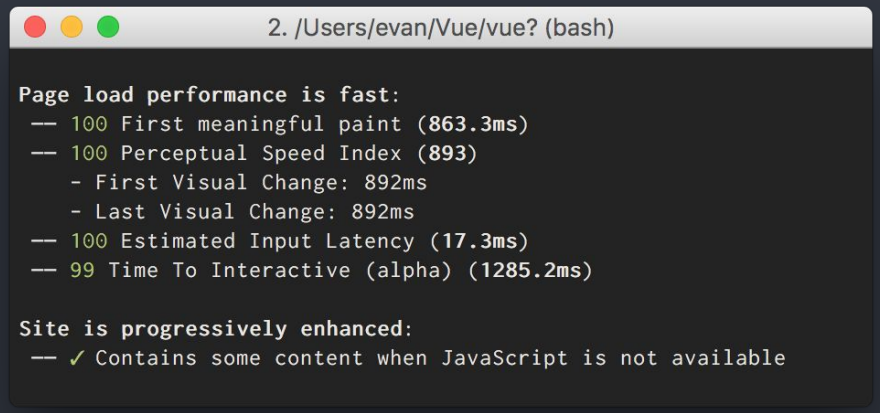

Trigger
re-render
Component
Render Function
Watcher
render
Collect
as Dependency
Notify
"Touch"
Data
getter
setter
Virtual DOM Tree

Server-Side Rendering

vue-hn.now.sh

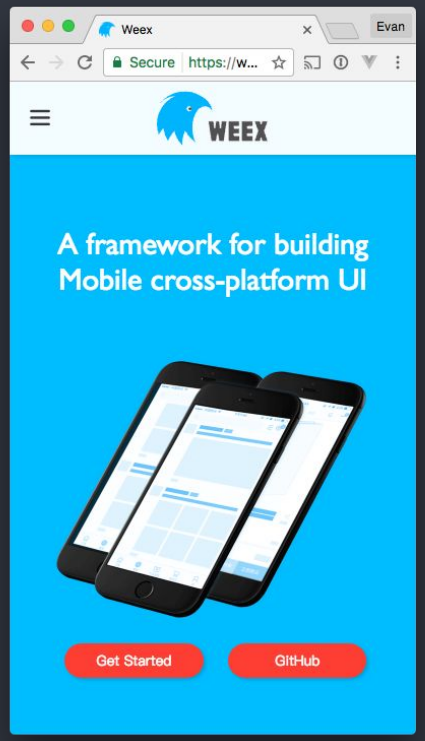


Lighthouse audit results

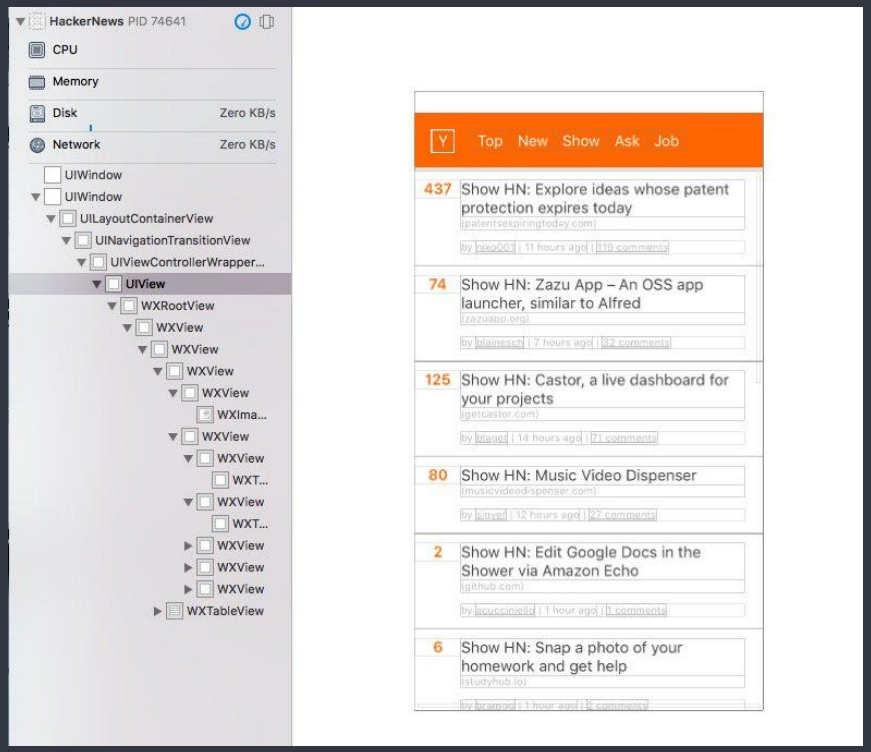


Native Rendering via Weex Project

weex-project.io



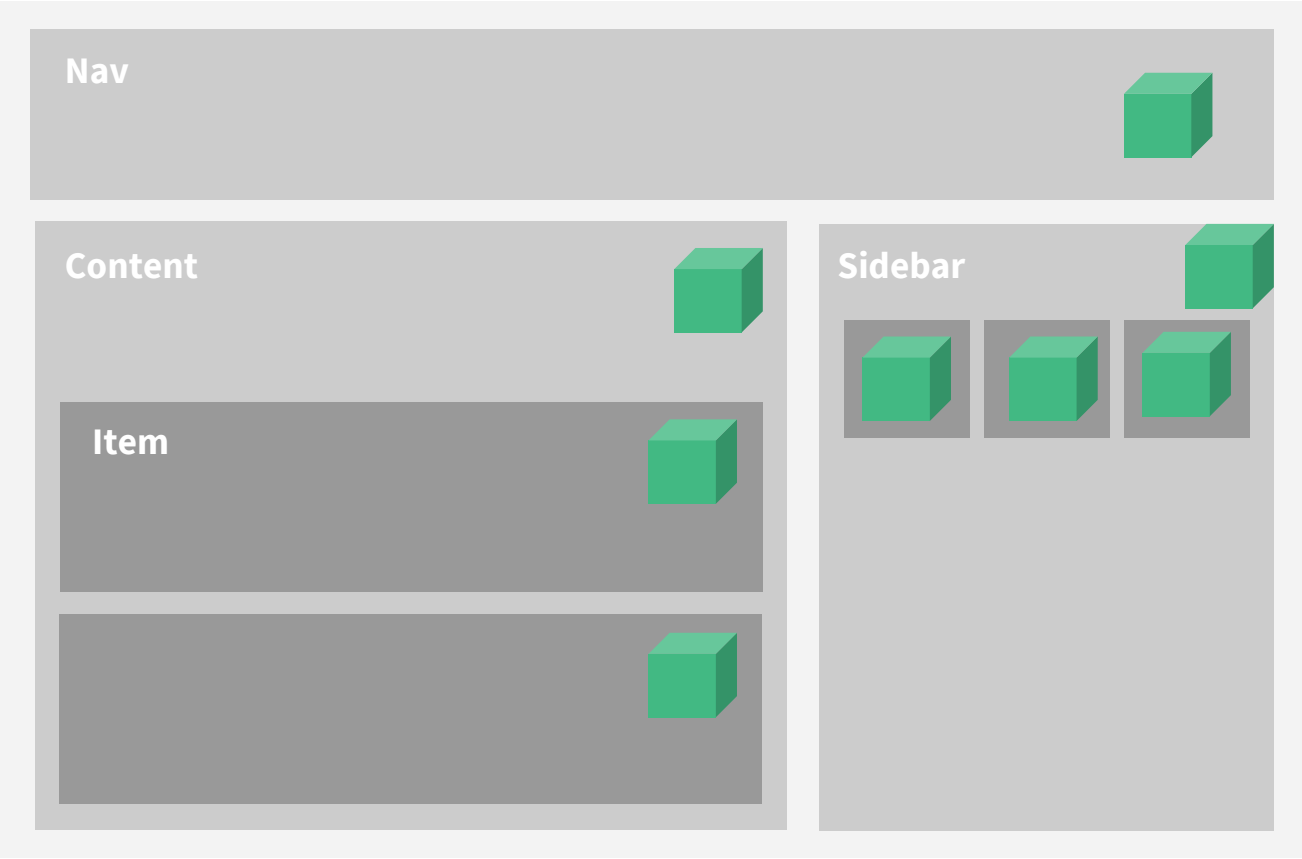
HN demo implemented with Weex



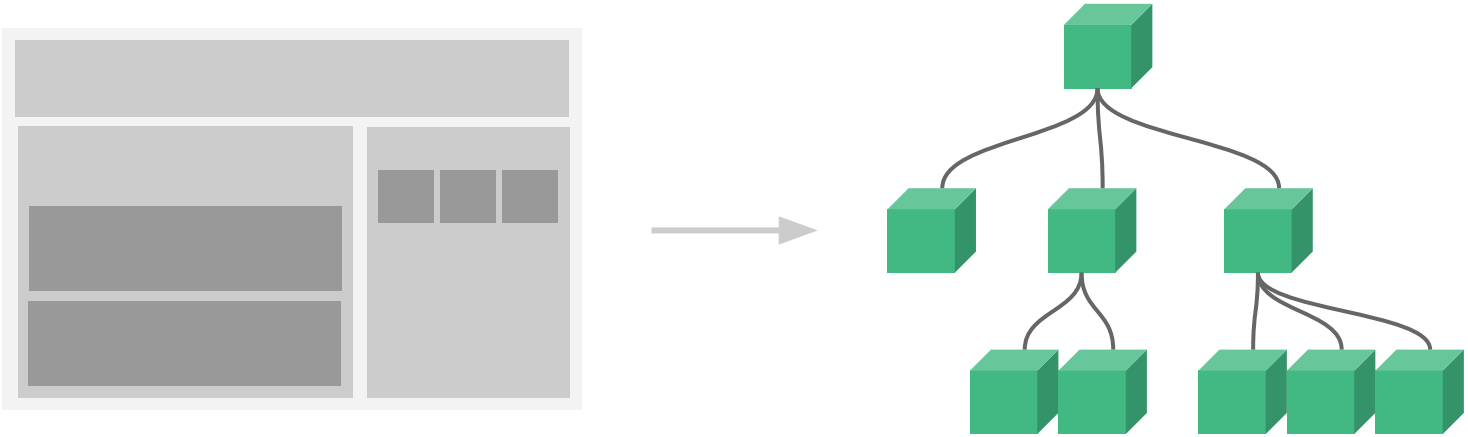
Component System

Most App UIs can be broken down into components

Every component is responsible for managing a piece of DOM

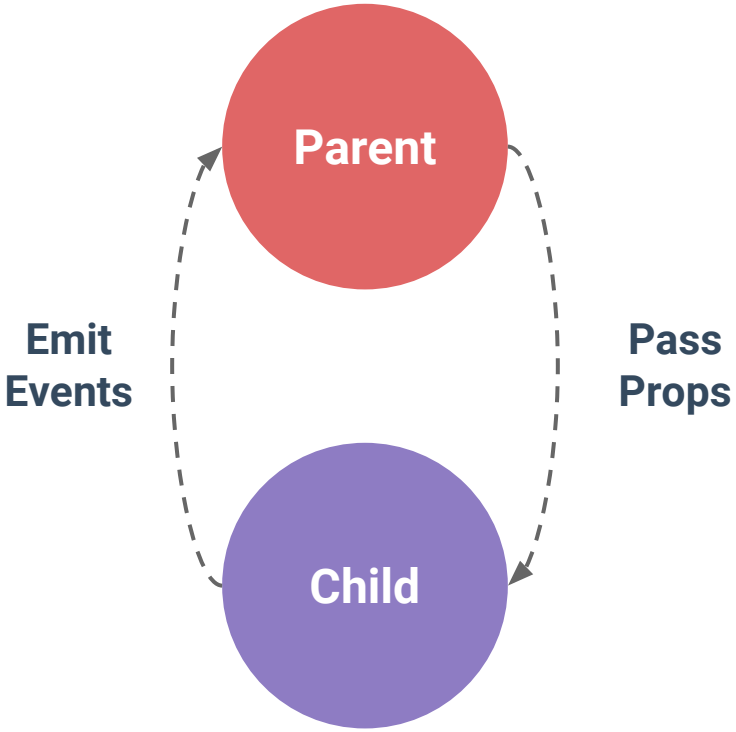


The entire UI can be abstracted
into a tree of components



Component Communication:

Props in, Events out

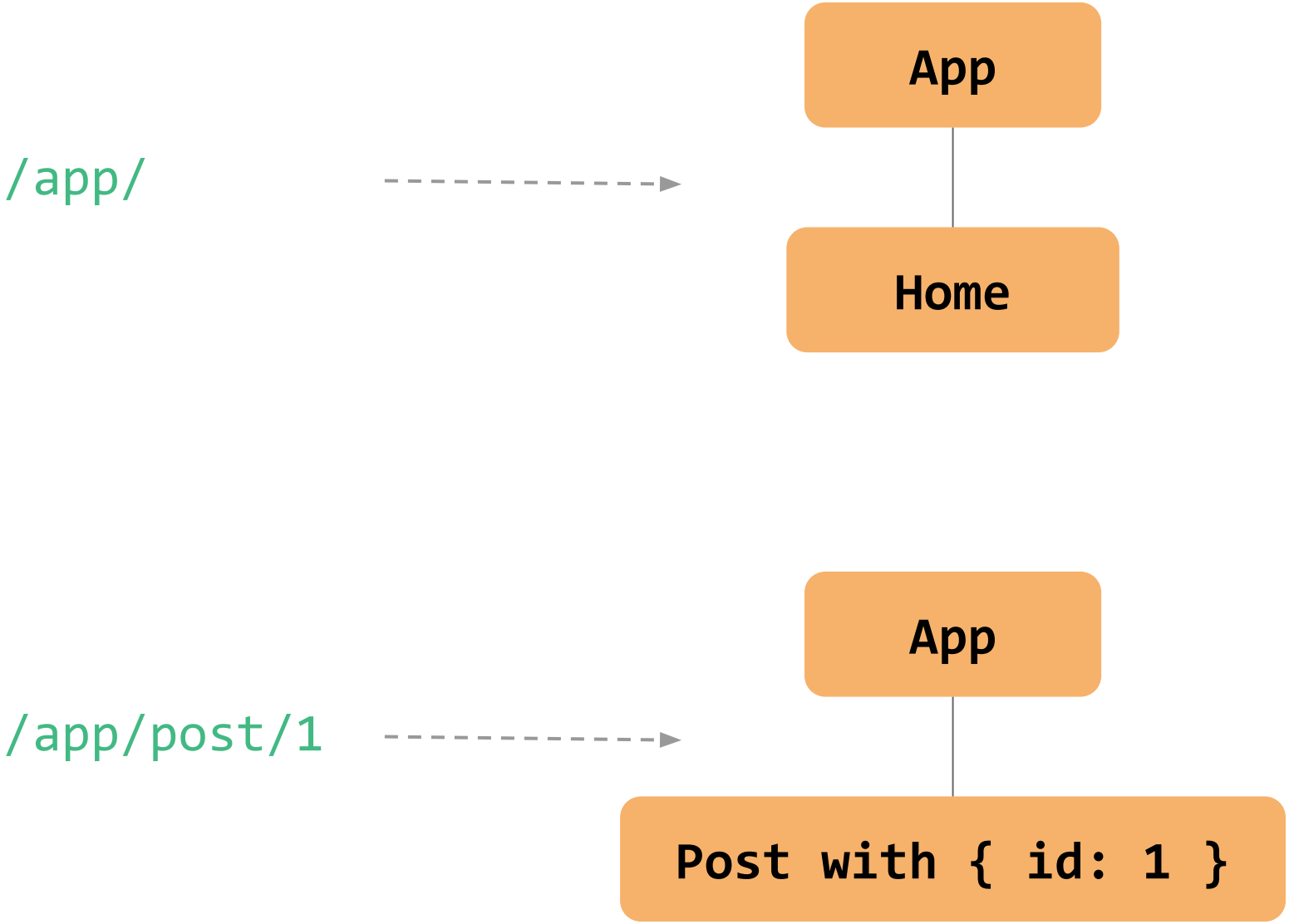


Todo demo using components

```
<div id="app">
  <ul>
    <todo
      v-for="todo in todos"
      :todo="todo"
      @changed="todo.done = !todo.done">
    </todo>
  </ul>
</div>
```

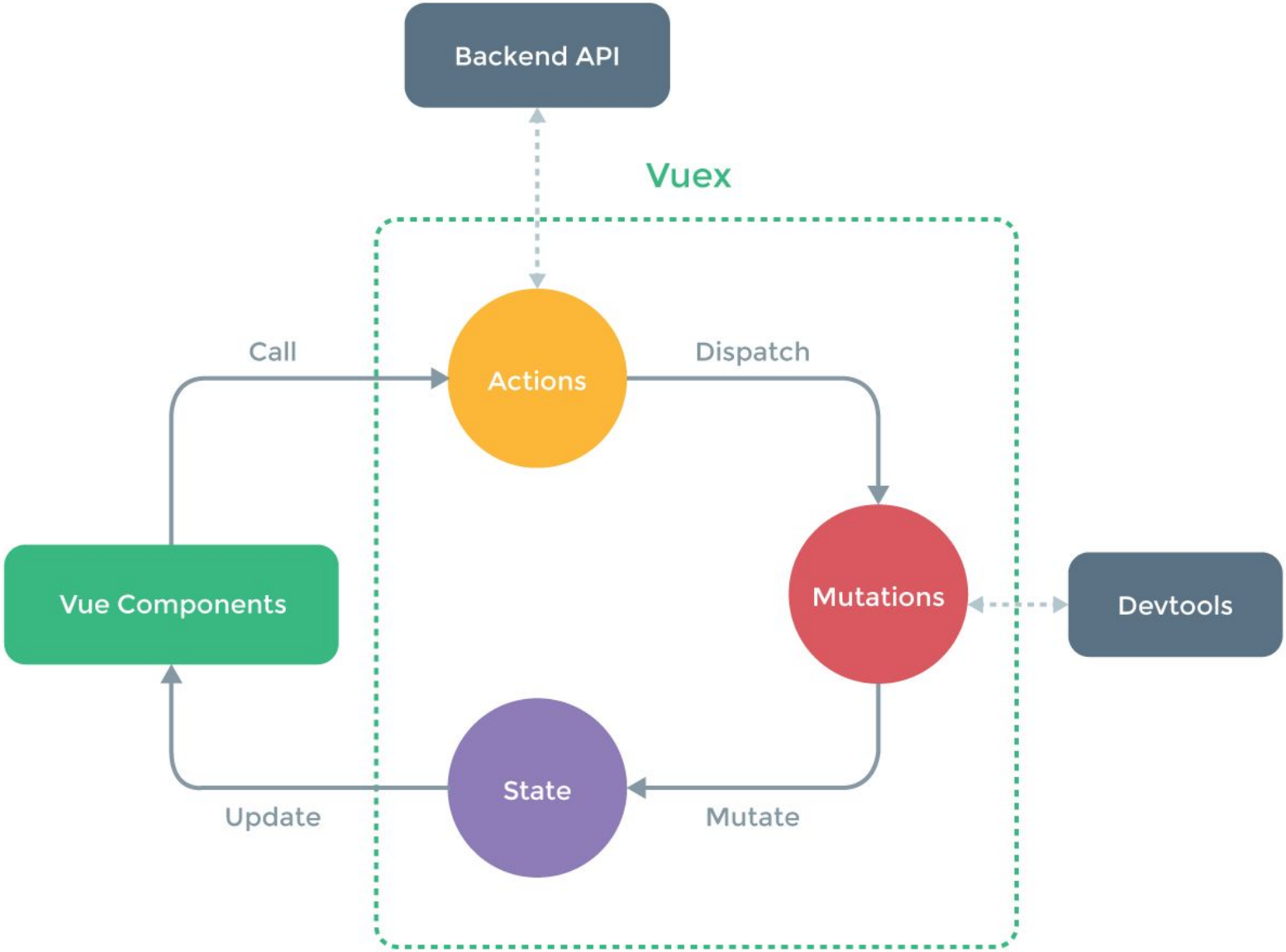
Client-Side Routing

https://github.com/vuejs/vue-router


/app/
App
Home
/app/post/1
App
Post with { id: 1 }

Large-Scale State Management

https://github.com/vuejs/vuex


Backend API
Vuex
Call
Actions
Dispatch
Vue Components
Mutations
Devtools
State
Update
Mutate

Build System
/
Development Experience

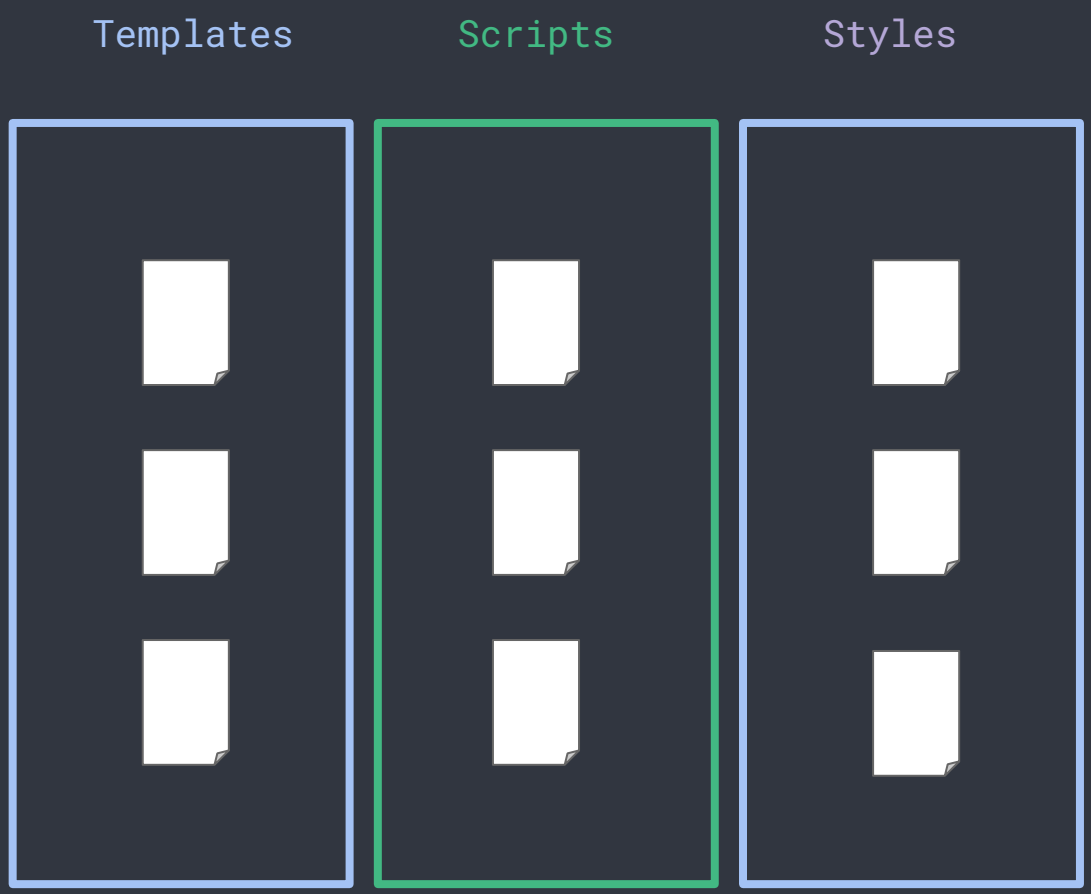
```
npm install -g vue-cli
vue init webpack-simple my-app
cd my-app
npm install
npm run dev
```

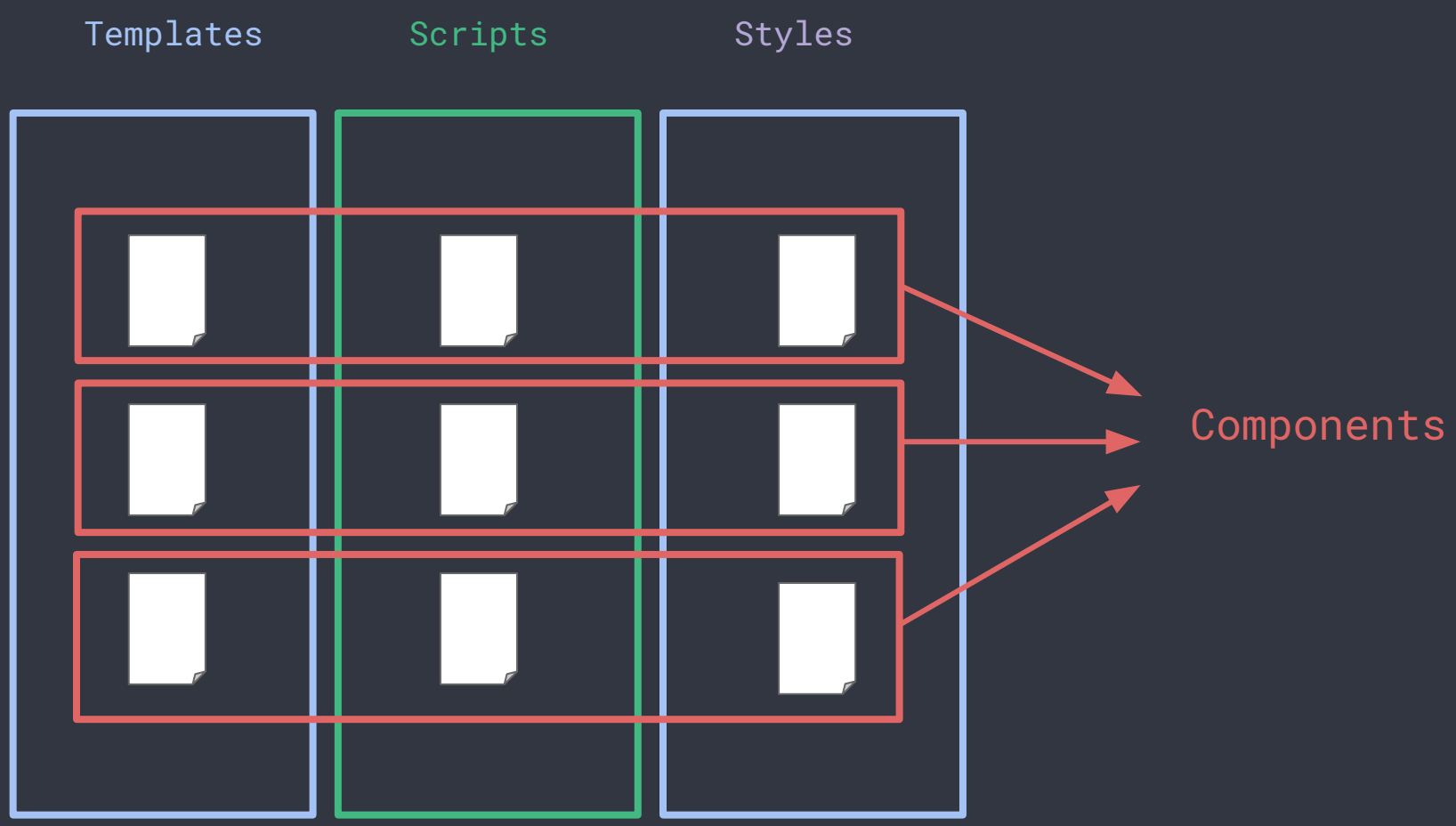
Single File Vue Components

example.vue

```
<template lang="jade">
div.my-component
  h1 {{ msg }}
  other-component
</template>

<script>
import OtherComponent from './other-component.vue'

export default {
  components: { OtherComponent },
  data() {
    return {
      msg: 'Hello Vue.js!'
    }
  }
}
</script>

<style lang="sass" scoped>
$font-stack: Helvetica, sans-serif;
$primary-color: #333;

.my-component {
  font: 100% $font-stack;
  color: $primary-color;
}
</style>
```

Line 1, Column 23 Spaces: 2 Vue Component


Templates
Scripts
Styles


Templates
Scripts
Styles
Components

example.vue

```vue
<template lang="jade">
div.my-component
  h1 {{ msg }}
  other-component
</template>

<script>
import OtherComponent from './other-component.vue'

export default {
  components: { OtherComponent },
  data() {
    return {
      msg: 'Hello Vue.js!'
    }
  }
}
</script>

<style lang="sass" scoped>
$font-stack: Helvetica, sans-serif;
$primary-color: #333;

.my-component {
  font: 100% $font-stack;
  color: $primary-color;
}
</style>
```

Line 1, Column 23 Spaces: 2 Vue Component

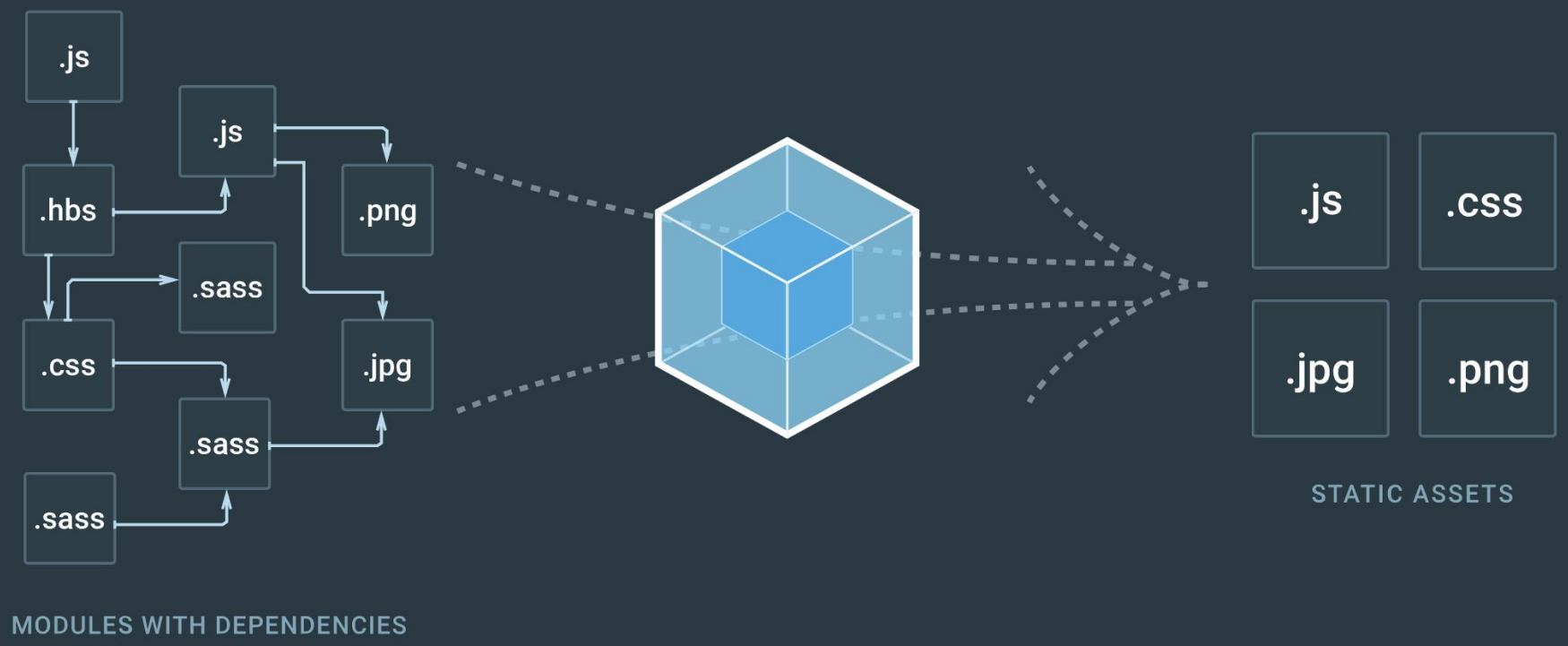
Template

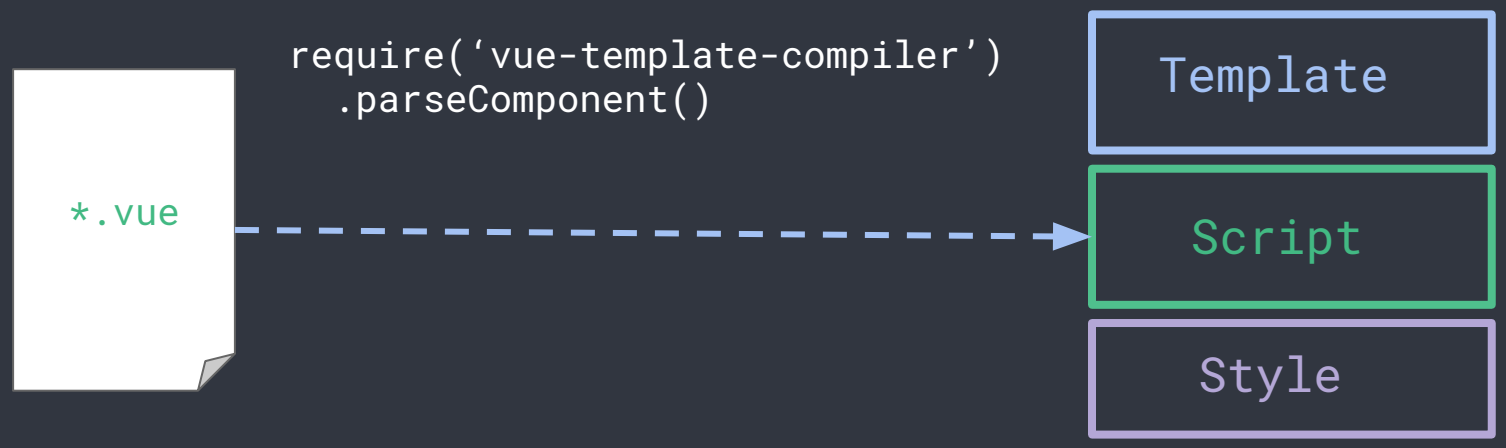
Script

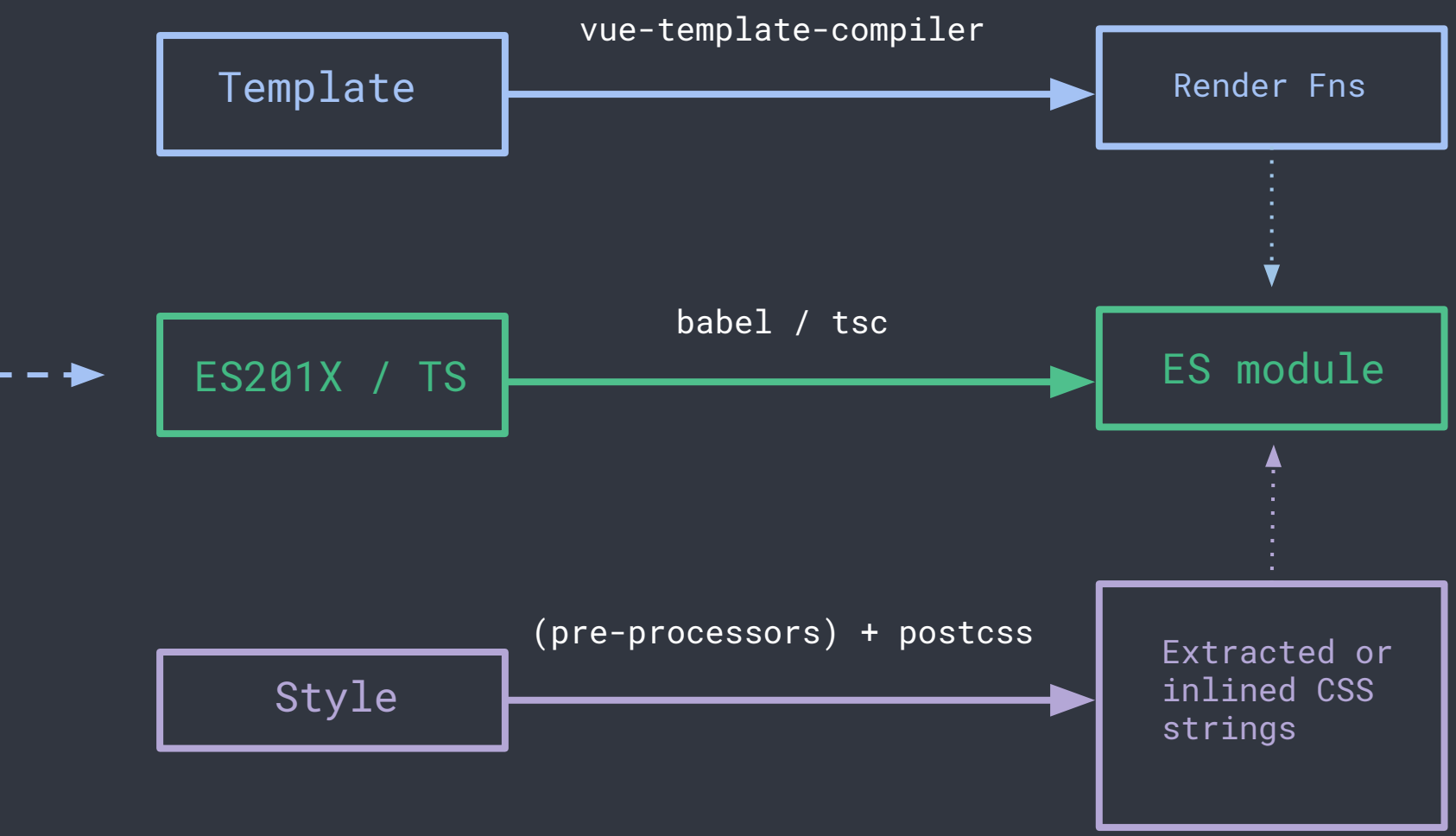
Style

Collocation is awesome!

webpack & vue-loader




require('vue-template-compiler')
.parseComponent()
*.vue
Template
Script
Style

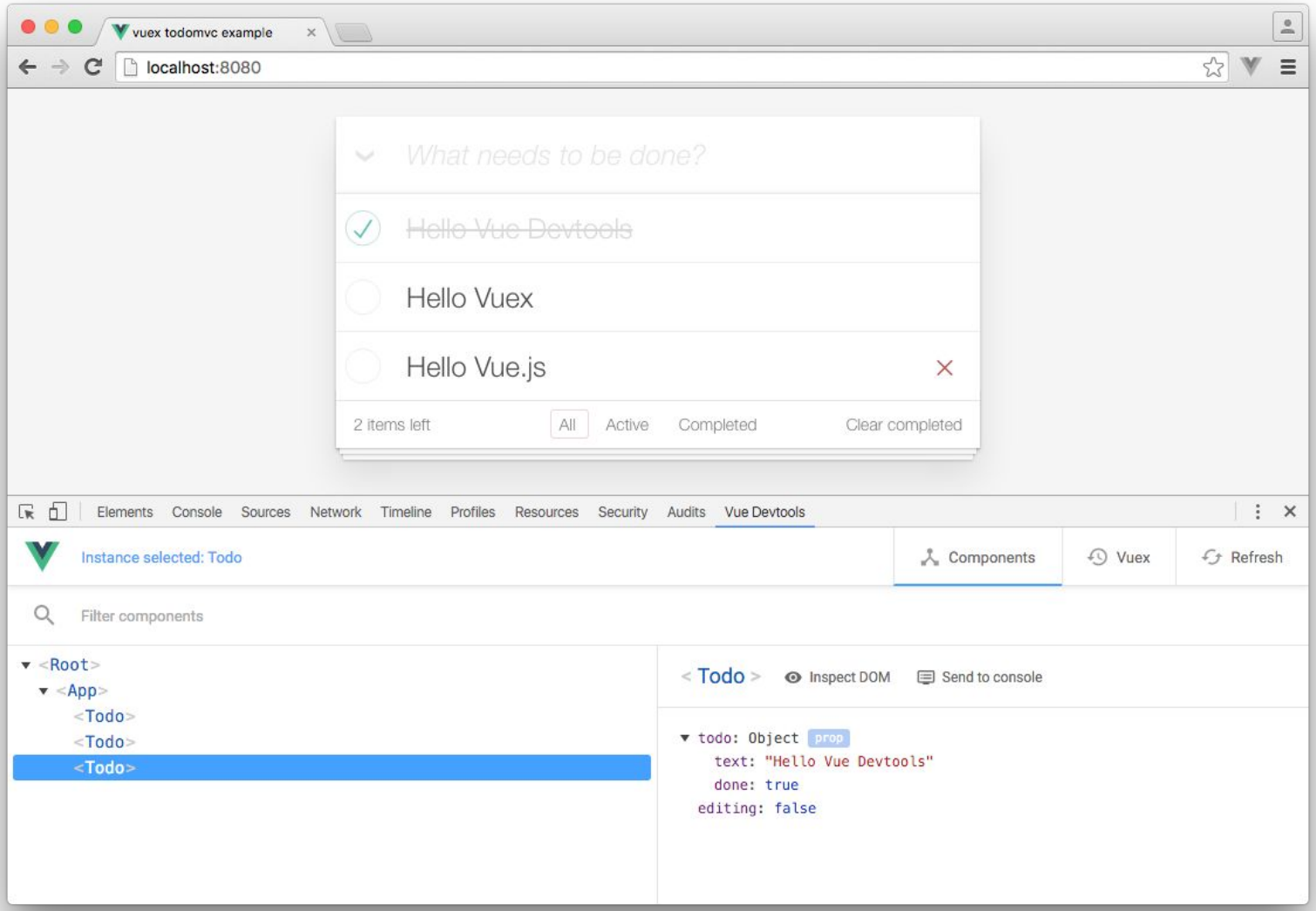

Template
vue-template-compiler
Render Fns
ES201X / TS
babel / tsc
ES module
Style
(pre-processors) + postcss
Extracted or inlined CSS strings

Single File Vue Components

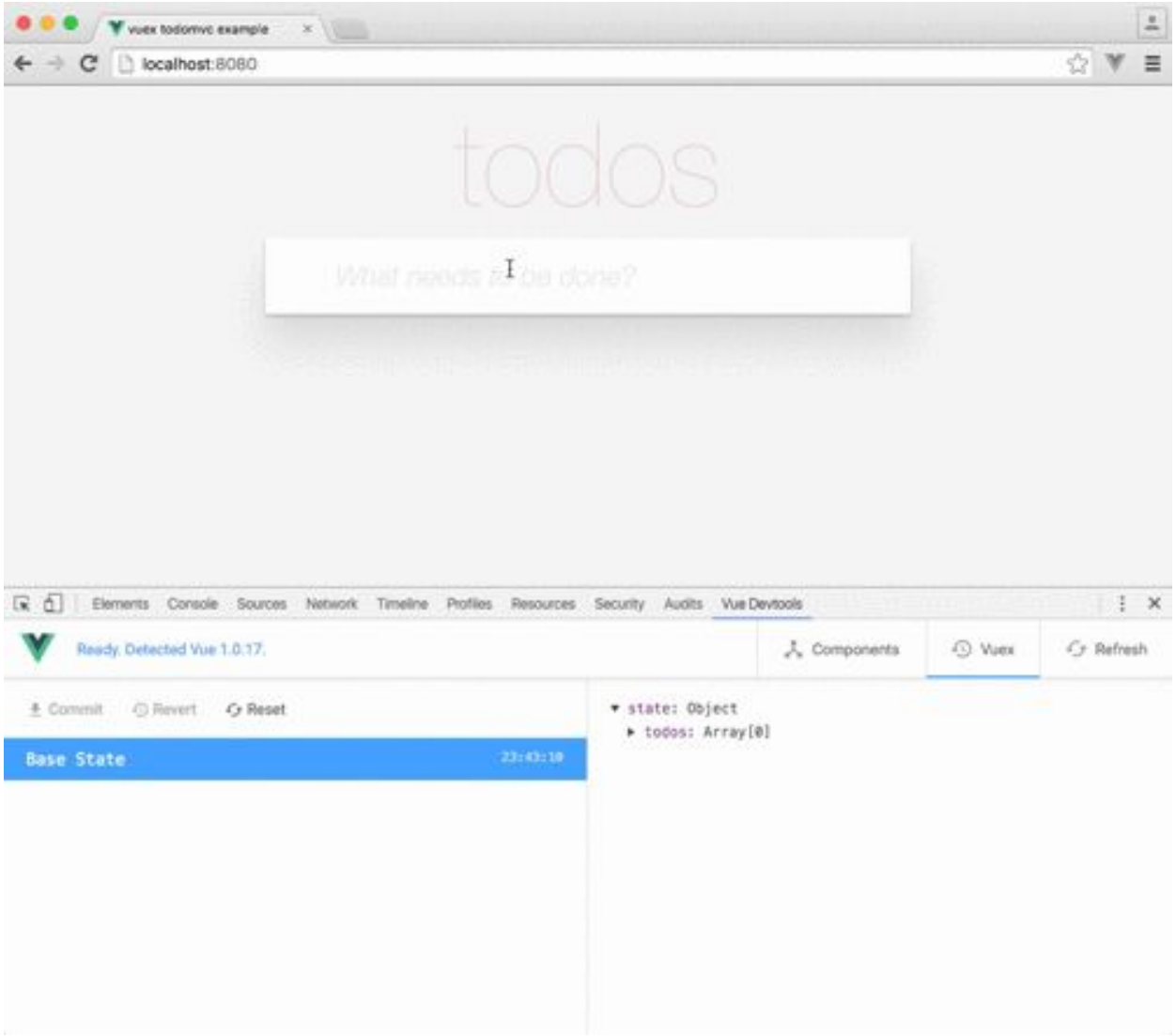
```
<template lang="jade">
div.my-component
  h1 {{ msg }}
  other-component
</template>

<script>
import OtherComponent from './other-component.vue'

export default {
  components: { OtherComponent },
  data() {
    return {
      msg: 'Hello Vue.js!'
    }
  }
}
</script>

<style lang="sass" scoped>
$font-stack: Helvetica, sans-serif;
$primary-color: #333;

.my-component {
  font: 100% $font-stack;
  color: $primary-color;
}
</style>
```

- Collocation of Template, Logic & Style
- Just use what you already know: HTML, CSS & JavaScript
- Imported as a ES2015 module (thus easily testable)
- Embedded pre-processor support: seamlessly use Babel, SASS or even Pug in the same file
- Hot-reload out of the box
- Component-scoped CSS with a single attribute

Official Chrome DevTools Extension



Out of the box time-travel debugging with Vuex



Other Concerns

Is it fast?
Yep.

http://www.stefankrause.net/js-frameworks-benchmark5/webdriver-ts/table.html

Do people actually use it? Yep.

https://www.quora.com/How-popular-is-VueJS-in-the-industry

Server-side Rendering? Yep.

(with streaming, caching & code-split support!)

vue-server-renderer

https://github.com/vuejs/vue-hackernews-2.0/

Native iOS/Android Rendering? Yep.

Weex Project

Why should I use it instead of React, anyway?

You may or may not, but read this and decide for yourself!

But it’s not backed by Google or Facebook!

It’s backed by the Community
And you can be part of it.

Thanks!


vuejs.org


@vuejs


vuejs/vue