BADR

Knowledge Academy

A Point Of Vue

Monday, Sept 11

Who Am I?

Frontend Developer @ BADR
AASTMT Grad - CS Major

Sessions Roadmap

1- Vue.js in Depth

Vue.js core concepts, Library Internals,
Vue Instance Components.

2- REST APIs & Routing

Consuming REST APIs, Services,
What's Routing, Nested Routing, More

3- State Management & Wrap up

Manage components state, Interaction
between different components

Capstone
Project

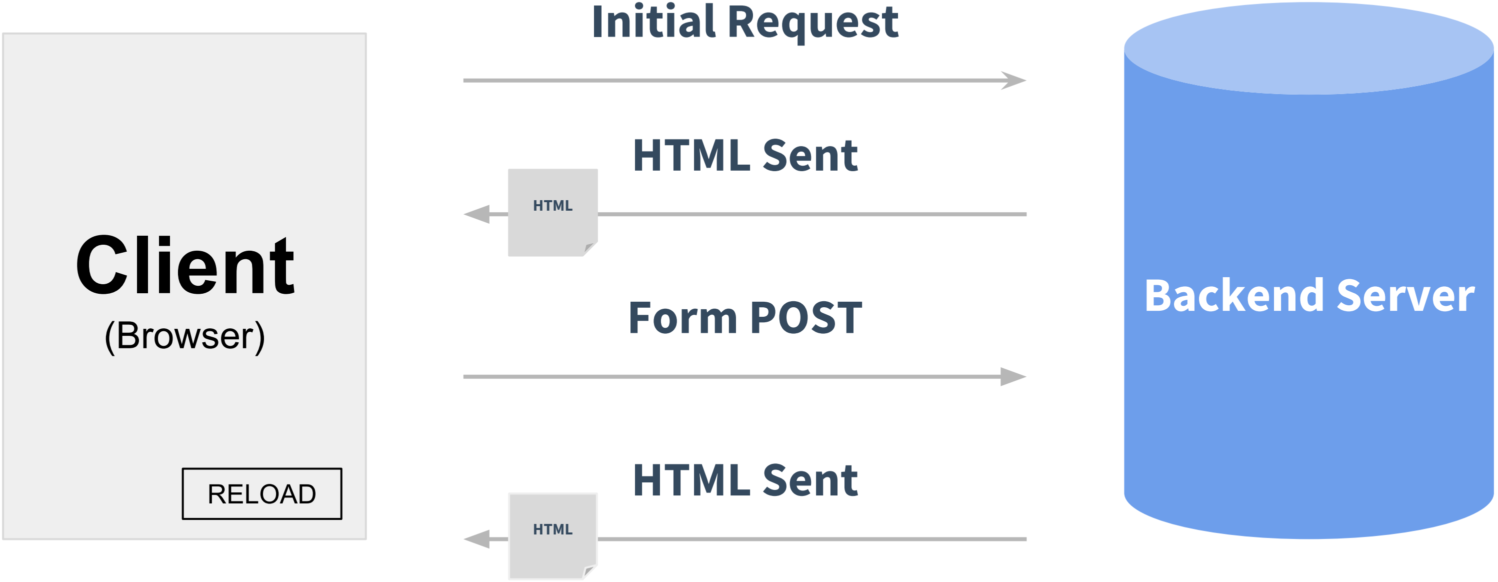
New Tab Dashboard

A New Tab Dashboard with a set of widgets:
Weather, Todo List and New Feed Reader.

Session Agenda

- Traditional Website
- Evolution
- What’s a single page app?
- SPA Challenges
- FED Frameworks Status Quo
- What’s Vue.js?
- Core Concepts (TLDR)
- Why Vue.js?
- Sneak peak & Environment Setup

Traditional Website



Client

(Browser)

RELOAD

Initial Request

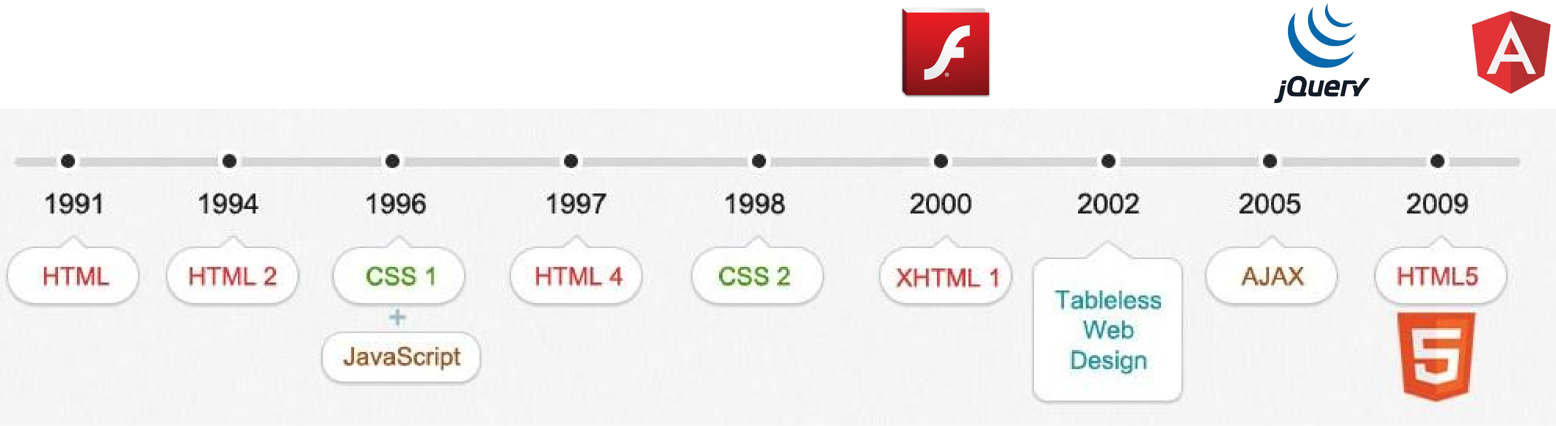
HTML Sent

Form POST

HTML Sent

Backend Server

Evolution | Timeline of Web Technologies



1991
HTML

1994
HTML 2

1996
CSS 1
+
JavaScript

1997
HTML 4

1998
CSS 2

2000
XHTML 1

2002
Tableless Web Design

2005
AJAX

2009
HTML5

Evolution

- **Before 1999**

  Simple, Backend does 99% of the work.

- **1999 & 2000**

  Microsoft creates **XMLHTTP**, Mozilla creates **XMLHTTPRequest**, The beginning of **AJAX**

- **2000 - 2004**

  The Dark Ages

- **2005 & 2006**

  JavaScript Renaissance, **jQuery** sees the light

- **2007 - 2009**

  We build complex web applications using our great new frameworks (**Prototype.js, YUI.js**)

- **2010**

  Birth of real **Single Page Applications** - Angularjs

What’s a single page application?

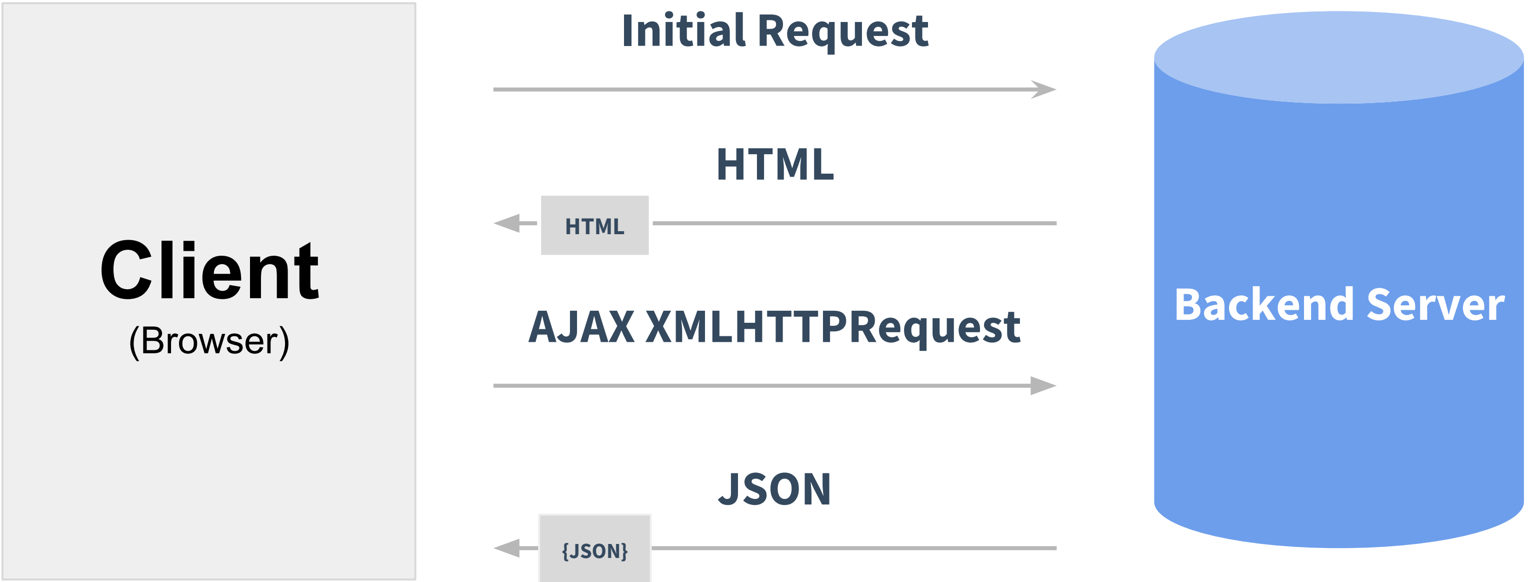
> “
>
> *A single-page application (SPA) is a web application or website that fits on a single web page with the goal of providing a more fluid user experience similar to a desktop application.*
>
> ”

What’s a single page application?

- More Fluid, Snappy UX
- More decoupled Architecture
    - Frontend Dedicated to handling UI Complexity
    - Backend Focused on data logic and exposing API endpoints
- Offline state in mind

Not every app should be an SPA.

What’s a single page application?



Client
(Browser)

Initial Request

HTML

HTML

AJAX XMLHTTPRequest

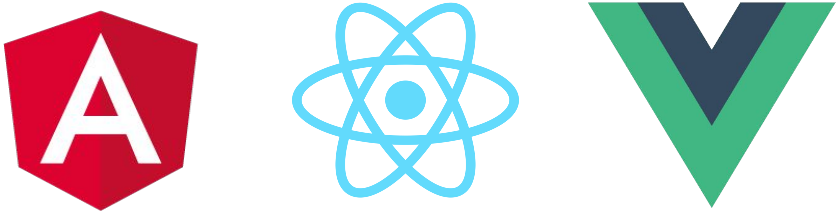
JSON

{JSON}

Backend Server

SPA Challenges

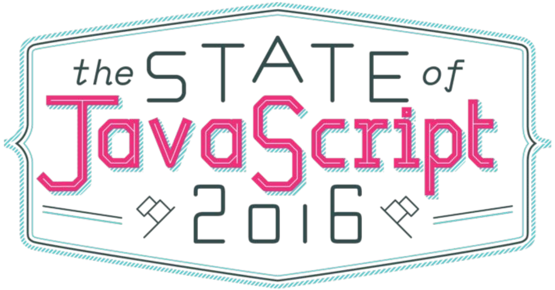
- Displaying semantic URLs without reloading the page
- Dynamically render correct UI based on URL navigations
- Speed of Initial Load
- Browser History
- Search Engine Optimization



JavaScript FED Frameworks

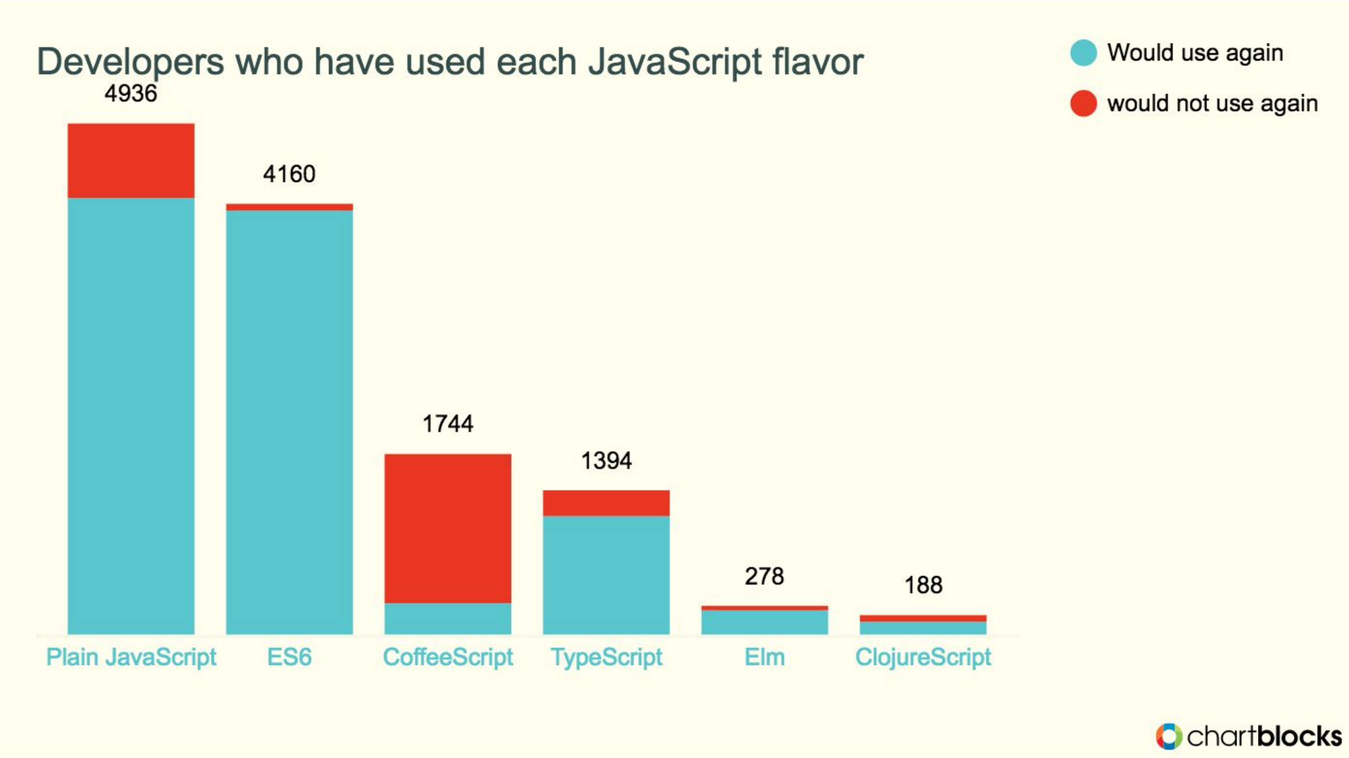
FED Frameworks Status Quo

The State Of JavaScript:
Survey about Front-End Frameworks

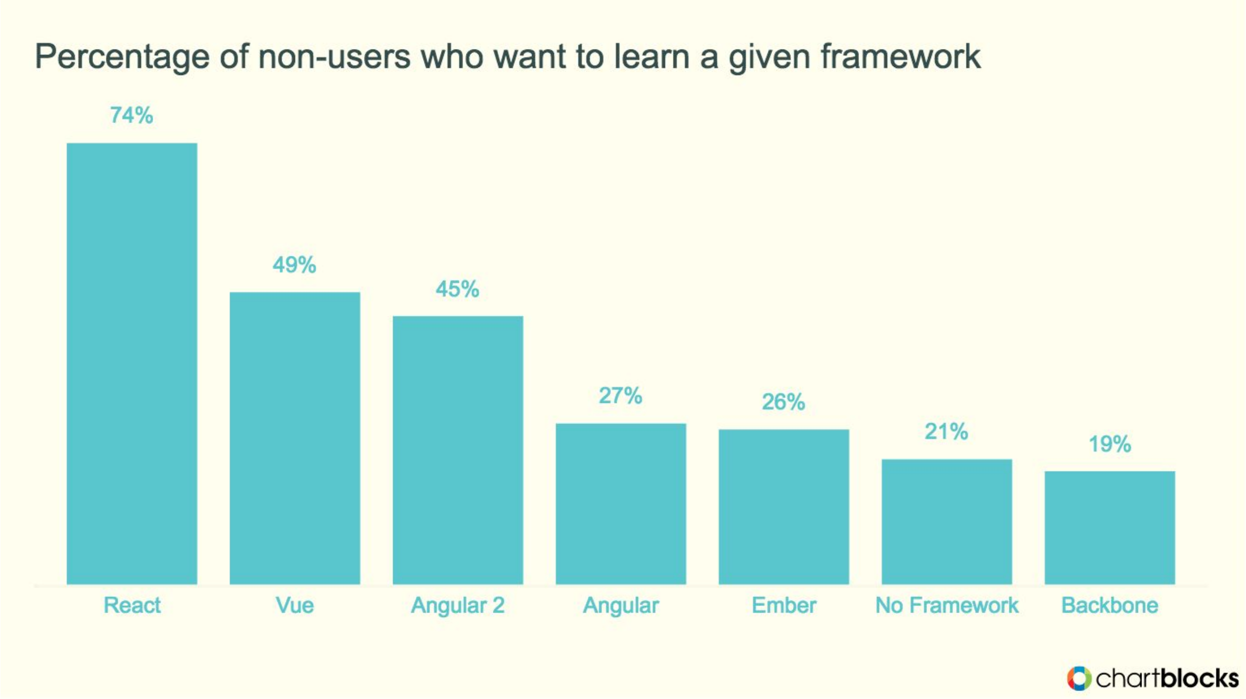


Over 3000 responses from worldwide Front-End developers.

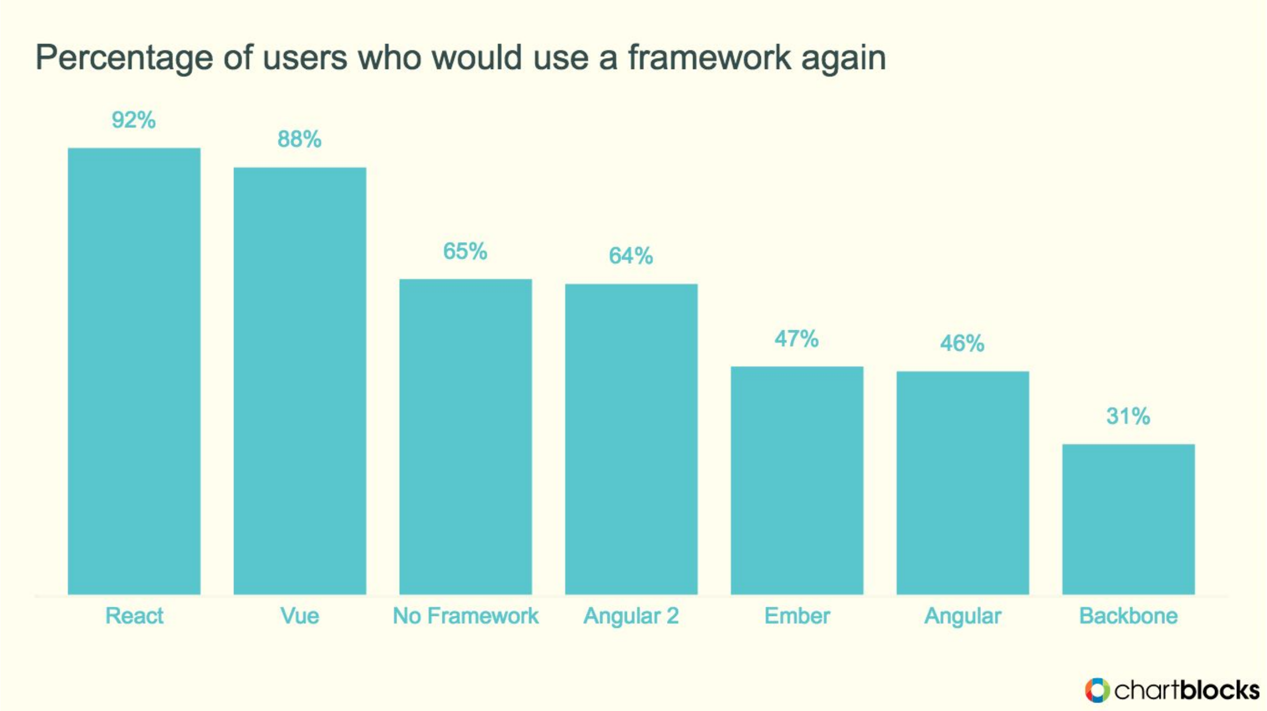
FED Frameworks Status Quo



FED Frameworks Status Quo



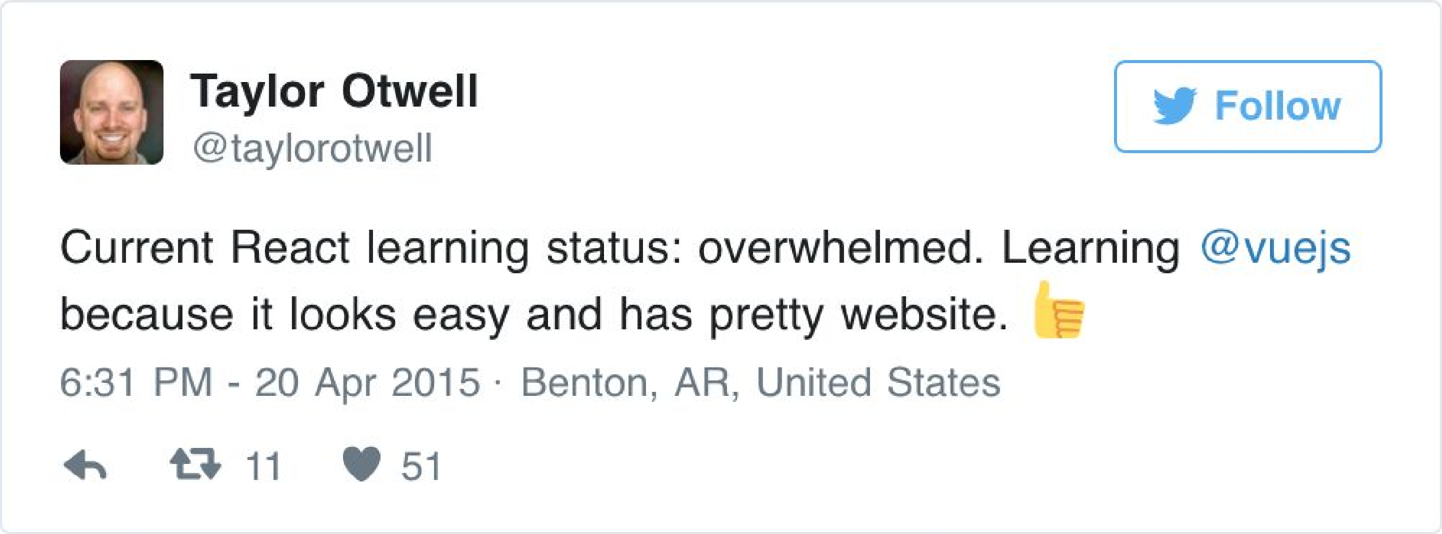
FED Frameworks Status Quo



What’s Vue.js

- Started as **View-only** library, Now considered as **Progressive** Framework
- Started in Late 2013 by **Evan You**, Latest release in **July 21 2017**
- **+66K** Stars on Github, **657K Download/Week** on **NPM**

What’s Vue.js

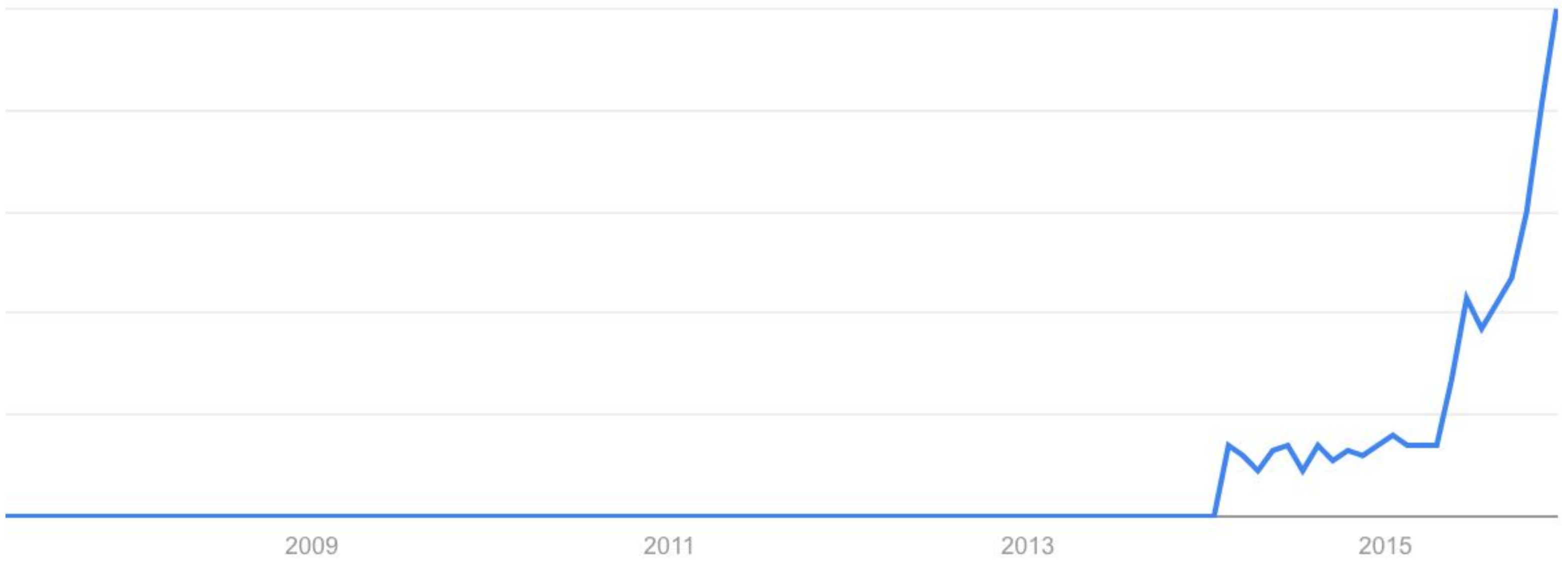


Taylor Otwell
@taylorotwell

Current React learning status: overwhelmed. Learning @vuejs because it looks easy and has pretty website. 👍

6:31 PM - 20 Apr 2015 · Benton, AR, United States

What’s Vue.js



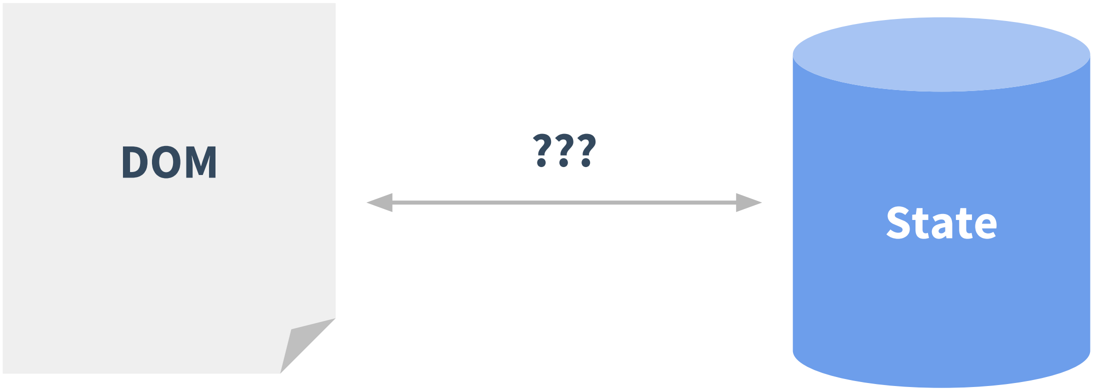
What’s Vue.js

CODESHIP
CLEMENGER BBDO
Optimizely
Baidu 百度
Alibaba Group
BROSA
GitLab
Expedia
Nintendo

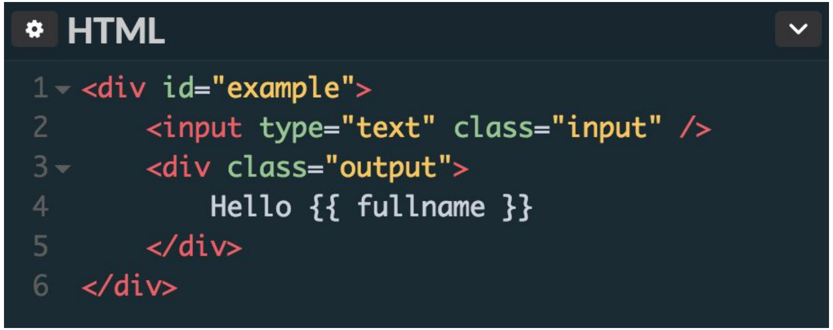
Core Concepts

1. **Declarative & Reactive Rendering**
2. **Component Oriented Architecture**

Core Concepts | Declarative & Reactive Rendering

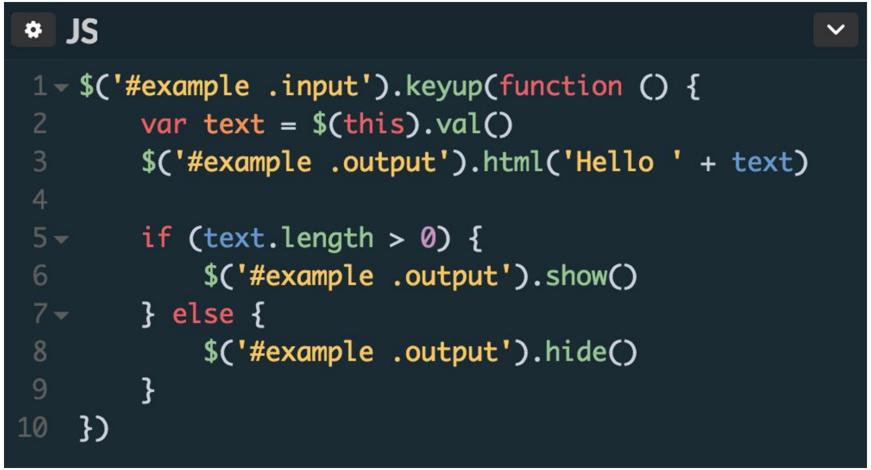


Problems with DOM-Driven approach is problematic



```html
<div id="example">
    <input type="text" class="input" />
    <div class="output">
        Hello {{ fullname }}
    </div>
</div>
```

If we rename an element, or if we change the html structure, the code will be broken.



```javascript
$('#example .input').keyup(function () {
    var text = $(this).val()
    $('#example .output').html('Hello ' + text)

    if (text.length > 0) {
        $('#example .output').show()
    } else {
        $('#example .output').hide()
    }
})
```

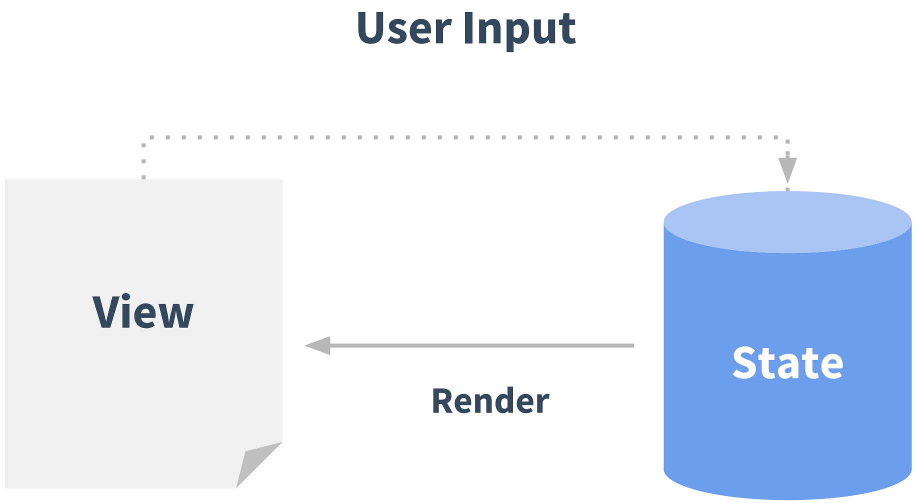
Code complexity.

Core Concepts | Declarative & Reactive Rendering

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

Core Concepts | Declarative & Reactive Rendering



View is just a declarative mapping from the state

State should be the single source of truth

Core Concepts | Declarative & Reactive Rendering

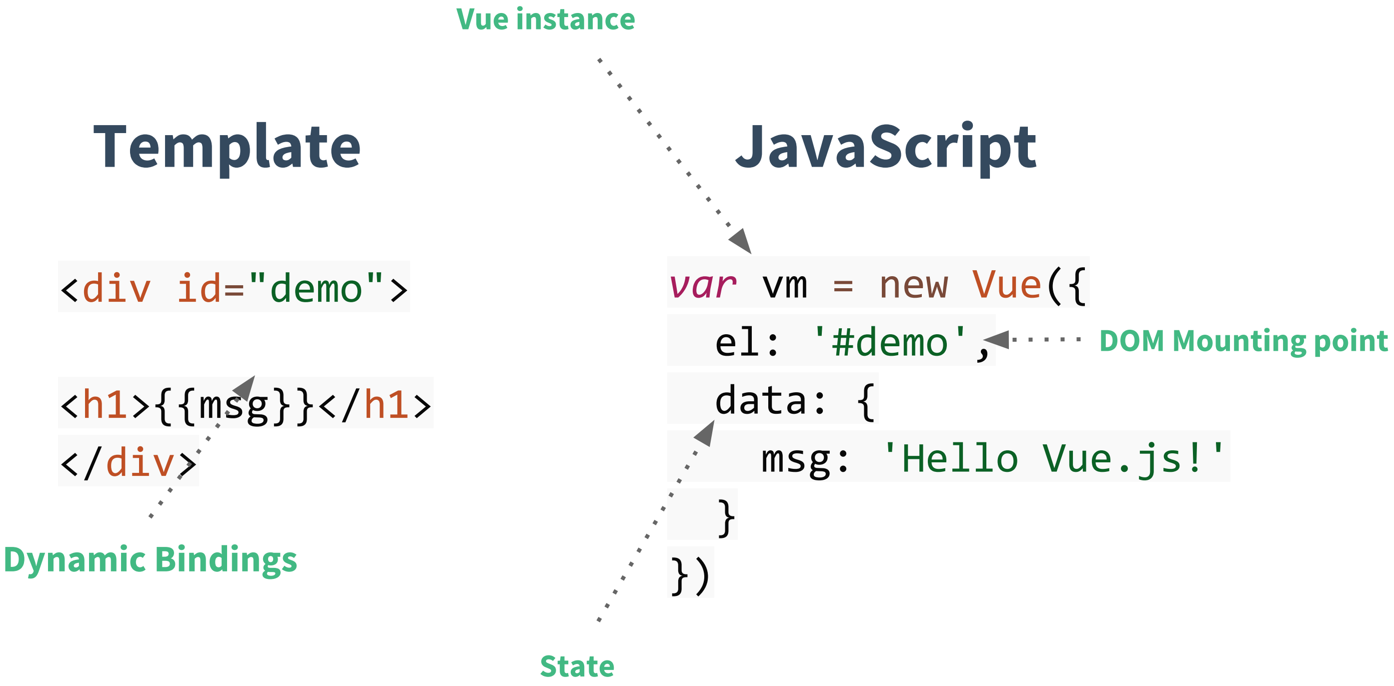
HTML

```html
<div id="demo">
  <h1>{{msg}}</h1>
</div>
```

JavaScript

```javascript
var vm = new Vue({
  el: '#demo',
  data: {
    msg: 'Hello Vue.js!'
  }
})
```

Core Concepts | Declarative & Reactive Rendering



Vue instance

Template

```html
<div id="demo">

<h1>{{msg}}</h1>
</div>
```

Dynamic Bindings

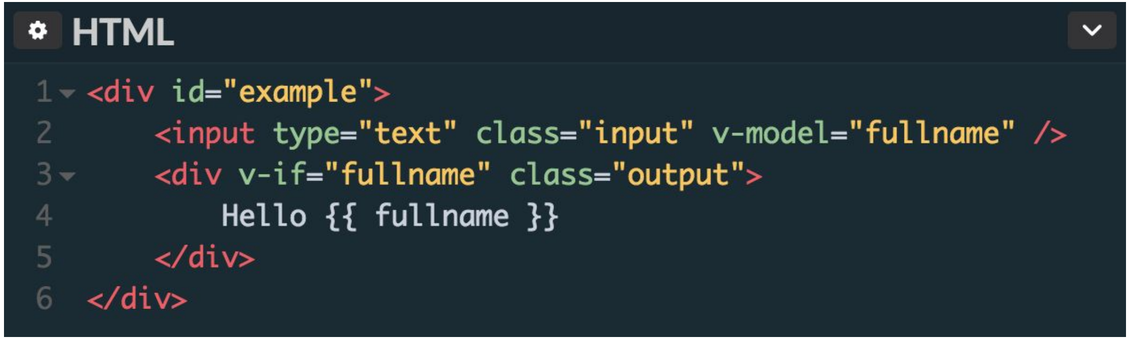
JavaScript

```javascript
var vm = new Vue({
  el: '#demo',
  data: {
    msg: 'Hello Vue.js!'
  }
})
```

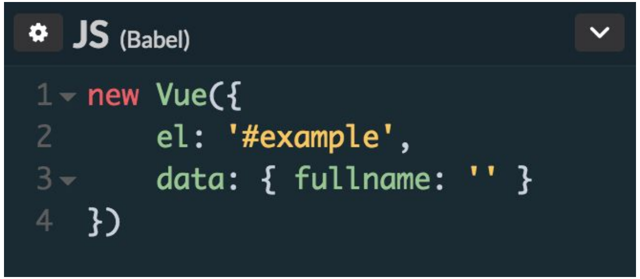
DOM Mounting point

State

Core Concepts | Declarative & Reactive Rendering



```html
<div id="example">
    <input type="text" class="input" v-model="fullname" />
    <div v-if="fullname" class="output">
        Hello {{ fullname }}
    </div>
</div>
```

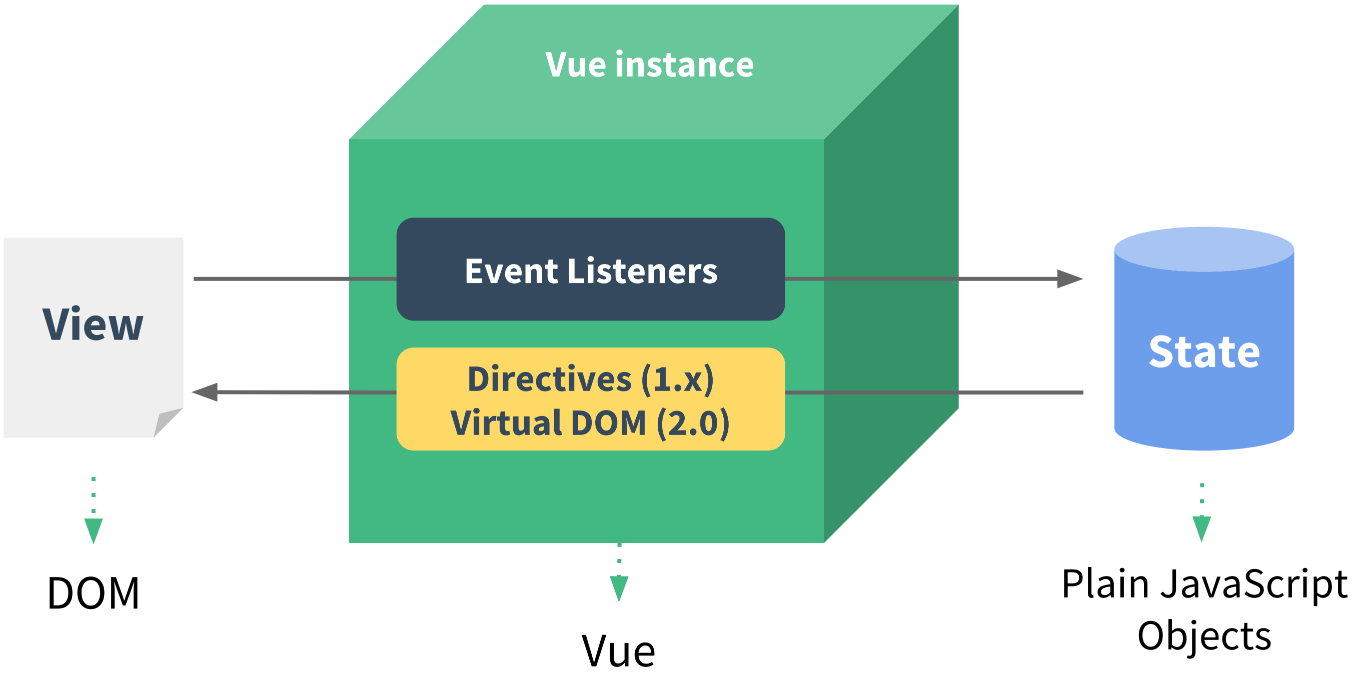


```js
new Vue({
    el: '#example',
    data: { fullname: '' }
})
```

We can change the html structure and rename elements without breaking the code.

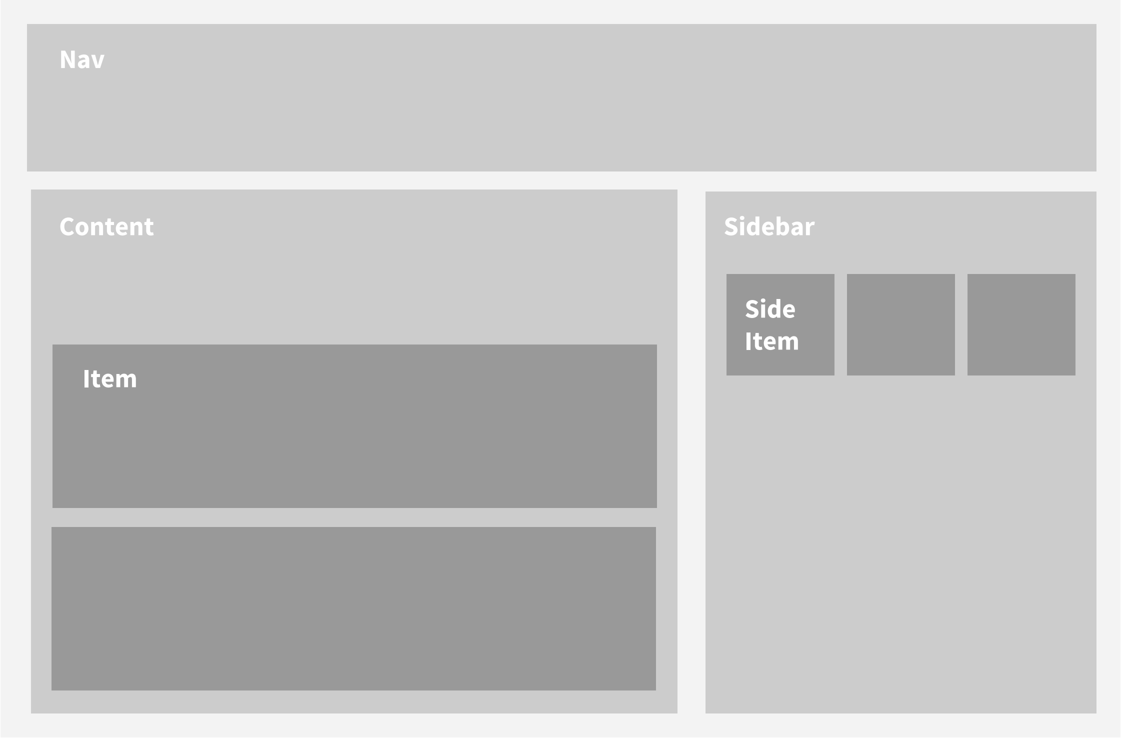
Minimalist code.

Core Concepts | Declarative & Reactive Rendering



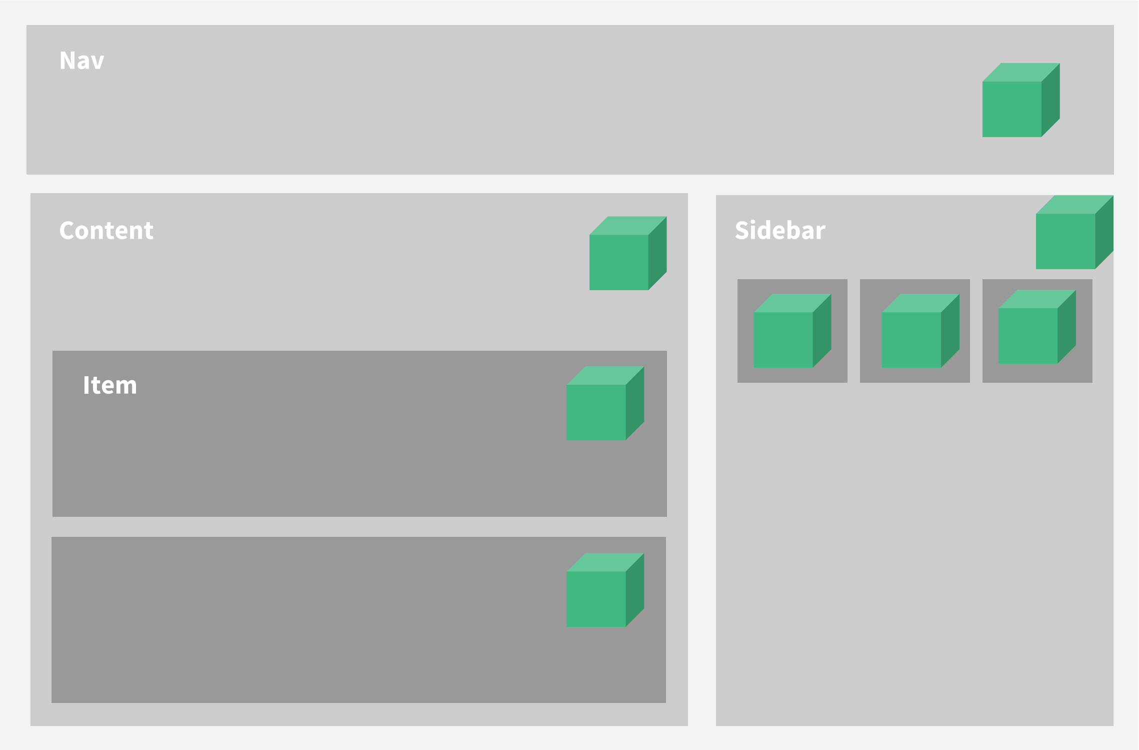
Core Concepts | Component Oriented

Most App UIs can be broken down into components

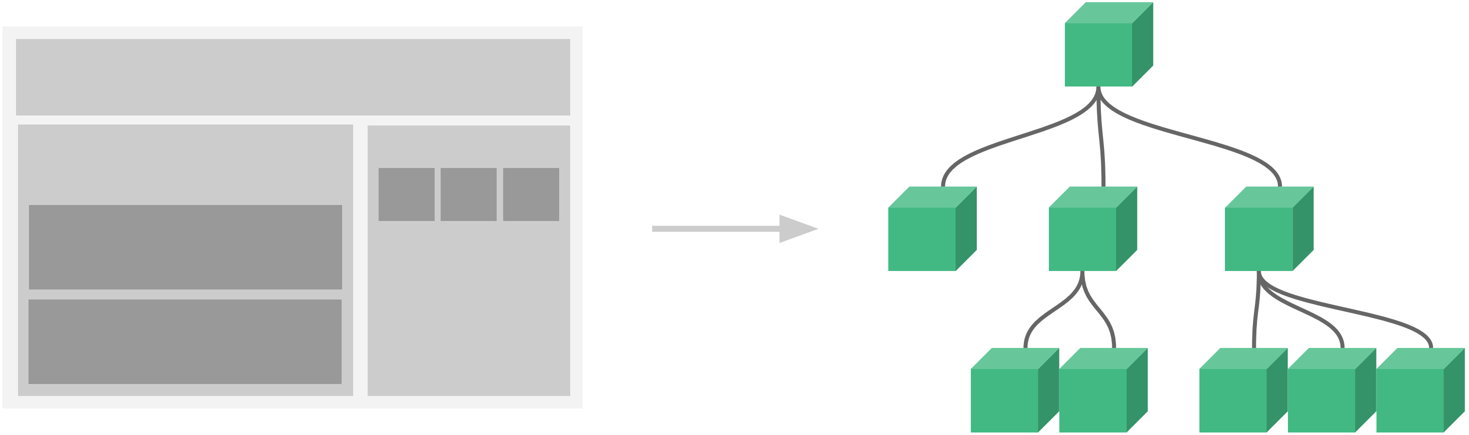


Core Concepts | Component Oriented

Every component is responsible for managing a piece of DOM



Core Concepts | Component Oriented



The entire UI can be abstracted into a tree of components

Core Concepts | Component Oriented

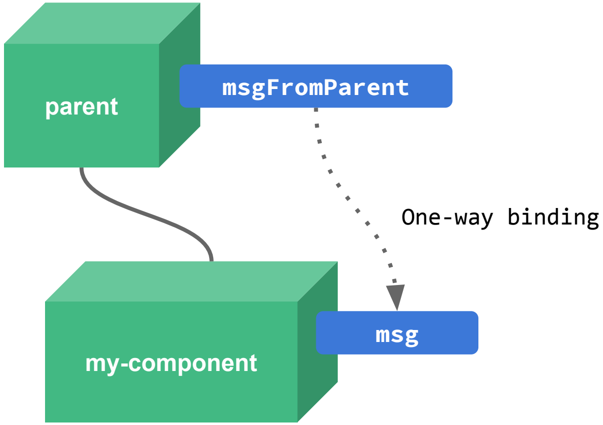
```
<my-component msg="Hello!">
</my-component>
```

my-component's template will be inserted into the container element

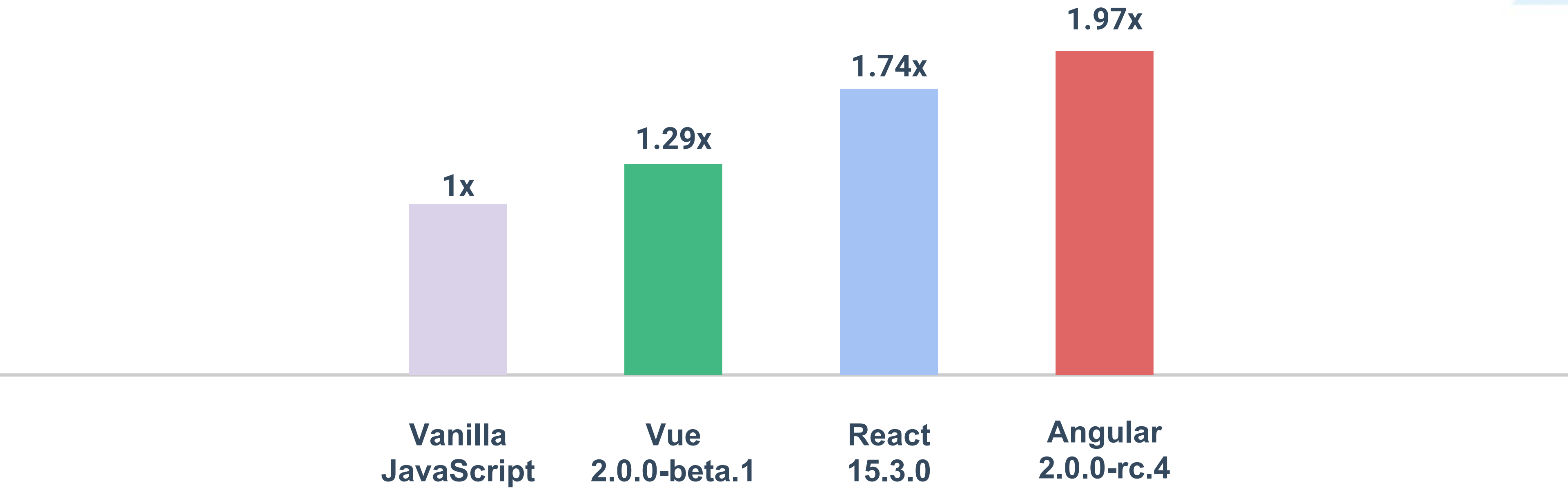
```
<p>{{msg}}</p>
```

`msg` will be passed in as data state.

```
<p>Hello!</p>
```



Why Vue.js? - Fast

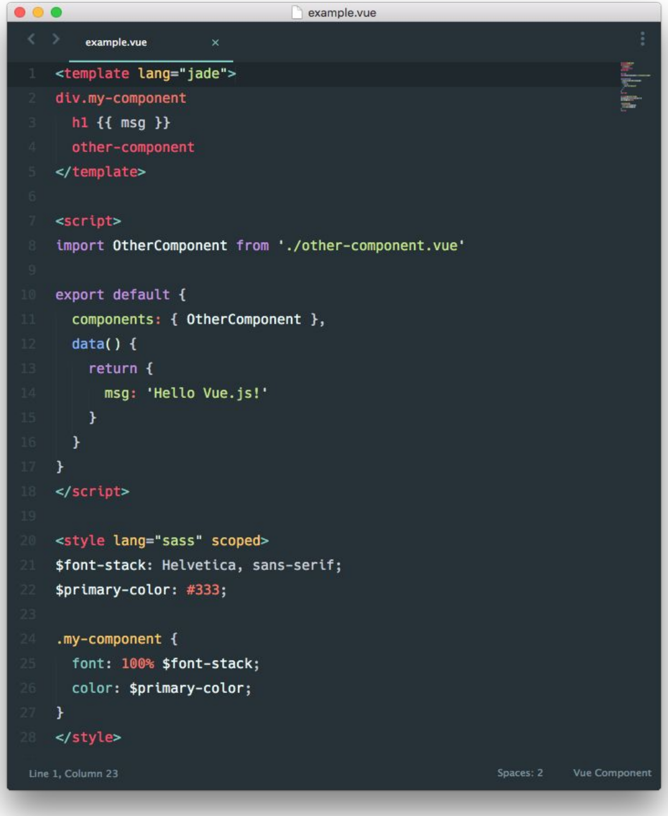


Based on independent 3rd party comprehensive rendering benchmark

Why Vue.js? - Approachable

```
<script src="vue.js"></script>
<script>
  new Vue({
    // ...
  })
</script>
```

Why Vue.js? - Approachable



```
<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>
```

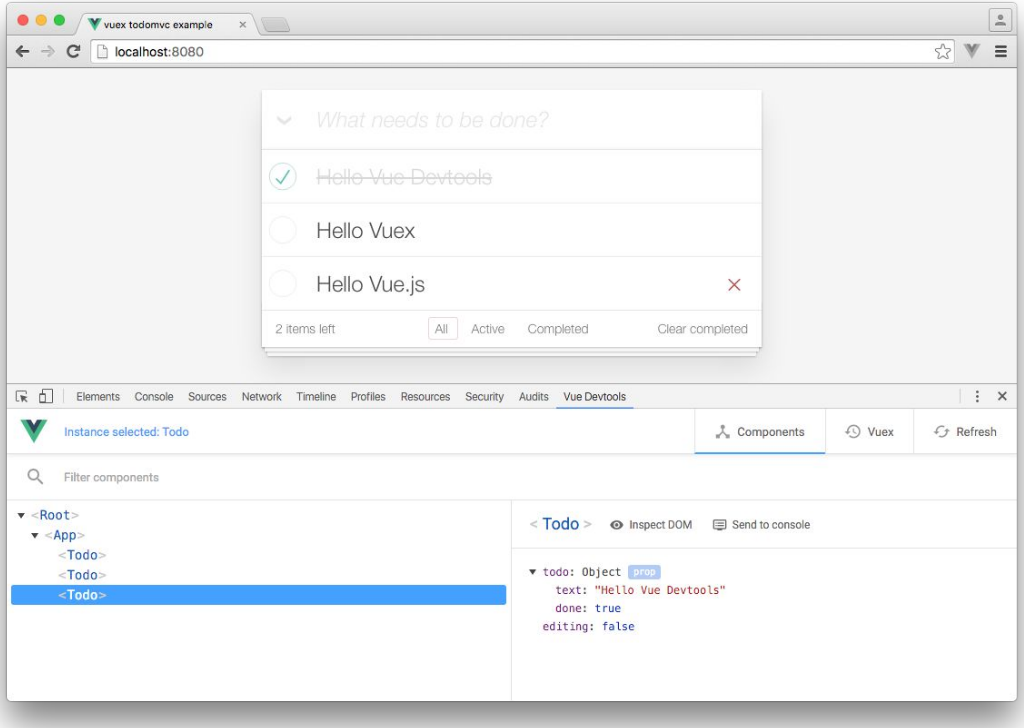
Single File Vue Components

- Imported as a ES6 module (easily testable)
- Collocation of Template, Logic & Style
- Just use what you already know:

  HTML, CSS & JavaScript
- Embedded preprocessor support:

  Sass, Less, Pug, Stylus
- Component-scoped CSS with a single attribute (no conflicts)

Why Vue.js? - Good DEV-XP



~~Hello Vue Devtools~~

Hello Vuex

Hello Vue.js

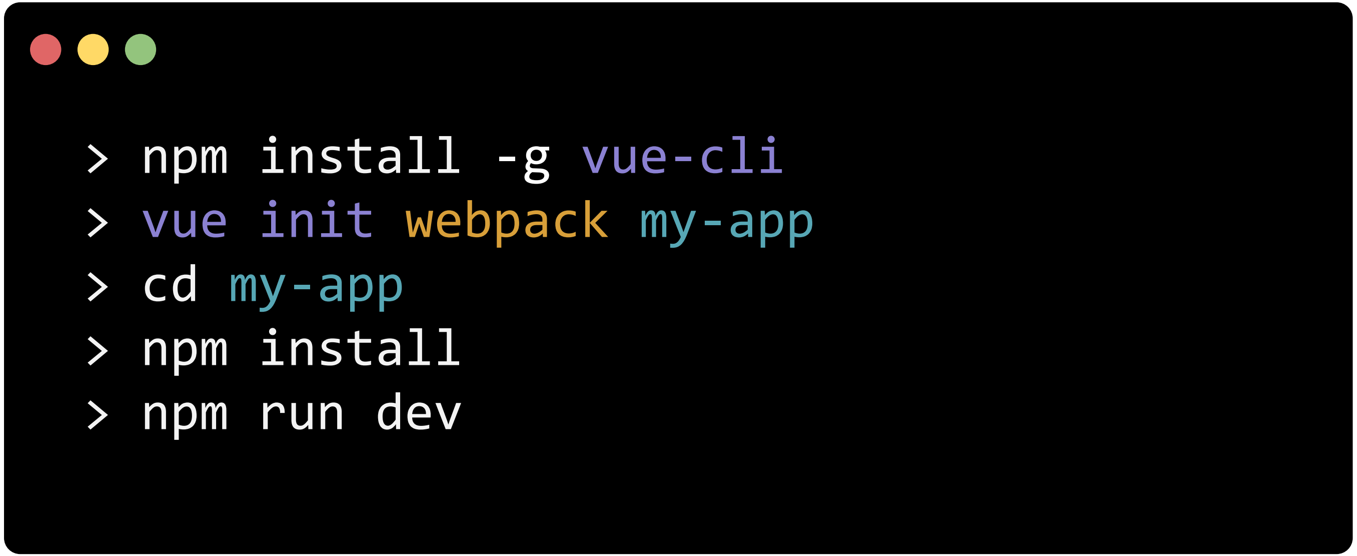
Instance selected: Todo

```
▼ <Root>
  ▼ <App>
      <Todo>
      <Todo>
      <Todo>
```

< Todo >

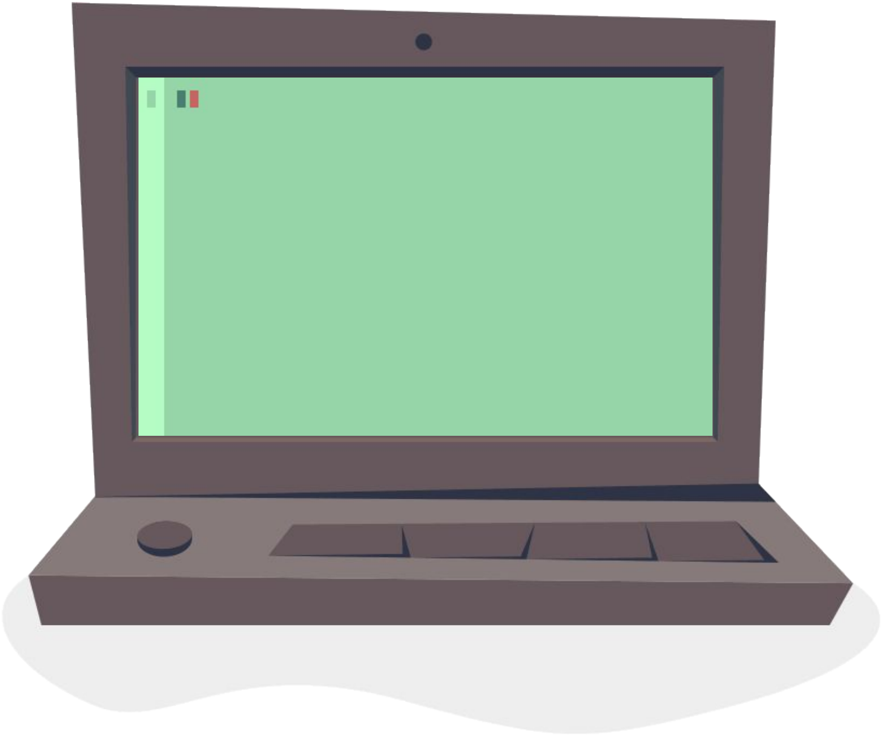
```
▼ todo: Object
    text: "Hello Vue Devtools"
    done: true
  editing: false
```

Sneak peak & Environment Setup



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

goo.gl/3QJEcq



BADR
Knowledge Academy

That’s All for now

See you next Wednesday (13 Sept.)