

Why Vue.js?

Finding a javascript framework that just worked



Brian Wells
[@brianjwells](#)

Who am I?

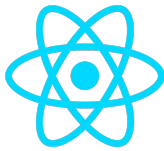
- Senior Lab Software Developer at Riverbed
- Recently moved to Nashville area in May
- Currently use Vue.js in production
- Working remotely



Angular, Ember, React, etc



ember



METE  R



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."

<https://medium.com/@ericclemmons/javascript-fatigue-48d4011b6fc4>



Lessons I've Learned

Lessons I've Learned

- You'll embrace a framework once it solves a problem you have
- There will always be a popular new framework
- Choose one that is stable and master it
- Don't be afraid to tinker first
- Never stop learning



The progressive framework

The progressive framework

“

Use a progressive framework composed of incrementally adoptable libraries

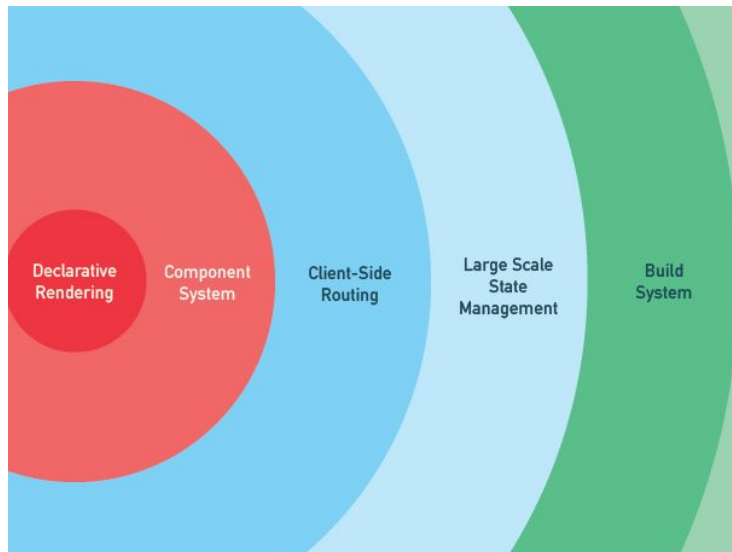
@vuejs

<https://twitter.com/vuejs/status/753636851504799744>



The progressive framework

1. The Core Vue.js Framework
2. Components
3. HTTP Resources
4. Client-Side Routing
5. State management



The progressive framework

“

We can scale up the framework's complexity incrementally, only when the project's inherent complexity demands it.

Evan You, Creator of Vue.js

<http://blog.evanyou.me/2015/12/20/vuejs-2015-in-review/>

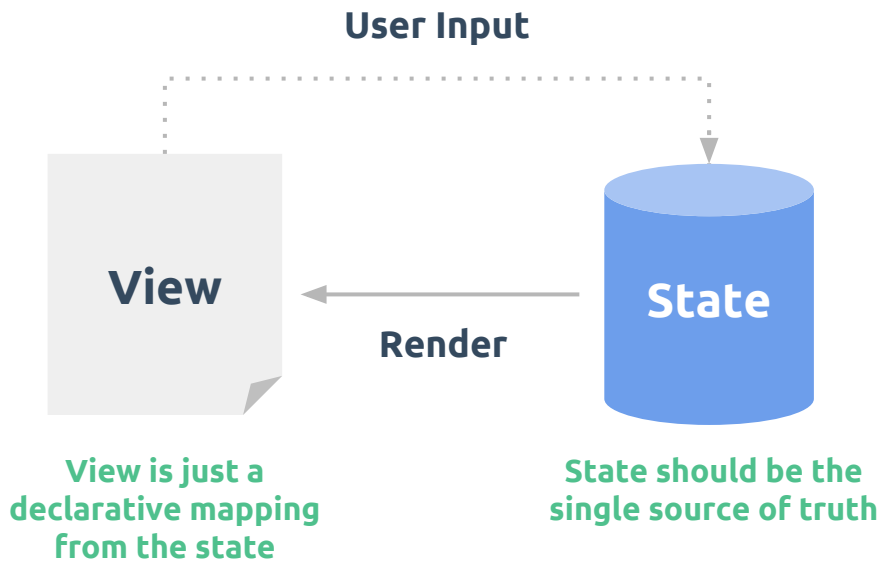


Let's Get Started

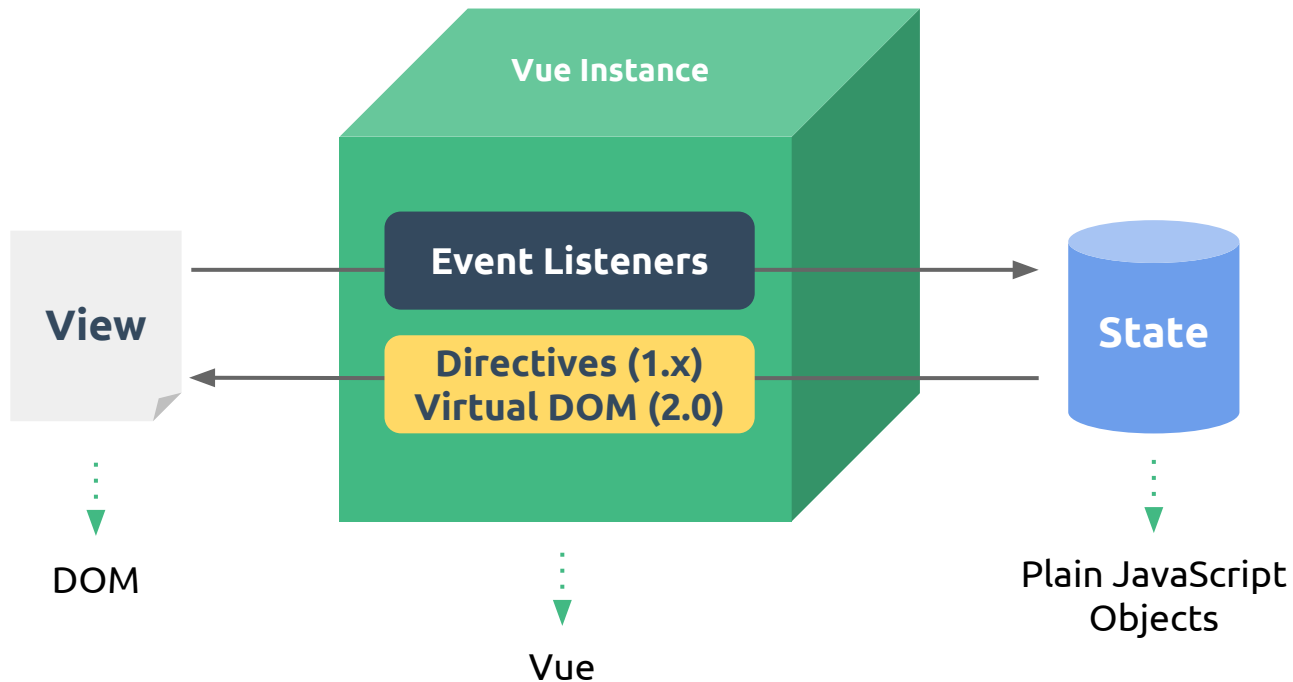
The Core Vue.js Framework

- Reactive data binding
- View layer only
- Automatic

- `// AngularJS`
`$scope.$apply()`
`// Ember.js`
`Ember.Object.create()`
`// React`
`this.shouldComponentUpdate()`
`this.setState()`
`// Knockout`
`ko.observable()`



The Core Vue.js Framework



The Core Vue.js Framework

- Mustache syntax

- `{{ text }}` `{{* once }}` `{{{ html }}}}`
- `{{ ok ? 'YES' : 'NO' }}`

- Directives

- `v-model="message"`
- `v-bind:href="message.url"`
 - `:href="message.url"`
- `v-on:click="sendMessage"`
 - `@click="sendMessage"`

```
<div id="app">
  <p>{{ message }}</p>
  <input v-model="message">
</div>
```



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



The Core Vue.js Framework

- Methods

- `methods: { ... }`

- Handle native DOM events

- `@click` or `v-on:click`

- Automatic event listeners

- `// jQuery example`
`$('#button').on('click',`
 `function (event) {`
 `...`
 `});`

```
<div id="app">
  <button @click="b">Increment</button>
  a={{ a }}
</div>
```



```
new Vue({
  el: '#app',
  data: {
    a: 1
  },
  methods: {
    b: function (event) {
      // `this` points to the vm instance
      this.a = this.a + 1
    }
  }
})
```



The Core Vue.js Framework

- Computed properties

- `computed: { ... }`

- Replaces binding expressions

- `{{ ok ? 'YES' : 'NO' }}`

- Computed setter

- `computed: {
 b: {
 get: function () { ... },
 set: function (newValue) { ... }
 }
}`

```
<div id="app">  
  a={{ a }} b={{ b }}  
</div>
```



```
new Vue({  
  el: '#app',  
  data: {  
    a: 1  
  },  
  computed: {  
    // a computed getter  
    b: function () {  
      // `this` points to the vm instance  
      return this.a + 1  
    }  
  }  
})
```



The Core Vue.js Framework

- Class and style bindings

- `:class` or `v-bind:class`
- `:style` or `v-bind:style`

- Conditional rendering

- `v-if="hasMessages"`
- `v-show="hasMessages"`
- `v-else`

```
<div class="static" :class="{ 'class-a': isA }">  
  ...  
</div>
```



```
<h1 v-if="hasMessages">Yes</h1>  
<h1 v-else>No</h1>
```

```
<h1 v-show="hasMessages">Yes</h1>  
<h1 v-else>No</h1>
```



The Core Vue.js Framework

- JSON data format

- `var object = { ... }`

- List rendering

- `v-for="todo in todos"`
 - `todos: [{...}, {...}, {...}]`

```
<div id="app">
  <ul>
    <li v-for="todo in todos">
      {{ todo.text }}
    </li>
  </ul>
</div>
```



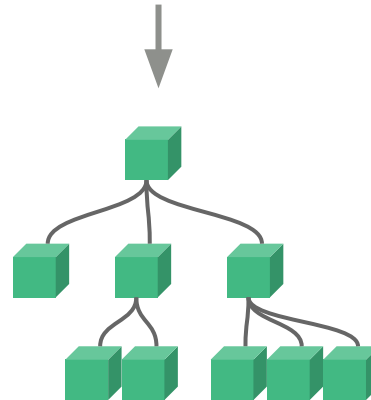
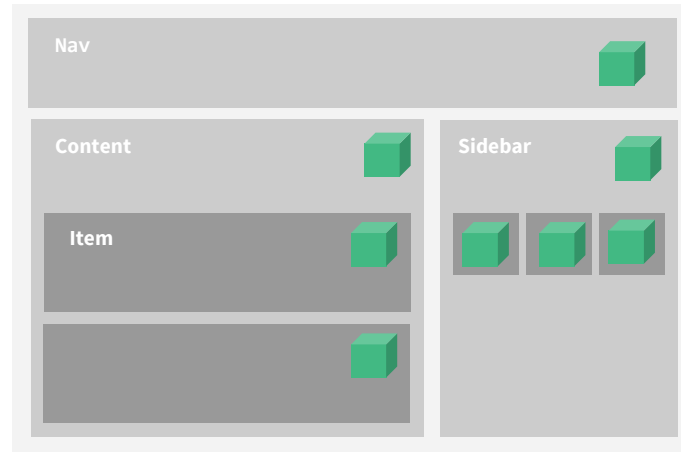
```
new Vue({
  el: '#app',
  data: {
    todos: [
      { text: 'Learn JavaScript' },
      { text: 'Learn Vue.js' },
      { text: 'Build Something Awesome' }
    ]
  }
})
```



Components

Components

- Small
- Self-contained
- Often reusable



Components

- Custom DOM elements
 - `<my-component></my-component>`
- Replaced with template HTML

```
<div id="app">  
  <app-nav></app-nav>  
  <app-content>  
    <example></example>  
  </app-content>  
</div>
```



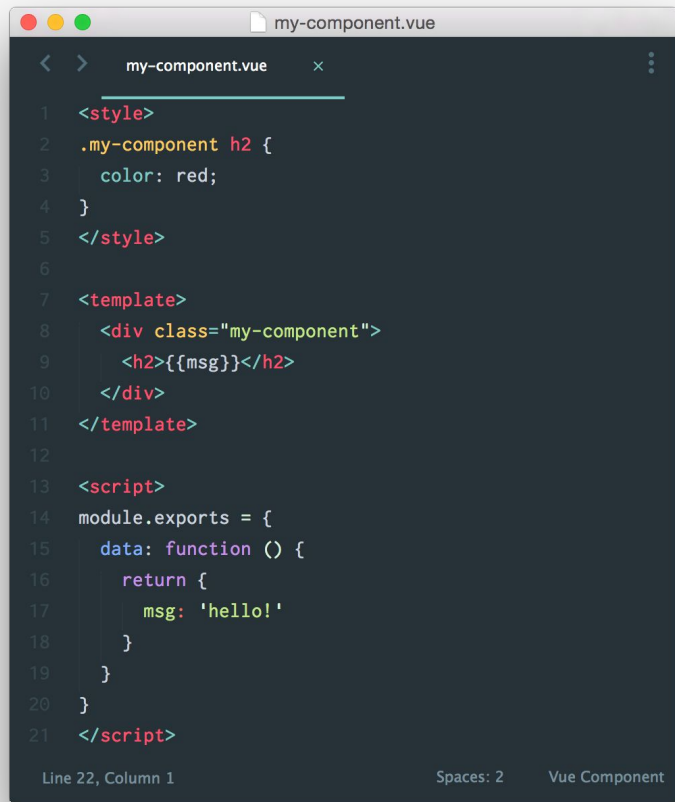
```
var Example = Vue.extend({  
  template: '<div>{{ message }}</div>  
  data: function() {  
    return {  
      message: 'Hello Vue.js!'  
    }  
  }  
})
```

```
// register it with the tag <example>  
Vue.component('example', Example)
```



Components

- Single File Components
- Reusability
 - Props
 - Content Slots
 - Dynamic Components



```
1 <style>
2 .my-component h2 {
3   color: red;
4 }
5 </style>
6
7 <template>
8   <div class="my-component">
9     <h2>{{msg}}</h2>
10  </div>
11 </template>
12
13 <script>
14 module.exports = {
15   data: function () {
16     return {
17       msg: 'hello!'
18     }
19   }
20 }
21 </script>
```

Line 22, Column 1 Spaces: 2 Vue Component



HTTP Resources

HTTP with vue-resource

- HTTP requests/responses
- RESTful API friendly
- Async Promises
- Alternatives:
 - SuperAgent
 - \$.ajax
 - No-backend (i.e. Firebase)

```
// GET /api/movies
this.$http.get('/api/movies')
  .then((response) => {
    // success callback
  }, (response) => {
    // error callback
  });
```



Client-Side Routing

Routing with vue-router

- Single page application (SPA)
- Native application

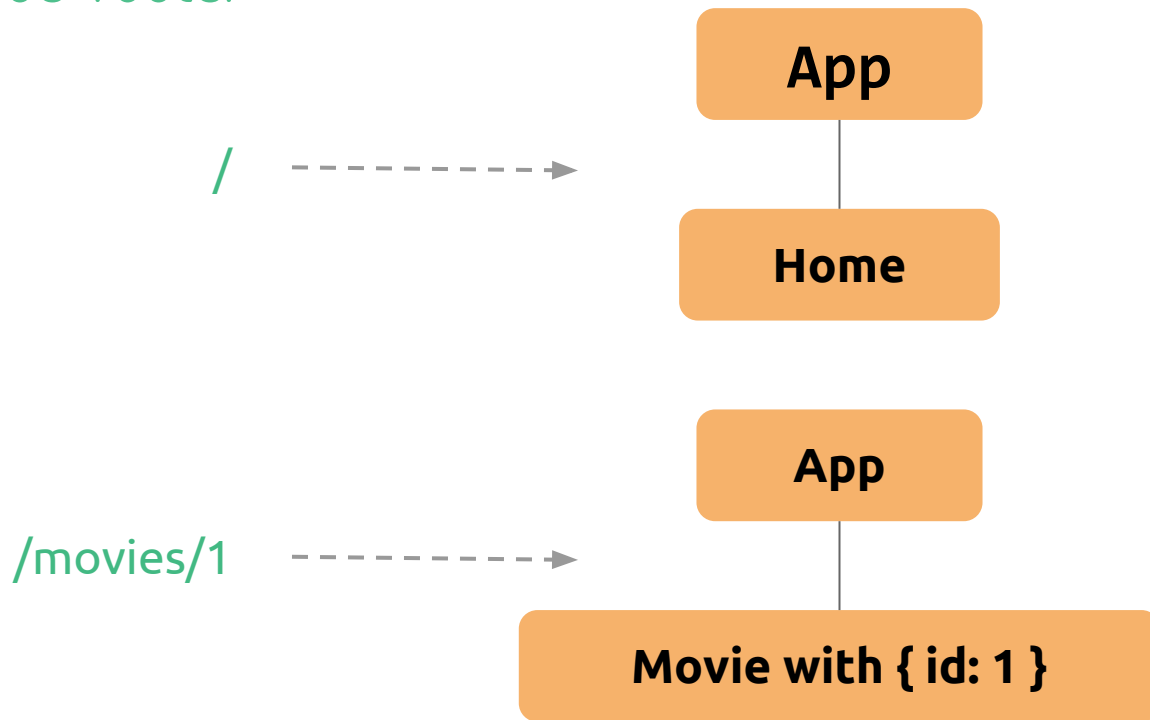
```
<a v-link="{ path: '/movies' }">Movies</a>  
<a v-link="{ path: '/settings' }">Settings</a>  
<router-view></router-view>
```



```
var App = Vue.extend({})  
var router = new VueRouter()  
router.map({  
  '/movies': {  
    component: Movies  
  },  
  '/settings': {  
    component: Settings  
  }  
})  
router.start(App, '#app')
```



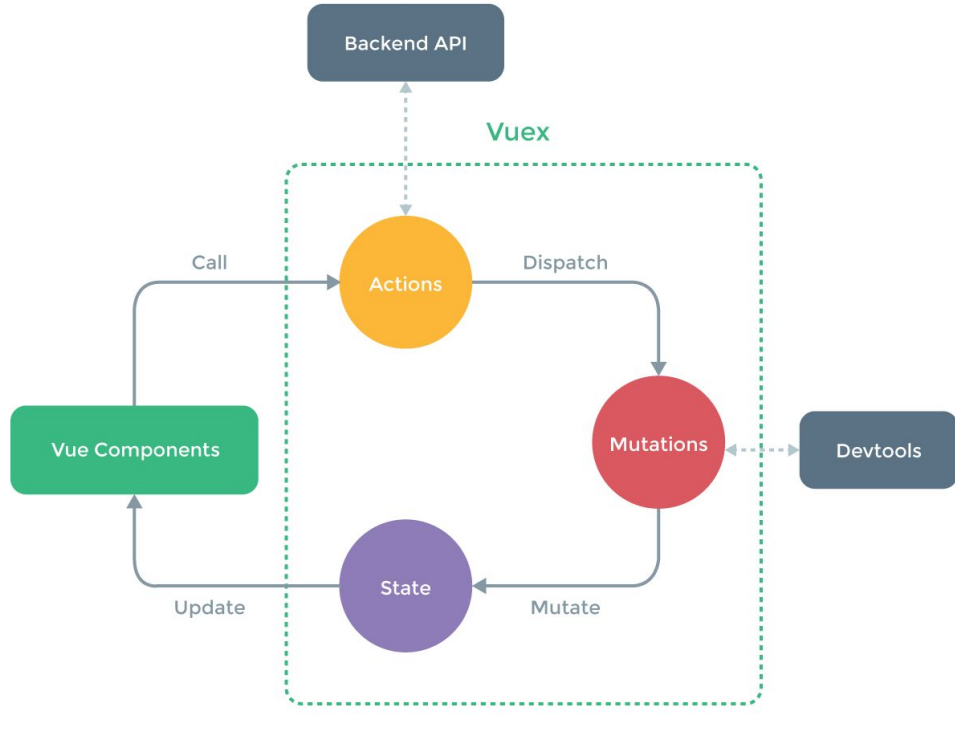
Routing with vue-router



State management

State management with vuex

- Shared vs. scattered state
- One-way data flow
- Centralized store
- Alternatives:
 - Flux
 - Redux



Development Tools

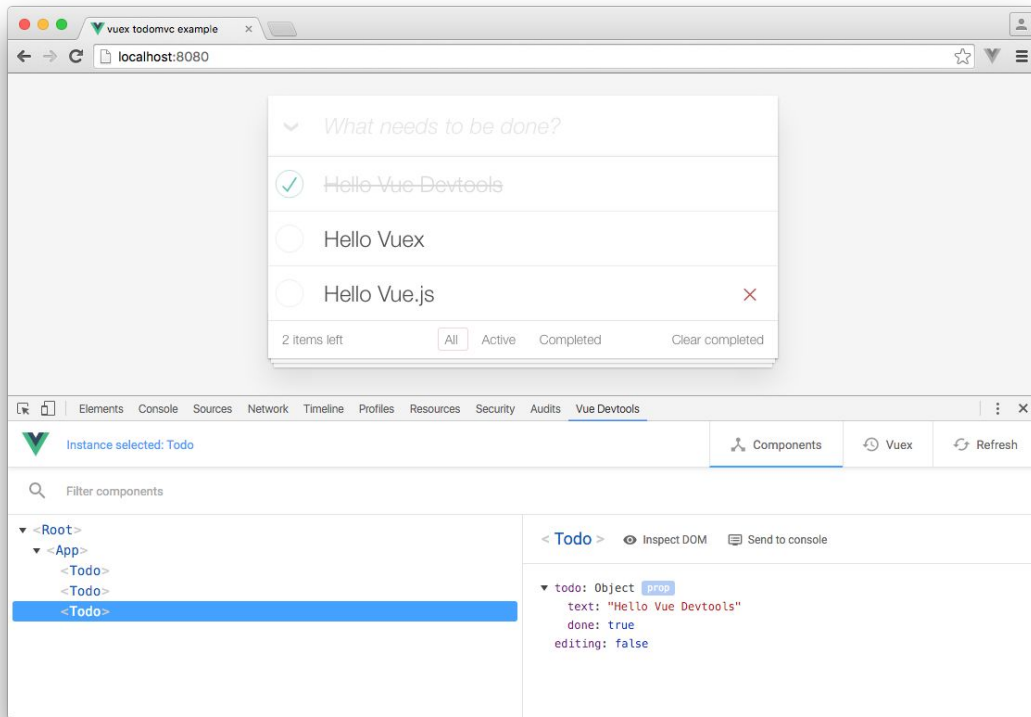
Development Tools

- Documentation
 - vuejs.org/guide
- Scaffolding tool
 - [vue-cli](https://cli.vuejs.org/)
- Laravel Elixir (aka Gulp)
 - [laravel-elixir-vue](https://laravel-elixir-vue.com/)
- Vue Devtools (Chrome)
 - *Time traveling* with [vuex](https://vuex.vuejs.org/)
- Hot Reloading
 - [vue-loader](https://vue-loader.vuejs.org/)
 - [vueify](https://vueify.vuejs.org/)

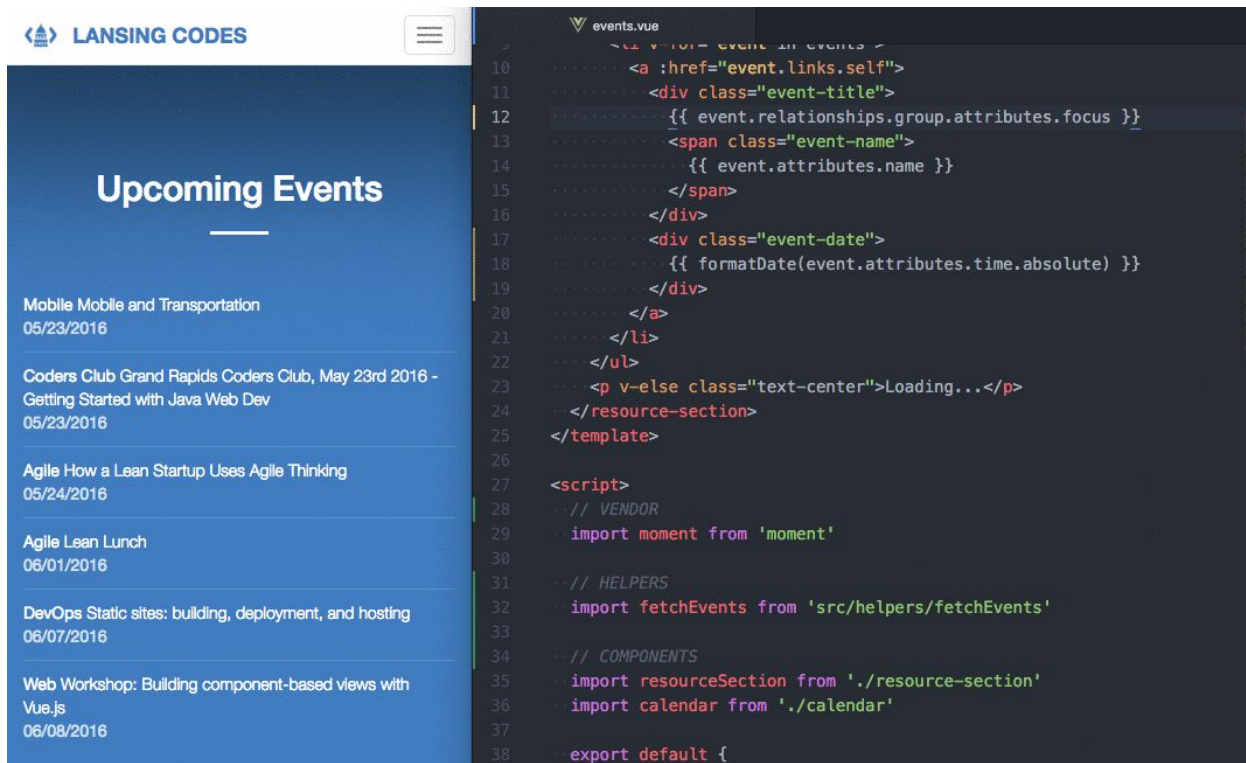
```
$ npm install -g vue-cli
$ vue init <template-name> <project-name>
$ vue init webpack my-project
$ vue init browserify my-other-project
```



Development Tools – Vue Devtools



Development Tools – Hot Reloading



The image displays a web application interface on the left and its source code on the right. The web application, titled 'LANSING CODES', features a blue sidebar with the heading 'Upcoming Events'. It lists five events: 'Mobile Mobile and Transportation' (05/23/2016), 'Coders Club Grand Rapids Coders Club, May 23rd 2016 - Getting Started with Java Web Dev' (05/23/2016), 'Agile How a Lean Startup Uses Agile Thinking' (05/24/2016), 'Agile Lean Lunch' (06/01/2016), and 'DevOps Static sites: building, deployment, and hosting' (06/07/2016). The source code on the right is a Vue.js component named 'events.vue'. It uses a template to render event links and details, and a script section for imports and logic. The code is numbered from 10 to 38.

Web Application Interface (Left):

Upcoming Events

- Mobile Mobile and Transportation
05/23/2016
- Coders Club Grand Rapids Coders Club, May 23rd 2016 - Getting Started with Java Web Dev
05/23/2016
- Agile How a Lean Startup Uses Agile Thinking
05/24/2016
- Agile Lean Lunch
06/01/2016
- DevOps Static sites: building, deployment, and hosting
06/07/2016
- Web Workshop: Building component-based views with Vue.js
06/08/2016

Source Code (Right):

```
10 <li v-for="event in events">
11   <a href="event.links.self">
12     <div class="event-title">
13       <span class="event-name">
14         {{ event.relationships.group.attributes.focus }}
15       </span>
16     </div>
17     <div class="event-date">
18       {{ formatDate(event.attributes.time.absolute) }}
19     </div>
20   </a>
21 </li>
22 </ul>
23 <p v-else class="text-center">Loading...</p>
24 </resource-section>
25 </template>
26
27 <script>
28   // VENDOR
29   import moment from 'moment'
30
31   // HELPERS
32   import fetchEvents from 'src/helpers/fetchEvents'
33
34   // COMPONENTS
35   import resourceSection from './resource-section'
36   import calendar from './calendar'
37
38   export default {
```



Vue.js 2.0

Vue.js 2.0

- 2 - 4x faster
- 12 - 17 kb min + gzip (vs. React 15 with 44kb min + gzip)
- Virtual DOM (still reactive)
- [Streaming Server Side Rendering \(SSR\)](#)
- JSX/Hyperscript rendering and/or HTML templates

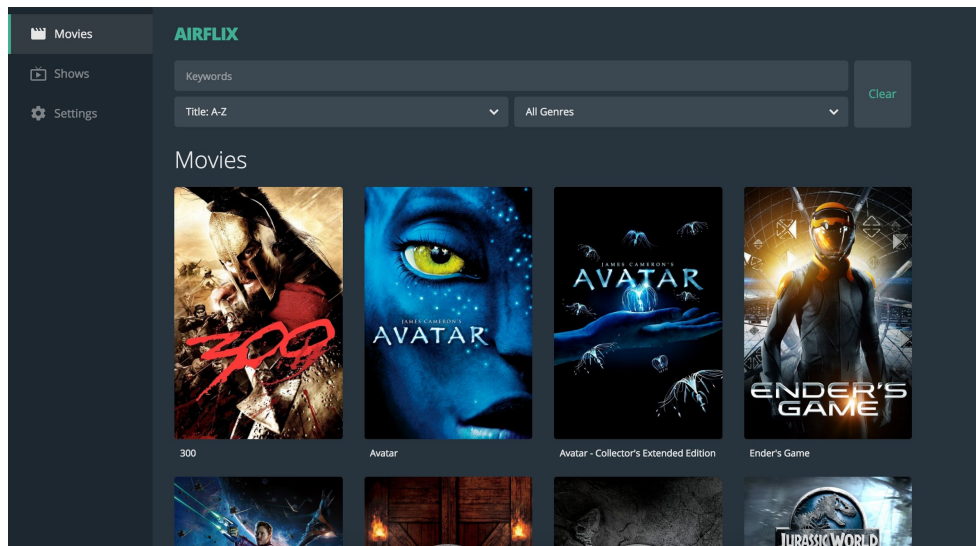
<https://rc.vuejs.org/>



Vue.js In Practice

Project Idea

- AirPlay movies from server
- Simple clean interface
- JSON-API 1.0
- Laravel
- Vue.js



Live Demo

Additional Resources

Get the slides: <https://goo.gl/yxU0w7>

Learn all the things!

- vuejs.org
- laravel.com
- laracasts.com
- github.com/wells/airflox

Why use Vue.js?

- [It is very fast](#)
- [People actually use it](#)
- [Comparison with React](#)
- [Backed by the community](#)

Books

- [You Don't Know JS - Book Series](#)
- [JavaScript: The Good Parts](#)

