Look at You, Vue: Chariot SPA Day 2018

Speaker - Ken Rimple

What is Vue?

What is Vue?

- A single-page JavaScript framework
- Created by Evan You, a former Googler who worked on AngularJS projects

Design goal

"I figured, what if I could just extract the part that I really liked about Angular and build something really lightweight without all the extra concepts involved?"

— Source: Wikipedia

Key Vue Technologies

- The Vue instance
- Vue components, directives, filters
- Vue Mixins add *features* to Vue
  - The Vue Router
  - VueX - an application state manager
  - Numerous component libraries

The Vue Instance

Vue apps start with a Vue Instance

```
var app = new Vue({
   el: '#app',                                       (1)
   data: { ... },                                    (2)
   computed: {...},                                  (3)
   methods: { ... },                                 (4)
   filters: { ... },                                 (5)
   ... etc ...
});
```

1 The element to assign this Vue instance to
2 The properties exposed and watched by Vue
3 Computations of derived data to expose
4 Methods to execute from events
5 Data transformation functions

Demo: a simple app, without ECMAScript modules…

Vue Components...

Components...

- Hold state (with instance variables)
- Accept incoming parameters
- May emit events to parent components
- Define their own methods
- Wire methods to DOM events
- Contain their own templates
- Are styled via CSS

Component Syntax

Components can be globally registered

```
Vue.component('alert', {
  props: ['message'],                                    (1)
  name: 'alert',                                         (2)
  template: `<div>Alert:  {{message}}</div>`             (3)
});
```

1 Define props to accept input data
2 Define the component name
3 Define a template to render the component

4.3

Component mounted in an app, bound to an element…

```
Vue.component('alert', {
  props: ['message'],
  name: 'alert',
  template: `<div class='box'>ALERT: {{message}}</div
});

const app = { template: `
   <div><alert :message="'10 minutes left'"></alert><
` };

new Vue({
  el: '#app',
  render: view => view(app)
});
```

Most developers use Single-File Components instead…

Single File Component: `alert.vue`

```
<template>
  <div class="box">{{ message }}</div>
</template>

<script>
  export default {
    el: 'alert',
    props: ['message']
  }
</script>

<style>
  .box { ... }
</style>
```

SFCs and Styles

With **node-sass** installed you can compile **sass** templates in SFCs

```
<style lang="scss">
  $input-radius: 6px;

  .form {
    select {
      border-radius: $input-radius;
    }
  }
</style>
```

SFCs require a Webpack build and the Vue compiler

4.6

Vue Style Scoping

You can namespace your SASS.

```
<style lang="scss" scoped>                                   (1)
  input {
    background-color: red;
  }                                                          (2)
</style>
```

1 Now the mounted component and styles have an attached hash attribute, computed for this component and *its children*

2 The input style won’t bleed to other external components

4.7

Advanced: Vue Styles with CSS Modules

Vue also supports SASS Modules

```
<template>
  <input class="$styles.foo"  />
</template>
<style lang="scss" module>
  input.foo {                                         (1)
    background-color: red;                            (2)
  }
</style>
```

1 Adding module to the style declaration enables CSS Modules

2 You must add classes for all HTML elements, or they will affect everything (are not namespaced without an attached class)

4.8

Good Blog Article on Scoped styles -vs- CSS Modules

http://www.netguru.co/codestories/vue.js-scoped-styles-vs-css-modules

```
Use CSS Scopes for simple designs, and CSS Modules when you want
precise control for widget libraries
```

4.9

Changing Data

Using the **`created`** lifecycle method - https://codesandbox.io/s/k58z4qvv3

```
export default {
  data: function() { return {msg: 'Hello'}; },
  created: function() {
    setTimeout(() => {
      this.msg = 'Goodbye!';
    }, 4000);
  },
  template: `<alert :message="msg" />`
}
```

Objects placed in the data property are reactive and update their view when changed

4.10

Vue and Events

A simple method in a SFC - https://codesandbox.io/s/4w6j0m8vn4

```
<template>
  <button @click="poke">Poke!</button>
  <span> Poked {{ pokeCount }} times</span>
</template>
<script>
  export default {
    data: function () { return { pokeCount: 0 }; },
    methods: {
      function poke() {
        this.pokeCount++;
      }
    }
  }
</script>
```

4.11

A simple Vue Form

```
<form class="form" @submit.prevent="submit">
 <input type="text" required v-model="form.name">(1)
 <input type="text" required v-model="form.email">(2)
 <select required v-model="form.treatment">      (2)
    <option v-for="(option, index) in treatments"
           :key="index">                         (3)
      {{ option }}
    </option>
 </select>
 <button>Register!</button>
</form>
```

1 @submit event, modified with .prevent to prevent post action

2 v-model binds form fields to data properties, allows for validation

3 v-for iterator similar to AngularJS / Angular, needs :key to improve re-render performance like React

4.12

Vue Router

Vue Router

- A router developed by the Vue team
  - Provides component navigational management
  - Configured using the `Router` class

Sample Router config

provide in a file like **`router-config.js`**

```
// Add Router mixin...
Vue.use(Router);

export default new Router({
  routes: [
    {path: '/', redirect: '/schedule'},
    {path: '/schedule', component: Registration},
    {path: '/register/:id', component: Registration},
    ...
  ]
});
```

`Vue configures routes in JavaScript, similar to Angular`

Mounting the Router

Add the Router to the Vue instance properties

```
import router from './router-config.js';

new Vue({
  data: function() { ... },
  ...
  router,                                         (1)
  ...,
  render: h => h(App)
}).$mount('#app');
```

1 Shorthand for router: router, a feature of ECMAScript 2015.

Once installed, the $route and $router properties are made available to components

5.4

Using the Router

```
<template>
  <h1>SPA Day</h1>
  <router-link to="/">Home</router-link>
  <router-link to="/registration">Register</router-linl
  <hr/>
  <router-outlet></router-outlet>
</template>
```

the Vue Router router-outlet component renders all views below the horizontal rule

Other Router Link Examples

```
<template>
  <router-link to="/register/12">
      Register for session 12
  </router-link>

  <router-link
     :to="{ path: '/register', params: {id: currentId}
      Register
  </router-link>
</template>
```

currentId above could be a data or computed property

5.6

VueX State Management

VueX is…

- A state management library that…
  - Maintains state as an object graph
  - Dispatches *actions* to request state changes
  - Manages state with *mutations*

A VueX store

provide in a file like **`store-conf.js`**

```
// Add the VueX mixin to Vue
Vue.use(Vuex);

// Create the store...
export default new Vuex.Store({
  state: {...},
  actions: {...},
  mutations: {...},
  getters: {...},
...
})
```

6.3

Mounting VueX in your Vue instance

```
import store from 'store-conf.js';

new Vue({
  data: function() { ... },
  ...
  router,
  store,
  ...
  render: h => h(App)
}).$mount('#app');
```

Once installed, components can access the $store property and helper methods

Installing the store in your Vue instance

Import the store and install it as a property in your Vue instance

```
import store from './store';
...
new Vue({
  router,
  store,
  ...
}).$mount('App');
```

6.5

VueX and Redux compared

Activity	VueX	Redux
Changes to State	"Simple" Mutation Methods	Immutable Reducer Function
Request A Change	Call an Action	Dispatch an Action
Bindings to Components	`mapState`, `mapAction` helpers, `getters`	Higher-order Components and connect
Asynchronous Operations	Action methods (built-in)	Middleware-driven (select one)
Challenges	Most mutations just work, except collections…you need to replace them or use a supported function (like filter)	Everything *should* be treated as immutable, `reducers` can be tough to reason about

Wrap-up

Strengths

- Vue *single page components* nicely contain scripts, styles, templates
- *VueX* is not as complicated as Redux with its reducer and forced immutability everywhere
- The *Vue CLI* is well thought-out and does just what it needs to do

Strengths

- Vue *slots* make wrapping nested content nicer than Angular
- `Typescript` support in CLI
- CLI supports modifying Webpack config without 'ejecting'
- `Chrome and Firefox Vue DevTools` is useful and cover components, events, VueX

7.3

Weaknesses

- No built-in service layer
- No built-in dependency injection system
- Pedantic eslint / tslint settings with `Vue CLI`
- Vue's exceptions are not wonderful. They tend to be cryptic and confusing
- Vue feels like a riff on AngularJS and React

Idiosyncracies

- VueX (and Vue) state mutation is a bit complex - since it holds proxies it can be tough to figure out what’s going on
- Things to watch with VueX…
  - You *can* mutate data without making it immutable, not in all cases
  - Not if you normally use `push` to add to an array
  - Instead use a new object - `[ …oldobj, newentry ]`
  - Or use `Vue.set` to make changes

7.5

Code Walkthrough: The SPA App