Vue

slides at https://github.com/mvolkmann/talks

R. Mark Volkmann
Object Computing, Inc.
http://objectcomputing.com
Email: mark@objectcomputing.com
Twitter: @mark_volkmann
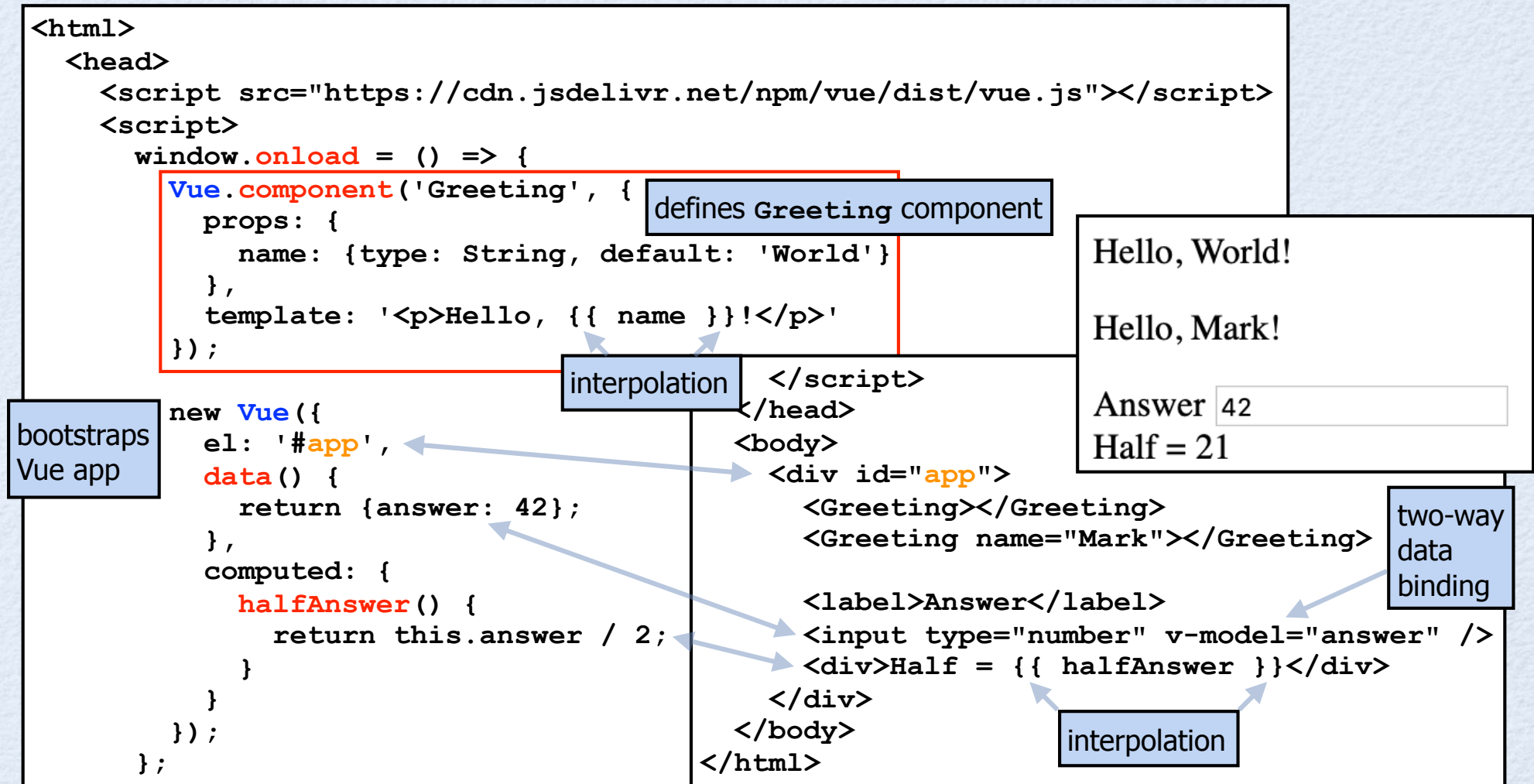GitHub: mvolkmann
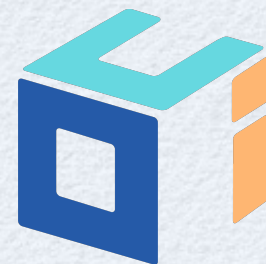
OCI | TRAINING

Outline

1. Intro - exercise
2. Tools
3. Components - exercise
4. Templates - exercise
5. Styling
6. Todo App
7. CLI
8. Browser Devtools - exercise
9. Component Lifecycle - exercise

10. Events
11. Slots
12. REST - exercise
13. Router - exercise
14. Vuex
15. Vue vs. React
16. Jest
17. Cypress
18. Storybook

Vue Overview

- A web application framework that competes with frameworks such as React, Angular, and Svelte

- Generally viewed as easier to use than Angular

- Appeals to developers that dislike use of JSX in React

  - though Vue also supports JSX

- Uses a Virtual DOM like React

  - unlike Angular and Svelte

  - can quickly determine changes to actual DOM that are needed

- Small runtime - ~20KB minified and gzipped

- Implemented in TypeScript

No Build Steps Needed

- While this can be done, it is mostly only for demo apps

```html
<html>
  <head>
    <script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
    <script>
      window.onload = () => {
        Vue.component('Greeting', {      defines Greeting component
          props: {
            name: {type: String, default: 'World'}
          },
          template: '<p>Hello, {{ name }}!</p>'
        });                              interpolation
                                   </script>
                                   </head>
        new Vue({                  <body>
          el: '#app',               <div id="app">
bootstraps   data() {               <Greeting></Greeting>
Vue app        return {answer: 42};  <Greeting name="Mark"></Greeting>
          },
          computed: {               <label>Answer</label>
            halfAnswer() {           <input type="number" v-model="answer" />
              return this.answer / 2;  <div>Half = {{ halfAnswer }}</div>
            }                       </div>
          }                        </body>
        });                        </html>
      };
```

Hello, World!

Hello, Mark!

Answer 42
Half = 21

two-way data binding

interpolation

4

Getting Real

- Most real Vue apps start with Vue CLI

- Can create file structure for new projects

- Can add features to existing projects

  - warning: commit existing files to version control in case it overwrites some files

- Configures use of Babel for transpiling

- Configures use of Webpack for module bundling

- Provides npm scripts to
  `serve` the app and `build` for production

- Can request support for
  linting, unit tests, end-to-end (E2E) tests, and more

Installing and Running

- Install Node from https://nodejs.org

- To install, `npm install -g @vue/cli`

- Two ways to run
  - command line
  - web UI

- We'll cover basic command line usage now

- Other Vue CLI options will be covered later

Create and Run a Project

- Create a new project
  - `vue create project-name`
  - press enter to select "default" preset
  - runs for a minute or so
- Follow directions output at end
  - `cd project-name`
  - `npm run serve`
  - starts a local HTTP server
- Browse local URL this outputs
  - doesn't automatically open a tab in default browser
  - http://localhost:8080 unless something else is already listening on that port
  - bookmark this

Welcome to Your Vue.js App

For a guide and recipes on how to configure / customize this project, check out the vue-cli documentation.

Installed CLI Plugins

babel   eslint

Essential Links

Core Docs   Forum   Community Chat   Twitter   News

Ecosystem

vue-router   vuex   vue-devtools   vue-loader   awesome-vue

Project Structure

- Projects created by Vue CLI have following structure

  - `node_modules` - contains modules used by project

  - `public`

    - `favicon.ico` - icon displayed in browser tab

    - `index.html` - contains `<div id="app"></div>`

  - `src`

    - `main.js` - bootstraps application targeting `#app`

    - `App.vue` - top/root component

    - `views` directory - contains definitions of Vue components that represent "pages" (if "Router" requested)

    - `components` directory - contains definitions of Vue components used on pages

    - `assets` directory - contains images and other media files

    - `router.js` - configures use of vue-router for page routing (if "Router" requested)

    - `store.js` - configures use of Vuex for state management (if "Vuex" requested)

    - `package.json` - lists project dependencies and defines npm scripts

    - `package-lock.json` - lists specific versions of project dependencies to "lock" to them

    - `babel.config.js` - configures Babel transpiler

    - `.gitignore` - lists directories and files that should not be committed in Git

8

Things Nobody Does

- While many things are possible in Vue,
  there are some things almost nobody does

  - start a project without using Vue CLI

  - create components using the `Vue.component` function
    instead of using single file components (SFCs) defined in `.vue` files

  - use `v-bind:` instead of the `:` shorthand

  - use `v-on:` instead of the `@` shorthand

- More on all of these later

Exercise

- Install Node.js
  - browse https://nodejs.org
  - click one of the green buttons
  - double-click downloaded installer to run it

- Create a new project using Vue CLI
  - `npm install -g @vue/cli`
  - `vue create my-project`

- Run it
  - `cd my-project`
  - `npm run serve`
  - `browse localhost:8080`

- Files for upcoming exercises are at
  https://github.com/mvolkmann/vue-workshop
  - clone the repo or download a zip file