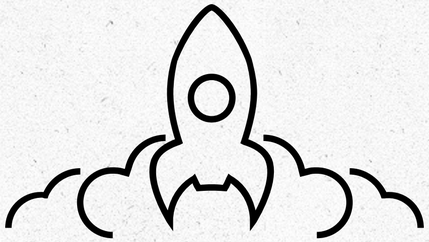


From MVC to React

@ericclemmons

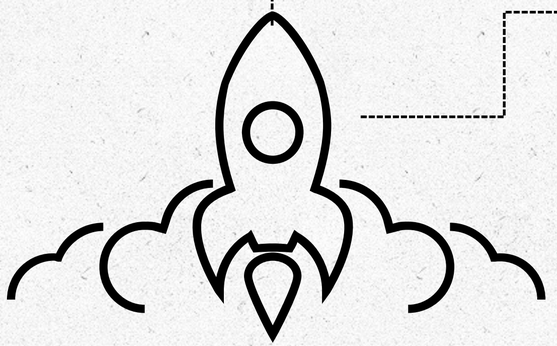
History

How this all started

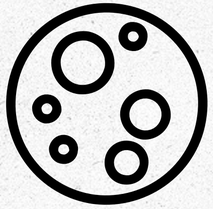
2011
Prototype reached peak success.

2011
Grew team & developed prototype.

2010
Joined CMN.com



2015



Explosive growth & optimization.

2012

Threw away the prototype.

A/B/x Testing

Responsive UI
Mobile vs. Desktop assets

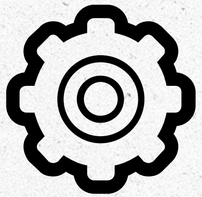
Performance
Biggest impact on revenue.

Dismantling the Monolith

From PHP to Node

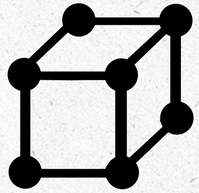
Admin

API



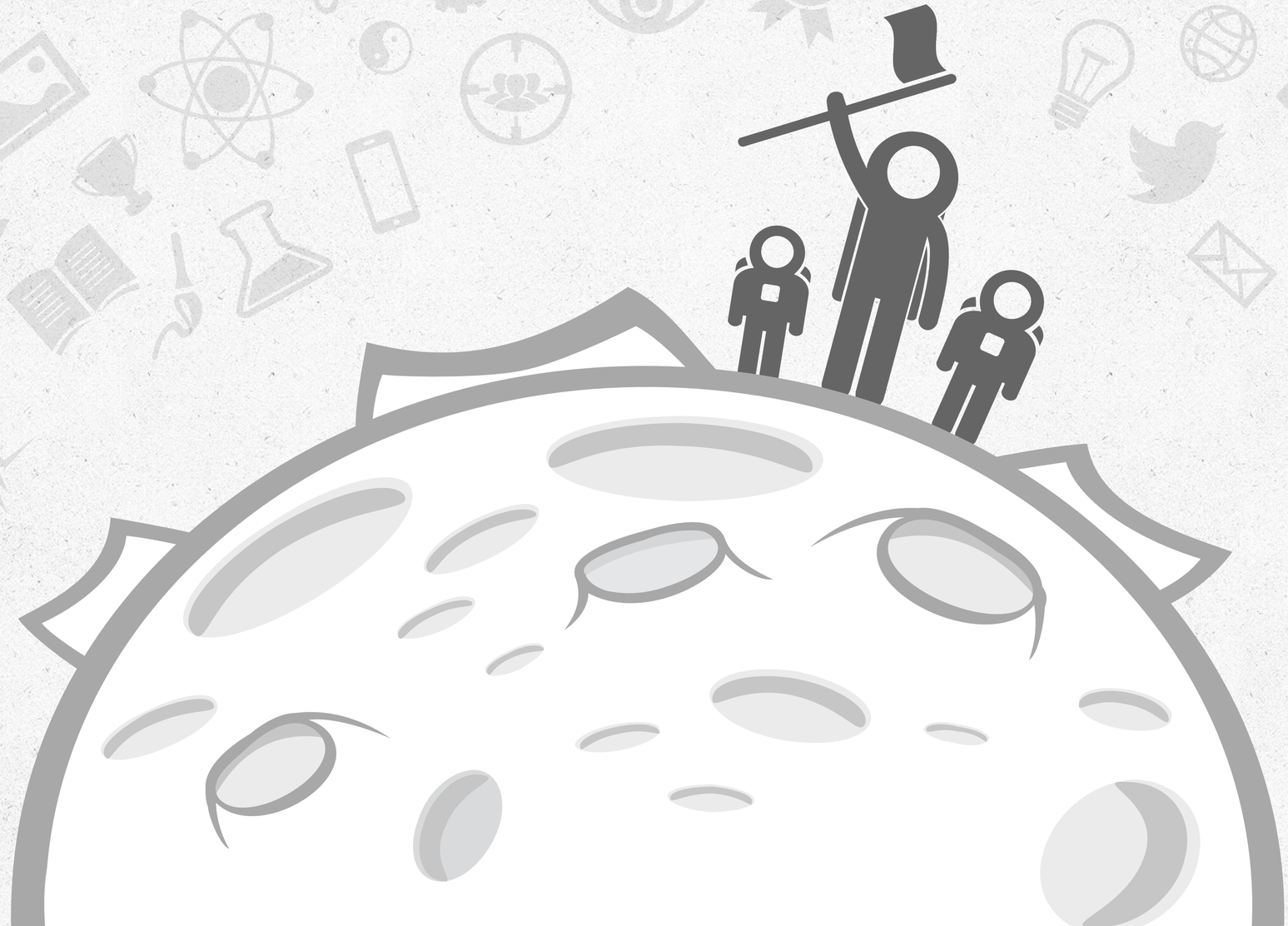
Business Logic

UI

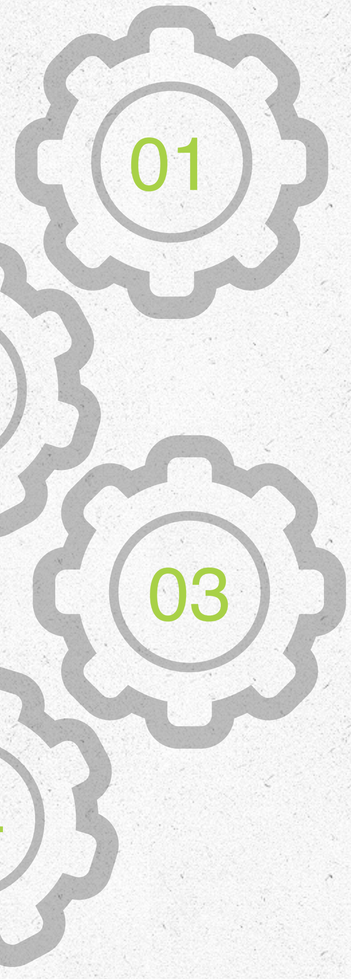


Profit?

Javascript is awesome!



Requirements



Isomorphic

Rendering on the server is just *smart*.

Simple

Conceptually easier to understand how logic works.

Modular

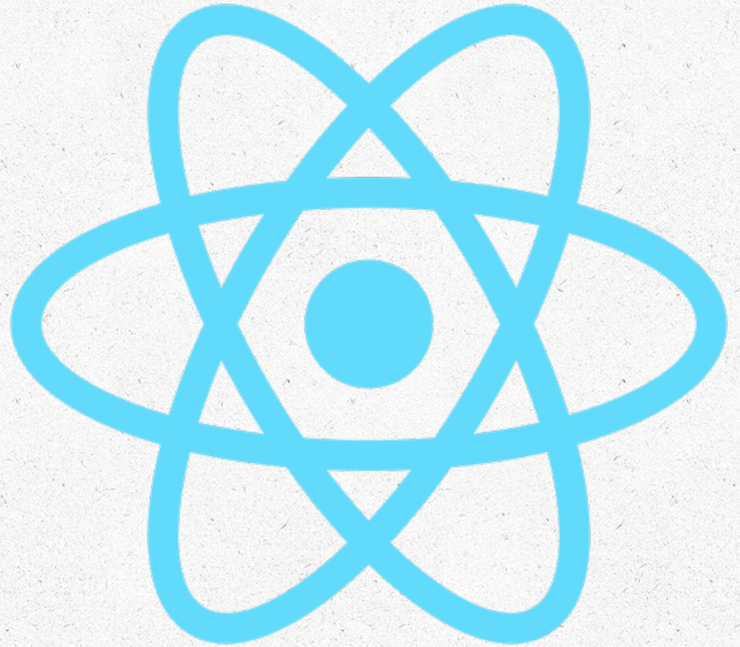
Technology moves fast enough to justify replacing parts within a year.

Fast

Server responses, rendering, validation.

The Solution

As of 2015



React

Facebook

Benefits

That match our requirements

Adoption

Companies with much larger problem sets are leverage React.

Community

Active F/OSS developers supporting their tools and React in mediums such as Slack pushing Javascript forward.

Isomorphic

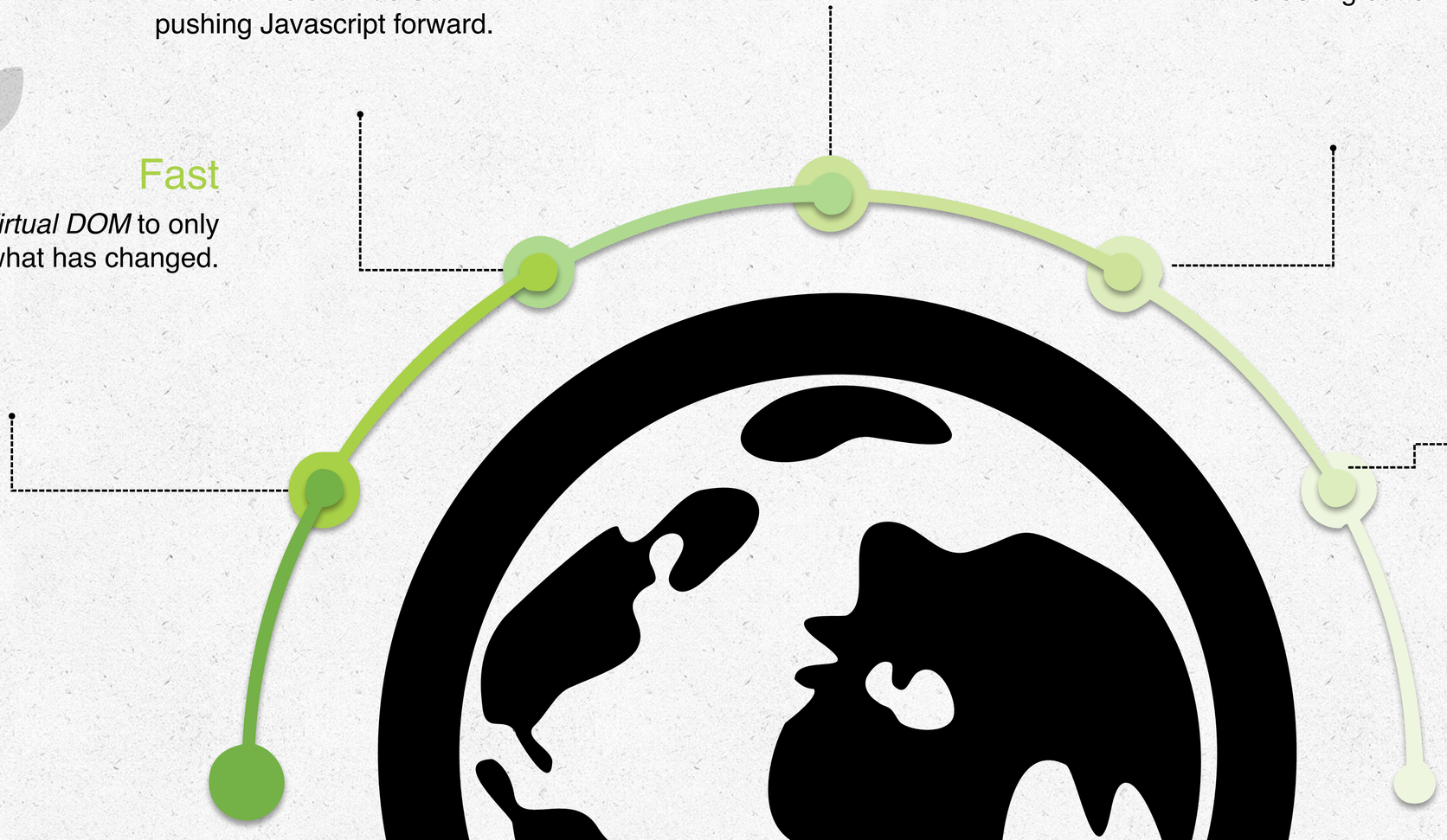
Server-side & client-side rendering out of the box.

Fast

Utilizes a *Virtual DOM* to only render what has changed.

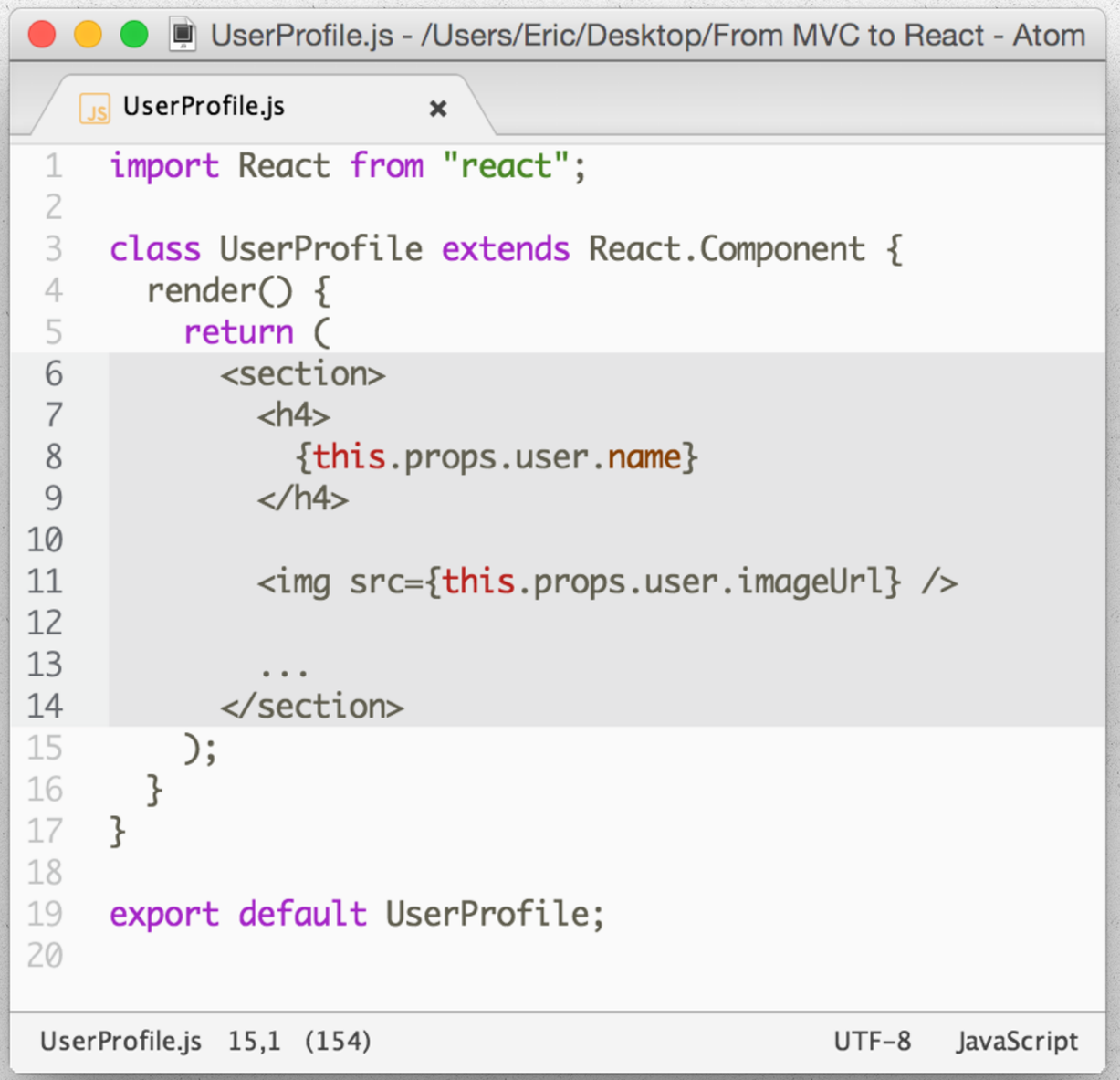
Simple

Just the View layer, the rest of the architecture is up to you.



What does it look like?

Simple Example



server.js - /Users/Eric/Desktop/From MVC to React - Atom

UserProfile.js | client.js | server.js

```
1  import React from "react";
2
3  import UserProfile from "./components/UserProfile";
4
5  const markup = React.renderToString(<UserProfile user={user} />);
6
7  reply(markup);
8
```

server.js 5,66 (65)

client.js - /Users/Eric/Desktop/From MVC to React - Atom

UserProfile.js | client.js

```
1  import React from "react";
2
3  import UserProfile from "./components/UserProfile";
4
5  React.render(
6    <UserProfile user={user} />,
7    document.getElementById("app")
8  );
9
```

client.js 6,22 (4) UTF-8 JavaScript

Components

Allows the author to describe *what* the UI consists of (e.g. TableGrid, UserProfile, CodeEditor, etc.) rather than *how.*

Properties

Usually shown as HTML-like attributes, these are external values explicitly passed down to components by their parent.

State

Internal properties that changes how a component is rendered (e.g. "on" vs. "off").

Context

Advanced, scarcely documented feature that allows deeply-nested components to opt-in to a shared state for data-sharing and communication.

From MVC...

What we know...

Model

Domain logic

View data

Constraints

Validation

?

Controller

Handles request

Fetches data

Performs logic

Renders view

Sends response

?

View

Fast

Simple

Isomorphic

One-way data flow

React

How does our current app work?

High-level overview

Request

A request is made to our server.

Cart Page

User wants to adjust what's in their shopping cart.

Front Dispatcher

Entry-point receives the request.

Router

The router identifies which route(s) can respond to the request.

Controller

The controller fetches the data it designates useful for the view.

View

The view uses the normalized data from the Controller to present it in a useful manner for the user.

Interaction

The user interacts with the page (e.g. a form).

Progressive Enhancement

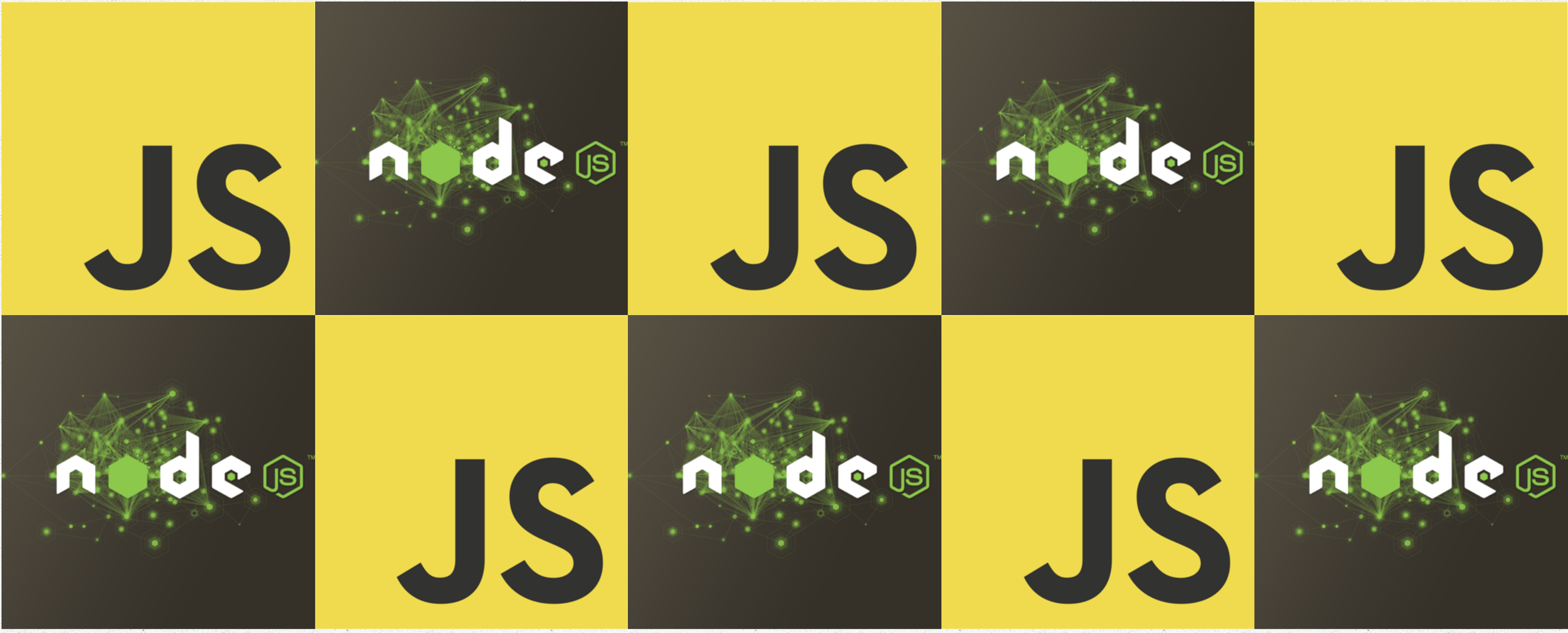
Javascript updates the view as the user interacts to save round-trips to the server.

HTTP POST

Data is sent to controller, validated, and prepared for storage.

Submit

User finalizes edits & submits the form.



Now with Javascript.

How can our Javascript app work?

High-level overview

Cart Page

User wants to adjust what's in their shopping cart.

Request

Node

Front Dispatcher

Hapi, Express, Koa, etc.

Router

Server Router + React Router

Handler

Component that handles primarily route layout.

Components

Deeply-nested components that render the view.

Interaction

User triggers actions that update props & state.

Progressive Enhancement

React re-renders the view accordingly.

AJAX POST

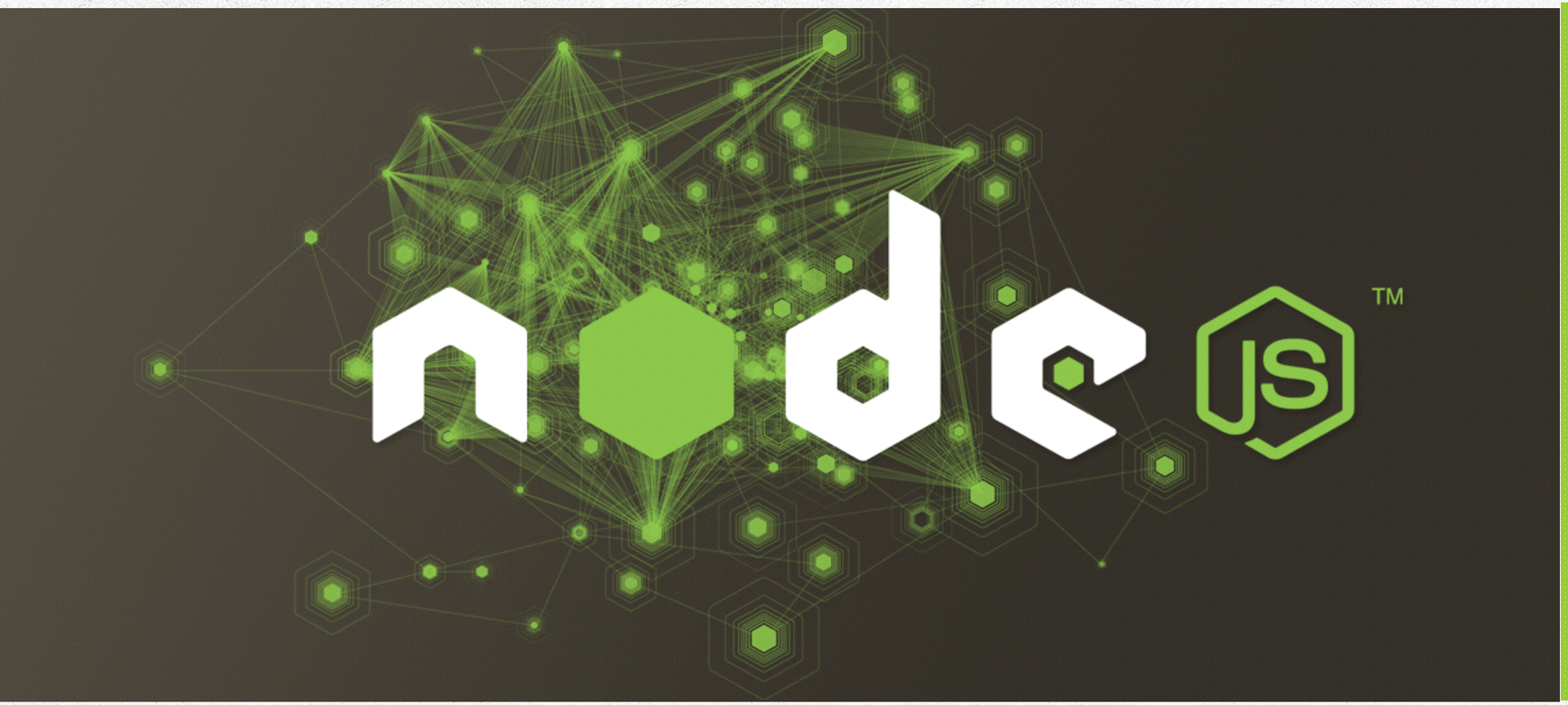
Data is validated and sent to an API end-point for storage.

Submit

User finalizes edits & submits the form.

Example Project

https://github.com/ericclemmons/mvc-to-react



Let's See Some Code!

Server

Hapi + React + ES6

server.js - /Users/Eric/Projects/ericclemmons/mvc-to-react - Atom

server.js

```
1  import Hapi from "hapi";
2  import React from "react";
3
4  const server = new Hapi.Server();
5
6  server.connection({ port: 3000 });
7
8  server.route({
9    method: "GET",
10   path: "/{path*}",
11   handler: function(request, reply) {
12     const view = React.renderToString(<h1>Howdy!</h1>);
13
14     const layout = `
15       <!DOCTYPE html>
16       <html lang="en">
17         <head>
18           <meta charset="UTF-8">
19         </head>
20         <body>
21           <div id="app">${view}</div>
22         </body>
23       </html>
24     `;
25
26     reply(layout);
27   }
28 });
29
30 server.start();
```

Client & Server

Hapi + React + Browserify + ES6

server.js - /Users/Eric/Projects/ericclemmons/mvc-to-react - Atom

client.js | server.js

```
 8  server.route({
 9    method: "GET",
10    path: "/client.min.js",
11    handler: function(request, reply) {
12      reply.file(`${__dirname}/client.min.js`);
13    }
14  });
15
16  server.route({
17    method: "GET",
18    path: "/{path*}",
19    handler: function(request, reply) {
20      const view = React.renderToString(<h1>Howdy!</h1>);
21
      const layout = `
        <!DOCTYPE html>
        <html lang="en">
          <head>
            <meta charset="UTF-8">
          </head>
          <body>
            <div id="app">${view}</div>
            <script src="client.min.js" async defer></script>
          </body>
        </html>
      `;
33
34
35      reply(layout);
36    }
37  });
```

client.js - /Users/Eric/Projects/ericclemmons/mvc-to-react - Atom

client.js | server.js

```
1  import React from "react";
2
3  React.render(<h1>Howdy there!</h1>, document.getElementById("app"));
4
```

client.js 1,1 UTF-8 JavaScript (JSX) master 8 2

Props

Client & Server

server.js - /Users/Eric/Projects/ericclemmons/mvc-to-react - Atom

server.js | Home.js

```
1  import Hapi from "hapi";
2  import React from "react";
3
4  import Home from "./components/Home";
5
6  const server = new Hapi.Server();
7
8  server.connection({ port: 3000 });
9
10 server.route({…
17
18 server.route({
19   method: "GET",
20   path: "/{path*}",
21   handler: function(request, reply) {
22     const view = React.renderToString(<Home path={request.path} />);
23
24     const layout = `
25       <!DOCTYPE html>
26       <html lang="en">
27         <head>
28           <meta charset="UTF-8">
```

server.js 23,1 (69) UTF-8 JavaScript (JSX) master 6 2

Home.js - /Users/Eric/Projects/ericclemmons/mvc-to-react - Atom

server.js | Home.js

```
1  import React from "react";
2
3  class Home extends React.Component {
4    render() {
5      return (
6        <h1>Howdy from <code>{this.props.path}</code>!</h1>
7      );
8    }
9  }
10
11 export default Home;
12
```

components/Home.js 7,1 (58) UTF-8 JavaScript (JSX) master 6 2

client.js - /Users/Eric/Projects/ericclemmons/mvc-to-react - Atom

server.js | Home.js | client.js

```
1  import React from "react";
2
3  import Home from "./components/Home";
4
5  React.render(
6    <Home path={window.location.pathname} />,
7    document.getElementById("app")
8  );
9
```

client.js* 4,1 (38) UTF-8 JavaScript (JSX) master 6 2

State

Home

Home.js - /Users/Eric/Projects/ericclemmons/mvc-to-react - Atom

server.js | Home.js

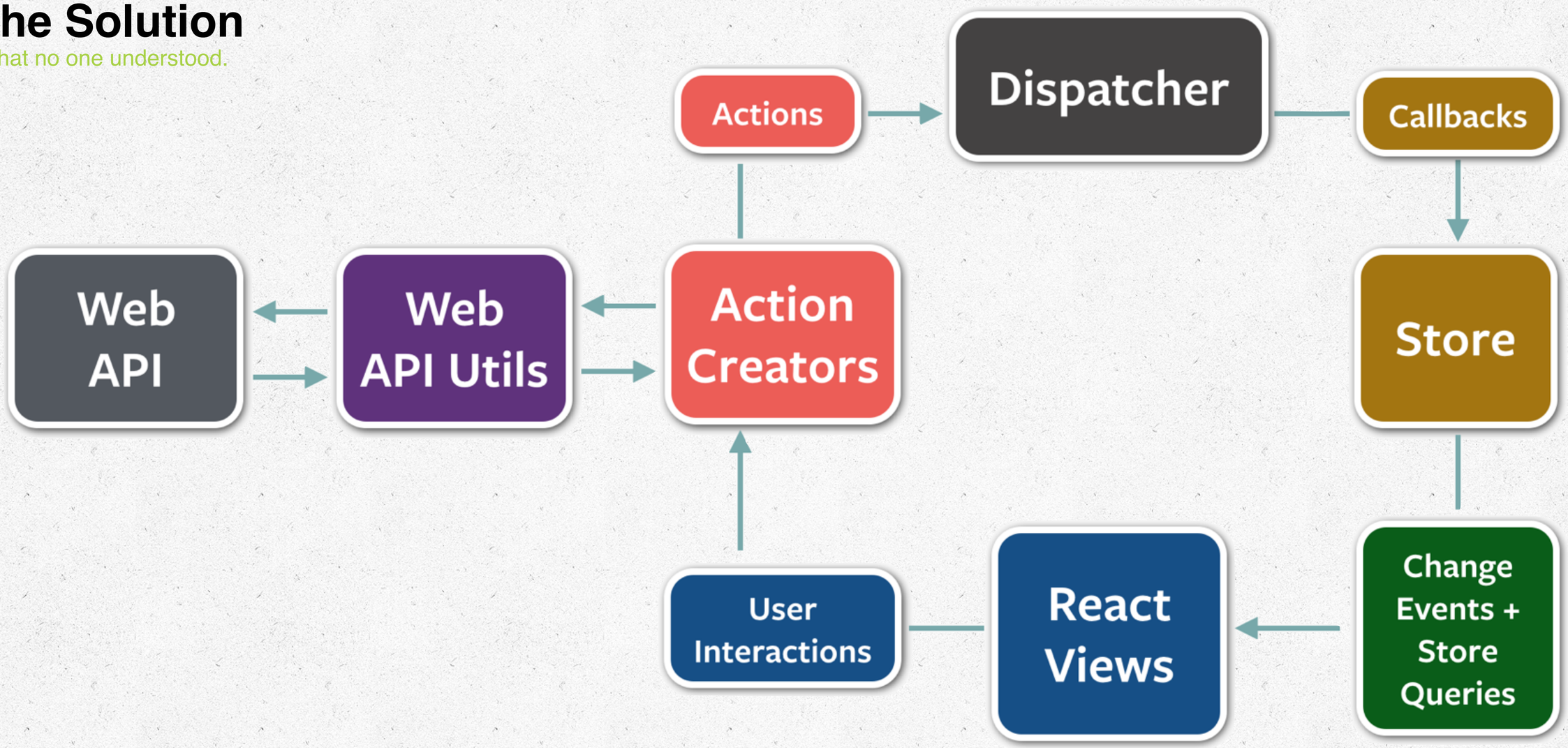
```
1  import React from "react";
2
3  class Home extends React.Component {
4    constructor() {
5      this.state = {
6        color: "black"
7      };
8    }
9
10   changeColor() {
11     const newColor = Math.floor(Math.random() * 0xffffff).toString(16);
12
13     this.setState({
14       color: `#${newColor}`
15     });
16   }
17
18   render() {
19     const styles = { color: this.state.color };
20
21     return (
22       <h1 onMouseMove={this.changeColor.bind(this)} style={styles}>
23         Howdy from <code>{this.props.path}</code>!
24       </h1>
25     );
26   }
27 }
28
```

Events

Untangling the complexity that is Progressive Enhancement.

The Solution

...that no one understood.

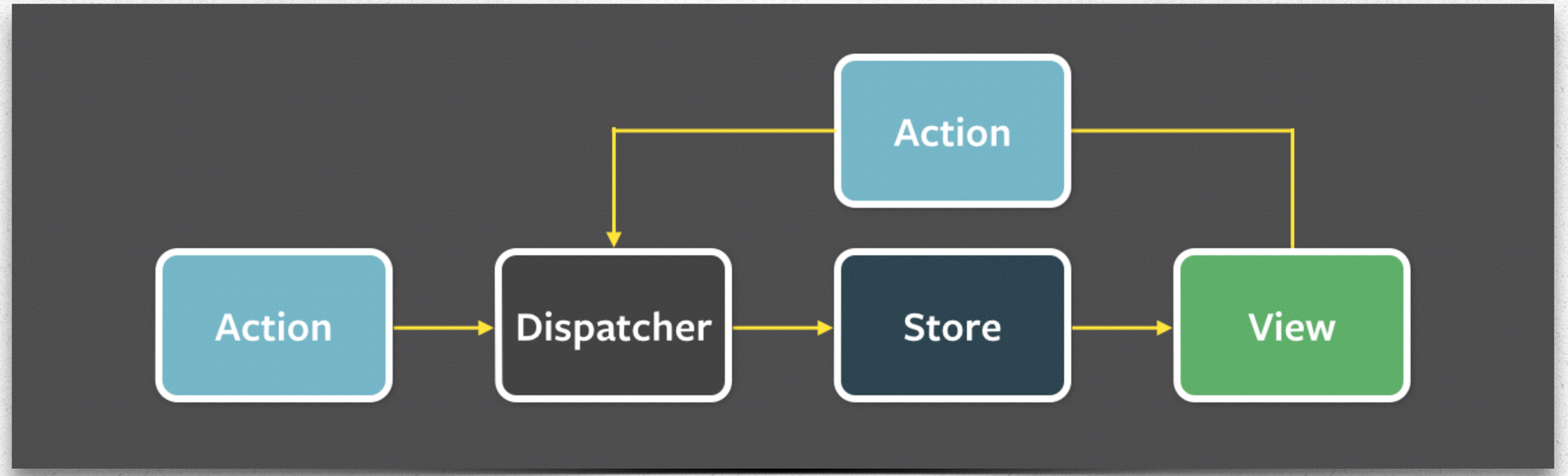


Flux

Facebook

The Solution

...that makes sense.

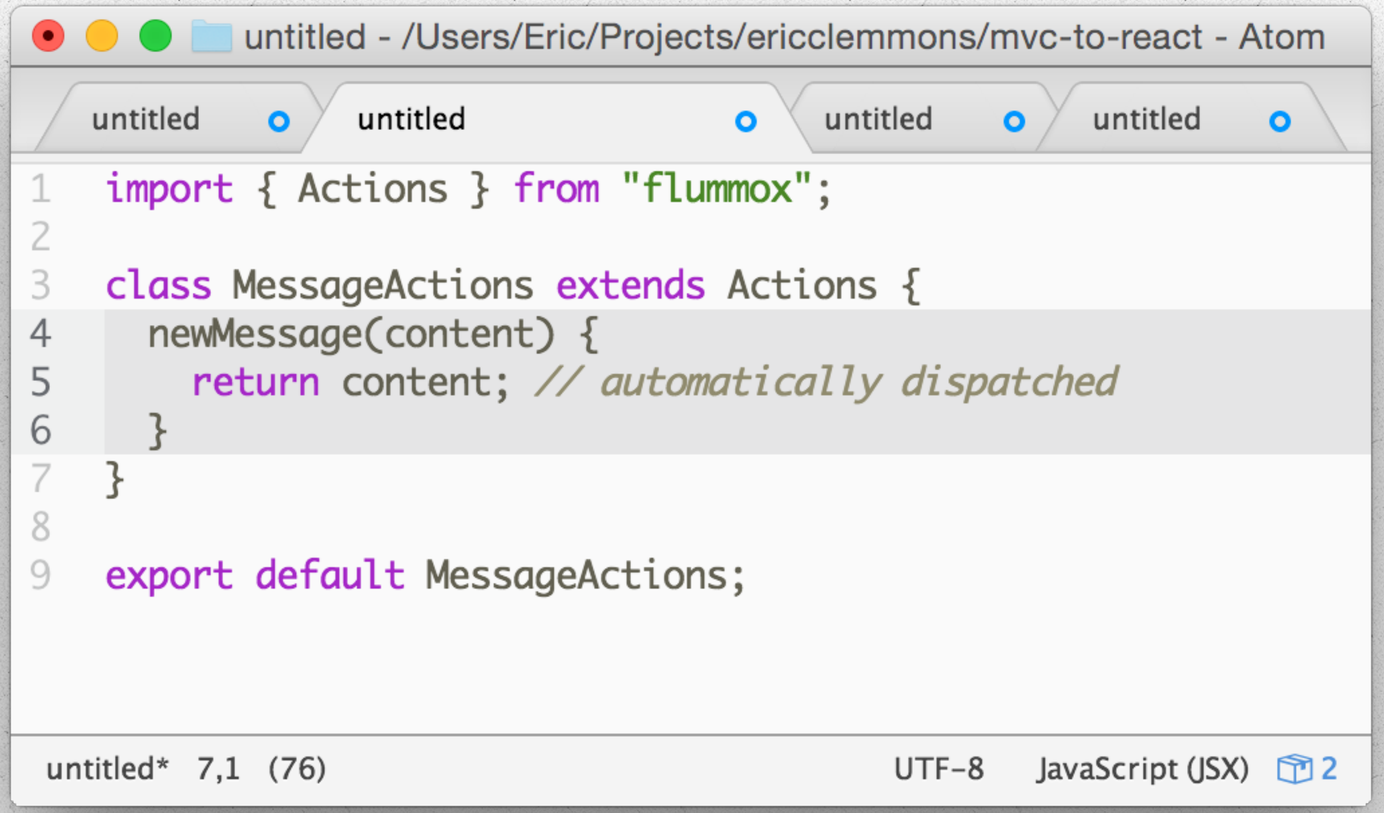


Flux

Facebook

Actions

Flummox



Store

Flummox

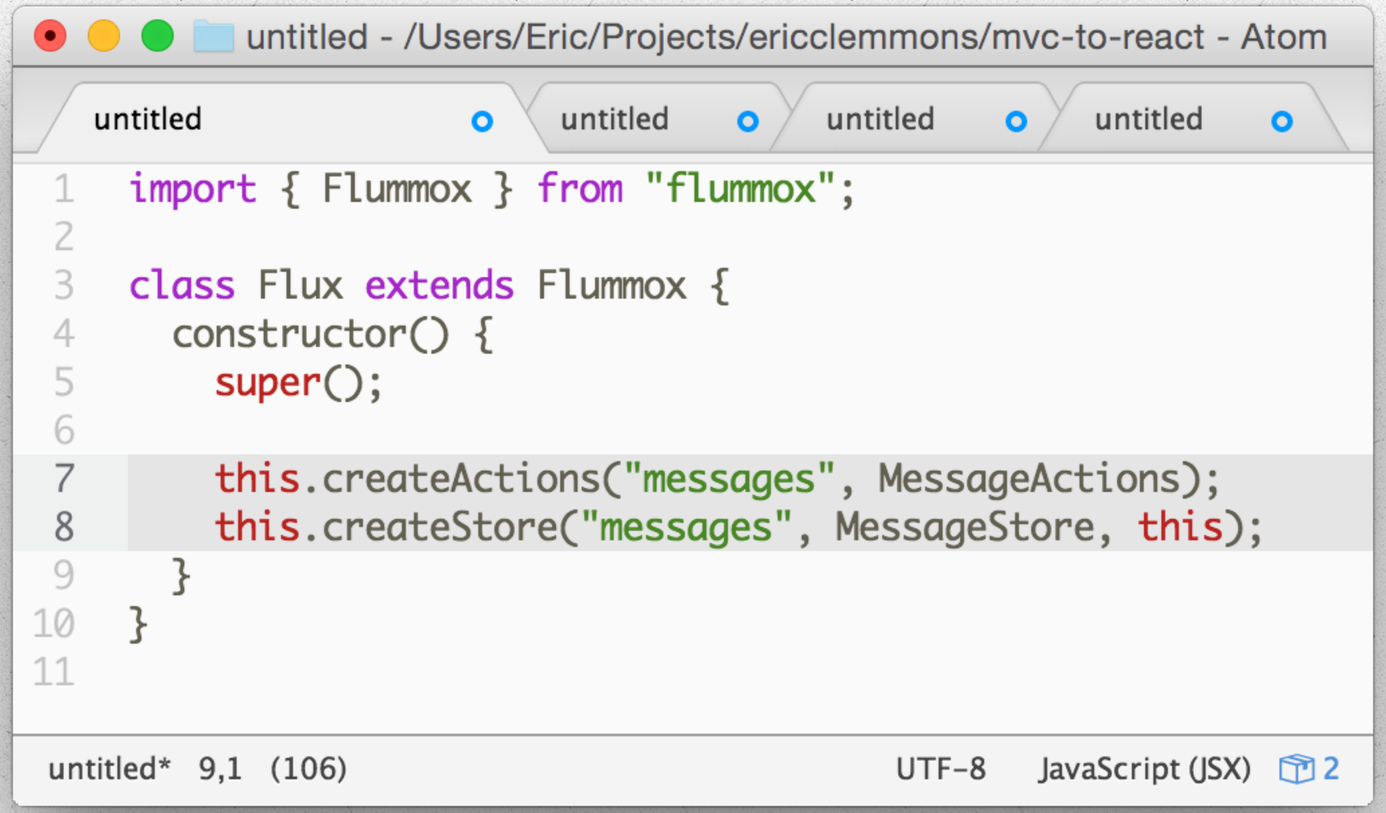
untitled - /Users/Eric/Projects/ericclemmons/mvc-to-react - Atom

untitled | untitled | untitled | untitled

```
1  import { Store } from "flummox";
2
3  class MessageStore extends Store {
4    constructor(flux) {
5      super();
6
7      const actions = flux.getActions("messages");
8
9      this.register(actions.newMessage, this.handleNewMessage);
10
11     this.messageCounter = 0;
12     this.state = {};
13   }
14
15   handleNewMessage(content) {
16     const id = this.messageCounter++;
17
18     this.setState({
19       [id]: {
20         content,
21         id,
22       },
23     });
24   }
25 }
```

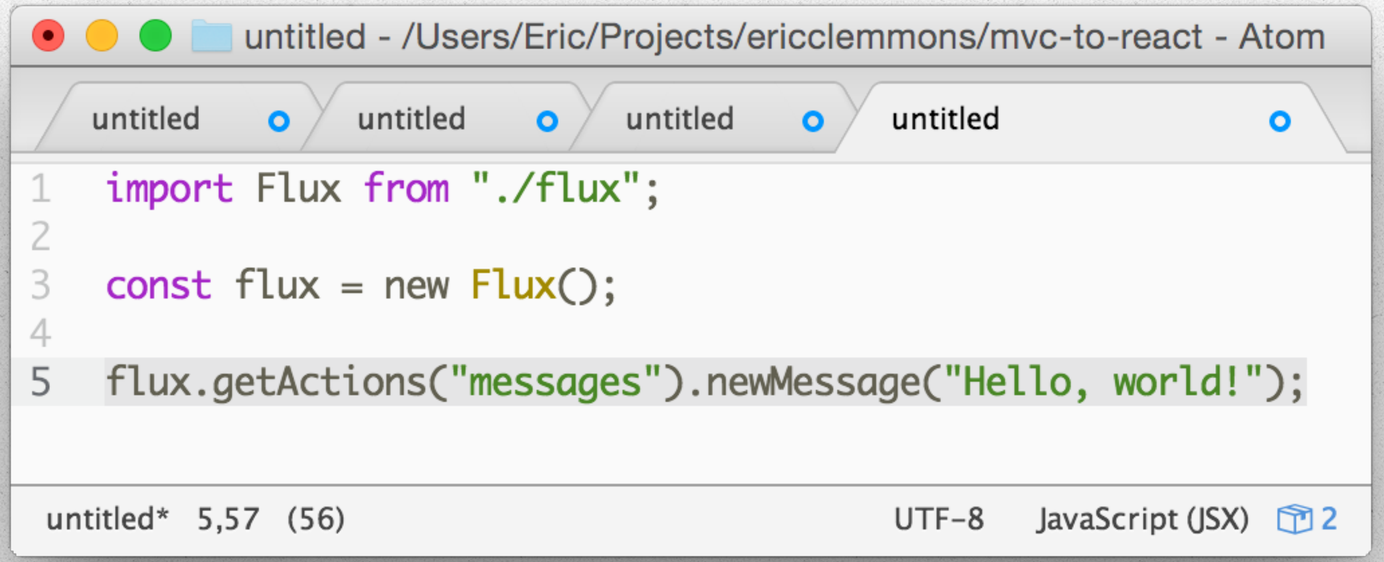
App Flux

Flummox



Usage

Flummox



Separation of Concerns

> “The value of separation of concerns is simplifying development and maintenance of [components]. When concerns are well-separated, individual sections can be reused, as well as developed and updated independently.”

Complex Container

Separation of Concerns

untitled - /Users/Eric/Projects/ericclemmons/mvc-to-react - Atom

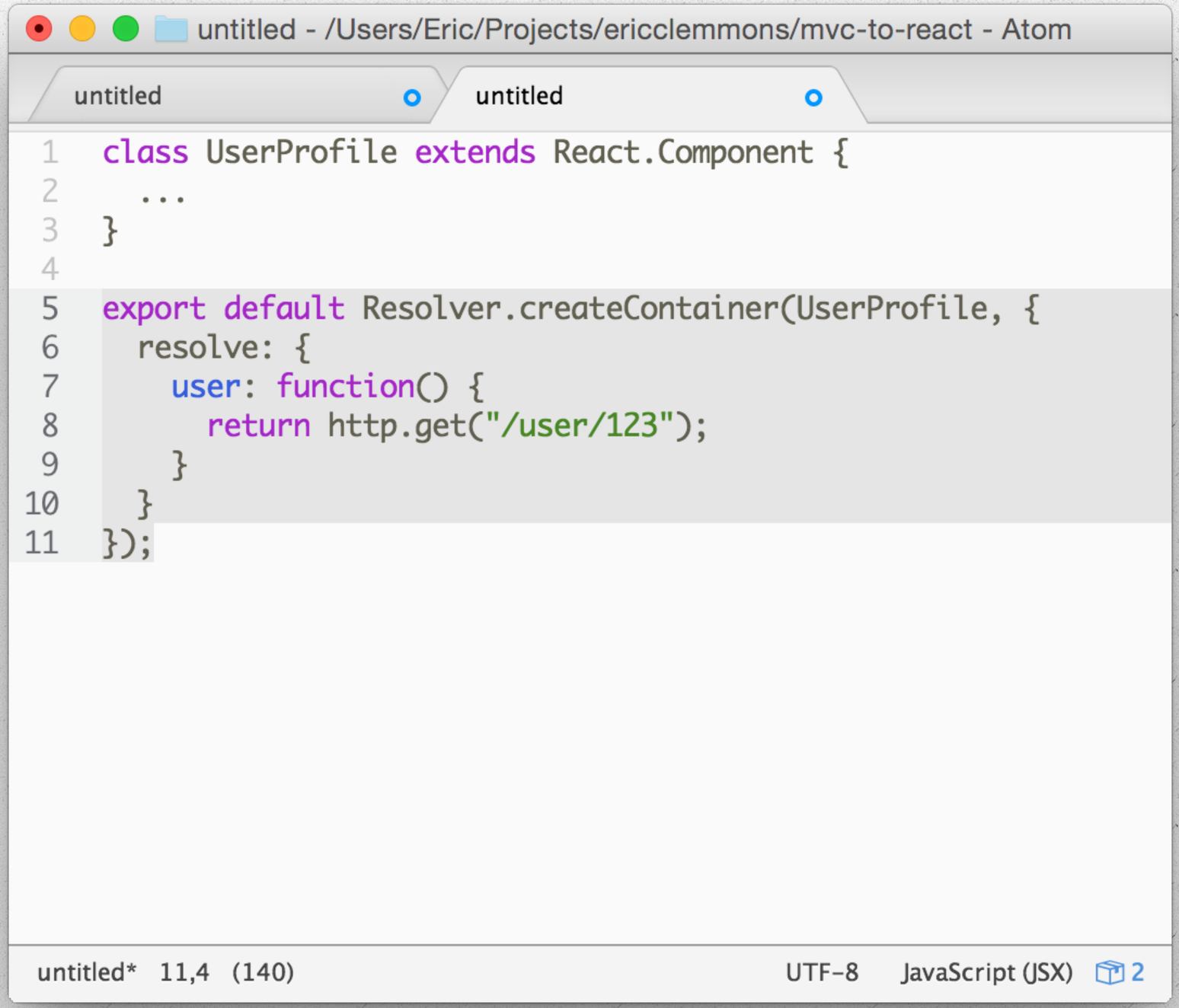
untitled | untitled

```
1  class UserProfileContainer extends React.Component {
2    constructor() {
3      this.state = { user: null };
4    }
5
6    componentWillMount() {
7      http.get("/user/123").then((response) => {
8        this.setState({ user: response });
9      });
10   }
11
12   render() {
13     if (!this.state.user) {
14       return false;
15     }
16
17     return <UserProfile user={this.state.user} />
18   }
19 }
20
```

untitled* 18,1 (50) UTF-8 JavaScript (JSX) 2

Simple Container

Separation of Concerns



Releasing soon:
ericclemmons/react-resolver

Wrapping Up

Deliverables

- Isomorphic Development
- Separation of Concerns
- Rapid Development
- Performance
- Clearer Code
- Thriving Ecosystem

Learning

http://facebook.github.io/react/
http://conf.reactjs.com/schedule.html
https://www.tildedave.com/2014/11/15/introduction-to-contexts-in-react-js.html
https://github.com/ericclemmons/mvc-to-react

Getting Started

https://github.com/RickWong/react-isomorphic-starterkit
https://github.com/rackt/react-router/

Other Environments

https://github.com/reactjs/react-rails
https://github.com/reactjs/React.NET
https://github.com/reactjs/react-python
https://github.com/reactjs/react-php-v8js

Flux Implementations

http://goatslacker.github.io/alt/
https://github.com/acdlite/flummox/
https://facebook.github.io/flux/
http://fluxible.io/
http://martyjs.org/
https://github.com/spoike/refluxjs/

Community

http://reactiflux.com/
https://reactiflux.herokuapp.com/
https://join-reactiflux.herokuapp.com/

Slack

Reactiflux

STARRED
alt
flummox
general
needhelp
reflux
houston-reactifluxers

CHANNELS
cool-links
jobs
random
react-router
relay
webpack
+17 more...

DIRECT MESSAGES
slackbot
aclark

ericclemmons
online

#general

Search

react, and rails, enabling the es6 and jsx transpilers, and node integration.

1255

hstove 8:16 PM
joined #general

buck 8:55 PM
@justin_gordon: i've derived our work off of your react-webpack-rails tutorial (i'm bdefore on github) ... thanks for that

justin_gordon 8:56 PM
anything I can grab from what you've done?

or care to make a PR?

buck 8:56 PM
for our implementation i found the alt flux implementaiton by @josh_perez to be the easiest to onboard our engineers

i have been hoping to find the time to make a PR for you but i've been unsuccessful

https://github.com/goatslacker/alt

GitHub
goatslacker/alt
alt - Isomorphic Flux

Thank You!

https://github.com/ericclemmons/

https://twitter.com/ericclemmons/