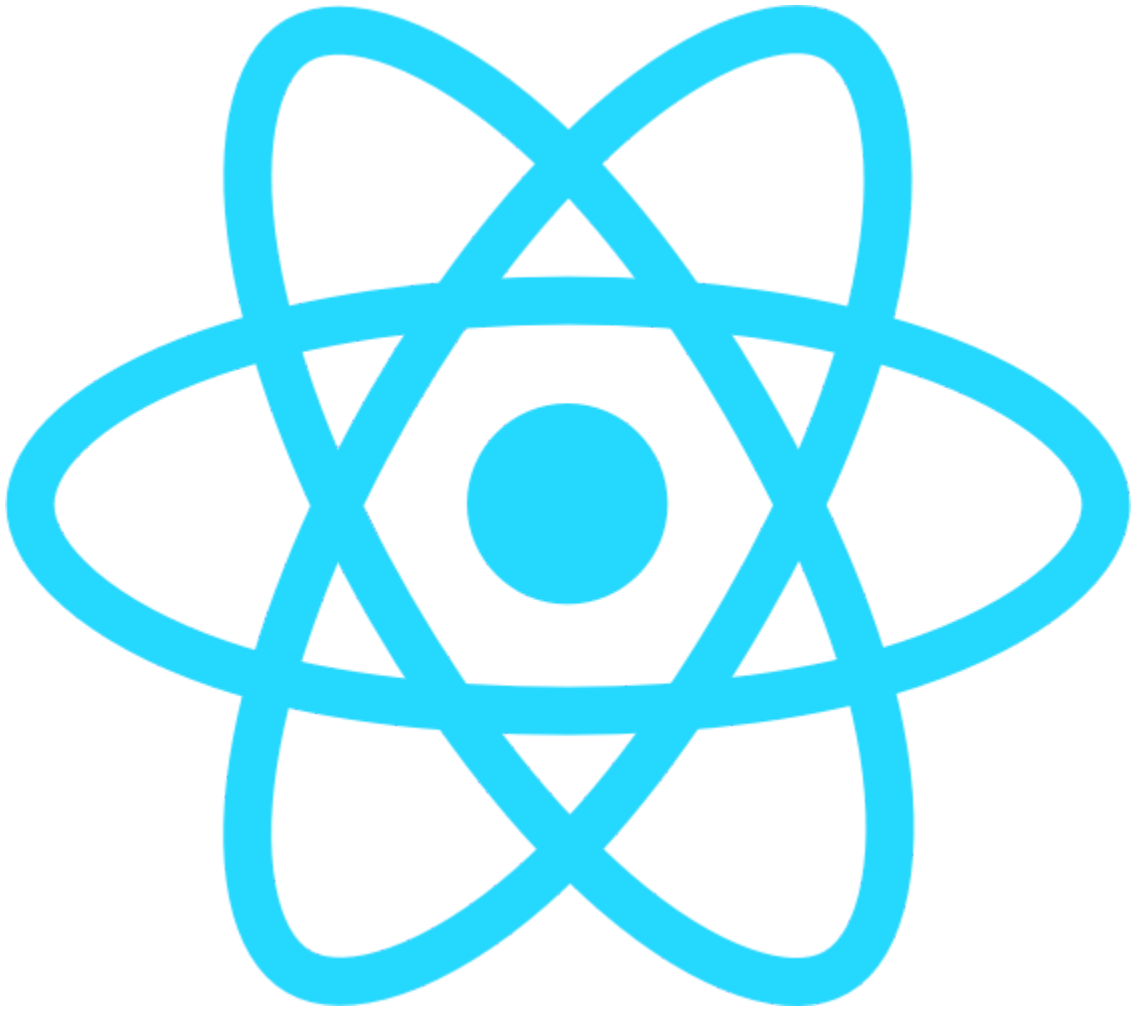


REACT

A JAVASCRIPT LIBRARY FOR BUILDING USER INTERFACES

WHAT IS REACT?

- A javascript library for rendering highly performant UIs in the browser
- Claims to be the View in MVC
- React was built by the instagram team, shared with facebook and open sourced in May 2013
- Derided when first open sourced, but is now considered a valuable rethink of SPA architecture

SPA - SINGLE PAGE APPLICATION

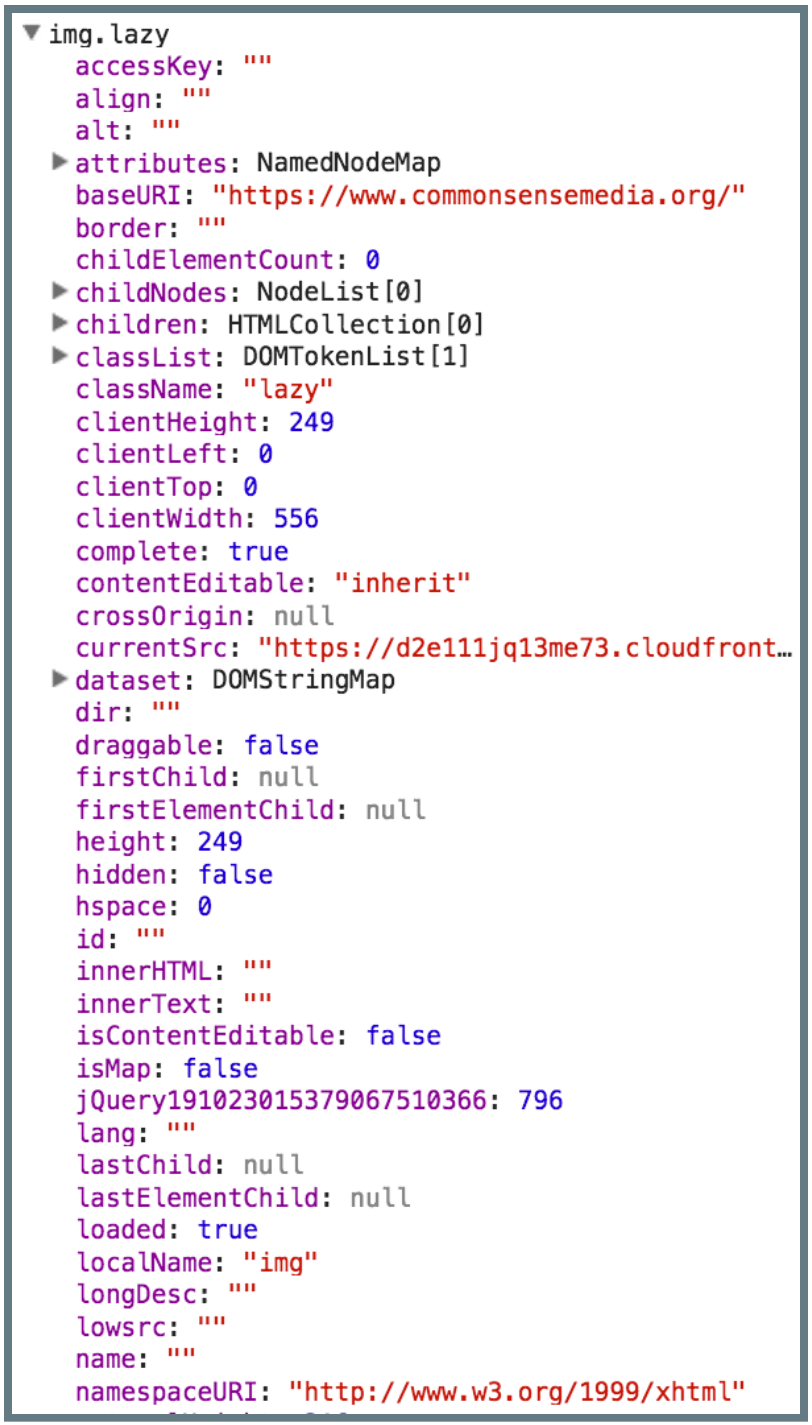
- All HTML, CSS and Javscript is loaded once
- Updates to the UI are handled by javascript
- Fetching new data is handled by AJAX or WebSockets, ***NOT*** by reloading the entire page like traditional client / server web applications
- Provides a faster feedback loop for the user and better User Experience
- Popular SPA frameworks include Backbone, Ember, Angular and countless others

DOM IS SLOW

Accessing the DOM, adding and removing HTML elements, and changing HTML element styles and properties is very slow.

React was built to address this one major bottleneck in browser performance.

DOM IS SLOW


▼ img.lazy
accessKey: ""
align: ""
alt: ""
▶ attributes: NamedNodeMap
baseURI: "https://www.commonsensemedia.org/"
border: ""
childElementCount: 0
▶ childNodes: NodeList[0]
▶ children: HTMLCollection[0]
▶ classList: DOMTokenList[1]
className: "lazy"
clientHeight: 249
clientLeft: 0
clientTop: 0
clientWidth: 556
complete: true
contentEditable: "inherit"
crossOrigin: null
currentSrc: "https://d2e111jq13me73.cloudfront…
▶ dataset: DOMStringMap
dir: ""
draggable: false
firstChild: null
firstElementChild: null
height: 249
hidden: false
hspace: 0
id: ""
innerHTML: ""
innerText: ""
isContentEditable: false
isMap: false
jQuery191023015379067510366: 796
lang: ""
lastChild: null
lastElementChild: null
loaded: true
localName: "img"
longDesc: ""
lowsrc: ""
name: ""
namespaceURI: "http://www.w3.org/1999/xhtml"

DOM IS SLOW - REFLOW

Reflow happens when the browser engine builds the layout geometry of HTML elements based on style properties.
Reflow occurs:

- on page load
- on browser window resize
- when layout information retrieved
- when layout styles are applied
- when DOM nodes are added or removed

Reflow cascades down the DOM tree, so changes to the body element cause the browser to recalculate the geometry *for every element on the page*

DOM IS SLOW - PAINT

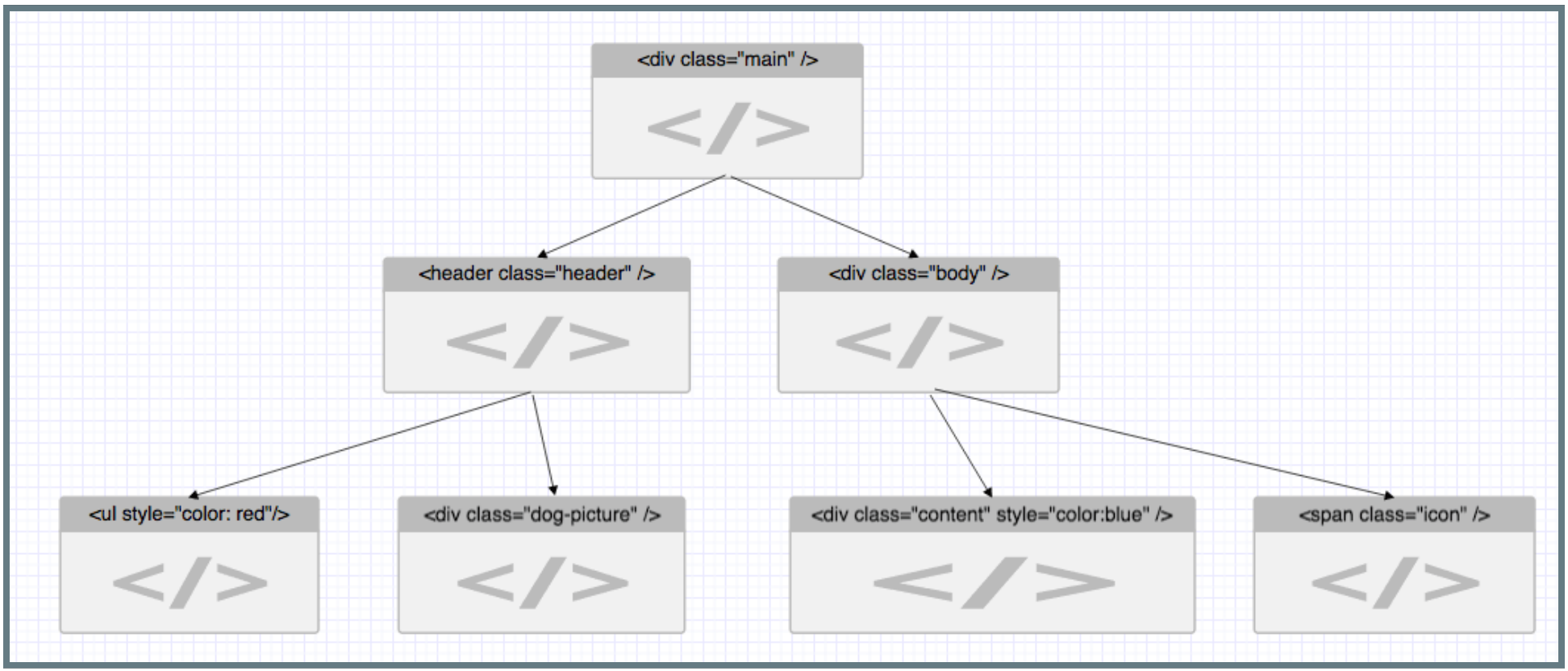
Paint occurs when the browser draws calculated styles to the screen.

- elements are painted to the screen
- images are decoded and resized if they do not match their element width / height
- CSS styles are calculated and painted to screen
- box-shadow, gradients and opacity are expensive styles to paint
- the browser window is painted frame by frame during scrolling

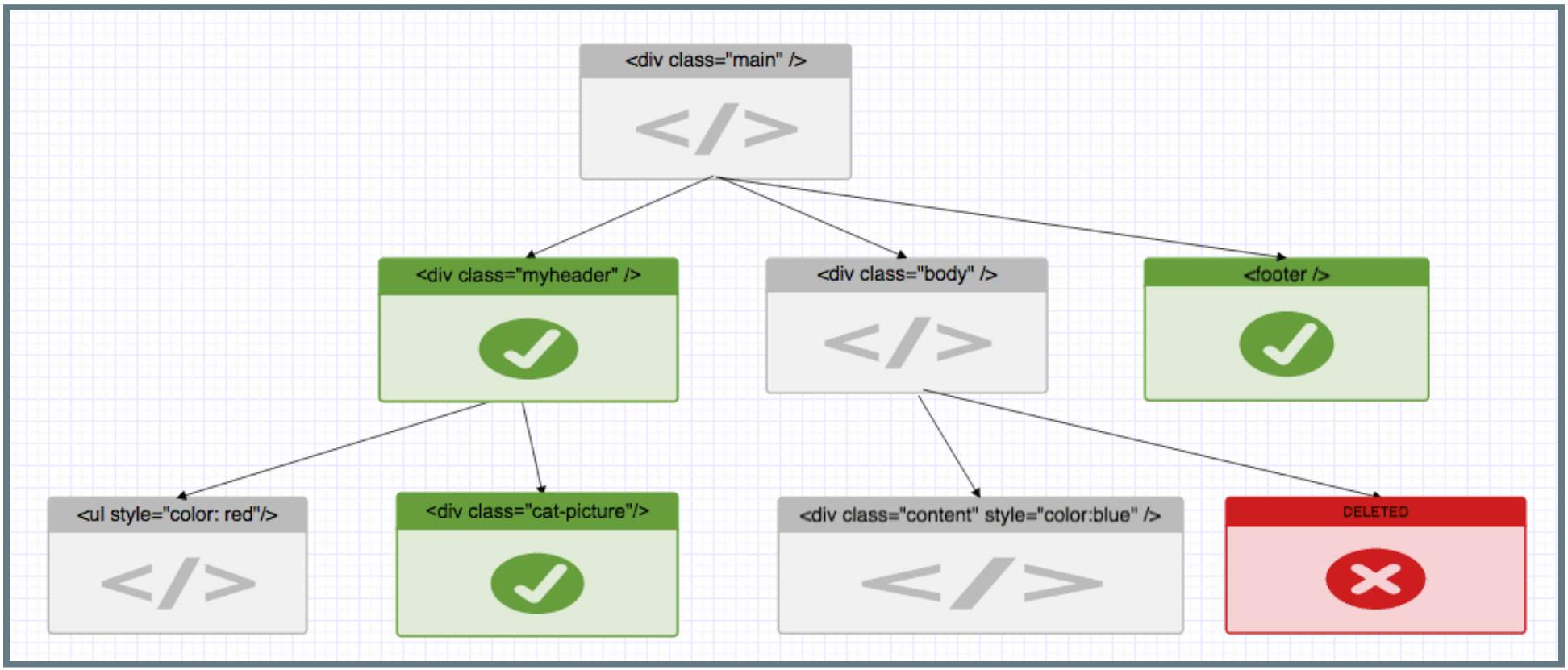
REACT - VIRTUAL DOM

- React minimizes Reflows and Paints
- React creates a "Virtual DOM" representation of the actual DOM in javascript
- When a change is made, a diff is made between the actual DOM and the virtual DOM to find all changes to the HTML elements and properties
- This diff between the virtual DOM is then reconclied with the actual DOM and the minimum amount of updates are made to the actual DOM, all in one pass

REACT - ACTUAL DOM

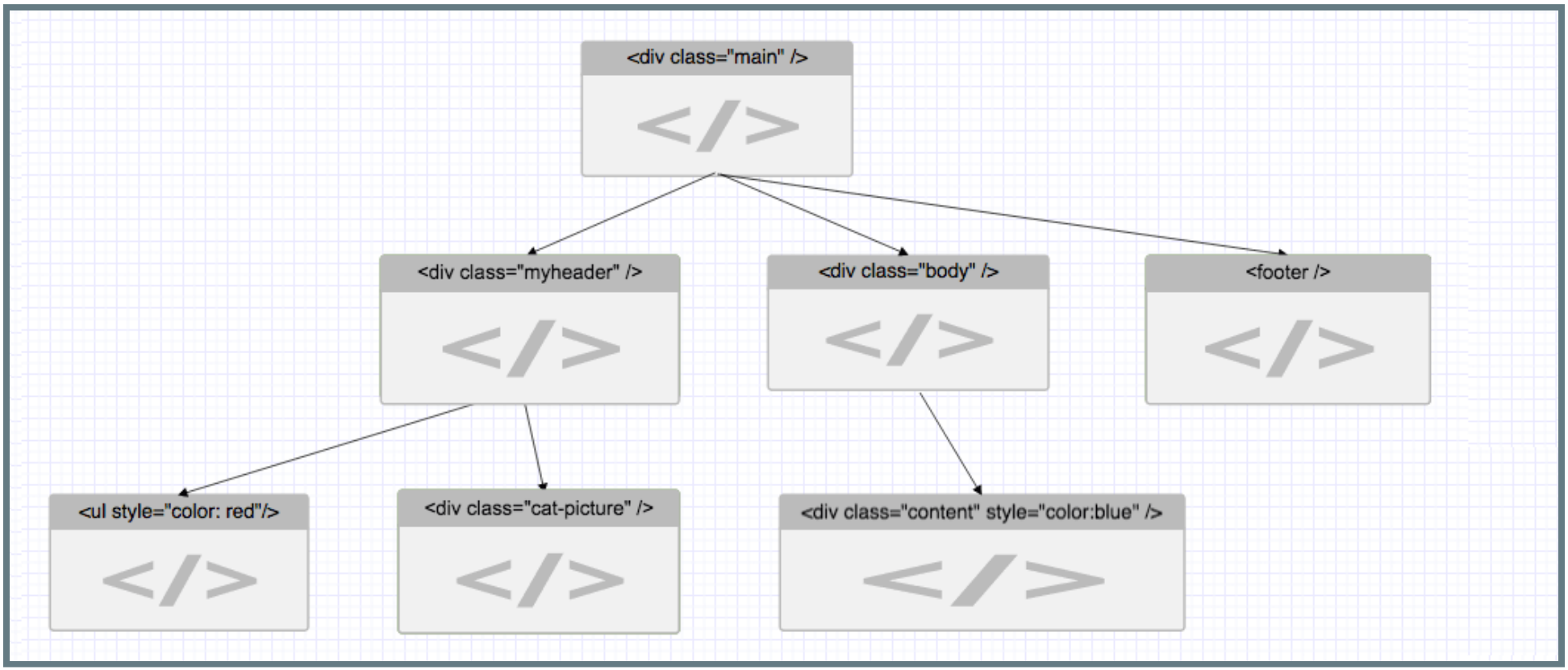


REACT - DIFF



UI updates are made. React creates a DIFF between actual DOM and Virtual DOM

REACT - UPDATED DOM



DIFF applied to Actual DOM in one pass, minimizing Reflow and Paint.

DOM IS SLOW - REACT IS FAST

Ryan Florence compares Ember, Angular and React performance.

REACT CODE EXAMPLE 1

HTML

```
<!doctype html>
<html>
<body>
  <script type="text/javascript" src="node_modules/react/dist/react.js"></script>
  <script type="text/javascript" src="1.js"></script>
</body>
</html>

```

JSX

```
var MyFirstComponent = React.createClass({
  render: function() {
    return (
      <div>Hello {this.props.name}</div>
    )
  }
});

React.render(<MyFirstComponent name="Egg" />, document.body);

```

JSX

- JSX is a representation of HTML in javascript
- Simplifies component design
- Ubiquitous - difficult to find examples not using JSX
- Requires a build step. Options:
  - jsx command line tool
  - gulp watch task using browserify
  - webpack with Babel for ES6

REACT CODE EXAMPLE 2

Composability

```
1   var Box = React.createClass({
2     render: function() {
3       return (
4         <div className="box">
5           {Math.random()}
6         </div>
7       )
8     }
9   });
10
11  var App = React.createClass({
12    render: function() {
13      return (
14        <div>
15          <h1>{this.props.exampleNumber}. My React App - {this.props.subtitle}</h1>
16          <Box />
17          <Box />
18          <Box />
19        </div>
20      )
21    }
22  });
23
24  React.render(<App exampleNumber="2" subtitle="Composability" />, document.body);
25
```

REACT CODE EXAMPLE 3

this.state && this.props

```
1  var Box = React.createClass({
2    getInitialState: function() {
3      return {
4        colors: ['red','yellow','blue','green','teal','grey'],
5        number: Math.round(Math.random() * 5),
6      }
7    },
8    render: function() {
9      return (
10       <div className="box" style={{backgroundColor: this.state.colors[this.state.number]}}>
11         {this.state.number}
12       </div>
13     )
14   }
15 });
16
```

REACT CODE EXAMPLE 4

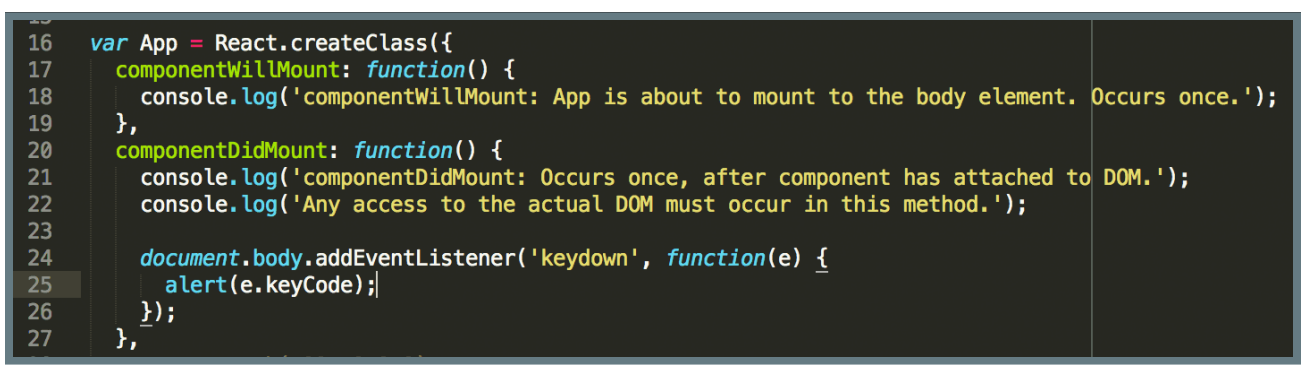
Private Functions

```
16  var App = React.createClass({
17    // return rgb(123,12,342);
18    _createColor: function() {
19      return 'rgb(' + Math.round(Math.random() * 255) + ',' + Math.round(Math.random() * 255) + ',' + Math.round(Math.random() * 255) + ')';
20    },
21    // return a random number of Box components with random bgcolors
22    _createBoxes: function() {
23      var boxes = []
24      for (var i = 0; i < Math.round(Math.random() * 20); i++) {
25        boxes.push(this._createColor())
26      }
27
28      return boxes.map(function(color) {
29        return <Box bgcolor={color} />;
30      });
31    },
32    render: function() {
33      return (
34        <div>
35          <h1>{this.props.exampleNumber}. My React App - {this.props.subtitle}</h1>
36          {this._createBoxes()}
37        </div>
38      )
39    }
40  });
41
42  React.render(<App exampleNumber="4" subtitle="Private functions" />, document.body);
43
```

REACT CODE EXAMPLE 5

Lifecycle methods

- You can hook into various points of the React render cycle with lifecycle hooks.
- componentWillMount
- componentDidMount



```
16  var App = React.createClass({
17    componentWillMount: function() {
18      console.log('componentWillMount: App is about to mount to the body element. Occurs once.');
19    },
20    componentDidMount: function() {
21      console.log('componentDidMount: Occurs once, after component has attached to DOM.');
22      console.log('Any access to the actual DOM must occur in this method.');
23
24      document.body.addEventListener('keydown', function(e) {
25        alert(e.keyCode);
26      });
27    },
```

REACT CODE EXAMPLE 6

Lifecycle methods

- You can hook into various points of the React render cycle with lifecycle hooks.
- componentWillReceiveProps
- shouldComponentUpdate

```
28    componentWillReceiveProps: function(nextProps) {
29      console.log('componentWillReceiveProps: Occurs any time properties or props are
          passed into this component.');
30    },
31    shouldComponentUpdate: function(nextProps, nextState) {
32      console.log('shouldComponentUpdate: return false to cancel rendering component.
          Very useful for performance optimization of many small components.')
33    },
```

REACT CODE EXAMPLE 7

Lifecycle methods

- componentWillUpdate
- componentDidUpdate
- componentWillUnmount

```
34    componentWillUpdate: function() {
35      console.log('componentWillUpdate: component is about to render to the DOM.');
36    },
37    componentDidUpdate: function() {
38      console.log('componentDidUpdate: fired immediately after component has rendered to
          the DOM.');
39    },
40    componentWillUnmount: function() {
41      console.log('componentWillUnmount: component will detach from the DOM. Cleanup any
          timers, event listeners or other variables in this hook.');
42
43      document.body.removeEventListener('keydown', function(e) {
44        console.log(e.keyCode);
45      });
46    },
```

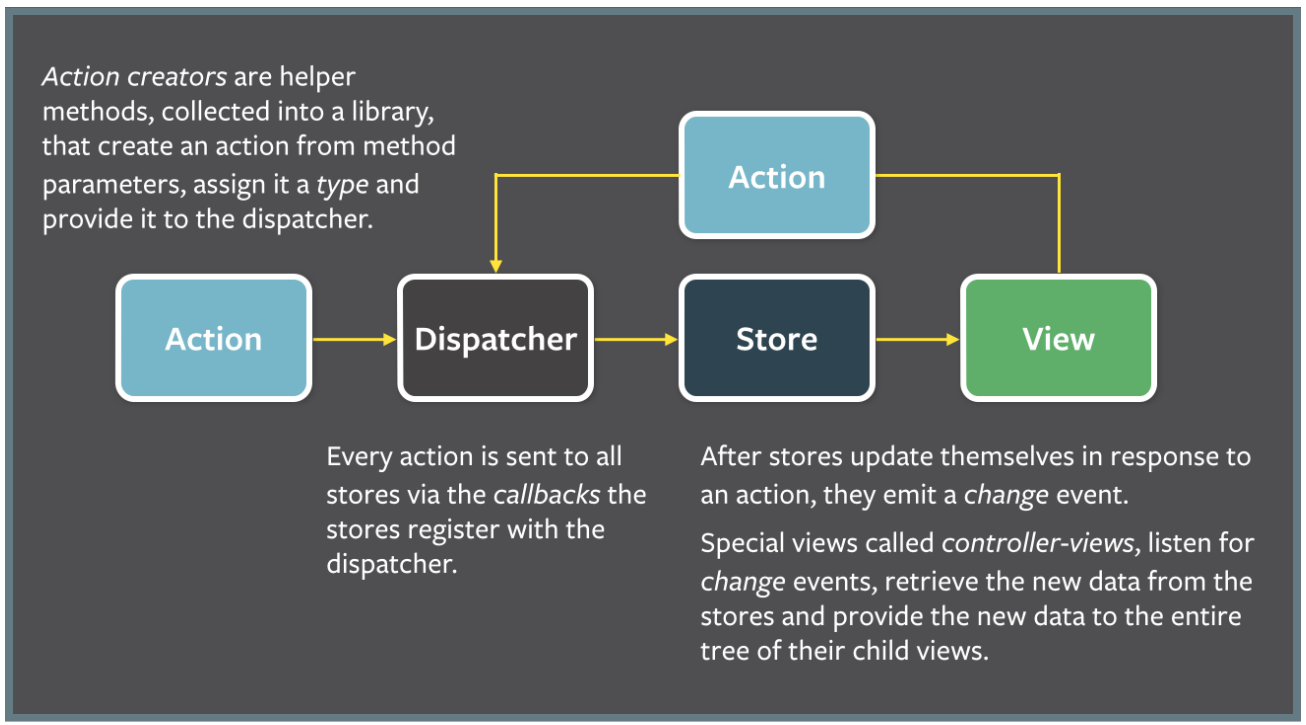
REACT CODE EXAMPLE 8

CSS

- className
- backgroundColor

FLUX

- one way data flow
- handling data updates



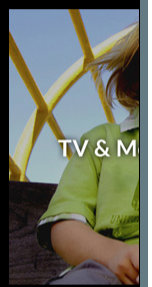
COMCAST APP

- app.jsx
- componentDidMount
- flux
- browserify
- shouldComponentUpdate example in product row

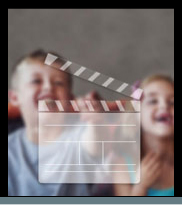
common sense media

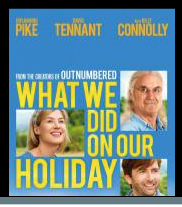
We rate, educate, and advocate for kids, families, and schools

ages 2-4 5-7 8-9 10-12 13+



New Reviews

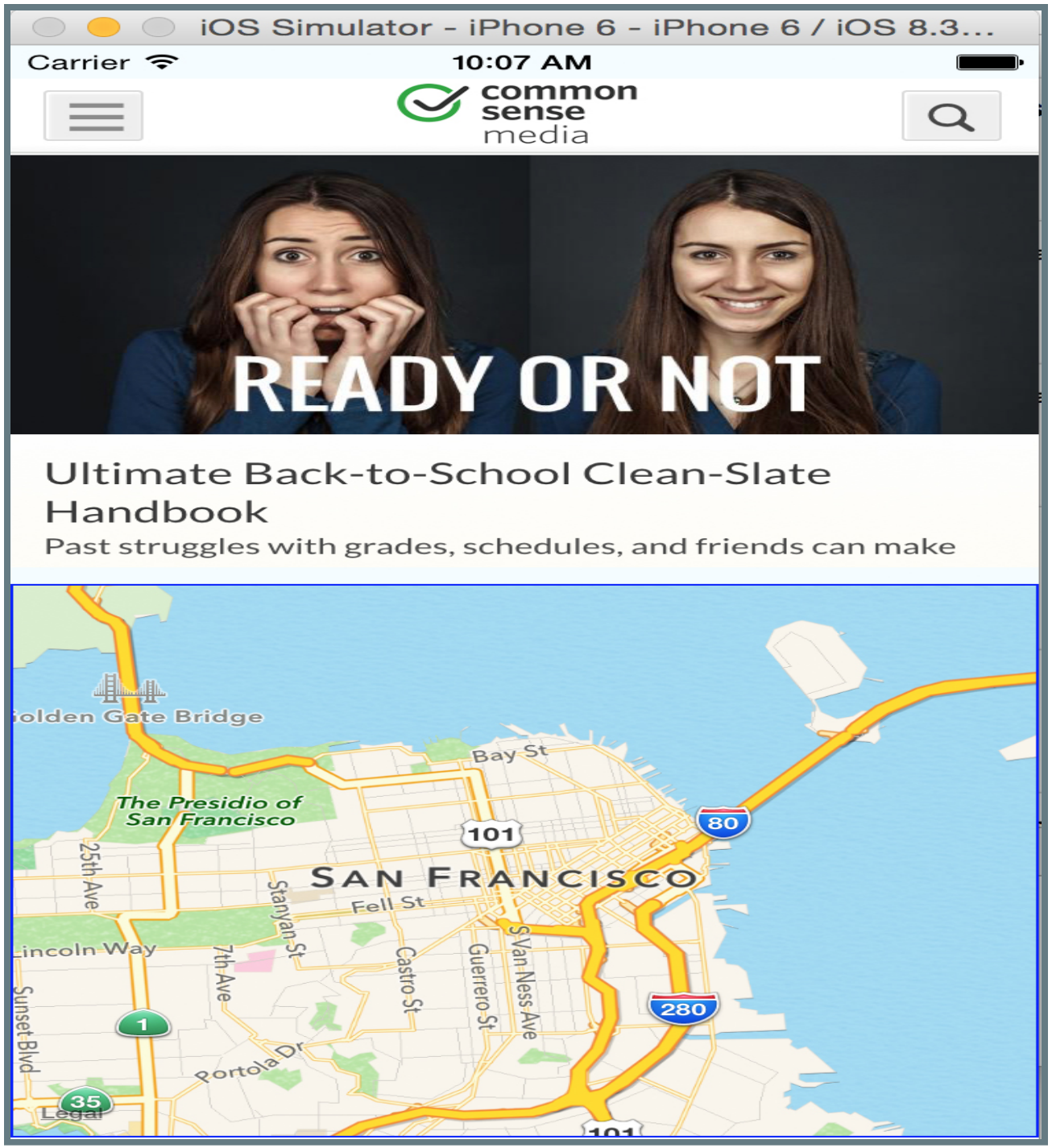




REACT NATIVE

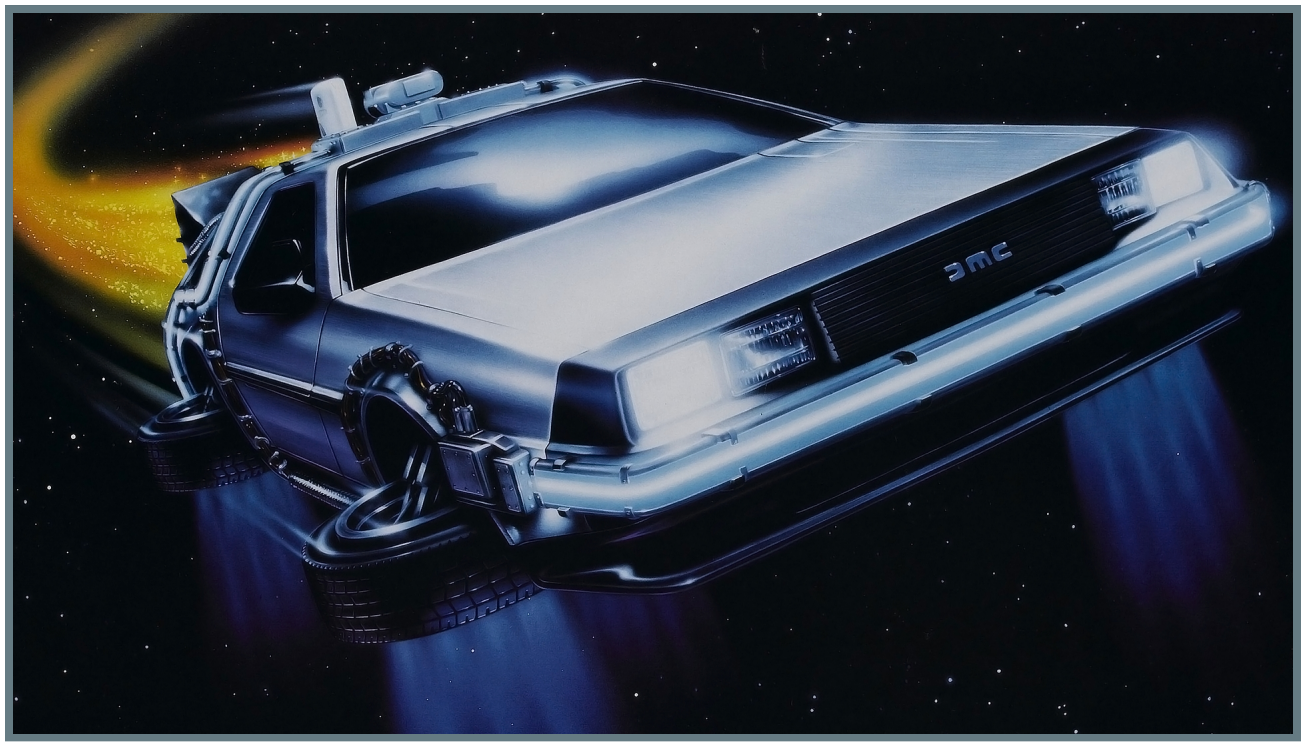
Native mobile app performance, written in javascript

- share code between web pages and native apps
- use css flexbox for native layout
- early days - iOS support only, Android coming in about a month


iOS Simulator - iPhone 6 - iPhone 6 / iOS 8.3...
Carrier
10:07 AM
common sense media
READY OR NOT
Ultimate Back-to-School Clean-Slate Handbook
Past struggles with grades, schedules, and friends can make
Golden Gate Bridge
The Presidio of San Francisco
Bay St
101
80
SAN FRANCISCO
Fell St
25th Ave
Stanyan St
Lincoln Way
7th Ave
Castro St
Guerrero St
S Van Ness Ave
280
Sunset Blvd
1
Portola Dr
35
Legal
101

FUTURE OF REACT

- Docs will use ES6 syntax, Babel transpilation encouraged
- Immutable data structures
- React Native for Android
- Flux library will mature
- Server side rendering


DMC

RESOURCES

- React docs
- Flux docs
- React Native docs
- Scotch.io tutorial
- Egghead.io React tutorial videos

DISCUSSION