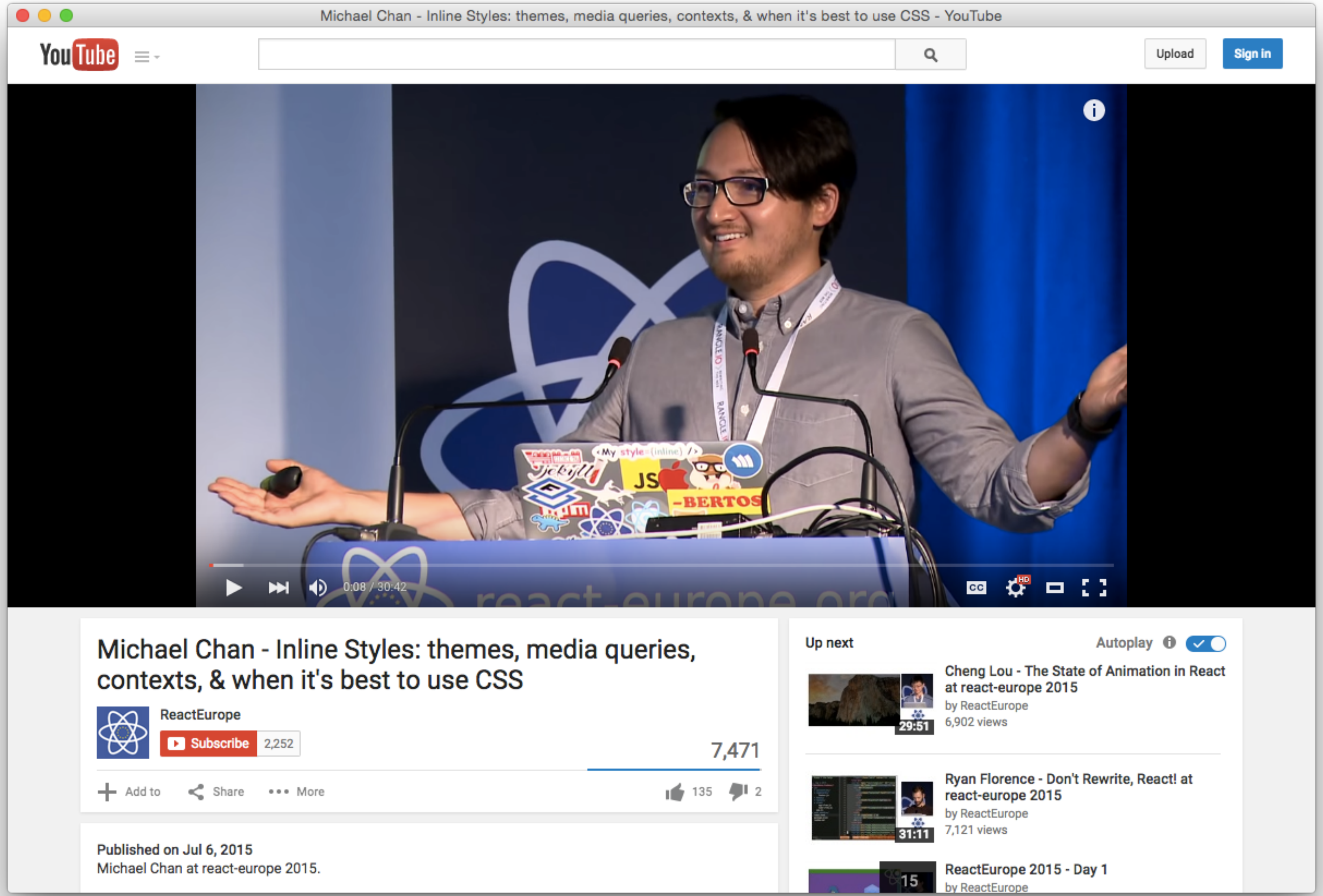
# Inline Styles and React

潘睿 //@plrthink

modao.cc

# Great thanks to


Michael Chan - Inline Styles: themes, media queries, contexts, & when it's best to use CSS - YouTube
Upload
Sign in
0:08 / 30:42
Up next
Autoplay
Add to
Share
More
135
2

Michael Chan - Inline Styles: themes, media queries, contexts, & when it's best to use CSS

ReactEurope

Subscribe 2,252

7,471

Published on Jul 6, 2015
Michael Chan at react-europe 2015.

Cheng Lou - The State of Animation in React at react-europe 2015
by ReactEurope
6,902 views
29:51

Ryan Florence - Don't Rewrite, React! at react-europe 2015
by ReactEurope
7,121 views
31:11

ReactEurope 2015 - Day 1
by ReactEurope

https://www.youtube.com/watch?v=ERB1TJBn32c

## Inline Styles
and React

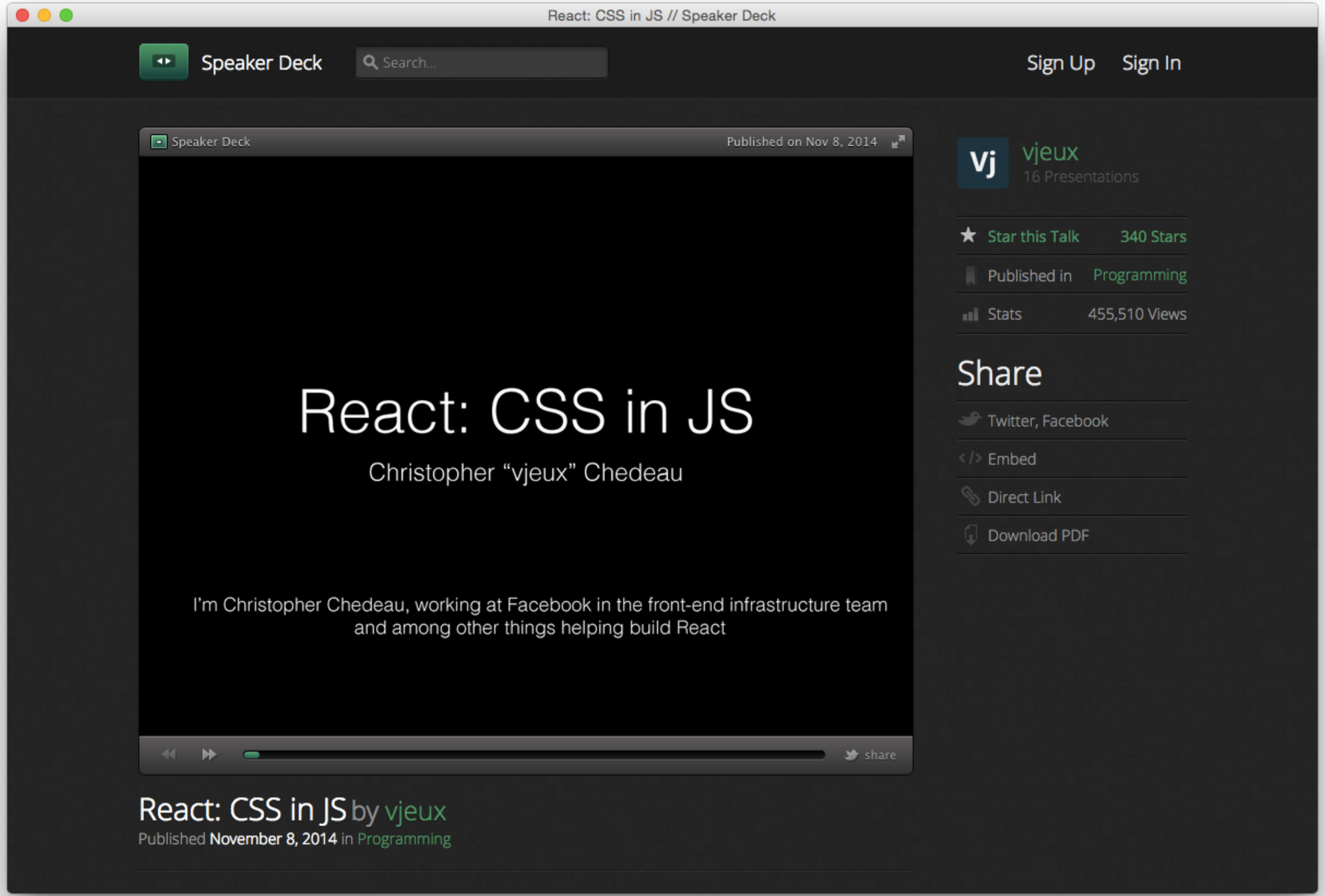
# NOT

```
<ul>
  <li style="color: #030303">one</li>
  <li style="color: #030303">two</li>
</ul>
```

## InlineStyles.js

```
var divStyle = {
  color: 'white',
  backgroundImage: 'url(' + imgUrl + ')',
  WebkitTransition: 'all', // note the capital 'W' here
  msTransition: 'all' // 'ms' is the only lowercase vendor prefix
};

React.render(<div style={divStyle}>Hello World!</div>, mountNode);
```

https://facebook.github.io/react/tips/inline-styles.html


React: CSS in JS // Speaker Deck

Speaker Deck Search... Sign Up Sign In

Speaker Deck Published on Nov 8, 2014

React: CSS in JS

Christopher “vjeux” Chedeau

I'm Christopher Chedeau, working at Facebook in the front-end infrastructure team and among other things helping build React

share

React: CSS in JS by vjeux
Published November 8, 2014 in Programming

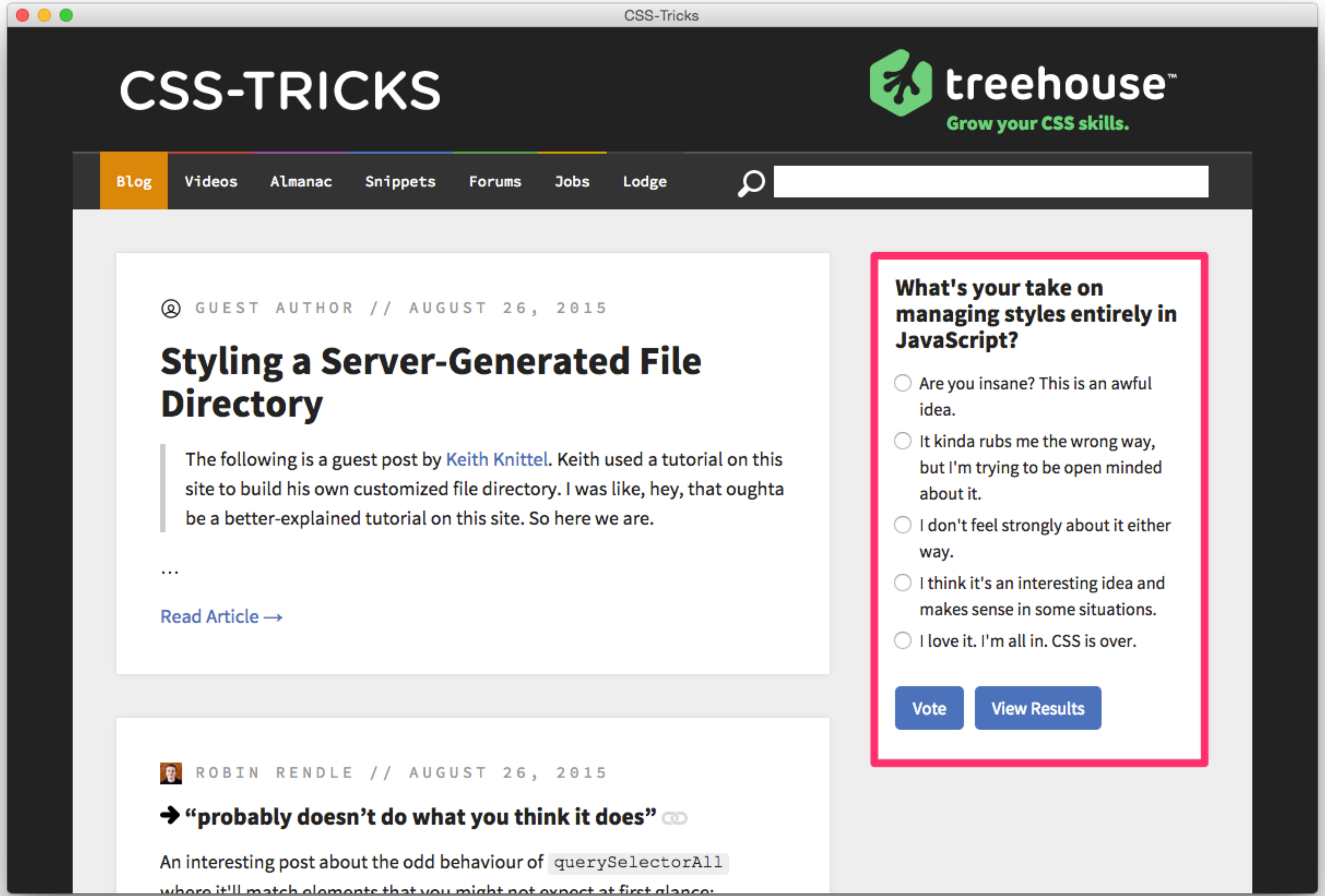
vjeux
16 Presentations

| | |
|---|---|
| Star this Talk | 340 Stars |
| Published in | Programming |
| Stats | 455,510 Views |

Share

- Twitter, Facebook
- Embed
- Direct Link
- Download PDF

https://speakerdeck.com/vjeux/react-css-in-js



GUEST AUTHOR // AUGUST 26, 2015

Styling a Server-Generated File Directory

The following is a guest post by Keith Knittel. Keith used a tutorial on this site to build his own customized file directory. I was like, hey, that oughta be a better-explained tutorial on this site. So here we are.

...

Read Article →

ROBIN RENDLE // AUGUST 26, 2015

→ "probably doesn't do what you think it does"

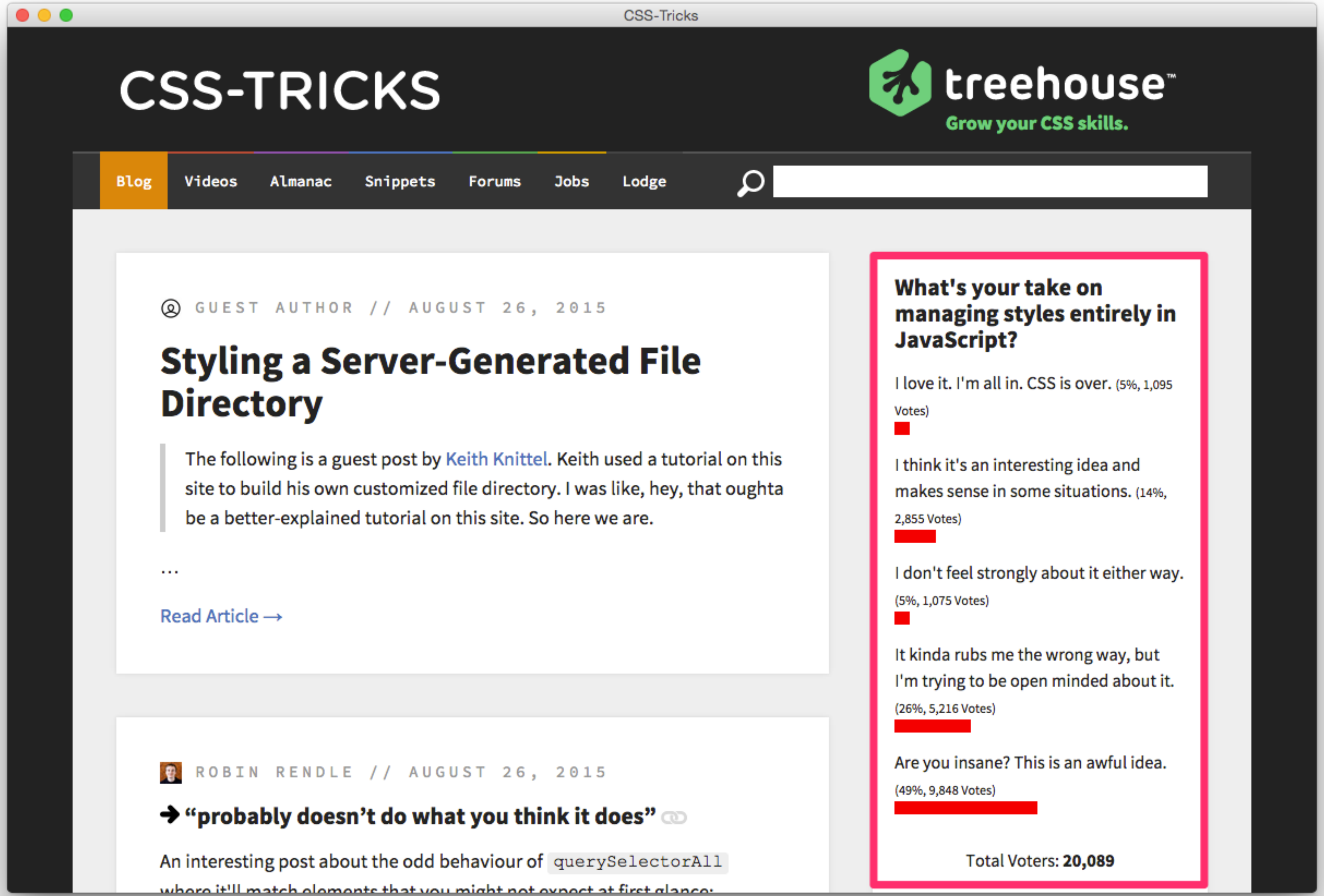
An interesting post about the odd behaviour of `querySelectorAll` where it'll match elements that you might not expect at first glance:

What's your take on managing styles entirely in JavaScript?

- ○ Are you insane? This is an awful idea.
- ○ It kinda rubs me the wrong way, but I'm trying to be open minded about it.
- ○ I don't feel strongly about it either way.
- ○ I think it's an interesting idea and makes sense in some situations.
- ○ I love it. I'm all in. CSS is over.

Vote | View Results

*survey on css-tricks.com*



GUEST AUTHOR // AUGUST 26, 2015

**Styling a Server-Generated File Directory**

> The following is a guest post by Keith Knittel. Keith used a tutorial on this site to build his own customized file directory. I was like, hey, that oughta be a better-explained tutorial on this site. So here we are.

…

Read Article →

ROBIN RENDLE // AUGUST 26, 2015

**➔ "probably doesn't do what you think it does"**

An interesting post about the odd behaviour of `querySelectorAll`

**What's your take on managing styles entirely in JavaScript?**

I love it. I'm all in. CSS is over. (5%, 1,095 Votes)

I think it's an interesting idea and makes sense in some situations. (14%, 2,855 Votes)

I don't feel strongly about it either way. (5%, 1,075 Votes)

It kinda rubs me the wrong way, but I'm trying to be open minded about it. (26%, 5,216 Votes)

Are you insane? This is an awful idea. (49%, 9,848 Votes)

Total Voters: **20,089**

*survey on css-tricks.com*

**josh**

# JUNE 26, 2015

Would Chris need to change the name of his site to "css.js tricks"?

“This totally breaks separation of concerns”

## BOUNDARIES ?

HTML

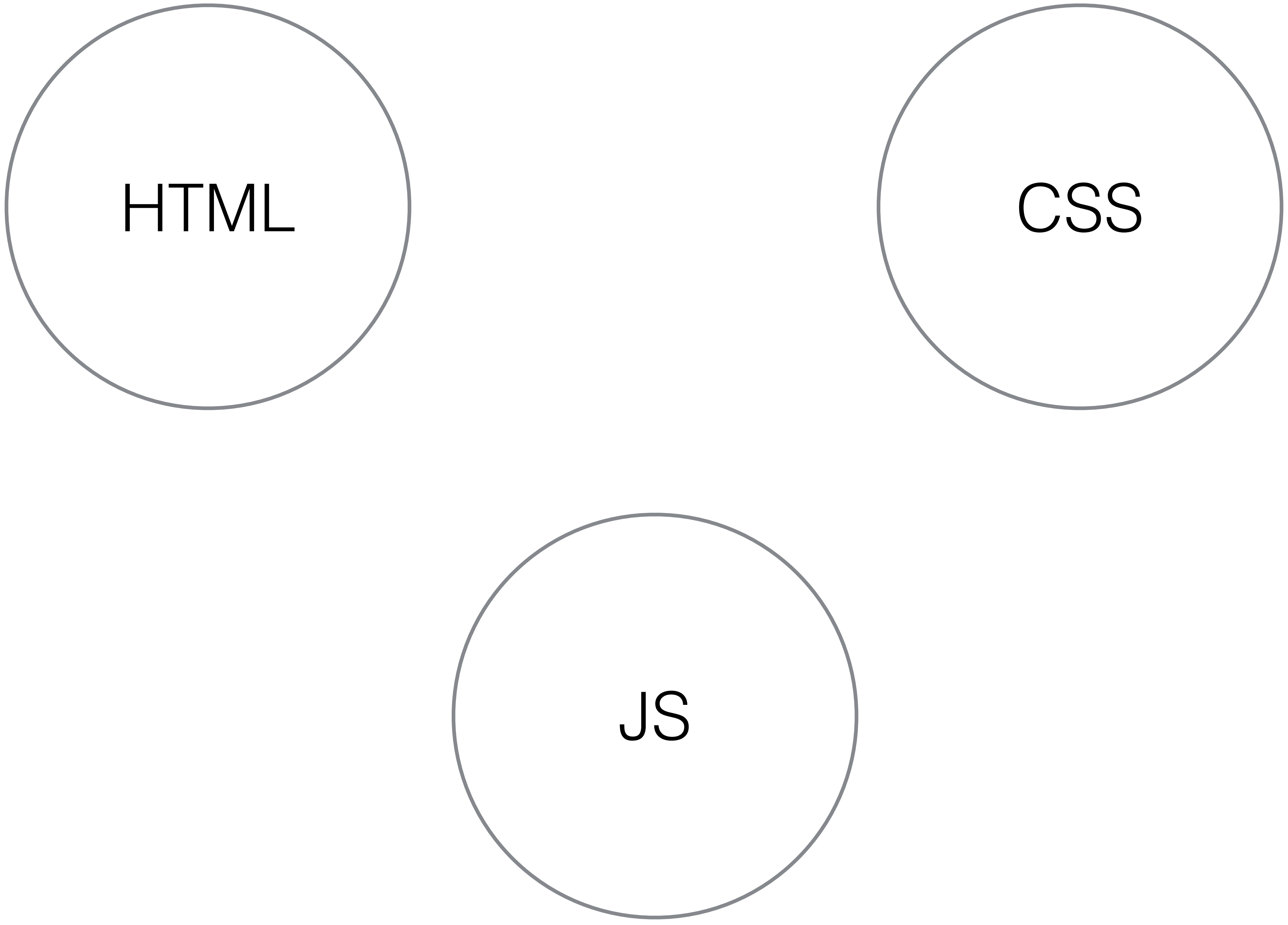
list.html

```
<ul>
  <li>one</li>
  <li>two</li>
</ul>
```

HTML
CSS

## list.sass

```
message {
  &--read {
    color: #D3D3D3;
  }
}
```


HTML
CSS
JS

## list.js

```
$('.message__read-mark').click(function(e) {
  $(this).closest('.message')
         .addClass('message--read')
})
```

. . .

## React

## Reactive

JSX
CSS

## Message.jsx

```
render () {
  classes = classnames({
    message: true,
    message--read: this.state.isRead,
  })
  return (
    <li className={classes}>
      this.props.title
    </li>
  )
}
```

## Message.jsx

```
render () {
  classes = classnames({
    message: true,
    message--read: this.state.isRead,
  })
  return (
    <li className={classes}>
      this.props.title
    </li>
  )
}
```

## Message.sass

```
message {
  &--read {
    color: #D3D3D3;
  }
}
```

## State class

“Style is not ‘CSS’.”

–Michael Chan

## Put styles in JS?

## Message.jsx

```
let styles = {
  message: { color: #030303, },
  isRead: { color: #D3D3D3, },
}
render () {
  return (
    <li style={Object.assign(
          styles.message,
          this.state.isRead && styles.isRead,
        )}
    >
      this.props.title
    </li>
  )
}
```

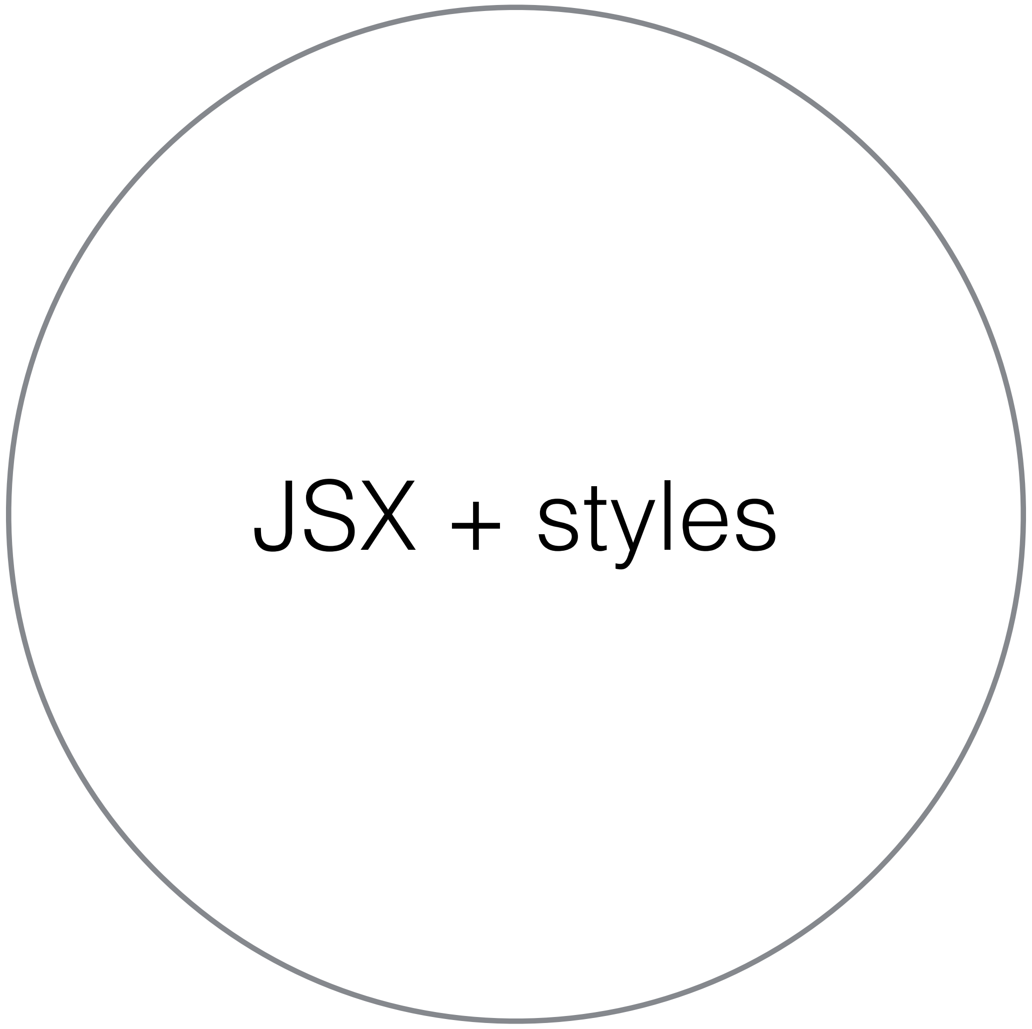
# Message.jsx

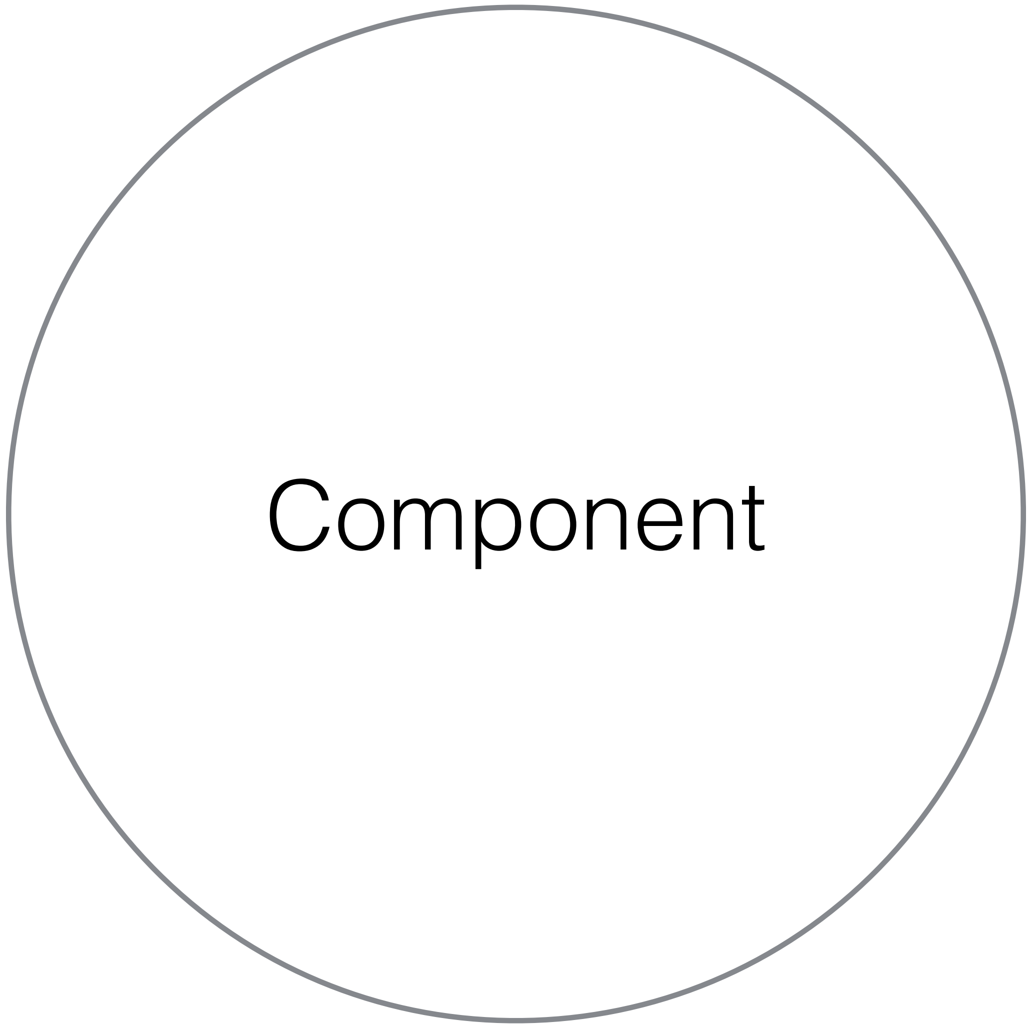
```
let styles = {
  message: { color: #030303, },
  isRead: { color: #D3D3D3, },
}
render () {
  return (
    <li style={Object.assign(
            styles.message,
            this.state.isRead && styles.isRead,
          )}
    >
      this.props.title
    </li>
  )
}
```

## DOM

```
<li style="color:#D3D3D3;" data-reactid="…">
  Hi
</li>
<li style="color:#030303;" data-reactid="…">
  w3ctech
</li>
```

~~State class~~


JSX + styles


Component

## Bonus

## Bonus

~~CSS global scope~~

## Global scope

- Styles are JS objects
- Directly apply styles

Bonus

## ~~Complex class name~~

Bonus

## Modularization

## List.jsx

```
// ...
import styleGlobals from './style/Globals'

styles = {
     color: styleGlobals.fooBlue,
     backgroundColor: styleGlobals.barRed,
}
// ...
```

Bonus

## Dead code elimination

## Maintainability

## Variables

# Functions

## Dynamic styles

ProgressBar.jsx

```
render () {
  styles = {
    // ...
    width: this.state.progrssValue,
    // ...
  }
  return (
    <span style={styles}
    >
   </span>
  )
}
```

Bonus

Shared on npm with styles

## Concerns

Concerns

## Theming

## Theming

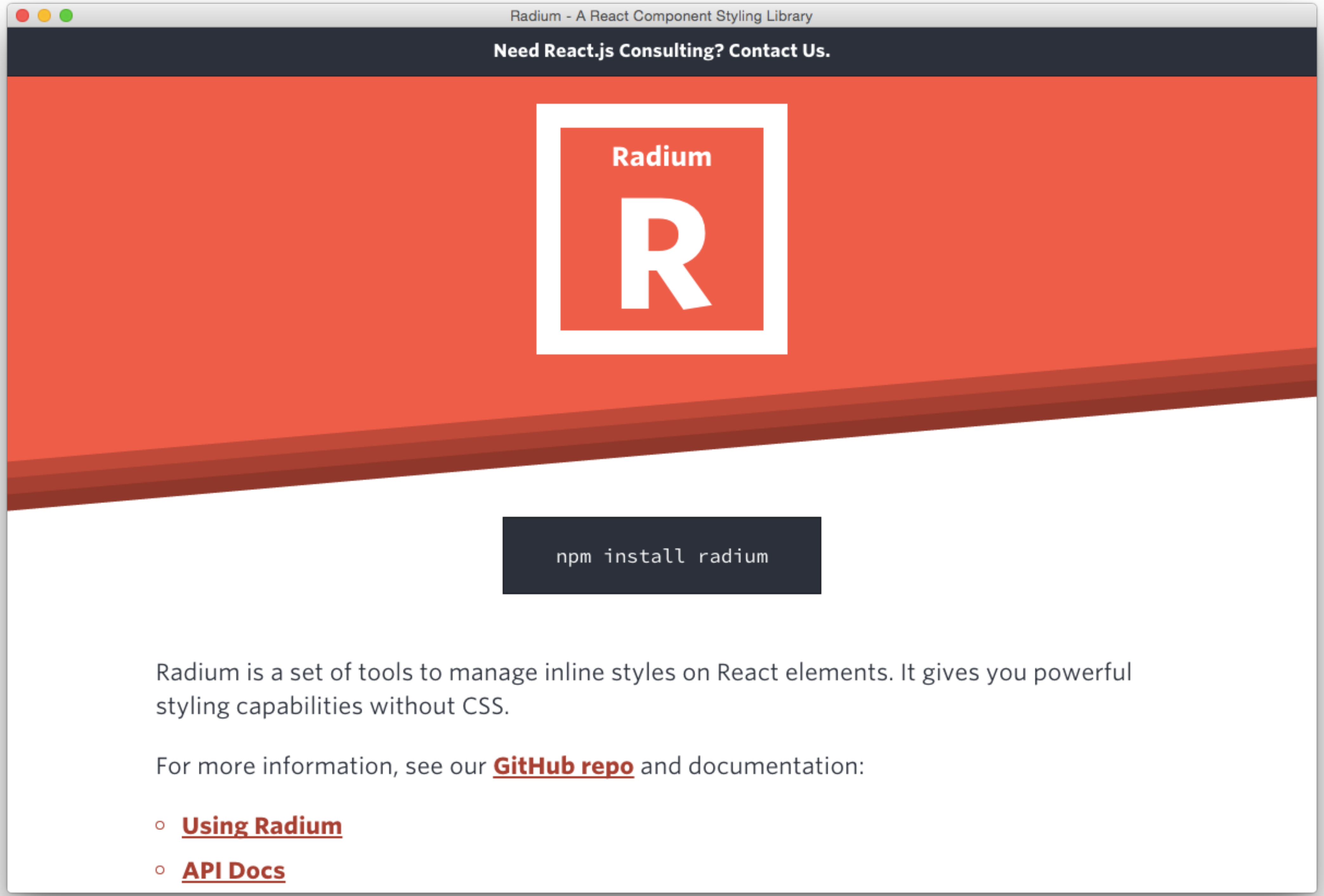
```
import dark from './themes/dark'
import light from './themes/light'
```

## Pseudo class

```
let styles = {
  message: {...},
  firstMessage: {...},
}

return (
  <ul className={messags}>
    {this.state.messages.map((message, i) =>{ return (
      <li style={Object.assign(
            styles.message,
            i === 0 && styles.firstMessage
       )}>
        {message.title}
      </li>
    ); })}
  </ul>
)
```

## Concerns

- :hover
- media queries
- vendor prefixes
- …



Radium - A React Component Styling Library

Need React.js Consulting? Contact Us.

```
npm install radium
```

Radium is a set of tools to manage inline styles on React elements. It gives you powerful styling capabilities without CSS.

For more information, see our GitHub repo and documentation:

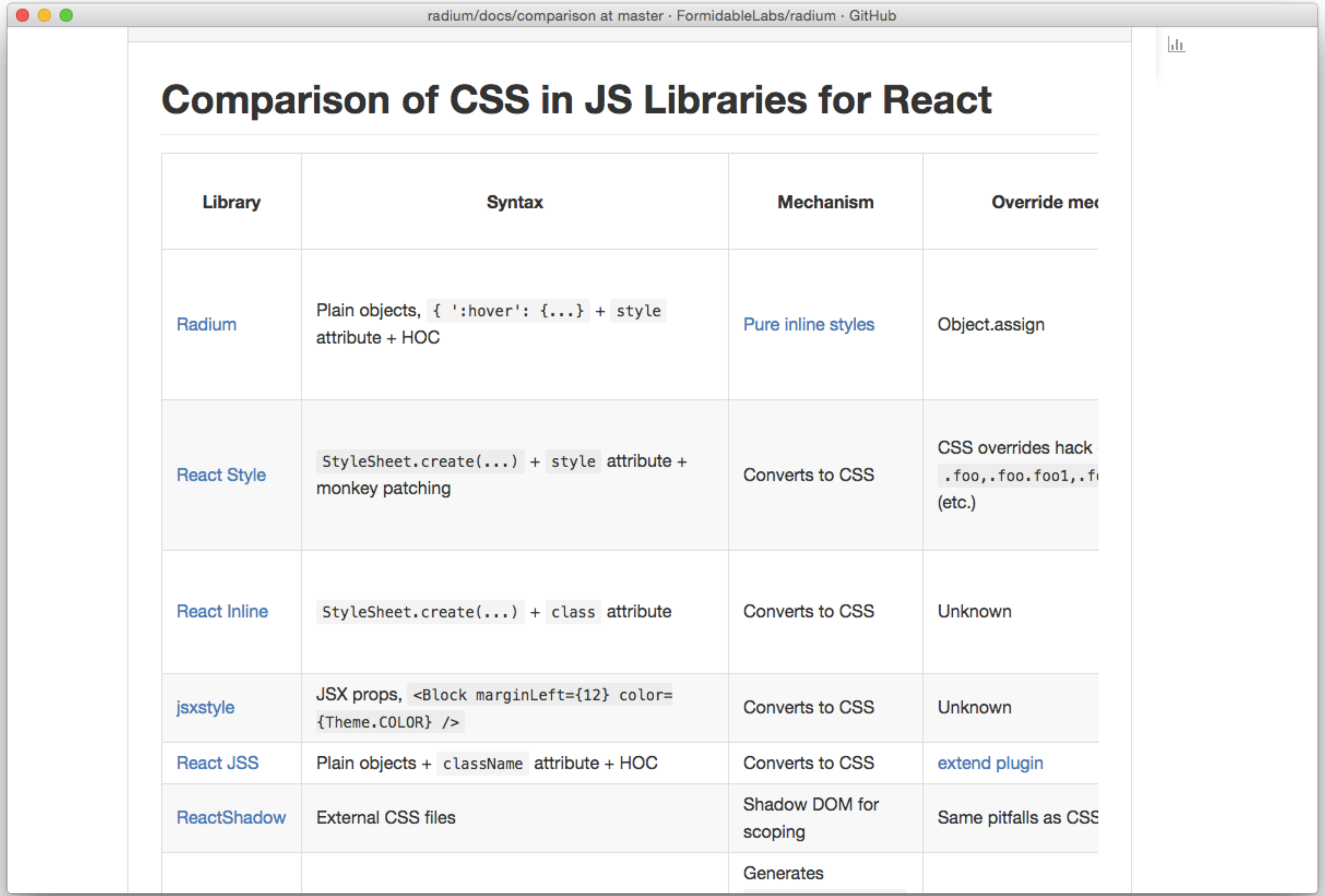
- Using Radium
- API Docs

Radium

```
styles = {
  base: {
    '@media (min-width: 320px)': {
      width: '100%',

      ':hover': {
        background: 'white'
      }
    }
  }
}
```

```
var Radium = require('radium');

// or
import Radium from 'radium'

// If you want to use the <Style /> component
// you can do
import Radium, { Style } from 'radium'
```

```
// For ES6 and ES7
@Radium
class Button extends React.Component {
  //...
}

// or
class Button extends React.Component {
  // ...
}
module.exports = Radium(Button);

// or
class Button extends React.Component {
  // ...
}
Button = Radium(Button);
```



radium/docs/comparison at master · FormidableLabs/radium · GitHub

# Comparison of CSS in JS Libraries for React

| Library | Syntax | Mechanism | Override mec |
|---|---|---|---|
| Radium | Plain objects, `{ ':hover': {...}` + `style` attribute + HOC | Pure inline styles | Object.assign |
| React Style | `StyleSheet.create(...)` + `style` attribute + monkey patching | Converts to CSS | CSS overrides hack `.foo,.foo.foo1,.f` (etc.) |
| React Inline | `StyleSheet.create(...)` + `class` attribute | Converts to CSS | Unknown |
| jsxstyle | JSX props, `<Block marginLeft={12} color={Theme.COLOR} />` | Converts to CSS | Unknown |
| React JSS | Plain objects + `className` attribute + HOC | Converts to CSS | extend plugin |
| ReactShadow | External CSS files | Shadow DOM for scoping | Same pitfalls as CSS |
| | | Generates | |

https://github.com/FormidableLabs/radium/tree/master/docs/comparison

## Best Practice ??

Give it a try !!