Why I am hooked on the future of React

@mauricedb

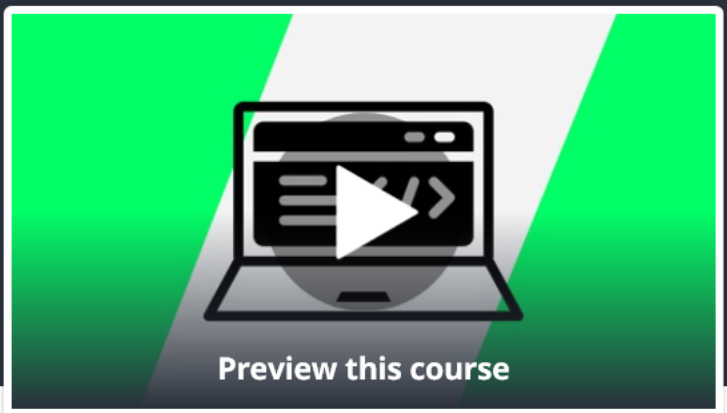
- Maurice de Beijer
- The Problem Solver
- Microsoft MVP
- Freelance developer/instructor
- Twitter: @mauricedb
- Web: http://www.TheProblemSolver.nl
- E-mail: maurice.de.beijer@gmail.com

Master RxJS 6 Without Breaking A Sweat

Learn how to solve common programming problems using RxJS

HIGHEST RATED 4.7 (11 ratings) 730 students enrolled

Created by Maurice de Beijer Last updated 12/2018 English English [Auto-generated]



€10.99 ~~€99.99~~ 89% off

5 hours left at this price!

Add to cart

Buy now

30-Day Money-Back Guarantee

This course includes

2.5 hours on-demand video

Full lifetime access

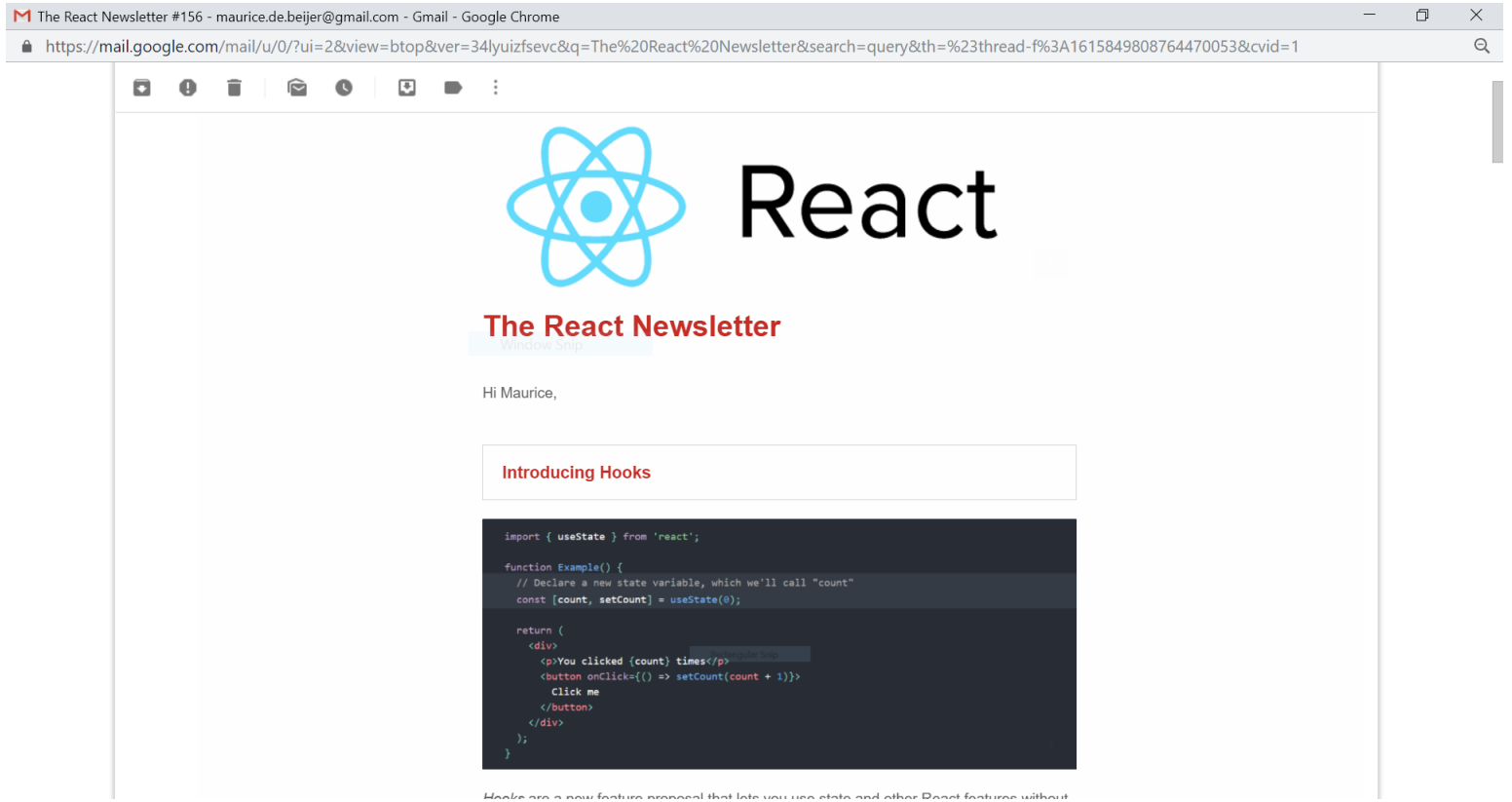
What you'll learn

- After this course you will be able to see where using RxJS makes sense.
- You will be able to solve common programming problems using RxJS.

Requirements

- Basic understanding of JavaScript is required.
- A PC with Node, NPM, a modern browser like Chrome or FireFox and a code editor you like is required.
- Any previous knowledge of RxJS is not required.

The React Newsletter



http://bit.ly/ReactNewsletter

History of Components

Original Component API

```
import React from 'react';

const Greeter = React.createClass({
  render: function() {
    return <div>Hello {this.props.firstName}</div>;
  }
});

export default Greeter;
```

Class Components

```
1 import React, { Component } from 'react';
2
3 class Greeter extends Component {
4   render() {
5     return <div>Hello {this.props.firstName}</div>;
6   }
7 }
8
9 export default Greeter;
```

State & Lifecycle

```jsx
import React, { Component } from 'react';

class Greeter extends Component {
  state = {
    loaded: false
  };

  componentDidMount() {
    this.setState({ loaded: true });
  }

  render() {
    return <div>Hello {this.props.firstName}</div>;
  }
}

export default Greeter;
```

The problem with classes

this

```
1 class Counter extends Component {
2   state = { count: 0 };
3
4   onClick() {
5     this.setState({ count: this.state.count + 1 });
```

```
Uncaught TypeError: Cannot read property 'setState' of undefined
    at onClick (Counter.js:79)
    at HTMLUnknownElement.callCallback (react-dom.development.js:147)
    at Object.invokeGuardedCallbackDev (react-dom.development.js:196)
    at invokeGuardedCallback (react-dom.development.js:250)
    at invokeGuardedCallbackAndCatchFirstError (react-dom.development.js:265)
    at executeDispatch (react-dom.development.js:622)
    at executeDispatchesInOrder (react-dom.development.js:647)
```

```
12         </button>
13     );
14   }
15 }
```

Fat arrow for some functions

```
1 class Counter extends Component {
2   state = { count: 0 };
3
4   onClick = () ⇒ {
5     this.setState({ count: this.state.count + 1 });
6   };
7
8   render() {
9     return (
10      <button onClick={this.onClick}>
11        Click me
12      </button>
13    );
14  }
15 }
```

Single responsibility in multiple functions

```
1 class Clock extends Component {
2   state = { now: new Date().toLocaleTimeString() };
3
4   componentDidMount() {
5     this.handle = setInterval(
6       () ⇒
7         this.setState({
8           now: new Date().toLocaleTimeString()
9         }),
10        1000
11    );
12  }
13  componentWillUnmount() {
14    clearInterval(this.handle);
15  }
16  render() {
17    return <div>{this.state.now}</div>;
18  }
19 }
```

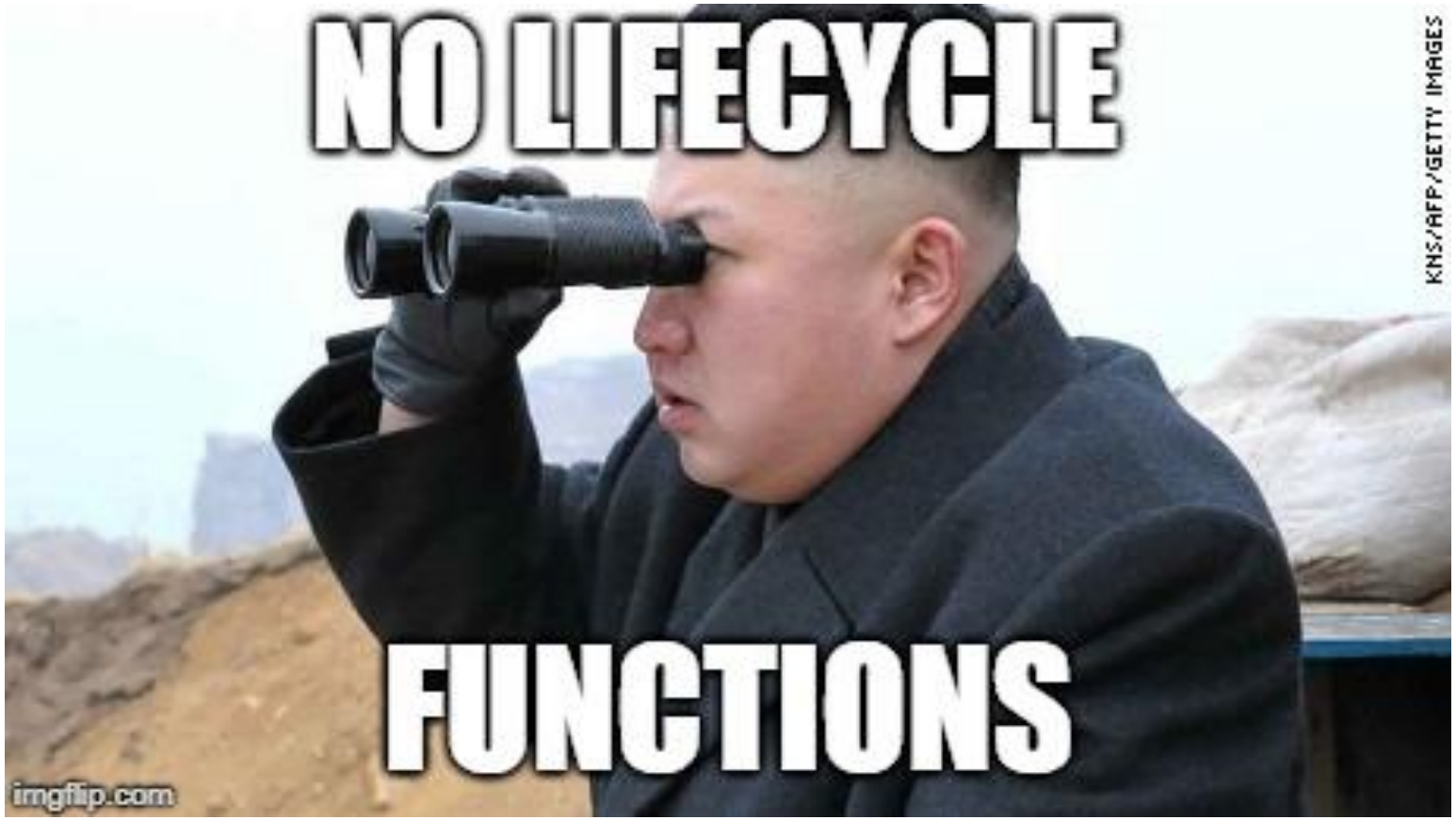
Functional Components

```
1 import React from 'react';
2
3 const Greeter = props ⇒ {
4   return <div>Hello {props.firstName}</div>;
5 };
6
7 export default Greeter;
```

The Problem

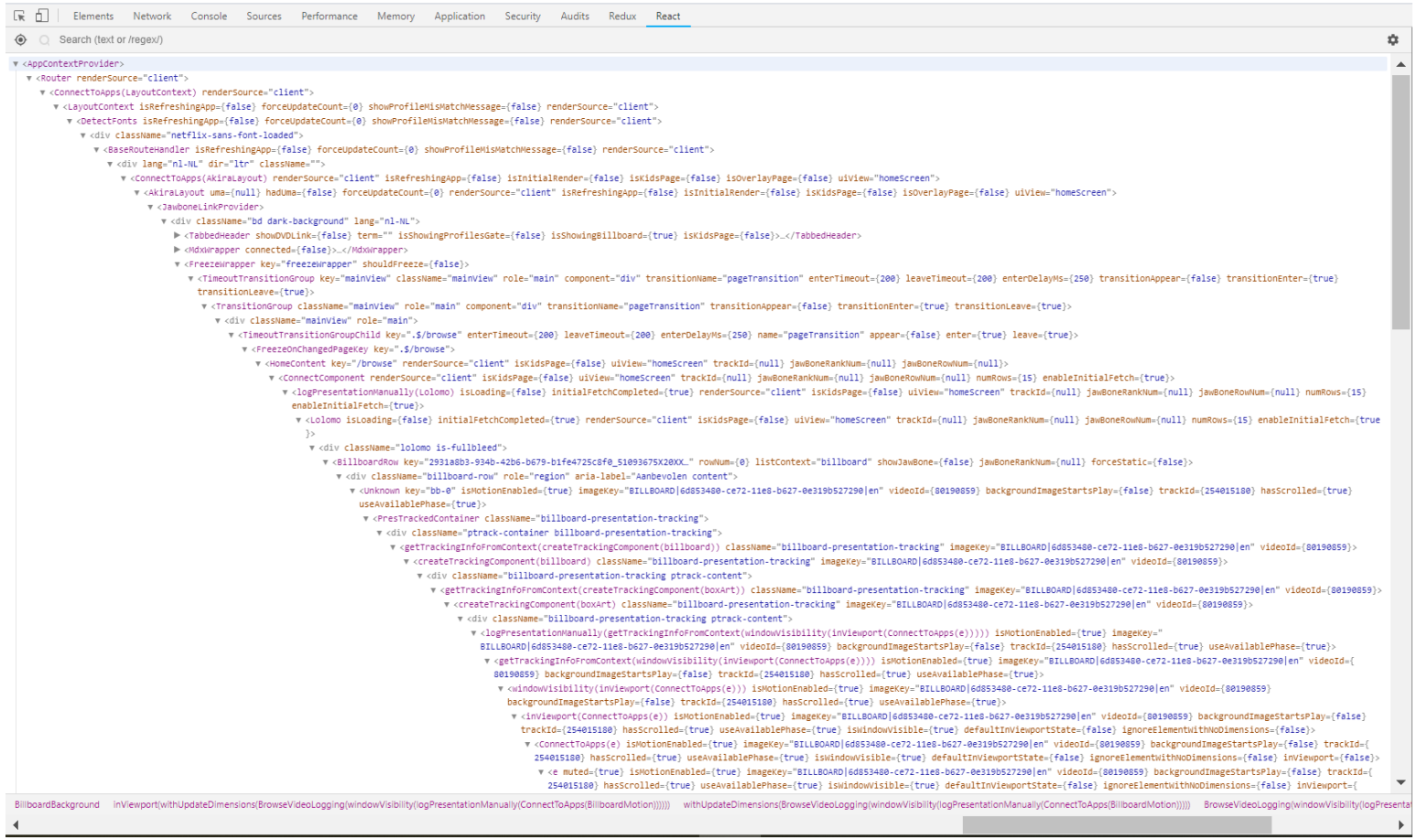

WHERE IS THE
STATE?
42-16607222 [RF] © www.visualphotos.com
memegenerator.net

And



If your only tool …

Deeply Nested Components

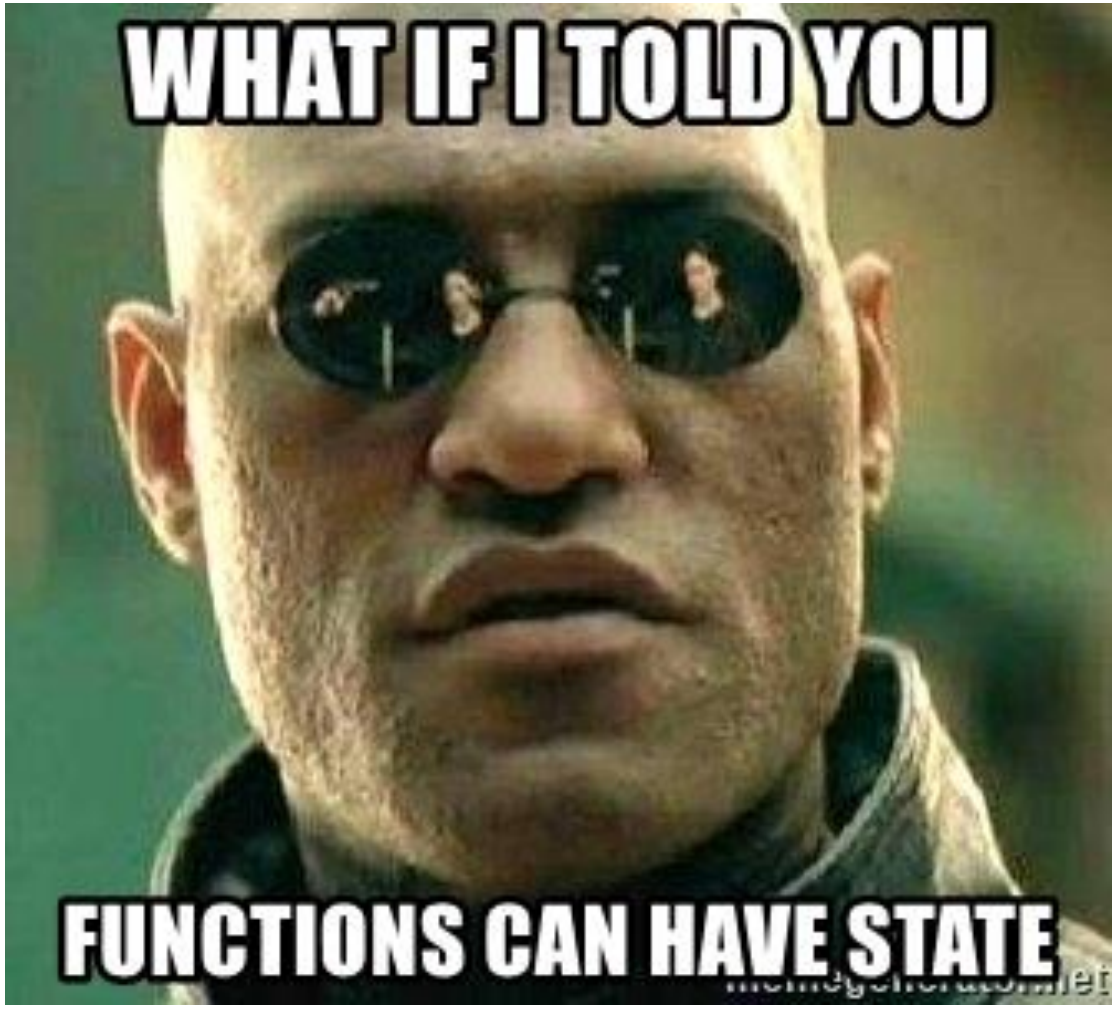


Many are wrappers to add props

Render Props

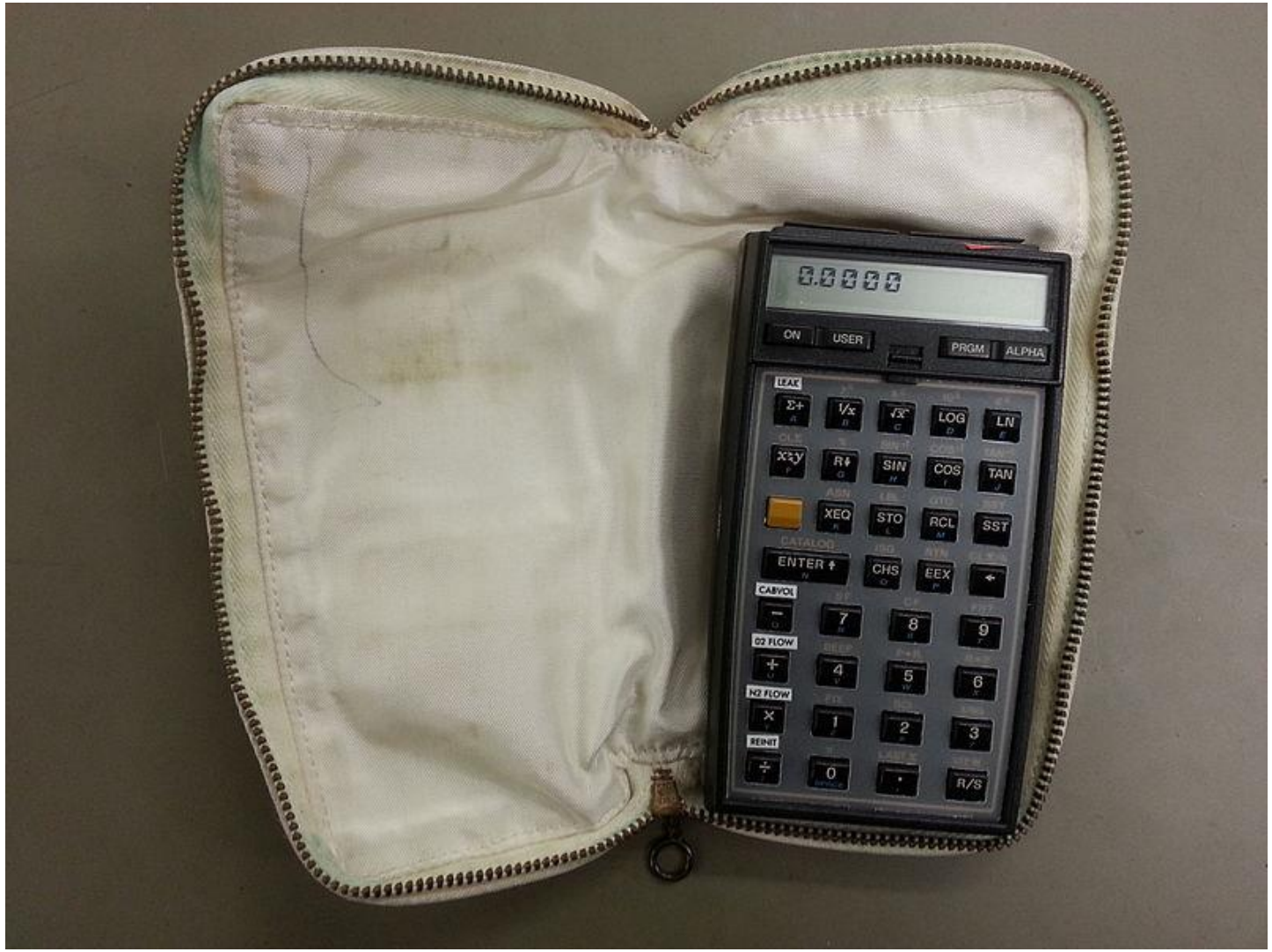
```jsx
import React, { Component } from 'react';
import TimeContext from './TimeContext';
import ThemeContext from './ThemeContext';
import AnalogClock from './AnalogClock';

class Clock extends Component {
  render() {
    return (
      <TimeContext.Consumer>
        {({ time }) => (
          <ThemeContext.Consumer>
            {({ theme }) => (
                <AnalogClock time={time} theme={theme} />
            )}
          </ThemeContext.Consumer>
        )}
      </TimeContext.Consumer>
    );
  }
}

export default Clock;
```

But...



Introducing Hooks

Open Mind

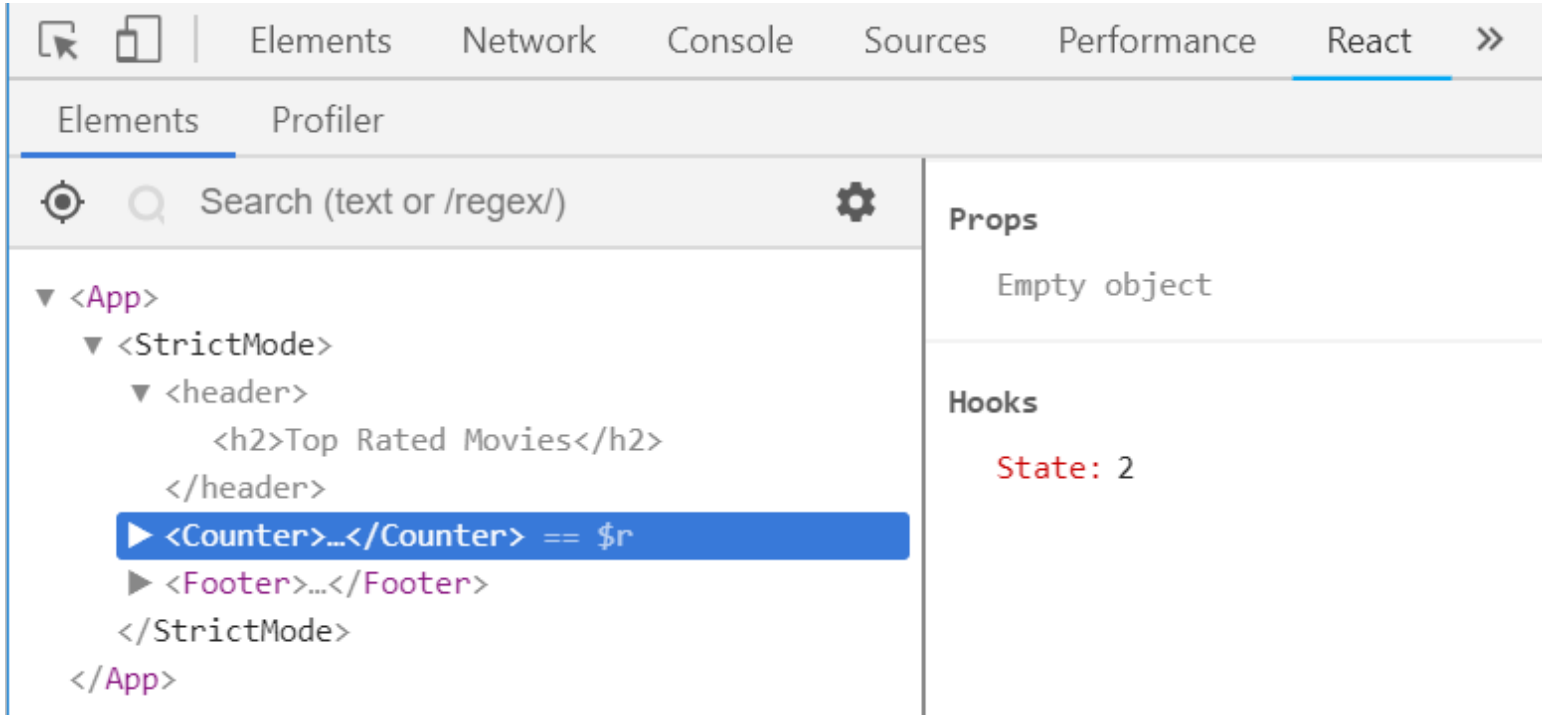


State

useState()

```jsx
import React, { useState } from 'react';

const Counter = () => {
  const [count, setCount] = useState(1);

  return (
    <div>
      <p>The counter is: {count}</p>
      <button onClick={
          () => setCount(count + 1)}>
        Increment
      </button>
    </div>
  );
};

export default Counter;
```

Just State



Side Effects

useEffect()

```jsx
import React, { useEffect, useState } from 'react';
import AnalogClock from './AnalogClock';

const Clock = ({ interval }) => {
  const [time, setTime] = useState(new Date());

  useEffect(
    () => {
      const handle = setInterval(
        () => setTime(new Date()), interval);

      return () => clearInterval(handle);
    },
    [interval]
  );

  return <AnalogClock time={time} />;
};

export default Clock;
```

Context

Render Props

```jsx
import React, { Component } from 'react';
import TimeContext from './TimeContext';
import ThemeContext from './ThemeContext';
import AnalogClock from './AnalogClock';

class Clock extends Component {
  render() {
    return (
      <TimeContext.Consumer>
        {({ time }) => (
          <ThemeContext.Consumer>
            {({ theme }) => (
                <AnalogClock time={time} theme={theme} />
            )}
          </ThemeContext.Consumer>
        )}
      </TimeContext.Consumer>
    );
  }
}

export default Clock;
```

useContext()

```jsx
import React, { useContext } from 'react';
import TimeContext from './TimeContext';
import ThemeContext from './ThemeContext';
import AnalogClock from './AnalogClock';

const Clock = () => {
  const time = useContext(TimeContext);
  const theme = useContext(ThemeContext);

  return <AnalogClock time={time} theme={theme}
};

export default Clock;
```

All hooks

- Basic hooks
  - useState()
  - useEffect()
  - useContext()
- Additional Hooks
  - useReducer()
  - useLayoutEffect()
  - useRef()
  - useCallback()
  - useMemo()
  - useImperativeHandle()
- Custom hooks
  - useDebugValue()
  - Your custom hooks

Custom Hooks

Custom Hooks

```jsx
import { useEffect, useState } from 'react';

const useTime = interval => {
  const [time, setTime] = useState(new Date());

  useEffect(
    () => {
      const handle = setInterval(
        () => setTime(new Date()), interval);

      return () => clearInterval(handle);
    },
    [interval]
  );
  return time;
};

export default useTime;
```

Using The Hook

```
1 import React from 'react';
2 import useTime from './useTime';
3 import AnalogClock from './AnalogClock';
4
5 const Clock = ({ interval }) => {
6   const time = useTime(interval);
7
8   return <AnalogClock time={time} />;
9 };
10
11 export default Clock;
```

Rules of Hooks

Rules of Hooks

- Hooks can only be used in functional components
  - Or in other hooks
  - Not in class based components
- Hooks must always be created in the same order
  - Must be outside loops, conditions or nested functions
- Hooks names must be prefixed with `use`
- There is an ESLint plugin to enforce these rules

Hooks are optional

Classes will keep on working

Maurice de Beijer

@mauricedb