

React Hooks

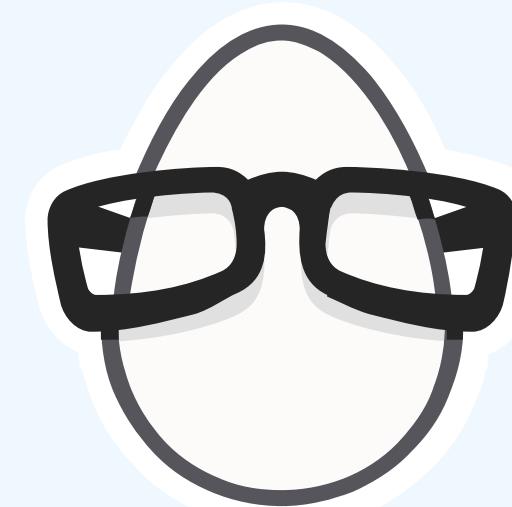
a guided tour





Dave
daveceddia.com

@dceddia

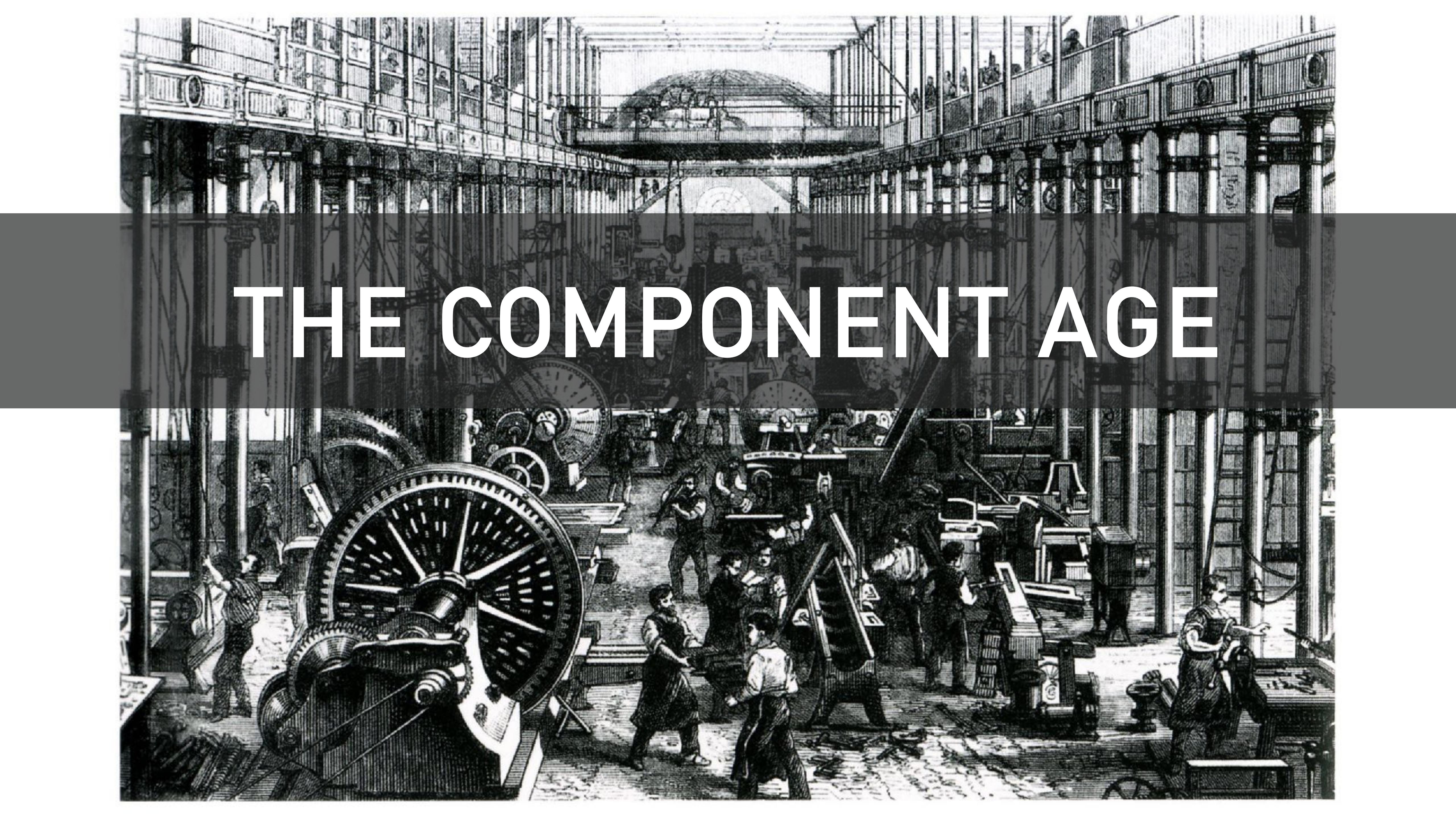


```
var Counter = React.createClass({
  getInitialState: function() {
    return {
      count: 1
    };
  },
  increment: function() {
    this.setState({
      count: this.state.count + 1
    });
  },
  decrement: function() {
    this.setState({
      count: this.state.count - 1
    });
  },
  render: function() {
    return (
      <div>
        Count is {this.state.count}
        <button onClick={this.increment}>Plus</button>
        <button onClick={this.decrement}>Minus</button>
      </div>
    );
  }
});
```

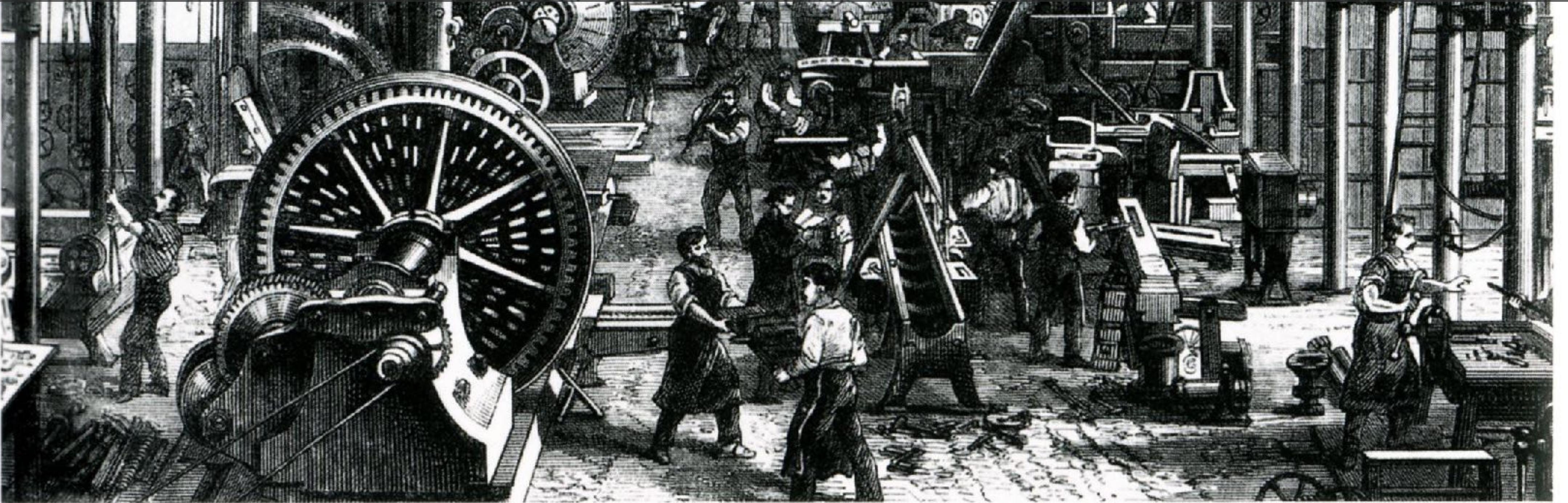
var!



```
var Counter = React.createClass({
  getInitialState: function() {
    return {
      count: 1
    };
  },
  increment: function() {
    this.setState({
      count: this.state.count + 1
    });
  },
  decrement: function() {
    this.setState({
      count: this.state.count - 1
    });
  },
  render: function() {
    return (
      <div>
        Count is {this.state.count}
        <button onClick={this.increment}>Plus</button>
        <button onClick={this.decrement}>Minus</button>
      </div>
    );
  }
});
```



THE COMPONENT AGE



```
class Counter extends React.Component {
  state = {
    count: 1
  };

  increment = () => {
    this.setState({
      count: this.state.count + 1
    });
  };

  decrement = () => {
    this.setState({
      count: this.state.count - 1
    });
  };

  render() {
    return (
      <div>
        Count is {this.state.count}
        <button onClick={this.increment}>Plus</button>
        <button onClick={this.decrement}>Minus</button>
      </div>
    );
  }
}
```

```
class Counter extends React.Component {
  state = {
    count: 1
  };

  increment = () => {
    this.setState({
      count: this.state.count + 1
    });
  };

  decrement = () => {
    this.setState({
      count: this.state.count - 1
    });
  };

  render() {
    return (
      <div>
        Count is {this.state.count}
        <button onClick={this.increment}>Plus</button>
        <button onClick={this.decrement}>Minus</button>
      </div>
    );
  }
}
```



Lifecycle Methods

```
class BlogPost extends React.Component {
  state = {
    comments: [],
    loading: false
  };

  componentDidMount() {
    this.updateComments(this.props.postId);
  }

  componentDidUpdate(prevProps) {
    if (prevProps.postId !== this.props.postId) {
      this.updateComments(this.props.postId);
    }
  }

  updateComments(postId) {
    this.setState({ loading: true });
    fetchComments(postId).then(comments => {
      this.setState({
        comments,
        loading: false
      });
    });
  }

  render() {
    if (this.state.loading) {
      return <Loading />;
    }

    return (
      <ul>
        {this.state.comments.map(comment => (
          <li key={comment.id}>{comment.text}</li>
        )));
      </ul>
    );
  }
}
```

```
componentDidMount() {
  this.updateComments(this.props.postId);
}

componentDidUpdate(prevProps) {
  if (prevProps.postId !== this.props.postId) {
    this.updateComments(this.props.postId);
  }
}

updateComments(postId) {
  this.setState({ loading: true });
  fetchComments(postId).then(comments => {
    this.setState({
      comments,
      loading: false
    });
  });
}
```

```
componentDidMount() {
  this.updateComments(this.props.postId);
}

componentDidUpdate(prevProps) {
  if (prevProps.postId !== this.props.postId) {
    this.updateComments(this.props.postId);
  }
}

updateComments(postId) {
  this.setState({ loading: true });
  fetchComments(postId).then(comments => {
    this.setState({
      comments,
      loading: false
    });
  });
}
```

```
componentDidMount() {
  this.updateComments(this.props.postId);
}

componentDidUpdate(prevProps) {
  if (prevProps.postId !== this.props.postId) {
    this.updateComments(this.props.postId);
  }
}

updateComments(postId) {
  this.setState({ loading: true });
  fetchComments(postId).then(comments => {
    this.setState({
      comments,
      loading: false
    });
  });
}
```

```
componentDidMount() {
  this.updateComments(this.props.postId);
}

componentDidUpdate(prevProps) {
  if (prevProps.postId !== this.props.postId) {
    this.updateComments(this.props.postId);
  }
}

updateComments(postId) {
  this.setState({ loading: true });
  fetchComments(postId).then(comments => {
    this.setState({
      comments,
      loading: false
    });
  });
}
```


React Conf 2018

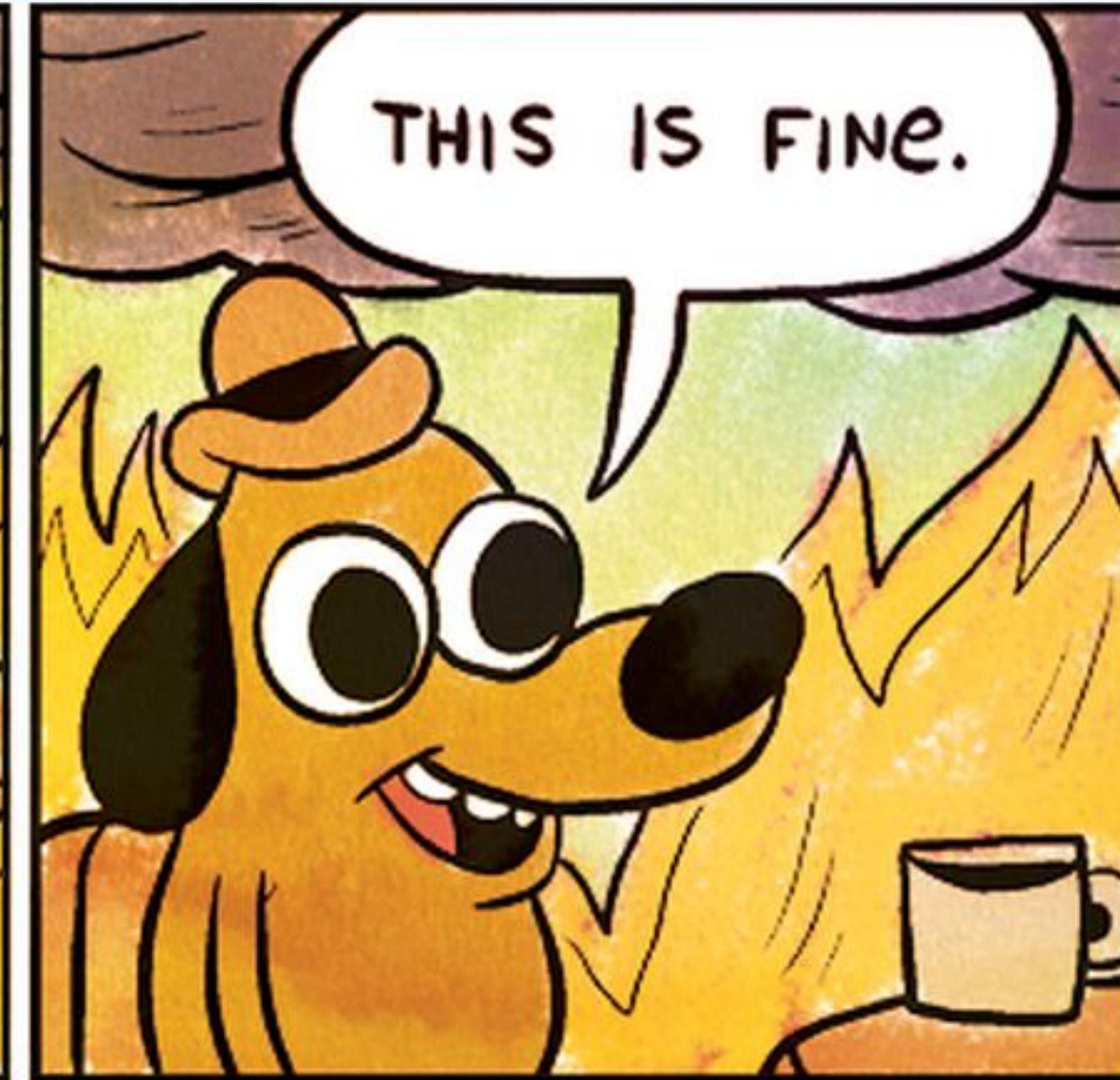


We have **a proposal**.

- 👉 **No breaking changes**
- 💡 **Proposed APIs are new**
- 💬 **We need your feedback**

A video frame showing a man with short dark hair speaking. He is wearing a black t-shirt with a React logo. The video player interface at the bottom shows a progress bar at 15:49 / 1:35:29.







shawn swyx wang  liked



Tanner Linsley @tannerlinsley · Oct 25

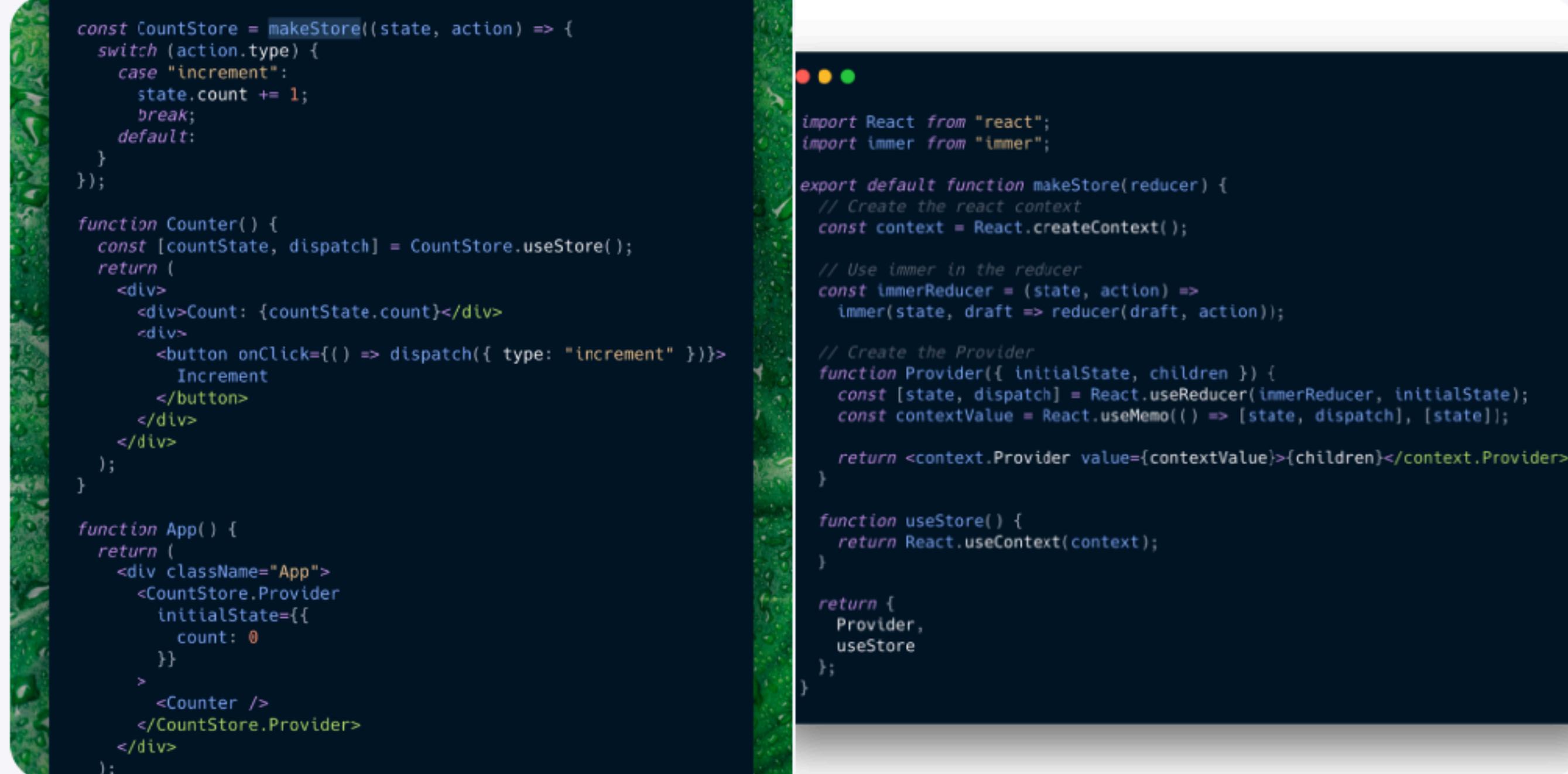
React **Hooks** + immer + useContext + useReducer = Provider/hook factory for global state.



HOLD UP... Did I just made my own **redux** in 2 tiny functions???



#ReactConf2018 #ReactHooks



The image shows a mobile device screen with two code snippets displayed side-by-side. The left snippet is a React component named `CountStore` which contains a reducer function and a `Counter` component. The right snippet is a `makeStore` function which creates a React context, uses `immer` for the reducer, and provides a `Provider` component along with a `useStore` hook.

```
const CountStore = makeStore((state, action) => {
  switch (action.type) {
    case "increment":
      state.count += 1;
      break;
    default:
  }
});

function Counter() {
  const [countState, dispatch] = CountStore.useStore();
  return (
    <div>
      <div>Count: {countState.count}</div>
      <div>
        <button onClick={() => dispatch({ type: "increment" })}>
          Increment
        </button>
      </div>
    </div>
  );
}

function App() {
  return (
    <div className="App">
      <CountStore.Provider
        initialState={{
          count: 0
        }>
        <Counter />
      </CountStore.Provider>
    </div>
  );
}
```

```
import React from "react";
import immer from "immer";

export default function makeStore(reducer) {
  // Create the react context
  const context = React.createContext();

  // Use immer in the reducer
  const immerReducer = (state, action) =>
    immer(state, draft => reducer(draft, action));

  // Create the Provider
  function Provider({ initialState, children }) {
    const [state, dispatch] = React.useReducer(immerReducer, initialState);
    const contextValue = React.useMemo(() => [state, dispatch], [state]);

    return <context.Provider value={contextValue}>{children}</context.Provider>;
  }

  function useStore() {
    return React.useContext(context);
  }

  return {
    Provider,
    useStore
  };
}
```



2



6



33

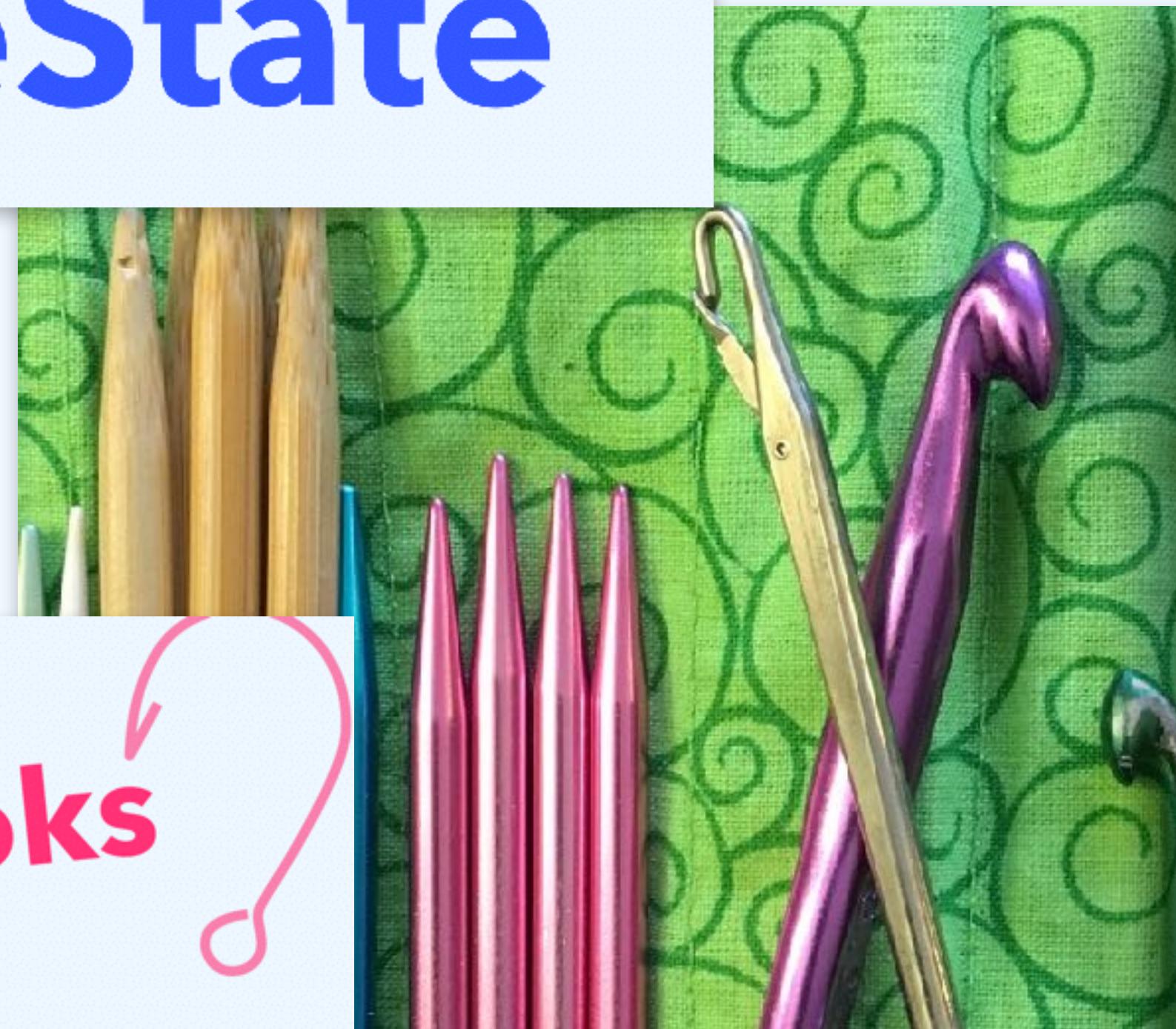






Hooks Week™

React **Hooks**
useState

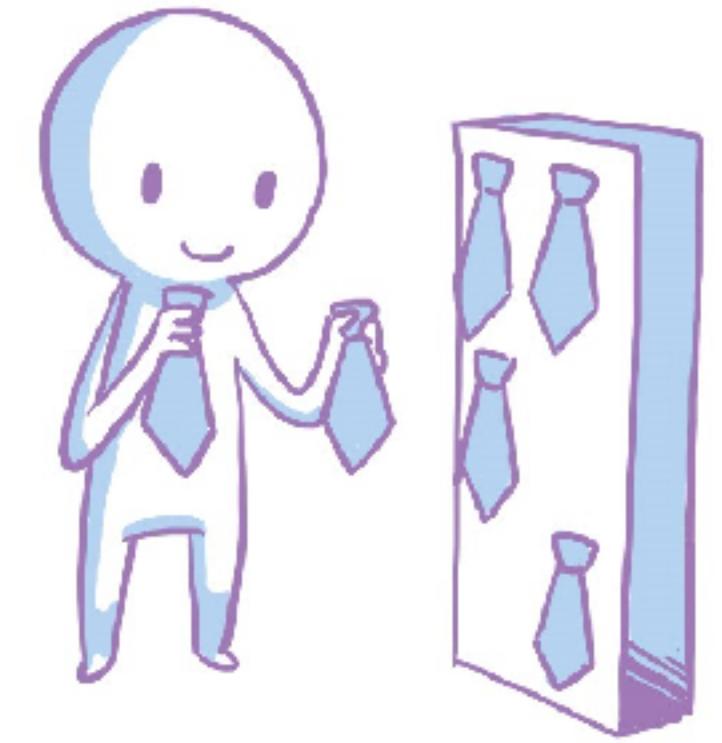


React **Hooks**
useEffect

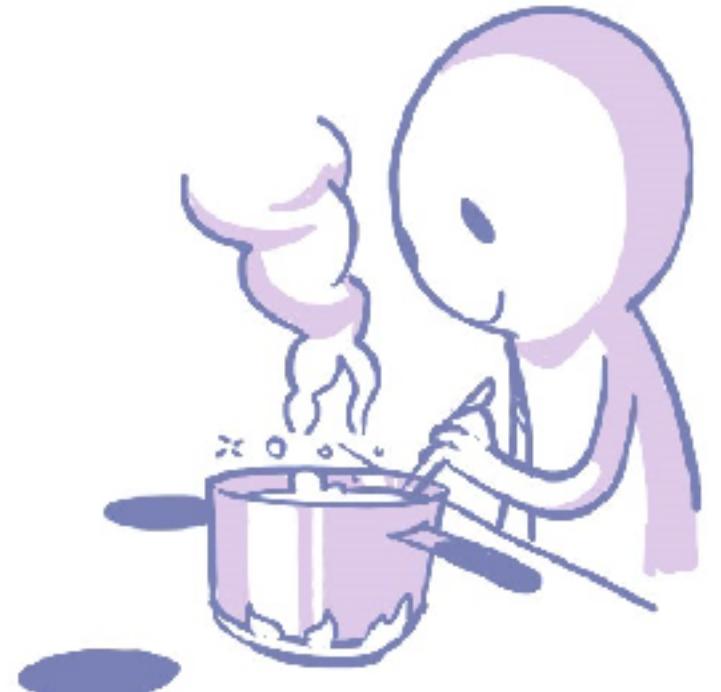
React **Hooks**
useReducer

React
Hooks
an introduction

React **Hooks**
useContext



SHOP FOR **hooks**



MAKE **hooks**



DO **hooks**



DON'T LET THE EXISTENTIAL
DREAD SET IN.



DON'T LET IT SET IN.



VACUUM THE **hooks**



DON'T LET IT SET IN.

Filter Hide data URLs All | XHR JS CSS Img Media Font Doc WS Manifest Other

2000 ms 4000 ms 6000 ms 8000 ms 10000 ms 12000 ms 14000 ms 16000 ms 18000 ms 20000 ms 22000 ms

Name	Meth...	Status	Type	Initiator	Size	Time	Waterfall
reactjs.json www.reddit.com/r	GET	200	json	Other	21.3 KB 128 KB	226 ms 220 ms	
reactjs.json www.reddit.com/r	GET	200	json	Other	21.7 KB 128 KB	203 ms 198 ms	
reactjs.json www.reddit.com/r	GET	200	json	Other	21.4 KB 128 KB	223 ms 219 ms	
reactjs.json www.reddit.com/r	GET	200	json	Other	21.3 KB 128 KB	201 ms 196 ms	
reactjs.json www.reddit.com/r	GET	200	json	Other	21.7 KB 128 KB	220 ms 214 ms	
reactjs.json www.reddit.com/r	GET	200	json	Other	21.3 KB 128 KB	260 ms 255 ms	
reactjs.json www.reddit.com/r	GET	200	json	Other	21.3 KB 128 KB	290 ms 286 ms	
reactjs.json www.reddit.com/r	GET	200	json	Other	21.8 KB 128 KB	217 ms 212 ms	
reactjs.json www.reddit.com/r	GET	200	json	Other	21.4 KB 128 KB	198 ms 189 ms	
reactjs.json www.reddit.com/r	GET	200	json	Other	21.3 KB 128 KB	263 ms 259 ms	
reactjs.json www.reddit.com/r	GET	200	json	Other	21.7 KB 128 KB	226 ms 221 ms	
reactjs.json www.reddit.com/r	GET	200	json	Other	21.4 KB 128 KB	223 ms 217 ms	



React Hooks

a guided tour





Hooks are additive.

They don't replace classes.

```
JS index.js x
1 import React from 'react';
2 import ReactDOM from 'react-dom';
3 import './index.css';
4
5 function App() {
6   return (
7     <div>
8       <h1>Hello React Boston!</h1>
9     </div>
10 );
11 }
12
13 ReactDOM.render(
14   <App />,
15   document.querySelector('#root')
16 );
17
```

React App x +

localhost:3000

Hello React Boston!

JS index.js

```
1 import React from 'react';
2 import ReactDOM from 'react-dom';
3 import './index.css';
4
5 function App() {
6   return (
7     <div>
8       <h1>Hello React Boston!</h1>
9     </div>
10 );
11 }
12
13 ReactDOM.render(
14   <App />,
15   document.querySelector('#root')
16 );
17
```

React App

localhost:3000

Hello React Boston!

How does that even work?

```
JS index.js x
1 import React, { useState } from 'react';
2 import ReactDOM from 'react-dom';
3 import './index.css';
4
5 function App() {
6   const [people, setPeople] = useState(300);
7
8   return (
9     <div>
10    <h1>Hello React Boston!</h1>
11    <p>People here today: {people}</p>
12    <button onClick={() => setPeople(people + 1)}>
13      Welcome One More
14    </button>
15  </div>
16);
17}
18
19 ReactDOM.render(
```

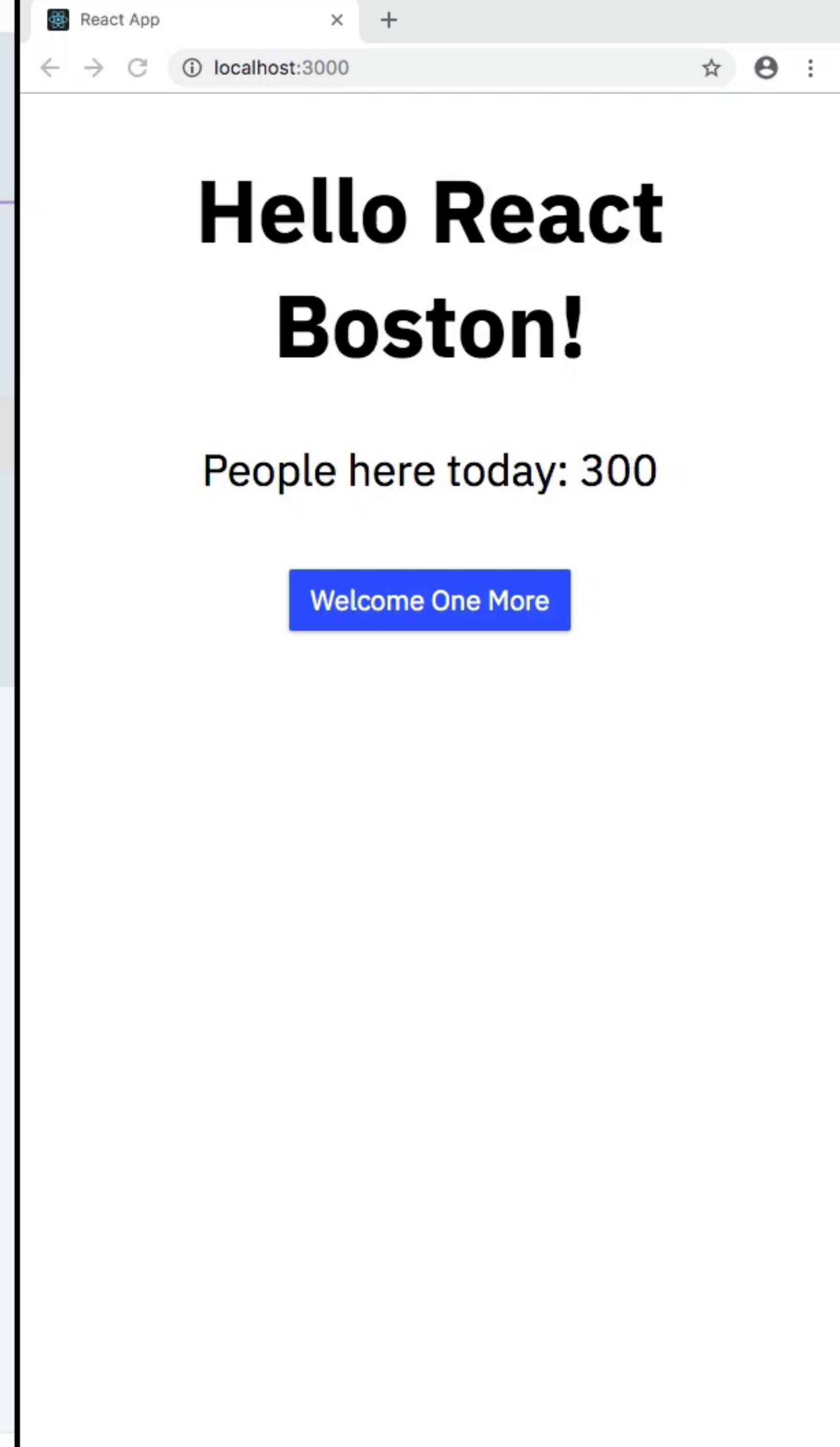
React App x +

localhost:3000

Hello React Boston!

People here today: 300

Welcome One More

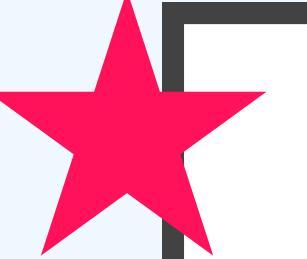


```
function App() {  
  const [people, setPeople] = useState(300);  
  
  return (  
    <div>  
      <h1>Hello React Boston!</h1>  
      <p>People here today: {people}</p>  
      <button onClick={() => setPeople(people + 1)}>  
        Welcome One More  
      </button>  
    </div>  
  );  
}
```

hooks

```
function App() {  
  const [people, setPeople] = useState(300);  
  
  return (  
    <div>  
      <h1>Hello React Boston!</h1>  
      <p>People here today: {people}</p>  
      <button onClick={() => setPeople(people + 1)}>  
        Welcome One More  
      </button>  
    </div>  
  );  
}
```

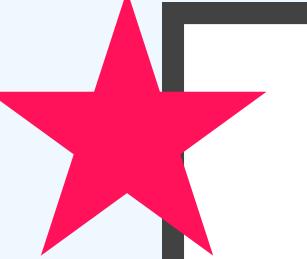
hooks



```
function App() {
  const [people, setPeople] = useState(300);

  return (
    <div>
      <h1>Hello React Boston!</h1>
      <p>People here today: {people}</p>
      <button onClick={() => setPeople(people + 1)}>
        Welcome One More
      </button>
    </div>
  );
}
```

hooks

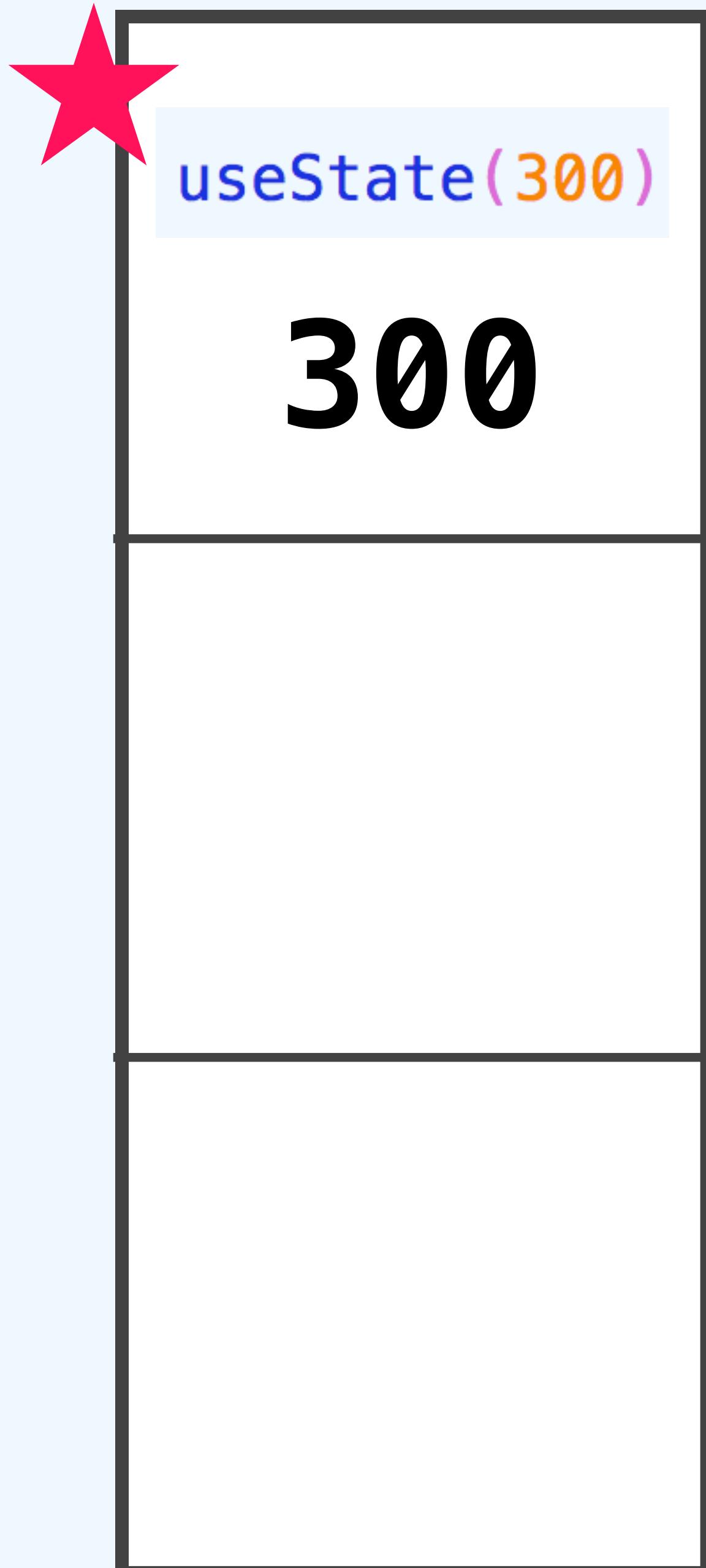


```
function App() {
  const [people, setPeople] = useState(300);

  return (
    <div>
      <h1>Hello React Boston!</h1>
      <p>People here today: {people}</p>
      <button onClick={() => setPeople(people + 1)}>
        Welcome One More
      </button>
    </div>
  );
}
```

hooks

```
function App() {  
  const [people, setPeople] = useState(300);  
  
  return (  
    <div>  
      <h1>Hello React Boston!</h1>  
      <p>People here today: {people}</p>  
      <button onClick={() => setPeople(people + 1)}>  
        Welcome One More  
      </button>  
    </div>  
  );  
}
```

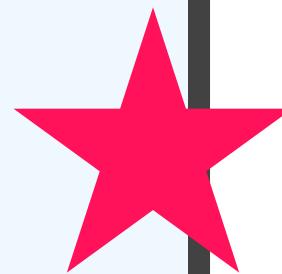


hooks

```
function App() {  
  const [people, setPeople] = useState(300);  
  
  return (  
    <div>  
      <h1>Hello React Boston!</h1>  
      <p>People here today: {people}</p>  
      <button onClick={() => setPeople(people + 1)}>  
        Welcome One More  
      </button>  
    </div>  
  );  
}
```

useState(300)

300

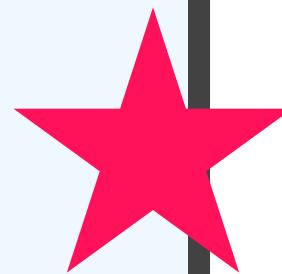


hooks

```
function App() {  
  const [people, setPeople] = useState(300);  
  
  return (  
    <div>  
      <h1>Hello React Boston!</h1>  
      <p>People here today: {people}</p>  
      <button onClick={() => setPeople(people + 1)}>  
        Welcome One More  
      </button>  
    </div>  
  );  
}
```

useState(300)

300



```
function App() {  
  const [people, setPeople] = useState(300);  
  
  return (  
    <div>  
      <h1>Hello React Boston!</h1>  
      <p>People here today: {people}</p>  
      <button onClick={() => setPeople(people + 1)}>  
        Welcome One More  
      </button>  
    </div>  
  );  
}
```

hooks

useState(300)

301

```
function App() {  
  const [people, setPeople] = useState(300);  
  
  return (  
    <div>  
      <h1>Hello React Boston!</h1>  
      <p>People here today: {people}</p>  
      <button onClick={() => setPeople(people + 1)}>  
        Welcome One More  
      </button>  
    </div>  
  );  
}
```

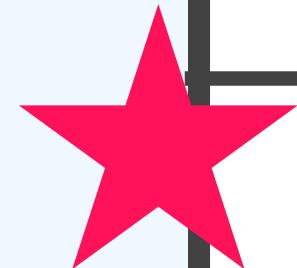


hooks

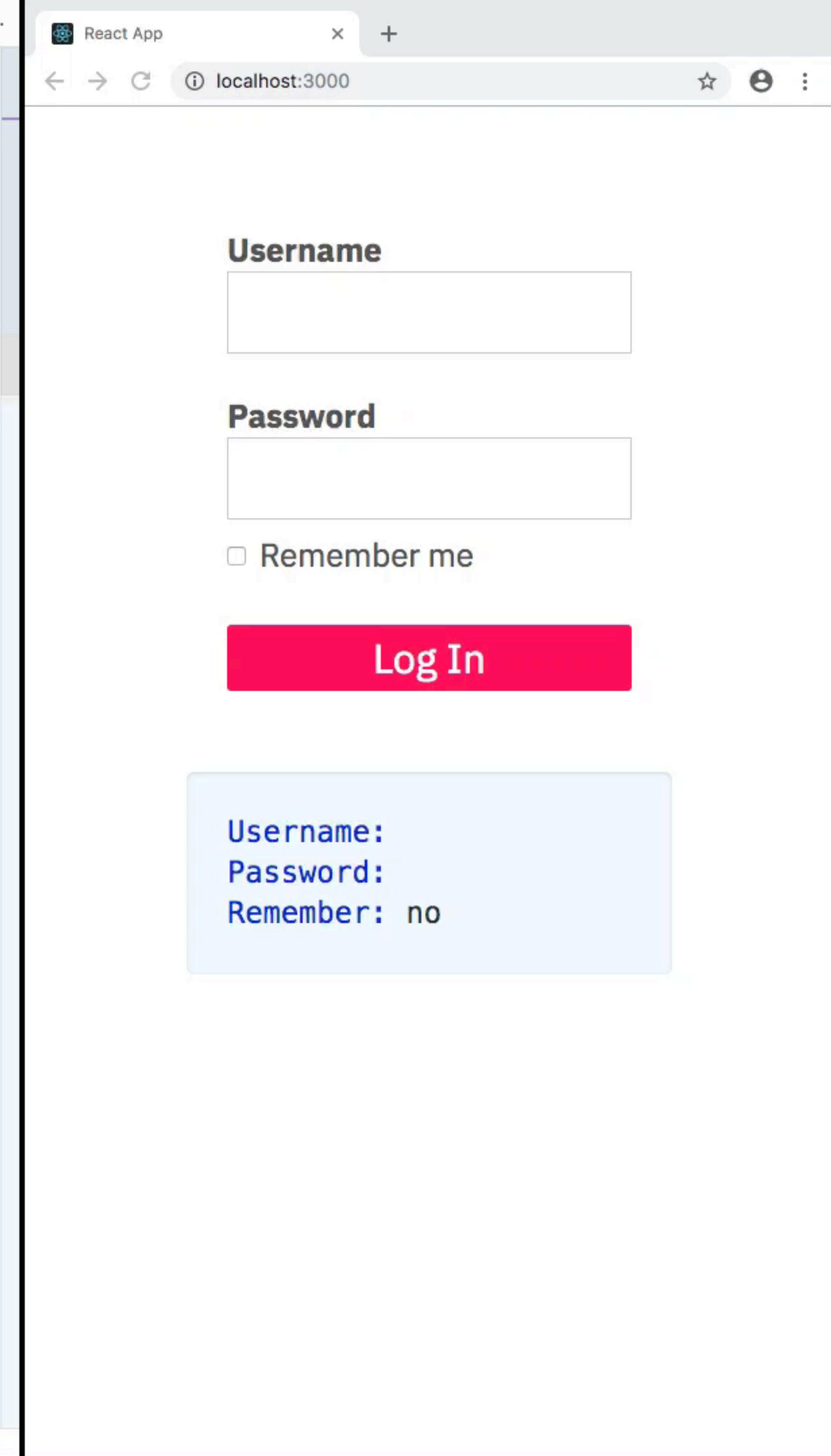
```
function App() {  
  const [people, setPeople] = useState(300);  
  
  return (  
    <div>  
      <h1>Hello React Boston!</h1>  
      <p>People here today: {people}</p>  
      <button onClick={() => setPeople(people + 1)}>  
        Welcome One More  
      </button>  
    </div>  
  );  
}
```

useState(300)

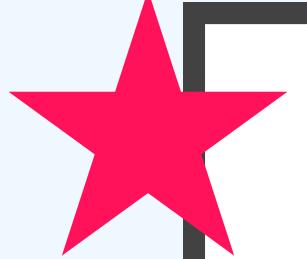
301



```
JS index.js x
1 import React, { useState } from 'react';
2 import ReactDOM from 'react-dom';
3 import './index.css';
4
5 const LoginForm = () => {
6   const [username, setUsername] = useState('');
7   const [password, setPassword] = useState('');
8   const [remember, setRememberMe] = useState(false);
9
10  return (
11    <form>
12      <label htmlFor="username">Username</label>
13      <input
14        value={username}
15        onChange={e => setUsername(e.target.value)}
16        id="username"
17        type="text"
18      />
19    )
}
```



hooks

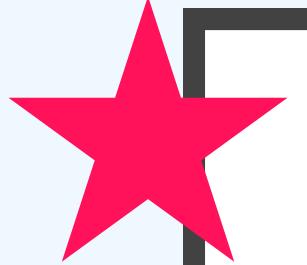


```
const LoginForm = () => {
  const [username, setUsername] = useState('');
  const [password, setPassword] = useState('');
  const [remember, setRememberMe] = useState(false);

  return (
    <form>
      <label htmlFor="username">Username</label>
      <input
        value={username}
        onChange={e => setUsername(e.target.value)}
        id="username"
        type="text"
      />
    ...
  )
}
```

hooks

```
const LoginForm = () => {  
  const [username, setUsername] = useState('');  
  const [password, setPassword] = useState('');  
  const [remember, setRememberMe] = useState(false);  
  
  return (  
    <form>  
      <label htmlFor="username">Username</label>  
      <input  
        value={username}  
        onChange={e => setUsername(e.target.value)}  
        id="username"  
        type="text"  
      />  
    ...  
  )
```



hooks

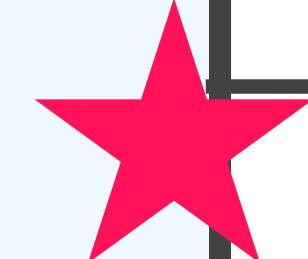
```
const LoginForm = () => {
  const [username, setUsername] = useState('');
  const [password, setPassword] = useState('');
  const [remember, setRememberMe] = useState(false);

  return (
    <form>
      <label htmlFor="username">Username</label>
      <input
        value={username}
        onChange={e => setUsername(e.target.value)}
        id="username"
        type="text"
      />
    ...
  )
}
```

useState('')

..

(empty string)



hooks

```
const LoginForm = () => {
  const [username, setUsername] = useState('');
  const [password, setPassword] = useState('');
  const [remember, setRememberMe] = useState(false);

  return (
    <form>
      <label htmlFor="username">Username</label>
      <input
        value={username}
        onChange={e => setUsername(e.target.value)}
        id="username"
        type="text"
      />
    ...
  )
}
```

useState('')

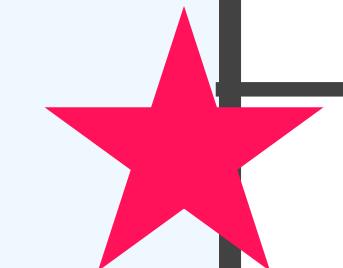
..

(empty string)

useState('')

..

(empty string)



hooks

```
const LoginForm = () => {
  const [username, setUsername] = useState('');
  const [password, setPassword] = useState('');
  const [remember, setRememberMe] = useState(false);

  return (
    <form>
      <label htmlFor="username">Username</label>
      <input
        value={username}
        onChange={e => setUsername(e.target.value)}
        id="username"
        type="text"
      />
    ...
  )
}
```

useState('')

..

(empty string)

useState('')

..

(empty string)

useState(false)

false

hooks

```
const LoginForm = () => {
  const [username, setUsername] = useState('');
  const [password, setPassword] = useState('');
  const [remember, setRememberMe] = useState(false);

  return (
    <form>
      <label htmlFor="username">Username</label>
      <input
        value={username}
        onChange={e => setUsername(e.target.value)}
        id="username"
        type="text"
      />
    ...
  )
}
```

useState('')

d

useState('')

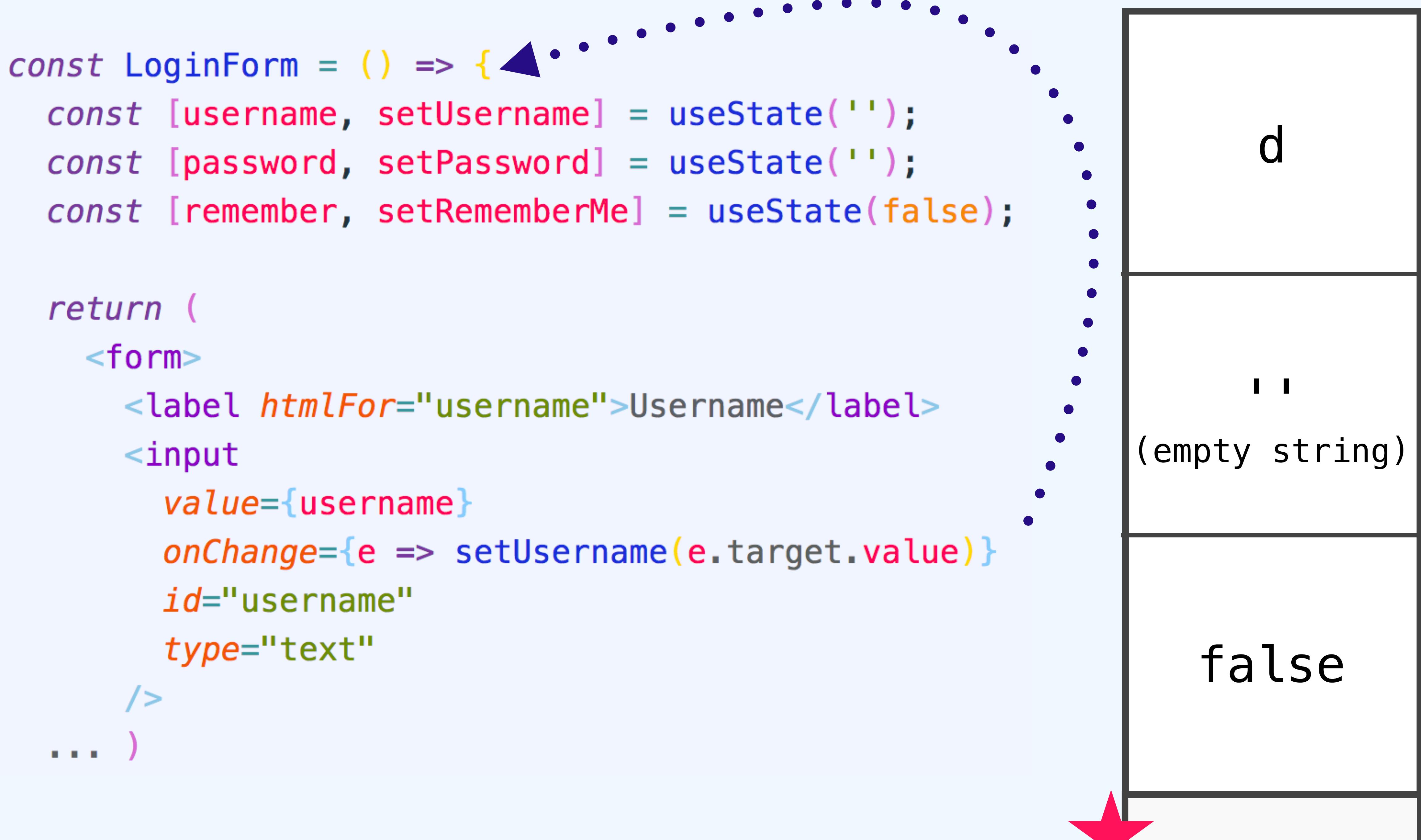
..

(empty string)

useState(false)

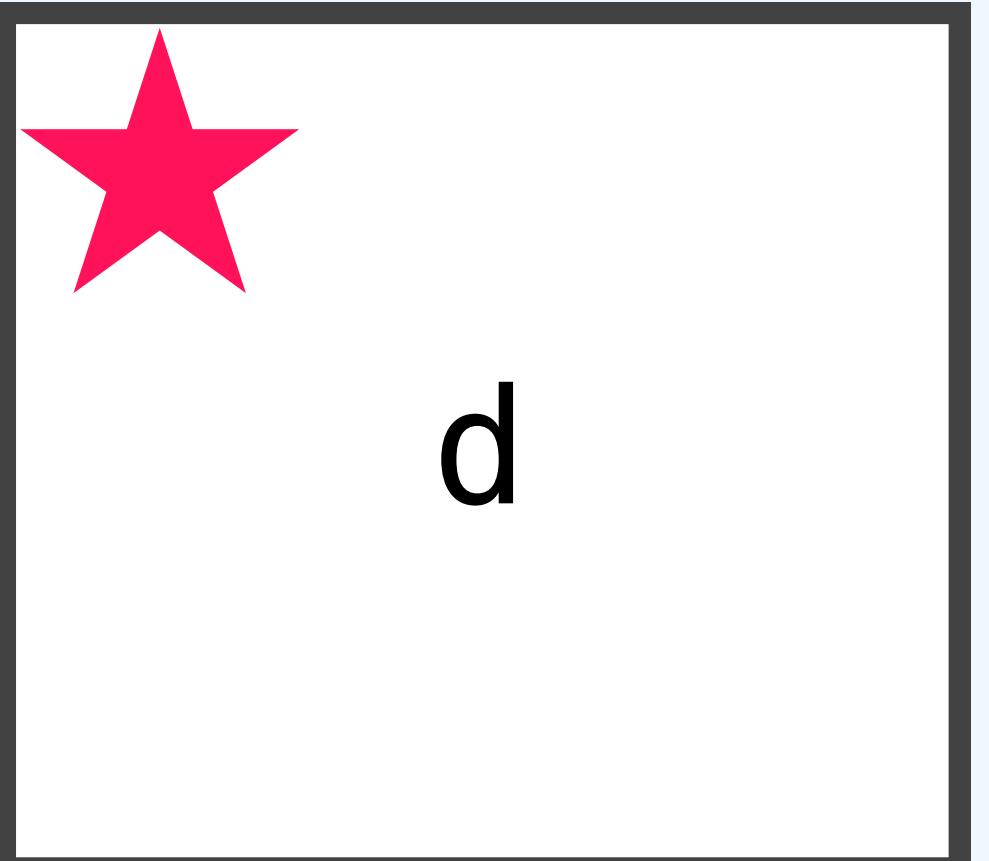
false

hooks



hooks

```
const LoginForm = () => {  
  const [username, setUsername] = useState('');  
  const [password, setPassword] = useState('');  
  const [remember, setRememberMe] = useState(false);  
  
  return (  
    <form>  
      <label htmlFor="username">Username</label>  
      <input  
        value={username}  
        onChange={e => setUsername(e.target.value)}  
        id="username"  
        type="text"  
      />  
    ...  
  )
```



d

..

(empty string)

false

hooks

```
const LoginForm = () => {
  const [username, setUsername] = useState('');
  const [password, setPassword] = useState('');
  const [remember, setRememberMe] = useState(false);

  return (
    <form>
      <label htmlFor="username">Username</label>
      <input
        value={username}
        onChange={e => setUsername(e.target.value)}
        id="username"
        type="text"
      />
    ...
  )
}
```

	d
''	(empty string)
	false

Rules of Hooks

1. Call order must be stable

No loops, conditionals, nested functions.

2. Only call from function components

...or custom hooks. Sorry, classes.

3. Names should start with “use”

Help the linter out.

React Hooks

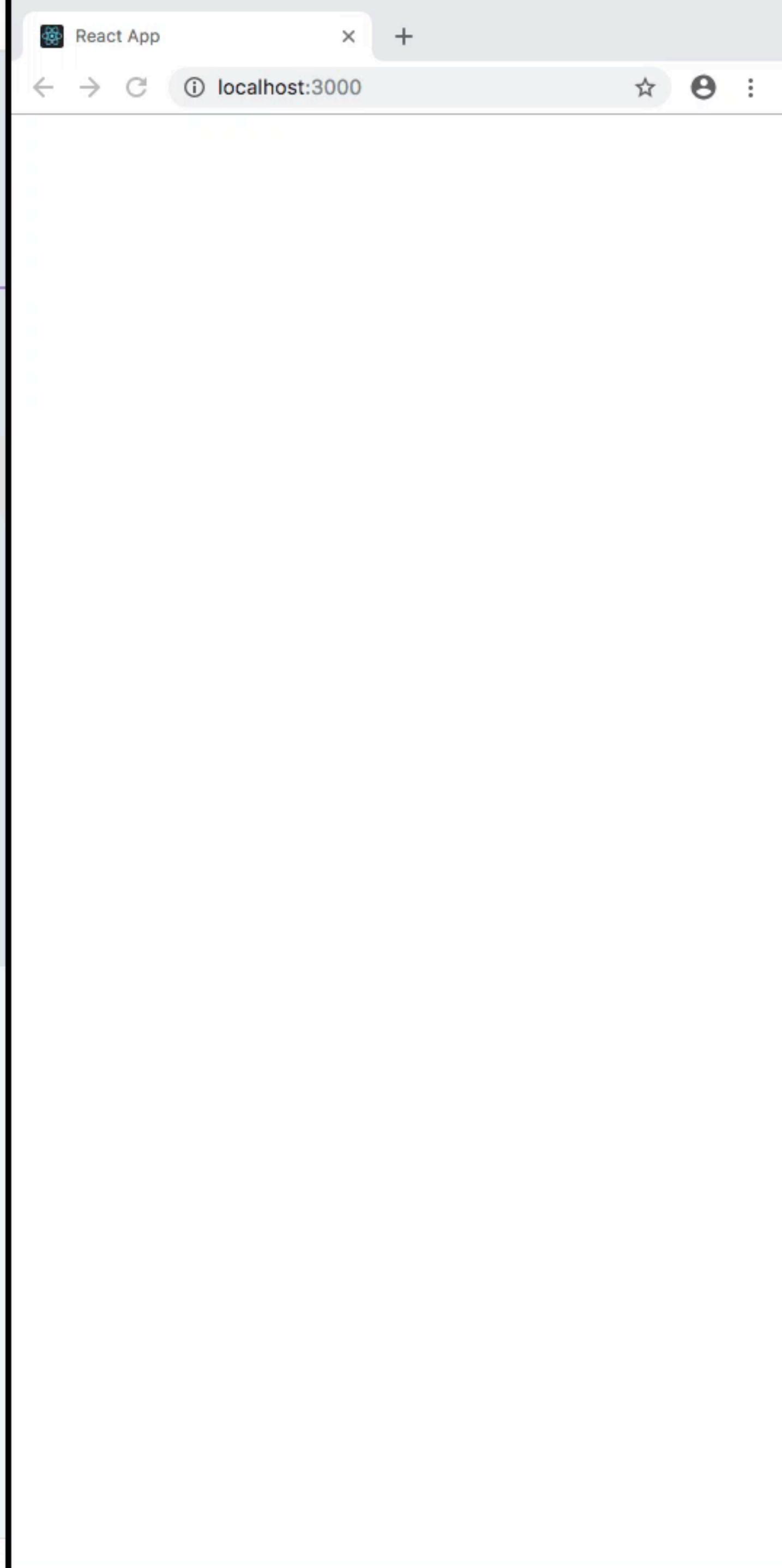


useReducer

JS index.js

```
1 import React from 'react';
2 import ReactDOM from 'react-dom';
3 import './index.css';
4
5 function Room() {
6
7 }
8
9 ReactDOM.render(
10   <Room />,
11   document.querySelector('#root')
12 );
13
```

Ln 6, Col 3 Spaces: 2 JavaScript Prettier: ✓



When to useReducer?

1. State depends on other state
2. Update logic is complex
3. You feel like it.

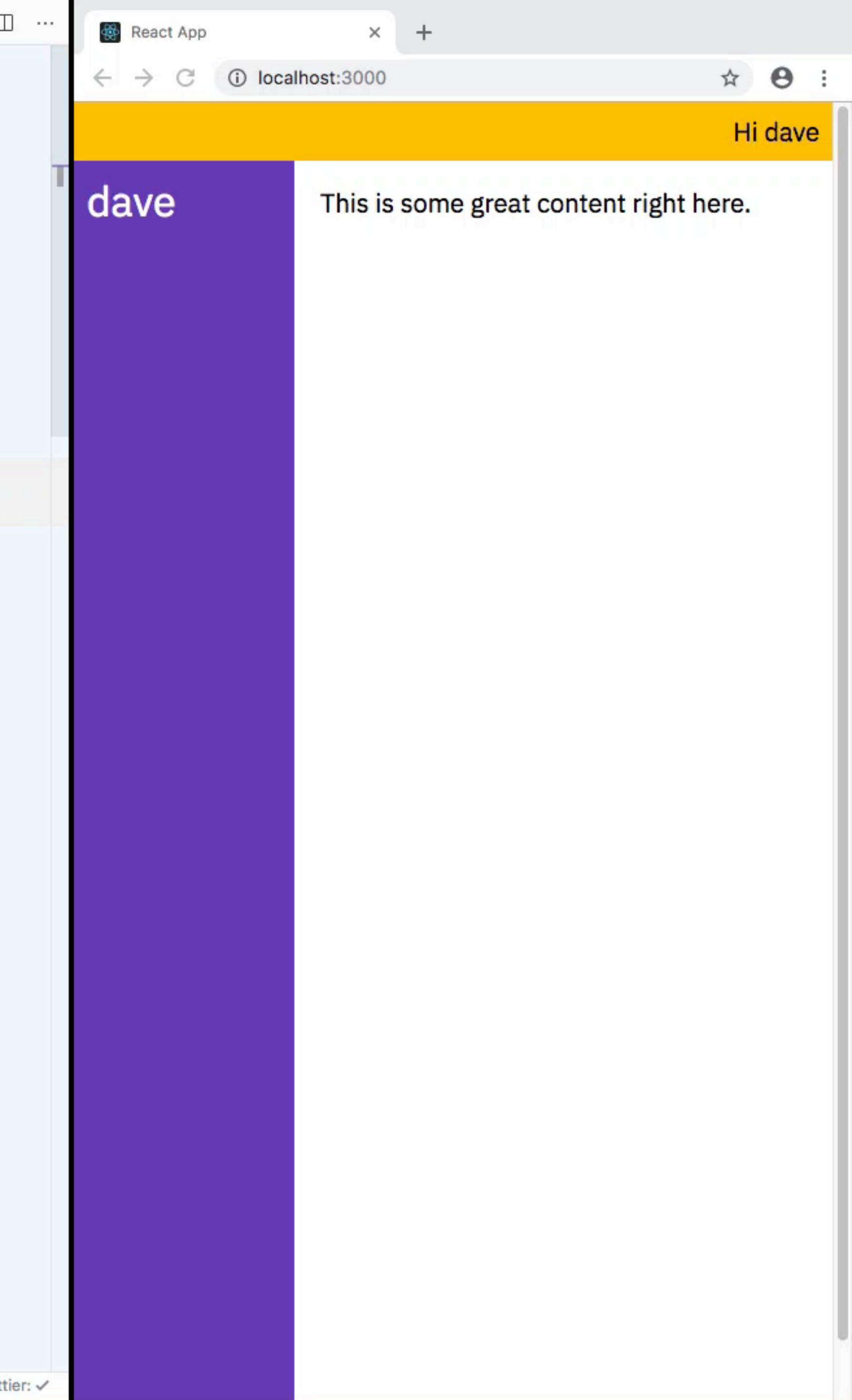
React Hooks



useContext

```
JS index.js x
1 import React, { useState } from 'react';
2 import ReactDOM from 'react-dom';
3 import './index.css';
4
5 function App() {
6   const [user, setUser] = useState({
7     username: 'dave'
8   });
9
10  return (
11    <>
12      <Header user={user} />
13      <Body user={user} />
14    </>
15  );
16}
17
18 function Header({ user }) {
19  return <header>Hi {user.username}</header>;

```



The image shows a code editor on the left and a browser window on the right, demonstrating the use of the `UserContext.Provider` component in React.

index.js (Code Editor):

```
JS index.js
10 });
11
12 return (
13   <UserContext.Provider value={user}>
14     <>
15       <Header user={user} />
16       <Body />
17     </>
18   </UserContext.Provider>
19 );
20 }
21
22 function Header({ user }) {
23   return <header>Hi {user.username}</header>;
24 }
25
26 function Body() {
27   return (
28     <main>
29       <Sidebar />
```

Browser Output:

The browser window displays the rendered React application at `localhost:3000`. The header shows "Hi dave". The sidebar displays "dave" and the main content area displays "This is some great content right here."

Can I replace Redux?

maybe! ↗

(it depends)

Just a few values? Simple ones?

like an auth token or whatever?



useContext

Huge bundle of app state?



useRedux*

* (**useSelector**, actually)

React Hooks



useEffect

useEffect:

componentDidMount

+

componentDidUpdate

+

componentWillUnmount

useEffect:

componentDidMount

componentDidUpdate

componentWillUnmount



The image shows a code editor and a web browser side-by-side. The code editor on the left contains the following code:

```
JS index.js
1 import React from 'react';
2 import ReactDOM from 'react-dom';
3 import './index.css';
4
5 const App = () => {
6
7 };
8
9 ReactDOM.render(
10   <App />,
11   document.querySelector('#root')
12 );
13
```

The browser window on the right displays the rendered output of the code at localhost:3000. The page is completely blank, showing only the default white background of the browser.

JS index.js

```
1 import React, { useState, useEffect } from 'react';
2 import ReactDOM from 'react-dom';
3
4 const Reddit = () => {
5
6 };
7
8 ReactDOM.render(
9   <Reddit />,
10  document.querySelector('#root')
11 );
12
```

React App

localhost:3000

```
JS index.js x
1 import React, { useState, useEffect } from 'react';
2 import ReactDOM from 'react-dom';
3 import './index.css';
4
5 const App = () => {
6   const [title, setTitle] = useState('');
7
8   useEffect(() => {
9     document.title = title;
10   });
11
12   return (
13     <input
14       value={title}
15       onChange={e => setTitle(e.target.value)}
16     />
17   );
18 };
19
```

Hello Boston

localhost:3000

Ln 6, Col 42 Spaces: 2 JavaScript Prettier: ✓

mount!

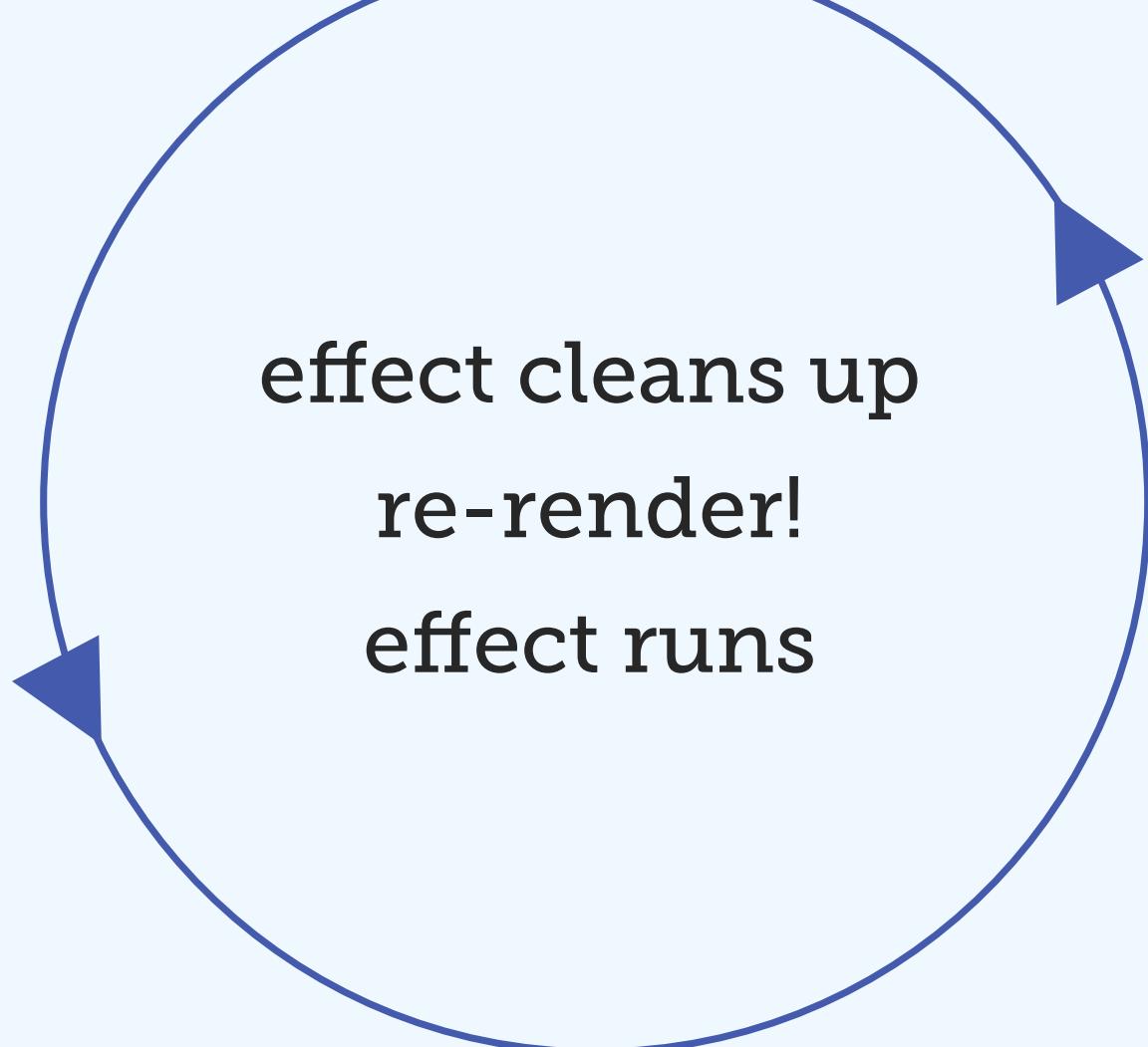
effect runs

⋮

effect cleans up

re-render!

effect runs



⋮

effect cleans up

un-mount!

```
useEffect(() => {  
    // set up  
  
    return () => {  
        // clean up  
    }  
})
```

```
useEffect(() => {
  const timer = setTimeout(5000)

  return () => {
    clearTimeout(timer)
  }
})
```

if-this-then-that

WHEN
~~if-~~**this-then-that**

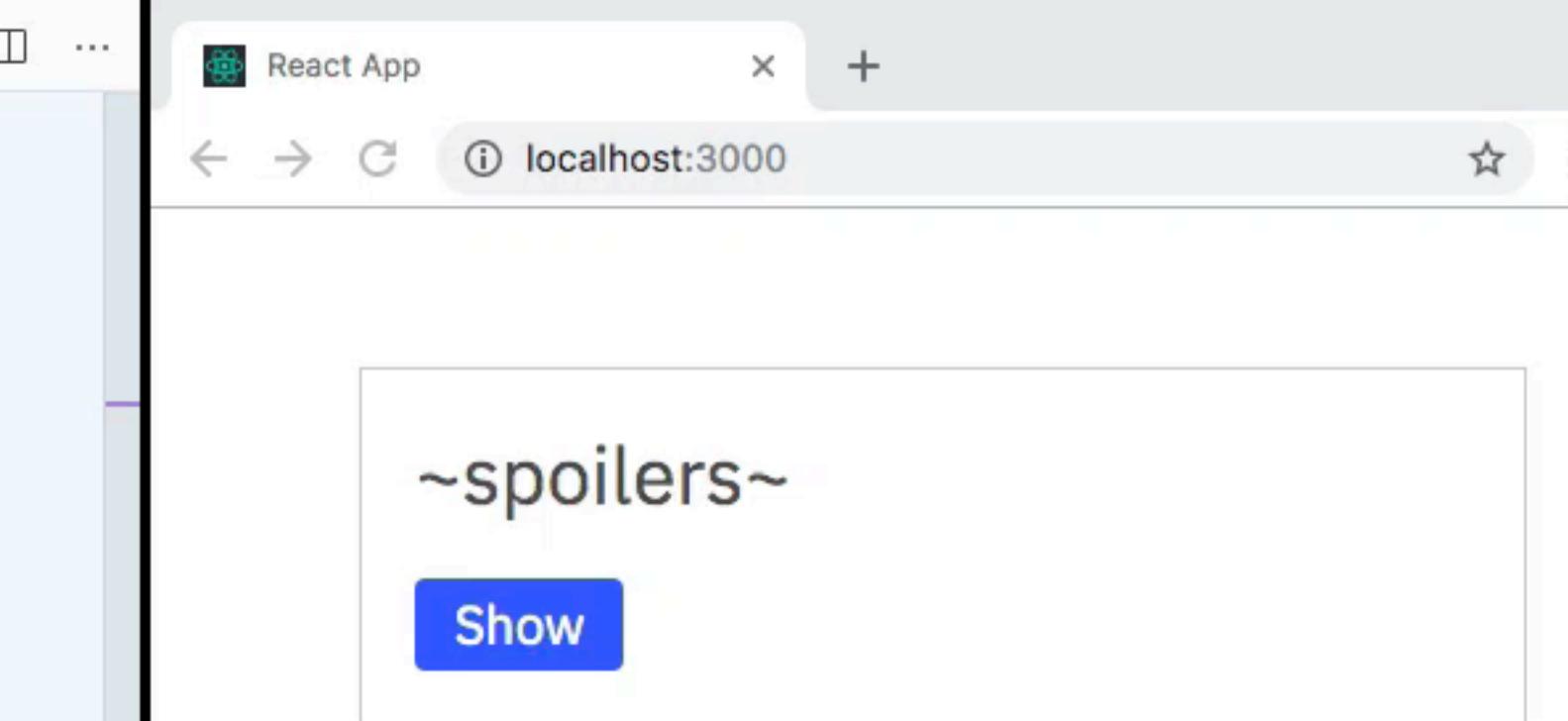
when postId changes, then fetch comments

```
useEffect( () => {  
  fetchComments(postId)  
, [postId] )
```

Custom Hooks



```
JS index.js x
1 import React, { useState } from 'react';
2 import ReactDOM from 'react-dom';
3 import './index.css';
4
5 function SpoilerAlert({ text }) {
6   const [isVisible, setVisible] = useState(false);
7
8   return (
9     <div>
10    {isVisible && <span>{text}</span>}
11    {!isVisible && (
12      <span className="hidden">~spoilers~</span>
13    )}
14    <button onClick={() => setVisible(!isVisible)}>
15      {isVisible ? 'Hide' : 'Show'}
16    </button>
17  </div>
18);
19}
```



A Few Custom Hook Ideas

useLocalStorage

useAuth

useFetch

useAxios

useLocation

useInterval

<https://nikgraf.github.io/react-hooks/>

useAudio

useApolloClient

useArray

useHistory

<https://github.com/rehooks/awesome-react-hooks>

More Hooks...

useMemo - memoize expensive computations

useCallback - memoize callbacks

useRef - create refs to DOM nodes

useLayoutEffect - like useEffect, but runs before paint

useImperativeHandle

useDebugValue - label custom hooks in DevTools

Resources

Official Docs

<https://reactjs.org/hooks>

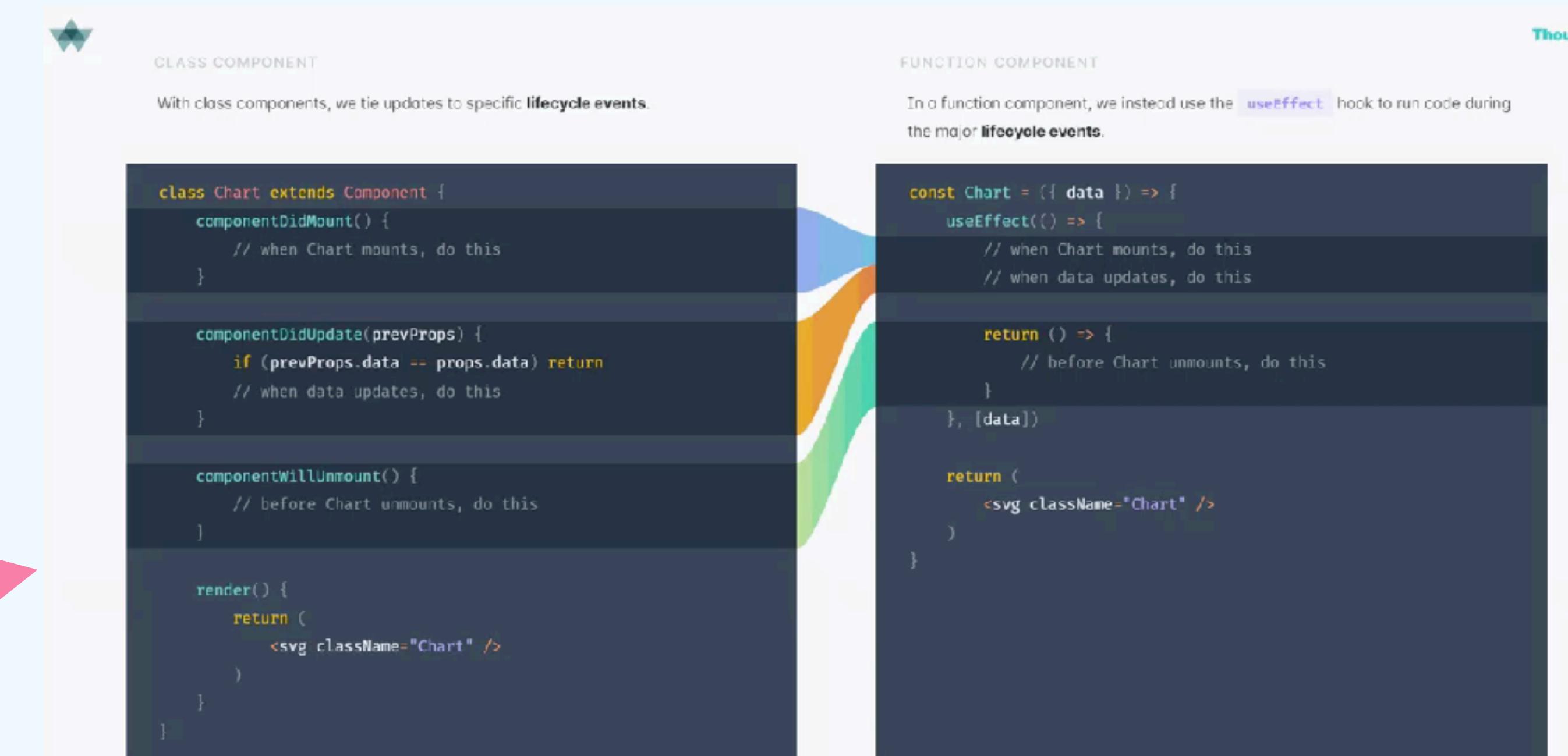
Hooks Week!

<https://daveceddia.com/hooks>

Thinking in React Hooks →

Amelia Wattenberger

<https://wattenberger.com/blog/react-hooks/>



Thanks!

Examples and slides:

<https://daveceddia.com/boston>



@dceddia

daveceddia.com