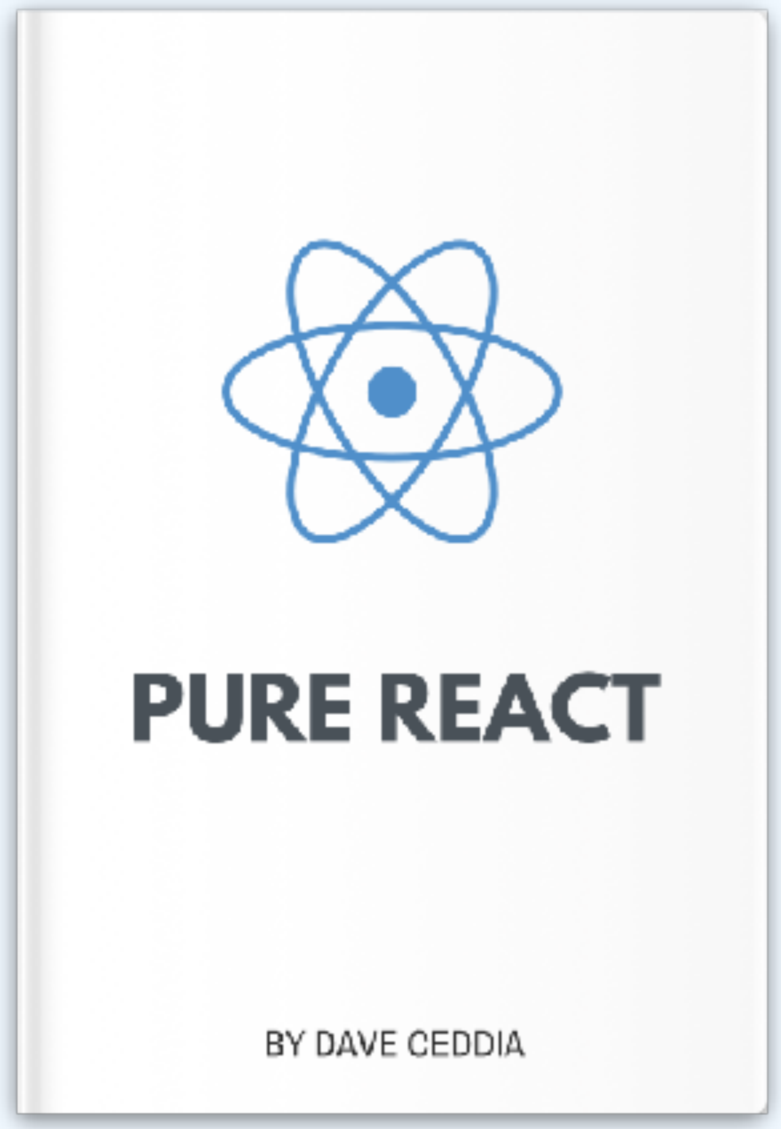
React **Hooks**

a guided tour

Dave

daveceddia.com

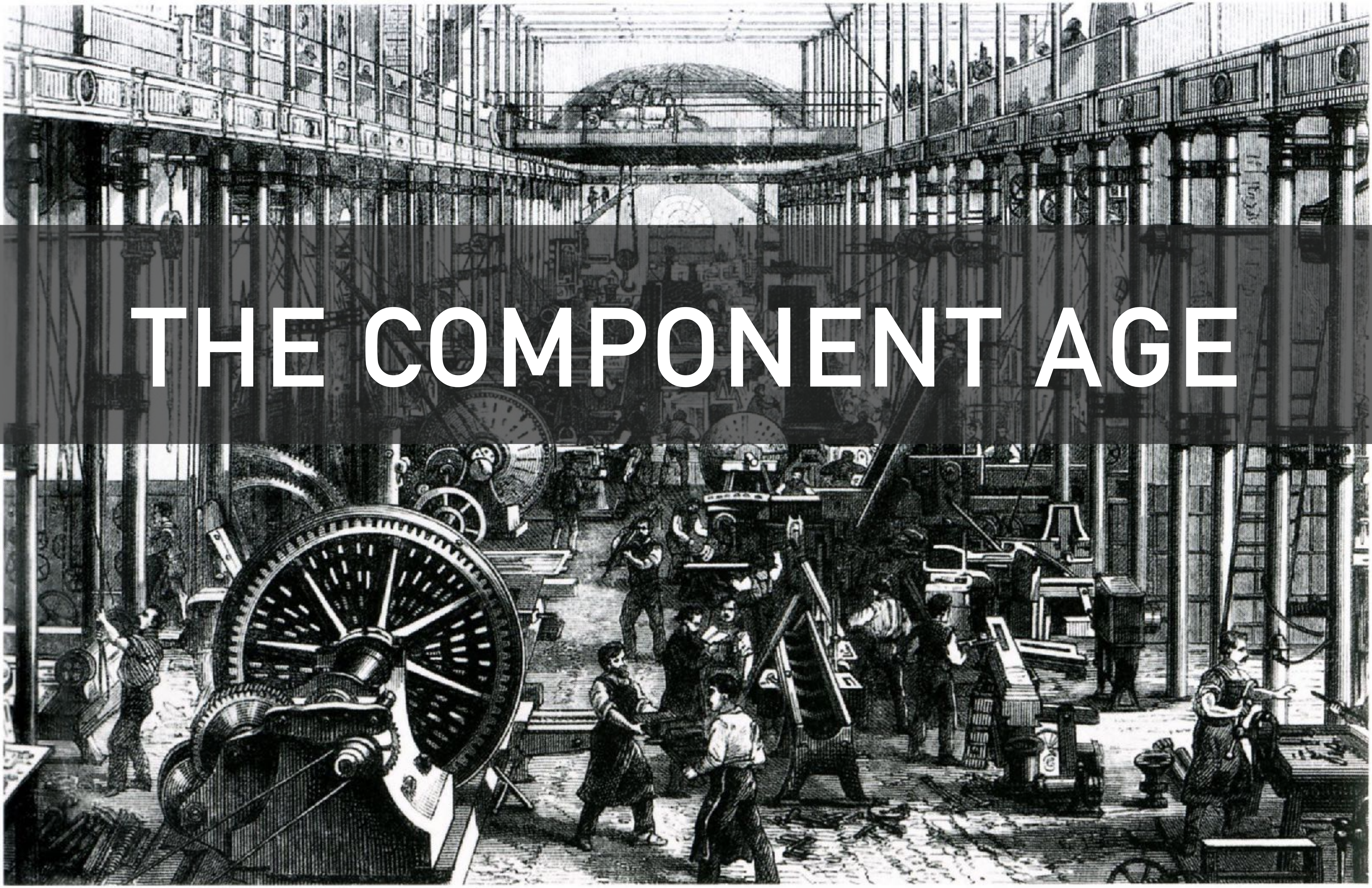
@dceddia



```
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>
    );
  }
}
```

```jsx
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
    });
  });
}
```

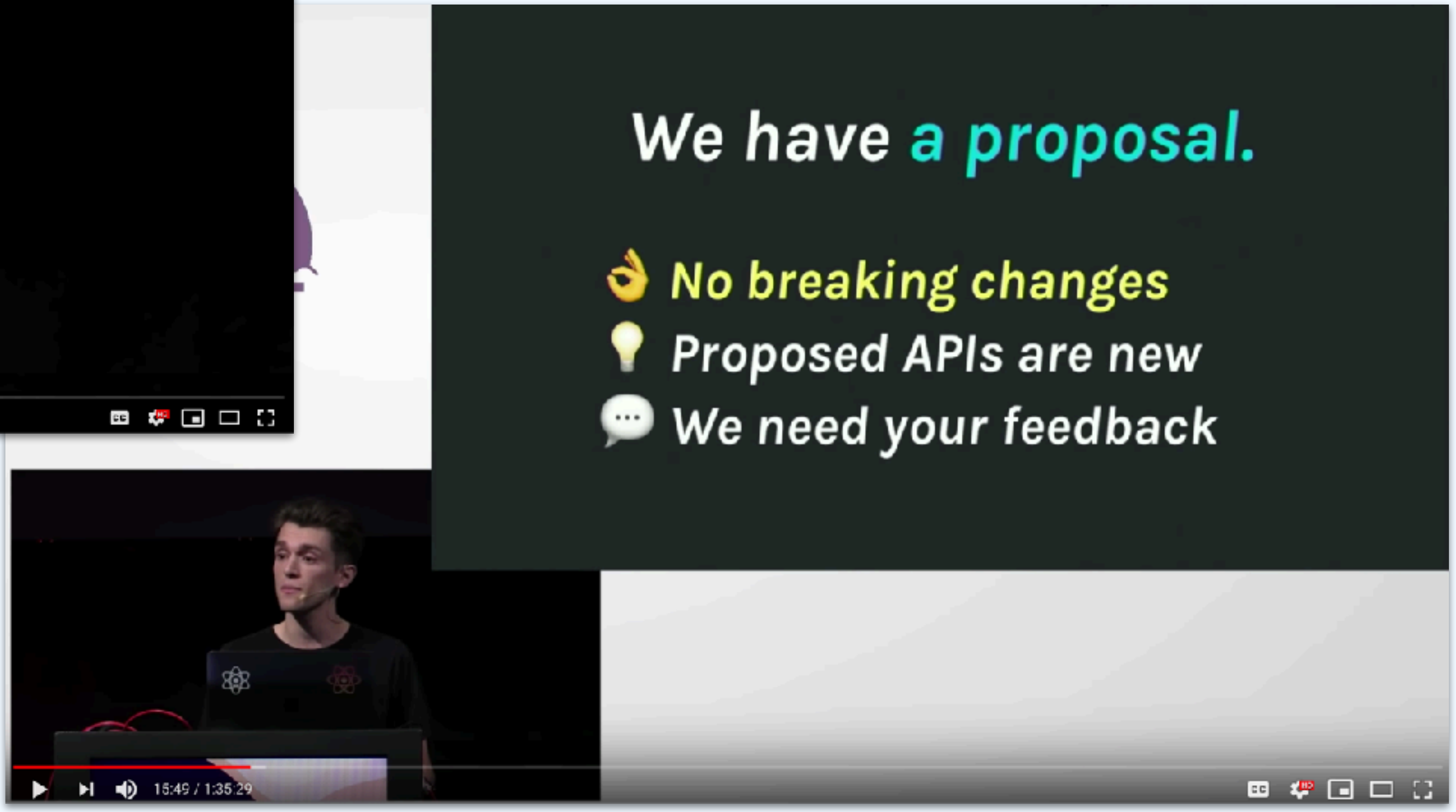
```
componentDidMount() {
  this.updateComments(this.props.postId);
}

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
    });
  });
}
```

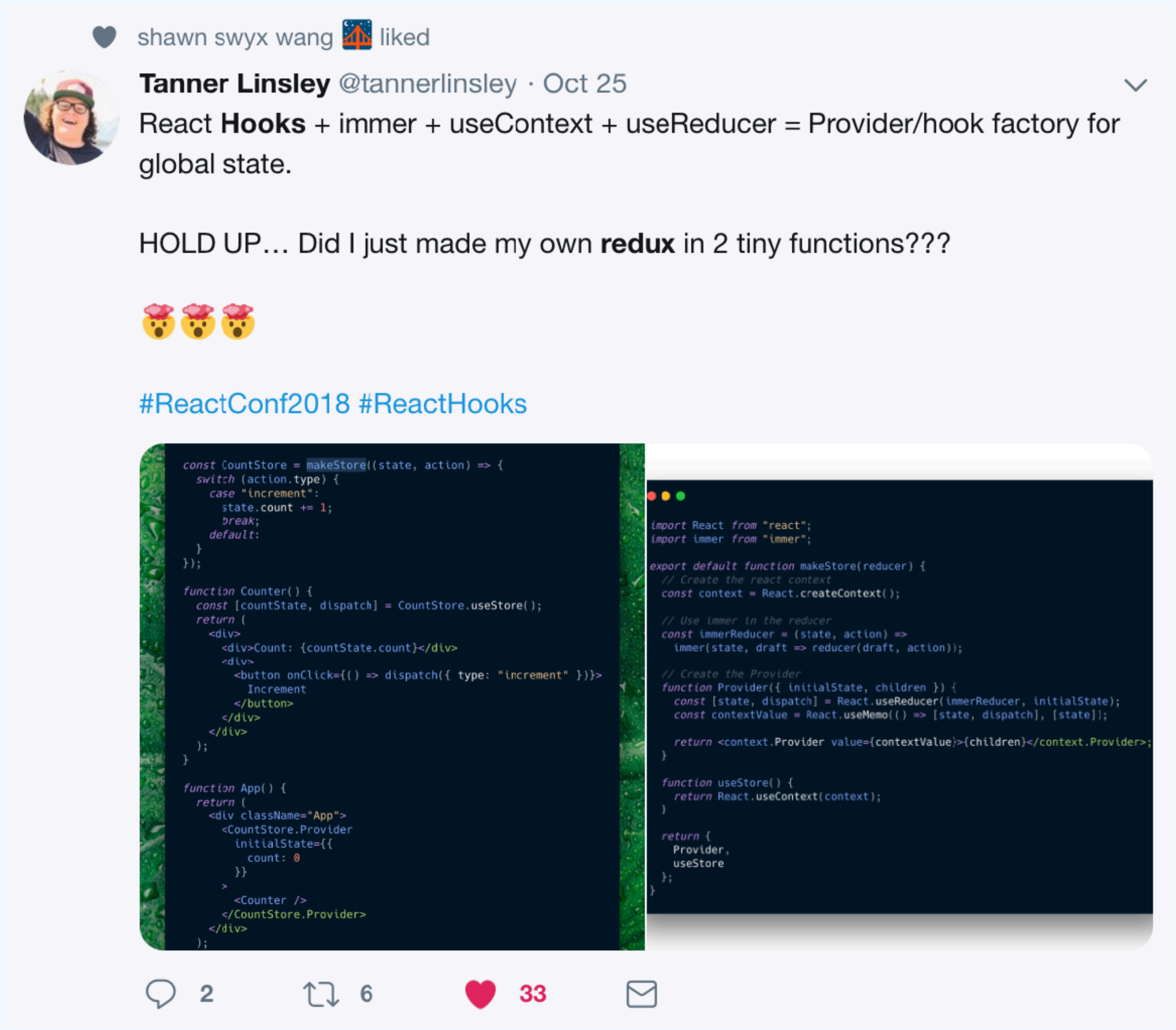
```
componentDidMount() {
  this.updateComments(this.props.postId);
}

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

THIS IS FINE.



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???

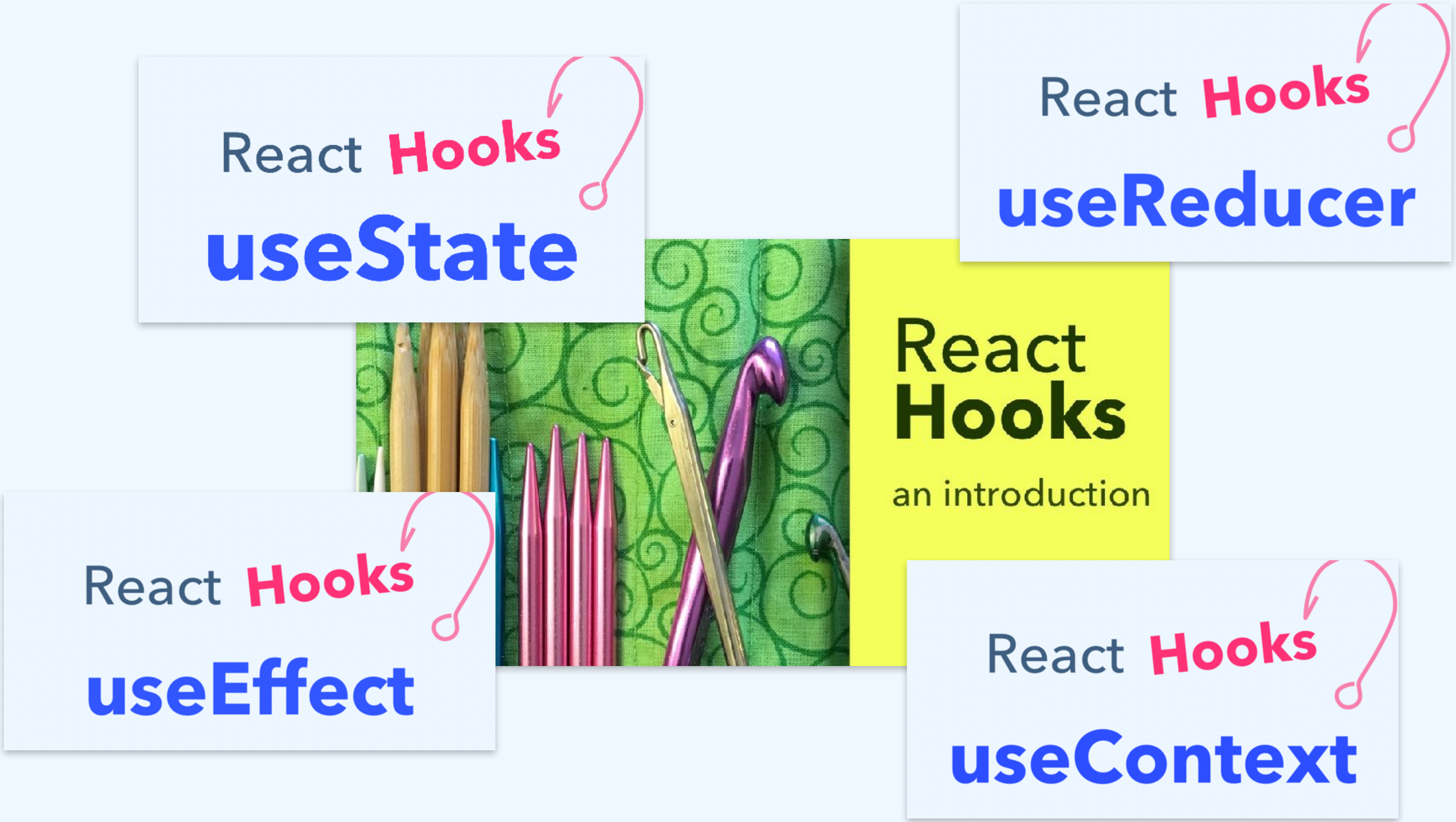
🤯🤯🤯

#ReactConf2018 #ReactHooks

```
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
  };
}
```

Hooks Week™

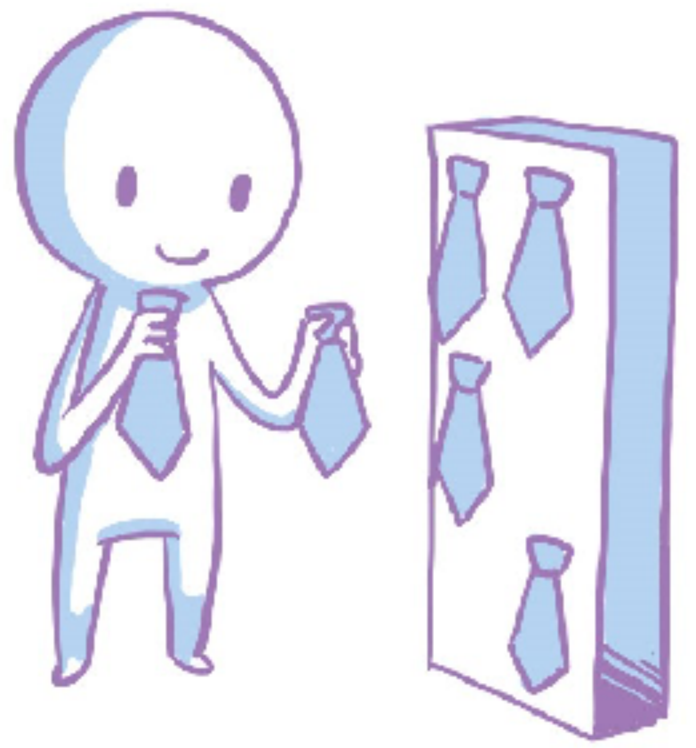

React Hooks
useState

React Hooks
useReducer

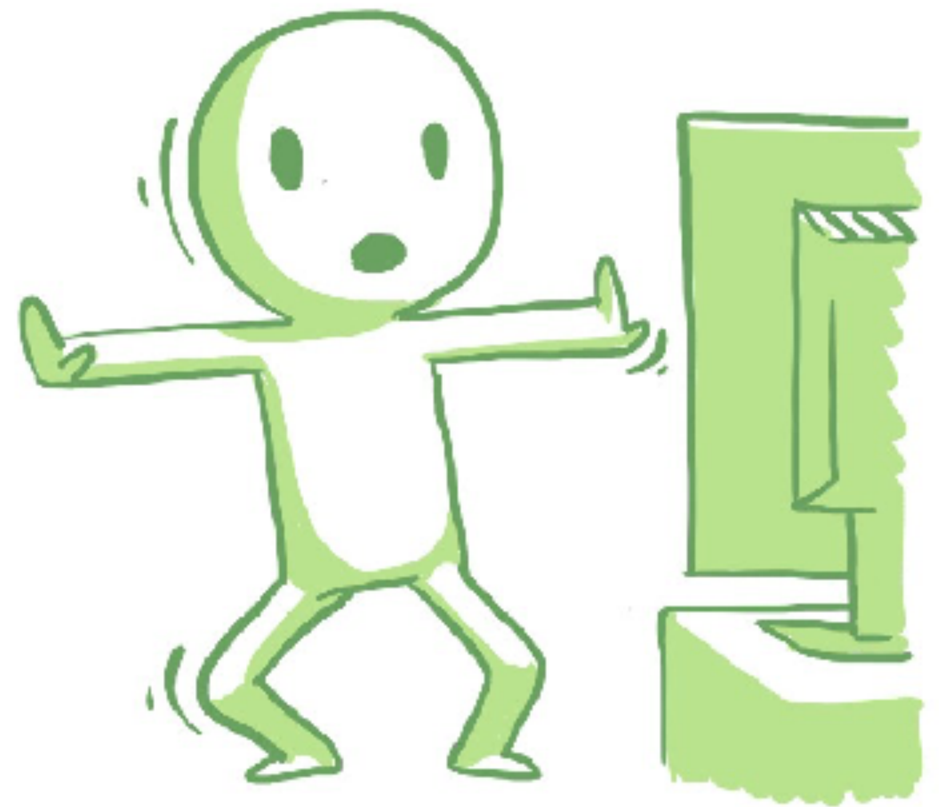
React
Hooks
an introduction

React Hooks
useEffect

React Hooks
useContext


SHOP FOR **hooks**

MAKE **hooks**

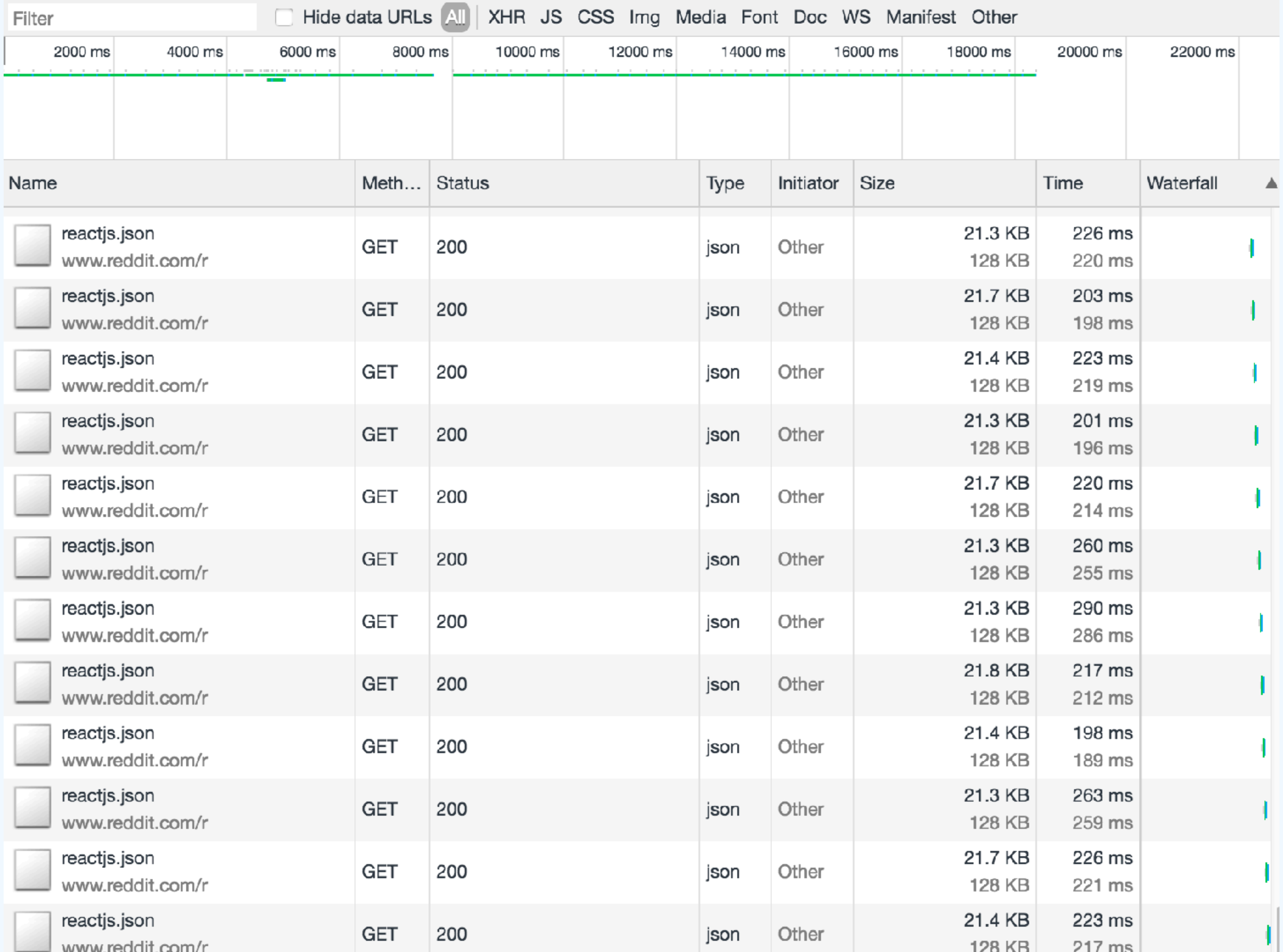

DO **hooks**

DON'T LET THE EXISTENTIAL DREAD SET IN.

DON'T LET IT SET IN.

VACUUM THE **hooks**

DON'T LET IT SET IN.



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.

index.js
React App
localhost:3000
⊗ 0 ⚠ 0 -- INSERT --
Ln 8, Col 35 Spaces: 2 JavaScript Prettier: ✓

```javascript
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';

function App() {
  return (
    <div>
      <h1>Hello React Boston!</h1>
    </div>
  );
}

ReactDOM.render(
  <App />,
  document.querySelector('#root')
);

```

```
Hello React
Boston!
```

index.js
React App
localhost:3000
-- INSERT --
Ln 7, Col 10
Spaces: 2
JavaScript
Prettier: ✓

```
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';

function App() {
  return (
    <div>
      <h1>Hello React Boston!</h1>
    </div>
  );
}

ReactDOM.render(
  <App />,
  document.querySelector('#root')
);
```

Hello React Boston!

How does that even work?

```
import React, { useState } from 'react';
import ReactDOM from 'react-dom';
import './index.css';

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>
  );
}

ReactDOM.render(
```

index.js
-- INSERT --
Ln 6, Col 45
Spaces: 2
JavaScript
Prettier: ✓
React App
localhost:3000

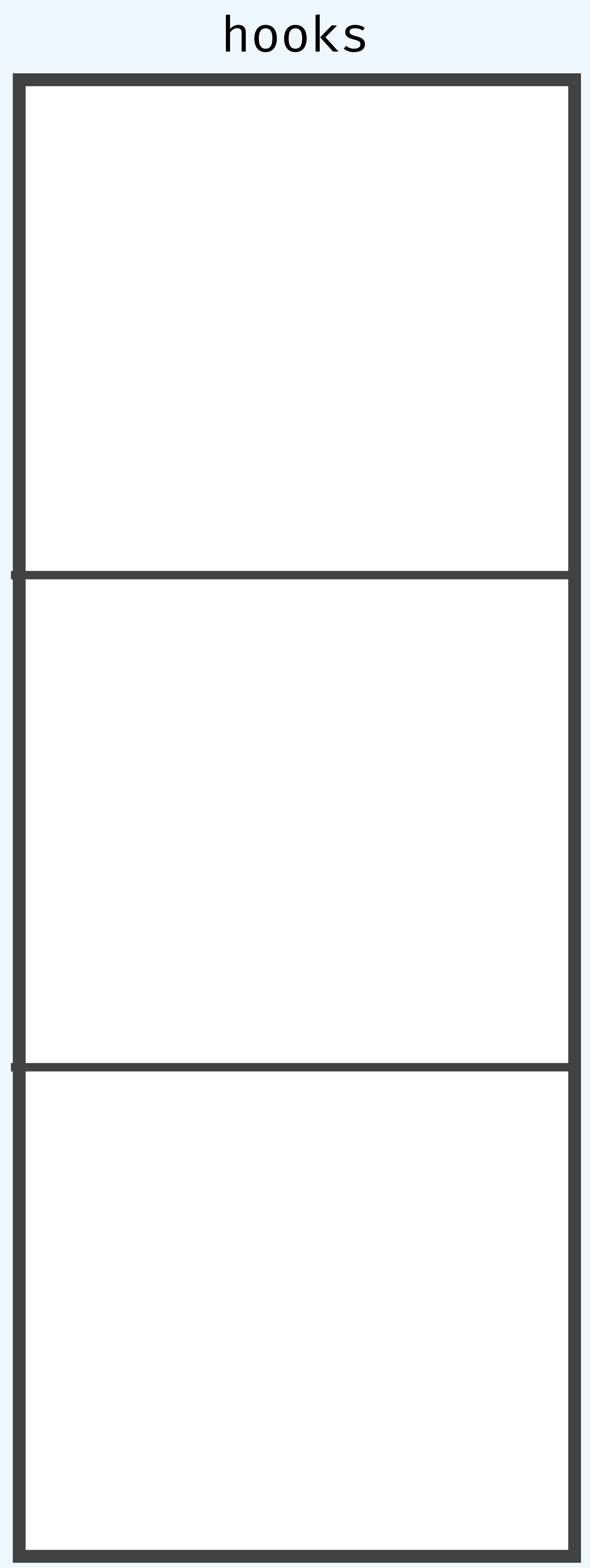
Hello React Boston!

People here today: 300

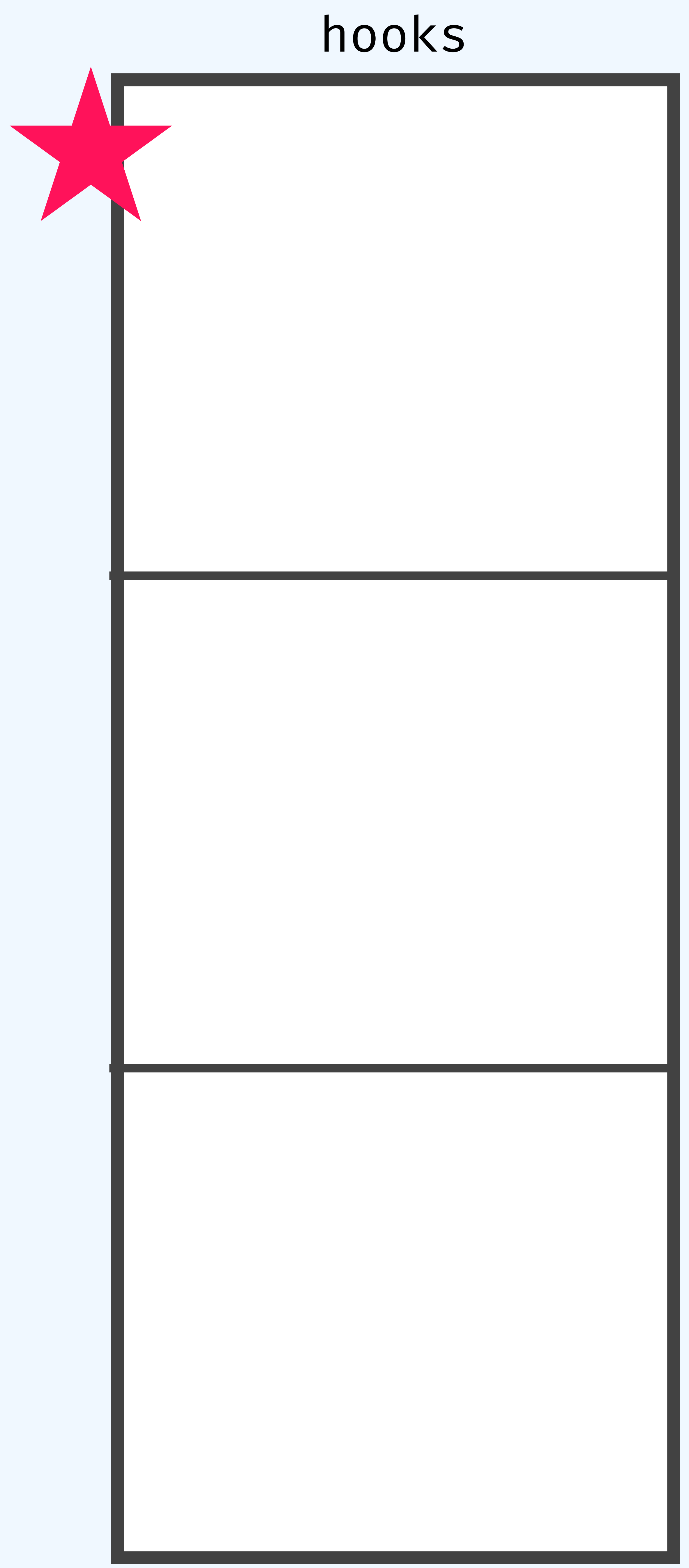
Welcome One More

```
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>
  );
}
```

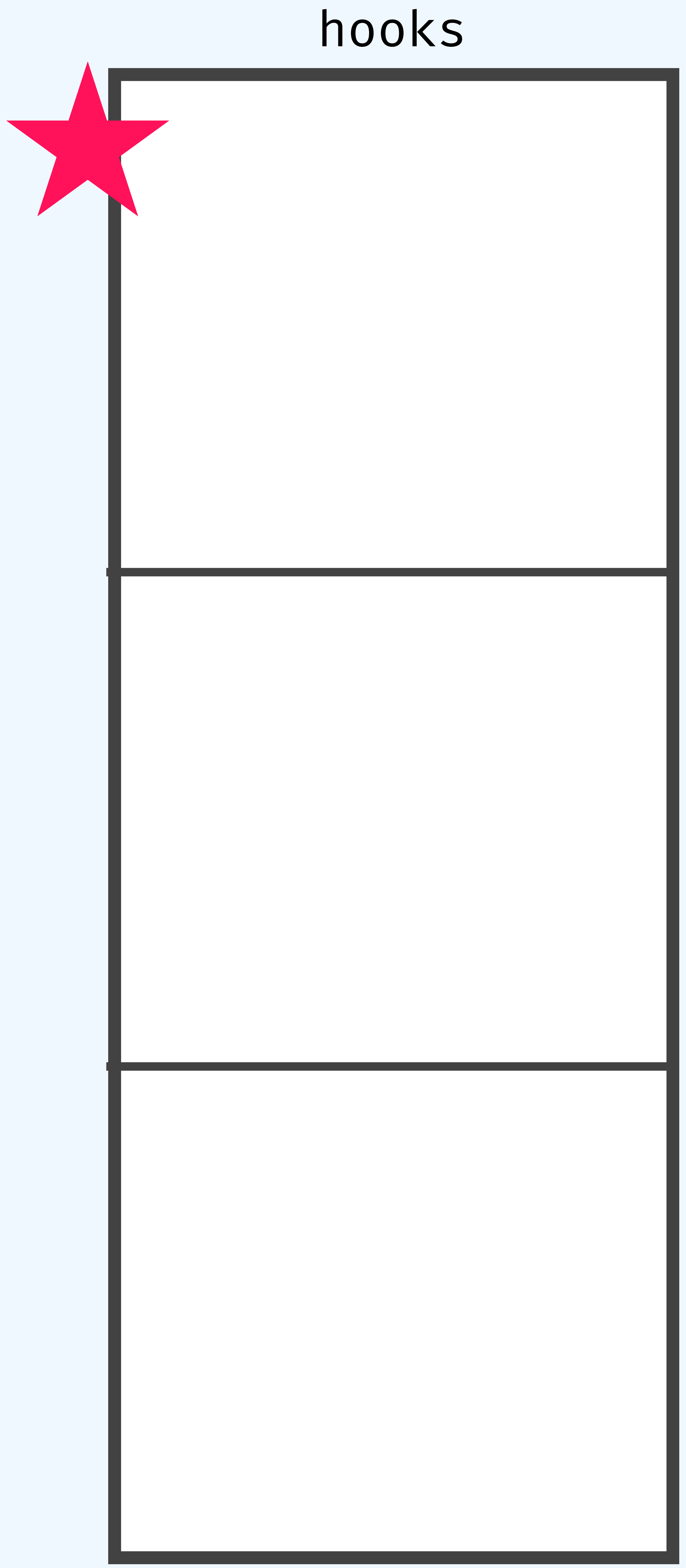
```
function App() {
  const [people, setPeople] = useState(300);

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



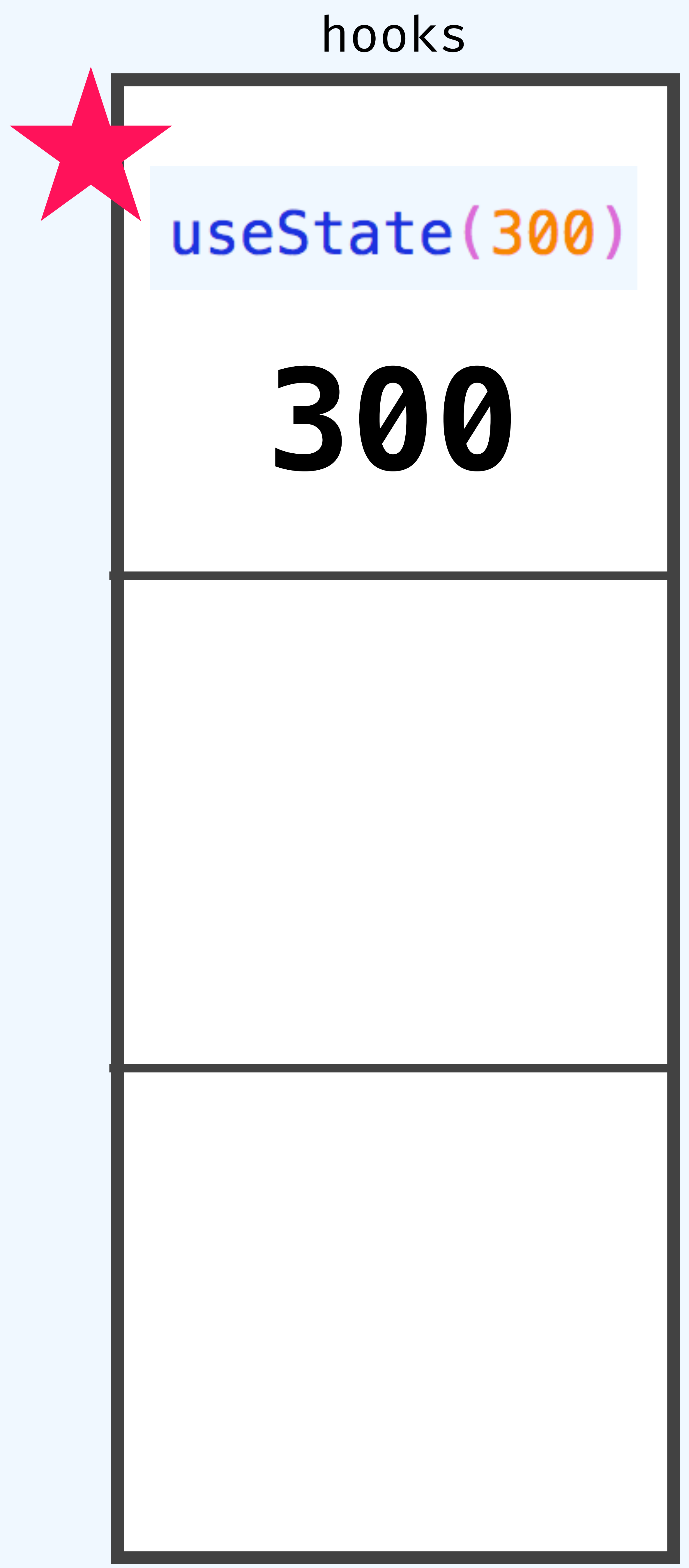
```
function App() {
  const [people, setPeople] = useState(300);

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



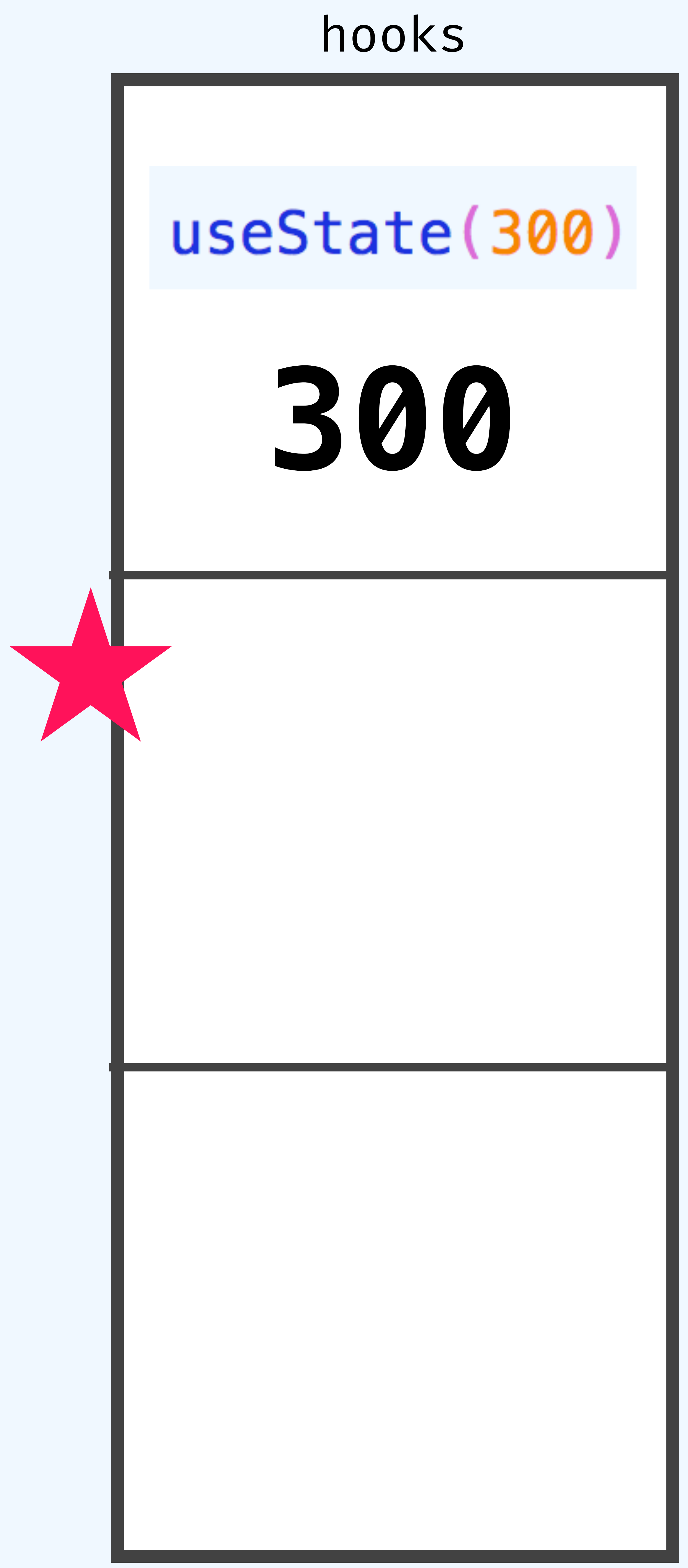
```
function App() {
  const [people, setPeople] = useState(300);

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



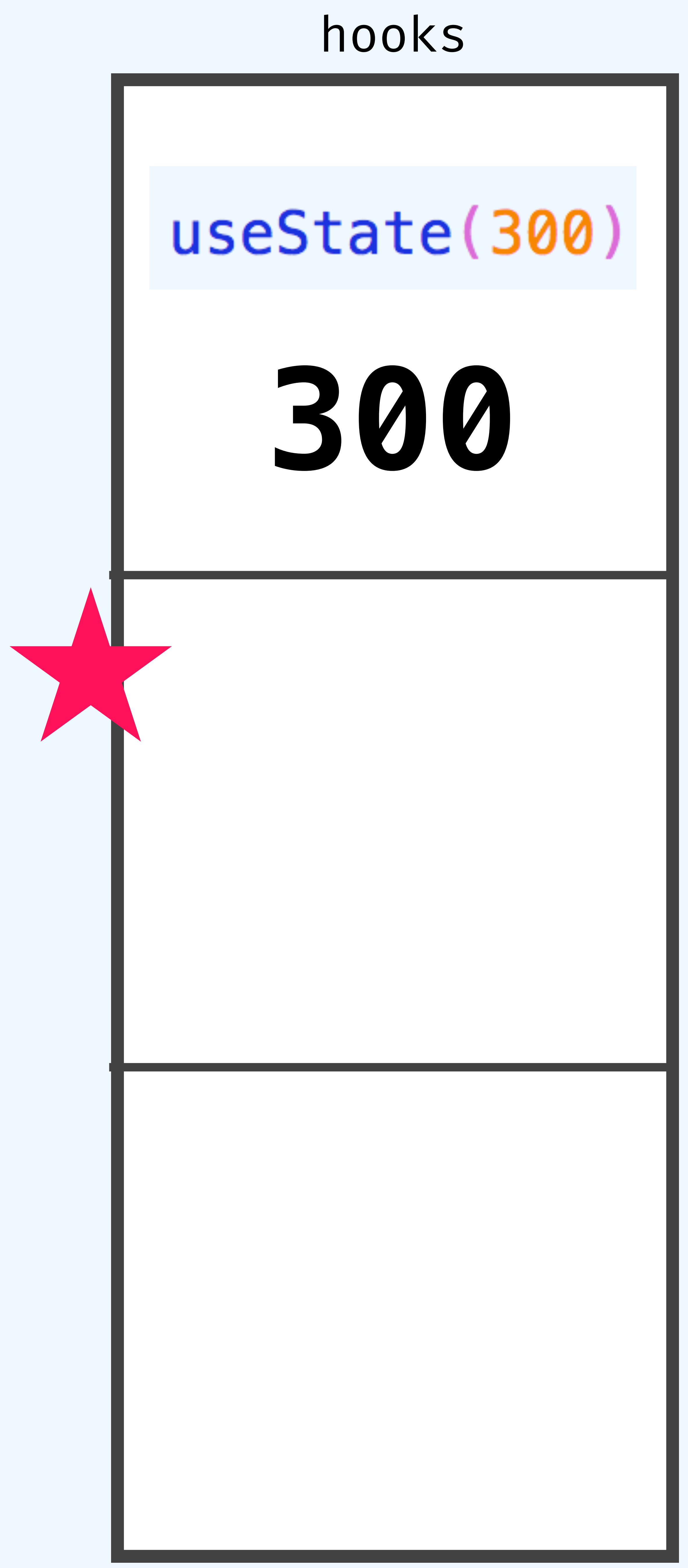
```jsx
function App() {
  const [people, setPeople] = useState(300);

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

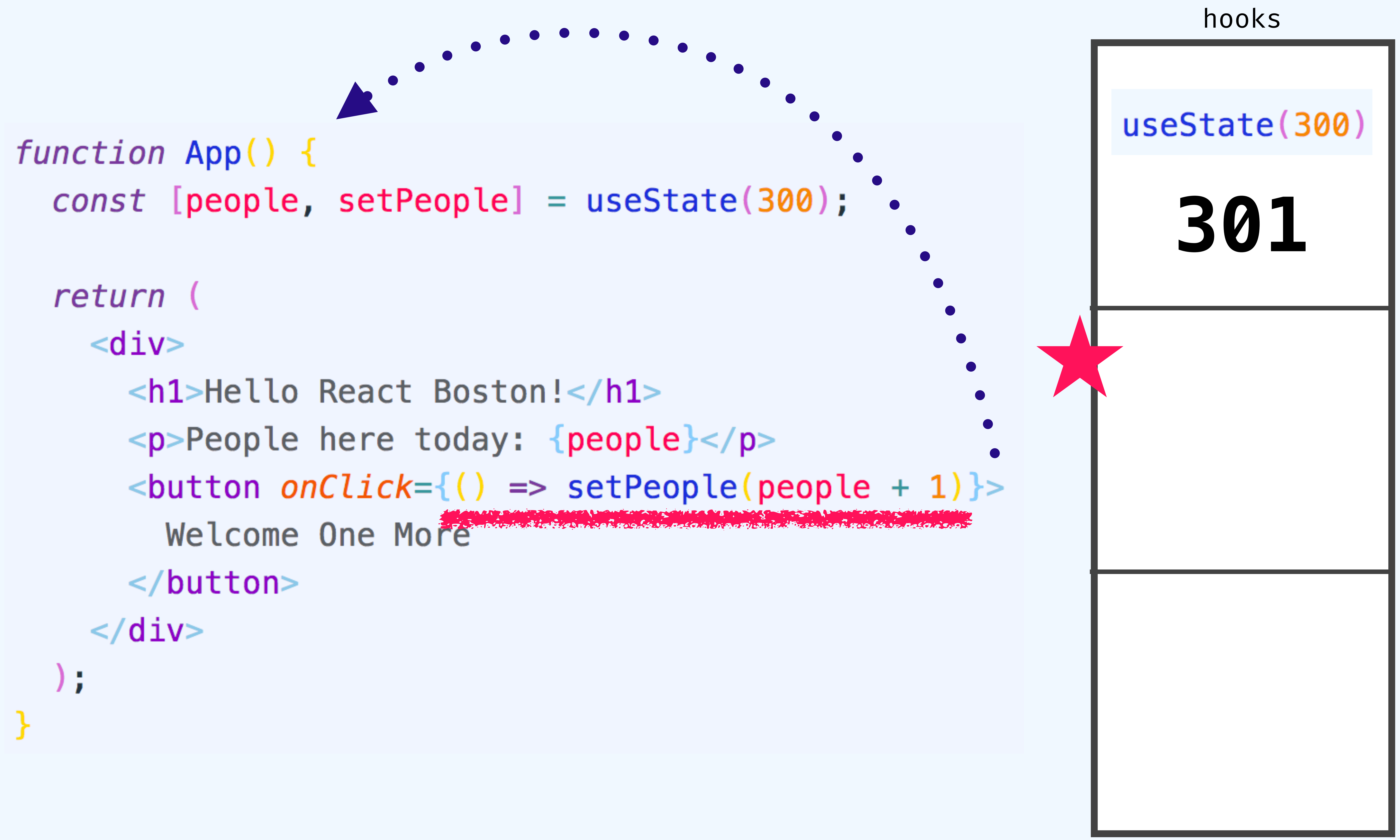


```
function App() {
  const [people, setPeople] = useState(300);

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

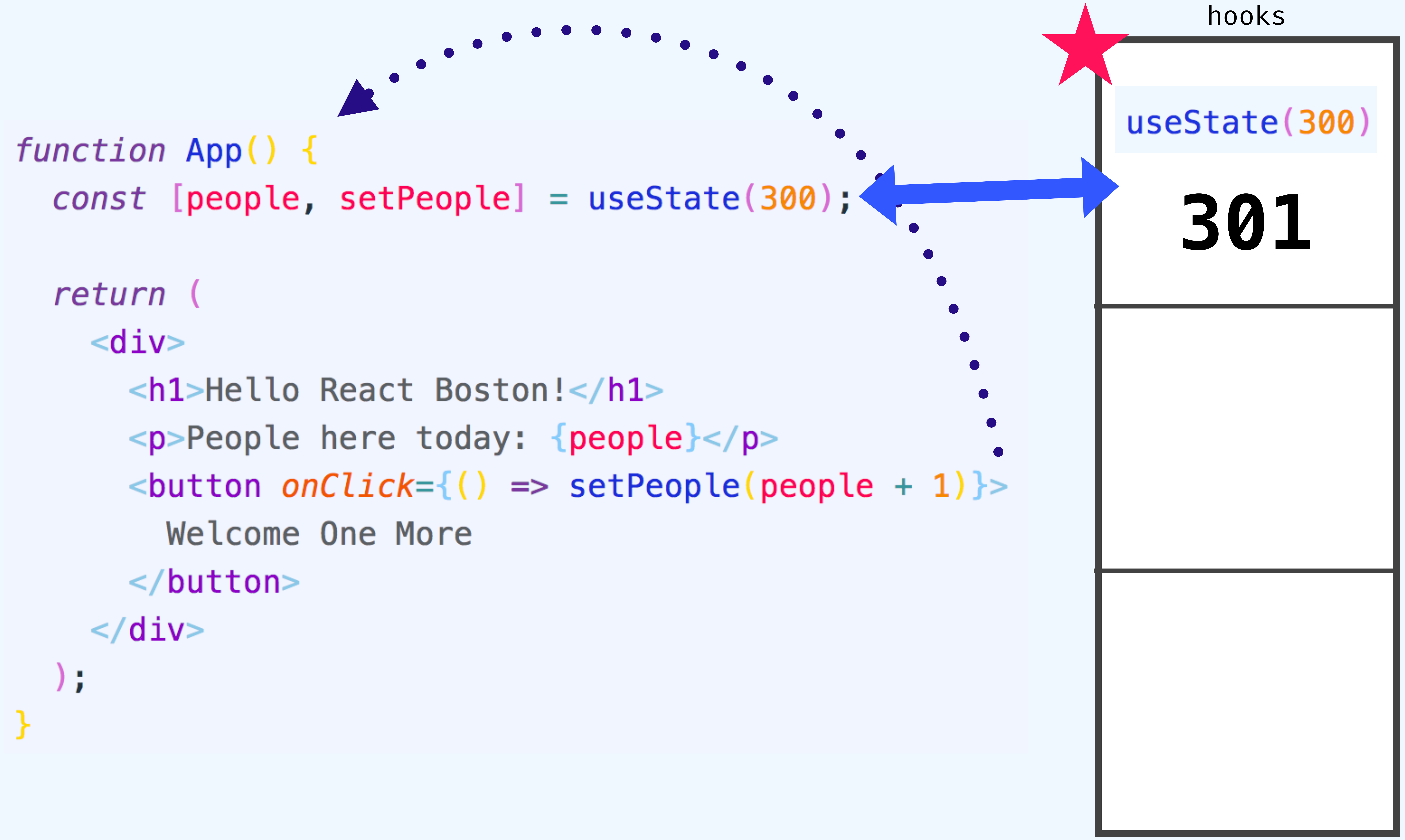


```jsx
function App() {
  const [people, setPeople] = useState(300);

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



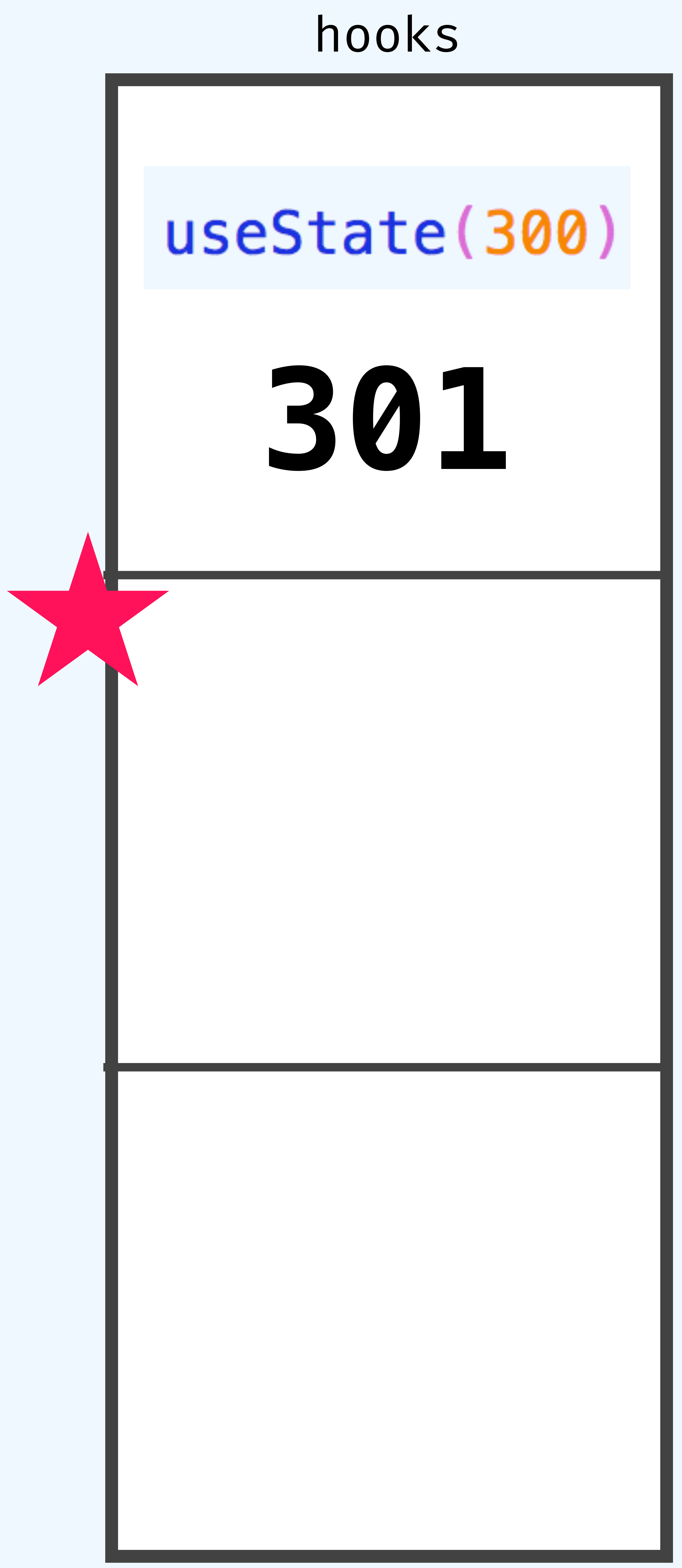


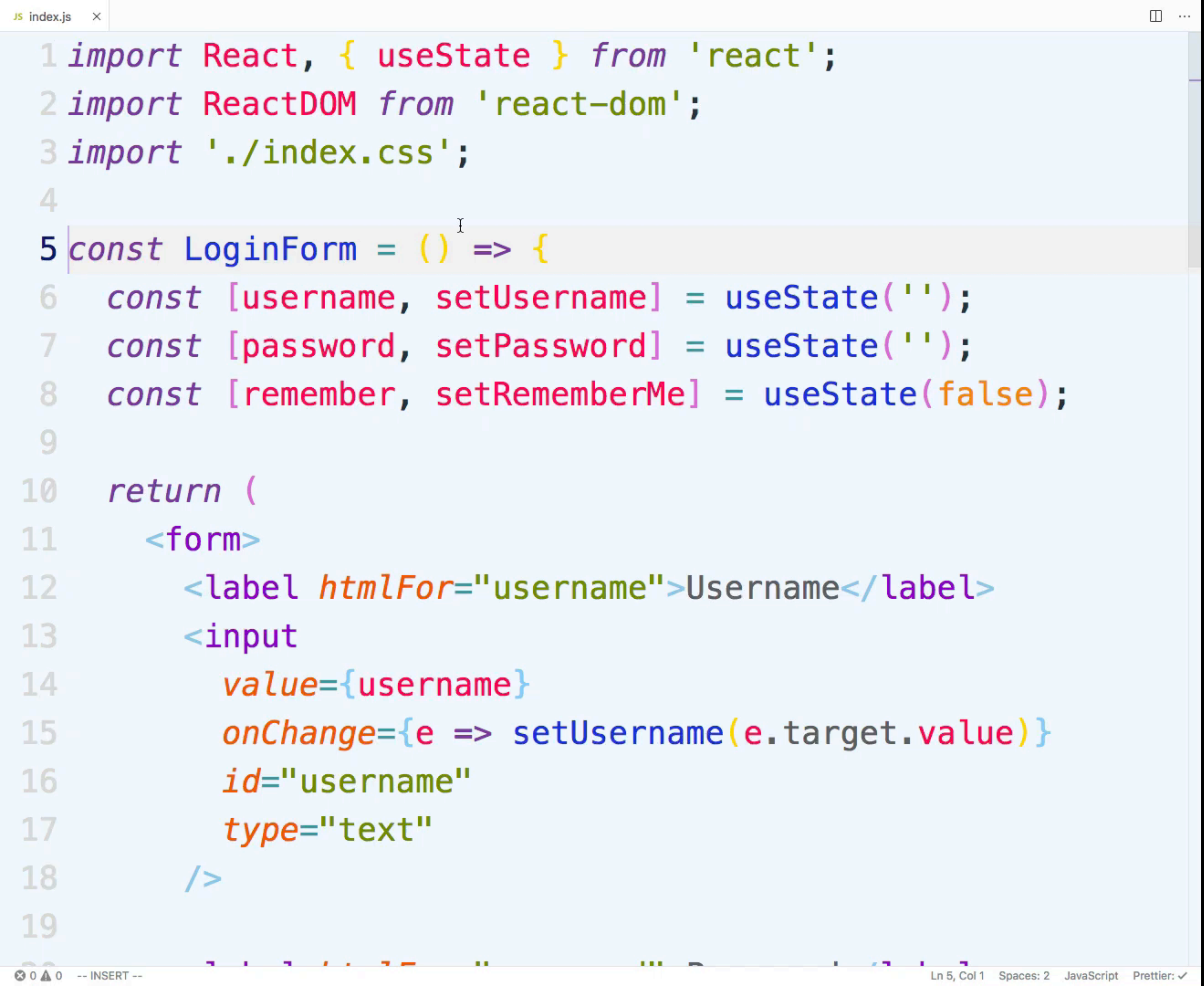
```
function App() {
  const [people, setPeople] = useState(300);

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



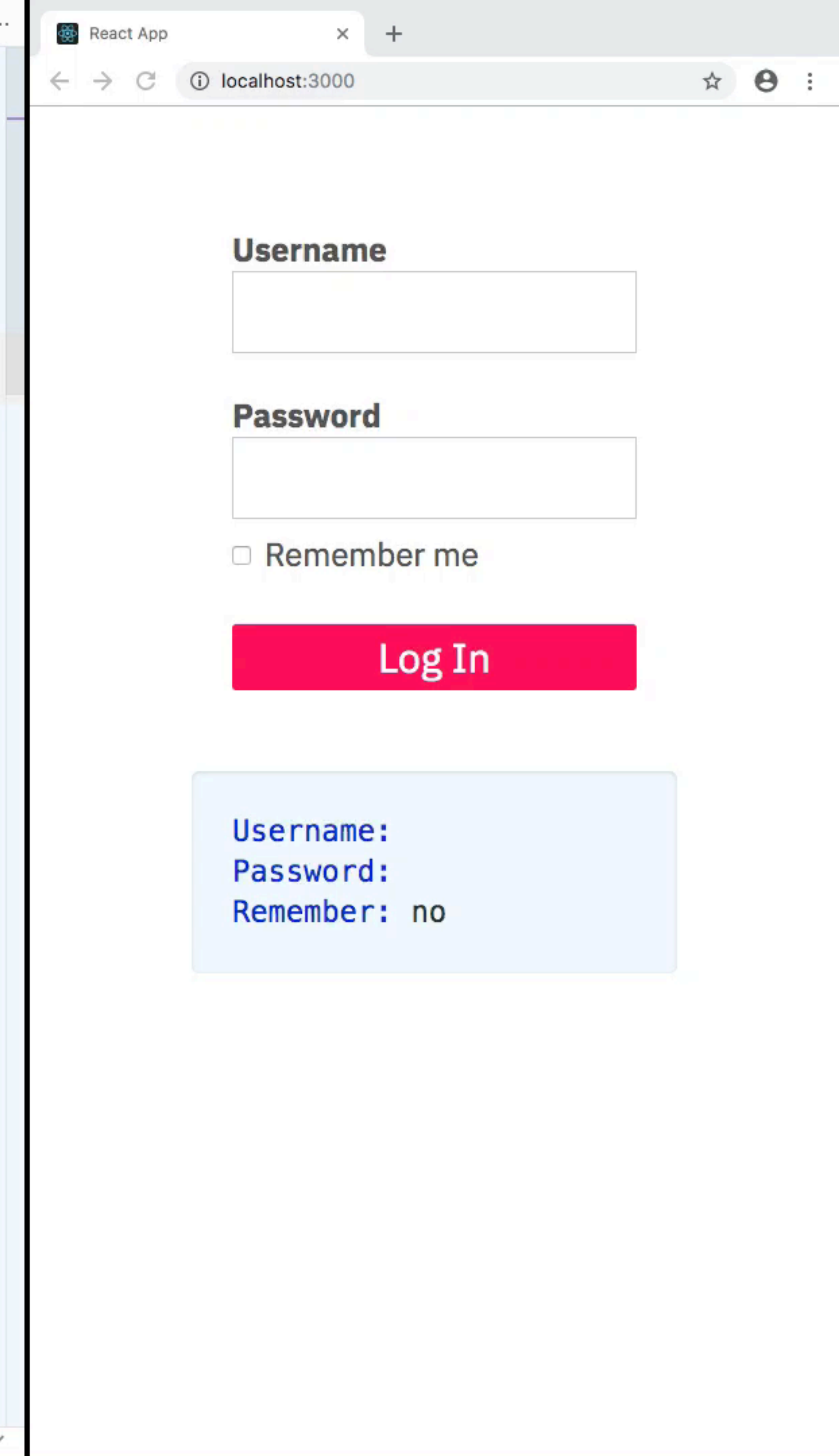
```
function App() {
  const [people, setPeople] = useState(300);

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

```
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>
  );
}
```



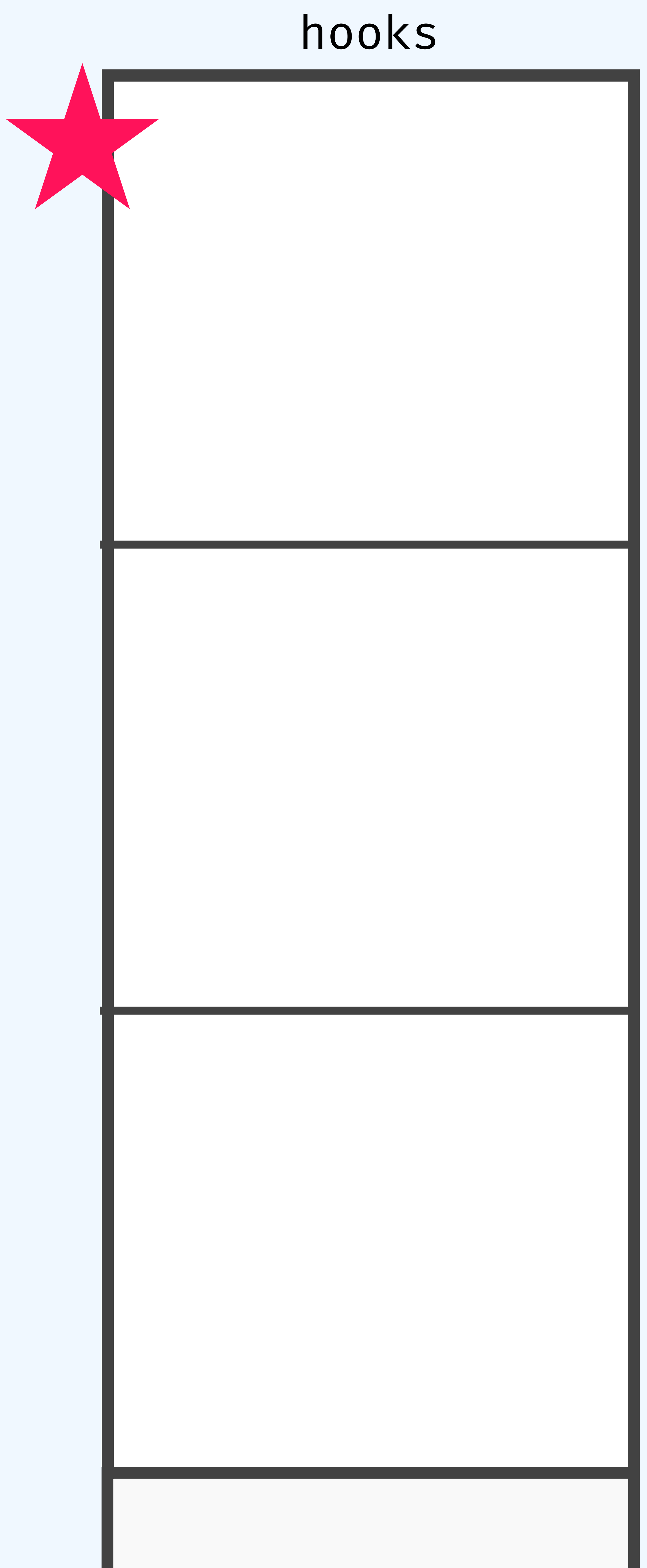


```javascript
import React, { useState } from 'react';
import ReactDOM from 'react-dom';
import './index.css';

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"
      />

      <label htmlFor="password">Password</label>
```

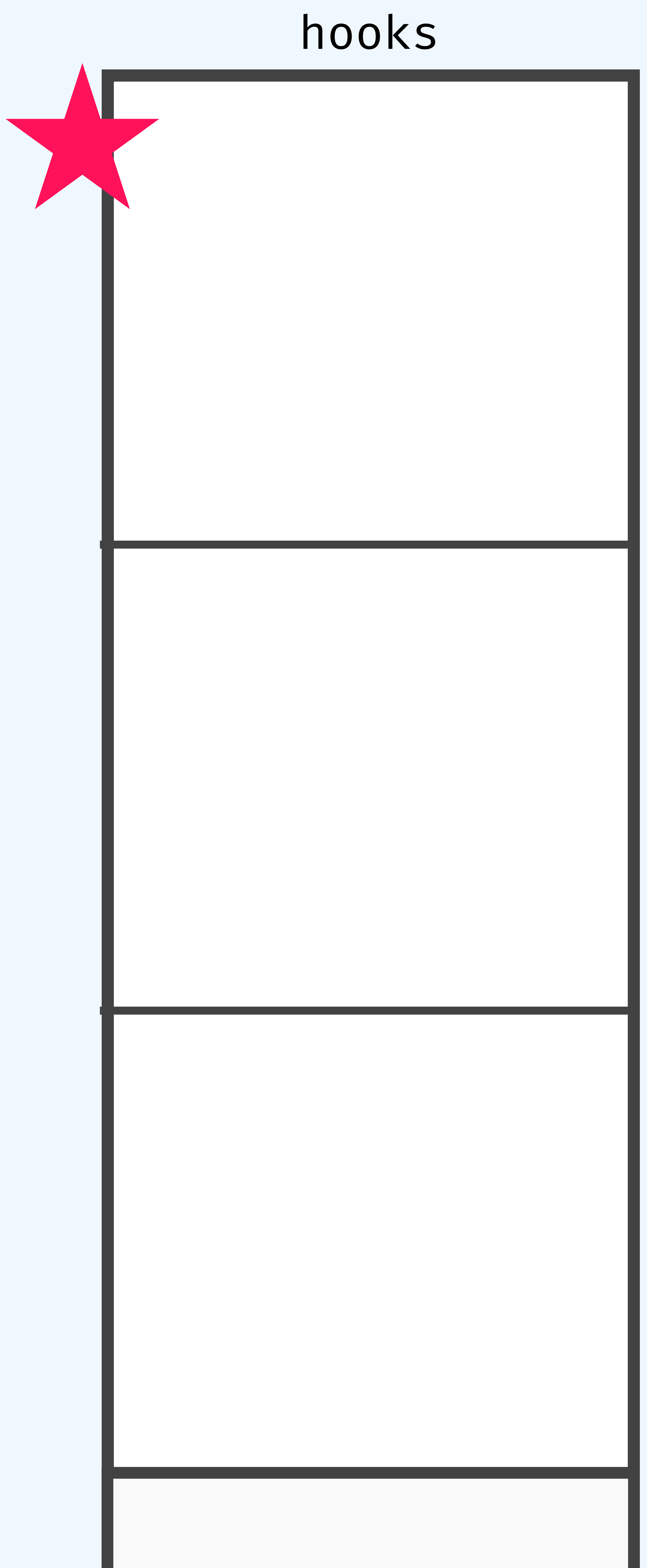


```
Username:
Password:
Remember: no
```

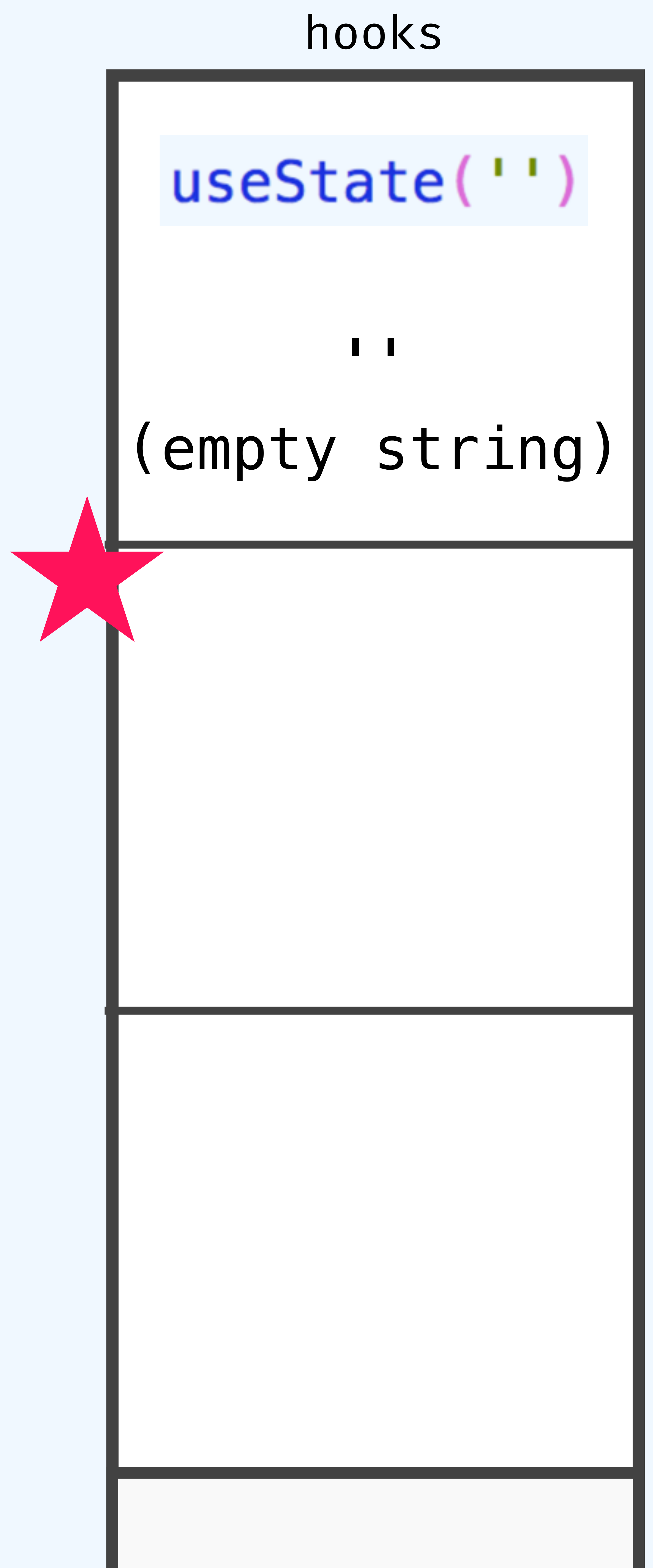
```
const LoginForm = () => {
  const [username, setUsername] = useState('');
  const [password, setPassword] = useState('');
  const [remember, setRememberMe] = useState(false);

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



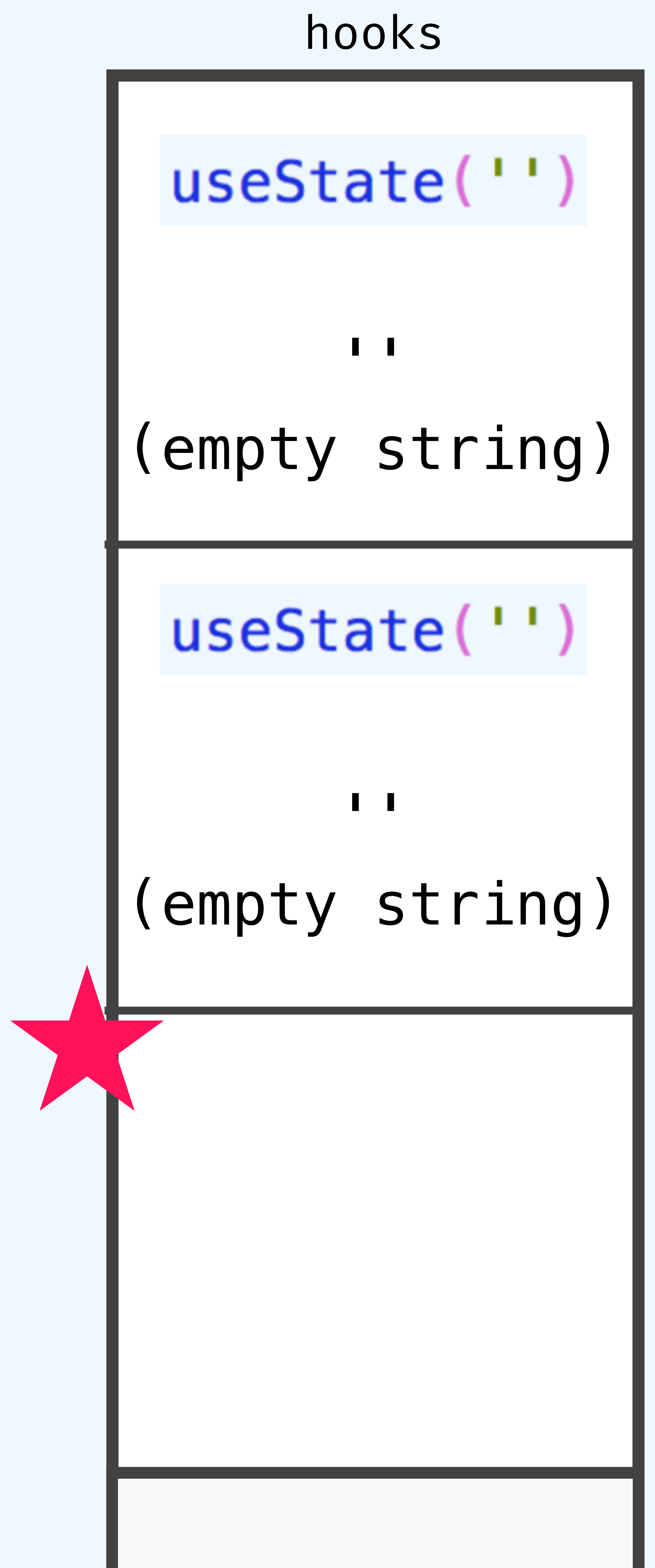
```
const LoginForm = () => {
  const [username, setUsername] = useState('');
  const [password, setPassword] = useState('');
  const [remember, setRememberMe] = useState(false);

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



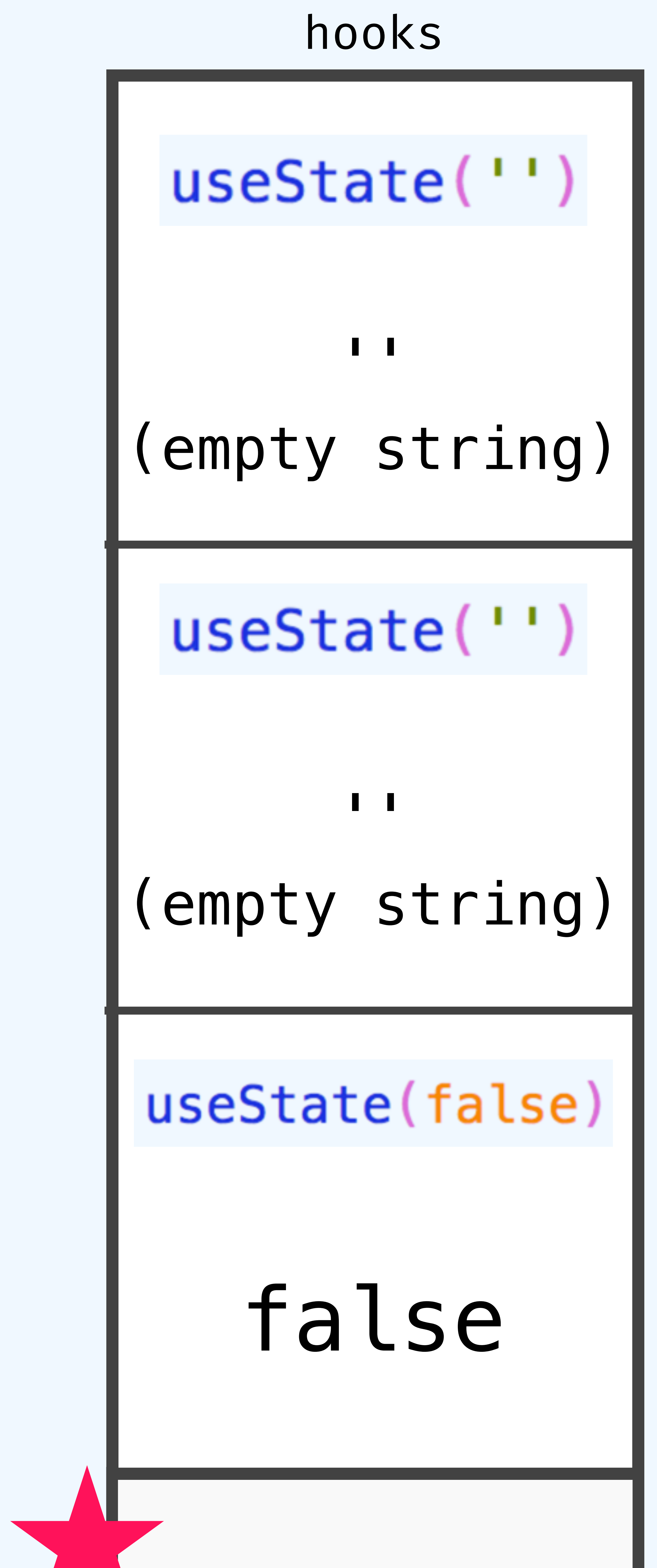
```
const LoginForm = () => {
  const [username, setUsername] = useState('');
  const [password, setPassword] = useState('');
  const [remember, setRememberMe] = useState(false);

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



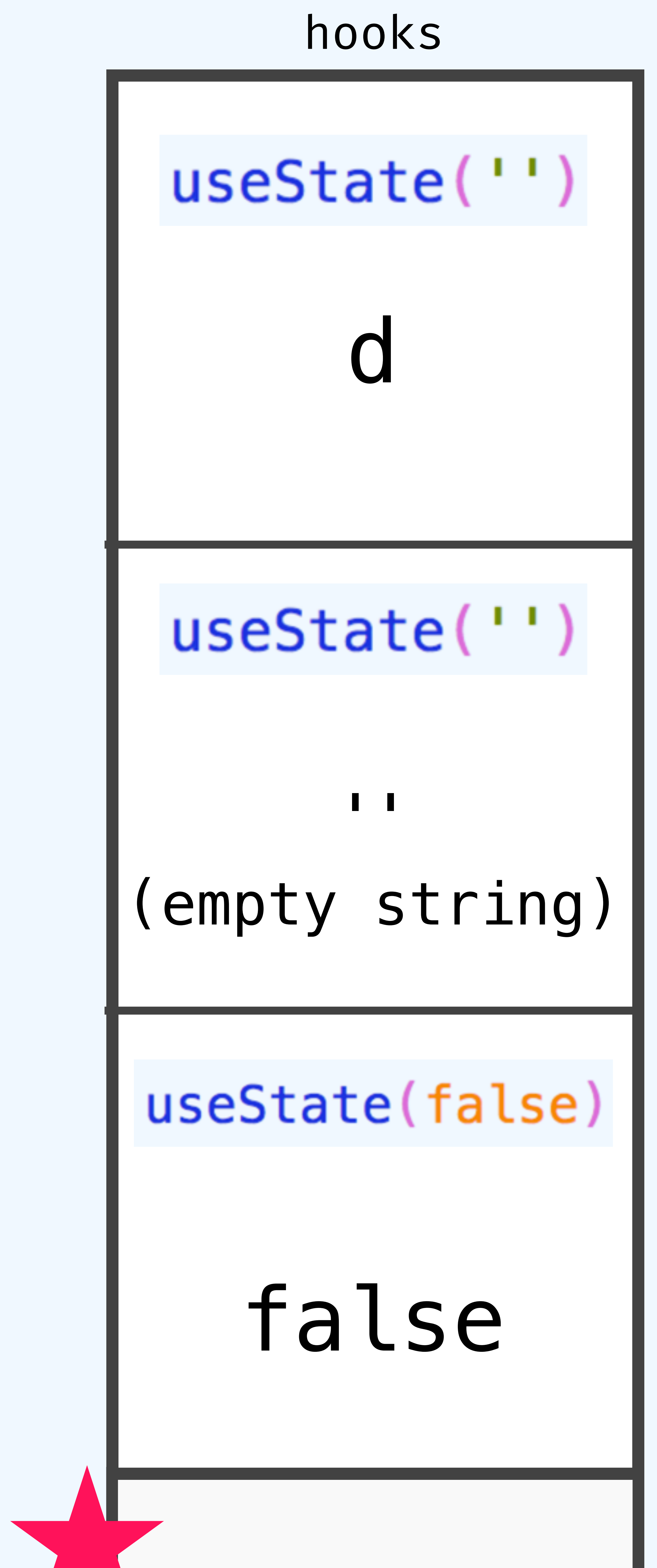
```jsx
const LoginForm = () => {
  const [username, setUsername] = useState('');
  const [password, setPassword] = useState('');
  const [remember, setRememberMe] = useState(false);

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

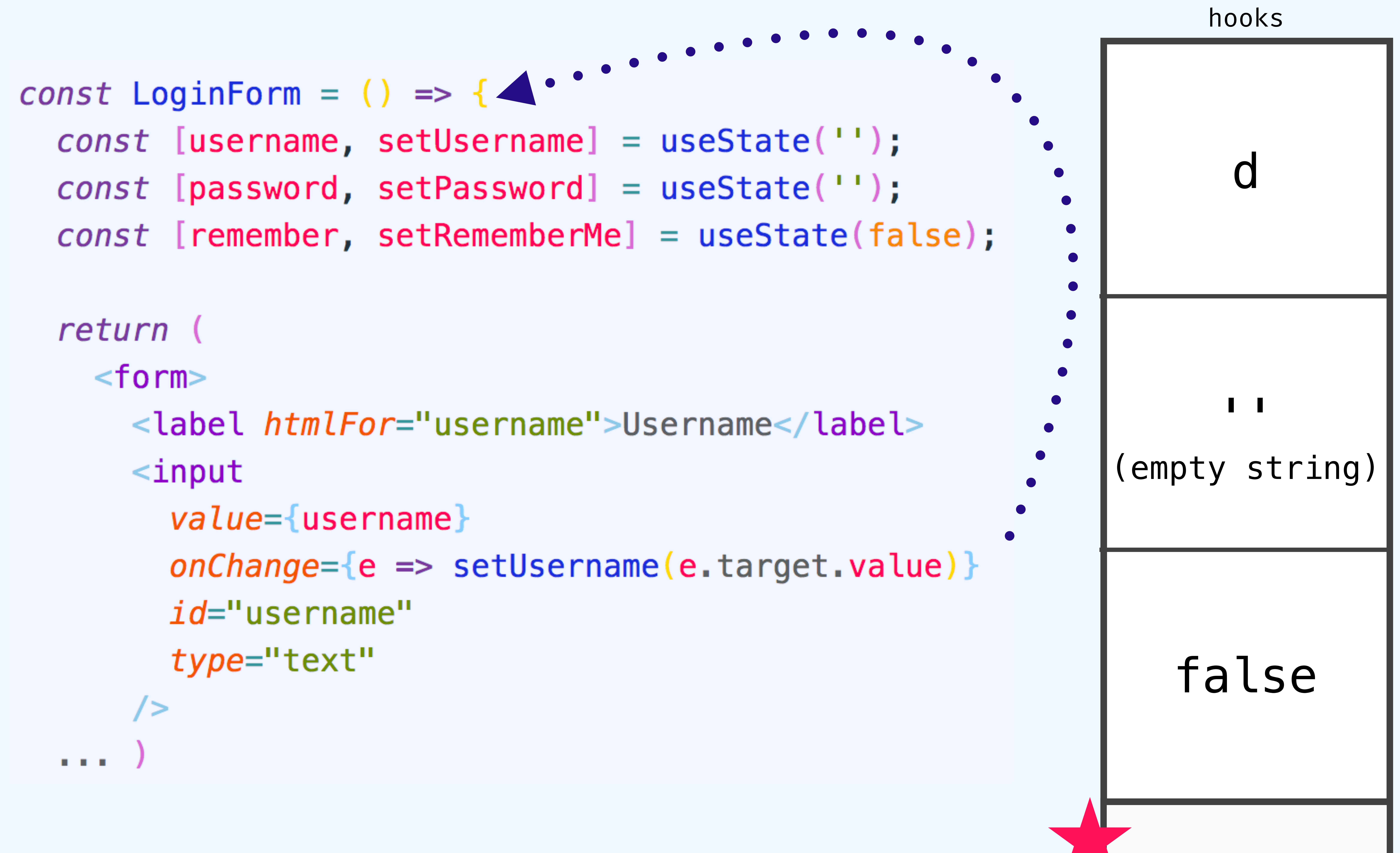


```
const LoginForm = () => {
  const [username, setUsername] = useState('');
  const [password, setPassword] = useState('');
  const [remember, setRememberMe] = useState(false);

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



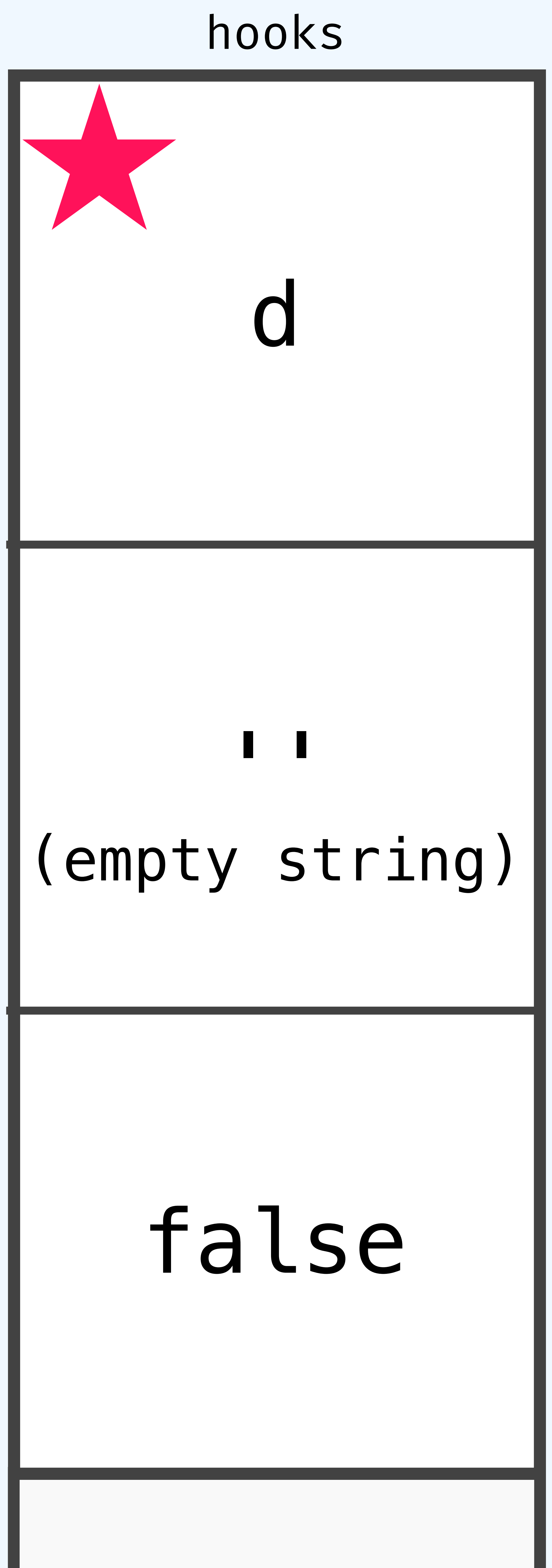
```
const LoginForm = () => {
  const [username, setUsername] = useState('');
  const [password, setPassword] = useState('');
  const [remember, setRememberMe] = useState(false);

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



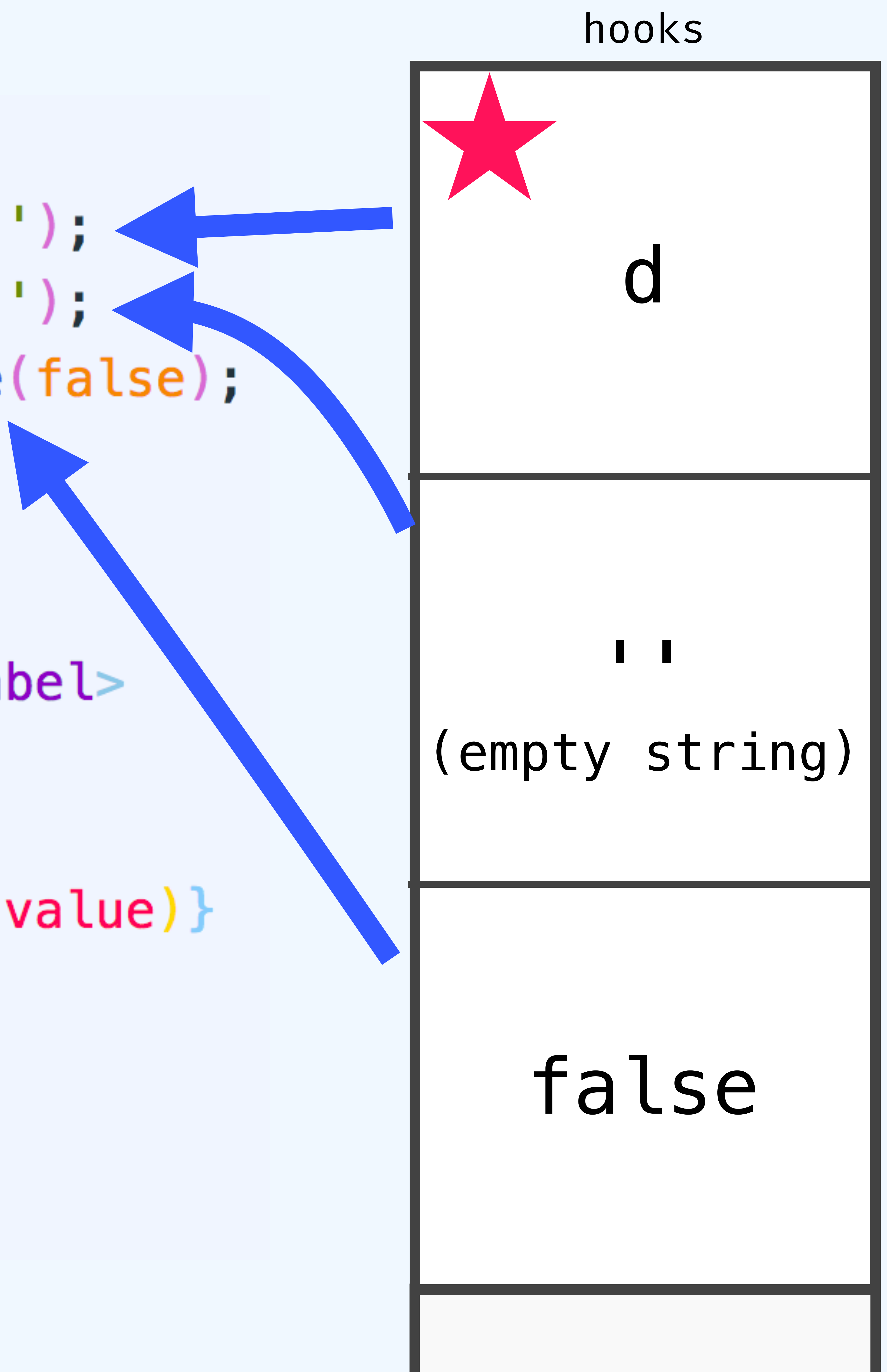


```
const LoginForm = () => {
  const [username, setUsername] = useState('');
  const [password, setPassword] = useState('');
  const [remember, setRememberMe] = useState(false);

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

```
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"
      />
  ... )
```



```
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"
      />
  ... )
```



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

index.js

```
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';

function Room() {
  
}

ReactDOM.render(
  <Room />,
  document.querySelector('#root')
);

```

0 0 -- INSERT -- Ln 6, Col 3 Spaces: 2 JavaScript Prettier: ✓

React App

localhost:3000

When to useReducer?

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

React **Hooks**

useContext

```
import React, { useState } from 'react';
import ReactDOM from 'react-dom';
import './index.css';

function App() {
  const [user, setUser] = useState({
    username: 'dave'
  });

  return (
    <>
      <Header user={user} />
      <Body user={user} />
    </>
  );
}

function Header({ user }) {
  return <header>Hi {user.username}</header>;
```

index.js
0 0 -- NORMAL --
Ln 7, Col 20 Spaces: 2 JavaScript Prettier: ✓
React App
localhost:3000
Hi dave
dave

This is some great content right here.

index.js
React App
localhost:3000
-- INSERT --
Ln 16, Col 15
Spaces: 2
JavaScript
Prettier: ✓

```
  });

  return (
    <UserContext.Provider value={user}>
      <>
        <Header user={user} />
        <Body />
      </>
    </UserContext.Provider>
  );
}

function Header({ user }) {
  return <header>Hi {user.username}</header>;
}

function Body() {
  return (
    <main>
      <Sidebar />
```

Hi dave

dave

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

index.js
React App
localhost:3000
-- INSERT --
Ln 1, Col 13
Spaces: 2
JavaScript
Prettier: ✓

```javascript
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';

const App = () => {

};

ReactDOM.render(
  <App />,
  document.querySelector('#root')
);

```

index.js
React App
localhost:3000
-- INSERT --
Ln 5, Col 3
Spaces: 2
JavaScript
Prettier: ✓

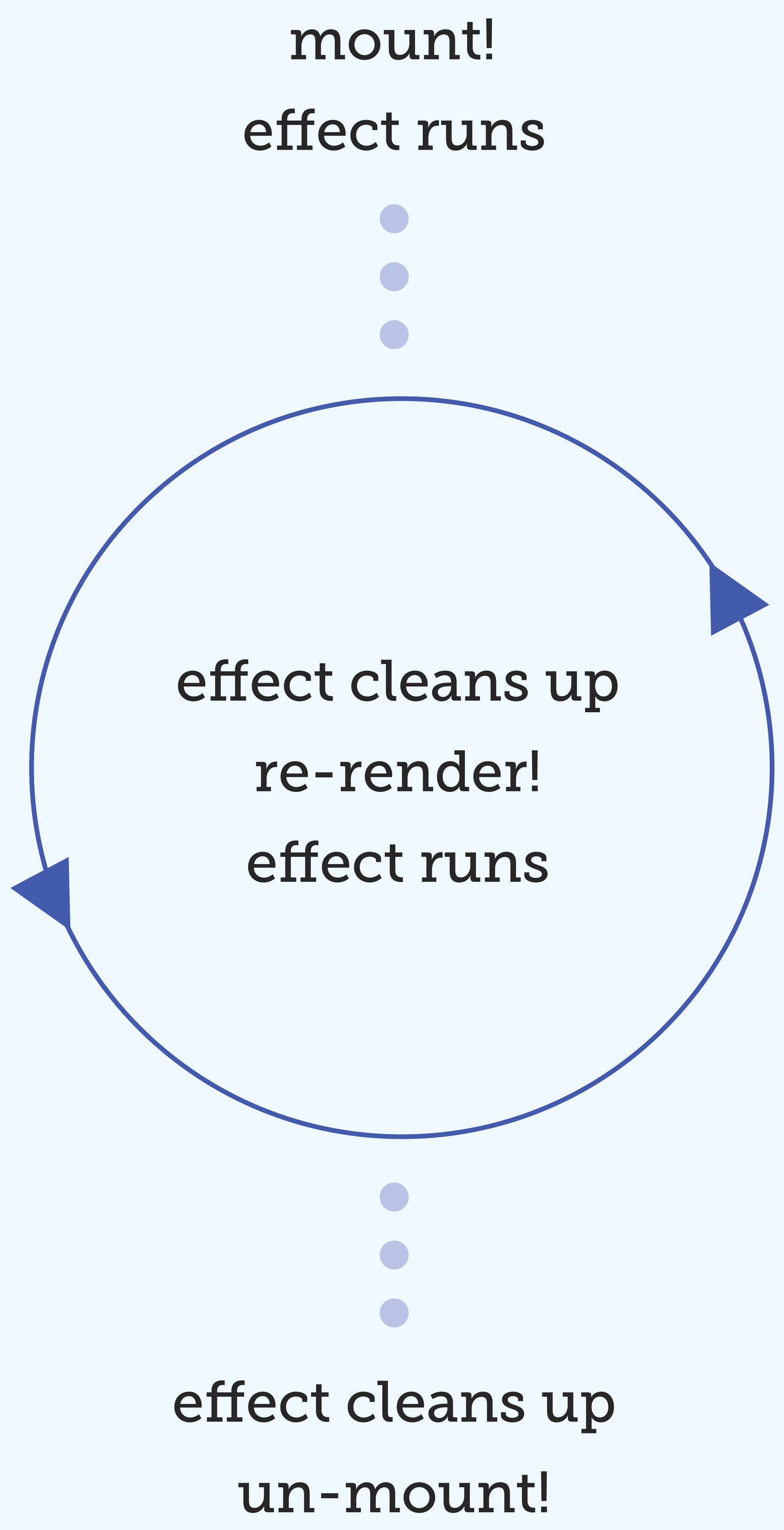
```
import React, { useState, useEffect } from 'react';
import ReactDOM from 'react-dom';

const Reddit = () => {
  
};

ReactDOM.render(
  <Reddit />,
  document.querySelector('#root')
);

```

index.js

```
import React, { useState, useEffect } from 'react';
import ReactDOM from 'react-dom';
import './index.css';

const App = () => {
  const [title, setTitle] = useState('');

  useEffect(() => {
    document.title = title;
  });

  return (
    <input
      value={title}
      onChange={e => setTitle(e.target.value)}
    />
  );
};

```

-- INSERT --
Ln 6, Col 42
Spaces: 2
JavaScript
Prettier: ✓
Hello Boston
localhost:3000

Hello Boston


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

index.js
React App
localhost:3000
-- INSERT --
Ln 6, Col 51
Spaces: 2
JavaScript
Prettier: ✓

```
import React, { useState } from 'react';
import ReactDOM from 'react-dom';
import './index.css';

function SpoilerAlert({ text }) {
  const [isVisible, setVisible] = useState(false);

  return (
    <div>
      {isVisible && <span>{text}</span>}
      {!isVisible && (
        <span className="hidden">~spoilers~</span>
      )}
      <button onClick={() => setVisible(!isVisible)}>
        {isVisible ? 'Hide' : 'Show'}
      </button>
    </div>
  );
}
```

~spoilers~

Show

A Few Custom Hook Ideas

`useLocalStorage`

`useAudio`

`useAuth`

`useLocation`

`useApolloClient`

`useFetch`

`useArray`

`useInterval`

`useHistory`

`useAxios`

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

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

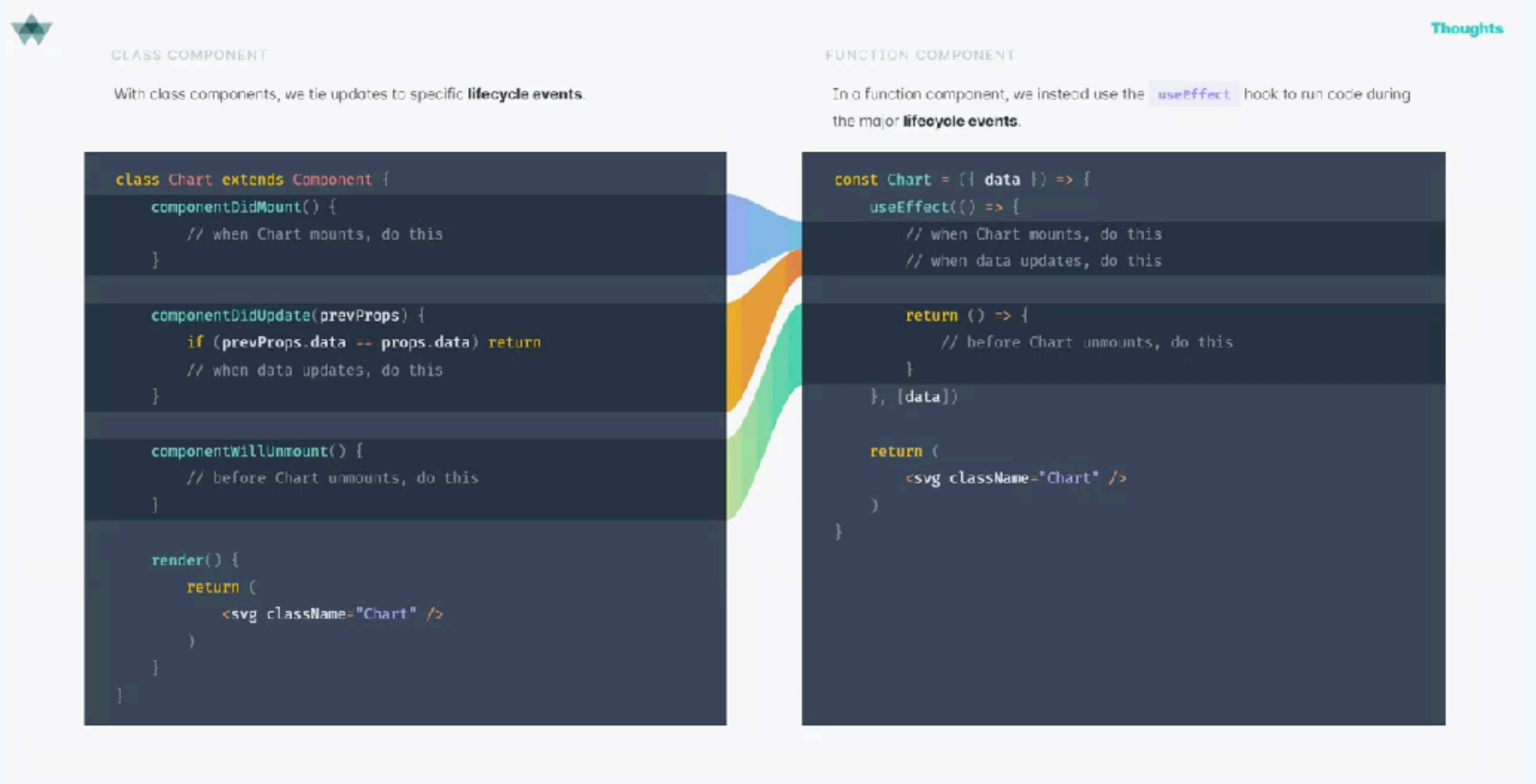
https://reactjs.org/hooks

Hooks Week!

https://daveceddia.com/hooks

Thinking in React Hooks

Amelia Wattenberger



CLASS COMPONENT

With class components, we tie updates to specific **lifecycle events**.

```
class Chart extends Component {
    componentDidMount() {
        // when Chart mounts, do this
    }

    componentDidUpdate(prevProps) {
        if (prevProps.data == props.data) return
        // when data updates, do this
    }

    componentWillUnmount() {
        // before Chart unmounts, do this
    }

    render() {
        return (
            <svg className="Chart" />
        )
    }
}
```

FUNCTION COMPONENT

In a function component, we instead use the `useEffect` hook to run code during the major **lifecycle events**.

```
const Chart = ({ data }) => {
    useEffect(() => {
        // when Chart mounts, do this
        // when data updates, do this

        return () => {
            // before Chart unmounts, do this
        }
    }, [data])

    return (
        <svg className="Chart" />
    )
}
```

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

Thanks!

Examples and slides:

https://daveceddia.com/boston

@dceddia

daveceddia.com