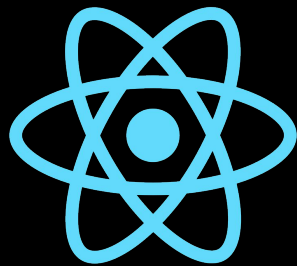


< React.js />



*React is a declarative, efficient, and flexible JavaScript library for **building user interfaces**. It lets you compose complex UIs from small and isolated pieces of code called "**components**".*

reactjs.org

# Components

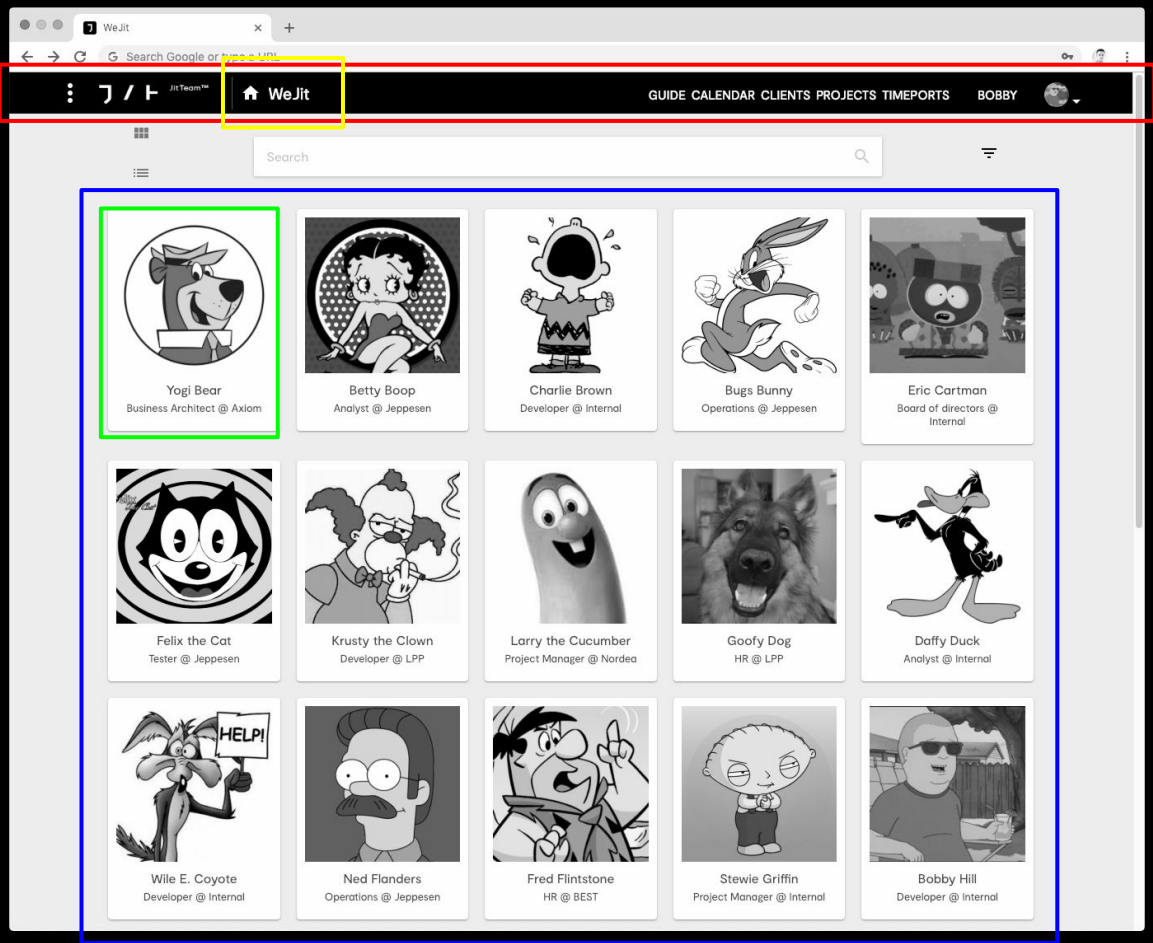
## Main

- Header

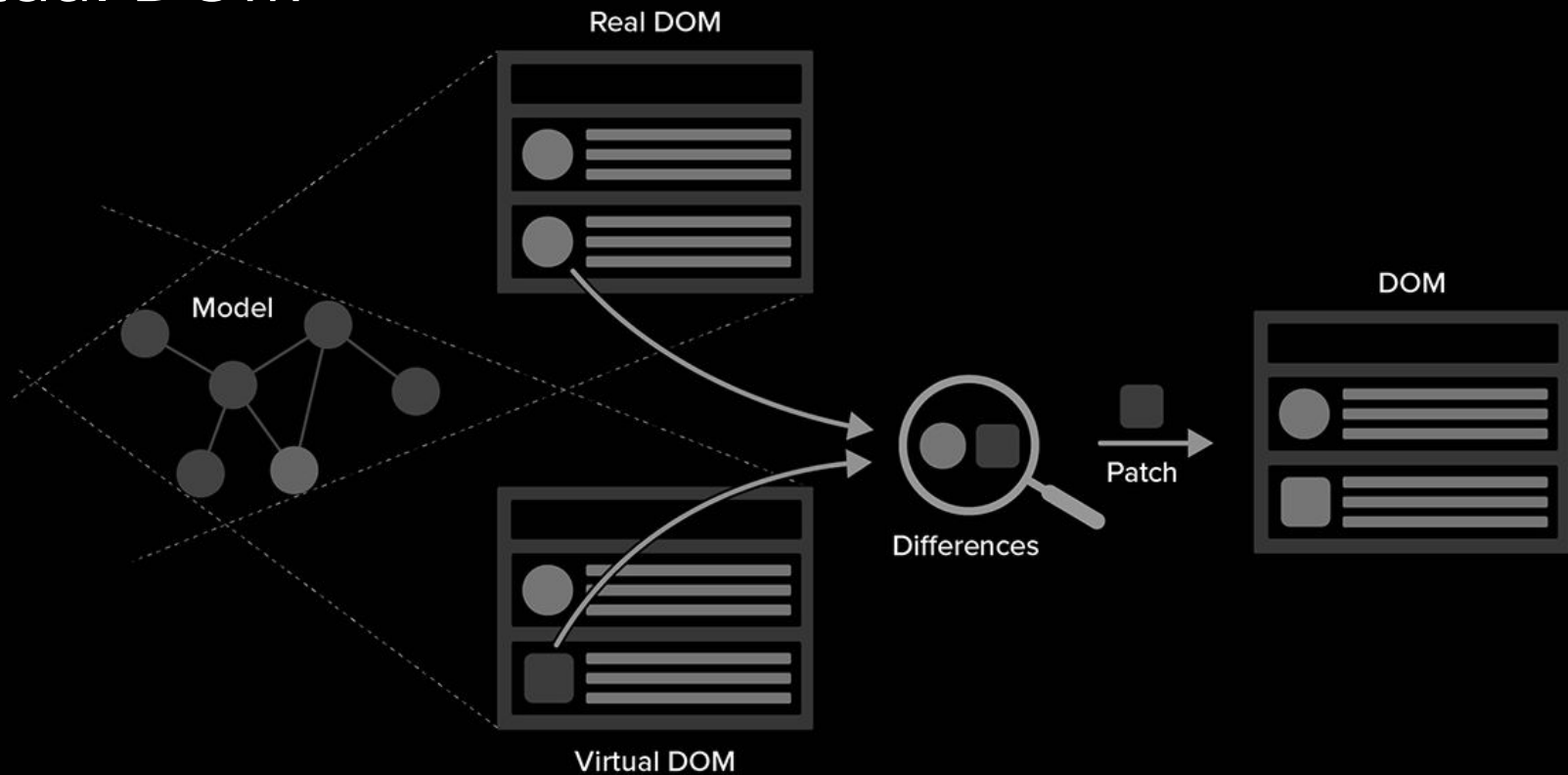
- Logo

- UserList

- UserItem



# Virtual DOM



# JSX

Syntax extension *not required*

HTML + JavaScript

markup + logic

# JSX examples

```
const name = '3LO';  
  
const element = <h1>Hello, {name}</h1>;
```

```
const element = <img src={user.avatarUrl} />
```

# JSX element

```
const element = (  
  <h1 className="greeting">  
    Hello, world!  
  </h1>  
);
```

=

```
const element = React.createElement(  
  'h1',  
  {className: 'greeting'},  
  'Hello, world!'  
);
```

```
// Note: this structure is simplified  
const element = {  
  type: 'h1',  
  props: {  
    className: 'greeting',  
    children: 'Hello, world!'  
  }  
};
```

# JSX rendering

```
ReactDOM.render(  
  element,  
  document.getElementById('root')  
);
```



# JSX updating

```
function tick() {  
  const element = (  
    <div>  
      <h1>Hello, world! </h1>  
      <h2>It is {new  
Date().toLocaleTimeString()} </h2>  
    </div>  
  );  
  ReactDOM.render(element,  
document.getElementById('root'));  
}  
  
setInterval(tick, 1000);
```

*React Only Updates  
What's Necessary*

# Components

*user-defined component*

```
function Welcome() {  
  return <h1>Hello, 3LO</h1>;  
}
```

*function component*

=

```
class Welcome extends React.Component {  
  render() {  
    return <h1>Hello, 3LO</h1>;  
  }  
}
```

*class component*

ES6

# Function component

```
function Welcome () {  
  return <h1>Hello, 3LO</h1>;  
}  
  
const element = <Welcome/>;  
  
ReactDOM.render(  
  element,  
  document.getElementById('root')  
)
```

1. We call `ReactDOM.render()` with the `<Welcome/>` element.
2. React calls the `Welcome` component.
3. Our `Welcome` component returns a `<h1>Hello, 3LO</h1>` element as the result.
4. React DOM efficiently updates the DOM to match `<h1>Hello, 3LO</h1>`.

# Class component

## *function component*

```
function Welcome () {  
  return <h1>Hello, 3LO</h1>;  
}  
  
const element = <Welcome/>;  
  
ReactDOM.render(  
  element,  
  document.getElementById('root')  
)
```



## *class component*

```
class Welcome extends React.Component {  
  render() {  
    return (  
      <div>  
        <h1>Hello, 3LO!</h1>  
      </div>  
    );  
  }  
}  
  
ReactDOM.render(  
  <Welcome/>,  
  document.getElementById('root')  
)
```

# Props

```
class Clock extends React.Component {  
  render() {  
    return (  
      <div>  
        <h1>Hello, 3LO! </h1>  
        <h2>It is  
        {this.props.date.toLocaleTimeString()} </h2>  
      </div>  
    );  
  }  
}  
  
ReactDOM.render(  
  <Clock date={new Date()} />,  
  document.getElementById('root')  
);
```

*Events flows up*

```
function Grandparent(props) {  
  return (  
    <Parent  
      name="Sara"  
      onClick={this.handleClick}  
    />  
  )  
}
```

```
function Parent(props) {  
  return (  
    <Child  
      name={props.name}  
      onClick={props.handleClick}  
    />  
  )  
}
```

```
function Child(props) {  
  return (  
    <div className="itemContainer">  
      <p>{props.name}</p>  
      <input onClick={props.onClick} />  
    </div>  
  )  
}
```

*Data flows down*

# State

```
class Clock extends React.Component {  
  constructor(props) {  
    super(props);  
    this.state = {date: new Date()};  
  }  
  render() {  
    return (  
      <div>  
        <h1>Hello, world!</h1>  
        <h2>It is {this.state.date.toLocaleTimeString()}.</h2>  
      </div>  
    );  
  }  
}
```

# *setState()*

```
this.state.comment = 'Hello';
```

*wrong*

```
this.setState({comment: 'Hello'});
```

*correct*



# Handling Events

## HTML

```
<a href="#" onclick="console.log('The link  
was clicked.');" return false" >
```

Click me

```
</a>
```

*addEventListener?*

## REACT

```
function ActionLink() {  
  function handleClick(e) {  
    e.preventDefault();  
    console.log('The link was clicked.');  }  
  
  return (  
    <a href="#" onClick={handleClick}>  
      Click me  
    </a>  
  );  
}
```

# Handling Events **bind**

```
class Toggle extends React.Component {  
  constructor(props) {  
    super(props);  
    this.state = {isToggleOn: true};  
    this.handleClick = this.handleClick.bind(this);  
  }  
  handleClick() {  
    this.setState(state => ({  
      isToggleOn: !state.isToggleOn  
    }));  
  }  
  render() {  
    return (  
      <button onClick={this.handleClick}>  
        {this.state.isToggleOn ? 'ON' : 'OFF'}  
      </button>  
    );  
  }  
}
```

# ES6 arrow functions

( parameters ) => { statement }

---

1. Shorter syntax
2. No binding of *this*

# ES6 arrow functions

*Shorter syntax*

```
function sayHello(person) {  
  return `Hello, ${person}`;  
}  
sayHello('3LO');  
// Hello, 3LO
```

```
var sayHello = (person) => { `Hello, ${person}` }  
sayHello('3LO');  
// Hello, 3LO
```

```
var sayHello = person => `Hello, ${person}`
```

# ES6 arrow functions

*No binding of **this***

```
<button onClick={ (e) => this.deleteRow(id, e) }>Delete Row</button>
```

=

```
<button onClick={this.deleteRow.bind(this, id)}>Delete  
Row</button>
```

# Handling Events **class fields**

```
class LoggingButton extends React.Component {  
  handleClick = () => {  
    console.log('this is:', this);  
  }  
  
  render() {  
    return (  
      <button onClick={this.handleClick}>  
        Click me  
      </button>  
    );  
  }  
}
```

=

```
class LoggingButton extends React.Component {  
  handleClick() {  
    console.log('this is:', this);  
  }  
  
  render() {  
    return (  
      <button onClick={(e) => this.handleClick(e)}>  
        Click me  
      </button>  
    );  
  }  
}
```

# Lists & Keys

## MAP

```
const numbers = [1, 2, 3, 4, 5];  
const listItems = numbers.map((number) =>  
  <li>{number}</li>  
);
```

## RENDER

```
ReactDOM.render(  
  <ul>{listItems}</ul>,  
  document.getElementById('root')  
);
```

```
function NumberList(props) {  
  const numbers = props.numbers;  
  const listItems = numbers.map((number) =>  
    <li key={number.toString()}>  
      {number}  
    </li>  
  );  
  return (  
    <ul>{listItems}</ul>  
  );  
}
```

```
const numbers = [1, 2, 3, 4, 5];  
  
ReactDOM.render(  
  <NumberList numbers={numbers} />,  
  document.getElementById('root')  
);
```

# Forms

type

value

**onChange**

**onSubmit**

checked

selected

```
class NameForm extends React.Component {
  constructor(props) {
    super(props);
    this.state = {value: ''};

    this.handleChange = this.handleChange.bind(this);
    this.handleSubmit = this.handleSubmit.bind(this);
  }

  handleChange(event) {
    this.setState({value: event.target.value});
  }

  handleSubmit(event) {
    alert('An essay was submitted: ' + this.state.value);
    event.preventDefault();
  }

  render() {
    return (
      <form onSubmit={this.handleSubmit}>
        <label>
          Name:
          <input type="text" value={this.state.value}
onChange={this.handleChange} />
        </label>
        <input type="submit" value="Submit" />
      </form>
    );
  }
}
```



# Lifecycle **methods**

## Mounting

```
constructor()  
static getDerivedStateFromProps()  
render()  
componentDidMount()
```

## Updating

```
static getDerivedStateFromProps()  
shouldComponentUpdate()  
render()  
getSnapshotBeforeUpdate()  
componentDidUpdate()
```

## Unmounting

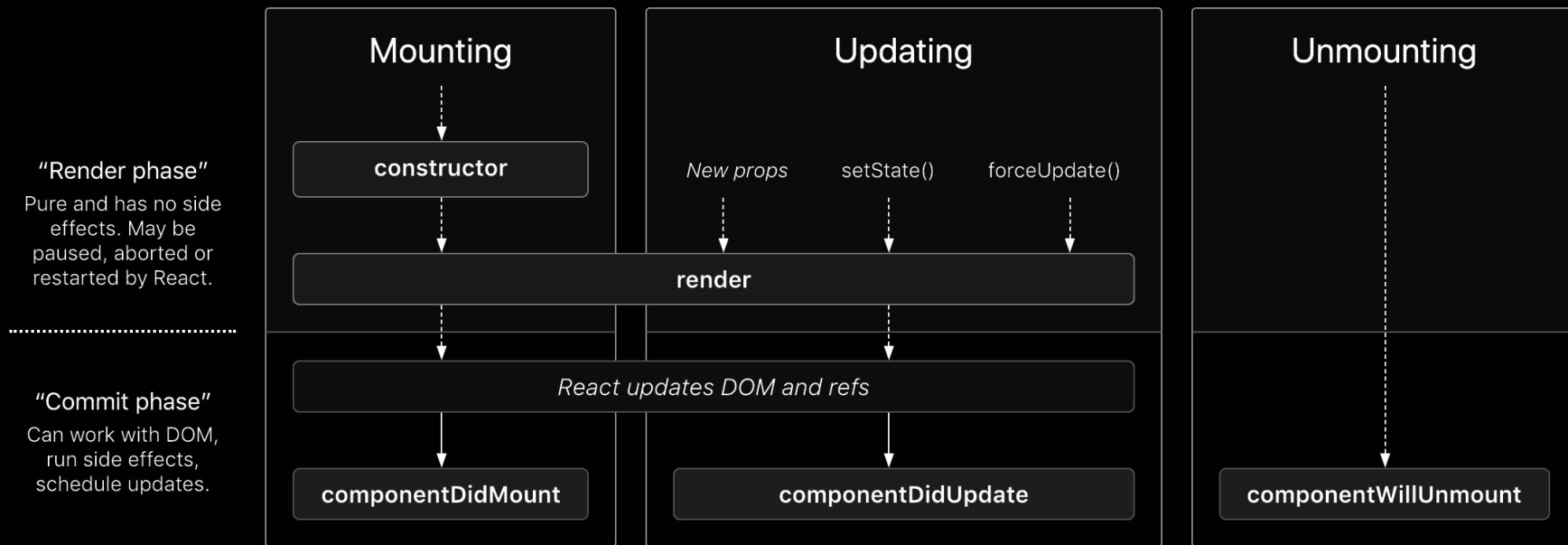
```
componentWillUnmount()
```

## Error Handling

```
static getDerivedStateFromError()  
componentDidCatch()
```

```
class Clock extends React.Component {  
  constructor(props) {  
    ...  
  }  
  
  componentDidMount() {  
    this.timerID = setInterval(  
      () => this.tick(),  
      1000  
    );  
  }  
  
  componentWillUnmount() {  
    clearInterval(this.timerID);  
  }  
  
  tick() {  
    this.setState({  
      date: new Date()  
    });  
  }  
  
  render() {  
    ...  
  }  
}
```

# Lifecycle **methods**



# Conditional rendering

*If statement*

```
render() {  
  const isLoggedIn = this.state.isLoggedIn;  
  let button;  
  
  if (isLoggedIn) {  
    button = <LogoutButton onClick={this.handleLogoutClick} />;  
  } else {  
    button = <LoginButton onClick={this.handleLoginClick} />;  
  }  
  
  return (  
    <div>  
      <Greeting isLoggedIn={isLoggedIn} />  
      {button}  
    </div>  
  );  
}
```

# Conditional rendering

*Logical && Operator*

```
function Mailbox(props) {  
  const unreadMessages = props.unreadMessages;  
  return (  
    <div>  
      <h1>Hello!</h1>  
      {unreadMessages.length > 0 &&  
        <h2>  
          You have {unreadMessages.length} unread  
messages.  
        </h2>  
      }  
    </div>  
  );  
}
```

# Conditional rendering

*Conditional Operator*

*condition ? true : false.*

```
render() {  
  const isLoggedIn = this.state.isLoggedIn;  
  return (  
    <div>  
      The user is <b>{isLoggedIn ? 'currently' :  
'not'}</b> logged in.  
    </div>  
  );  
}
```

```
render() {  
  const isLoggedIn = this.state.isLoggedIn;  
  return (  
    <div>  
      {isLoggedIn ? (  
        <LogoutButton onClick={this.handleLogoutClick} />  
      ) : (  
        <LoginButton onClick={this.handleLoginClick} />  
      )}  
    </div>  
  );  
}
```

# Toolchain

Kind	Tools	Task
Package Manager	Yarn, npm	It lets you take advantage of a vast ecosystem of third-party packages, and easily install or update them.
Bundler	Webpack, Parcel	It lets you write modular code and bundle it together into small packages to optimize load time.
Compiler	Babel	It lets you write modern JavaScript code that still works in older browsers.

# npm examples

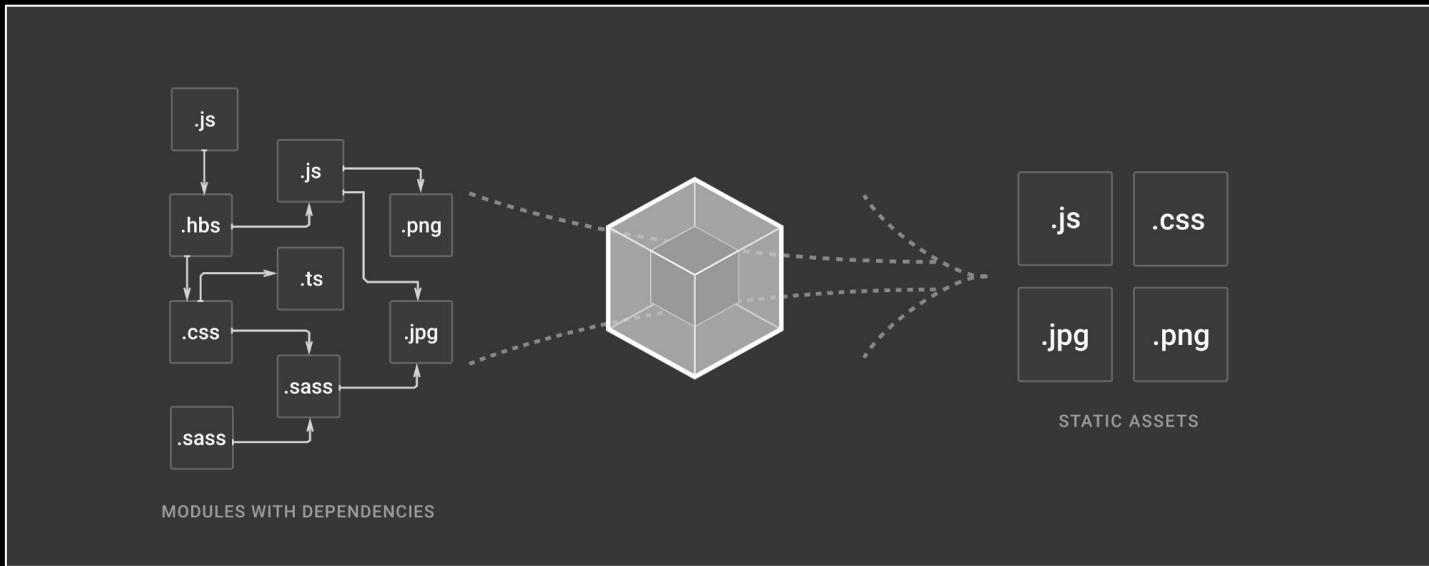
```
npm init
```

```
npm install npm@latest -g
```

```
npm install --save-dev webpack@4.19.1
```

*<https://docs.npmjs.com/>*

# Webpack



<https://webpack.js.org/>



# Babel

IN

```
// ES2015 arrow function  
[1, 2, 3].map((n) => n + 1);
```

```
[1, 2, 3].map(function(n) {  
  return n + 1;  
});
```

OUT

*<https://babeljs.io/>*

# New app

**node.js** + **npm** <https://nodejs.org/en/download/>

```
npx create-react-app my-app
```

```
cd my-app
```

*Development*

```
npm start
```

*Production*

```
npm run build
```

# Workshop #1

Tic-tac-toe  
game

# Homework #1

Postcard app  
reactivation

*<https://github.com/mackankowski/frontend-bootcamp/tree/master/playground/react/postcard-app>*