

More React and `fetch()`

Edgardo Molina, PhD | Head Instructor
CUNY Tech Prep 2017-2018

React and ES6

ES5 library methods are now deprecated in React

Prefer to use ES6 classes and features

this

works differently in ES6

have to use *autobinding* or *arrow functions*



Best Practices

Capitalize Component names

Use composition instead of inheritance

Don't use these

Mixins

No inheritance (in your own classes/components)



Component Definition with Classes

```
// ES6 Classes (preferred)
class Greeting extends React.Component {
  render() {
    return <h1>Hello, {this.props.name}</h1>;
  }
}
```

```
// ES5 Prototypal Objects
var Greeting = React.createClass({
  render: function() {
    return <h1>Hello, {this.props.name}</h1>;
  }
});
```




Functional Components

```
// A functional component
function Welcome(props) {
  return <h1>Hello, {props.name}</h1>;
}

// A class component (same as above)
class Welcome extends React.Component {
  render() {
    return <h1>Hello, {this.props.name}</h1>;
  }
}
```





Stateless vs Stateful Components

Stateless vs Stateful

You can use *functional* or *class* style for stateless components

Prefer to

- lift state up (from child to parent) and
- pass data down (from parent to child, via props)

Use `this.setState({ ... })` to update state






Initiating State

```
// ES6 Initial State
class Counter extends React.Component {
  constructor(props) {
    super(props); // MUST CALL SUPER
    this.state = {count: props.initialCount};
  }
  // ...
}
```

```
// ES5 Initial State
var Counter = React.createClass({
  getInitialState: function() {
    return {count: this.props.initialCount};
  },
  // ...
});
```




Autobinding

```
// ES6 Manual Binding
class SayHello extends React.Component {
  constructor(props) {
    super(props);
    this.state = {message: 'Hello!'};
    // THIS LINE IS IMPORTANT!
    this.handleClick = this.handleClick.bind(this);
  }

  handleClick() {
    alert(this.state.message);
  }

  render() {
    // Because `this.handleClick` is bound, we can use it as an event handler.
    return (
      <button onClick={this.handleClick}>
        Say hello
      </button>
    );
  }
}
```



```
// ES5 Autobinding
var SayHello = React.createClass({
  getInitialState: function() {
    return {message: 'Hello!'};
  },

  handleClick: function() {
    alert(this.state.message);
  },

  render: function() {
    return (
      <button onClick={this.handleClick}>
        Say hello
      </button>
    );
  }
});
```



Default Props and PropTypes

```
// ES6 PropTypes and Default Props
class Greeting extends React.Component {
  // ...
}

Greeting.propTypes = {
  name: React.PropTypes.string
};

Greeting.defaultProps = {
  name: 'Mary'
};
```

```
// ES5 PropTypes and Default Props
var Greeting = React.createClass({
  propTypes: {
    name: React.PropTypes.string
  },

  getDefaultProps: function() {
    return {
      name: 'Mary'
    };
  },

  // ...
});
```



Component Lifecycle

React Component Lifecycle

React provides methods that we can implement that run at specific times, such as when the component is:

- Mounting
- Updating
- Unmounting

Read more here:

<https://facebook.github.io/react/docs/react-component.html>

<https://engineering.musefind.com/react-lifecycle-methods-how-and-when-to-use-them-271b692b1>



References

<https://babeljs.io/learn-es2015/>

<https://facebook.github.io/react/docs/react-without-es6.html>



Other Important React Topics

Lists and Keys

<https://facebook.github.io/react/docs/lists-and-keys.html>

Refs and Interacting with the DOM

<https://facebook.github.io/react/docs/refs-and-the-dom.html>

Form Inputs

<https://facebook.github.io/react/docs/forms.html>



The fetch API

fetch()

fetch()

- is a newer standard for AJAX requests
 - [https://en.wikipedia.org/wiki/Ajax_\(programming\)](https://en.wikipedia.org/wiki/Ajax_(programming))
- allows our frontend javascript code to access URL's
 - For example, to retrieve data from an API
- Uses promises



```
// GET JSON
```

```
fetch('/users.json')  
  .then(function(response) {  
    return response.json()  
  }).then(function(json) {  
    console.log('parsed json', json)  
  }).catch(function(ex) {  
    console.log('parsing failed', ex)  
  });
```

```
// POST JSON
```

```
fetch('/users', {  
  method: 'POST',  
  headers: {  
    'Content-Type': 'application/json'  
  },  
  body: JSON.stringify({  
    name: 'Hubot',  
    login: 'hubot',  
  })  
});
```




References

<https://github.com/github/fetch> (Polyfill)

<https://jakearchibald.com/2015/thats-so-fetch/> (lot's of details!)





Where does `fetch()`
go in a React
component?



Good Question!

It depends...

Options for calling fetch within components...

NOT here:

- constructor()
- componentWillMount()
- render()

Where?

- componentDidMount()
- Event handlers
(but *beware*, this could cause a lot of fetch()
calls which will impact performance and costs!)

