React Component Syntax

In this lesson, we'll study some ways that can make our React component declaration shorter and simpler.

Component syntax evolution

React's way of defining components has evolved. In its early stages, the React.createClass() method was the default way of creating a React class component. Nowadays, it isn't used anymore, because, with the rise of JavaScript ES6, the previously used React class component syntax became the default.

However, JavaScript is continually evolving, and thus JavaScript enthusiasts pick up new ways of doing things all the time. That's why you will often find different syntaxes for React class components. One way of defining a React class component, with state and class methods, is the following:

```
import React from 'react';
require('./style.css');

import ReactDOM from 'react-dom';
import Counter from './app.js';

ReactDOM.render(
  <Counter />,
  document.getElementById('root')
);
```

A shorter way of defining a React class

However, when implementing lots of React class components, the binding of class methods in the constructor and having a constructor in the first place becomes a tedious implementation detail. Fortunately, there is a shorthand syntax for getting rid of both annoyances:

```
import React from 'react';
require('./style.css');

import ReactDOM from 'react-dom';
```

```
import Counter from './app.js';

ReactDOM.render(
  <Counter />,
  document.getElementById('root')
);
```

By using JavaScript arrow functions, you can auto-bind class methods without having to bind them in the constructor. Also, the constructor can be left out, when not using the props, by defining the state directly as a class property. (Note: Be aware that class properties are not in the JavaScript language yet.) Therefore you can say that this way of defining a React class component is way more concise than the other version.

Now you know why React uses JavaScript classes for defining React class components. They are used when you need access to React's API (lifecycle methods, `this.state` and `this.setState()`). In the following, you will see how React components can be defined differently, without using a JavaScript class, because you may not always need class methods, lifecycle methods, and state.