State in React

What a React component looks like is a reflection of its state - in this lesson, we'll look at how to use it and some potfalls

What is state?

The core of every React component is its state. The state determines what the component looks like, and you can update that as you go. The entire page won't reload if a component's state is updated only the component will - hence, it allows you to create pages that are dynamic and interactive. You can think of a component's state like water's temperature; changing temperature can significantly change what water looks like. If the temperature is brought below 0°C the water will solidify, if it is between 0°C and 100°C it will be a liquid, and if brought above 100°C it will turn into a gas and evaporate. You can change how water behaves and looks just by changing the value of temperature and that is exactly what state is in React, you can change how components behave and look just by changing the state.

Creating a component with state

To be a little less abstract, the state is just an object in a React component that you can save and get data from such as strings and numbers. Let's see how the state is defined in a React component.

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

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

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

Changing state

Changing state

Most beginners try to change state in React by using the assignment operator like so `state. data = newValue` but this is *wrong* and will generate an error! So instead, React provides the `setState()` method that you can use to update state. The function, `setTimeout()`, as in the following, on **line 20** , is used like so `setTimeout(function(){}, timeInMS)`. It 'calls back' the function after the given delay in milliseconds. In this case, it calls back the `setState()` function which changes the components `lastname` to 'hood' after 5 seconds. Also, notice that we had to bund this callback function with `this`. That is because `this` within the function refers to the function itself and not the component, so without binding it, your program would crash.

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

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

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

Caveats of Changing States

There are some pitfalls you should be aware of. `setState()` does not *immediately* update your component. React may decide to update it later in a batch with a bunch of other components. If you want to check if your component has updated, you can use the `componentDidMount()` method which will be touched upon in the following lifecycle methods chapter.