React Components

React render() method

A React component class must contain the `render()` method. Inside this method, there should be a `return` statement. This method returns the JSX representation of that component instance. The following code block shows how a `render()` method should be constructed in a component:

```
render() {
    return ;
  }
```

React component base class

In order to create any React component, it needs the base class `React.Component` . React component classes follow regular JavaScript class syntax. The code block above could be used to create a React component using the base class `React.Component` .

```
class DarkMode extends React.Component {

}
```

Javascript React Library

In order to use React, we must first import the React library. We can import the React library by using the code block above. When we import the library it creates an object that contains properties needed to make React work such as using JSX or creating custom components.

```
import React from 'react';
```

React Component

A React component is a reusable piece of code used to define the appearance, behavior, and state of a portion of a web app's interface. The component is defined as a function or class. Using the component as a factory, an infinite number of component instances can be created.

JSX capitalization

React requires that the first letter of a component class be capitalized. This is required because based on capitalization JSX can tell the difference between an HTML tag and a component instance. If the first letter of a name is capitalized, then JSX realizes it's a component instance; if not, then it's an HTML tag.

```
// This would be considered a component by
React.
<ThisComponent />

// This would be considered a JSX HTML tag.
<div>
```

JavaScript ReactDOM.render()

In order to use the method `ReactDOM.render()` we must first import ReactDOM using the following code block. When `ReactDOM.render()` receives a component instance as its first argument, it calls in the render method from that component.

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

multi-line JSX Expression

Parentheses are used when writing a multi-line JSX expression. In the code block example we see that the component's `render()` method is split over multiple lines. Therefore it is wrapped in parentheses.

```
render() {
  return (
   <blockquote>
  <p>
   Be the change you wish to see in the
world.
  </p>
  <cite>
    <a target="_blank"

href="https://en.wikipedia.org/wiki/Mahatma_Gan

      Mahatma Gandhi
    </a>
  </cite>
</blockquote>
  );
```

JavaScript functions in React Components

A React component can contain JavaScript before the `return` statement. The JavaScript before the return statement informs the logic necessary to render the component. In the example block of code we see JavaScript prior to the `return` statement. This JavaScript function informs the logic necessary to render the component. The JavaScript math function in this example is

```
Math.floor(Math.random() * 10);
```

and this function in JavaScript will return a random integer from 0 to 9.

```
class Integer extends React.Component {
  render() {
    const randInteger =
Math.floor(Math.random() * 10);
    return <p>{randInteger}</p>;
  }
}
```

JavaScript Objects JSX Attribute values

In React, JSX attribute values can be set through data stored in regular JavaScript objects. We see this in the example block of code.

In our code example we first see our JavaScript object `seaAnemones` and the values stored with this image.

We then see how these stored values are used to set the attributes in our JSX expression for `class SeaAnemones`.

In our code example since `seaAnemones.src`, `seaAnemones.alt`, and `seaAnemones.width` are JavaScript properties they are wrapped in curly braces.

```
const seaAnemones = {
  src:
'https://commons.wikimedia.org/wiki/Category:Im

  alt: 'Sea Anemones',
  width:  '300px'
};

class SeaAnemones extends React.Component {
  render() {
    return (
      <div>
        <h1>Colorful Sea Anemones</h1>
        <img
          src={seaAnemones.src}
          alt={seaAnemones.alt}
          width={seaAnemones.width} />
      </div>
    );
  }
}

ReactDOM.render(
  <SeaAnemones/>,
  document.getElementById('app')
);
```