code|**cademy**

React Components

return statement

React function components must contain a return
statement. This should return some React elements
created with JSX.

```
function MyComponent() {
  return <h1>Hello from MyComponent!
</h1>;
}
```

Function Component Base

React function components follow regular JavaScript
class syntax in declaration and returns JSX elements.
This example shows a simple React function
component.

```
function MyComponent() {
  return <h1>Hello world!</h1>;
}
```

Importing React

In order to use React, we must first import the React
library. When we import the library, it creates an object
that contains properties needed to make React work,
including JSX and creating custom components.

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

React Components

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. Components are defined as
functions. Using the component as a factory, an infinite
number of component instances can be created.

```
import React from 'react';

function MyFunctionComponent() {
  return <h1>Hello from a function
component!</h1>;
}


class MyClassComponent extends
React.Component {
  render() {
    return <h1>Hello from a class
component!</h1>;
  }
}
```

JSX Capitalization

React requires that the first letter of components be capitalized. JSX will use this capitalization to tell the difference between an HTML tag and a component instance. If the first letter of a name is capitalized, then JSX knows it's a component instance; if not, then it's an HTML element.

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

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

Importing and Exporting Components in React

React components can be modularly structured and made reusable by placing them into their own files. Components can be exported and imported into a top-level file and rendered.

In **Greeting.js**:

```
function Greeting() {
  return (
    <>
      <h1>Hello, welcome to...</h
      <h2>Learn React!</h2>
    </>
  );
}


export default Greeting;
```

In **App.js**:

```
import Greeting from './Greeting'
```

Rendering a Component

A React function component can be rendered by creating a root container and rendering the component into the root container.

```
//Component to be rendered
function MyComponent() {
  return <h1>Hello, World!</h1>
}

//Rendering the component
ReactDOM.createRoot(
  document.getElementById('app')
).render(<MyComponent />);
```

Multi-line JSX Expressions

Parentheses are used when writing a multi-line JSX expression. In the example, we see that the component's `return` statement is split over multiple lines. Therefore it is wrapped in parentheses.

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

Logic Before return

A React component can contain JavaScript before any JSX is returned. The JavaScript before the `return` statement informs any logic necessary to render the component.
In the example code, we see JavaScript prior to the `return` statement which rounds the `value` to an integer.

```
function Integer() {
  const value = 3.14;
  const asInteger = Math.round(value);
  return <p>{asInteger}</p>;
}
```

Object Properties As 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 `<img>` attributes in our JSX expression for the `SeaAnemones` component.

```
const seaAnemones = {
  src:

'https://commons.wikimedia.org/wiki/Categ
ory:Images#/media/File:Anemones_0429.jpg'
,
  alt: 'Sea Anemones',
  width: '300px',
};

function SeaAnemones () {
  return (
    <div>
      <h1>Colorful Sea Anemones</h1>
      <img
        src={seaAnemones.src}
        alt={seaAnemones.alt}
        width={seaAnemones.width}
      />
    </div>
  );
}
```

↓ Print   ⚹ Share ▼