

What does ReactDOM.render() do?

Used to render React elements (components) into the DOM (Document Object Model). This method renders a React app to the web page. Render method takes 2 arguments, the first argument is the react element/component you want to render and the second argument is a reference to a DOM element, meaning, where you want to render the React element, where your react component will be inserted. When ReactDOM.render() is called, this is what happens:

- 1) A virtual representation of your React element is created (A virtual DOM)
- 2) The virtual DOM is compared with the actual DOM
- 3) The differences between the two are calculated(reconciliation)
- 4) The actual DOM is updated to match the virtual DOM efficiently by minimizing the number of changes needed.

As the application state changes or user interactions occur, React will re-render the components as needed, making it a powerful and efficient way to build dynamic and interactive web applications.

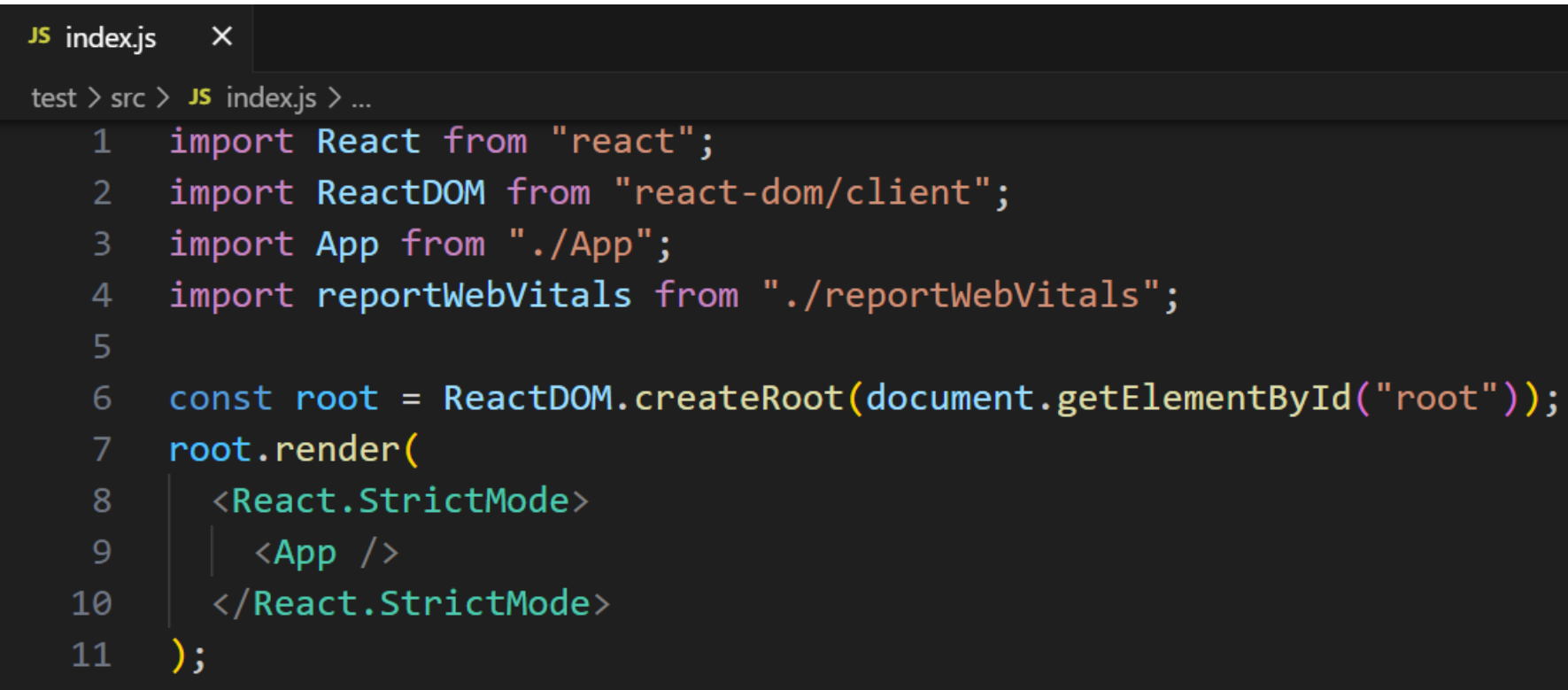
JS index.js X

test > src > JS index.js > ...

```
1  import React from "react";
2  import ReactDOM from "react-dom";
3
4  function MyComponent() {
5    |   return <h1>Hello, React!</h1>;
6  }
7
8  // Render the component into a DOM element with id 'root'
9  ReactDOM.render(<MyComponent />, document.getElementById("root"));
```

In the code above we used **react-dom** package. **react-dom** package is responsible for rendering React components into the DOM (Document Object Model) of a web page.

There is also **react-dom/client** package.



```
JS index.js  X
test > src > JS index.js > ...
1  import React from "react";
2  import ReactDOM from "react-dom/client";
3  import App from "./App";
4  import reportWebVitals from "./reportWebVitals";
5
6  const root = ReactDOM.createRoot(document.getElementById("root"));
7  root.render(
8    <React.StrictMode>
9    |   <App />
10   </React.StrictMode>
11 );
```

react-dom/client package provides methods just for client development. Client-specific initialization techniques are available in the **react-dom/client** package. **createRoot()**, **hydrateRoot()** functions are available on **react-dom/client** package

createRoot(): Used to create a root React component that acts as the container for the whole React application. It sets up the necessary context and provides the root component with access to the React DOM.

What is a class Component in React? Give an example. use "state" in the class component

```
1  import React from "react";
2
3  class Employee extends React.Component {
4    constructor(props) {
5      super(props);
6      this.state = { name: "Adam" };
7    }
8    render() {
9      return (
10         <div>
11           <h1>Employee {this.state.name}</h1>
12         </div>
13       );
14     }
15   }
16
17   export default Employee;
```

From React 16.3 you don't need to explicitly define a constructor in your class-based components. You can initialize state directly in the class property without using a constructor. However, if you need to perform additional setup or bindings, you can still use the constructor. So it's not mandatory for initializing state.

If there is a constructor, you must always pass the "props" to the "constructor" and the "super" method

React class components have a built-in **state** object. The **property** values are stored on the state object.

When the state object changes the component re-renders.

We can define a class without a constructor like below from React 16.3

```
1  import React from "react";
2
3  class Employee extends React.Component {
4      state = { name: "Adam" };
5
6      render() {
7          return (
8              <div>
9                  <h1>Employee {this.state.name}</h1>
10             </div>
11         );
12     }
13 }
14
15 export default Employee;
```

Why do we need to call `super()` in a constructor?

Because it calls the parent's constructor method and allows the component to inherit methods from its parent.