

W7D1

miguel.garrido@ironhack.com



Quick review

1. SPA vs MPA
2. JS Libraries
3. What is a Component
4. What is a Prop



States in react

States are objects that store a component's dynamic data

They determine how the component behaves.

When the state changes, React re-renders the component to reflect the updated data,

This allow us interactive and dynamic user interfaces.



useState

useState is a React Hook that lets you add a state variable to your component.

Hooks are special functions in React:

```
import { useState } from 'react';
```



useState

Normal variable (or non-state variable)

```
let index = 0;
```

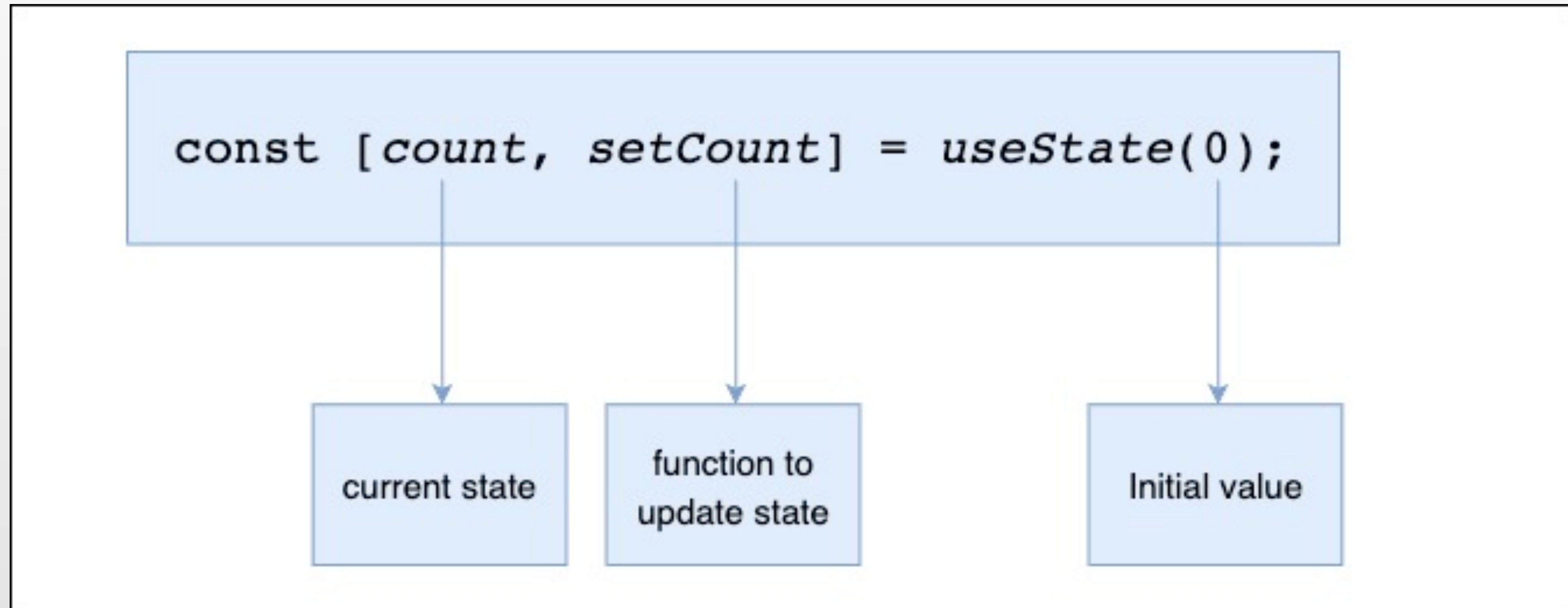
State Variable

```
const [index, setIndex] = useState(0);
```

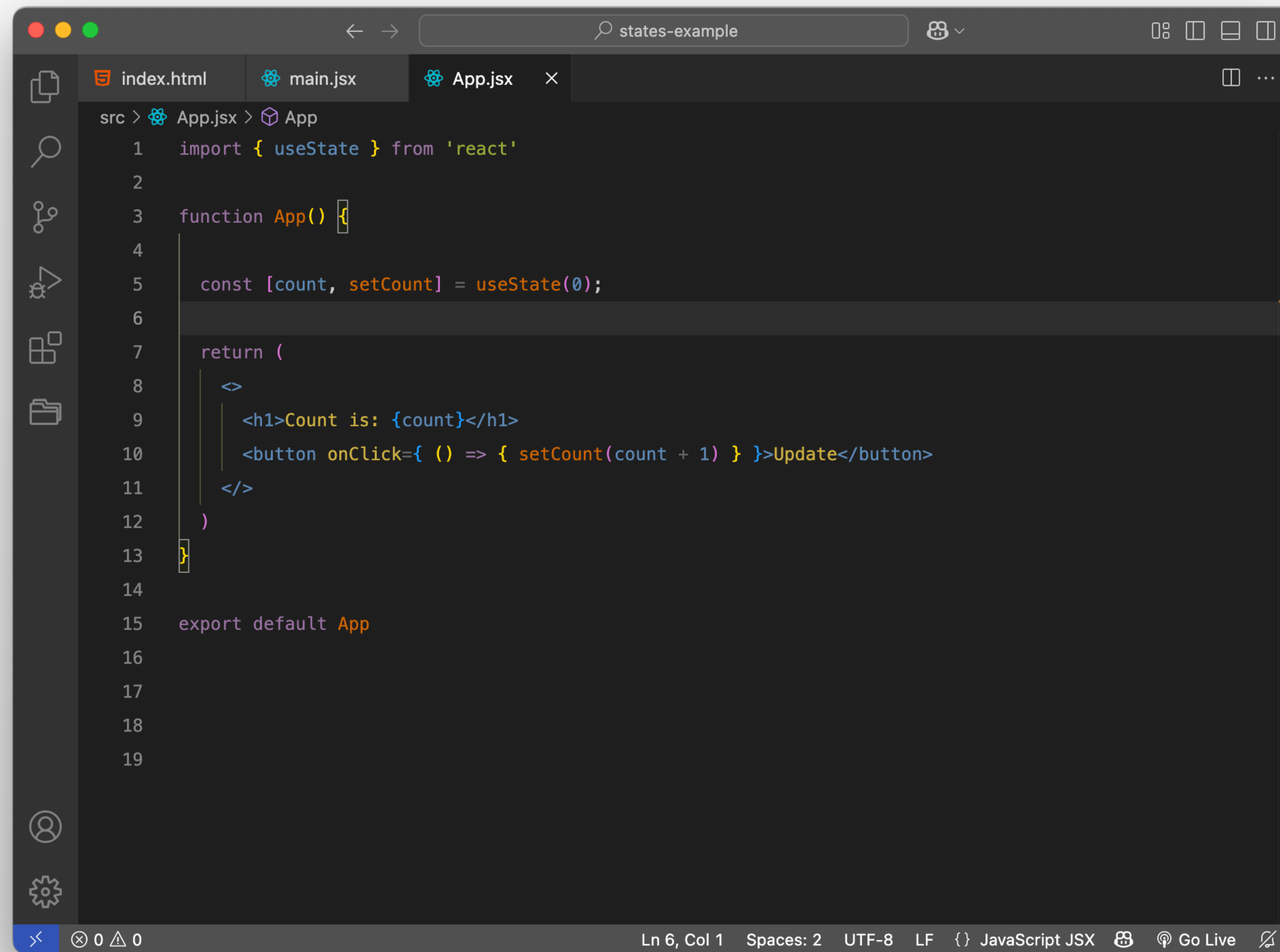
index is a state variable and **setIndex** is the setter function.



useState



useState example #1



The screenshot shows a code editor with a dark theme. The browser tab at the top is labeled 'states-example'. The editor has three tabs: 'index.html', 'main.jsx', and 'App.jsx'. The 'App.jsx' tab is active, showing the following code:

```
src > App.jsx > App
1  import { useState } from 'react'
2
3  function App() {
4
5      const [count, setCount] = useState(0);
6
7      return (
8          <>
9              <h1>Count is: {count}</h1>
10             <button onClick={ () => { setCount(count + 1) } }>Update</button>
11          </>
12      )
13  }
14
15  export default App
16
17
18
19
```

The status bar at the bottom indicates 'Ln 6, Col 1', 'Spaces: 2', 'UTF-8', 'LF', and 'JavaScript JSX'. There are also icons for a debugger, a live preview window, and a 'Go Live' button.

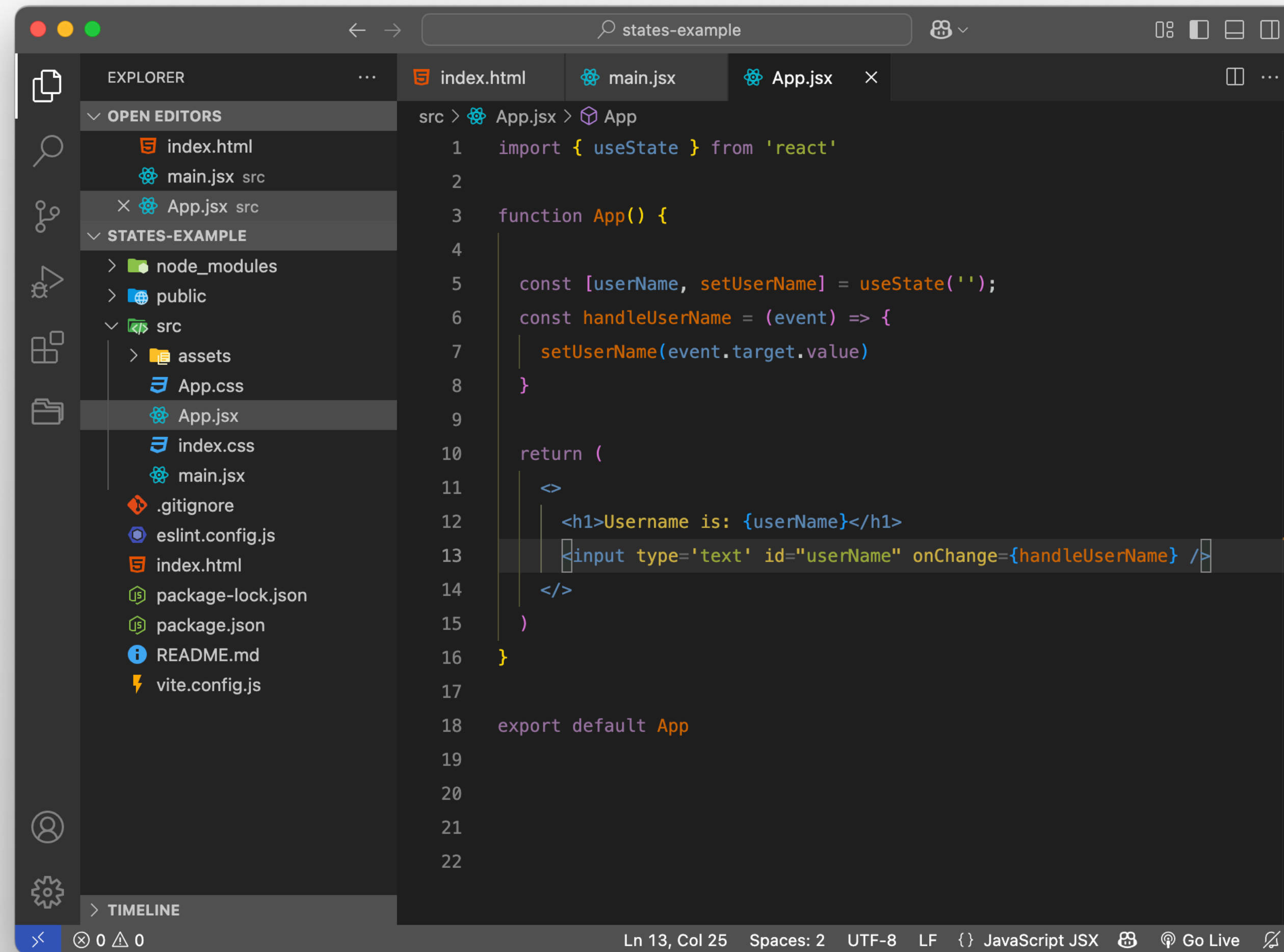


useState example #2

```
1 import { useState } from 'react'
2
3 function App() {
4
5   const [count, setCount] = useState(0);
6   const updateCount = () => {
7     setCount(count + 1);
8   }
9
10  return (
11    <>
12      <h1>Count is: {count}</h1>
13      <button onClick={updateCount}>Update</button>
14    </>
15  )
16 }
17
18 export default App
19
20
21
22
```



useState example #3



The screenshot shows a code editor with a dark theme. The Explorer panel on the left shows a project structure with a 'src' directory containing 'App.jsx', 'App.css', 'index.css', and 'main.jsx'. The main editor displays the code for 'App.jsx'. The code imports 'useState' from 'react' and defines a function 'App()' that uses 'useState' to manage a 'username' state. It also defines a 'handleUserName' function to update the state. The JSX returns a heading and a text input field that updates the state on change. The status bar at the bottom indicates the current position is at line 13, column 25.

```
1  import { useState } from 'react'
2
3  function App() {
4
5      const [username, setUsername] = useState('');
6      const handleUserName = (event) => {
7          setUsername(event.target.value)
8      }
9
10     return (
11         <>
12             <h1>Username is: {username}</h1>
13             <input type='text' id='username' onChange={handleUserName} />
14         </>
15     )
16 }
17
18 export default App
```

Further reading

<https://react.dev/learn/state-a-components-memory>

<https://react.dev/reference/react/useState>

[https://developer.mozilla.org/en-US/docs/
Learn web development/Core/Frameworks libraries/
React interactivity events state](https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Frameworks_libraries/React_interactivity_events_state)

