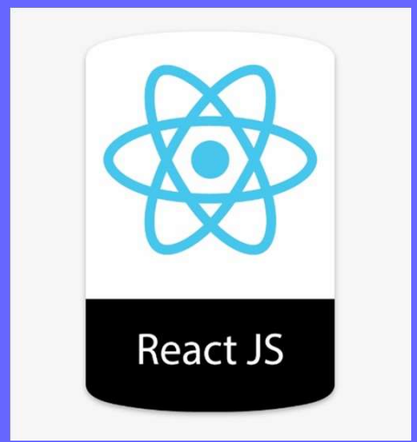
6

State in React

Objectives

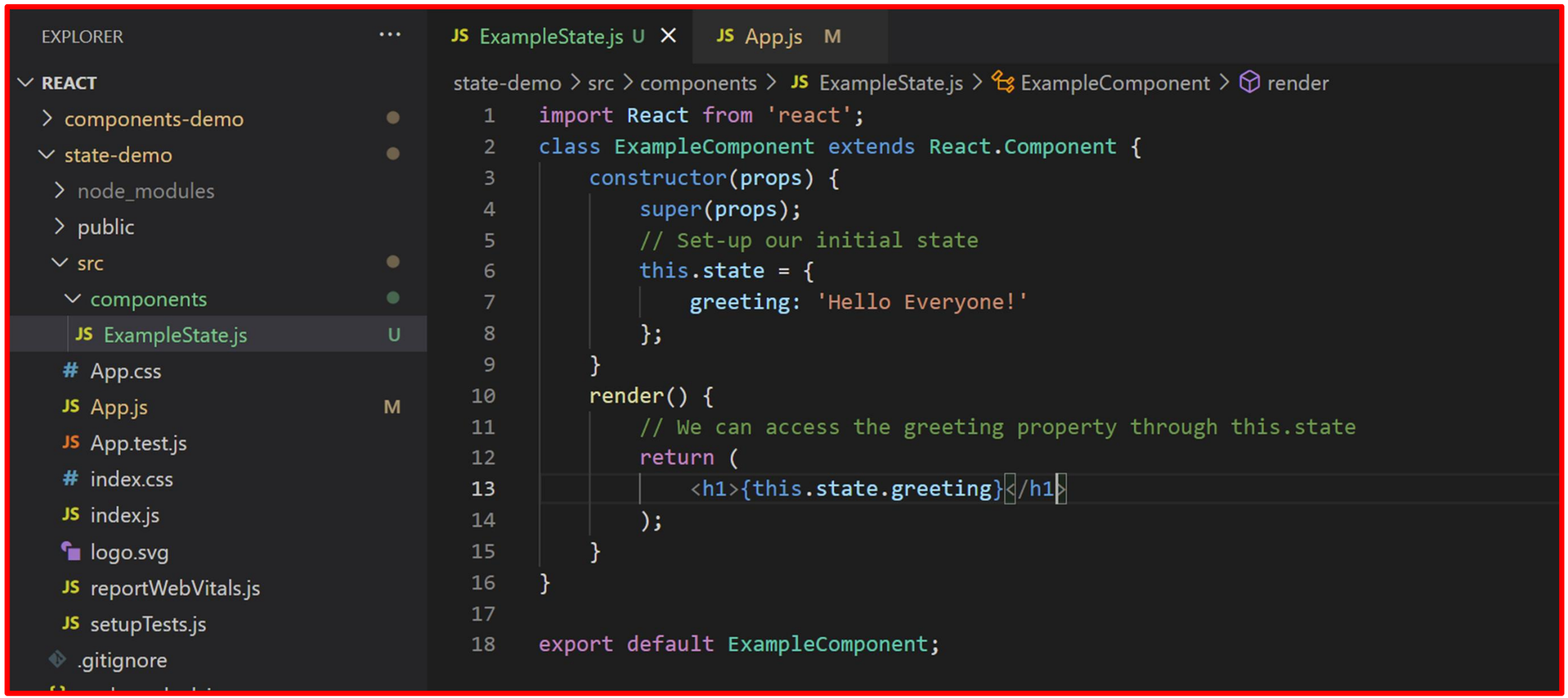
After completing this lesson, you should be able to do the following:

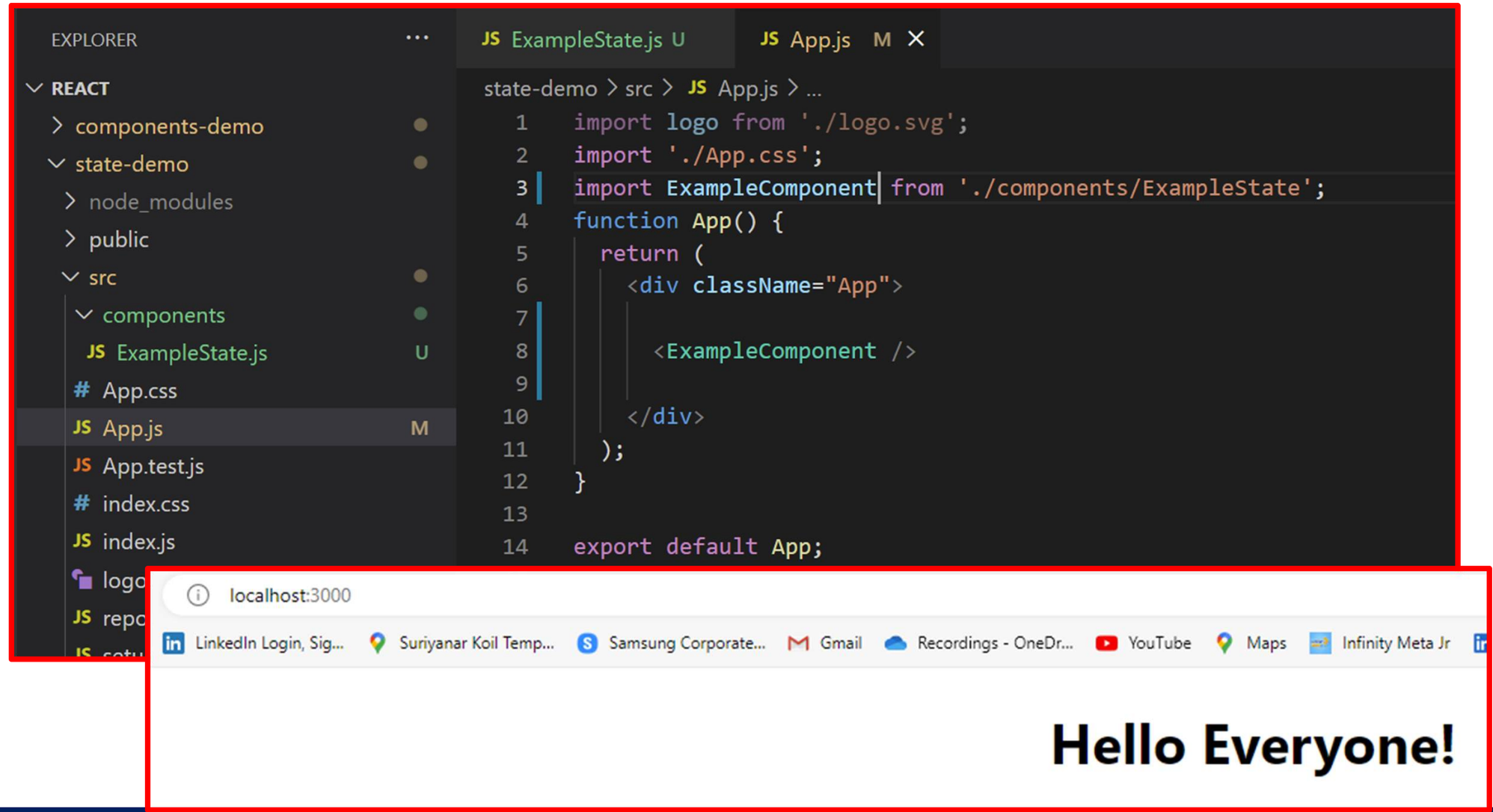
- Basics in State
- setState
- State, Events and Managed Controls

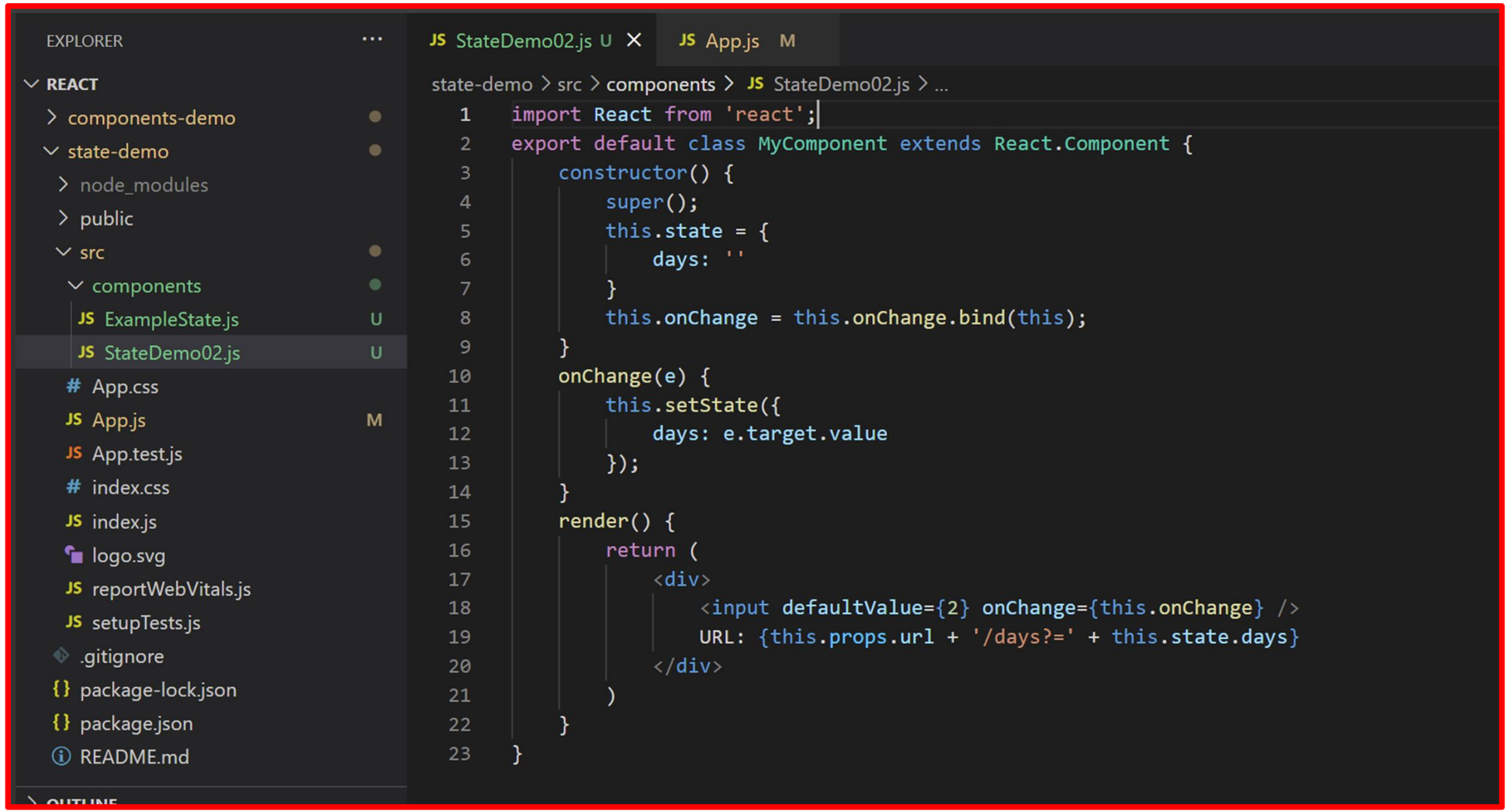


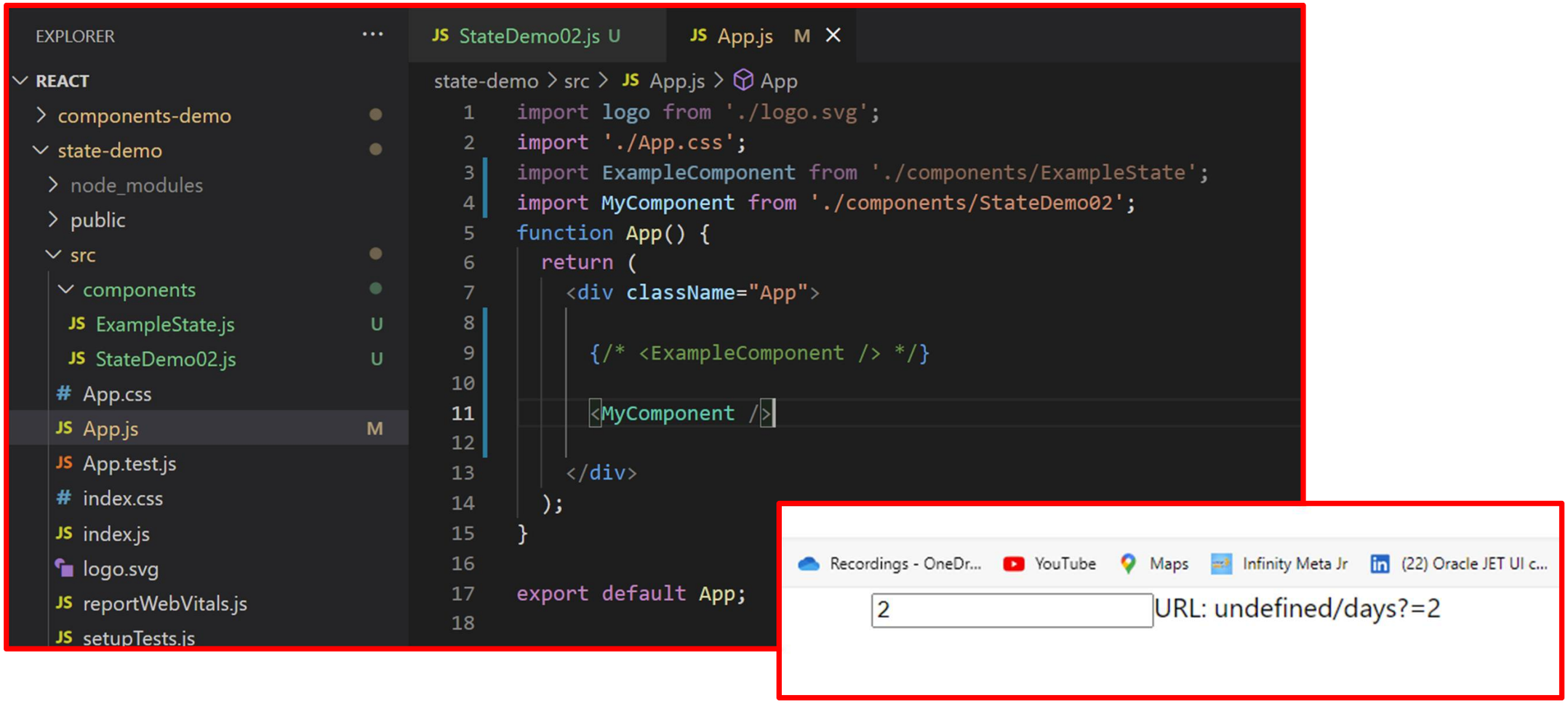
Introduction to React State

- State in React components is essential to manage and communicate data in your application. It is represented as a
- JavaScript object and has *component level* scope, it can be thought of as the private data of your component.


EXPLORER
REACT
components-demo
state-demo
node_modules
public
src
components
ExampleState.js U
App.css
App.js M
App.test.js
index.css
index.js
logo.svg
reportWebVitals.js
setupTests.js
.gitignore
ExampleState.js U
App.js M
state-demo > src > components > ExampleState.js > ExampleComponent > render
1 import React from 'react';
2 class ExampleComponent extends React.Component {
3 constructor(props) {
4 super(props);
5 // Set-up our initial state
6 this.state = {
7 greeting: 'Hello Everyone!'
8 };
9 }
10 render() {
11 // We can access the greeting property through this.state
12 return (
13 <h1>{this.state.greeting}</h1>
14);
15 }
16 }
17
18 export default ExampleComponent;


EXPLORER
REACT
components-demo
state-demo
node_modules
public
src
components
ExampleState.js U
App.css
App.js M
App.test.js
index.css
index.js
ExampleState.js U
App.js M
state-demo > src > App.js > ...
1 import logo from './logo.svg';
2 import './App.css';
3 import ExampleComponent from './components/ExampleState';
4 function App() {
5 return (
6
7
8 <ExampleComponent />
9
10
11);
12 }
13
14 export default App;
localhost:3000
LinkedIn Login, Sig...
Suriyanar Koil Temp...
Samsung Corporate...
Gmail
Recordings - OneDr...
YouTube
Maps
Infinity Meta Jr
Hello Everyone!


EXPLORER
REACT
components-demo
state-demo
node_modules
public
src
components
ExampleState.js U
StateDemo02.js U
App.css
App.js M
App.test.js
index.css
index.js
logo.svg
reportWebVitals.js
setupTests.js
.gitignore
package-lock.json
package.json
README.md
StateDemo02.js U
App.js M
state-demo > src > components > StateDemo02.js > ...
1 import React from 'react';
2 export default class MyComponent extends React.Component {
3 constructor() {
4 super();
5 this.state = {
6 days: ''
7 }
8 this.onChange = this.onChange.bind(this);
9 }
10 onChange(e) {
11 this.setState({
12 days: e.target.value
13 });
14 }
15 render() {
16 return (
17
18 <input defaultValue={2} onChange={this.onChange} />
19 URL: {this.props.url + '/days?=' + this.state.days}
20
21)
22 }
23 }


EXPLORER
REACT
components-demo
state-demo
node_modules
public
src
components
ExampleState.js U
StateDemo02.js U
App.css
App.js M
App.test.js
index.css
index.js
logo.svg
reportWebVitals.js
setupTests.js
StateDemo02.js U
App.js M
state-demo > src > App.js > App
1 import logo from './logo.svg';
2 import './App.css';
3 import ExampleComponent from './components/ExampleState';
4 import MyComponent from './components/StateDemo02';
5 function App() {
6 return (
7
8
9 {/* <ExampleComponent /> */}
10
11 <MyComponent />
12
13
14);
15 }
16
17 export default App;
18
Recordings - OneDr...
YouTube
Maps
Infinity Meta Jr
(22) Oracle JET UI c...
2
URL: undefined/days?=2

setState()

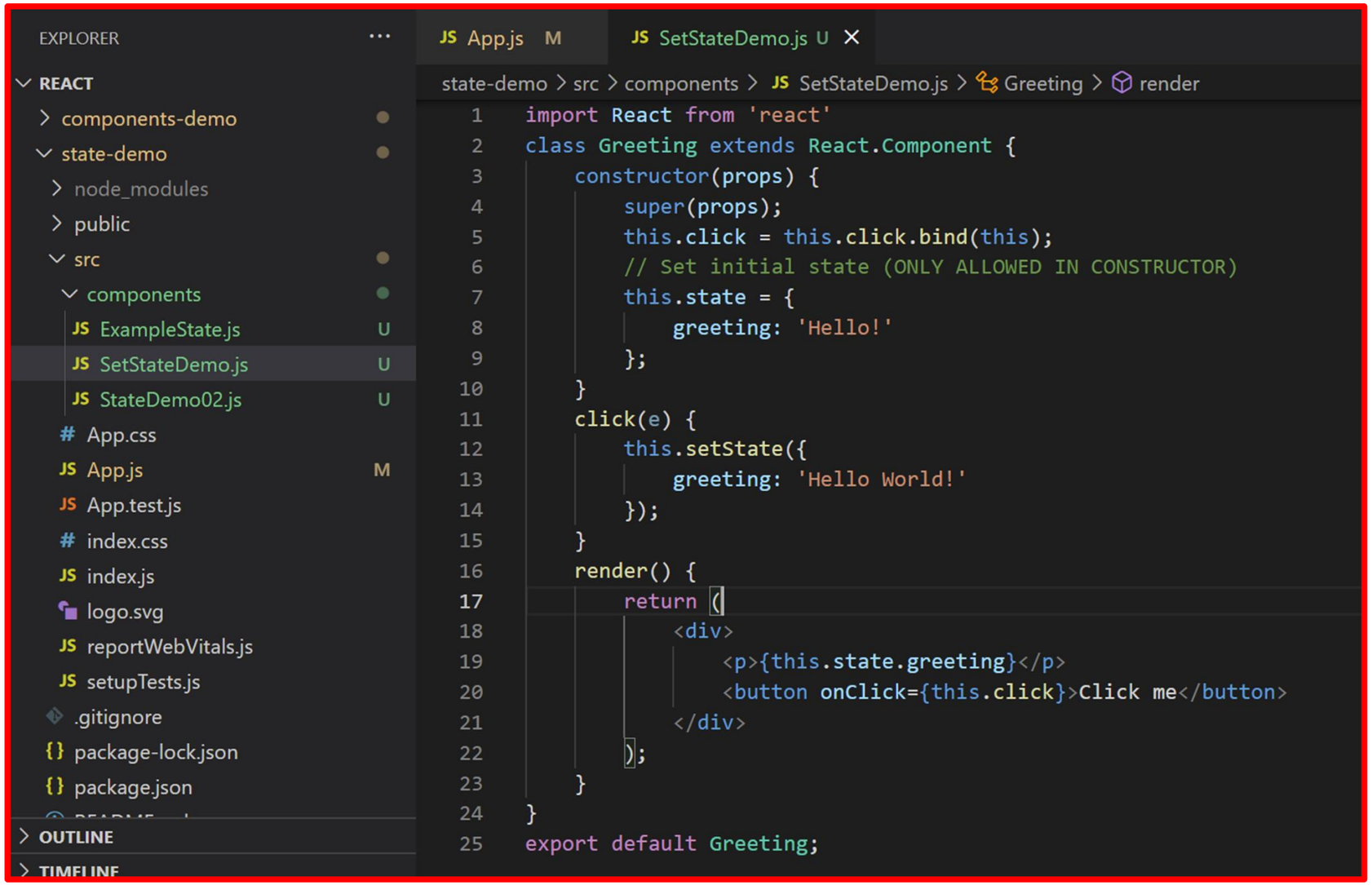
- The primary way that you make UI updates to your React applications is through a call to the setState() function.
- This function will perform a *shallow merge* between the new state that you provide and the previous state, and will trigger a re-render of your component and all decedents

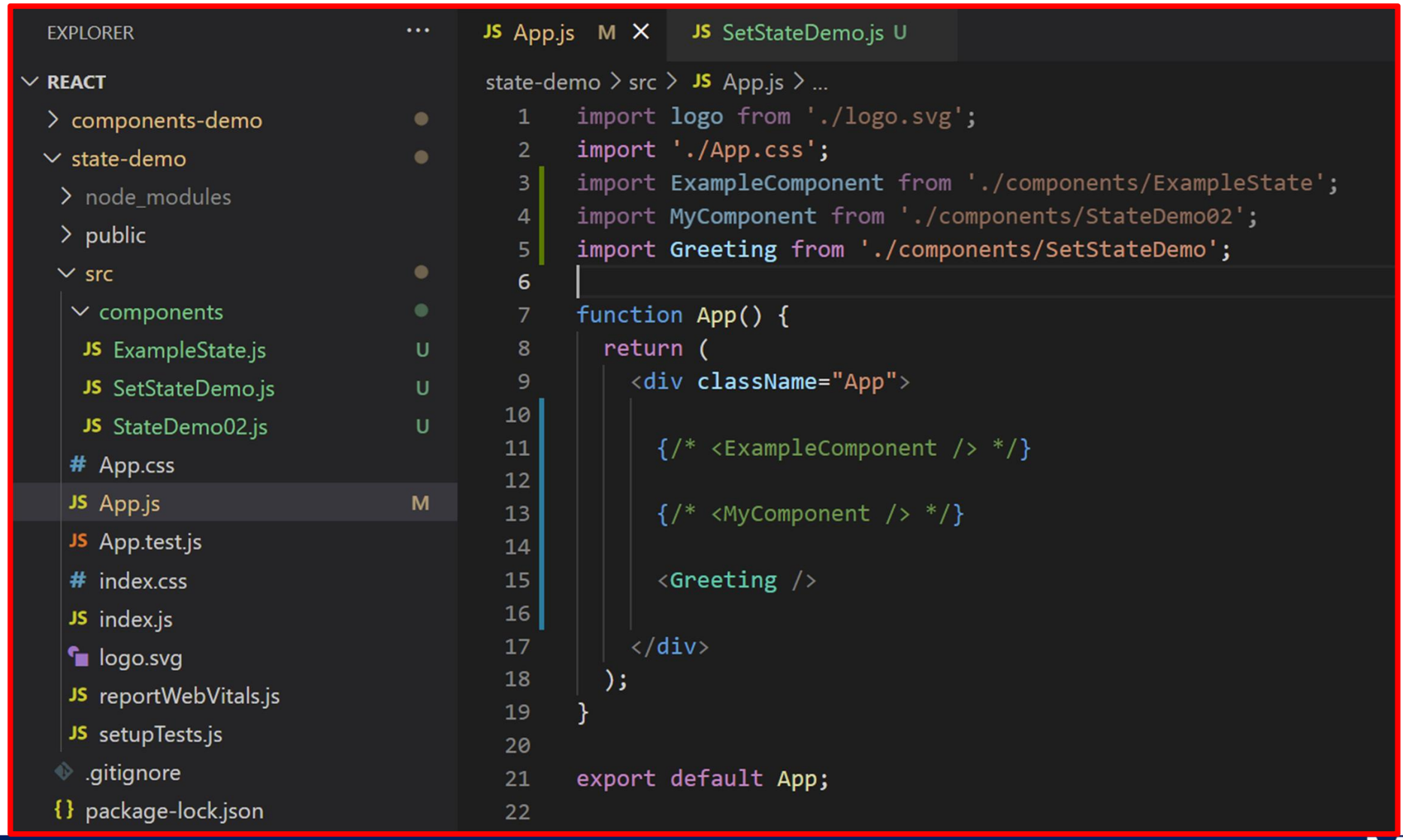
Parameters

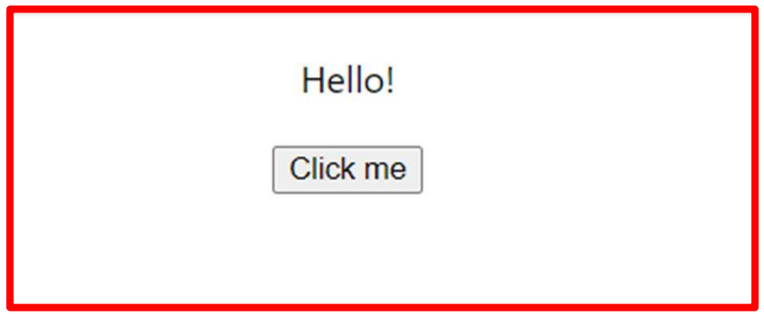
1. updater: It can be an object with a number of key-value pairs that should be merged into the state or a function that returns such an object.

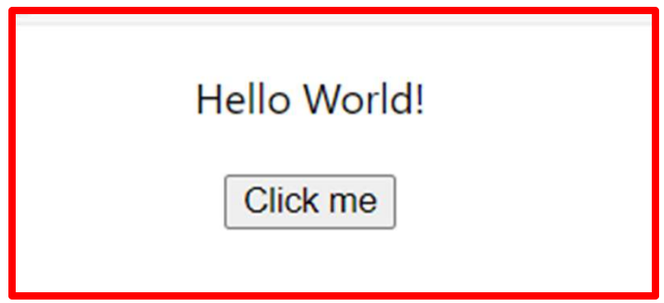
2. callback (optional): a function which will be executed after setState() has been executed successfully.

Due to the fact that calls to setState() are not guaranteed by React to be atomic, this can sometimes be useful if you want to perform some action after you are positive that setState() has been executed successfully.


EXPLORER
REACT
components-demo
state-demo
node_modules
public
src
components
ExampleState.js U
SetStateDemo.js U
StateDemo02.js U
App.css
App.js M
App.test.js
index.css
index.js
logo.svg
reportWebVitals.js
setupTests.js
.gitignore
package-lock.json
package.json
OUTLINE
TIMELINE
App.js M
SetStateDemo.js U
state-demo > src > components > SetStateDemo.js > Greeting > render
1 import React from 'react'
2 class Greeting extends React.Component {
3 constructor(props) {
4 super(props);
5 this.click = this.click.bind(this);
6 // Set initial state (ONLY ALLOWED IN CONSTRUCTOR)
7 this.state = {
8 greeting: 'Hello!'
9 };
10 }
11 click(e) {
12 this.setState({
13 greeting: 'Hello World!'
14 });
15 }
16 render() {
17 return (
18
19 {this.state.greeting}
20 <button onClick={this.click}>Click me</button>
21
22);
23 }
24 }
25 export default Greeting;


EXPLORER
REACT
components-demo
state-demo
node_modules
public
src
components
ExampleState.js U
SetStateDemo.js U
StateDemo02.js U
App.css
App.js M
App.test.js
index.css
index.js
logo.svg
reportWebVitals.js
setupTests.js
.gitignore
package-lock.json
App.js M
SetStateDemo.js U
state-demo > src > App.js > ...
1 import logo from './logo.svg';
2 import './App.css';
3 import ExampleComponent from './components/ExampleState';
4 import MyComponent from './components/StateDemo02';
5 import Greeting from './components/SetStateDemo';
6
7 function App() {
8 return (
9
10
11 {/* <ExampleComponent /> */}
12
13 {/* <MyComponent /> */}
14
15 <Greeting />
16
17
18);
19 }
20
21 export default App;
22


Hello!
Click me


Hello World!
Click me

Using setState() with a Function as updater

```
//
// This is most often used when you want to check or make use
// of previous state before updating any values.
//

this.setState(function(previousState, currentProps) {
  return {
    counter: previousState.counter + 1
  };
});
```

```
this.setState({ counter: this.state.counter + 1 });
```

Calling setState() with an Object and a callback function

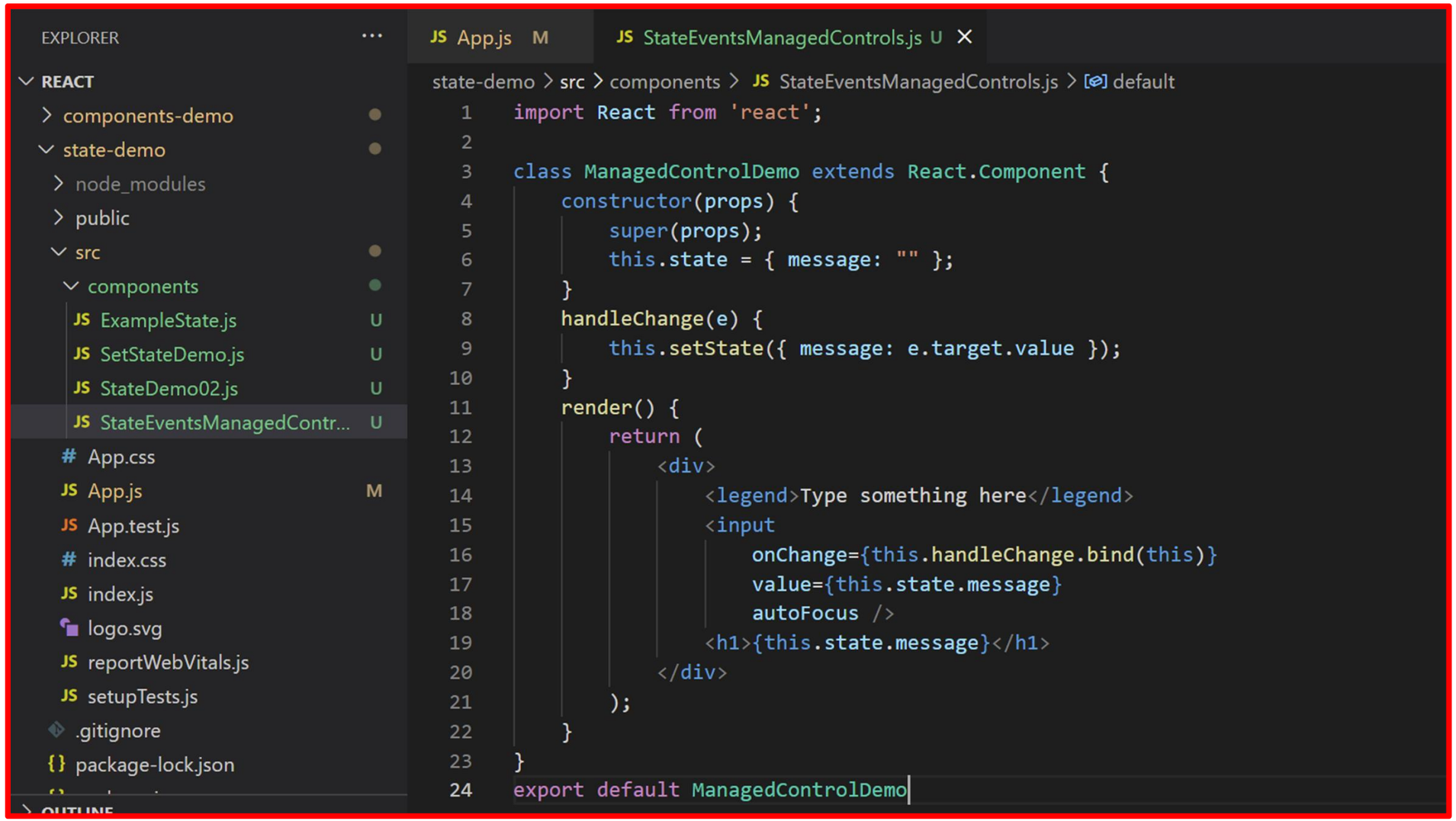
Calling `setState()` with an Object and a callback function

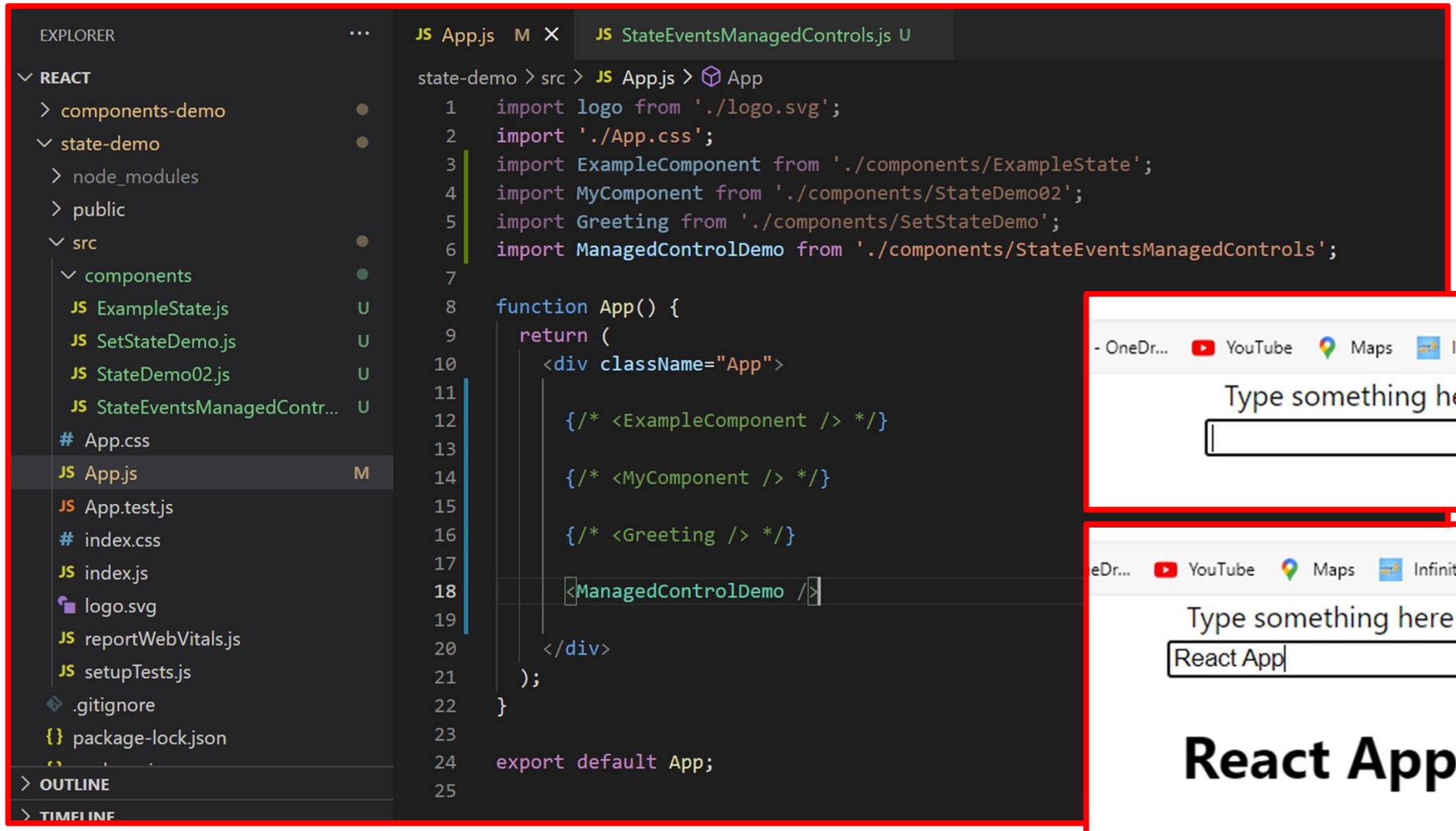
```
//
// 'Hi There' will be logged to the console after setState completes
//

this.setState({ name: 'John Doe' }, console.log('Hi there'));
```

State, Events And Managed Controls

- A React component with a "managed" input field. Whenever the value of the input field
- changes, an event handler is called which updates the state of the component with the new value of the input field.
- The call to setState in the event handler will trigger a call to render updating the component in the dom.


EXPLORER
App.js M
StateEventsManagedControls.js U
state-demo > src > components > StateEventsManagedControls.js > default
REACT
components-demo
state-demo
node_modules
public
src
components
ExampleState.js U
SetStateDemo.js U
StateDemo02.js U
StateEventsManagedContr... U
App.css
App.js M
App.test.js
index.css
index.js
logo.svg
reportWebVitals.js
setupTests.js
.gitignore
package-lock.json
OUTLINE
1 import React from 'react';
2
3 class ManagedControlDemo extends React.Component {
4 constructor(props) {
5 super(props);
6 this.state = { message: "" };
7 }
8 handleChange(e) {
9 this.setState({ message: e.target.value });
10 }
11 render() {
12 return (
13
14 <legend>Type something here</legend>
15 <input
16 onChange={this.handleChange.bind(this)}
17 value={this.state.message}
18 autoFocus />
19 <h1>{this.state.message}</h1>
20
21);
22 }
23 }
24 export default ManagedControlDemo


EXPLORER
REACT
components-demo
state-demo
node_modules
public
src
components
ExampleState.js U
SetStateDemo.js U
StateDemo02.js U
StateEventsManagedContr... U
App.css
App.js M
App.test.js
index.css
index.js
logo.svg
reportWebVitals.js
setupTests.js
.gitignore
package-lock.json
OUTLINE
TIMELINE
App.js M
StateEventsManagedControls.js U
state-demo > src > App.js > App
1 import logo from './logo.svg';
2 import './App.css';
3 import ExampleComponent from './components/ExampleState';
4 import MyComponent from './components/StateDemo02';
5 import Greeting from './components/SetStateDemo';
6 import ManagedControlDemo from './components/StateEventsManagedControls';
7
8 function App() {
9 return (
10
11
12 {/* <ExampleComponent /> */}
13
14 {/* <MyComponent /> */}
15
16 {/* <Greeting /> */}
17
18 <ManagedControlDemo />
19
20
21);
22 }
23
24 export default App;
25
- OneDr... YouTube Maps Infinity Meta Jr
Type something here
eDr... YouTube Maps Infinity Meta Jr
Type something here
React App
React App

- Its very important to note the runtime behavior. Every time a user changes the value in the input field
  1. handleChange will be called and so
  2. setState will be called and so
  3. render will be called

Summary

In this lesson, you should have learned how to:

- Basics in State
- setState
- State, Events and Managed Controls