# Module 2: Components and Styling the Application Layout

## Demo Document 3

# Unmount Component Lifecycle

Step 1: Start building the application using: ***using create-react-app unmount***

Step 2: Change the location and navigate to unmount directory using: ***cd unmount***

Step 3: Delete the all other available files and place only ***index.js file.***

Step 4: In ***index.js file*** start writing the code, first ***import*** the required libraries to execute the application.

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

Step 5: Create a ***class-based component*** and define a constructor in it.

```
class Lifecycle extends React.Component {
   constructor(props) {
      super(props);
```

Step 6: Set the ***initial state*** of component to zero.

```
      this.state = {
         data: 0
      }
```

Step 7: Set the ***final state***.

```
      this.setNewValue = this.setNewValue.bind(this)
   };
   setNewValue() {
      this.setState({data: this.state.data + 1})
   }
```

Data binding to get new sate

Increments the value by 1 on clicking the button

Step 8: Define another component to ***carry the new value*** on clicking the button.

```
class Content extends React.Component {

   render() {
      return (
         <div>
            <h3>{this.props.Number}</h3>
         </div>
      );
   }
}
```

Step 9: Define a ***render() function***.

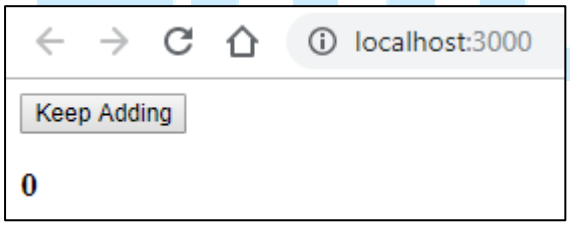
```
render() {
      return (
         <div>
            <button onClick = {this.setNewValue}>Keep Adding</button>
            <Content Number = {this.state.data}></Content>
         </div>
      );
   }
}

ReactDOM.render(<Lifecycle/>, document.getElementById('root'))
```

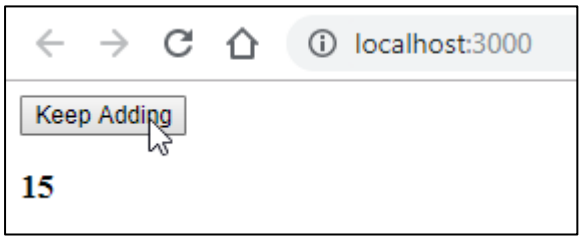
Step 10: Define a ***timer*** using ***unmount lifecycle***, to delete the component by itself after certain duration.

```
setTimeout(() => {
    ReactDOM.unmountComponentAtNode(document.getElementById('root'));}, 10000);
```

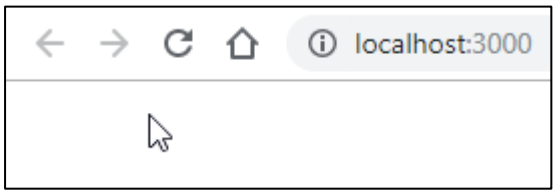
Step 11: Save your application code and start it using ***npm start***. Check its working at localhost: 3000.




Step 11: Keep clicking the button. You will see the component will display the number on every new click.




Till the duration of 1000 seconds component displays number. After that the component disappears.



## Conclusion:

We have successfully ***updated the state*** of a component on click of button and deleted the component by itself using ***unmount*** component lifecycle.