React JS

Juliane Marubayashi

2022-07-14

Contents

Attention before reading: this document is not original. It is deeply based on the resources. Thus, I don't have any credit on the biggest part of this text.

1 React lifecycle

React components can go in four different of its life.

- **Initialization**: In this stage the component is constructed with the given **props** and default state. This is done in the constructor of the component class.
- **Mounting**: Is the stage of rendering the JSX returned by the render method itself.
- **Updating**: The stage the application state is updated and the application is repainted.
- **Unmounting**: Is the final step where the component will be removed from the page.

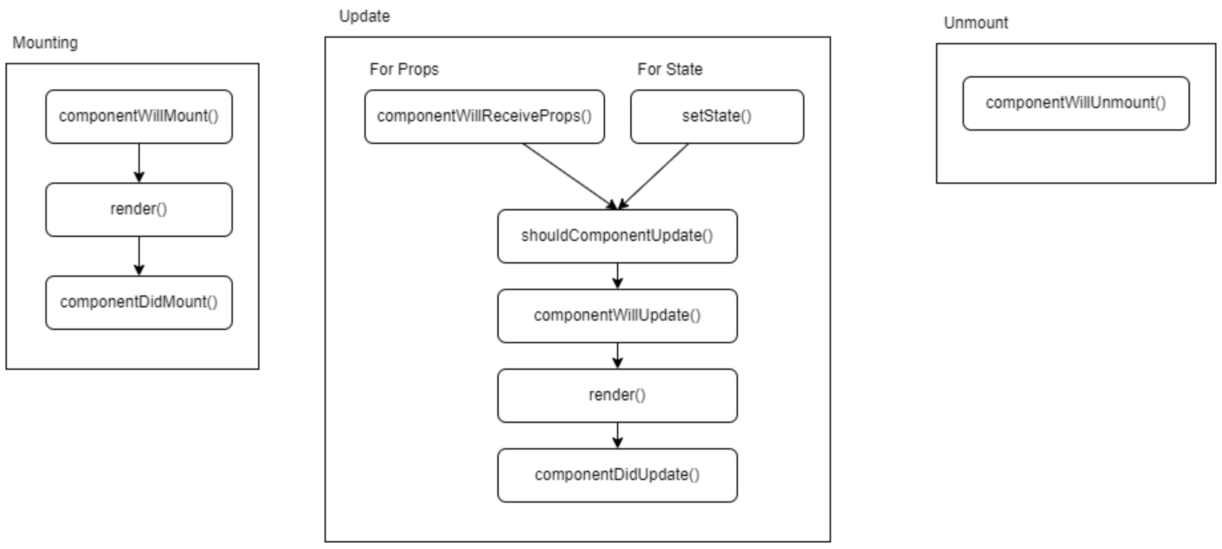

Figure 1: React lifecycle

https://www.geeksforgeeks.org/reactjs-lifecycle-components/

2 React Use Effect

You can think of `useEffect` as `componentDidMount`, `componentDidUpdate`, and `componentWillUnmount` combined.

2.1 Effect without cleanup

Sometimes, we want to run some additional code **after** React component has updated the DOM. In React the `render` method, shouldn't cause any side effect. We usually want to perform effect **after** React has updated the DOM.

This is why usually we put side effects in `componentDidMount` and `componentDidUpdate`.

An example is:

```
class Example extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      count: 0
    };
  }
  componentDidMount() {
    document.title = `You clicked ${this.state.count} times`;
  }
  componentDidUpdate() {
    document.title = `You clicked ${this.state.count} times`;
  }

  render() {
    return (
      <div>
        <p>You clicked {this.state.count} times</p>
        <button onClick={() => this.setState({ count: this.state.count + 1 })}>
          Click me
        </button>
      </div>
    );
  }
}
```