



Skills
Network

React components

Module 2 React components

Package/Method	Description	Code Example
React state	The state object is where you keep the component's property values.	<pre> 1. 1 2. 2 3. 3 4. 4 5. 5 6. 6 7. 7 8. 8 9. 9 10. 10 1. class TestComponent extends React.Component { 2. constructor() { 3. this.state = { 4. id: 1, 5. name: "John", 6. age: 28 7. }; 8. } 9. render() { 10. return ({this.state.name}{this.state.age}) 10. }} </pre> <div>Copied!</div> <pre> 1. 1 2. 2 3. 3 4. 4 5. 5 6. 6 7. 7 </pre>
Props	Props is short for properties and it is used to pass data between React components.	<pre> 1. class TestComponent extends React.Component { 2. render() { 3. return 4. Hi {this.props.name} 5. //passing the props as examples to the test component 6. TestComponentname='John' 7. TestComponentname='Jill' </pre> <div>Copied!</div> <pre> 1. 1 2. 2 3. 3 4. 4 5. 5 6. 6 7. 7 8. 8 9. 9 10. 10 11. 11 12. 12 13. 13 14. 14 15. 15 16. 16 </pre>
mounting	When a component instance is created and added to the DOM, these methods are invoked in the following order: constructor(), getDerivedStateFromProps(), render(), componentDidMount().	<pre> 1. class Header extends React.Component { 2. constructor(props) { 3. super(props); 4. console.log("Inside the constructor") 5. } 6. componentDidMount = () => { 7. console.log("Inside component did mount") 8. } 9. render() { 10. console.log("Inside render method") 11. return (12. The component is rendered 13.) 14. } 15. } 16. export default App </pre>
updating	When a component is updated, five methods are called in the same order: getDerivedStateFromProps(), shouldComponentUpdate(), render(), getSnapshotBeforeUpdate(), componentDidUpdate()	<div>Copied!</div> <pre> 1. 1 2. 2 3. 3 4. 4 5. 5 6. 6 7. 7 8. 8 9. 9 10. 10 </pre>

```
11. 11
12. 12
13. 13
14. 14
15. 15
16. 16
17. 17
18. 18
19. 19
20. 20
21. 21
22. 22

1. class App extends React.Component{
2. state = {counter: "0"};
3. incrementCounter = () => this.setState({counter:parseInt(this.state.coun
4. shouldComponentUpdate(){
5.   console.log("Inside shouldComponent update")
6.   return true;
7. }
8. getSnapshotBeforeUpdate(prevProps,prevState){
9.   console.log("Inside getSnapshotBeforeUpdate")
10.  console.log("Prev counter is" +prevState.counter);
11.  console.log("New counter is" +this.state.counter);
12.  return prevState
13. }
14. componentDidUpdate(){
15.   console.log("Inside componentDidUpdate")
16. }
17. render(){
18.   console.log("Inside render")
19.   return(<button onClick={this.incrementCounter}>Click Me!</button>{this
20. })
21. }}
22. export default App;
```

Copied!

```
1. 1
2. 2
3. 3
4. 4
5. 5
6. 6
7. 7
8. 8
9. 9
10. 10
11. 11
12. 12
13. 13
14. 14
15. 15
16. 16
17. 17
18. 18
19. 19
20. 20
21. 21
22. 22
23. 23
```

Unmounting

When a component is removed or unmounted from the DOM, the `componentWillUnmount()` method enables us to run React code.

```
1. import React from 'react';
2. class ComponentOne extends React.Component {
3.   componentWillMount() {
4.     console.log('The component is going to be unmounted');
5.   }
6.   render() {
7.     return Inner component;
8.   }
9. }
10. class App extends React.Component {
11. state = { innerComponent:<AppInner/>};
12. componentDidMount() {
13.   setTimeout(()=>{
14.     this.setState({ innerComponent:unmounted});
15.   },5000)
16. }
17. render() {
18.   console.log("Inside render")
19.   return (
20.     {this.state.innerComponent});
21. }
22. }
23. export default App;
```

Copied!

Changelog

Date	Version	Changed by	Change Description
20-10-2022	1.1	Sapthashree K S	Cheatsheet updated

