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.	class TestComponentextendsReact.component{
		constructor() {
		this.state= {
		id: 1,
		name: “John”
		age: 28
		};}
		render() {
		return ({this.state.name}{this.state.age}
		)}}
		
		Copied!

Props	Props is short for properties and it is used to pass data between React components.	class TestComponentextendsReact.component{
		render() {
		return
		Hi {this.props.name}}
		//passing the props as examples to the test component
		TestComponentname=‘John’
		TestComponentname=‘Jill’
		
		Copied!

mounting	When a component instance is created and added to the DOM, these methods are invoked in the following order: constructor(),getDerivedStateFromProps(), render(),componentDidMount().	class Header extends React.Component {
		constructor(props) {
		super(props);
		console.log("Inside the constructor")
		}
		componentDidMount =()=> {
		console.log("Inside component did mount")
		}
		render() {
		console.log("Inside render method")
		return (
		The component is rendered
		)
		}
		}
		export default App
		
		Copied!

updating	When a component is updated,five methods are called in the same order: getDerivedStateFromProps(),shouldComponentUpdate(),render(), getSnapshotBeforeUpdate(),componentDidUpdate()	

```
 class App extends React.Component{
state = {counter: "0"};
incrementCounter = () => this.setState({counter:parseInt(this.state.coun
shouldComponentUpdate(){
 console.log("Inside shouldComponent update")
  return true;
}
getSnapshotBeforeUpdate(prevProps,prevState){
console.log("Inside getSnapshotBeforeUpdate")
console.log("Prev counter is" +prevState.counter);
console.log("New counter is" +this.state.counter);
return prevState
}
componentDidUpdate(){
  console.log("Inside componentDidUpdate")
}
render(){
  console.log("Inside render")
  return(<button onClick={this.incrementCounter}>Click Me!</button>{this
)
}}
export default App;
```

Copied!

Unmounting

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

```
 import React from 'react';
class ComponentOne extends React.Component {
componentWillUnmount() {
console.log('The component is going to be unmounted');
}
render() {
return Inner component;
}
}
class App extends React.Component {
state = { innerComponent:<AppInner/>};
componentDidMount() {
setTimerout(()=>{
this.setState({ innerComponent:unmounted});
},5000)
}
render() {
console.log("Inside render")
return (
{this.state.innerComponent});
}
}
export default App;
```

Copied!

Changelog

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