React Conditional Rendering

In React, we can create multiple components which encapsulate behavior that we need. After that, we can render them depending on some conditions or the state of our application. In other words, based on one or several conditions, a component decides which elements it will return. In React, conditional rendering works the same way as the conditions work in JavaScript. We use JavaScript operators to create elements representing the current state, and then React Component update the UI to match them.

From the given scenario, we can understand how conditional rendering works. Consider an example of handling a **login/logout** button. The login and logout buttons will be separate components. If a user logged in, render the **logout component** to display the logout button. If a user not logged in, render the **login component** to display the login button. In React, this situation is called as **conditional rendering**.

There is more than one way to do conditional rendering in React. They are given below.

* if
* ternary operator
* logical && operator
* switch case operator
* Conditional Rendering with enums

Ex: conditional rendering using IF statement

**function UserLoggin(props) {**

**return <h1>Welcome back!</h1>;**

**}**

**function GuestLoggin(props) {**

**return <h1>Please sign up.</h1>;**

**}**

**function SignUp(props) {**

**const isLoggedIn = props.isLoggedIn;**

**if (isLoggedIn) {**

**return <UserLogin />;**

**}**

**return <GuestLogin />;**

**}**

**====should be added in App.js inside render()**

**<SignUp isLoggedIn={false} />,**

**Ex: rendering using Ternary operator**

**import React, { Component } from 'react';**

**function Message(props)**

**{ if (props.isLoggedIn) //using IF**

**return <h1>Welcome Back!!!</h1>;**

**else**

**return <h1>Please Login First!!!</h1>;**

**}**

**function Login(props)**

**{ return(**

**<button onClick = {props.clickInfo}> Login </button>**

**);**

**}**

**function Logout(props)**

**{ return(**

**<button onClick = {props.clickInfo}> Logout </button>**

**);**

**}**

**class App extends Component{**

**constructor(props)**

**{ super(props);**

**this.handleLogin = this.handleLogin.bind(this);**

**this.handleLogout = this.handleLogout.bind(this);**

**this.state = {isLoggedIn : false};**

**}**

**handleLogin()**

**{**

**this.setState({isLoggedIn : true});**

**}**

**handleLogout()**

**{**

**this.setState({isLoggedIn : false});**

**}**

**render(){**

**return(**

**<div>**

**<h1> Conditional Rendering Example </h1>**

**<Message isLoggedIn = {this.state.isLoggedIn}/>**

**{**

**(this.state.isLoggedIn)?(**

**<Logout clickInfo = {this.handleLogout} />**

**) : (**

**<Login clickInfo = {this.handleLogin} />**

**)**

**}**

**</div>**

**);**

**}**

**}**

**export default App;**

**Ex: using &&**

**// Example Component**

**function Example()**

**{**

**return(<div>**

**{**

**(1 > 5) && alert('This alert will be shown!')**

**}**

**</div>**

**);**

**}**

**export default App;**

**Ex:**

**class Book extends Component {**

**constructor(props) {**

**super(props);**

**this.state = {**

**show: false**

**};**

**}**

**toggleInfo = () => {**

**this.setState({show: !this.state.show});**

**}**

**render() {**

**const {book, author} = this.props.info;**

**return (**

**<article>**

**<h3>book: {book}</h3>**

**<h5>author: {author}</h5>**

**<button type="button" onClick={this.toggleInfo}>Toggle Paragraph</button>**

**{**

**this.state.show && <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Ipsam, nobis!</p>**

**}**

**</article>**

**)**

**}**

**}**