npx create-react-app react-conditional-rendering-example

cd react-conditional-rendering-example

**App.js**

import React, { Component } from "react";

import './App.css';

class App extends Component {

  constructor(props) {

    super(props);

    this.state = {

      isLoggedIn: true

    };

  }

  render() {

    return (

      <div className="App">

        <h1>

          This is a Demo showing several ways to implement Conditional Rendering in React.

        </h1>

        <button>Login</button>

        <button>Logout</button>

      </div>

    );

  }

}

export default App;

**App.css**

body {

  padding: 1em;

}

h1 {

  font-size: 3em;

  font-weight: 500;

  text-align: center;

  margin-bottom: 1em;

  margin-right: 1em;

  padding: 0;

}

button {

  appearance: none;

  background-color: #246bec;

  border: 1px solid #246bec;

  border-radius: 0;

  box-sizing: border-box;

  color: #ffffff;

  display: block;

  font-size: 2em;

  font-weight: 500;

  margin-bottom: 1em;

  margin-top: 1em;

  padding: .5em;

  width: 100%;

  transition: border-color, background-color 300ms ease-in-out;

}

button:focus {

  border-color: #00006D;

}

button:hover {

  background-color: #0B52D3;

}

button:active {

  background-color: #00006D;

}

At this point, you will have a React application that displays a **Login** and a **Logout** button. Your goal is to have only one of these buttons display. Let’s look at conditional rendering approaches to accomplish this.

**Only Display Logout Button**

import React, { Component } from "react";

import './App.css';

class App extends Component {

  constructor(props) {

    super(props);

    this.state = {

      isLoggedIn: true

    };

  }

  render() {

    let {isLoggedIn} = this.state;

    const renderAuthButton = () => {

      if (isLoggedIn) {

        return <button>Logout</button>;

      } else {

        return <button>Login</button>;

      }

    }

    return (

      <div className="App">

        <h1>

          This is a Demo showing several ways to implement Conditional Rendering in React.

        </h1>

        {renderAuthButton()}      </div>

    );

  }

}

export default App;

**Create one Component**

**AuthButton .js**

import React from "react";

const AuthButton = props => {

  let { isLoggedIn } = props;

  if (isLoggedIn) {

    return <button>Logout</button>;

  } else {

    return <button>Login</button>;

  }

};

export default AuthButton;

**import it in App.js**

import AuthButton from "./AuthButton";

// import React, { Component } from "react";

// import './App.css';

// class App extends Component {

//   constructor(props) {

//     super(props);

//     this.state = {

//       isLoggedIn: true

//     };

//   }

//   render() {

//     let {isLoggedIn} = this.state;

//     const renderAuthButton = () => {

//       if (isLoggedIn) {

//         return <button>Logout</button>;

//       } else {

//         return <button>Login</button>;

//       }

//     }

//     return (

//       <div className="App">

//         <h1>

//           This is a Demo showing several ways to implement Conditional Rendering in React.

//         </h1>

//         {renderAuthButton()}      </div>

//     );

//   }

// }

// export default App;

import React, { Component } from "react";

import './App.css';

import AuthButton from "./AuthButton";

class App extends Component {

  constructor(props) {

    super(props);

    this.state = {

      isLoggedIn: true

    };

  }

  render() {

    return (

      <div className="App">

        <h1>

          This is a Demo showing several ways to implement Conditional Rendering in React.

        </h1>

        <AuthButton isLoggedIn={this.state.isLoggedIn} />

      </div>

    );

  }

}

export default App;

**Using a switch Statement**

**AuthButton1.js**

import React from "react";

const AuthButton1 = props => {

  let { isLoggedIn } = props;

  switch (isLoggedIn) {

    case true:

      return <button>Logout</button>;

      break;

    case false:

      return <button>Login</button>;

      break;

    default:

      return null;

  }

};

export default AuthButton1;

Calling part is same in App.js file

## Using Element Variables

## No need to make separate Components

import AuthButton from "./AuthButton";

import React, { Component } from "react";

import './App.css';

class App extends Component {

  constructor(props) {

    super(props);

    this.state = {

      isLoggedIn: true

    };

  }

  render() {

    let { isLoggedIn } = this.state;

    let AuthButton;

    if (isLoggedIn) {

      AuthButton = <button>Logout</button>;

    } else {

      AuthButton = <button>Login</button>;

    }

    return (

      <div className="App">

        <h1>

          This is a Demo showing several ways to implement Conditional Rendering in React.

        </h1>

        {AuthButton}

      </div>

    );

  }

}

export default App;

## Using Ternary Operators

import React, { Component } from "react";

import './App.css';

class App extends Component {

  constructor(props) {

    super(props);

    this.state = {

      isLoggedIn: false

    };

  }

  render() {

    let { isLoggedIn } = this.state;

    return (

      <div className="App">

        <h1>

          This is a Demo showing several ways to implement Conditional Rendering in React.

        </h1>

        {isLoggedIn ? <button>Logout</button> : <button>Login</button>}

      </div>

    );

  }

}

export default App;

## Using Logical && (Short Circuit Evaluation)

import React, { Component } from "react";

import './App.css';

class App extends Component {

  constructor(props) {

    super(props);

    this.state = {

      isLoggedIn: true

    };

  }

  render() {

    let { isLoggedIn } = this.state;

    return (

      <div className="App">

        <h1>

          This is a Demo showing several ways to implement Conditional Rendering in React.

        </h1>

        {isLoggedIn && <button>Logout</button>}

      </div>

    );

  }

}

export default App;

**SECOND DEMO**

import React from 'react';

import ReactDOM from 'react-dom';

// Message Component

function Message(props)

{

  if (props.isLoggedIn)

    return <h1>Welcome User</h1>;

  else

    return <h1>Please Login</h1>;

}

// Login Component

function Login(props)

{

return(

    <button onClick = {props.clickFunc}>

      Login

    </button>

  );

}

// Logout Component

function Logout(props)

{

  return(

    <button onClick = {props.clickFunc}>

      Logout

    </button>

  );

}

// Parent Homepage Component

class Homepage extends React.Component{

  constructor(props)

  {

    super(props);

    this.state = {isLoggedIn : false};

    this.ifLoginClicked = this.ifLoginClicked.bind(this);

    this.ifLogoutClicked = this.ifLogoutClicked.bind(this);

  }

  ifLoginClicked()

  {

    this.setState({isLoggedIn : true});

  }

  ifLogoutClicked()

  {

    this.setState({isLoggedIn : false});

  }

  render(){

    return(

      <div>

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

        {

          (this.state.isLoggedIn)?(

          <Logout clickFunc = {this.ifLogoutClicked} />

          ) : (

          <Login clickFunc = {this.ifLoginClicked} />

          )

        }

      </div>

      );

  }

}

ReactDOM.render(

  <Homepage />,

  document.getElementById('root')

);

**THIRD DEMO**

import React from 'react';

import ReactDOM from 'react-dom';

// Example Component

function Example()

{

  const counter = 5;

  return(<div>

      {

        (counter==5) &&

        <h1>Hello World!</h1>

      }

    </div>

    );

}

ReactDOM.render(

  <Example />,

  document.getElementById('root')

);

You can clearly see in the above output that as the condition (counter == 5) evaluates to true so the <h1> element is successfully rendered on the screen.

**Preventing Component from Rendering**

import React from 'react';

import ReactDOM from 'react-dom';

// Example Component

function Example(props)

{

    if(!props.toDisplay)

        return null;

    else

        return <h1>Component is rendered</h1>;

}

ReactDOM.render(

    <div>

        <Example toDisplay = {true} />

        <Example toDisplay = {false} />

    </div>,

    document.getElementById('root')

);

You can clearly see in the above output that the *Example* component is rendered twice but the <h1> element is rendered only once as on the second render of the *Example* component, *null* is returned as its rendering output.