Conditional Rendering in React

In this lecture, we will learn how to show or hide parts of the user interface based on conditions. This is called conditional rendering.

Using if Statements

You can use a normal JavaScript `if` statement inside a component:

```
function App() {
  const isLoggedIn = true

  if (isLoggedIn) {
    return <h2>Welcome back!</h2>
  } else {
    return <h2>Please log in.</h2>
  }
}

export default App
```

Here, the UI changes depending on the value of `isLoggedIn`.

Using the Ternary Operator

A shorter way to write conditional logic is the ternary operator:

```
function App() {
  const isLoggedIn = false

  return (
    <div>
      {isLoggedIn ? <h2>Welcome back!</h2> : <h2>Please log in.</h2>}
```

```
    </div>
  )
}
```

Using Logical AND (&&)

When you only want to render something if a condition is true, you can use the logical AND operator:

```
function App() {
  const showMessage = true

  return (
    <div>
      <h1>Hello User</h1>
      {showMessage && <p>This is a special message!</p>}
    </div>
  )
}
```

If `showMessage` is `false`, the paragraph will not be rendered.

Example: Toggle Visibility

Let's combine state and conditional rendering:

```
import { useState } from "react"

function App() {
  const [visible, setVisible] = useState(true)

  return (
    <div>
      <button onClick={() => setVisible(!visible)}>
        {visible ? "Hide" : "Show"}
      </button>
```

```
      {visible && <p>This text can be toggled on and off.</p>}
    </div>
  )
}

export default App
```

Clicking the button toggles the paragraph on and off.

Recap

- Use `if` statements, the ternary operator, or logical AND (`&&`) for conditional rendering.
- Conditional rendering allows the UI to change based on state or props.
- You can combine state and conditions to build interactive features.