**[Bài đọc] State**

1. **Khái niệm**

* Trong ReactJS, state trong class component là một đối tượng chứa dữ liệu có thể thay đổi và ảnh hưởng đến việc hiển thị của component.
* Khi trạng thái (state) của một class component thay đổi, React sẽ tự động render lại component để phản ánh sự thay đổi này.

1. **Cú pháp khai báo state**

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

**class Counter extends Component {**

**constructor(props) {**

**super(props);**

**// Khởi tạo state**

**this.state = {**

**count: 0,**

**};**

**}**

**render() {**

**return (**

**<div>**

**<p>Count: {this.state.count}</p>**

**</div>**

**);**

**}**

**}**

**export default Counter;**

* Trong phương thức constructor, this.state được sử dụng để khởi tạo trạng thái ban đầu của component. Trong trường hợp này, count được khởi tạo với giá trị là 0.
* Trong phương thức render, giá trị của count được hiển thị trong JSX.

1. **Cập nhật state**

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

**class Counter extends Component {**

**constructor(props) {**

**super(props);**

**// Khởi tạo state**

**this.state = {**

**count: 0,**

**};**

**}**

**// Một phương thức tăng giá trị của state.count**

**increaseCount = () => {**

**this.setState({ count: this.state.count + 1 });**

**};**

**// Một phương thức giảm giá trị của state.count**

**decreaseCount = () => {**

**this.setState({ count: this.state.count - 1 });**

**};**

**render() {**

**return (**

**<div>**

**<p>Count: {this.state.count}</p>**

**<button onClick={this.increaseCount}>Increase</button>**

**<button onClick={this.decreaseCount}>Decrease</button>**

**</div>**

**);**

**}**

**}**

**export default Counter;**

* Các phương thức như **increaseCount** và **decreaseCount** sử dụng this.setState để cập nhật giá trị của **count**. **this.setState** là một phương thức được cung cấp bởi React để thông báo rằng trạng thái đã thay đổi và cần được render lại.
* Trong phương thức render, giá trị của count được hiển thị trong JSX. Khi this.setState được gọi, React sẽ tự động gọi lại phương thức render để cập nhật giao diện người dùng.
* Lưu ý rằng việc sử dụng this.state trực tiếp để cập nhật state không được khuyến khích, vì **this.setState** có thể được thực hiện bất đồng bộ. Thay vào đó, bạn nên sử dụng hàm **callback** khi cần sử dụng giá trị trước đó của state. Ví dụ:

**this.setState((prevState) => ({**

**count: prevState.count + 1,**

**}));**