NIIT | ICT Hanoi
A LICENSEE OF NIIT INDIA

Component và Props, State

Giảng viên: **Phạm Nhân Nghĩa**

Email: **nghiapn89@gmail.com**

Các nội dung chính

I. Component

II. Props

III. State

I. Component

1. Giới thiệu Component
2. Function component
3. Class component

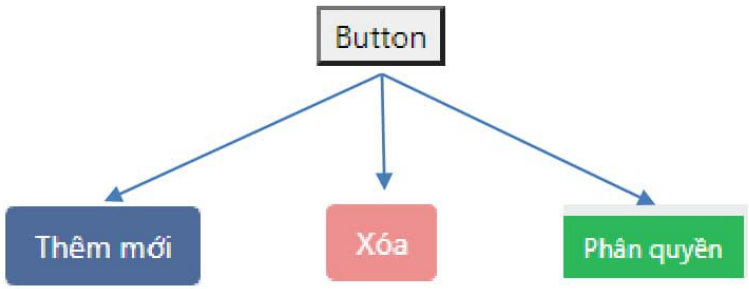
I.1. Giới thiệu Component

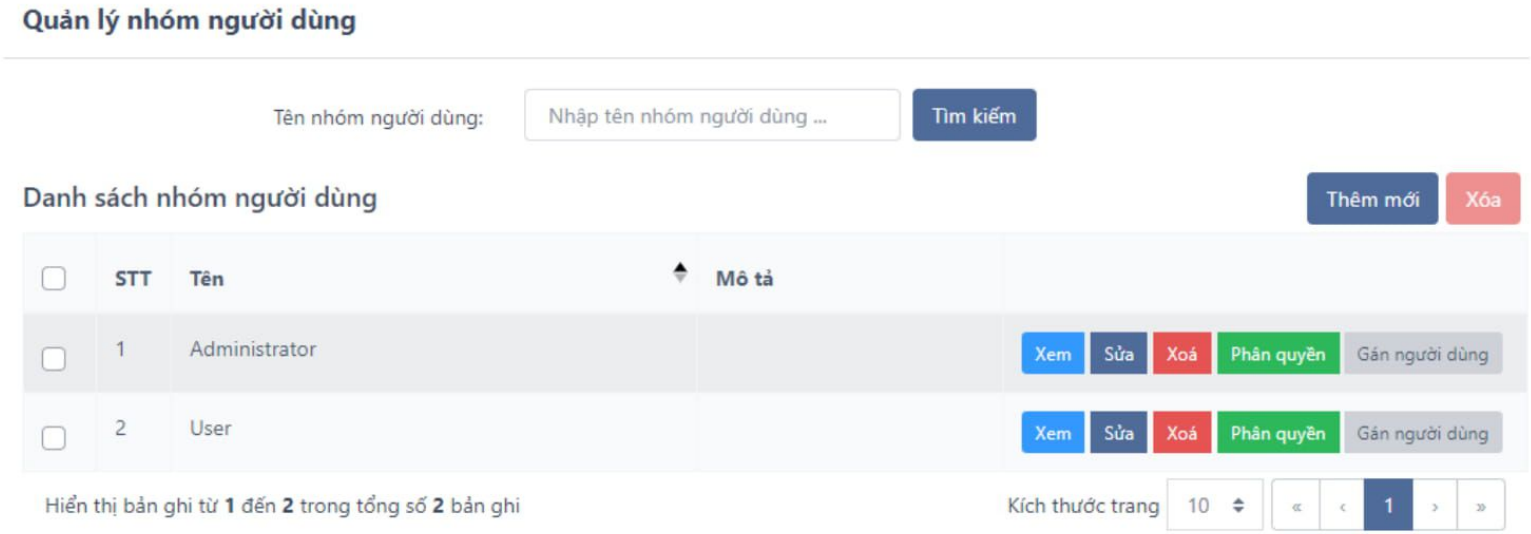
- Định nghĩa Component :
  - Là các thành phần độc lập và có thể sử dụng lại. Dùng để chia UI thành nhiều phần nhỏ
  - Thực hiện công việc giống như các functions trong JavaScript nhưng chúng độc lập và nhiệm vụ chính là trả về HTML thông qua hàm Render
  - Dùng để chia nhỏ code để dễ dàng maintain
  - Có 2 loại component là Function Component và Class Component.

I.1.Giới thiệu Component

- Sự tương đồng Component và Class trong OOP :
  - Tương tự khái niệm Class trong các ngôn ngữ hướng đối tượng
  - Có thể kế thừa
  - Class là định nghĩa trừu tượng
  - Mỗi element là một instance (thể hiện) của một class

I.1.Giới thiệu Component





I.2. Function Component

- *Còn được gọi là* Stateless Component
- *Là cách viết component với pure function*
- *Có thể nhận các tham số truyền vào*
- *Trả về React element thông qua cú pháp return()*
- *Đây là cách viết đơn giản, dễ hiểu*
- *Không có các khái niệm như state, life cycles, events.*

I.2. Function Component

- Cú pháp **Return()** trả về HTML
- Chỉ có thể trả về 1 phần tử HTML
- **Export default** là bắt buộc.

```
import React, { FunctionComponent } from 'react';
export const FunctionComponentExample : FunctionComponent<any> = () => {
    return (
        <h1>Function component!</h1>
    )
}
export default FunctionComponentExample;
```

- Cách viết ngắn gọn hơn

```
export const FunctionComponentExample : FunctionComponent<Props> = () =>
    <h1>Function component!</h1>
```

I.2. Function Component

- Cách sử dụng component
- Cả ứng dụng web của chúng ta là 1 function component lớn trong file App.tsx

```
mport './App.css';

function App() {
  return (
    <div className="App">
    </div>
  );
}
export default App;
```

I.2. Function Component

- Cách sử dụng component
- Sử dụng Import để sử dụng component đã định nghĩa

```
import FunctionComponentExample from './Example/FunctionComponentExample';
```

- Thêm component vào bên trong component App
- Component được viết như một thẻ HTML

```
function App() {
  return (
    <div className="App">
      <FunctionComponentExample/>
    </div>
  );
}
```

I.3. Class Component

- *Thường được gọi là* Stateful component (Tuy nhiên, nếu Class component không có State thì vẫn được gọi là Stateless component)
- *Là cách viết component dưới dạng class*
- Tổ chức code theo cấu trúc mô hình OOP
- *Được sử dụng khi làm việc việc với* các chức năng của component như *events, state, lifce cycles*

I.3. Class Component

- Class **Component** là class mặc định của ReactJS và tất cả các Component khác đều phải extend nó
- Hàm Render() trả về một thẻ HTML
- Hàm Render là bắt buộc phải khai báo

```
import React, { Component } from "react";
class ClassComponentExample extends Component<any,any>{
    constructor(props: any){
        super(props);
        this.state = {
            content : “Class component!"
        };
    }
    render(){
        return(
            <h1>{this.state.content}</h1>
        );
    }
}
export default ClassComponentExample;
```

3.4. Constructor trong Class Component

- Là phương thức để khởi tạo component. Giống khái niệm Contructor trong OOP
- Thực hiện việc thiết lập State cho component.

```
class ClassComponentExample extends Component{
    //Không sử dụng this ở đây
    constructor(props){
        //sử dụng this trong contructor
        super(props);
        this.state = {
            content : "Hello world!"
        };
    }
```

3.4. Constructor trong Class Component

- Gọi super trong Contructor nhằm khởi tạo biến **this** từ **Parent**, vì tất cả component được thừa kế từ **React.Component**.
- Sau khi gọi super, có thể truy cập được **this**.

```
class Component {
    constructor(props) {
      this.props = props;
      // ...
    }
}
```

3.5. So sánh Class vs Function Component

	Class component	Function compnent
Cú pháp	Cấu trúc giống class OOP	Đơn giản, chỉ có function trả về HTML
Constructor	Có	Không
Nhiệm vụ	Nhận props để render UI và bắn event khi cần thiết cho Container components.	Dùng để tạo nên những Container Components, chúng có nhiệm vụ nhận event, quản lí state, logic
Làm việc với *events, state, lifce cycles*	Có	Không

1. Kết hợp các component

Lồng component trong component

```
class Component {
    constructor(props) {
      this.props = props;
      // ...
    }
}
```

1. Kết hợp các component

- Một class có thể kế thừa 1 class khác
- Tuy nhiên cách này không được khuyến khích sử dụng bởi Facebook:
  - Chỉ cần sử dụng props để truyền dữ liệu
  - Props chấp nhận tất cả các kiểu dữ liệu
  - Nếu muốn tái sử dụng các function không liên quan tới giao diện người dùng, nên tách biệt nó ra những module Javascript riêng. Các component có thể nhập nó và sử dụng các hàm, đối tượng hoặc class, mà không phải mở rộng nó.

```
class Dog extends Animal{
    constructor(props:any){
        super(props);
        this.state = {
        };
    }
```

II. Props

1. Giới thiệu Props
2. Cách khai báo props
3. Default Props

II.1. Giới thiệu Props

- Props là viết tắt của Properties (tương đương attributes trong HTML)
- Là thuộc tính tự định nghĩa cho một Component

```
import React, { Component } from "react";
interface Props{
    name?: string;
}
class Student extends Component<any,any>{
    constructor(props: Props){
        super(props);
    }
    render(){
        return(
            <div>
                <h1>Name : {this.props.name}</h1>
            </div>
        );
    }
}
export default Student;
```

II.1. Giới thiệu Props

- Props là một object được truyền vào trong một components, mỗi components sẽ nhận vào props và trả về react element.
- Props có thể là bất kể kiểu dữ liệu nào bao gồm kiểu dữ liệu basic hoặc dữ liệu có cấu trúc, hay một component khác
- Props *cho phép giao tiếp giữa các components với nhau bằng cách truyền tham số qua lại giữa các components*.
- Khi một components cha truyền cho component con một props thì components con chỉ có thể đọc và không có quyền chỉnh sửa nó bên phía components cha.