CHANGED 11 DAYS AGO

Mục tiêu: Xử lý sự kiện, cài đặt thư viện, call API bằng thư viện
Nội dung:
Xử lý sự kiện (event handling)
useEffect hook và lifecycle cơ bản
map, filter trong JSX
Cài đặt thư viện: axios, ...
Thực hành:
Tạo component ProductCard
Form nhập dữ liệu sản phẩm đơn giản

🧠 1. Kiến thức nền tảng

✅ 1.1 Xử lý sự kiện (Event Handling) trong React

📌 Tổng quan

- Trong React, bạn có thể xử lý các sự kiện giống như trong HTML thông thường, nhưng với một số điểm khác biệt:
- Tên sự kiện viết bằng camelCase thay vì chữ thường (onClick, onChange thay vì onclick, onchange)
- Giá trị của event handler là một hàm, không phải chuỗi như trong HTML truyền thống.

📘 Ví dụ cơ bản:

```
function Counter() {
  const [count, setCount] = useState(0);

  return (
    <div>
      <p>Bạn đã nhấn {count} lần</p>
      <button onClick={() => setCount(count + 1)}>Tăng</button>
    </div>
  );
}
```

🔍 Một số sự kiện phổ biến:

Sự kiện	Mô tả
`onClick`	Khi người dùng click vào element
`onChange`	Khi input thay đổi giá trị
`onSubmit`	Khi form được submit
`onMouseEnter`, `onMouseLeave`	Hover chuột vào hoặc rời khỏi element
`onKeyDown`, `onKeyUp`	Khi nhấn hoặc thả phím

⚠️ Lưu ý:
Luôn viết hàm xử lý bên trong component hoặc truyền callback vào.

Không gọi trực tiếp hàm bên trong onClick, mà dùng hàm callback.

```
// Sai
<button onClick={setCount(count + 1)}>Tăng</button>

// Đúng
<button onClick={() => setCount(count + 1)}>Tăng</button>
```

Tránh dùng event.preventDefault() nếu không cần thiết, ví dụ khi không làm việc với form submission.

📘 Ví dụ xử lý form:

```
function ProductForm() {
  const [productName, setProductName] = useState("");

  const handleSubmit = (e) => {
    e.preventDefault(); // Ngăn chặn trong form
    alert("Tên sản phẩm: " + productName);
  };

  return (
    <form onSubmit={handleSubmit}>
      <input
        type="text"
        value={productName}
        onChange={(e) => setProductName(e.target.value)}
      />
      <button type="submit">Thêm sản phẩm</button>
    </form>
  );
}
```

✅ 1.2 Side Effects với useEffect

📌 Side Effect là gì?
"Side Effect" là bất kỳ hành động nào xảy ra ngoài phạm vi render của component, ví dụ:

- Gọi API
- Cập nhật DOM thủ công
- Đăng ký hoặc huỷ đăng ký event listeners
- setTimeout, setInterval
- Cập nhật tiêu đề trang (document.title)

📘 Cú pháp cơ bản của useEffect:

```
useEffect(() => {
  // Side effect ở đây
}, [dependencies]);
```

⚙️ Cách hoạt động:

- Chạy lần đầu tiên sau khi component render.
- Chạy lại khi bất kỳ giá trị nào trong [dependencies] thay đổi.
- Nếu [ ] là mảng rỗng, chỉ chạy một lần như componentDidMount.
- Có thể return một hàm để làm cleanup (giống componentWillUnmount trong class).

Loại 1: []
Loại 2: [dependencies]
Loại 3: Không có dependencies

```
useEffect(() => {
  console.log("Component mounted");
  return () => {
    console.log("Component unmounted");
  };
}, []);
```

📘 Ví dụ: Gọi API khi component mount

```
import { useEffect, useState } from "react";

function ProductList() {
  const [products, setProducts] = useState([]);

  useEffect(() => {
    fetch("https://fakestoreapi.com/products")
      .then(res => res.json())
      .then(data => setProducts(data));
  }, []); // chỉ gọi 1 lần khi component mount

  return (
    <ul>
      {products.map(p => (
        <li key={p.id}>{p.title}</li>
      ))}
    </ul>
  );
}
```

📘 Ví dụ: Cập nhật tiêu đề trang

```
function Counter() {
  const [count, setCount] = useState(0);

  useEffect(() => {
    document.title = `Bạn đã nhấn ${count} lần`;
  }, [count]); // cập nhật mỗi khi count thay đổi

  return (
    <button onClick={() => setCount(count + 1)}>Tăng</button>
  );
}
```

✅ 1.3. Lifecycle trong Functional Component

🔄 So sánh với Class Component:

Lifecycle Class	Functional Component (Hooks)
`componentDidMount`	`useEffect(() => {}, [])`
`componentDidUpdate`	`useEffect(() => {}, [deps])`
`componentWillUnmount`	`useEffect(() => { return () => {} }, [])`

📘 Ví dụ có cleanup:

```
useEffect(() => {
  const timer = setInterval(() => {
    console.log("Tick...");
  }, 1000);

  return () => {
    clearInterval(timer); // cleanup khi unmount
  };
}, []);
```

✅ 1.4 Cài đặt thư viện trong React

)

- 1. Dùng npm:

```
npm install axios
```

- 2. Import thư viện vào file:

```
import axios from 'axios';
```

✅ 1.5 Làm việc với danh sách

Duyệt và render danh sách:

```
products.map((item) => (
  <ProductCard key={item.id} {...item} />
))
```

Lọc:

```
const filtered = products.filter(p => p.price > 1000000);
```

💻 2. Thực hành tổng hợp

🔹 2.1 Tạo component ProductCard

```
function ProductCard({ name, price, description }) {
  return (
    <div style={{ border: '1px solid #ccc', padding: 10, marginBottom: 8 }}>
      <h2>{name}</h2>
      <p>Giá: {price} VND</p>
      {description && <small>{description}</small>}
    </div>
  );
}
```

🔹 2.2 Tạo form nhập sản phẩm và render danh sách

```
import { useState } from 'react';
import ProductCard from './ProductCard';

function App() {
  const [form, setForm] = useState({ name: '', price: '', description: '' });
  const [products, setProducts] = useState([]);

  const handleChange = (e) => {
    const { name, value } = e.target;
    setForm(prev => ({ ...prev, [name]: value }));
  };

  const handleSubmit = () => {
    if (!form.name || !form.price) return;
    setProducts(prev => [...prev, { ...form, id: Date.now() }]);
    setForm({ name: '', price: '', description: '' });
  };

  return (
    <div>
      <h1>Quản lý sản phẩm</h1>
      <input name="name" placeholder="Tên sản phẩm" value={form.name} onChange={han
      <input name="price" placeholder="Giá" value={form.price} onChange={handleChar
      <input name="description" placeholder="Mô tả" value={form.description} onChar
      <button onClick={handleSubmit}>Thêm sản phẩm</button>

      <div style={{ marginTop: 20 }}>
        {products.map(p => <ProductCard key={p.id} {...p} />)}
      </div>
    </div>
  );
}
```

🔹 2.3 Gọi API giả (jsonplaceholder)

```
import { useEffect, useState } from 'react';

function ProductFetcher() {
  const [products, setProducts] = useState([]);

  useEffect(() => {
    // Mô phỏng API fetch
    fetch('https://jsonplaceholder.typicode.com/posts?_limit=5')
      .then(res => res.json())
      .then(data => {
        const mapped = data.map(item => ({
          id: item.id,
          name: item.title,
          price: Math.floor(Math.random() * 10000000),
          description: item.body
        }));
        setProducts(mapped);
      });
  }, []);

  return (
    <>
      <h2>Sản phẩm từ API giả</h2>
      {products.map(p => (
        <ProductCard key={p.id} {...p} />
      ))}
    </>
  );
}
```

Nguồn tham khảo:

- Life cycle: https://viblo.asia/p/lifecycle-component-trong-reactjs-gGJ59jzxKX2