TRƯỜNG ĐẠI HỌC THỦY LỢI

**KHOA CÔNG NGHỆ THÔNG TIN**

A blue and white logo

AI-generated content may be incorrect.

**NỀN TẢNG PHÁT TRIỂN WEB**

***Đề tài: Context API và Custom hook***

**Nhóm sinh viên thực hiện: Nhóm 3**

**Thành viên nhóm: Nguyễn Hữu Quân**

**Lục Văn Thận**

**Phương Văn Đức**

**Giảng viên phụ trách môn học: Tạ Chí Hiếu**

Mục lục

[1.Context API 2](#_Toc199112215)

[1.1 Định nghĩa: 2](#_Toc199112216)

[1.2 Cú pháp: 2](#_Toc199112217)

[1.2.1 react.createContext(): 2](#_Toc199112218)

[1.2.2 Context.Provider 2](#_Toc199112219)

[1.2.3 useContext 3](#_Toc199112220)

[1.3 Nên dùng: 3](#_Toc199112221)

[1.3 Nhược điểm: 4](#_Toc199112222)

[2.Custom hook 4](#_Toc199112223)

[2.1 Định nghĩa 4](#_Toc199112224)

[2.2 Nguyên tắc 4](#_Toc199112225)

[2.3 Cách tách logic 4](#_Toc199112226)

Phân công nhiệm vụ

|  |  |
| --- | --- |
| **Thành viên** | **Nhiệm vụ** |
| Lục Văn Thuận | Code demo + Thuyết trình |
| Nguyễn Hữu Quân | Lí thuyết + Thuyết trình |
| Phương Văn Đức | Làm slide + Thuyết trinh |

Nội dung báo cáo

# 1.Context API

## 1.1 Định nghĩa:

- Context API là một tính năng tích hợp sẵn trong React dùng để truyền dữ liệu toàn cục (global) cho các component mà không cần truyền props qua từng cấp trong cây component.

1.2 Cú pháp:

-Context API có 3 cú pháp chính là:

+ createContext

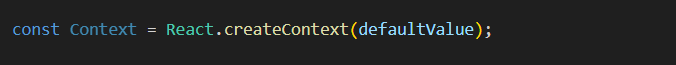
+ Context.Provider

+ useContext

### 1.2.1 react.createContext():

Đây là nơi bạn tạo ra một context mới để chứa dữ liệu cần chia sẻ. Khi sử dụng createContext(), bạn có thể chỉ định một giá trị mặc định cho context (dữ liệu mà các component sẽ sử dụng khi không có Provider).

Cú pháp:

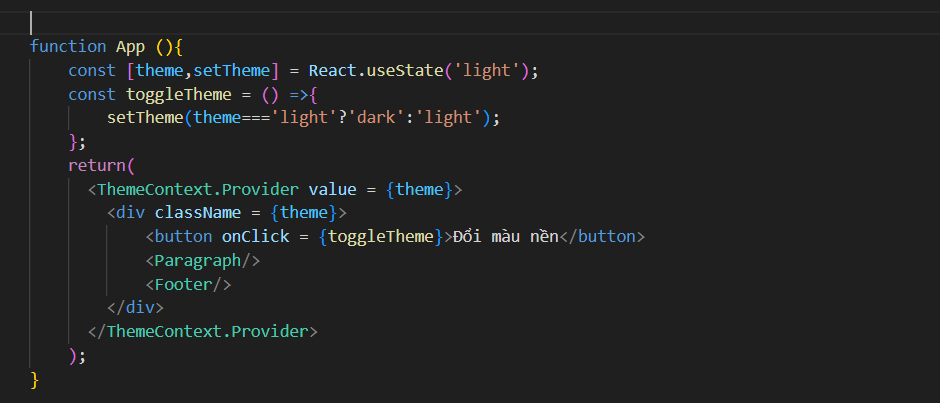


Trong đó:

+ defaultValue: là giá trị mặc định của Context (chỉ dùng khi component không nằm trong <Provider>)

### 1.2.2 Context.Provider

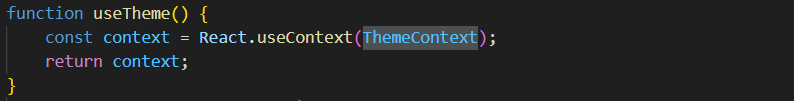
- "Context.Provider là một React component đặc biệt trong Context API, được dùng để truyền dữ liệu (thông qua prop value) từ component cha xuống toàn bộ cây component con nằm bên trong nó, mà không cần truyền qua từng cấp bằng props. Giá trị có thể là bất kỳ loại dữ liệu nào, bao gồm cả state."



### 

### 1.2.3 useContext

- useContext là một React hook cho phép component tiêu thụ dữ liệu được cung cấp bởi Context.Provider mà không cần truyền props qua từng cấp. Hook này nhận vào một Context object và trả về giá trị hiện tại của context đó."



## 1.3 Nên dùng:

* Khi bạn cần chia sẻ dữ liệu toàn cục (global data) cho nhiều component trong ứng dụng, đặc biệt là những dữ liệu mà nhiều component ở các cấp sâu trong cây component đều cần dùng.
* Khi bạn không muốn phải truyền props qua rất nhiều cấp component (gọi là "prop drilling") — tức là thay vì phải gửi props từ component cha đến con, rồi con lại truyền cho con nữa, bạn chỉ cần cung cấp dữ liệu ở một chỗ duy nhất bằng Provider, và mọi component cần sẽ lấy dữ liệu đó trực tiếp.
* Phù hợp với các dự án nhỏ hoặc những phần tính năng đơn giản. Nếu ứng dụng của bạn chưa quá phức tạp hoặc chưa cần quản lý state quá nhiều nơi, Context API là giải pháp nhẹ, dễ hiểu và nhanh triển khai.

## 1.3 Nhược điểm:

- Re-render rộng (tái render toàn bộ):

Khi giá trị trong Context.Provider thay đổi, tất cả các component sử dụng context đó sẽ được render lại, dù có thể chỉ một phần nhỏ dữ liệu thay đổi thôi. Điều này có thể gây tốn hiệu năng nếu cây component lớn hoặc giá trị context thay đổi thường xuyên.

- Context API không có nhiều tính năng nâng cao như:

+ Quản lý các action, reducer phức tạp (như Redux)

+ Lưu trữ trạng thái bên ngoài

+ Middleware, devtools hỗ trợ debugging

Vì vậy, đối với các ứng dụng lớn, phức tạp, bạn nên cân nhắc sử dụng các thư viện chuyên dụng.

- **Khó bảo trì nếu lạm dụng:**

**Nếu bạn dùng quá nhiều Context hoặc truyền nhiều loại dữ liệu khác nhau trong một context, code có thể trở nên rối, khó theo dõi, và khó debug.**

# 

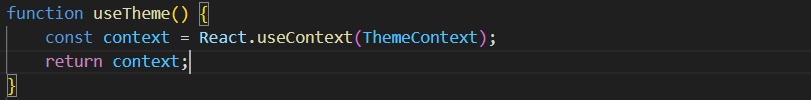
# 2.Custom hook

## 2.1 Định nghĩa

- Custom Hook là một hàm JavaScript do người dùng tự tạo trong React, dùng để tách biệt và tái sử dụng logic có trạng thái hoặc side effect giữa nhiều component.

Custom Hook thường được đặt tên bắt đầu bằng tiền tố use (ví dụ: useFetch, useWindowSize) và có thể sử dụng các hook chuẩn của React bên trong như useState, useEffect, useContext...

Việc sử dụng Custom Hook giúp giữ cho component gọn gàng, dễ đọc và dễ bảo trì, đồng thời tránh lặp lại code (giữ nguyên tắc DRY).



## 2.2 Nguyên tắc tạo custom hook

* Tên hook phải bắt đầu bằng use
* Hook là hàm JavaScript bình thường
* Hook có thể sử dụng các hook khác bên trong
* Tránh side effects ngoài phạm vi hook
* Trả về đúng dữ liệu cần thiết
* Giữ hook nhỏ, chuyên biệt

## 2.3 Cách tách logic

### 2.3.1 Mục đích:

* Việc tách logic giúp các component trở nên gọn nhẹ, tập trung vào giao diện (UI).
* Đưa phần logic xử lý trạng thái, side effects ra ngoài component.
* Tái sử dụng logic chung cho nhiều component khác nhau.
* Dễ dàng bảo trì và kiểm thử.

### 2.3.2 Các bước tách:

1. Xác định phần logic cần tách
2. Tạo hàm mới bắt đầu bằng use
3. Di chuyển logic liên quan vào hàm này
4. Trả về dữ liệu cần thiết
5. Sử dụng Custom Hook trong component

# 3.Tổng kết

* Context API giúp chia sẻ dữ liệu toàn cục trong ứng dụng React một cách tiện lợi, nhưng cần sử dụng cẩn thận để tránh ảnh hưởng đến hiệu năng.
* Custom Hook là công cụ mạnh mẽ giúp tái sử dụng logic, giữ component rõ ràng, dễ bảo trì.
* Kết hợp cả hai công cụ này đúng cách sẽ giúp phát triển ứng dụng React hiệu quả, sạch sẽ và dễ mở rộng.
* **Link Github nhóm :** [**https://github.com/quan-IT/2025\_CSE391\_NentangWeb**](https://github.com/quan-IT/2025_CSE391_NentangWeb)