* **useEffect()**
  + Side effect là những event có thể làm thay đổi DOM trong components.

VD:

* + - Get data
    - Subscription
  + Có thể truy cập được props và state
  + Run useEffect() sau khi render (DOM thay đổi)
    - Run sau componentDidMount(), componentDidUpdate()
    - Run trước componentWillUnmount()

Chỉ có 1 function duy nhất 🡺 có thể call api ở đây

* **Lifecycle**

|  |  |
| --- | --- |
|  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| *Open*  *componentDidMount()* | *Click re-render*  *componentWillUnmount() 🡪 componentDidMount()* | *Click hide component*  *componentWillUnmount()* | *Click show component*  *componentDidMount()* |

*Đọc thêm:* [*https://daveceddia.com/useeffect-hook-examples/*](https://daveceddia.com/useeffect-hook-examples/)

* Có 2 loại là
  + **không cần clean up**
    - Run một vài đoạn code sau khi React đã cập nhập DOM

VD: Sau khi call api sẽ tự ý thay đổi DOM, và logging

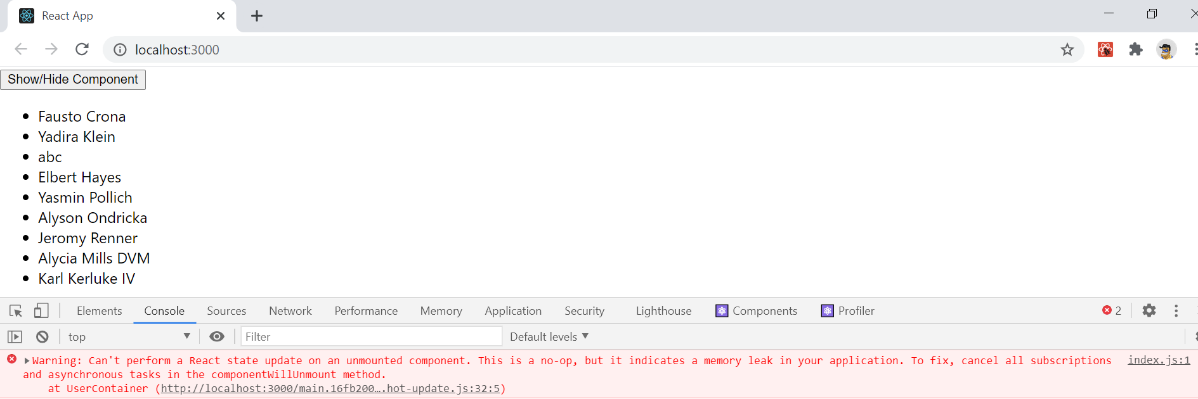
VD: update document title sau khi React thay đổi DOM

|  |  |
| --- | --- |
| *Effect hook* | *Class component*  *Bị lặp lại trong 2 lifecycle method* |

* + **Clean up**
    - Remove event (tránh xảy ra memory leak)

VD1: cancel request API

|  |  |
| --- | --- |
|  |  |



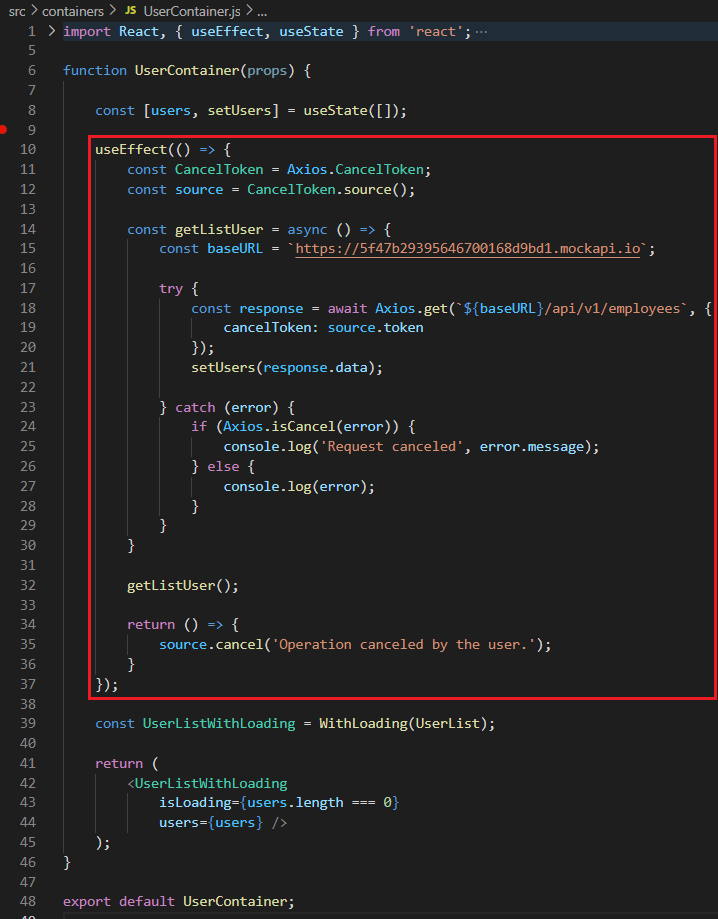
*Nếu ẩn hiện component quá nhanh thì sẽ gây ra memory leak*

* *Cách fix: cancel request khi unmounting*

[*https://github.com/axios/axios#cancellation*](https://github.com/axios/axios#cancellation)

|  |  |
| --- | --- |
|  |  |

*Fix*

**

*useEffect()*

VD2: subscribe vào tình trạng online của 1 danh sách friend

|  |  |
| --- | --- |
| *Effect hook*  *Cho phép tạo và xóa subscription trong cùng một effect.* | *Class component*  *Subscription và unsubscription ở trong 2 method khác nhau* |

VD3: call API



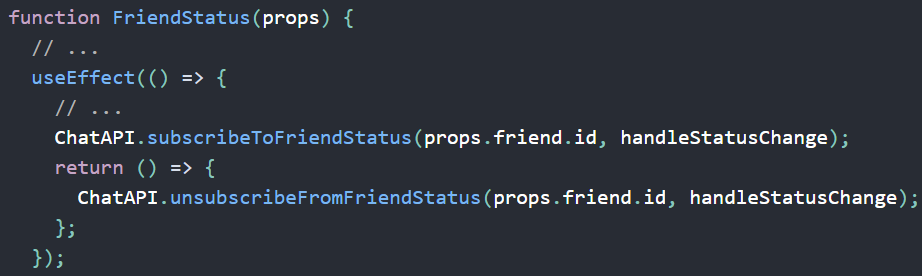
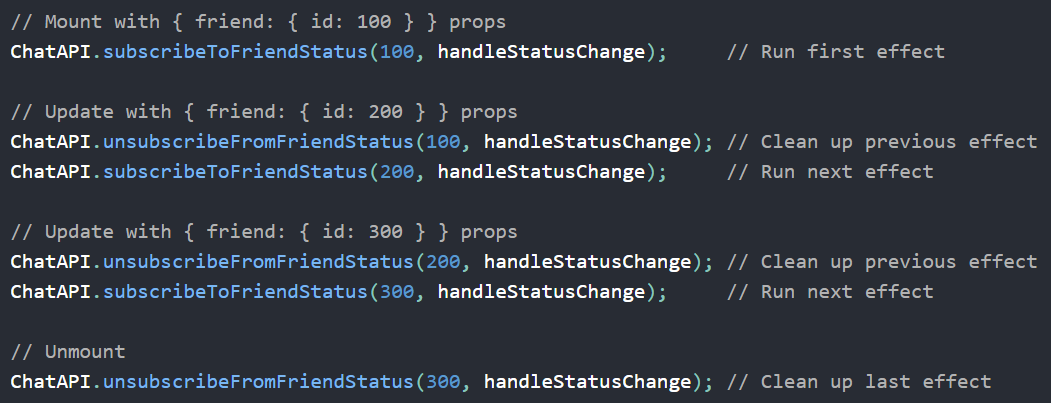
* **Note**
  + **Effect có thể cleanup hoặc không**

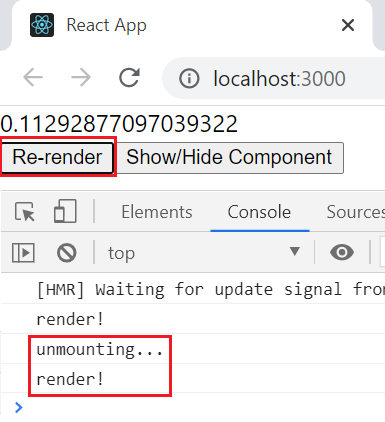
Nếu cleanup thì sẽ return ra 1 function

|  |  |
| --- | --- |
| *Clean up* | *Không cần cleanup* |

* + **Thực hiện cleanup effect mỗi khi re-render**

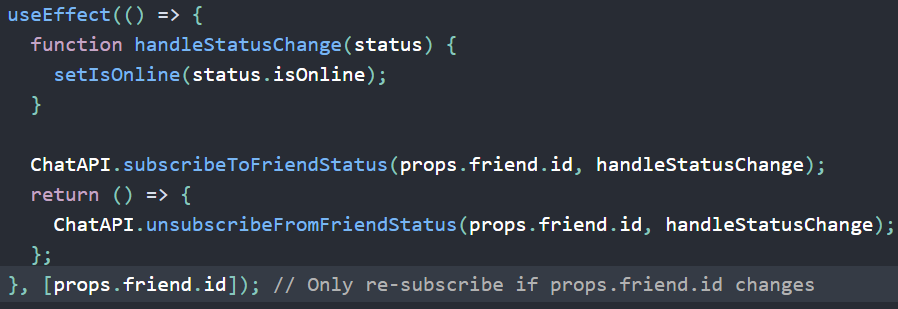
React đồng thời cleans up effect từ những lần render trước (unscription) và subscription cho lần render tiếp theo

 🡺 

**

* + **Check điều kiện khi re-render**

|  |  |
| --- | --- |
| *Effect hook* | *Class component* |



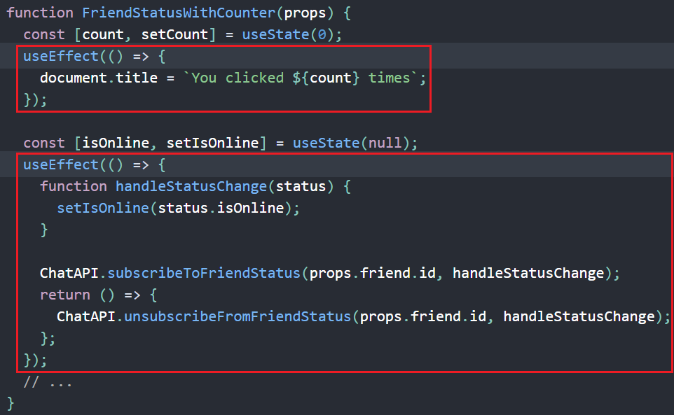
Array truyền thêm để so sánh 2 array cũ và mới, nếu có value khác thì mới thực hiện useEffecet())

Nếu truyền vào array rỗng thì React chỉ chạy 1 effect và clean nó 1 lần duy nhất (lúc mounting và unmounting) nghĩa là effect không phụ thuộc vào giá trị nào cả (thường sử dụng để tránh re-render quá nhiều)



* + **Sử dụng nhiều effect**

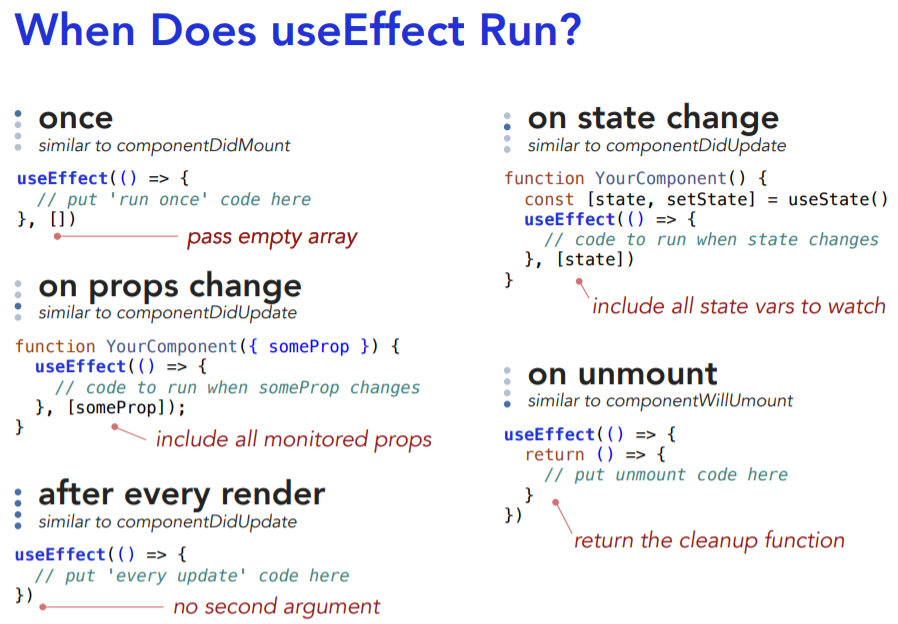
Apply theo thứ tự khai báo effect

****

*Tách code dựa trên những cái đang làm (chứ không phải theo lifecycle method)*

* **Summary**

****

****

*Đọc thêm:* [*https://daveceddia.com/useeffect-hook-examples/*](https://daveceddia.com/useeffect-hook-examples/)

* **Reference**

<https://reactjs.org/docs/hooks-effect.html>

<https://daveceddia.com/useeffect-hook-examples/>