**React useRef**

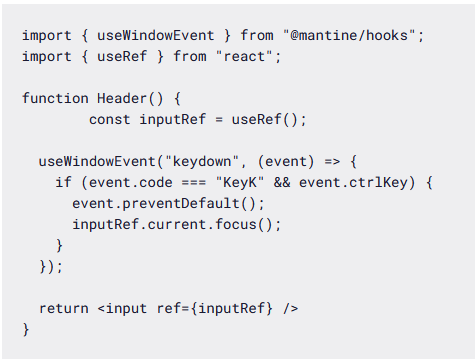
useRef allows us to get direct access to a JSX element.

To use useRef, call it, get the returned value, and put it on the ref prop for a given React element.

Refs do not have a built-in prop on components, only React elements.

Once a ref is attached to a given element, we can use the value stored on ref.current to access the element itself.

For example, if we wanted to write some code that focuses a search input when the users use the key combination Control + K.



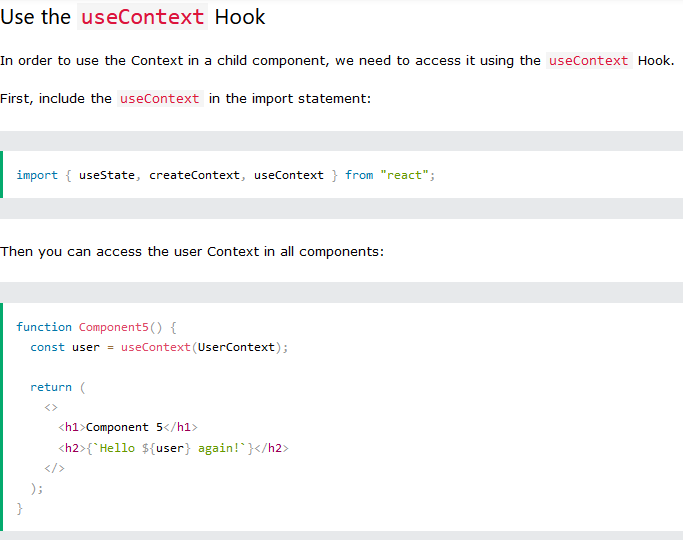
**useContext là gì?**

Chia sẻ state giữa các component là bài toán phổ biến trong React App. Giải pháp tốt nhất là bạn phải lifting state (đẩy state từ component con lên component cha). Nhưng nó yêu cầu bạn phải truyền props xuống cho các component con. Điều đó không phải là vấn đề quá lớn, nhưng trong một số trường hợp truyền props xuống quá nhiều cấp component lại rất khó khăn và tốn nhiều thời gian.

Để tránh được vấn đề này, bạn có thể tạo một context để thêm những shared state và sau đó component nào cần thì bạn gọi context đó ra xài, không cần phải truyền từ cha xuống con nữa. Nó cũng giống như biến global được chia sẻ và sử dụng ở nhiều nơi, nhưng context được quản lí tốt hơn để maintain code dễ hơn.







**useCallback**

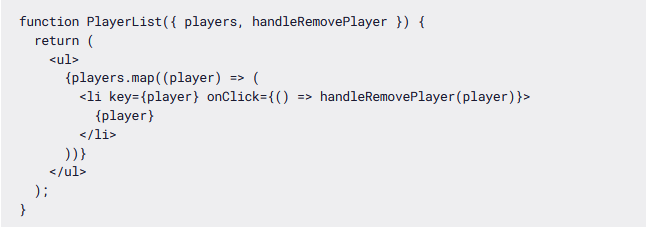
useCallback is a hook that we use to help with our app's performance.

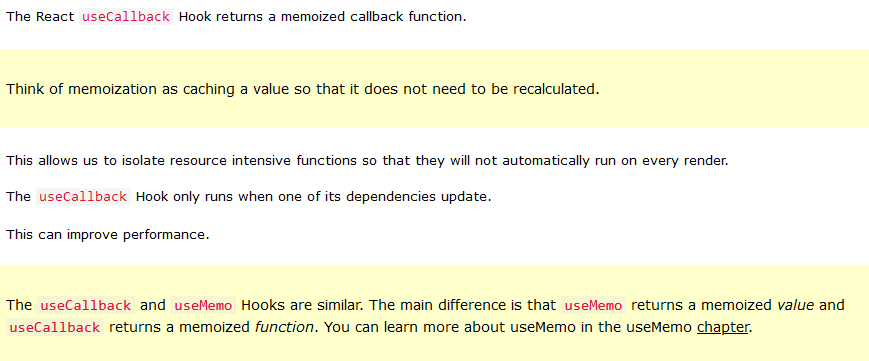
Specifically, it prevents functions from being recreated every time our component re-renders, which can hurt the performance of our app.

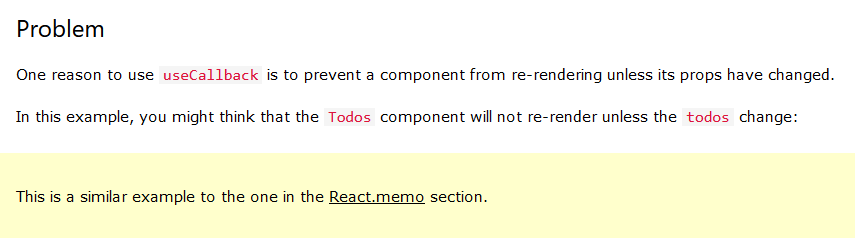
If we go back to our PlayerList example from earlier and add the ability to add players to our array, when we pass down a function to remove them (handleRemovePlayer) via props, the function will be recreated every time.

The way to fix this is to wrap our callback function in useCallback and to include its one argument player in the dependencies array:











Try running this and click the count increment button.

You will notice that the Todos component re-renders even when the todos do not change.

Why does this not work? We are using memo, so the Todos component should not re-render since neither the todos state nor the addTodo function are changing when the count is incremented.

This is because of something called "referential equality".

Every time a component re-renders, its functions get recreated. Because of this, the addTodo function has actually changed.



:

To fix this, we can use the useCallback hook to prevent the function from being recreated unless necessary.

Use the useCallback Hook to prevent the Todos component from re-rendering needlessly

