

FIAP

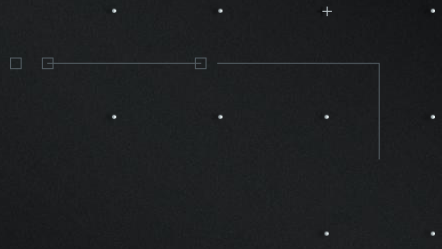
FIAP

SLIDER ▢■◀





React - Hooks



useCallback

O `useCallback` é um Hook do React usado para memorizar funções.

Ele retorna uma versão memorizada da função que só muda quando suas dependências mudam.

Isso é especialmente útil para otimizar performance, principalmente quando a função é passada para componentes filhos que podem ser renderizados desnecessariamente.

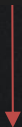
useCallback

Você o usa quando:

- Passa funções para componentes filhos
- Quer evitar que funções sejam recriadas em toda renderização
- Está lidando com listas, eventos ou efeitos que dependem de funções estáveis

useCallback

```
const Component = ({ prop1, prop2 }: any) => {  
  
  const myFunction = () => {  
    // Do something with useCallback  
  
    // Dependency array should include all props and state used inside the function  
    console.log("Function called with props:", prop1, prop2);  
  };  
};
```



```
const Component = ({ prop1, prop2 }) => {  
  
  const myFunction = useCallback(( ) => {  
    // Do something with useCallback  
  
    // Dependency array should include all props and state used inside the function  
    console.log("Function called with props:", prop1, prop2);  
  }, [prop1, prop2]);  
};
```


useMemo

O `useMemo` é um Hook que memoriza valores calculados.

Ou seja, ele só recalcula o valor quando as dependências mudam, evitando fazer cálculos desnecessários a cada renderização.

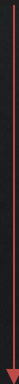
useMemo

Você o usa quando:

- O cálculo é pesado (ex: filtragem, ordenação, operações matemáticas)
- O valor derivado não muda com frequência
- Quer evitar que um componente re-renderize à toa

useMemo

```
const Component = ({ prop1, prop2 }: any) => {  
  const sumValue = prop1 + prop2; // Sum values
```



```
const Component = ({ prop1, prop2 }: any) => {  
  const sumValue = useMemo(() => prop1 + prop2, [prop1, prop2]);
```


<https://github.com/vinimarcili/react-basic/tree/master/performance>

Dúvidas, críticas ou sugestões?

FIAP