

React's 6 Main Hooks Explained

 @nikki.and.chris

 @NikkiSiapno

 @ChrisStaud

Next →

useState

by nikkiandchris.io

Create and update state values.

holds the
state value

used to update
the state value



```
const [count, setCount] = useState(0)  
setCount(1)
```

The diagram shows a code block with two lines of JavaScript code. The first line is `const [count, setCount] = useState(0)` and the second line is `setCount(1)`. Four arrows point from text annotations to parts of the code: one from 'holds the state value' to `count`, one from 'used to update the state value' to `setCount`, one from 'creates the state resources and sets the initial value to 0' to `useState(0)`, and one from 'changes the count value to 1' to `setCount(1)`. The code block has a white background and a subtle shadow, with three colored dots (red, yellow, green) in the top left corner.

changes the
count value to 1

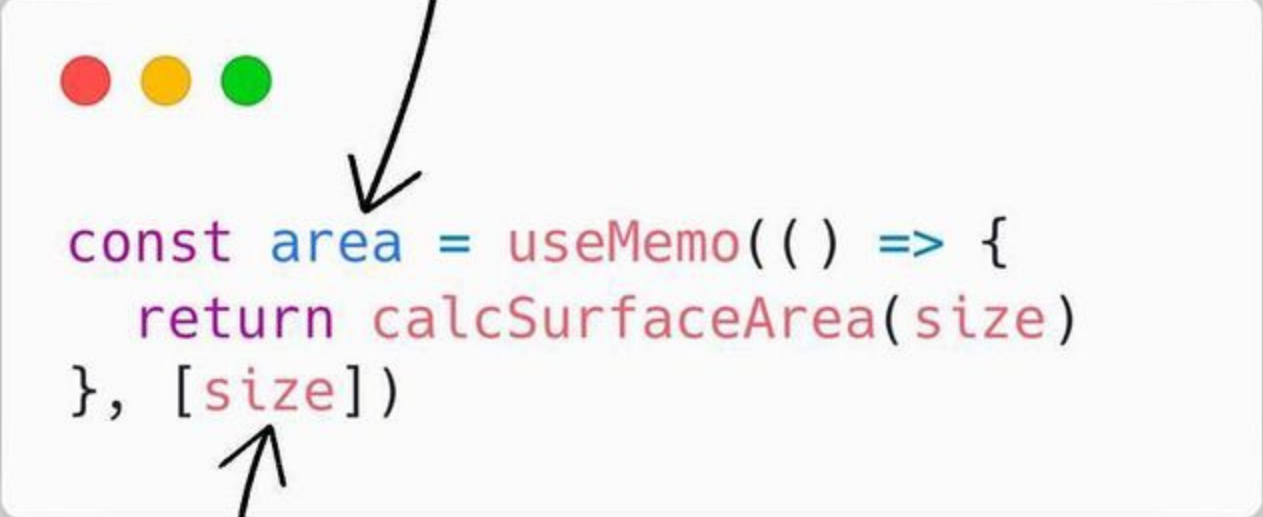
creates the
state resources
and sets the
initial value to 0

useMemo

by nikkiandchris.io

Returns a memoized value which only gets recalculated when the defined dependencies change.

holds the cached value returned by calcSurfaceArea



```
const area = useMemo(() => {  
  return calcSurfaceArea(size)  
}, [size])
```

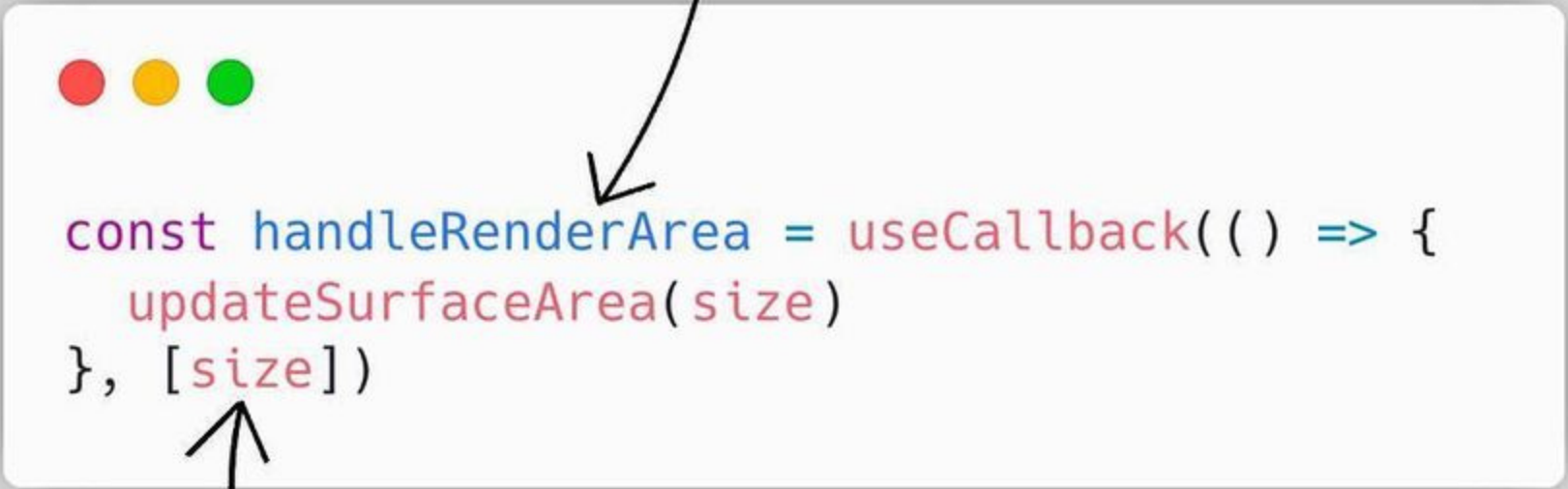
area updates every time size changes

useCallback

by nikkiandchris.io

Returns a memoized version of a callback that only changes when the dependencies change.

a memoized version of
updateSurfaceArea



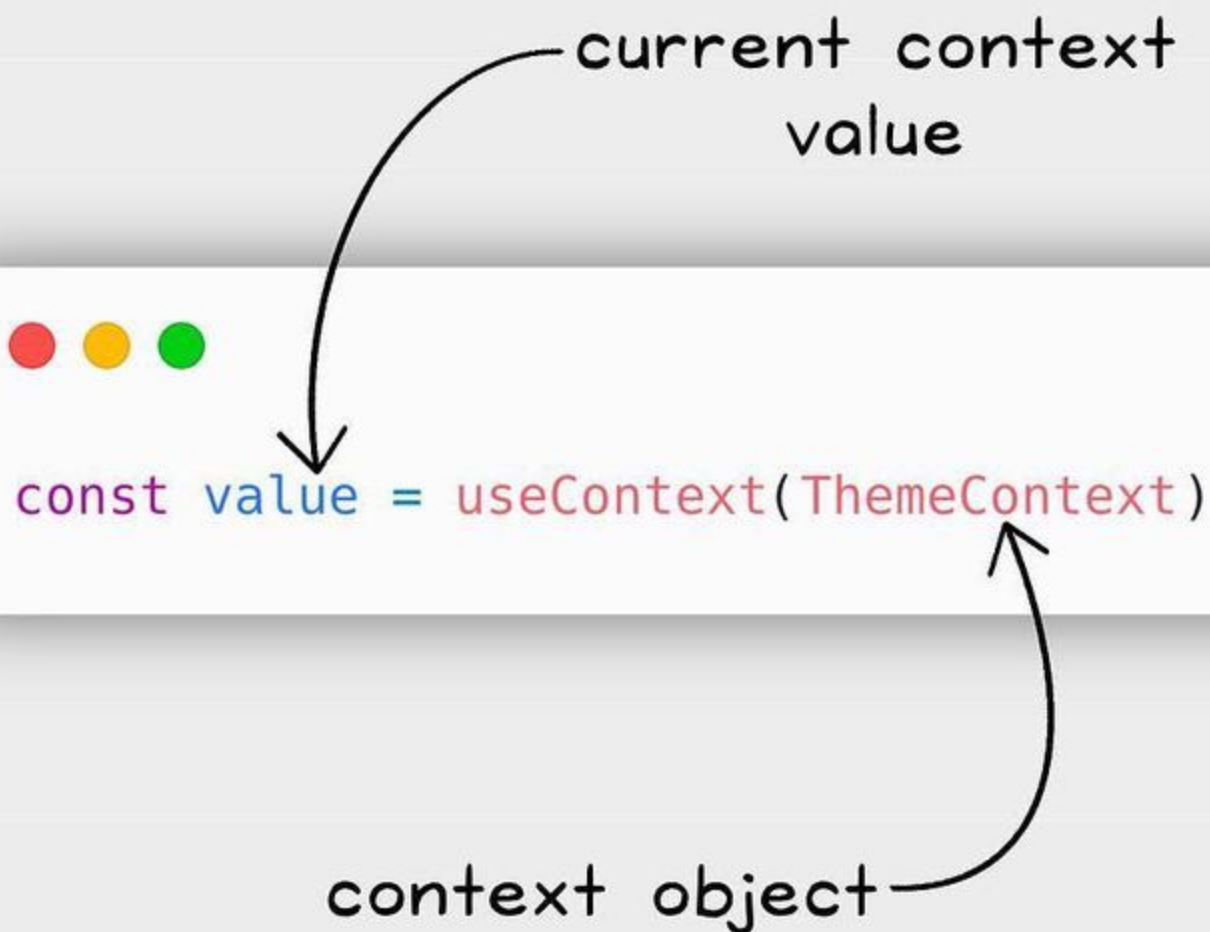
```
const handleRenderArea = useCallback(() => {  
  updateSurfaceArea(size)  
}, [size])
```

handleRenderArea
updates when size
changes value

useContext

by nikkiandchris.io

Accepts a context object that's created using `React.createContext`, and returns the current value of that context.



useEffect

by nikkiandchris.io

Used to run side effects in the component such as fetching data or adding listeners.

runs after the initial render

runs just before the component unmounts

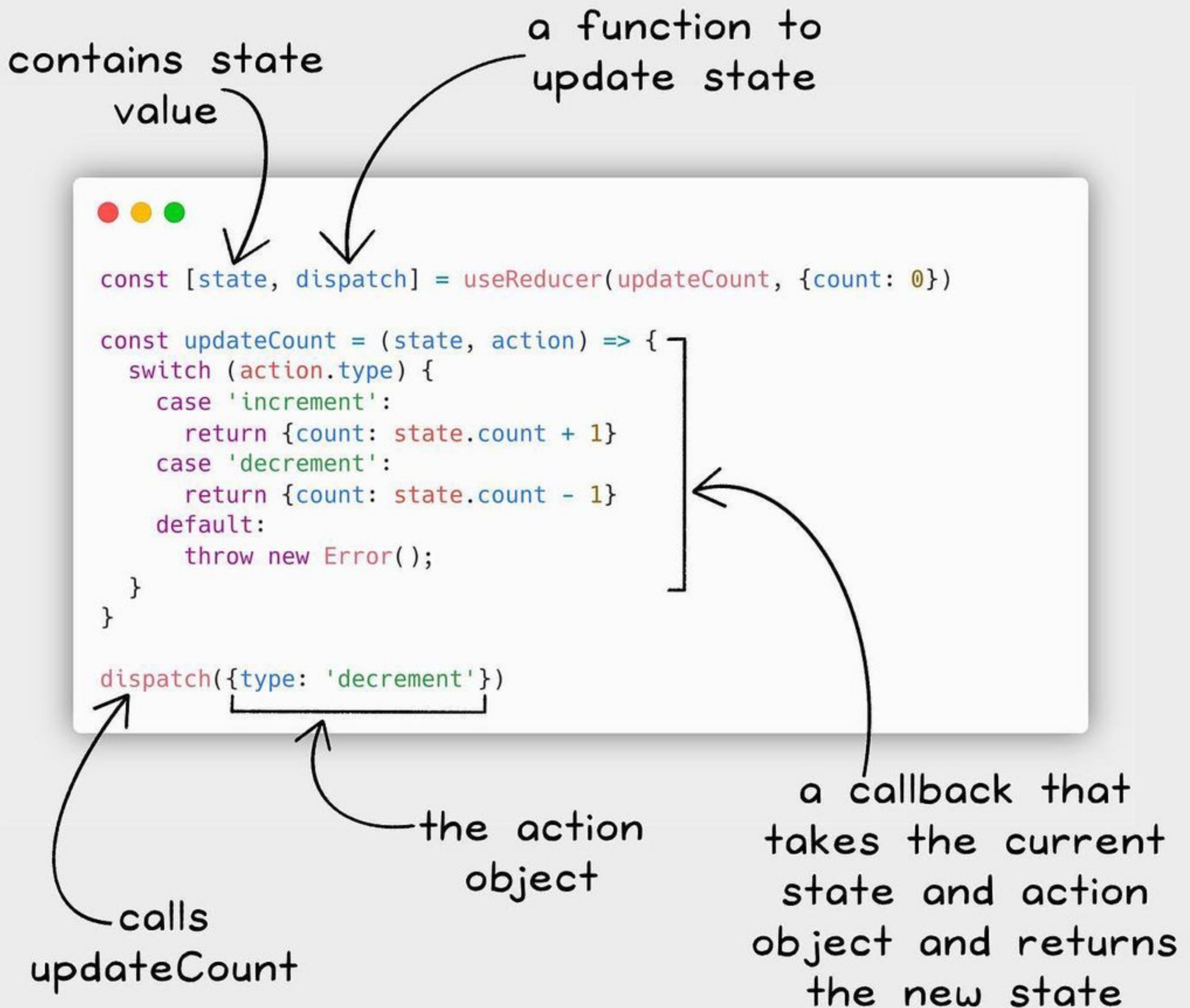
```
useEffect(() => {  
  → addListeners()  
  return () => {  
    removeListeners()  
  }  
})  
  
useEffect(() => {  
  → fetchUserInfo(userID)  
}, [userID])
```

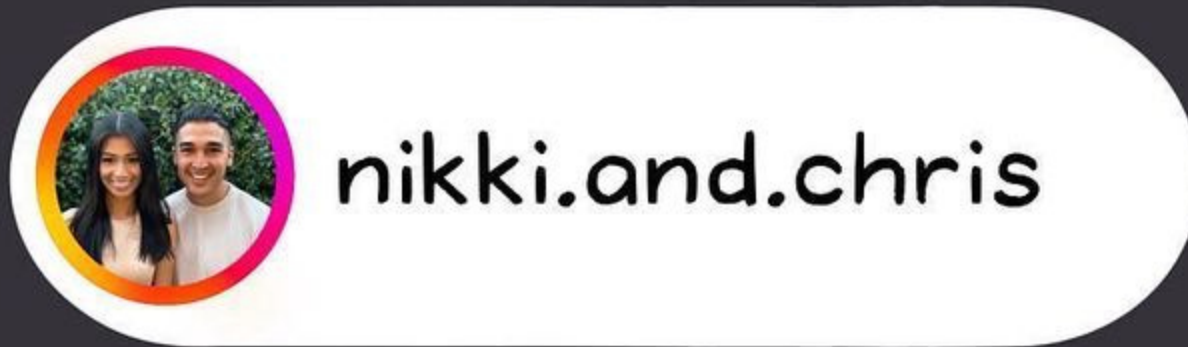
runs after the first render and every time userID updates

useReducer

by nikkiandchris.io

Similar to useState but also let's you use your own update state logic.





nikki.and.chris



code_tune

Was this useful?

Let us know in the
comments and follow us
for more!

