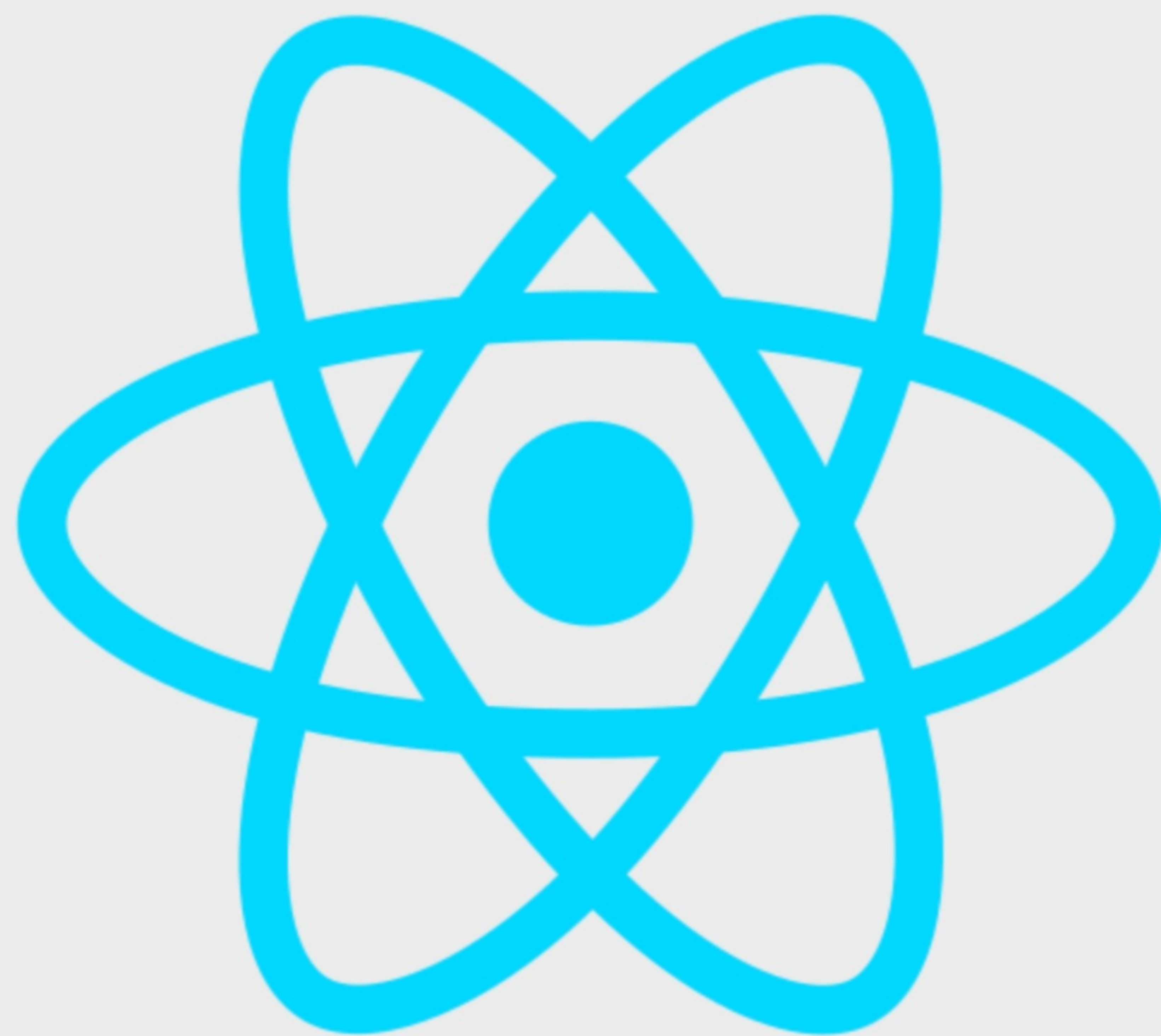


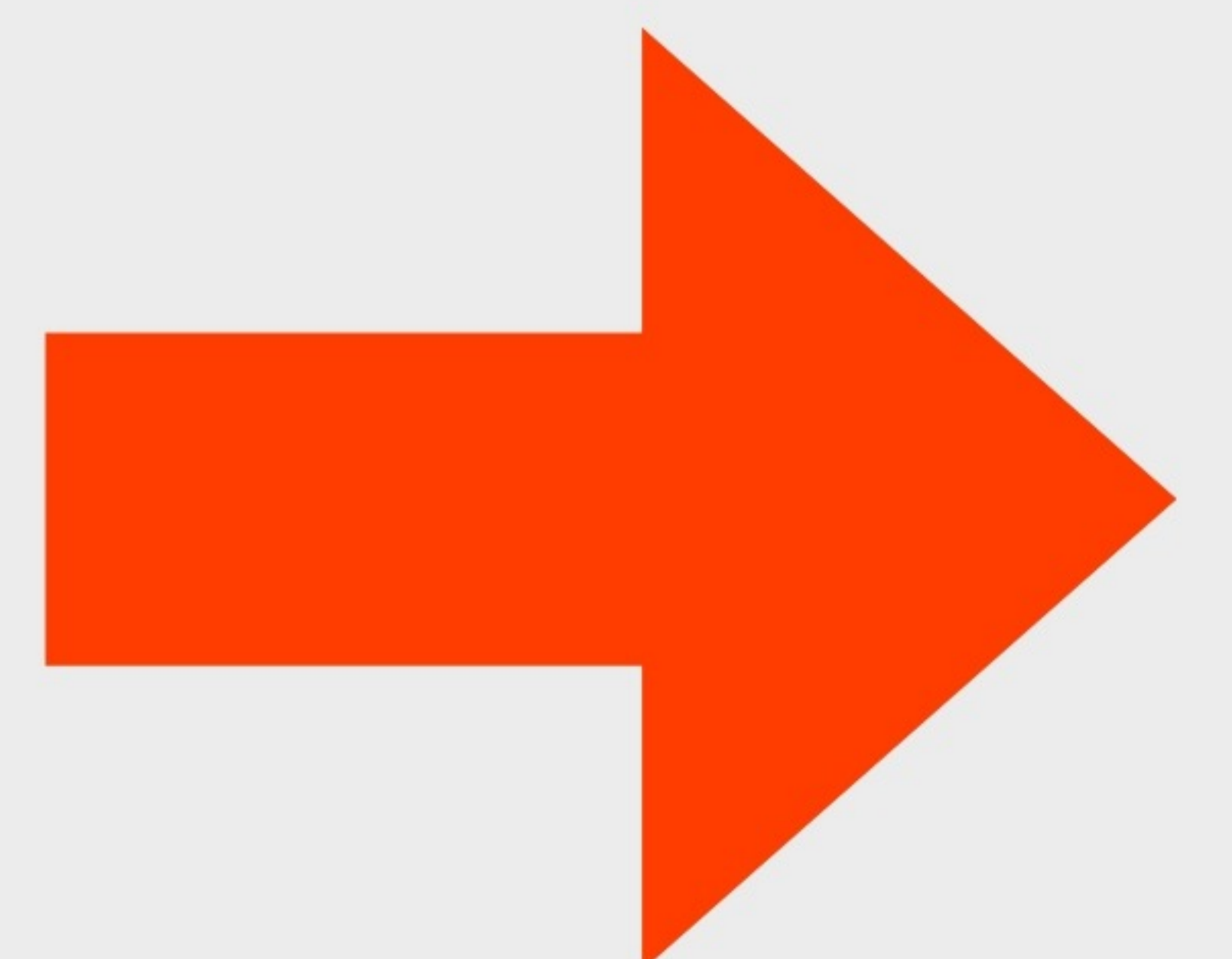
# Power of React

## 6 Hooks

## Simplified



**K RAKESH**  
*@webDeveloper*



# useState

Create and update state values.

holds the  
state value

used to update  
the state value



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

The diagram illustrates the useState hook usage. A code block contains two lines: `const [count, setCount] = useState(0)` and `setCount(1)`. Four arrows point from descriptive text to specific 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)`.

changes the  
count value to 1

creates the  
state resources  
and sets the  
initial value to 0

# useEffect

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

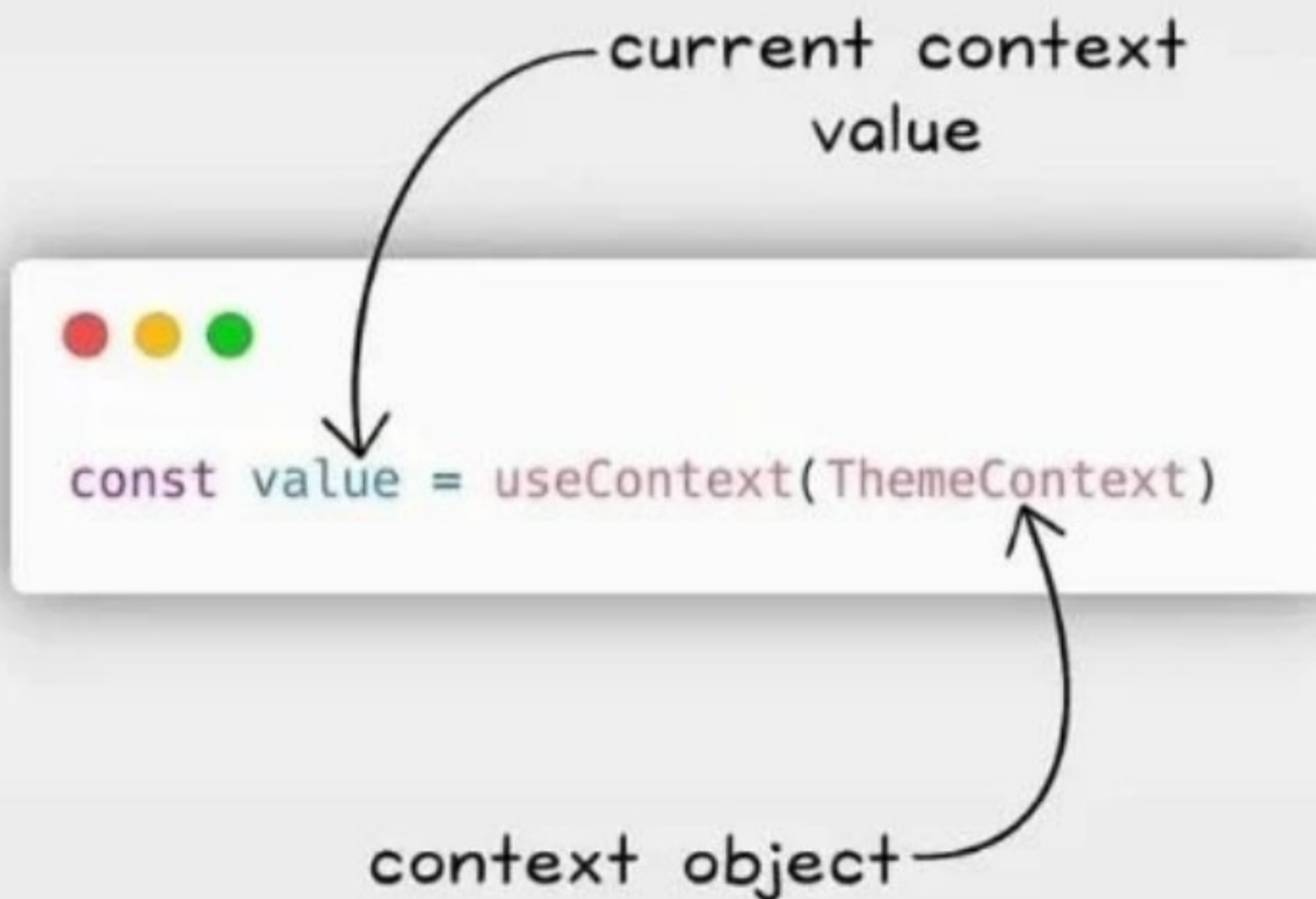


```
useEffect(() => {  
  addListeners()  
  return () => {  
    removeListeners()  
  }  
})  
  
useEffect(() => {  
  fetchUserInfo(userID)  
}, [userID])
```

runs after the first render  
and every time userID updates

# useContext


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



# useMemo

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

holds the cached value returned by  
calcSurfaceArea



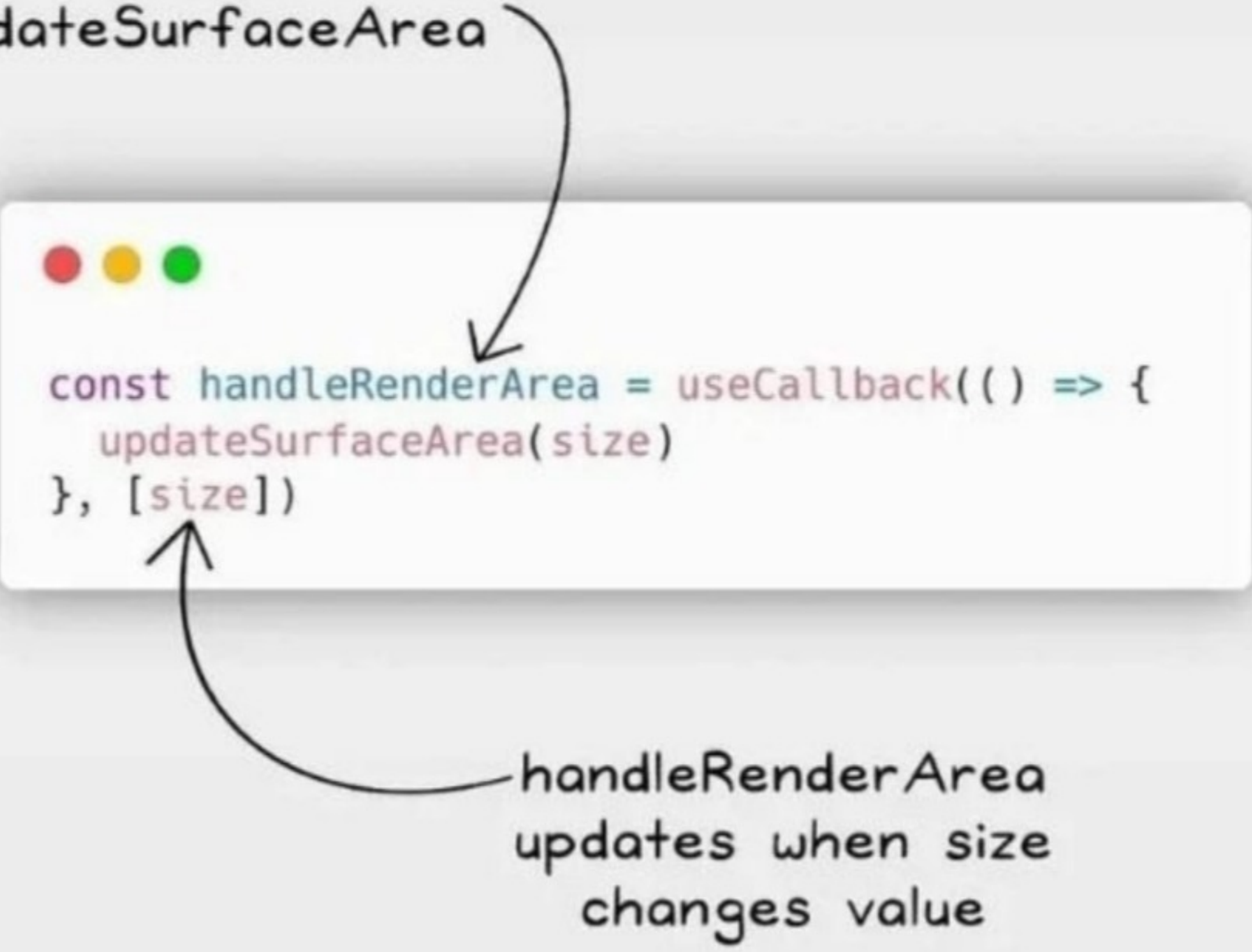
```
const area = useMemo(() => {  
  return calcSurfaceArea(size)  
}, [size])
```

area updates every  
time size changes

# useCallback

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

a memoized version of  
updateSurfaceArea

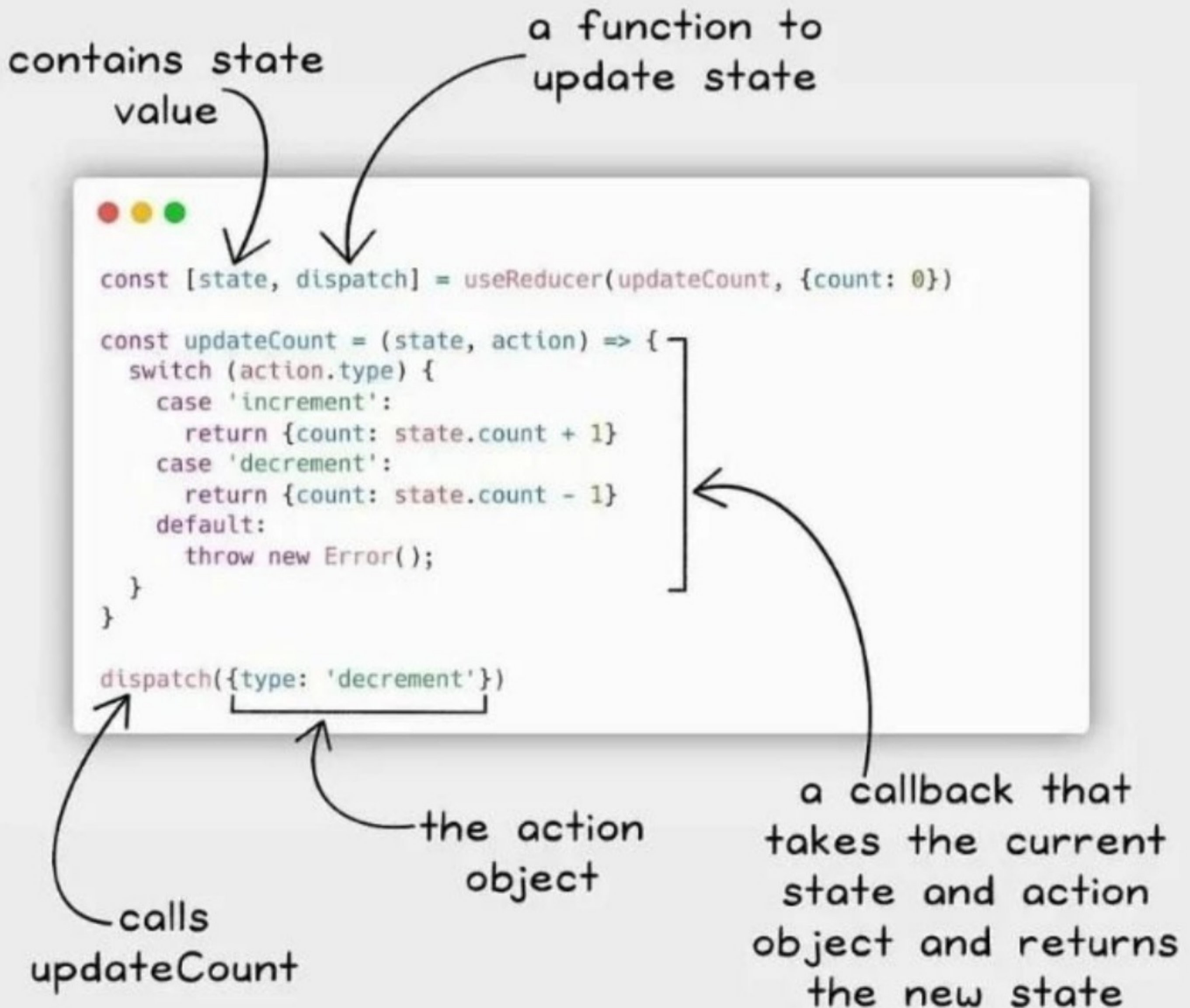


```
const handleRenderArea = useCallback(() => {  
  updateSurfaceArea(size)  
}, [size])
```

handleRenderArea  
updates when size  
changes value

# useReducer

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



# Did you find it useful ?

## Leave a **comment** !



**K. RAKESH**

**@Web Developer**

**FALLOW FOR MORE**

**Like**



**Comment**



**Repost**

