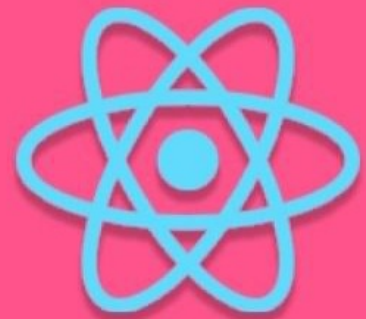




React 18 Broke useEffect



SWIPE



If you are a React developer you
must have already know that a
new version of React has been
launched with a new feature called
“concurrent rendering” and some
new hooks

But new updated broke a lot of
code related to useEffect

Before React 18 if you want to run something once a component mounts you'd do something like this

A code editor window with a dark theme. The title bar shows three colored circles (red, yellow, green) and a tab labeled 'JS App.js'. The code inside is a useEffect hook that logs a message to the console.

```
useEffect(() => {  
  console.log('This runs on mount');  
}, []);
```

But In React 18 because of
concurrent rendering, this will run
twice!



JS

App.js

```
useEffect(() => {  
  console.log('This run twice on mount');  
}, []);
```

This can be temporarily fix this by removing `<StrictMode>` from `index.js`


Because In React 18 `<StrictMode>` will force component to render twice on initial mount (only during development)

This is to make developers fix the errors in their code gradually after upgrading to React 18

Even after you've removed
`<StrictMode>` `useEffect` can run
twice if you are using any new hook
that uses concurrent feature of
React 18

So the complete workaround of that
is →

Use `useRef()` to check if the component has been mounted or not

A code editor window with a dark theme. The title bar shows three colored circles (red, yellow, green) and a tab labeled 'JS App.js'. The code is written in a light blue font on a dark background.

```
function App() {  
  const isMounted = useRef();  
  
  useEffect(() => {  
    if (isMounted.current) return;  
  
    console.log('This will run once');  
  
    isMounted.current = true;  
  }, [])  
}
```

This re-render is happening because of React's new concurrent rendering works, it makes rendering of UI faster and more responsive by pausing it when user is trying to interact with UI and then continue later

It can prepare new screens in the background without blocking the main thread So that the UI can respond immediately to user input even if it's in the middle of a large rendering task, creating a fluid user experience.