useEffect in React: From Headache to Mastery  
What I wish someone told me earlier…  
  
If you’ve ever written a React component and thought:  
  
“Why is this running twice?”  
“Why is this API call firing in a loop?”  
“Why isn’t my cleanup working!?”  
  
You’re not alone.  
  
Whether you’re just learning React or have built dozens of UIs, useEffect is one of those hooks that starts simple but turns tricky fast.  
  
After working with React for years, here’s a breakdown I often share with junior devs and honestly, senior folks appreciate the reminders too.  
  
🚀 What is useEffect actually doing?  
  
In one line:  
👉 It lets you run side effects in your components like fetching data, setting up listeners, timers, and syncing to external systems.  
  
✅ The 3 Common Patterns:  
  
🟢 Run once on mount:  
  
useEffect(() => {  
// Runs once when the component mounts  
}, []);  
  
🟡 Run when something changes  
  
useEffect(() => {  
// Runs when `count` changes  
}, [count]);  
  
🔴 Run on unmount (cleanup):  
  
useEffect(() => {  
const id = setInterval(...);  
return () => clearInterval(id);  
}, []);  
  
💣 Common Pitfalls (That I’ve Definitely Hit)  
  
• Missing the dependency array = infinite loops  
• Adding everything blindly into the array = unnecessary re-renders  
• Forgetting cleanup = memory leaks  
• Functions inside useEffect that keep changing = unexpected triggers  
  
💡 Tips that helped me (and now my team):  
  
1. Only add what’s necessary to the dependency array. If you’re watching [**user.name**](http://user.name/), don’t add the whole user object.  
  
2. Memoize your handlers with useCallback if you pass them to useEffect.  
  
3. Always clean up listeners, intervals, subscriptions, etc.  
  
4. Debug useEffect like a pro: Add console.log('Effect ran', varName) to see what’s changing and when.  
  
[**hashtag#ReactJS**](https://www.linkedin.com/search/results/all/?keywords=%23reactjs&origin=HASH_TAG_FROM_FEED) [**hashtag#JavaScript**](https://www.linkedin.com/search/results/all/?keywords=%23javascript&origin=HASH_TAG_FROM_FEED) [**hashtag#WebDevelopment**](https://www.linkedin.com/search/results/all/?keywords=%23webdevelopment&origin=HASH_TAG_FROM_FEED) [**hashtag#Frontend**](https://www.linkedin.com/search/results/all/?keywords=%23frontend&origin=HASH_TAG_FROM_FEED) [**hashtag#ReactHooks**](https://www.linkedin.com/search/results/all/?keywords=%23reacthooks&origin=HASH_TAG_FROM_FEED) [**hashtag#useEffect**](https://www.linkedin.com/search/results/all/?keywords=%23useeffect&origin=HASH_TAG_FROM_FEED) [**hashtag#CleanCode**](https://www.linkedin.com/search/results/all/?keywords=%23cleancode&origin=HASH_TAG_FROM_FEED) [**hashtag#TechTips**](https://www.linkedin.com/search/results/all/?keywords=%23techtips&origin=HASH_TAG_FROM_FEED) [**hashtag#DevCommunity**](https://www.linkedin.com/search/results/all/?keywords=%23devcommunity&origin=HASH_TAG_FROM_FEED) [**hashtag#LinkedInTech**](https://www.linkedin.com/search/results/all/?keywords=%23linkedintech&origin=HASH_TAG_FROM_FEED)