🎯 Why do we need `useEffectEvent`?

Sometimes inside `useEffect`, we use functions that depend on the latest state — but **React stores the old version of that function** → leading to **stale state** issues ❌

`useEffectEvent` fixes this by: ✓ Always giving the **latest updated function** inside `useEffect` ✓ Preventing unnecessary re-renders ✓ Making `useEffect` cleaner

---

🧠 Simple Explanation for Students

Concept	Purpose
`useEffect`	Run a side effect when something changes
`useEffectEvent`	Allow functions inside useEffect to always use *latest state* without becoming dependencies

> You can call it **"side-effect safe function"** 💡

😵 The Problem (Without useEffectEvent)

Example: Logging latest count inside timer but React logs old value 👇

```
import { useState, useEffect } from "react";

export default function App() {
  const [count, setCount] = useState(0);

  function handleLog() {
    console.log("Count inside effect:", count);
  }

  useEffect(() => {
    const interval = setInterval(() => {
      handleLog(); // ❌ Logs stale old values
    }, 1000);

    return () => clearInterval(interval);
  }, []); // handleLog not in dependency

  return (
    <button onClick={() => setCount(count + 1)}>
      Count: {count}
    </button>
  );
}
```

Even if UI updates → timer keeps logging old count 😬

✓ Solution: `useEffectEvent`

👉 We wrap the function using **useEffectEvent** 👉 Now effect sees the **latest state** always!

```
import { useState, useEffect, useEffectEvent } from "react";

export default function App() {
  const [count, setCount] = useState(0);

  const handleLog = useEffectEvent(() => {
    console.log("Latest Count:", count); // 🎯 Always updated!
  });

  useEffect(() => {
    const interval = setInterval(() => {
      handleLog();
    }, 1000);

    return () => clearInterval(interval);
  }, []); // Still empty dependency!

  return (
    <button onClick={() => setCount(count + 1)}>
      Count: {count}
    </button>
  );
}
```

💡 When to Use useEffectEvent?

Situation	Should use useEffectEvent?
Timer or interval that uses state	✓ Yes
Event listeners	✓ Yes
Cleanup functions that need latest values	✓ Yes
Simple effects based on dependencies	❌ No (normal useEffect enough)

🧠 Quick Summary for Students

Hook	Purpose
`useEffect`	Run side effects based on dependencies
`useEffectEvent`	Use updated logic **inside** `useEffect` without adding dependencies

> Think: ✨ “Is this function only needed inside useEffect?” → then useEffectEvent

---

🧩 Mini Task for Your Class

Ask them to build: ✓ A timer that displays time ✓ A button to restart time ✓ Timer should always use latest reset time 🔥 (using `useEffectEvent` properly)
