`useEffect` in React

`useEffect` is a **hook** that lets you perform **side effects** in function components.
It's similar to lifecycle methods in class components: `componentDidMount`, `componentDidUpdate`, and `componentWillUnmount`.

Syntax

js Copy Edit

```
useEffect(() => { // your side effect logic here }, [dependencies]);
```

Example: Run Once on Mount

jsx Copy Edit

```
import React, { useEffect } from 'react'; function Example() { useEffect(() => { console.log("Component mounted!"); }, []); // Empty array = run only once return <h2>Hello, useEffect!</h2>; }
```

Example: Run When State Changes

jsx Copy Edit

```
import React, { useState, useEffect } from 'react'; function Counter() { const [count, setCount] = useState(0); useEffect(() => { console.log(`Count changed to: ${count}`); }, [count]); // Runs every time `count` changes return ( <div> <h2>Count: {count}</h2> <button onClick={() => setCount(count + 1)}>➕</button> </div> ); }
```

🧹 Cleanup Function (Like `componentWillUnmount`)

jsx

Copy Edit

```jsx
useEffect(() => { const timer = setInterval(() => { console.log("Tick"); }, 1000); return () => {
clearInterval(timer); // Cleanup console.log("Component unmounted or re-rendered"); }; }, []);
```

💡 What are side effects?

- Fetching API data
- Setting up subscriptions
- Directly manipulating the DOM
- Timers / intervals

⚠️ Dependency Array

Dependency array	What it means
`[]`	Run once (on mount)
`[value]`	Run when `value` changes
no array	Run on every render (⚠️ usually not ideal)
