Rules of Hooks

Only call Hooks at the top level. Don’t call Hooks inside loops, conditions, or nested functions.

Each time a component renders, React keeps track of the current state and other data associated with that component. When you call a Hook, React adds some state or other data to that component's internal state. If you call a Hook conditionally or inside a loop, React won't be able to keep track of the correct state and data associated with that component.

Calling Hooks at the top level of your component ensures that React can accurately track the state and data associated with that component. This allows React to work correctly, and ensures that your component behaves predictably. If you violate this rule, you may encounter bugs, performance issues, or unexpected behavior in your application.

Only call Hooks from React function components. Don’t call Hooks from regular JavaScript functions. (There is just one other valid place to call Hooks — your own custom Hooks. We’ll learn about them in a moment.)

Class components have their own set of lifecycle methods, such as componentDidMount, componentDidUpdate, and componentWillUnmount, which are called at different points during the component's lifecycle. When using class components, you can use these methods to manage state and perform other tasks. However, when using function components, you don't have access to these lifecycle methods.

Calling Hooks from within a functional component, you can manage state and perform other tasks in a way that is similar to using lifecycle methods in class components. This makes it easier to write and maintain functional components, and eliminates the need to convert them to class components in order to manage state or perform other tasks.

Additionally, calling Hooks from outside of a function component, such as in a regular function or a class method, would not work as expected. The reason is that Hooks rely on the component instance and React's internal state management, which are only available inside function components.

useState

<https://medium.com/weekly-webtips/react-hooks-for-beginners-sweet-and-a-little-complex-31a1225bd2f7>

useEffect

<https://javascript.plainenglish.io/react-hooks-how-to-use-useeffect-ecea3e90d84f>