**Difference between a useState() and a useEffect() hook**

The **useState()** and **useEffect()** hooks in React serve different purposes and are used for different aspects of managing the state and side-effects in functional components. Here's a brief rundown of their differences:

**useState()**

* **Purpose**: Used to manage state in functional components.
* **What it does**: **useState()** returns a state variable and a function to update that variable.
* **When to use**: When you want to keep track of a value that can change over time and re-render the component when it changes.

**Example:**

import React, { useState } from 'react';

const Counter = () => {

const [count, setCount] = useState(0); // `count` is the state variable, `setCount` is the function to update it.

return (

<div>

<p>{count}</p>

<button onClick={() => setCount(count + 1)}>Increment</button>

</div>

);

};

**useEffect()**

* **Purpose**: Used for side-effects in functional components.
* **What it does**: **useEffect** runs side-effect code (like fetching data, manipulating DOM directly, etc.) after the component renders.
* **When to use**: When you need to execute code in response to component lifecycle events like mounting, updating, or unmounting.

**Example**:

import React, { useEffect, useState } from 'react';

const UserProfile = ({ userId }) => {

const [user, setUser] = useState(null);

useEffect(() => {

// Fetch user data when the component mounts or `userId` changes

fetch(`/api/users/${userId}`)

.then(res => res.json())

.then(data => setUser(data));

}, [userId]); // Dependency array

return (

<div>

{user ? <p>{user.name}</p> : 'Loading...'}

</div>

);

};

Here are a few points to understand about **useEffect()**:

* **Mounting and Updating**: By default, the effect function passed to **useEffect()** runs after every render, including the first one. This behavior can be customized by providing a second argument to **useEffect()**, which is an array of dependencies.
* **Cleanup**: The function returned by **useEffect()** is a cleanup function that gets executed when the component unmounts. It also runs before the effect is re-run due to dependency changes.
* **Dependencies**: If you pass an empty array **[]** as the second argument to **useEffect()**, it behaves like **componentDidMount**, running the effect only once after the initial render.