



# UNDERRATED **REACT HOOKS** YOU'RE MISSING OUT ON



Chiamaka Umeh

React Hooks let us use essential React features in functional components.

Before Hooks, developers had to switch between functional and class components to optimize React.

Now, with Hooks, we can use functional components for all project sizes.



Chiamaka Umeh

If you prefer using functional components in React and want to learn about some underrated Hooks, this carousel is for you.

We'll explore three useful Hooks:

- `useImperativeHandle`
- `useLayoutEffect`
- `useDebugValue`.



Chiamaka Umeh

## **REACT** useImperativeHandle **HOOK**

The `useImperativeHandle` Hook, allows us to expose a value, state, or function inside a child component to the parent component through `ref`

In simpler terms, it gives you control over what methods or properties are accessible from a parent component when it interacts with a child functional component.



Chiamaka Umeh

# SYNTAX

```
useImperativeHandle(ref, createHandle, [dependencies])
```

**ref**: The ref passed down from the parent component

**createHandle** : The value to be exposed to the parent component

**dependencies** : An array of values that cause the Hook to rerun when changed





## **REACT** `useLayoutEffect` **HOOK**

The React `useLayoutEffect` Hook is a powerful tool for managing side effects in your functional components.

It's similar to the `useEffect` Hook, but with a key difference: it runs synchronously after all DOM mutations.



Chiamaka Umeh

# SYNTAX

```
useLayoutEffect(callback, [dependencies])
```

**callback**: The function that contains the side effect logic

**dependencies**: An array of dependencies. The callback function is run again when the value of any of the dependencies change



## REACT `useDebugValue` HOOK

The React `useDebugValue` Hook helps you add debugging information to your custom Hooks.

It's especially useful when you want to provide more insights into how your custom Hook works when inspecting it in React DevTools.



Chiamaka Umeh



# SYNTAX



```
useDebugValue(value)
```

If you're familiar with React DevTools, then you know that whenever a built-in React Hook like `useState` or `useRef` is used in a custom Hook, it will debug its respective values within React DevTools.



Chiamaka Umeh

Most of the time, we don't need these react hooks, but they can be really helpful in specific tricky situations, making our tasks easier.



Chiamaka Umeh

# GOT VALUE FROM THIS?

Like -comment- share- save



Chiamaka Umeh