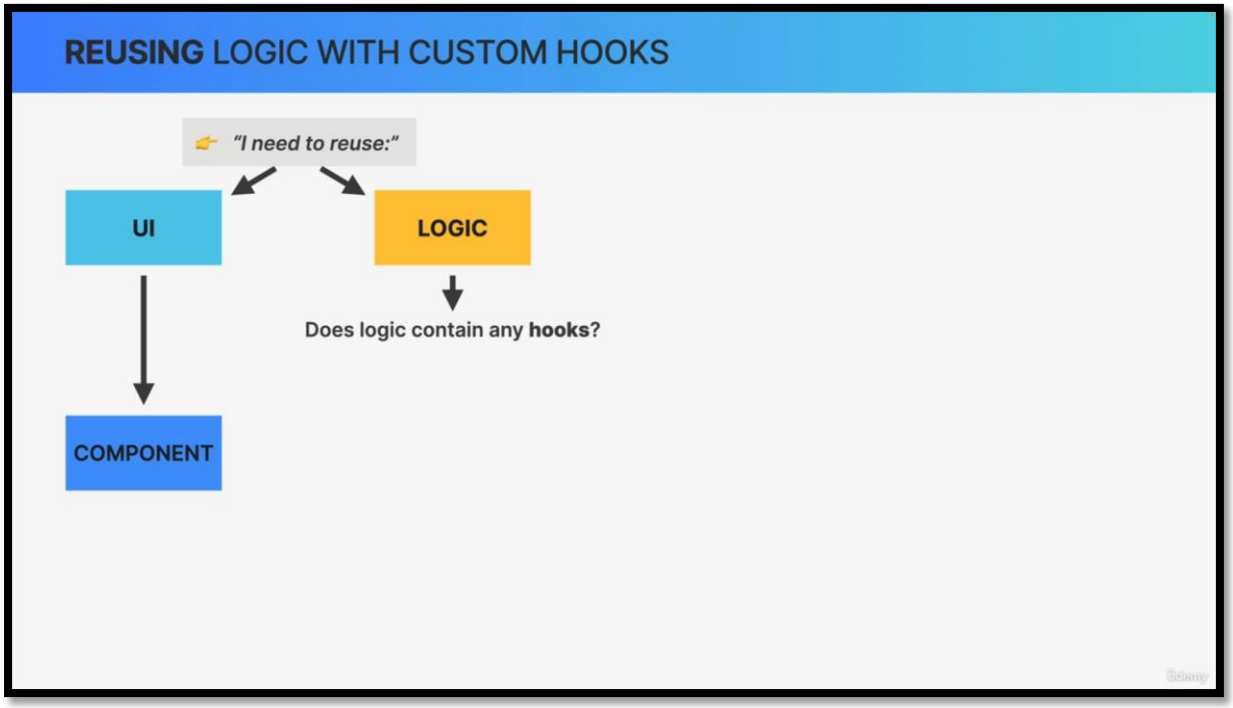
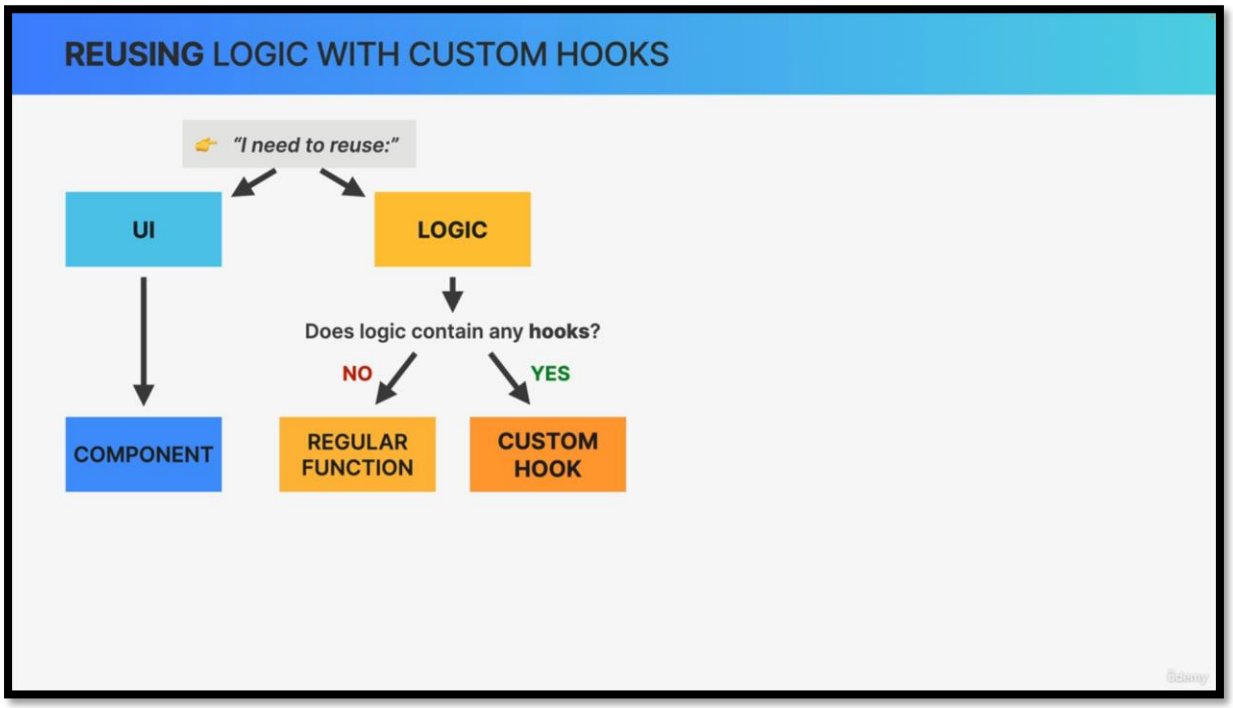
Custom hooks are all about reusability. In React, we have basically two types of things that we can reuse.

A piece of UI or a piece of logic.



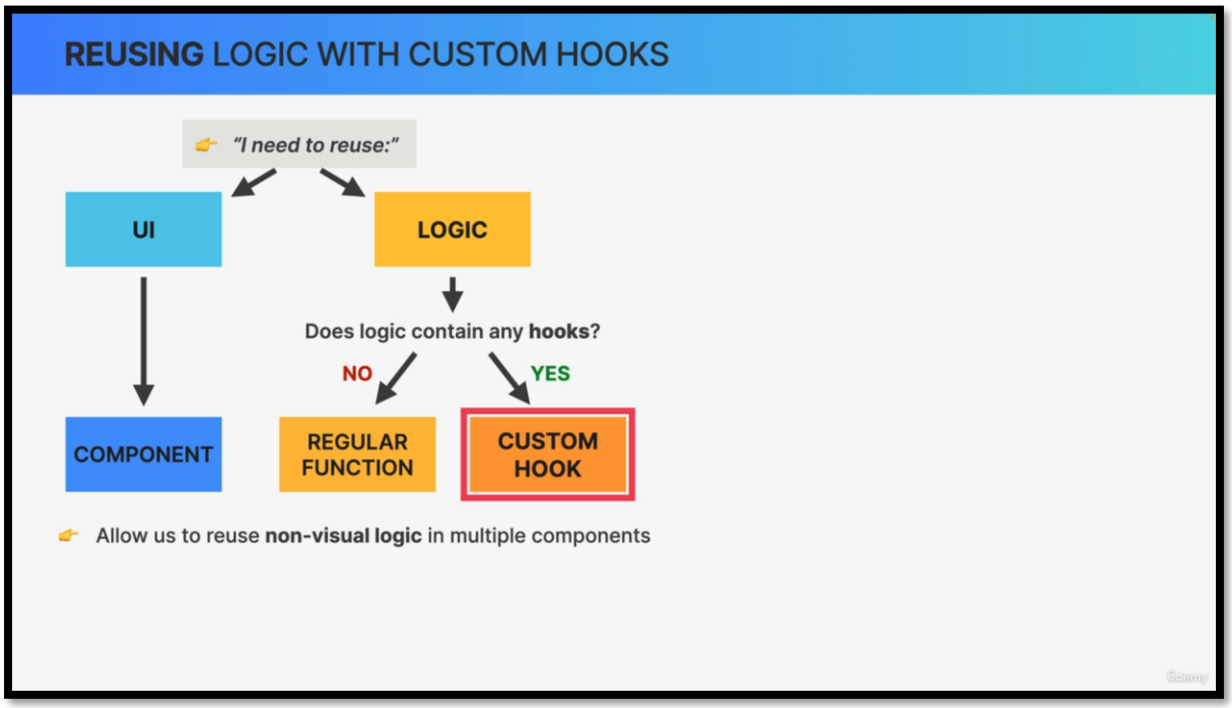
If we want to reuse a piece of UI, we use a component.



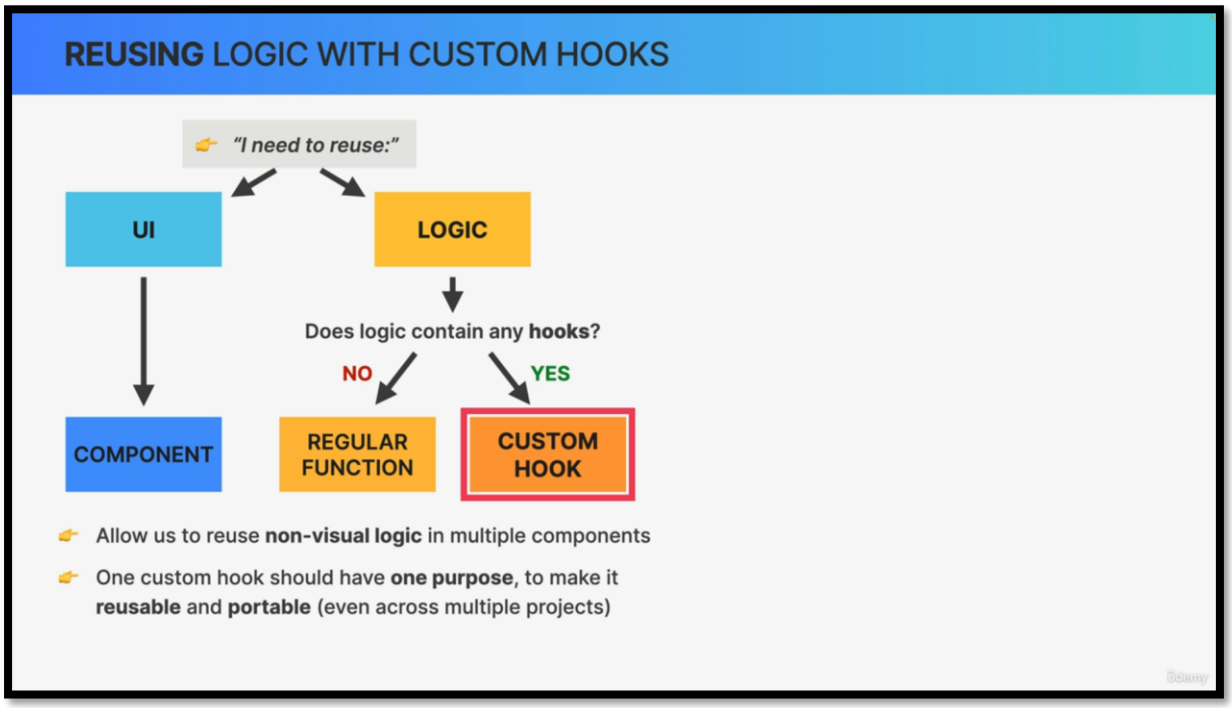
On the other hand, if you want to reuse logic in React, you first need to ask the question, does the logic that I want to reuse have any hooks?

If not, all you need is a regular function, which can live either inside or outside of your component.

However, if the logic does contain any React hook, you cannot just extract the logic into a regular function. Instead, what you need to create is a custom hook.

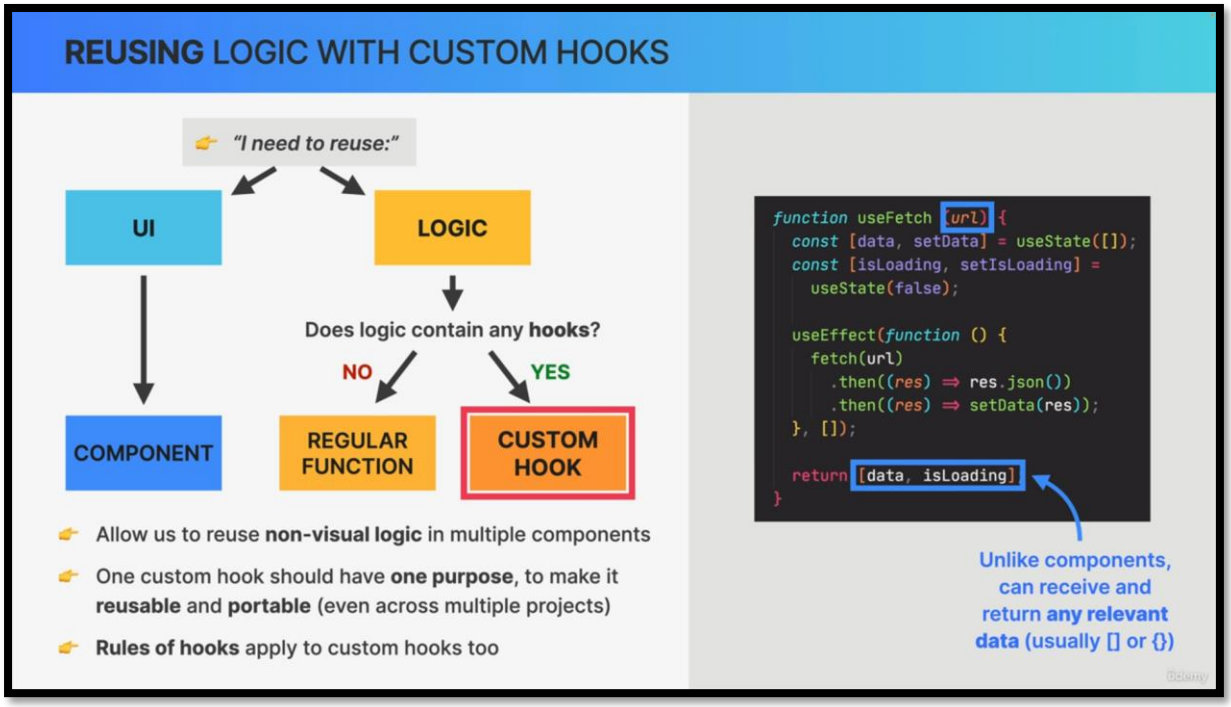


So basically, custom hooks allow us to reuse non-visual logic among multiple components.



Just like regular functions or components or effect, one hook should only have one purpose. So, it should only do one specific, well-defined thing. So, the idea is not to simply put all the hooks of a component into a custom hook.

The idea is to make custom hooks reusable and portable so that you can even use them in completely different projects.



So first, a custom hook is really just a JavaScript function, so it can receive and return any data that is relevant to this custom hook. In fact, it's very common to return an object or an array from a custom hook.