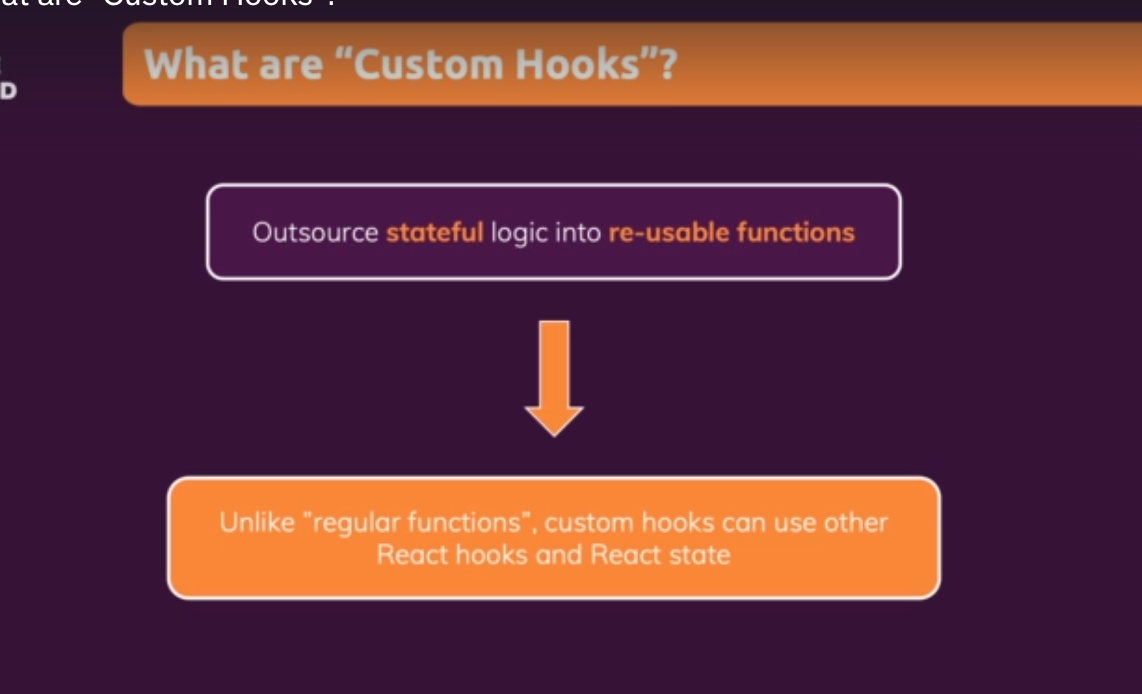
React custom hooks are used for outsourcing stateful logic into reusable functions.

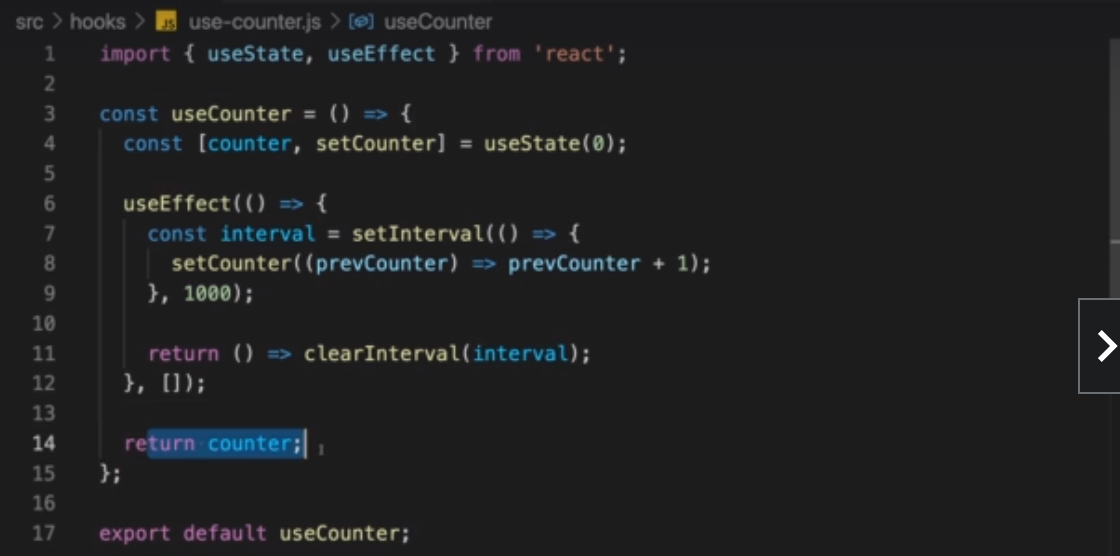


Using custom react hooks:-

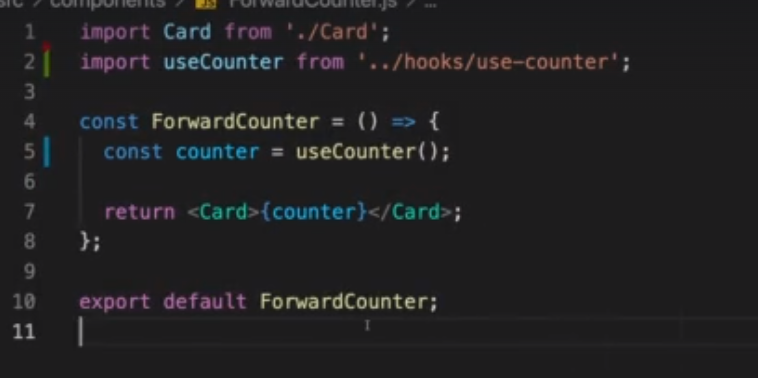


Inorder to create a new custom hook create a new file for example :- use-counter.js

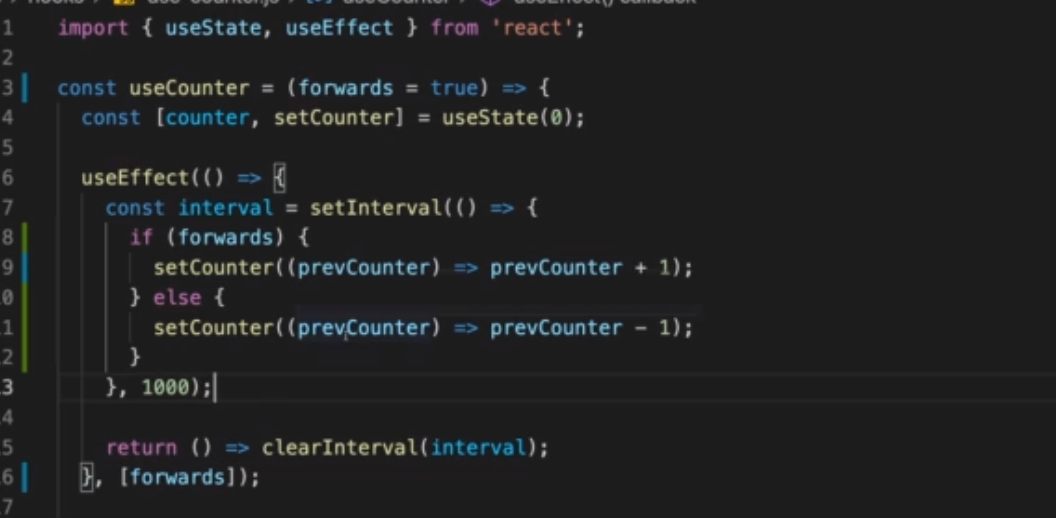
Custom hook function name should always start with use.



After creating a custom hook we need to use that custom hook inside a function. We need to import and call that counter inside the function inside which it is intended to use like below.



We can now make our custom hook configurable and reusable by passing in parameters to it as shown below. So that it works for both backward as well as forward counter.



Here we have set it to default true and pass forwards as an argument.

A more realistic project where a custom hook is used. One scenario where we might want to use a custom react hook is to use it for handling http loading, error state handlers for different types of requests. (like get, post, put etc).

Steps to write a reusable state based custom hook.

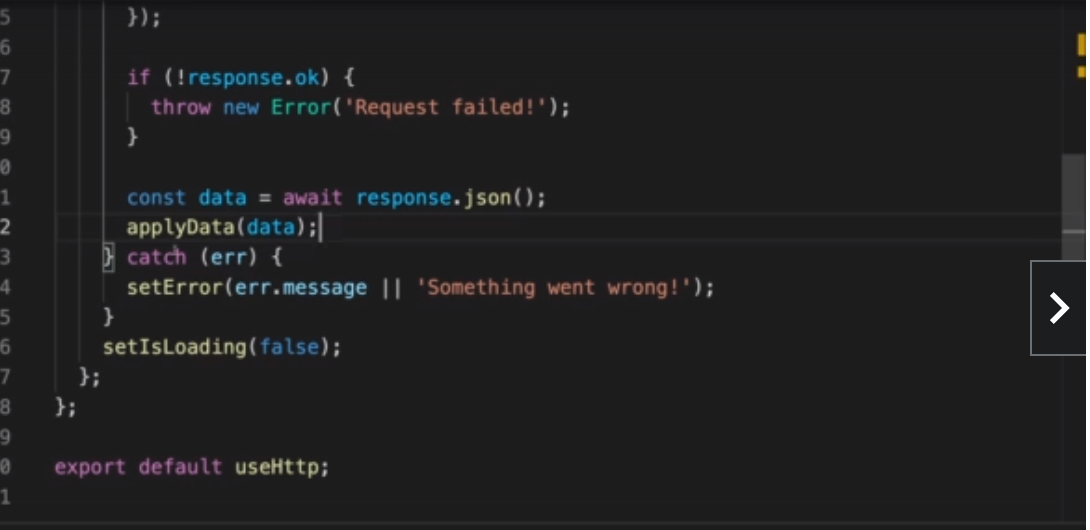
Create a hook folder (convention)

Create a file name use-http.(convention)

Create a function starting with use like (useHttp)

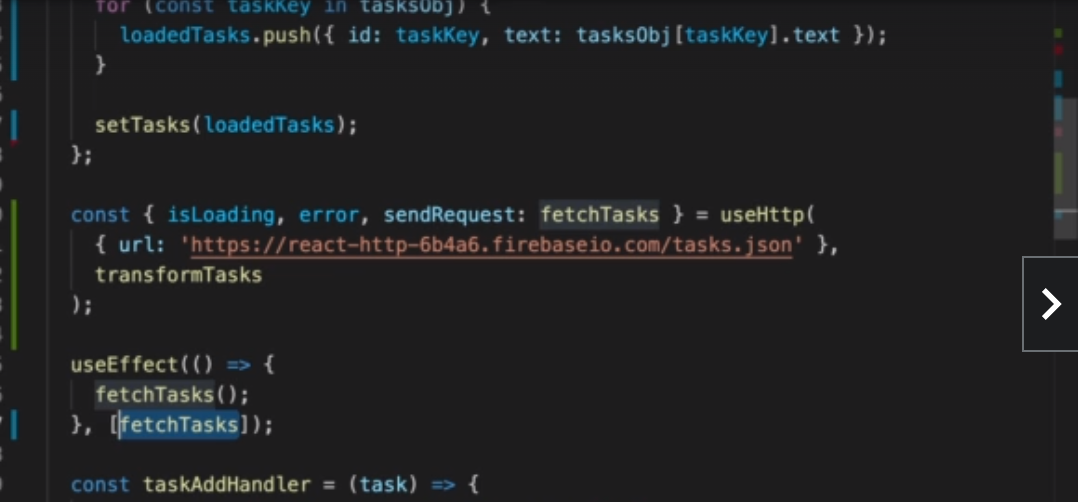
We want this hook to be reusable for any type of request and handle any type of state.





Apply data is a generic function that we are passing as an argument to process the returned data. This custom hook will return the following object return { isLoading, error, sendRequest,} as an object so that we can access them in the function wherever this is referred to.

Using this hook in the component function is straight forward. We need to call that hook and pass the relevant data to the function as below.



Final code for hook ➖



Using the custom hook in a component:-

