**Where to Place the Default Clause**

Standard convention is to place the default as the last clause. But you can place it before other cases too.

const food = "nuts";

switch (food) {

case "cake":

console.log("I like cake");

break;

case "pizza":

console.log("I like pizza");

break;

default:

console.log("I like all foods");

break;

case "ice cream":

console.log("I like ice cream");

break;

}

UseReducer

const [state, dispatch] = useReducer(reducer, init);

An alternative to **[useState](https://reactjs.org/docs/hooks-reference.html" \l "usestate)**. Accepts a reducer of type **(state, action) => newState**, and returns the current state paired with a **dispatch** method.

**useReducer** is usually preferable to **useState** when you have complex state logic that involves multiple sub-values or when the next state depends on the previous one.

**useReducer** also lets you optimize performance for components that trigger deep updates because [**you can pass dispatch down instead of callbacks**](https://reactjs.org/docs/hooks-faq.html#how-to-avoid-passing-callbacks-down).

This Hook accepts 2 arguments: a reducer function and an initial state.

useReducer(reducer, initialState);

It returns an array of two values which can be destructured to the current value of the state and a dispatch function.

Let's learn about its arguments and returned values:

* **state**: This is the current value of the initialState passed to the Hook.
* **reducer**: The reducer is a function that accepts the state and an action. Based on these arguments it determines how the value of state will change.
* **dispatch**: The dispatch function is how we pass an action to the reducer function. It dispatches the action to be used to update the state.
* **Action** is an object with type and payload as it properties.

Here’s the counter example :

const initialState = {count: 0};

function reducer(state, action) {

switch (action.type) {

case 'increment':

return {count: state.count + 1};

case 'decrement':

return {count: state.count - 1};

default:

throw new Error();

}

}

function Counter() {

const [state, dispatch] = useReducer(reducer, initialState);

return (

<>

Count: {state.count}

<button onClick={() => dispatch({type: 'decrement'})}>-</button>

<button onClick={() => dispatch({type: 'increment'})}>+</button>

</>

);

}

Specifying the initial state

There are two different ways to initialize **useReducer** state. You may choose either one depending on the use case. The simplest way is to pass the initial state as a second argument:

const [state, dispatch] = useReducer(

reducer,

{count: initialCount} );

https://codesandbox.io/s/quirky-jasper-k4unyk?file=/src/App.js