

# useReducer Hook

👉 **useReducer is used to manage state in our react application.**

👉 **In other words, useReducer works like a state management tool.**

## **What is State Management?**

👉 **State Management is used to manage all states of application in a simple way.**

👉 **Always use the useReducer hook when you have a lot of states and methods to handle.**

First We will make increment and decrement using **useState**

```
import './App.css';
import React, { useState } from "react";

function App() {
  const [count, setCount] = useState(0);

  const Increase = () => {
    setCount(count + 1);
  };

  const Decrease = () => {
    setCount(count - 1);
  };

  return (
    <div>
      <h1>Counter : {count}</h1>
      <button onClick={Increase}>Increase</button>
      <button onClick={Decrease}>Decrease</button>
    </div>
  );
}

export default App;
```

# Counter : 3

Increase Decrease

Now learn **useReducer**

**\*\*useReducer() will take 2 arguments**

1) reducer function → which will manage all states

2) default state

Handwritten code and annotations for the `useReducer` hook:

```
② const initialState = { count: 0 }

③ // now create reduce function
const reducer = (state, action) => {
  return {
    count: state.count + 1
  }
}
```

Annotations for the reducer function:

- `state`: current value
- `action`: updation

```
const APP = () => {
  ① const [state, dispatch] = useReducer(reducer, initialState)
  // ...
  const increase = () => {
    dispatch()
  }
}
```

Annotations for the `useReducer` call:

- `state`: current value
- `dispatch`: function used for updation
- `reducer`: It is a function manages all state

When you wanted to implement only increment →

```
import React, { useReducer } from "react";

const initialState = { count: 0 };

const reducer = (state, action) => {
  return { count: state.count + 1 };
};

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

  const Increase = () => {
    dispatch();
  };

  return (
    <div>
      <h1>Counter : {state.count}</h1>
      <button onClick={Increase}>Increase</button>
    </div>
  );
}

export default App;
```

# Counter : 6

Increase

When you wanted to do increment decrement and reset we have use switch case in action

```
import React, { useReducer } from "react";

const initialState = { count: 0 };

const reducer = (state, action) => {
  switch (action.type) {
    case "increase":
      return { count: state.count + 1 };
    case "decrease":
      return { count: state.count - 1 };
    case "reset":
      return { count: (state.count = 0) };
    default:
      return state;
  }
};

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

  const Increase = () => {
    dispatch({ type: "increase" });
  };
  const Decrease = () => {
    dispatch({ type: "decrease" });
  };
  const Reset = () => {
    dispatch({ type: "reset" });
  };

  return (
    <div>
      <h1>Counter : {state.count}</h1>
      <button onClick={Increase}>Increase</button>
      <button onClick={Decrease}>Decrease</button>
      <button onClick={Reset}>Reset</button>
    </div>
  );
}
```

**Counter : 3**

Increase Decrease Reset

