Advanced React useReducer

LIGHTHOUSE LABS

AGENDA

Why useReducer?

useReducer Hook

Demo

LIGHTHOUSE LABS

Why useReducer?

useState()

The most fundamental state management hook in React, useState() maintains a single state variable.

```
const [counter, setCounter] = useState(0);
```

Why useReducer?

useState()

- useState is simple to use and understand.

- As your React app grows larger, your app is filled with setState for every piece of

  state within your app

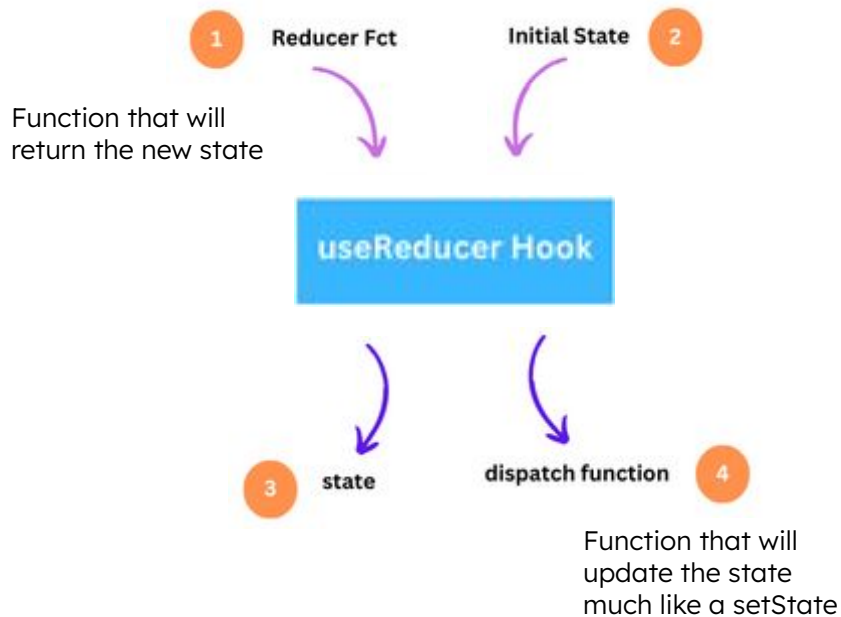- It makes it very difficult to to track state mutations
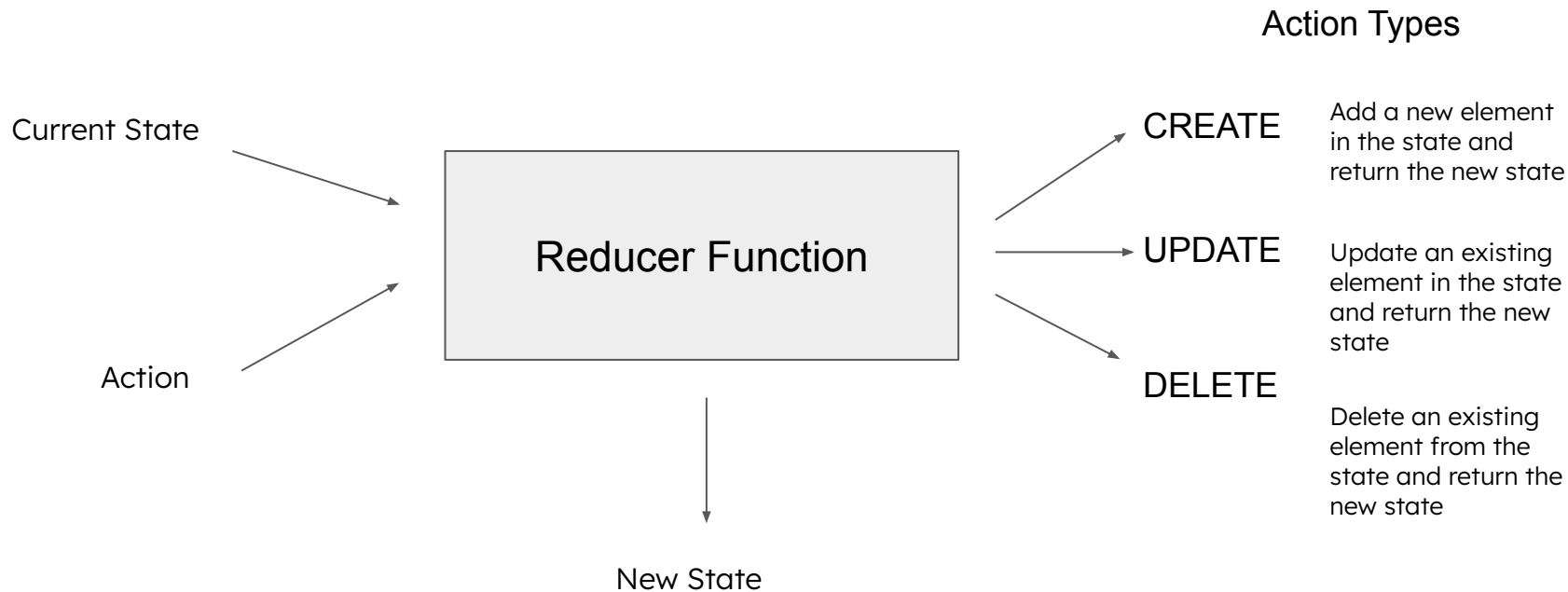
Why useReducer?

useReducer()

useReducer is a React hook that helps us manage the state in our app:

- Centralize all the state mutations in one place

- A clearer separation of concerns

- Reducers are pure functions and can be more easily tested

- State updates are declarative

- There's a bit of a learning curve initially

useReducer

Reducer Fct — Function that will return the new state

Initial State

useReducer Hook

state

dispatch function — Function that will update the state much like a setState

useReducer

Action Types

Current State → Reducer Function

Action → Reducer Function

Reducer Function → New State

Reducer Function → CREATE — Add a new element in the state and return the new state

Reducer Function → UPDATE — Update an existing element in the state and return the new state

Reducer Function → DELETE — Delete an existing element from the state and return the new state

Demo

- The goal of the demo is to demonstrate the useReducer hook.

- We're going to keep a the UI to a minimum on purpose to focus on the state mutations without too many distractions.

- We start by using conventional useState to mutate our state and we're going to transition to a useReducer step-by-step.

- We're going to add a few optimizations depending on time.

Questions?

LIGHTHOUSE LABS