React Third-party State Management Playbook

Picking a Library

Cory House
Consultant

@housecor | reactjsconsulting.com

Overview

Why use third-party state?

Third-party state categories

Picking a library

Why use Third-party State?

State Hook Limitations

Only accessible in React

```
const [loading, setLoading] = useState(false);
const [state, dispatch] = useReducer(myReducer, initialState);
const countRef = useRef(0);
const selectionContext = useContext(SelectionContext);
```

Can’t “protect” state, not global

Clunky DX, not global

Doesn’t render

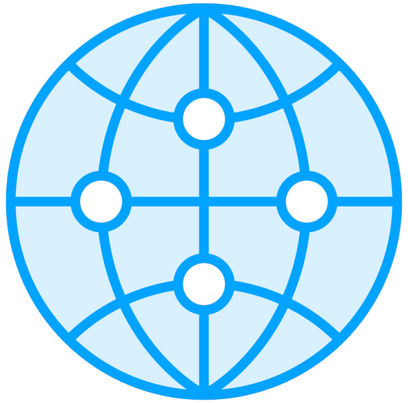
For data that changes *infrequently*

No granular render optimizations

Why Third-party State?

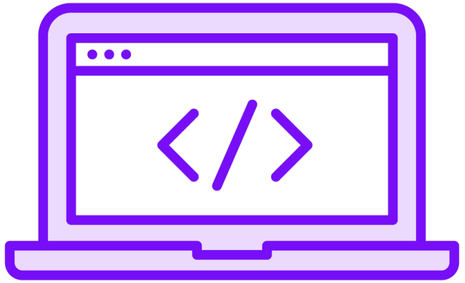
Performance

Render optimization
Caching



Flexibility

Access outside React
Global by default
“Protect” state



DX

Less code
Devtools

Third-party State Options

State Libraries by Category

Redux

Unidirectional

recoil

Jōtai

Atomic

Mobx valtio

Proxy

TanStack Query

APOLLO

SWR

RTK Query

Remote

XSTATE

State machine

Redux

Unidirectional

Store state outside of React
Protect changes to the state
You create the state change API
Redux: one store, Zustand: multiple

Atomic

Store small pieces of state
State is unprotected
Granular state updates

Store small pieces of state
Wrap state in a proxy object
State is mutable
Automatically optimizes performance

Proxy

TanStack Query

APOLLO

SWR

RTK Query

Remote

Store data fetched from a server
Handle loading and error state
Cache and dedupe requests
Automatically refetch when needed.

Handles complex state logic
Enforces state transitions
Generate state charts

State machine

Third-party State: Key Differences

General vs. Specific

State Libraries by Category

General

Redux

Unidirectional

Recoil

Jōtai

Atomic

Mobx

valtio

Proxy

Fetched Data

TanStack Query

APOLLO

SWR

RTK Query

Remote

Complex Rules

XSTATE

State machine

Third-party State: Key Differences

General vs. Specific

Mutable vs. Immutable

Mutable vs Immutable State

Immutable

Redux

Unidirectional

Recoil

Jōtai

Atomic

Mutable

Mobx valtio

Proxy

TanStack Query

APOLLO

SWR

RTK Query

Remote

Varies

XSTATE

State machine

Immutable

Immutable vs. Mutable State

Zustand

```
const useStore = create((set) => ({
  count: 1,
  inc: () => set((state) => (
    { count: state.count + 1 }
  )),
}));
```

Valtio

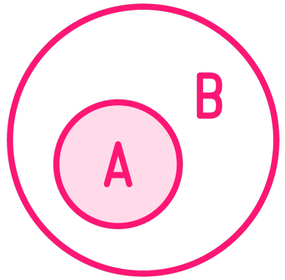
```
const state = proxy({
  count: 1
});

const incrementCount = () => {++state.count};
```

Third-party State: Key Differences

General vs. Specific

Mutable vs. Immutable



External vs. Internal

Internal vs External

External

Redux

Unidirectional

Varies

TanStack Query

APOLLO

SWR

RTK Query

Remote

Internal

Recoil

Jōtai

Atomic

External

External

Mobx valtio

Proxy

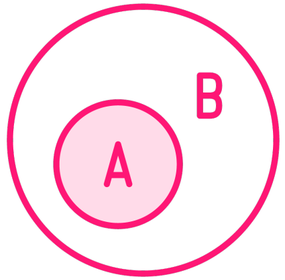
XSTATE

State machine

Third-party State: Key Differences

General vs. Specific

Mutable vs. Immutable



External vs. Internal

Auto vs. Manual

Render Optimization

Manual

Redux

Unidirectional

recoil

Jōtai

Atomic

Automatic

TanStack Query

APOLLO

SWR

RTK Query

Remote

Automatic

Mobx

valtio

Proxy

XSTATE

State machine

Manual vs. Automatic Render Optimization

Zustand – Manual selector

```
const Component = () => {
  const count = useStore(state => state.count)
  return <p>{count}</p>
}
```

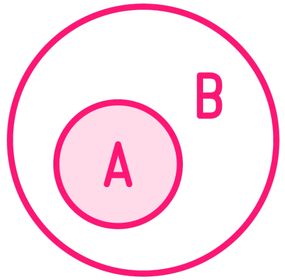
Valtio - Automatic

```
const Component = () => {
  const snap = useSnapshot(store)
  return <p>{snap.count}</p>;
}
```

Third-party State: Key Differences

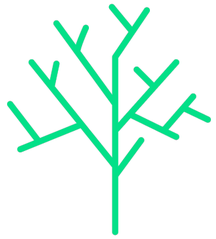
General vs. Specific

Mutable vs. Immutable



External vs. Internal

Auto vs. Manual



One Store vs. Multiple

Number of Stores

One

Redux

Many

recoil

Jōtai

Mobx

valtio

One

TanStack Query

APOLLO

SWR

RTK Query

Many

XSTATE

Top-down vs Bottom-up

Top-down

Start with the overview, then add details.

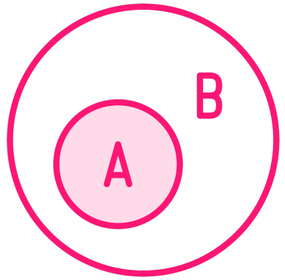
Bottom-up

Start with small pieces of state, then compose them.

Third-party State: Key Differences

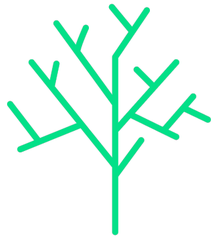
General vs. Specific

Mutable vs. Immutable

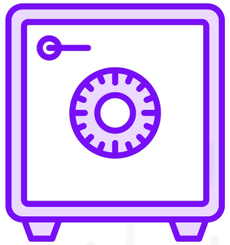


External vs. Internal

Auto vs. Manual



One Store vs. Multiple



Protected vs. Unprotected

“Protected” State

Protected

Redux

Unidirectional

Direct access

Recoil

Jōtai

Atomic

]v[Mobx valtio

Proxy

Varies

TanStack Query

APOLLO

SWR

RTK Query

Remote

Protected

XSTATE

State machine

Protected vs. Unprotected State

Zustand

```
// userStore.js
export const useStore = create((set) => ({
  user: {
    id: 1,
    name: "Cory House",
  },
  logout: () => set({ user: null }),
}));

const { user, logout } = useStore();
```

Jotai

```
// userAtom.js
export const userAtom = atom({
  id: 1,
  name: "Cory House",
});

const [user, setUser] = useAtom(userAtom);
```

Summary

Third-party State Categories

- Unidirectional
- Proxy
- Atomic
- Server
- State machine

How to pick a library

- General vs. specific
- Mutable vs. immutable
- External vs. internal
- Automatic vs. manual render optimization
- One store vs. multiple
- Protected vs. unprotected

Up Next:

Unidirectional State