

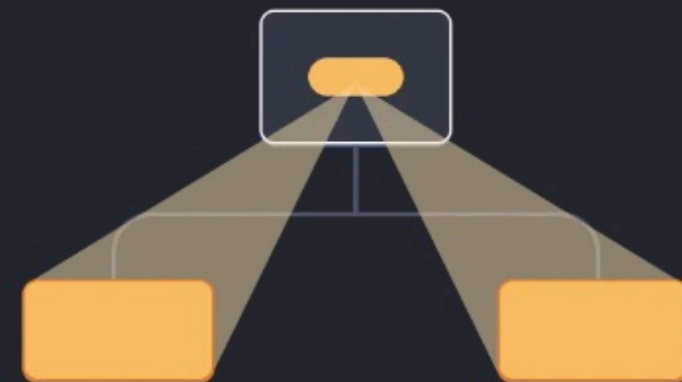
# 7.2 | Context, State management , recoil

Repo for today - <https://github.com/100xdevs-cohort-2/week-7>

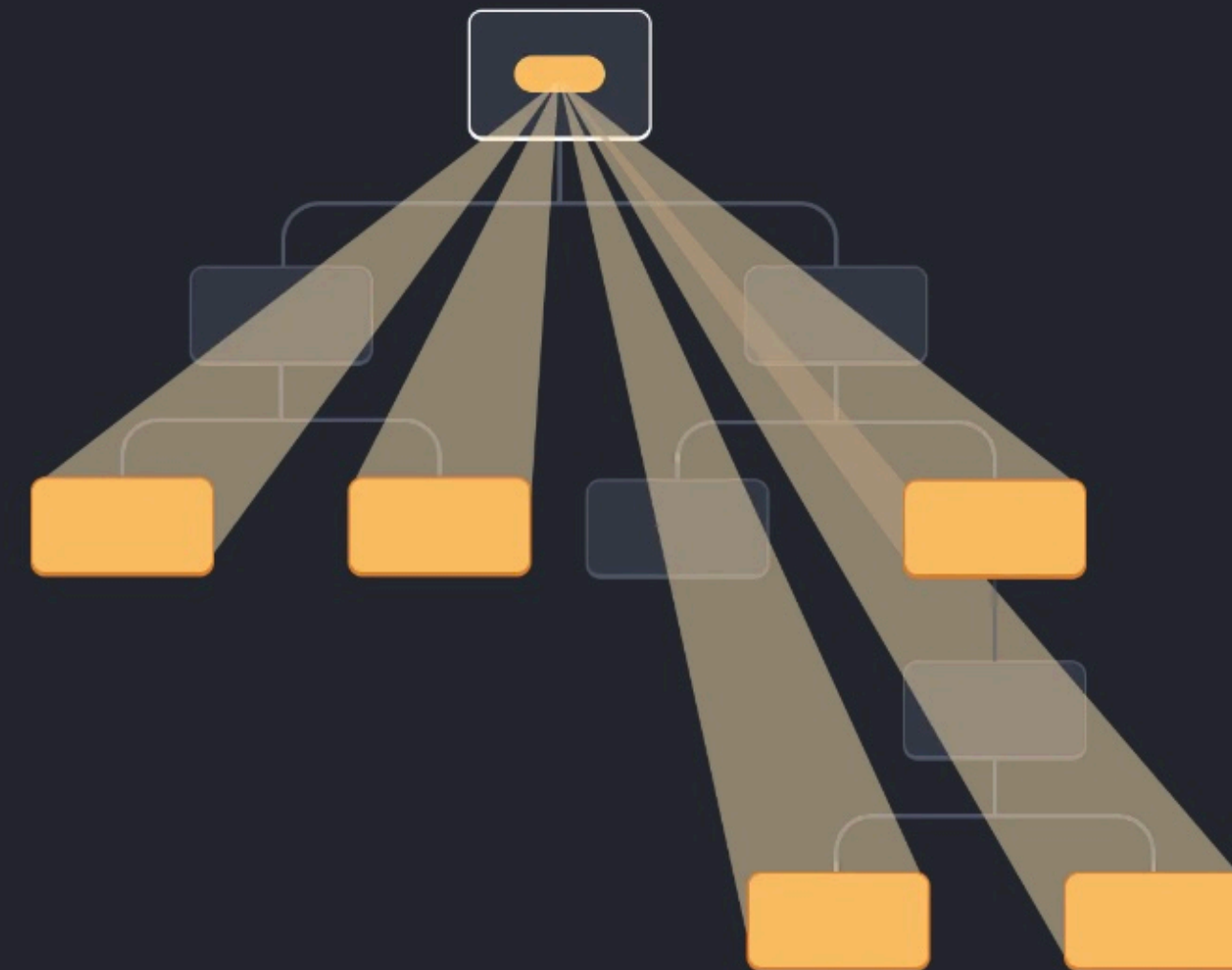
# Context

**Let's you teleport state values to distant children**  
**Helps you get rid of prop drilling**

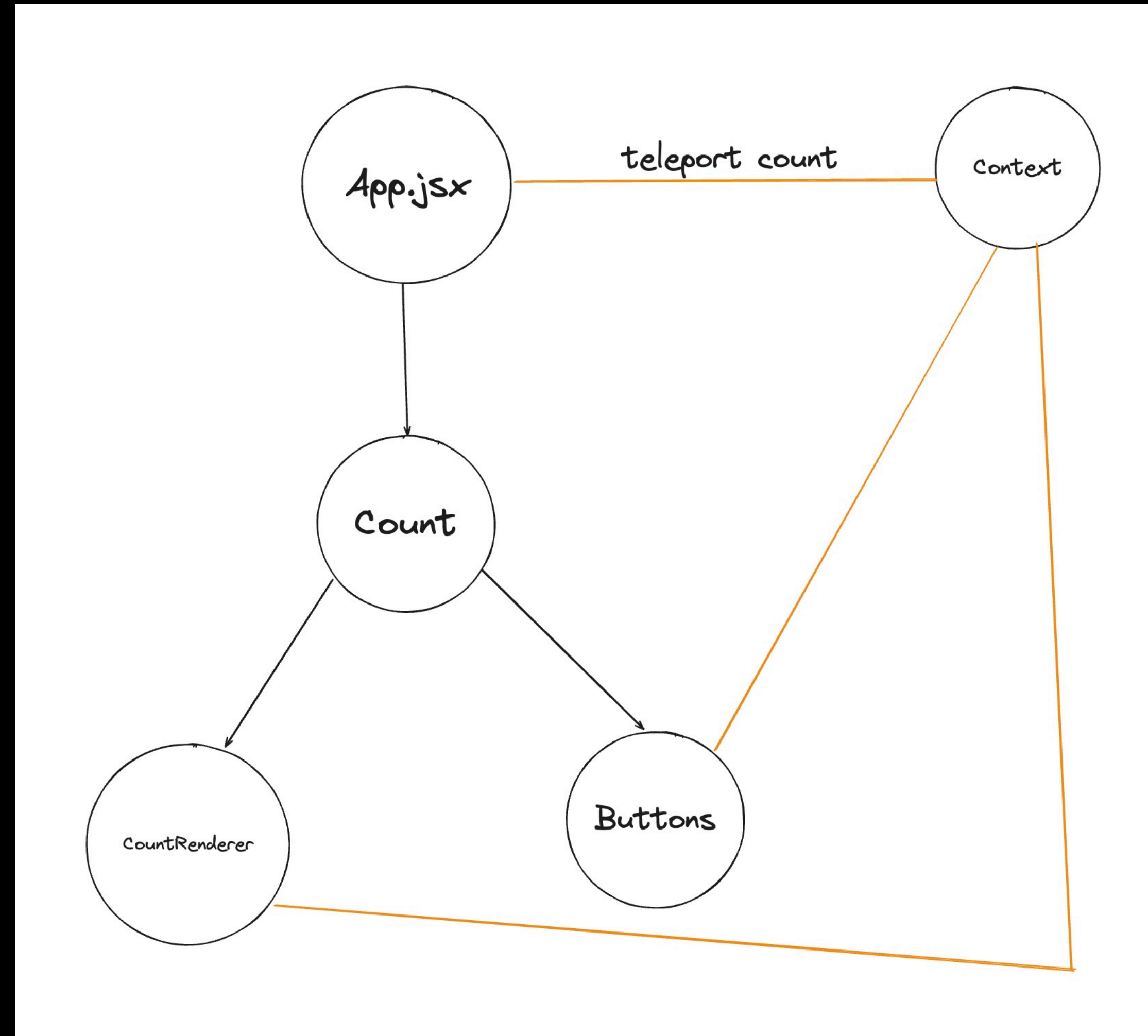
Using context in close children



Using context in distant children



# Yesterday, we did the Context API



# Yesterday, we did the Context API

```
✓ function App() {  
  const [count, setCount] = useState(0);  
  
  // wrap anyone that wants to use the teleported value inside a provider  
  return (  
    <div>  
      <CountContext.Provider value={count}>  
        <Count setCount={setCount} />  
      </CountContext.Provider>  
    </div>  
  )  
}
```

```
function CountRenderer() {  
  const count = useContext(CountContext);  
  
  return <div>  
    {count}  
  </div>  
}
```

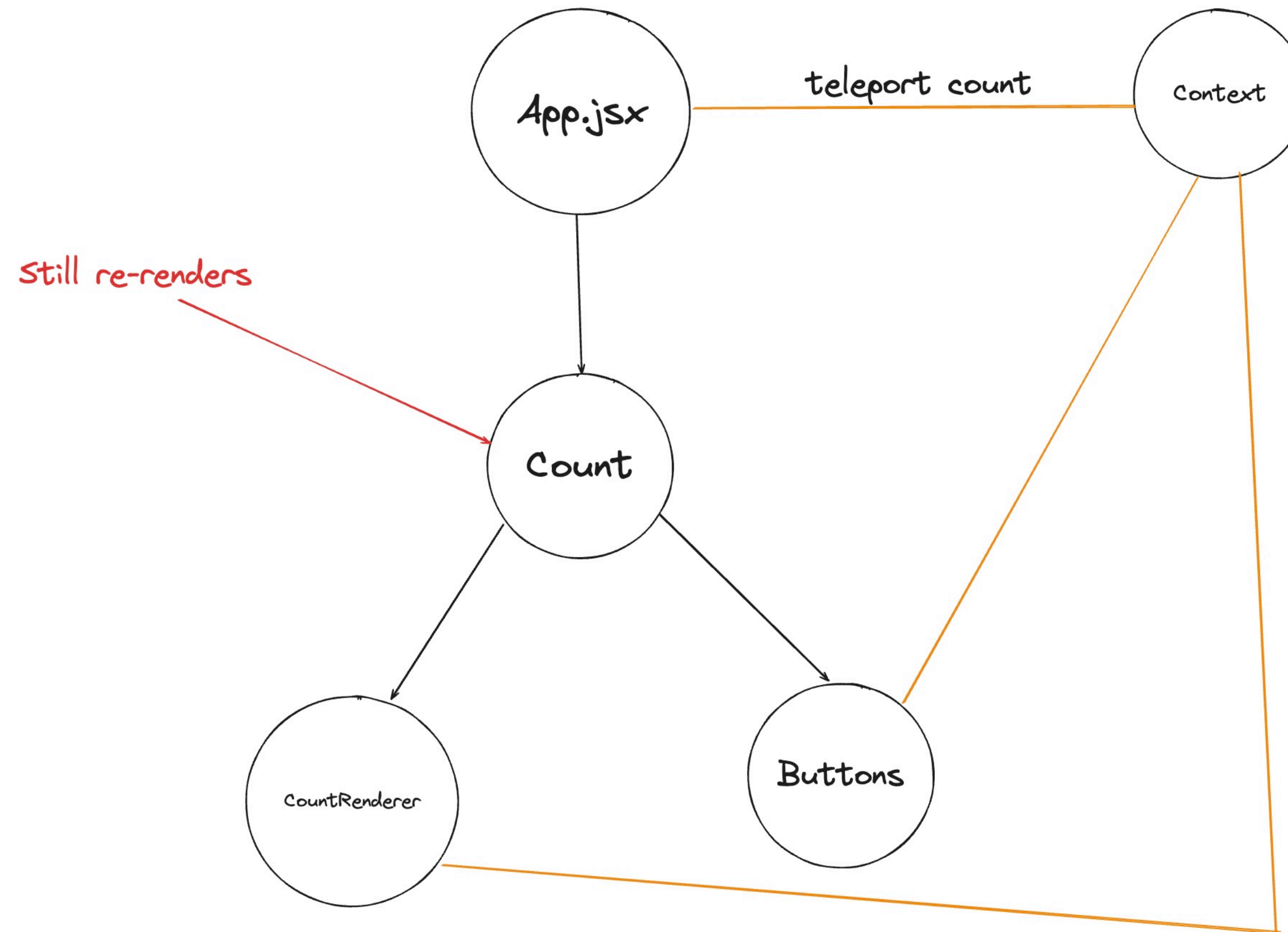
App.jsx

```
1 import { createContext } from "react";  
2  
3 export const CountContext = createContext(0);
```

context.js

# Problem with context?

## Doesn't fix re-rendering, only fixes prop drilling



## 7.2 | Context, State management, recoil



# State management using Recoil

## What is state management

**A cleaner way to store the state of your app**  
**Until now, the cleanest thing you can do is use the Context API.**  
**It lets you teleport state**

**But there are better solutions that get rid of the problems that Context Api has**  
**(unnecessary re-renders)**



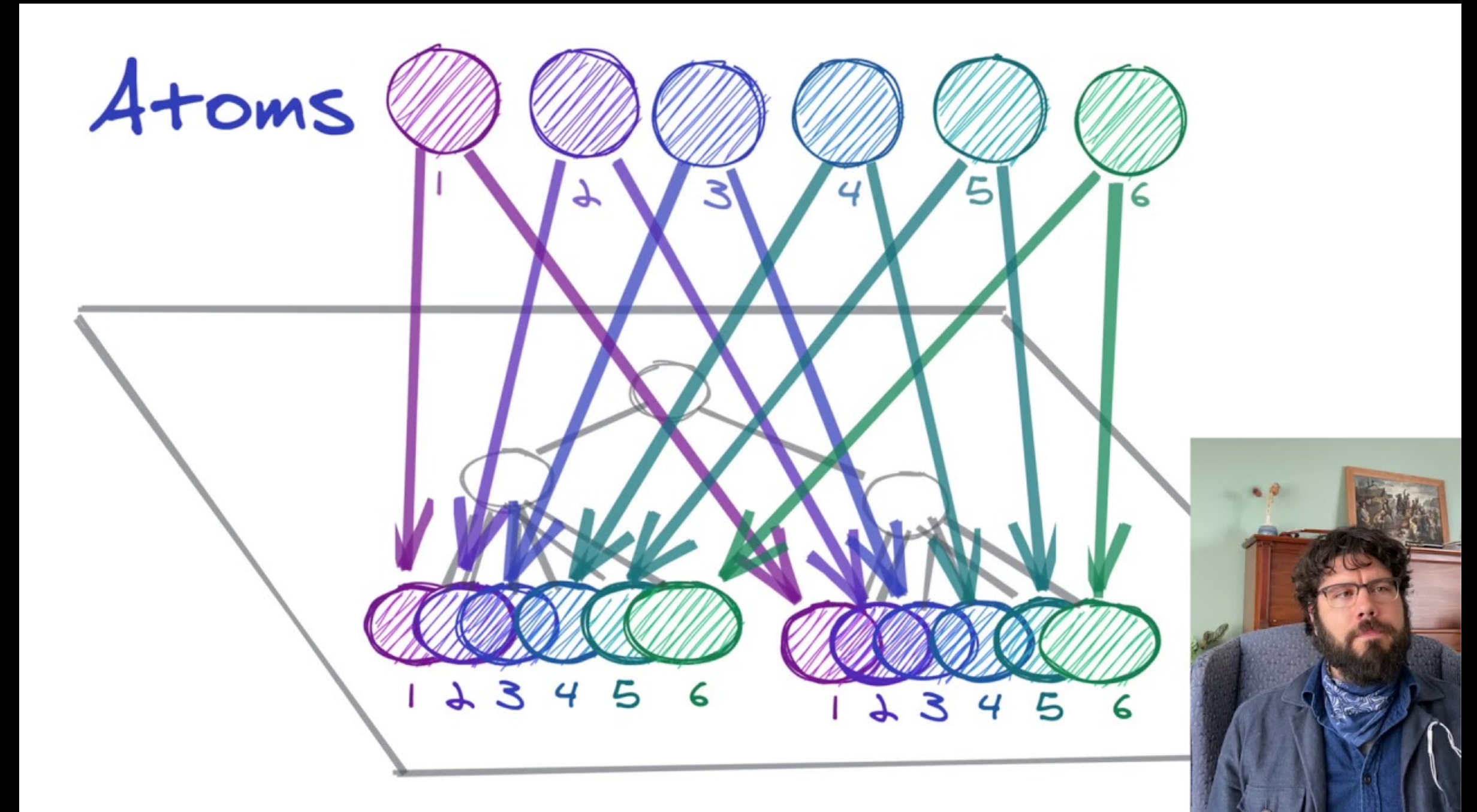
# State management using Recoil

## Recoil

A state management library for React  
Written by some ex React folks (I think)

Other popular ones -

1. Zustand
2. Redux





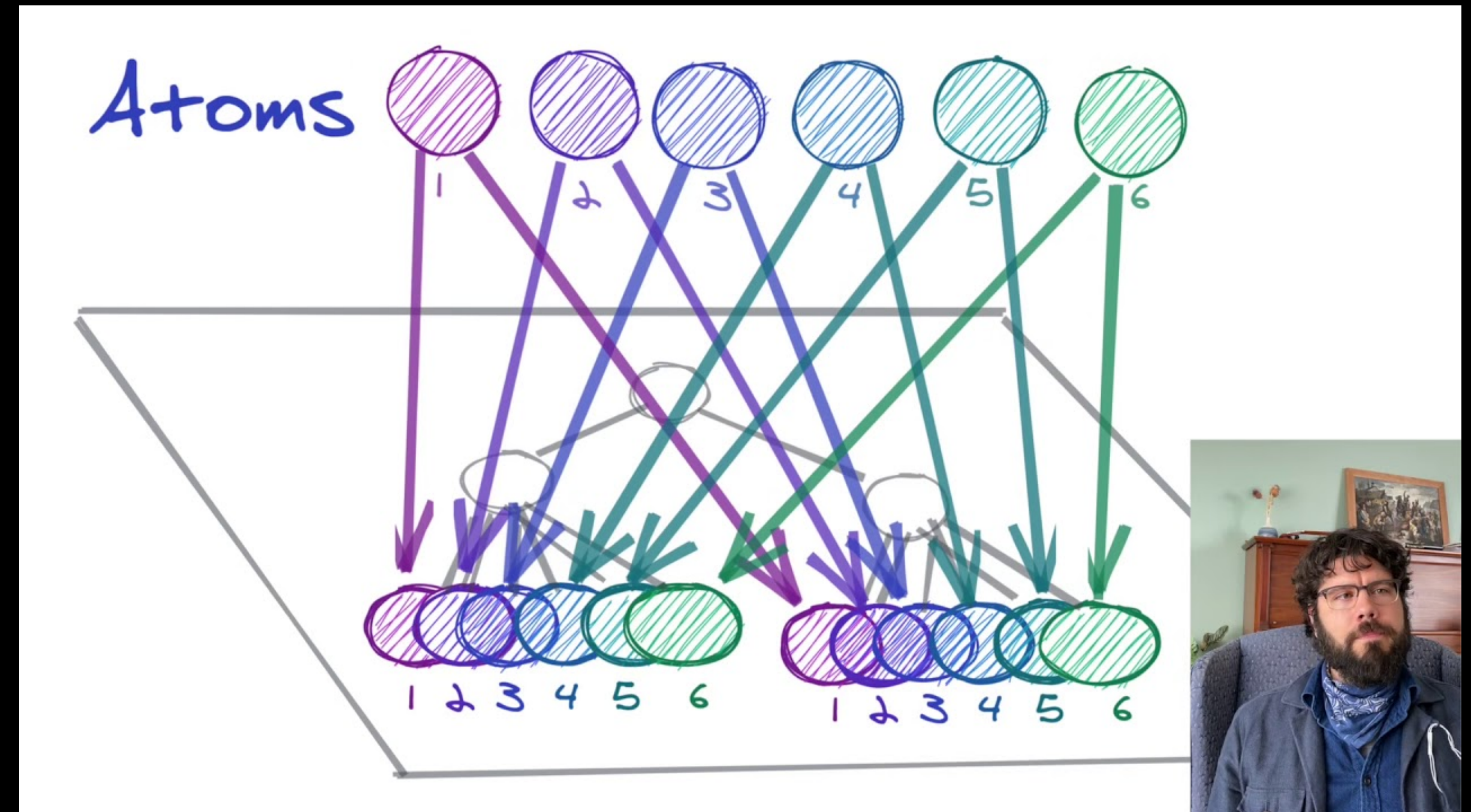
# State management using Recoil

## Recoil

Has a concept of an **atom** to store the state

An atom can be defined outside the component

Can be teleported to any component



# State management using Recoil

**Recoil**

**npm install recoil**

# State management using Recoil

**npm install recoil**

**Things to learn -**

**RecoilRoot**

**atom**

**useRecoilState**

**useRecoilValue**

**useSetRecoilState**

**selector**

# State management using Recoil

`npm install recoil`

Things to learn -  
`RecoilRoot`  
`atom`  
`useRecoilState`  
`useRecoilValue`  
`useSetRecoilState`  
`selector`

Let's say I ask you to render  
**IT IS EVEN**  
if the current count is even

