Writed by Ow-mahan-wO ⬛ https://github.com/Ow-mahan-wO

First of all, I want to give a basic explanation about stateManagement.

In our state managers, we have three types of architecture: flux, proxy, and atomic, each of which has a different structure and is not very similar. In flux architecture we have redux and zustand for example and in proxy architecture we have Mobx for example and in Atomic architecture we have Jotai or Recoil for example

What is Jotai ?

Jotai is a relatively new state management library for React which works with (atom). It's simple, but make no mistakes, it's a robust library. Jotai is based on the new Recoil pattern and library by Facebook.

Why we should learn Jotai ?!

Redux is often used in large-scale projects due to its ability to maintain states at a large scale. If we want to use state managers in a small project, Redux doesn't seem reasonable and we have to use the context API. However, Jotai comes to our aid and makes managing states in the program easy with its atomic structure and is very fast and convenient. This is one reason to learn Jotai, and another reason

may be that we have the opportunity to get acquainted with other architectures and state managers.

How to learn Jotai ?

We learn step by step with the official jotai document and its practice.
Let's go to start.

Lesson 1:

at first you most install jotai in your app:

npm command : npm install jotai

yarn command : yarn add jotai

Creating your first atom

Jotai atoms are small isolated pieces of state. Ideally, one atom contains very small data. Here's how you create your first atom.

```
import { atom } from 'jotai' ;
const counter = atom(0);
```

t is as simple to use as React's integrated useState hook, but all state is globally accessible.

```
const [count, setCounter] = useAtom(counter);
```

The atom we created is to be passed to useState hook with the help of jotai useAtom function, which returns an array, where the 1st element is the value of atom, and the 2nd element is a function used to set the value of the atom.

Jotai considers anything to be an atom so you can create any type of atom you want whether it is atom of objects, arrays, or nested objects.

```
const friendObj = atom({ name: "Shahin", online: false });

const cities = atom([ "Shiraz", "Tehran", "Kerman" ]);
const nestedObj = atom({ friend1: { name: "Korosh", age: 20 } })
```

Example Lesson 1 :

in this example we write a very small app with jotai so with Click in "Click for increased " Button add 1 count to counter variable and more ….

```
import { atom, useAtom } from 'jotai';

const counter = atom(0);

export default function ExLesson1() {
  const [count, setCounter] = useAtom(counter);
  const setCountHandler = () => setCounter(count => count +=1);
  return (
    <div>
      <button onClick={ setCountHandler }>Click for Increased</button>
      <p>{count}</p>
    </div>
  )
```

As we learned in the above material, we made a number type variable using atoms and stored it inside a variable, and using useAtom, which has a structure similar to useState in React, we were able to update or use it.

Lesson 2 :

Persisting state value

In this lesson, we will take a look at how we can persist the state value to **localStorage** with jotai **atoms**. Persisting state to **localStorage** can be challenging. You might want to persist the user's preferences or data for their next session.

Jotai **atomWithStorage** is a special kind of atom that automatically syncs the value provided to it with localstorage or sessionStorage, and picks the value upon the first load automatically. It's available in the **jotai/utils** module. To persist our theme atom simply create it with the **atomWithStorage** atom.

Note: In first Parameter we pass keyword for toggle and second Parameter we pass that status (true/false)

```
const theme = atomWithStorage('dark', false)
```

Example Lesson 2 :

In this Example we have a Toggler button for change new Theme in app So Click that button and to next Step we Refresh page and Theme changed and not be set to default theme . this is Amazing

```
import { useAtom } from 'jotai';
import { atomWithStorage } from 'jotai/utils';

const theme = atomWithStorage('dark', false);

export default function ExLesson2() {
  const [Theme, setTheme] = useAtom(theme);
  const handleThemeClick = () => setTheme(!Theme);
  return (
    <div className={Theme? 'dark': 'light'}>
      <h1>This is a theme switcher</h1>
      <button onClick={handleThemeClick}>{Theme? 'DARK': 'LIGHT'}</button>
    </div>
```