Redux-day-01[What is Redux]

npm i redux

What is Redux?

🔄 Redux is a state management library for JavaScript applications. It allows you to manage the state of your application in a centralized location called the store.

File Structure:

📁 A good file structure helps organize your Redux code effectively.

🗂️ Commonly, you'll have folders for actions, reducers, and sometimes selectors.

📝 Actions define the types of changes you want to make to your state.

🔄 Reducers specify how the state changes in response to those actions.

🔍 Selectors are used to extract specific pieces of state.

getState:

🕵️‍♂️ getState is a function in Redux that retrieves the current state of the application.

subscribe:

📧 subscribe is a method in Redux that adds a listener function to be called whenever an action is dispatched and the state might have changed.

dispatch:

📤 dispatch is a function in Redux that sends an action to the Redux store. It's like a messenger telling Redux what happened.

3 Principles of Redux

1:SINGLE SOURCE OF TRUTH(STATE IS READ ONLY)

2:CHANGES ARE MADE WITH PURE FUNCTIONS

3:STATE IS READ ONLY

Create store:-

import {legacy\_createStore} from "redux"

const initalstate={}

export const store=legacy\_createStore(reducer,initalstate)

The basic idea behind Redux is that your application’s state is represented by a single object called the store. The store is read-only, which means you can’t directly modify its state. Instead, you need to dispatch actions to the store, which describe what changes you want to make to the state.

Actions are plain JavaScript objects that have a type property, which describes the type of action you’re dispatching, and a payload property, which contains any additional data that the action needs to make the changes to the state.

When you dispatch an action to the store, the store calls a reducer function with the current state of the store and the action as arguments. The reducer function takes the current state and the action, applies the changes described by the action to the state, and returns a new state object.

The store then updates its state to the new state object returned by the reducer function. Any components that are connected to the store through React-Redux will be re-rendered with the updated state.

Here’s a summary of the key concepts in Redux:

* The store is the single source of truth for your application’s state.
* Actions are plain JavaScript objects that describe changes to the state.
* Reducers are functions that take the current state and an action, and return a new state.

- Dispatching an action to the store triggers the store to update its state by calling the reducer function with the current state and the action.

I hope that helps! Let me know if you have any further questions.

In a social media app, you can use Redux to manage the state of the application. Here are some examples of how Redux can be useful in a social media app:

1. Authentication state: Redux can store the user’s authentication state and allow the app to retrieve it from any component. This can include the user’s login status, token, and other related information.

2. User profile information: Redux can store the user’s profile information, such as name, email, and profile picture. This data can be easily accessed by any component, and any changes to the profile can be updated in real-time.

3. Posts and comments: Redux can store the posts and comments made by the user or their friends. This allows the app to display them in the correct order and update them in real-time.

4. Notifications: Redux can store the user’s notification state and allow the app to retrieve it from any component. This includes new messages, friend requests, or other relevant notifications.

By using Redux, you can simplify the management of the state of the application and make it more predictable. It also allows for better separation of concerns, making it easier to maintain and scale the app over time.