REDUX



* Redux is a library for managing and updating application state.
* It provides a centralized “store” for state that is shared across your entire application.
* The goal of redux is to provide scalable and predictable state management.

1. What is Store?
2. A Store is a container that holds and manages your application’s global state.
3. The store is the center of every redux application. It has the ability to update the state
4. Accessing the state should never be done directly and is achieved through functions provided by the store

import { createStore } from "redux";

import RootReducer from '../Redux/Reducers/index';

const Store = createStore(RootReducer);

export default Store;

1. What is a Reducer?
2. A Reducer is a plain JavaScript function that accepts the store’s current state and an action and returns the new state.
3. Reducers calculate the new state based on the action it receives. Reducers are the only way the store’s current change can be changed within the redux.
4. ONE\_WAY DATA FLOW
5. There are
6. State: - The current data used in the app
7. View: - The user interface displayed to users
8. Action: - Events that a user can take to change state
9. Store: - The UI is updates based on the new state

State 🡪 View 🡪 Action 🡪 Store







1. THE FLOW OF INFORMATION
2.  The state holds the current data used by the app’s components.



1. The view components display that state data.
2. The action will be the view updated to display the new state.

Note: Redux helps separate the state, the view and actions by requiring that the state be managed by a single source

INSTALL REDUX

* npm install redux

note: create Store and combine Reducers packages will be Imported with redux installing.

Examples:

Actions: -

export const USER = 'User';

export const anyUser = (user) => {

    return {

        type: 'User',

        payload: user,

    }; };

Reducer: - (index.js)

import { combineReducers } from "redux";

import Reducer from "../../complete redux/Redux/Reducer/Reducer";

const RootReducer = combineReducers({

    users: Reducer,

});

export default RootReducer;

Reducer: -(main.js)

import {USER,} from ‘. /Actions’;

const initialState = {

    users: []

};

const Reducer = (state = initialState, action) => {

    switch (action.type) {

        case 'User':

            return {

                ...state,

                users: [...state.users, action.payload],

            };

default:

            return state;

    }

}

export default Reducer;

React-Redux

* Npm install react-redux
* The few of the resources imported from react-redux are:

1. Provider
2. useSelector
3. useDispatch

Provider: -

* The provider component makes the redux store available to a nested child component.
* The store should be passed in as the store prop.

Example: -

import React from 'react';

import { Provider } from 'react-redux';

import Store from ‘. /Redux/Store';

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

  <Provider store={Store}>

    <Routing />

  </Provider>,

);

useSelector Hook: -

* The useSelector hook extracts state data from redux store using a selector function each time the state is updated.
* Selectors used with useSelector can be pre-defined functions or inline selectors.

Example: -

import React, { useState } from 'react'

import { useDispatch, useSelector } from 'react-redux'

const Example = () => {

    const dispatch = useDispatch();

    const users = useSelector((state) => state.users.users);

return (

//your program part

);

} export default Example

Selector: -

🡪 In redux selectors are user-defined functions that extract specific pieces of information from a store state value.

🡪Selectors are pure functions that takes the state as an argument and react components can use selectors to get specific data from the store.

useDispatch Hook: -

* The useDispatch hook returns a reference to the dispatch() method.
* A react component defines dispatch and assigns the reference returned by useDispatch() and dispatch() can then be used within the component to dispatch action objects.

Example:

import React from 'react';

import {useDispatch} from 'react-redux';

const Component = () => {

const dispatch = useDispatch();

return (

<button onClick = {() => dispatch(

{ type: ‘buttonClicked’}

)} >Click Me

</button>

); };