Store Management

martes, 1 de noviembre de 2022 13:16

* Subscribing to stores

→ Stores allow you to subscribe handler funcs that are invoked every time the store completes the dispatch of an action.

* Saving to local storage

→ Using the store's subscribe() func, we listen for state changes and save those changes to localStorage under the key 'redux-store'
→ Create a store and check if any data has been saved to localStorage and if so, load data as our initial state

```
const store = createStore(
     combineReducers({ colors, sort }),
     (localStorage['redux-store']) ?
          JSON.parse(localStorage['redux-store']) :
          {}
)
store.subscribe(() => {
     localStorage['redux-store'] = JSON.stringify(store.getState())
})
console.log('current color count', store.getState().colors.length)
console.log('current state', store.getState())
store.dispatch({
     type: "ADD_COLOR",
     id: uuid.v4(),
     title: "Party Pink",
     color: "#F142FF",
     timestamp: new Date().toString()
})
```

we subscribe a listener to the store that saves the store state everytime we dispatch an action.

↳ Everytime we refresh the page, we add the same color.

- stores hold and manage state data in Redux applications,
- and the only way to change state data is by dispatching actions through the store.
- The store holds application state as a single object.
- State mutations are managed through reducers.
- Stores are created by supplying a reducer along with optional data for the initial state.
- we can subscribe listeners to our store (and unsubscribe them later), and they will be invoked every time the store finishes dispatching an action.

* Action Creators

→ Action objects are simply JS literals.
→ Action creators are functions that create such literals
→ We can simplify this by adding action creator for each action type.

```
import C from './constants'
export const removeColor = id =>
     ({
          type: C.REMOVE_COLOR,
          id
     })
export const rateColor = (id, rating) =>
     ({
          type: C.RATE_COLOR,
          id,
          rating
     })

// whenever we need to disptah RATE_COLOR or REMOVE_COLOR
store.dispatch( removeColor("3315e1p5-3abl-0p523-30e4-8001l8yf2412") )
store.dispatch( rateColor("441e0p2-9ab4-0p523-30e4-8001l8yf2412", 5) )
```

* Middleware

→ Middleware is the glue between to layers (pieces) of software.

→ Redux has also middleware for the store's dispatch pipeline

[illegible]

$\downarrow$

series of functions
executed in a row in
the process of dispatching
an action