Redux 2.3_CW Exercises

ex01: setting up your redux project

challenge

Create a Redux store with a reducer and initial state.

You can use a simple counter reducer for this exercise.

solution

https://codesandbox.io/s/redux-without-toolkit-with-react-forked-y6gwyc

ex02: create the logging middleware

challenge

1. Create a new file called `loggerMiddleware.js` in your project's source directory.
2. In `loggerMiddleware.js`, define the logging middleware following the structure provided in the exercise description

solution

```
// loggerMiddleware.js
const loggerMiddleware = (store) => (next) => (action) => {
  console.log('Current state:', store.getState())
  console.log('Action:', action)
  next(action)
  console.log('New state:', store.getState())
}

export default loggerMiddleware
```

COPY

understanding

loggerMiddleware is a Redux middleware that logs the current state, dispatched actions, and the new state after an action is processed.

Middleware Structure:

```
const loggerMiddleware = (store) => (next) => (action) => {
  // Middleware logic
}
```

COPY

- The `loggerMiddleware` is a function that takes a `store` as its argument.
- Inside this function, there is another function that takes `next` as its argument.
- Inside the second function, there is yet another function that takes `action` as its argument.

ex03**: integrate middleware into redux Store**

challenge

In your Redux store setup file (e.g., `store.js`), import the `applyMiddleware` function from Redux and the `loggerMiddleware` you created.

solution

```
// store.js
import { createStore, applyMiddleware } from 'redux'
import loggerMiddleware from './loggerMiddleware'

// Your reducer and initial state
const reducer = (state = 0, action) => {
  // Your reducer logic here
}

const store = createStore(reducer, applyMiddleware(loggerMiddleware))

export default store
```

COPY

ex04: test the logging middleware

challenge

1. Click the "Increment" and "Decrement" buttons to trigger actions. Observe how the logging middleware logs the current state, action, and new state to the console.
2. Verify that the middleware is working correctly by checking the log messages and confirming that they match the state changes caused by the dispatched actions.

entire solution

https://codesandbox.io/s/redux-without-toolkit-with-react-rx2-3-forked-ctyqt5