Redux

FreeCodeCamp.org Redux in Front-End development libraries

- In redux all state updates are triggered by dispatching actions
- State is ready only,so the reducer must return a new copy of state and never modify state directly.
- CP- make the action types as read only
- first principle of Redux: all app state is held in a single state object in the store.

1. Redux store

Type-Object
method-CreateStore()
parameters-reducer

Methods-
Redux.createStore(param);<-creates the redux store with reducer parameters
storeName.getState()<- retrieves the current state from the store.
1.2 storeName.subscribe()<-allows you to subscribe listener function to the store, which is called whenever an action is dispatched against the store.

2. Action

- Action are js objects with type and payload(data)

Type-Object
method-NA
parameters-NA

Declaration-

```
Const actionName=(
        type : ‘type of action’,
        payload/data : ”Data in that action”)
```

3. Action Creator

- Action creator helps the redux store in updating state when an action occurs

Type-function

returns-action
parameters-NA

Declaration-

```
Function actioncreatorName(){
    Return action}
```

4. Dispatch

- Dispatch method is what you use to dispatch actions to the Redux store
- Calling store.dispatch() and passing the value returned from an action creator(return value is an action) back to the store.

Type-function
Returns-NA
parameters-actionCreator/action

Declaration-

```
store.dispatch(actionCreator()/{ type: 'LOGIN' });
```

5. Reducer

- After an action is created and dispatched , the redux store needs to know how to respond to that action, this is the job of the reducer.
- Is a pure function that takes state and action and returns a new state.

Type-function
Returns-new state
parameters-state,action

Declaration-

```
Const reducerName =(state <-param, action <-param){
    Return new state }
```

1.2 Store subscribe

- One simple use for this method is to subscribe a function to your store that simply logs a message every time an action is received and the store is updated.

Type-method
Returns-NA
parameters-function

Declaration-

storeName.subscribe(function_Name())

6. Combine Reducers

- In order to combine multiple reducers together this method.
- Key is the name by which we want to call our reducer
- Value is the actual name of the reducer
- Redux will then used the key as the new name for our reducers

Type-method
Returns-new state
parameters-key,value

Declaration-

```
Const rootReducerName=Redux.combineReducers({
        Key1 :  value1(The original name of reducer1),
        Key2 : value2(The original name of reducer2)});
```

Note - the value 1 and 2 are function but we will use them as variable names cause in react we can do so.

7. Redux Thunk middleware

- At some point you will need to call asynchronous endpoints in your redux app.
- Redux provides middleware especially for this purpose

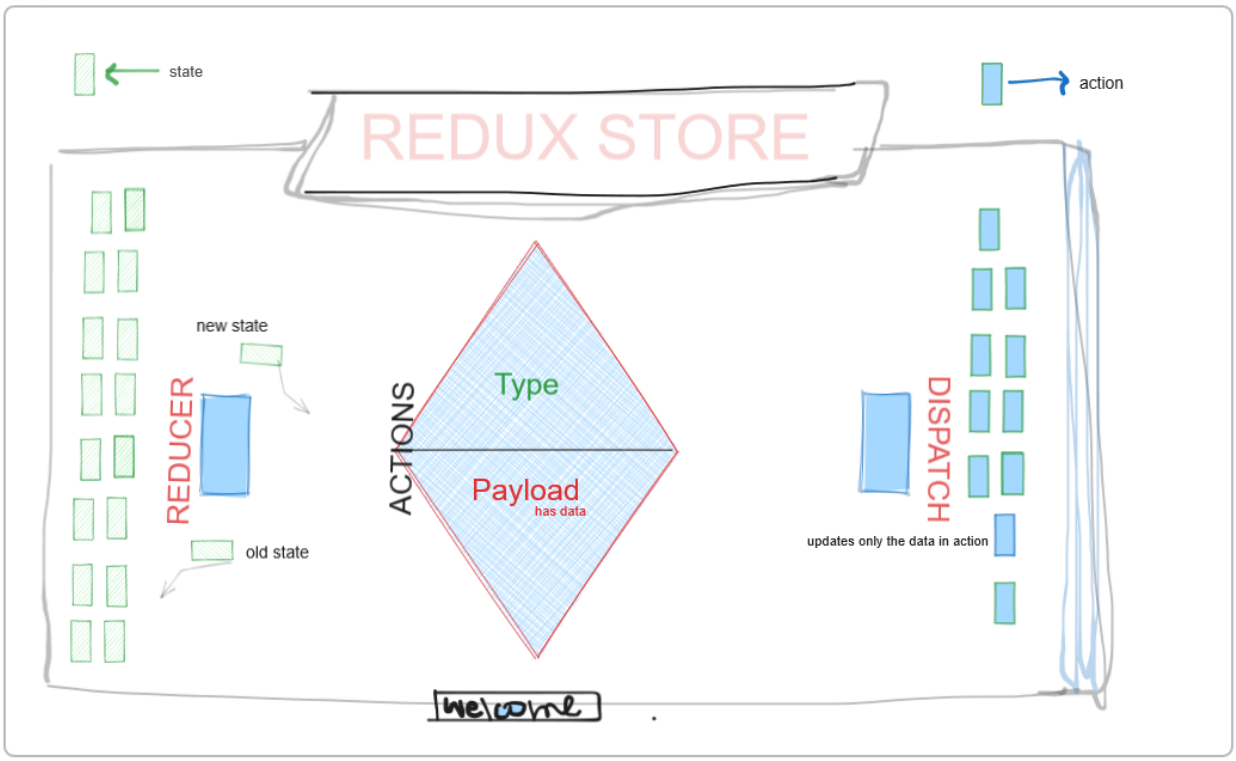
Type-parameter to createStore()
Returns-NA
parameters-RedyxThunk.default

Declaration-

Redux.applyMiddleware(ReduxThunk.default)

8. Since state should be immutable get better at using esp-6 function that maintain immutability and learn about object.assign.

Frame 1


state
action
REDUX STORE
new state
REDUCER
old state
ACTIONS
Type
Payload
has data
DISPATCH
updates only the data in action
welcome