Learn Redux

```
import { createStore } from "redux";

// Action creators
const increment = () => {
return {
type: 'INCREMENT',
};
};

const decrement = () => {
return {
type: 'DECREMENT',
};
};

// Reducer
const counter = (state = 0, action) => {
switch (action.type) {
case 'INCREMENT':
return state + 1;
case 'DECREMENT':
return state - 1;
default:
return state;
}
};

// Store
const store = createStore(counter);

store.subscribe(()=>{
console.log(store.getState());
})

//dispatch
store.dispatch(increment());
```

Step By Step Process

Redux can be divided into 4 main simples steps

1. Action => This are functions that assign a type and payload to the reducer. They tell the reducer what they want to do

```
export const increaseByNumber = (number)=>{
return {
type: "increaseByNumber",
payload: number
}
}
```

2. Reducers => The perform the actual function. If an action is [Hungry], then the Reducer will be [Eating]

```
const counterReducer = (state=0, action) =>{

switch(action.type){
case "INCREMENT":
return state+1;
case "DECREMENT":
return state-1;

case "increaseByNumber":
return state + action.payload
default:
return state
}

}

export default counterReducer;
```

3. Store => They store the reducer as a global state so that any component can access it and use it

   ```
   const store = createStore(counter);
   ```

4. Dispatch => This will push the action to the reducer

   ```
   store.dispatch(increment());
   ```

PROJECT

index.js (This is the root file) [This is where i will create the store]

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';
import App from './App';
import reportWebVitals from './reportWebVitals';
import {createStore, combineReducers} from "redux"
import combinder from './reducers/Combinder';
import { composeWithDevTools } from 'redux-devtools-extension';
import { Provider } from 'react-redux';




const myStore = createStore(
combinder,
composeWithDevTools()
);




const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
<React.StrictMode>
<Provider store={myStore}>
<App />
</Provider>

</React.StrictMode>
);
reportWebVitals();
```

Create 2 folders Reducers and Actions

In the **Reducers folder**, create three javascript

counter.js

```
const counterReducer = (state=0, action) =>{

switch(action.type){
case "INCREMENT":
return state+1;
case "DECREMENT":
return state-1;

case "increaseByNumber":
return state + action.payload
default:
return state
}

}

export default counterReducer;
```

isLogged.js

```
const loggedReducer = (state=false, action) =>{

switch(action.type){
case "SIGN_IN":
return !state;
default:
return state
}
}
export default loggedReducer;
```

Combinder.js

```
import counterReducer from "./counter";
import loggedReducer from "./isLogged";
import {combineReducers} from "redux"


const combinder = combineReducers({
counter: counterReducer,
isLogged: loggedReducer
})

export default combinder;
```

In Action Folder, create allAction.js

```
export const increament = ()=>{
return {
type: "INCREMENT"
}
}

export const decreament = ()=>{
return {
type: "DECREMENT"
}
}

export const increaseByNumber = (number)=>{
return {
type: "increaseByNumber",
payload: number
}
}

export const Login = ()=>{
return{
type:"SIGN_IN"
}
}
```

In the App.js implement the state

```
import React from 'react'
import { useSelector,useDispatch } from 'react-redux'
import { increament,decreament,increaseByNumber, Login } from
'./actions/allActions'

const App = () => {
// counter is defined as an object in the combinder.js
//It is a key that gives access to the counterReducer functionality
// The useSelector is calling the state defined here <Provider store={myStore}>
//In actual sense, the state goes into the store=> through the myStore =>
combinder => counter

const counter = useSelector(state=>state.counter)
const auth = useSelector(state=>state.isLogged)
const dispatch = useDispatch()

return (
<div>
<h2>Counter {counter} </h2>

<button onClick={()=>{
dispatch(increament())
}}>+</button>

<button onClick={()=>{
dispatch(decreament())
}}>-</button>

<button onClick={()=>{
dispatch(increaseByNumber(5))
}}>+5</button>

<button onClick={()=>{
dispatch(increaseByNumber(99))
}}>+99</button>
{
auth?(
<>
<h3>Secret Info I Should not see</h3>
</>):(<>
<h3>Login first to see</h3>
</>)
}
<button onClick={()=>{
dispatch(Login())
}}>Login</button>
</div>

)
}

export default App
```