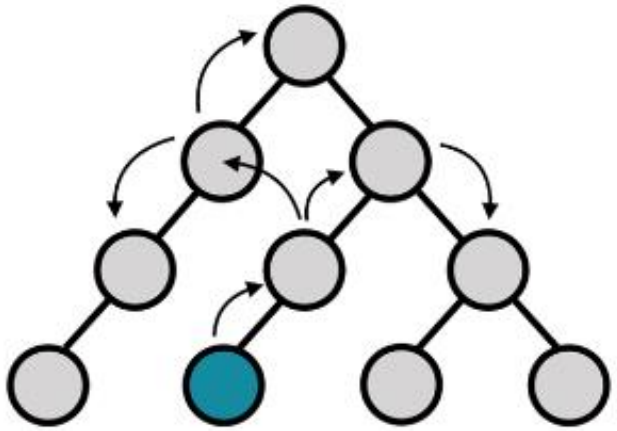
Redux

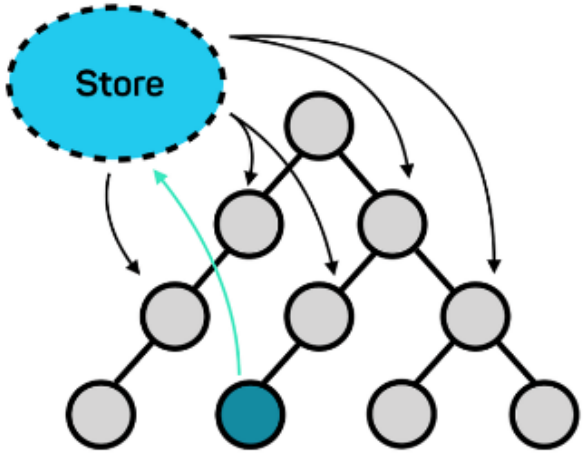
How React Manages the State

- Using **Props** Drilling
- Using Hooks
  - ✓ useState
  - ✓ useReducer
  - ✓ useContext
- Redux

Without Redux



With Redux



Component initiating change

Redux

A JS library for predictable and maintainable global state management

Get Started

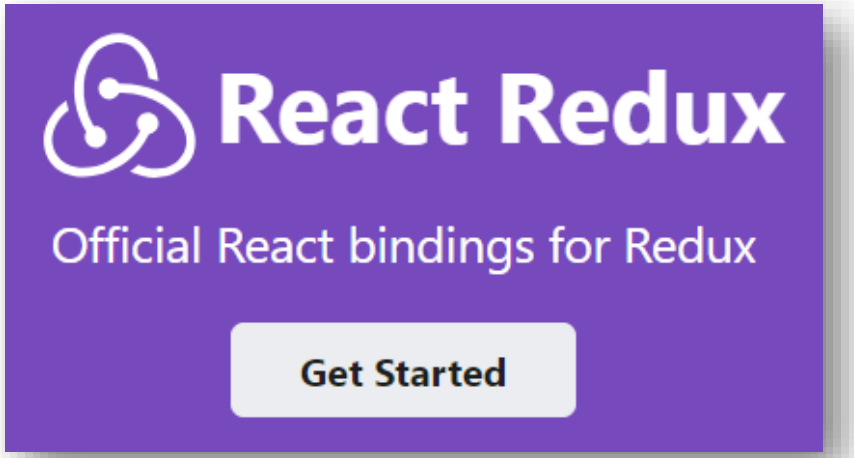
Other Libraries from the Redux Team

React-Redux

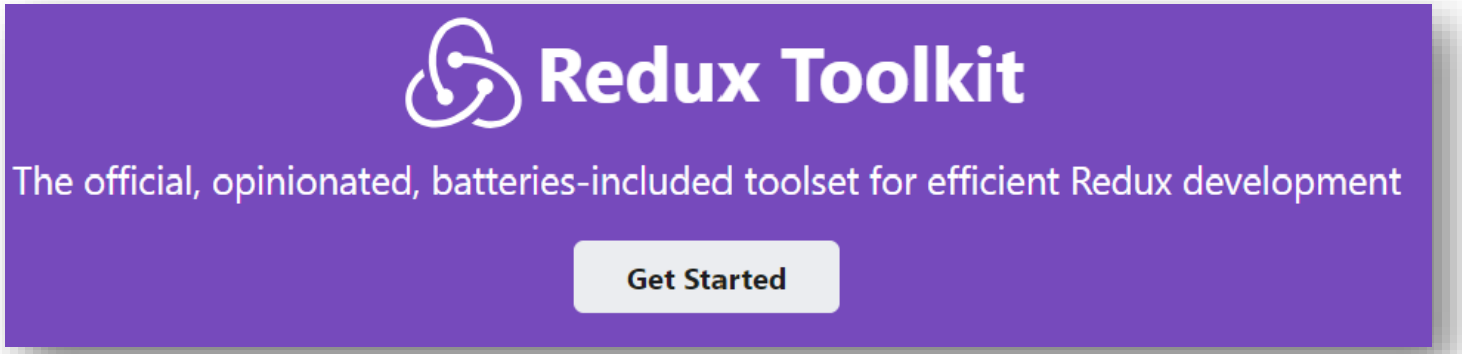
Official React bindings for Redux

Redux Toolkit

The official, opinionated, batteries-included toolset for efficient Redux development



- ✓ Simple
  - Store setup, creating reducers
- ✓ Opinionated
  - Includes most commonly used Redux built-in functions
- ✓ Powerful
  - ✓ Create entire "slices" of state automatically
- ✓ Effective
  - ✓ Create entire "slices" of state automatically



Why should we use Redux Toolkit

Redux Toolkit

Getting Started Tutorials Usage Guide

Introduction
- Getting Started
- Why Redux Toolkit is How To Use Redux Today

Tutorials
- Tutorials Overview
- Quick Start
- TypeScript Quick Start
- RTK Query Quick Start

Using Redux Toolkit
- Migrations

Introduction > Getting Started

Getting Started with Redux Toolkit

Purpose

The **Redux Toolkit** package is intended to be the standard way to write Redux logic. It was originally created to help address three common concerns about Redux:

- "Configuring a Redux store is too complicated"
- "I have to add a lot of packages to get Redux to do anything useful"
- "Redux requires too much boilerplate code"

What is Redux

- **Redux** is a **State Management Library** that allows us to Manage the State of any **JavaScript Application** more Efficiently & Predictably.

- *Assume you are constructing a house* and *keep track of all materials you use & how much money you spend*… Instead of keep track of everything in your head… You could use a ledger to track of all the transaction.

- Redux works similarly by keep track of our application's state in a single place called the **"store."**

Example

- Assume we are building an e-commerce application. You may need to keep track of the items in a user's cart, their payment information, and their shipping details. Instead of passing this information from component to component using props, Redux allows you to store them in one central location where they can be easily accessed and updated. This helps us to manage complex states and keep our application organized.

- **Important Note:** Redux is not only used in React, but we can use Redux with any frameworks including Vanilla JavaScript.

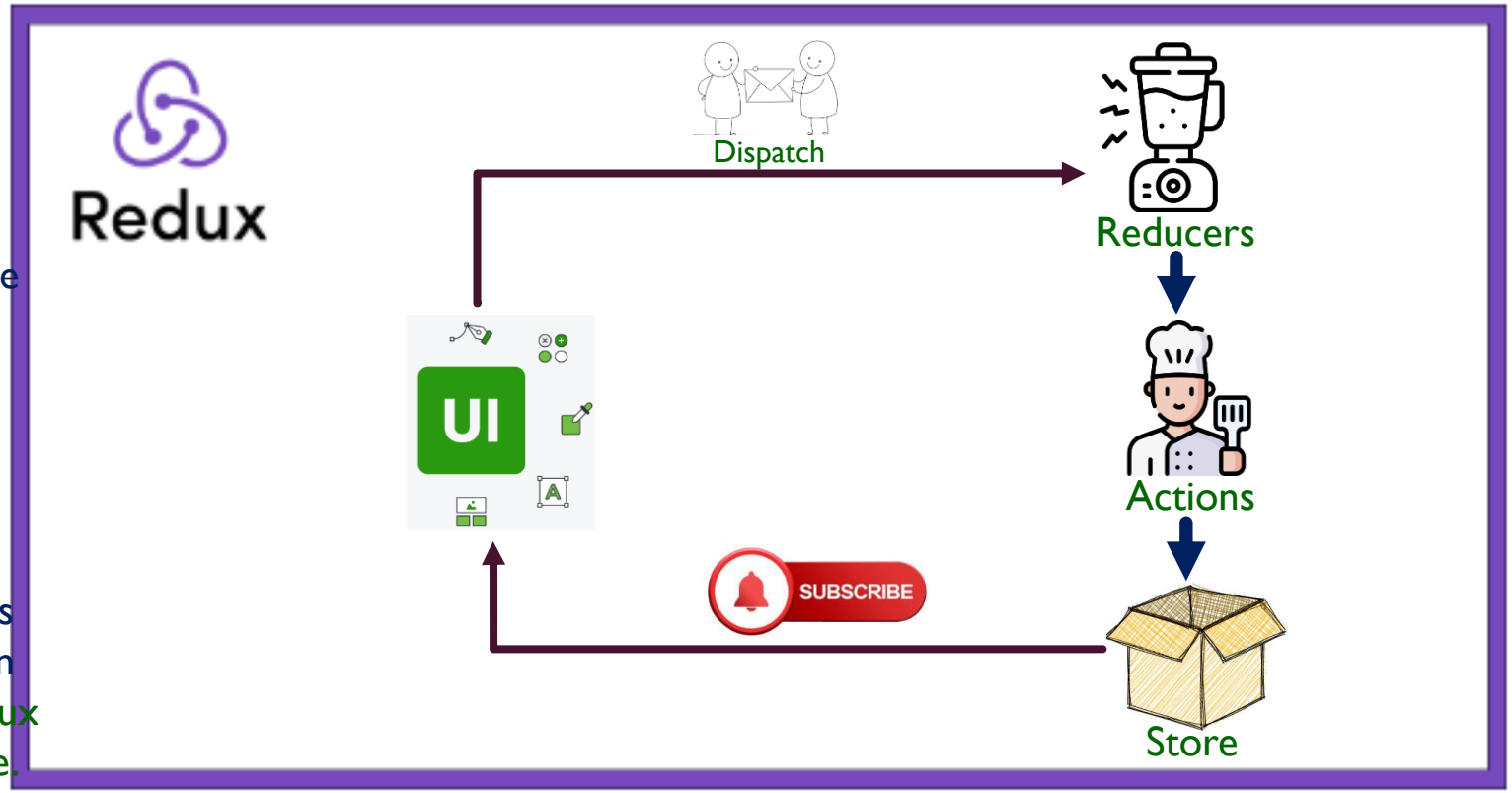
Why Should we Use Redux?

- Centralized state management
  - ✓ We can maintain the state of our application in a single store, so that it will be easy to manage and access data across all components.
- Predictable state updates
  - ✓ Redux has a clear flow of data, which means changes to the state can only happen when you create an action and send it through Redux.
- Better performance
  - ✓ By minimizing the number of state updates and reducing the need for prop drilling, Redux helps improve your application's performance.

How Does Redux Work

Key components which enables centralized State Management

- ✓ Store
- ✓ Action
- ✓ Dispatch
- ✓ Reducers



Store

- ✓ Redux store is like a container that holds all the data of different type(strings, numbers, arrays, objects, functions) of our application.
- ✓ Any component in our application can Add, Retrieve, update data present in store.

Action

- ✓ Action is an object that describes what changes need to be made to the state of our application
- ✓ Action sends data from our application to Redux store and it is the only way to update the store.
- ✓ Action will have the type(describes action), payload(represents data)

Dispatch

- ✓ Dispatch is a function provided by the store that allows us to send action to update the state

Reducers

- ✓ reducer is a function that takes current state of an application and an action as arguments, and returns a new state based on the action.

Installing Redux

- Create a folder called **redux-app**
- From workspace folder go to redux-app folder using below command

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

PS C:\React_7AM_WS> cd redux-app
PS C:\React_7AM_WS\redux-app>
```

- Create React Application using vite command

```
PS C:\React_7AM_WS\redux-app> npm create vite@latest
Need to install the following packages:
  create-vite@5.3.0
Ok to proceed? (y) y
√ Project name: ... react-redux-app
√ Select a framework: » React
√ Select a variant: » JavaScript

Scaffolding project in C:\React_7AM_WS\redux-app\react-redux-app...

Done. Now run:

  cd react-redux-app
  npm install
  npm run dev

PS C:\React_7AM_WS\redux-app>
```

Installing Redux …

- Install React using **npm install** command

```
Done. Now run:

  cd react-redux-app
  npm install
  npm run dev

PS C:\React_7AM_WS\redux-app> cd react-redux-app
PS C:\React_7AM_WS\redux-app\react-redux-app> npm install
npm WARN deprecated inflight@1.0.6: This module is not supported, and leaks memory. Do not use
o coalesce async requests by a key value, which is much more comprehensive and powerful.
npm WARN deprecated @humanwhocodes/config-array@0.11.14: Use @eslint/config-array instead
npm WARN deprecated rimraf@3.0.2: Rimraf versions prior to v4 are no longer supported
npm WARN deprecated glob@7.2.3: Glob versions prior to v9 are no longer supported
npm WARN deprecated @humanwhocodes/object-schema@2.0.3: Use @eslint/object-schema instead

added 278 packages, and audited 279 packages in 18s

103 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
PS C:\React_7AM_WS\redux-app\react-redux-app>
```

Installing Redux …

- Install Redux Toolkit, react-redux Using following commands

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS                powe

PS C:\React_7AM_WS\redux-app\react-redux-app> npm install @reduxjs/toolkit

added 5 packages, and audited 284 packages in 2s

104 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
PS C:\React_7AM_WS\redux-app\react-redux-app>
```

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

PS C:\React_7AM_WS\redux-app\react-redux-app> npm install react-redux

added 3 packages, and audited 287 packages in 1s

104 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
PS C:\React_7AM_WS\redux-app\react-redux-app>
```

Start React Server

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

PS C:\React_7AM_WS\redux-app\react-redux-app> npm run dev

> react-redux-app@0.0.0 dev
> vite


  VITE v5.3.2  ready in 798 ms

  ➜  Local:   http://localhost:5173/
  ➜  Network: use --host to expose
  ➜  press h + enter to show help
```

Clean up files

- ✓ Src/App.jsx
- ✓ Src/index.css
- ✓ main.jsx

App.jsx index.css

redux-app > react-redux-app > src > App.jsx > App

```jsx
function App() {


  return (
    <>
      <div>
        <h1>Welcome to React Redux!!!</h1>
      </div>
    </>
  )
}

export default App
```

index.css

redux-app > react-redux-app > src > index.css

```
1
```

Vite + React

localhost:5173

Welcome to React Redux!!!

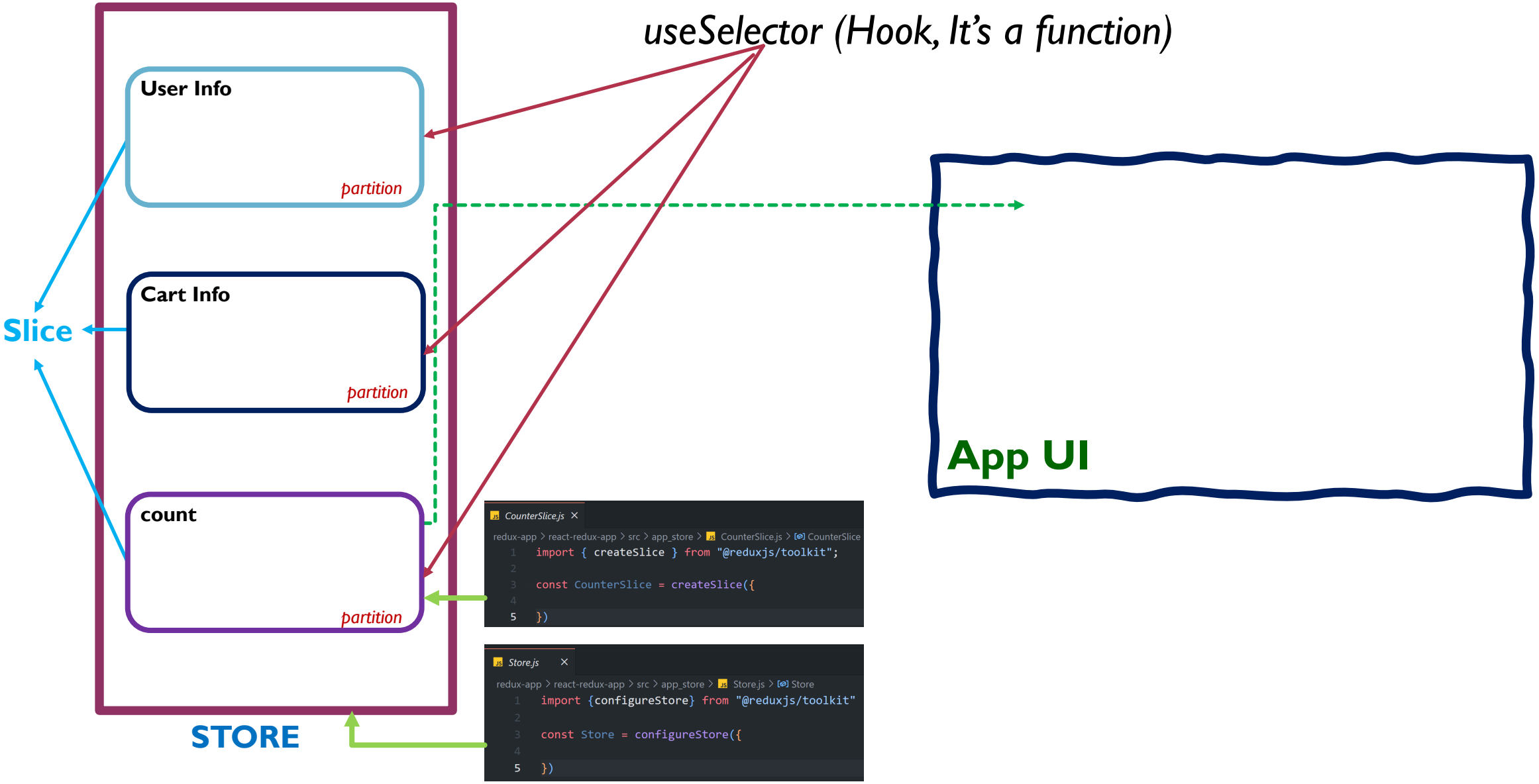
main.jsx

redux-app > react-redux-app > src > main.jsx

```jsx
import React from 'react'
import ReactDOM from 'react-dom/client'
import App from './App.jsx'


ReactDOM.createRoot(document.getElementById('root')).render(
  <React.StrictMode>
    <App />
  </React.StrictMode>,
)
```

Architecture of Redux toolkit



Create STORE, Slice, Changes in main.jsx

- ✓ Create a folder called "app_store" in src folder
- ✓ Create a file called store.js inside app_store folder & add logic for configure store
- ✓ Create a file called CounterSlice.js inside app_store folder & add logic for Create a Slice
- ✓ Changes in main.jsx… Keep <App /> component inside <Provider store={store}> component and pass store as a props to Provider component

Store.js

redux-app > react-redux-app > src > app_store > Store.js > Store

```
1  import {configureStore} from "@reduxjs/toolkit"
2
3  const Store = configureStore({
4
5  })
```

CounterSlice.js

redux-app > react-redux-app > src > app_store > CounterSlice.js > CounterSlice

```
1  import { createSlice } from "@reduxjs/toolkit";
2
3  const CounterSlice = createSlice({
4
5  })
```

main.jsx

redux-app > react-redux-app > src > main.jsx

```
1  import React from 'react'
2  import ReactDOM from 'react-dom/client'
3  import App from './App.jsx'
4  import {Provider} from "react-redux"
5  import store from './app_store/store.js'
6
7  ReactDOM.createRoot(document.getElementById('root')).render(
8    <Provider store={store}>
9      <App />
10   </Provider>,
11 )
```

Including Slice inside Store

1. Create a Slice
2. Include Slice in Store

Creating a Slice

- Creating a Slice is Four Steps process.
  - ✓ name
  - ✓ initialState
  - ✓ reducers
  - ✓ Export slice with reducer

CounterSlice.js

redux-app > react-redux-app > src > app_store > CounterSlice.js > CounterSlice > name

```
1  import { createSlice } from "@reduxjs/toolkit";
2
3  const CounterSlice = createSlice({
4
5      name: "counter", //Slice Name
6
7      initialState: { //Initial State of Slice
8          value: 0
9      },
10
11     reducers: {
12
13     }
14 })
15
16 export default CounterSlice.reducer
```

Including Slice inside Store

- Including Slice in Store is a Three Step process.
  - ✓ Create a reducer object inside configureStore object
  - ✓ If required include devTools: as true
  - ✓ Export store

store.js

redux-app > react-redux-app > src > app_store > store.js > default

```
1   import {configureStore} from "@reduxjs/toolkit"
2   import CounterSlice from "./CounterSlice"
3
4   const store = configureStore({
5       reducer: {
6           counter: CounterSlice,
7           userinfo: CounterSlice,
8           cartinfo: CounterSlice,
9           productdetails: CounterSlice,
10          locationinfo: CounterSlice,
11      },
12
13      devTools: true
14  })
15
16  export default store
```

Adding Redux DevTools In Chrome

Google

redux devtools

All Images Videos Shopping News Books Maps More Tools

Download Firefox Extension npm Install React Native Not working Edge Safari

Chrome Web Store
https://chromewebstore.google.com › redux-devtools

Redux DevTools - Chrome Web Store

Redux DevTools for debugging application's state changes. The extension provides power-ups for your Redux development workflow. Apart from Redux, it can be ...

store.js

redux-app > react-redux-app > src > app_store > store.js > store

```
1   import {configureStore} from "@reduxjs/toolkit"
2   import CounterSlice from "./CounterSlice"
3
4   const store = configureStore({
5       reducer: {
6           counter: CounterSlice
7       },
8
9       devTools: true
10  })
11
12  export default store
```

Redux DevTools

4.6 ★ (710 ratings)

Add to Chrome

Extension Developer Tools 1,000,000 users

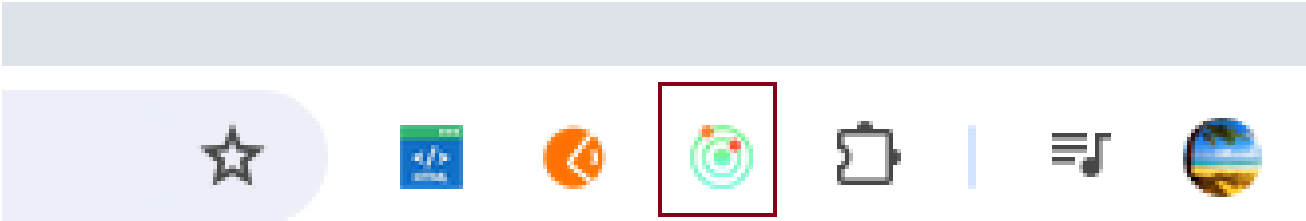
Add "Redux DevTools"?

It can:

Read and change all your data on all websites

Display notifications

Add extension Cancel



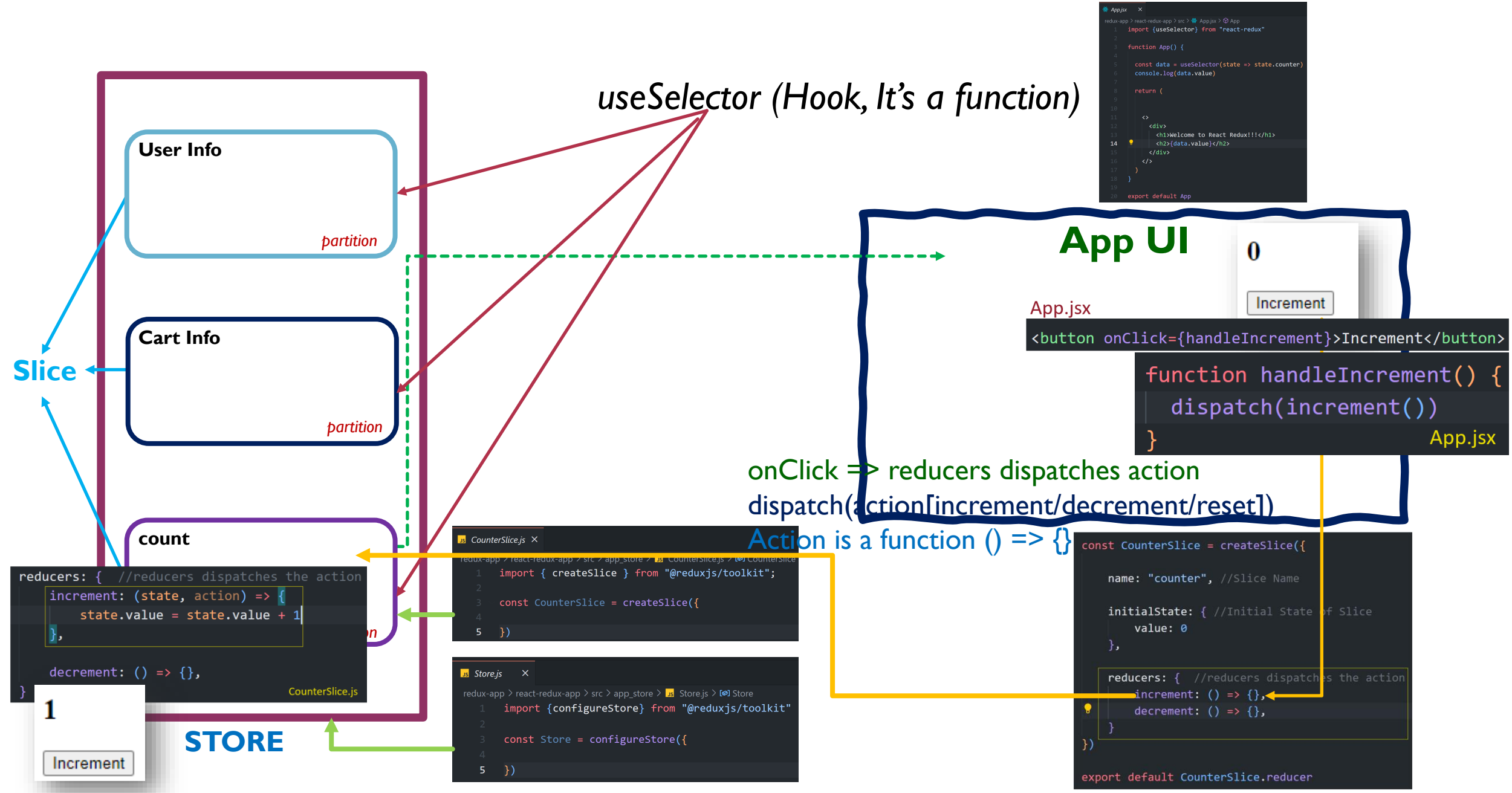
Using State in React Component

App.jsx

redux-app > react-redux-app > src > App.jsx > App

```
1   import {useSelector} from "react-redux"
2
3   function App() {
4
5     const data = useSelector(state => state.counter)
6     console.log(data.value)
7
8     return (
9
10
11      <>
12        <div>
13          <h1>Welcome to React Redux!!!</h1>
14          <h2>{data.value}</h2>
15        </div>
16      </>
17    )
18  }
19
20  export default App
```

Architecture of Redux toolkit …



UI Tring to update the Value in STORE

App.jsx

redux-app > react-redux-app > src > App.jsx > App

```
import {useDispatch, useSelector} from "react-redux"
import { increment } from "./app_store/CounterSlice"

function App() {

  const data = useSelector(state => state.counter)
  console.log(data.value)

  const dispatch = useDispatch()

  function handleIncrement() {
    //console.log("HandleIncrement got Called!!!")
    //dispatch(increment("Md Ali")) //"Md Ali" = Payload
    dispatch(increment())
  }

  return (


    <>
      <div>
        <h1>Welcome to React Redux!!!</h1>
        <h2>{data.value}</h2>

        <button onClick={handleIncrement}>Increment</button>
      </div>
    </>
  )
}

export default App
```

4 3 2 1

CounterSlice.js

redux-app > react-redux-app > src > app_store > CounterSlice.js > CounterSlice > reducers

```
import { createSlice } from "@reduxjs/toolkit";

const CounterSlice = createSlice({

    name: "counter", //Slice Name

    initialState: { //Initial State of Slice
        value: 0
    },

    reducers: {  //reducers dispatches the action
        increment: (state, action) => {
            //console.log(action.payload)
            state.value = state.value + 1
        },

        decrement: () => {},
    }
})

export const {increment, decrement}  = CounterSlice.actions

export default CounterSlice.reducer
```

1 2

store.js

redux-app > react-redux-app > src > app_store > store.js > default

```
import {configureStore} from "@reduxjs/toolkit"
import CounterSlice from "./CounterSlice"

const store = configureStore({
    reducer: {
        counter: CounterSlice,
        //userinfo: CounterSlice,
        //cartinfo: CounterSlice,
        //productdetails: CounterSlice,
    },

    devTools: true
})

export default store
```

Decrement

Reset