Contents

[Helloworld in react 2](#_Toc156169034)

[Create tags and structure 2](#_Toc156169035)

[HTML order 4](#_Toc156169036)

[Createe APP 4](#_Toc156169037)

[Bundler 5](#_Toc156169038)

[Ignit the app 5](#_Toc156169039)

[Browserslist 7](#_Toc156169040)

[Basic Fundamentals 7](#_Toc156169041)

[React Element 8](#_Toc156169042)

[React component 9](#_Toc156169043)

[React functioanal component 9](#_Toc156169044)

[Cross site scripting 10](#_Toc156169045)

[Props 10](#_Toc156169046)

[Java script properties like Join 11](#_Toc156169047)

[Export and import 11](#_Toc156169048)

[React Hooks 11](#_Toc156169049)

[Reconciliation algorithm 12](#_Toc156169050)

[Shimmer UI 13](#_Toc156169051)

COrsproxy.io

CDN is the place where react is hosted and we can get from below link

<https://legacy.reactjs.org/docs/cdn-links.html>

Copy below CDN links and add to your code.

<script crossorigin src="https://unpkg.com/react@18/umd/react.development.js"></script>

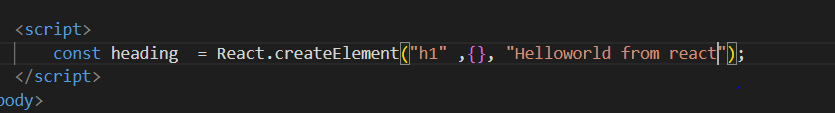
<script crossorigin src="https://unpkg.com/react-dom@18/umd/react-dom.development.js"></script>

Just by adding the above links browser can understand the react.

react.development.js is core react

react-dom.development.js is to modify the dom elements - it is like the bridge between the browser and react.

# Helloworld in react



We use React.createelemnt() it takes three inputs

1 element type

2 body – is used to provide the attributes like id or class

3 data that to put in the element

const heading  = React.createElement("h1" ,

{id : "Heading" , xyz : "abc"}, "Helloworld from react from app");

All abovevalues are stored in props. Data is put into children and attributes to attributes

We can provide attributes as shown above and alos we can name our own attribute names like xyz.

Now for the whole JS code to work e need a root where all the react code can work.

So create a root. And all the app code will be residing in this root block.

const root = ReactDOM.createRoot(document.getElementById("root"));

        root.render(heading)

The most costly operation is DOM manipulation.

In React every thing will be Javascript, not more Html.

If any thing is already available in the root then everything will be replaced with the content we are rendering in that root.

# Create tags and structure

How to create structure like below using react

/\*\*

 \* <div id = "parent">

 \*      <div id = "child">

 \*          <h1> I am h1 tag</h1>

 \*      </div>

 \* </div>

 \*/

Create as below.

 const parent = React.createElement("div" , {id : "parent"} ,

    React.createElement("div" , {id : "child"},

    React.createElement("h1" ,  {}, "I am h1 tag")

    ))

 console.log(parent);

    root.render(parent);

Every thing we write in react. And while rendering the elements it will convert everything into html and js and will render in UI

To create below structure like simlinks then we can add children to list as shown below.

/\*\*

 \* <div id = "parent">

 \*      <div id = "child">

 \*          <h1> I am h1 tag</h1>

 \*          <h2> I am h1 tag</h2>

 \*      </div>

 \* </div>

 \*/

const parent1 = React.createElement("div" , {id : "parent1"} ,

    React.createElement("div" , {id : "child1"},

    [React.createElement("h1" ,  {}, "I am h1 tag") ,

    React.createElement("h2" ,  {}, "I am h2 tag")]

    ))

    console.log(parent1);

    root.render(parent1);

To make below structure use this

/\*\*

 \* <div id = "parent2">

 \*      <div id = "child2">

 \*          <h1> I am h1 tag</h1>

 \*          <h2> I am h1 tag</h2>

 \*      </div>

 \*      <div id = "child2a">

 \*          <h1> I am h1 tag</h1>

 \*          <h2> I am h1 tag</h2>

 \*      </div>

 \* </div>

 \*/

const parent2 = React.createElement("div" , {id : "parent1"} ,

    [React.createElement("div" , {id : "child1"},

        [React.createElement("h1" ,  {}, "I am h1 tag") ,

        React.createElement("h2" ,  {}, "I am h2 tag")]

        ) ,

    React.createElement("div" , {id : "child2"},

        [React.createElement("h1" ,  {}, "I am h1 tag") ,

        React.createElement("h2" ,  {}, "I am h2 tag")]

        )

    ]

);

console.log(parent2);

root.render(parent2);

But in above cases as structure increases code complexity will increase. And it will create a mess.

So to create proper code we need JSX which makes life easy for making tags.

The above code is the actual react code.

# HTML order

All the react files should be imported first and then include the app.js. because app.js is using the react.

# Createe APP

Using normal code we need to do lot of things like optimize the code, image processing.

When we create a new create app it will create production ready app.

NPM – manages packages

It is a repository for lot of the libraries or utilities.

Create a folder and do npm init and provide the details. Then it will generate the package.json file.

**Package.json** is the configuration file for npm. All the dependencies need to add in this file.

## Bundler

Whole code need to be clean and bundle it or packages it and ships to production.

parcel is the one of the bundler

Npm install -D parcel

There will be 2 dependencies one is for developing and it is called as dev dependency and other is normal and it is used for prod.

In package.json if we have added ^ then it will upgrade minor upgrade and ~ will upgrade major updates automatically.

**Package-lock.json**

this file will keep the exact version upgraded in app while package.json will have close version.

Transitive dependencies – a package have dependencies and that dependencies might have some other dependencies llike goes on and this is called as transitive dependencies.

## Ignit the app

Npx parcel index.html

Parcel will create server and will start the app.

Npx means executing the package, npm means to get the specific package.

Onee way to get the react into code is add cdn links in the html startpage. If any new version of react came then every time we need to change the CDN url. And it is very tough to change every time

But instead of using CDN we can use the react package.

Npm install react

npm install react-dom

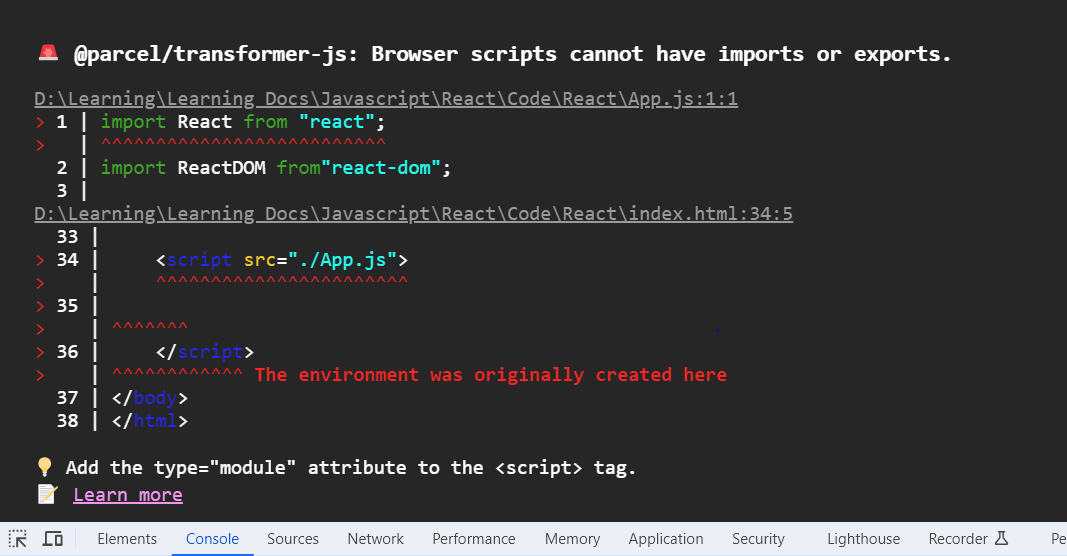
above both packages need to be installed into the app to use react.

Now e have removed the CDN links and nothing is defiend in the code.

Now to use the react I our app e need to import react

import React from "react";

import ReactDOM from"react-dom";



 <script src="./App.js">

Above code is a normal import and this normalimport means browser script it shpuld not contain the import and export.

SO to use import and export we need to create module.

<script type="module" src="./App.js">

After starting the app when ever we add new content and save immediately it will update in the UI. And this will be done by parcel.

Below will be done by parcel

Devbuild

Localserver

HMR = hot module replacement

Parcel wwill use file watching algorithm which is written in C++

Parcel also uses cache and using this it will build fast.

Image optimization

Minification of files

Bundling

Compress

Consistent hashing

Code spilliting

Differential bundling - to support older versions of browwsers , parcel will support by doing differentia building.

Diagnostic

Error handling

Https

Tree shaking - remove unused code

Different prod and dev build

Npm parcel build index.html.

Wwhen above command is run then it will build code and will bundle andwill put in dist folder.

## Browserslist

It is the configuration it indicates in which browsers does my app should support.

Got to browserslist.dev website to check what to add in the browserslist.

 "browserslist" :[

    "last 2 chrome version",

    "last 2 firefox version"

  ]

This above code need to add in the package.json file.

# Basic Fundamentals

In package.json create te scripts as below.

 "scripts": {

    "start" : "parcel index.html",

    "build" : "parcel build index.html",

    "test": "jest"

  },

Npm run start will run the script.

## React Element

React.createElement

Above statement will create object and when this element is rendered to DOM then it will become html.

const heading  = React.createElement("h1" ,

{id : "heading" , xyz : "abc"}, "Helloworld from react from app");

const root = ReactDOM.createRoot(document.getElementById("root"));

root.render(heading)

above logivc is used to create the tags. But writing this tags are not easy to add lot of tags it will be clumsy.

To write easy code jsx need to be used. Jsx is not part of react. We can develop react wwith out jsx. But jsx makes life easy.

Using JSX we can create same tag as below

const jsxHeading =  <h1>This is Jsx heading</h1>

it means jxs is the html like syntax. But jsx is not html.

The jsxheading is same as heading. What we create in heading same object will be created for jsxheading.

Only difference is jsxheading is easy and human friendly.

Jsengine will understand ecmasscript.

So the above code jsxheading will not understand by jsengine or browser. Then how does jsxheading Is working and displaying inUI.

The above wwork is done by parcel. The above code is transpiled(converted) by the parcel before it reaches the browser or jsengine.

In the parcel Babel is the package used for transpiled. So babel is compiler with takes jsx and converts in to the react.

Finally jsx code is converted to react and react to html.

Below flow is for normal react



Below code is for jsx



In html to provide class we give direct class, but in jsx we provide as className and camelCase will be used.

const jsxHeading =  <h1 id ="JsxId" className = "jsxclass">This is Jsx heading</h1>

how to write multiple lines of jsx. Use brackets as below

const jsxHeading1 =  (<h1 id ="JsxId" className = "jsxclass">

    This is Jsx heading

</h1>

)

## React component

Every thing in react is component.

There are two types of components

1. Class based – old way of writing
2. Functional based - new way of writing

### React functioanal component

These are normal javascript function whch returns the jsx .

//react functional component

const  Headingcomponent = () =>{

    return <h1 id ="JsxId" className = "jsxclass">This is functioanal component</h1>;

}

How to render the component.

Render component using brackets.

root.render(<Headingcomponent/>)

render one component into other component as below.

const  Title = () => <h1 id ="JsxId" className = "jsxclass">This is title component</h1>;

//for multiple

const  Headingcomponent2 = () =>{

    return ( <div>

             <Title/>

                <h1 id ="JsxId" className = "jsxclass">This is heading component</h1>

              </div>);

}

root.render(<Headingcomponent2/>)

When a flower bracket is used in component we can add any javascript code inside that as below.

const number = 100;

const  Headingcomponent3 = () =>{

    return ( <div>

                {number}

             <Title/>

                <h1 id ="JsxId" className = "jsxclass">This is heading component</h1>

              </div>);

}

When we are using {} then this will sanitise the data and will process. So that cross site scripting can be restricted.

# Cross site scripting

If we are calling any api and that api response is giving some commands and when we try to render the response in UI those commands will be executed and some info like cookies might be stolen. This is called as cross site scripting.

# Props

Props is the properties that we need to pass to component.

Props are like normal arguments to a function.

**Config driven UI**

based on the configurations data will be displayed. That configurations will come from backend as a Json.

# Java script properties like Join

What ever we are doing in the react can be done by other technologies. But why to use react. Because it will give better user experiencewith less code.

# Export and import

There are two ways to export

1. Export default component name

Import component from “path”

1. Export const component

Import {component} from “path”

# React Hooks

React hook is a normal javascript function given by react and it comes with some powers are also called as utility functions written by facebook and available in react.

To use this hooks need to import.

1.useState() hook

2. useEffect()

useState is used to create state variables.

**How to create state variables**

let [lisOfRestaurants] = useState([]);

 let [lisOfRestaurants] = useState(restaurants);

using above code state variable will be created. The default value will be passed as parameter to useState().

How to modify

 let [lisOfRestaurants, setLisOfRestaurants] = useState(restaurants);

we need to pass the functionname while declaring, in above case setListOfRestaurants is the method we are passing. So when we need to modify or update the ListOfRestaurants we need to call the set setListOfRestaurants and pass the updated value as below

   const filteredList = restaurants.filter(res =>

                        res.info.avgRating > 4.5

                    );

                    setLisOfRestaurants(filteredList);

So when ever a statevariable is changed then react will automatically rerender the component.

Never create usestate hook outside the component.

Also never create usestates in if and else conditions.

UseEffect

useEffect(() =>{} , []);

use effect will be used as above. It will take two parameters as input.

1. Callback function
2. Dependency Array

The callbackfunction will called when the component is lloaded. First it will render the component and then call the callback function.

After rendering the component then useeffect will call.

If dependency array is not defined then useeffect will be called when ever component is rendered.

If dependeny arry is empty then useeffect will call only initial render of the component (just one at first when component loaded)

If there is any value in the dependenyarray then when ever the value in the dependency array changes then useeffect will triggered.

For cors add allow cors extension in chrome.

# Reconciliation algorithm

Is also known as react fiber.

React will create virtual dom. Virtual dom is not an actual dom it is an representation of actualdom

Dom is like

<div>

<div>

<div>

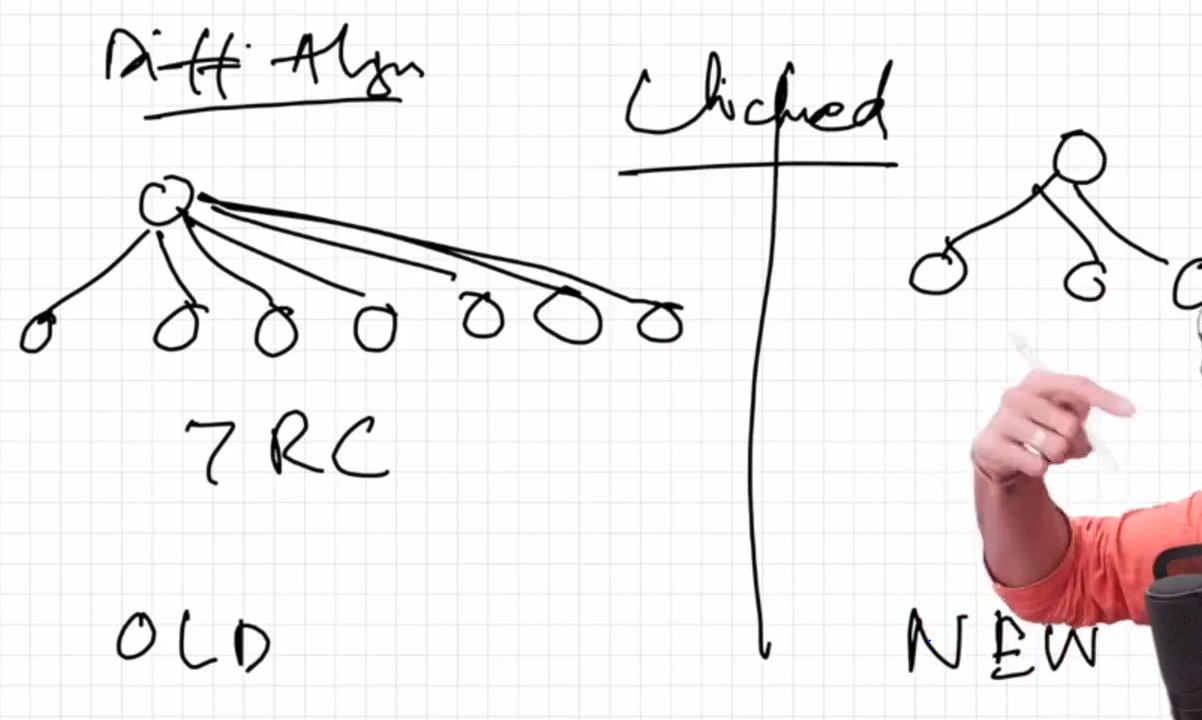
</div>

</div>

</div>

Virtual dom is like norml javascriptobject. Actual dom contains the all the object information. But virtual dom contains the info of specific component .

**Diffalgorithm**



React 16 came new algorithm called as reactfiber.

It will findthe diff between old virtualdom and new virtualdom and then updates the actual dom.

So it find the diff between two objects(virtualdom is object representation) and then updates the actual dom. So that react is fast. As react will not touch the actual dom .

React is doing efficient dom manupilations because of virtual dom.

# Shimmer UI

Instead of spinner we can load skeleton UI.

Once data is back then load the data.

**Conditional rendering** is rendering based on conditions.

When ever search text is updated react will rerender the component every time.

When ever starte variable changes react will reconcile or rerender the component.

# Router

Npm I react-router-dom

We have different routers.

Types of routing

Client side routing

Server side routing

State variable- when ever data updates then UI updates automatically.

Graphql

Using map always use id

# Class based componets

Function component is normal javascript function.

Class component is normal javascript class

class UserClass extends React.Component

to create classbased component use extends

contains render method

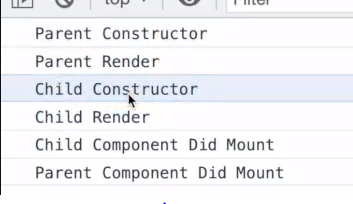
why to write super(props) in the constructor of class.

State variable is created when constructr is invoked in class components.

Life cycle:

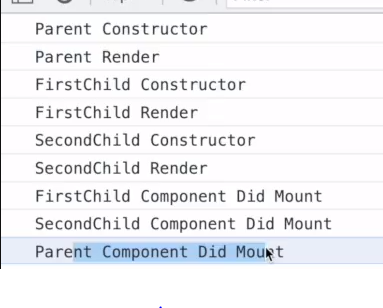
First when instantiated constructor is called and next render is called.

Once class is mounted then Componentdidmount is called

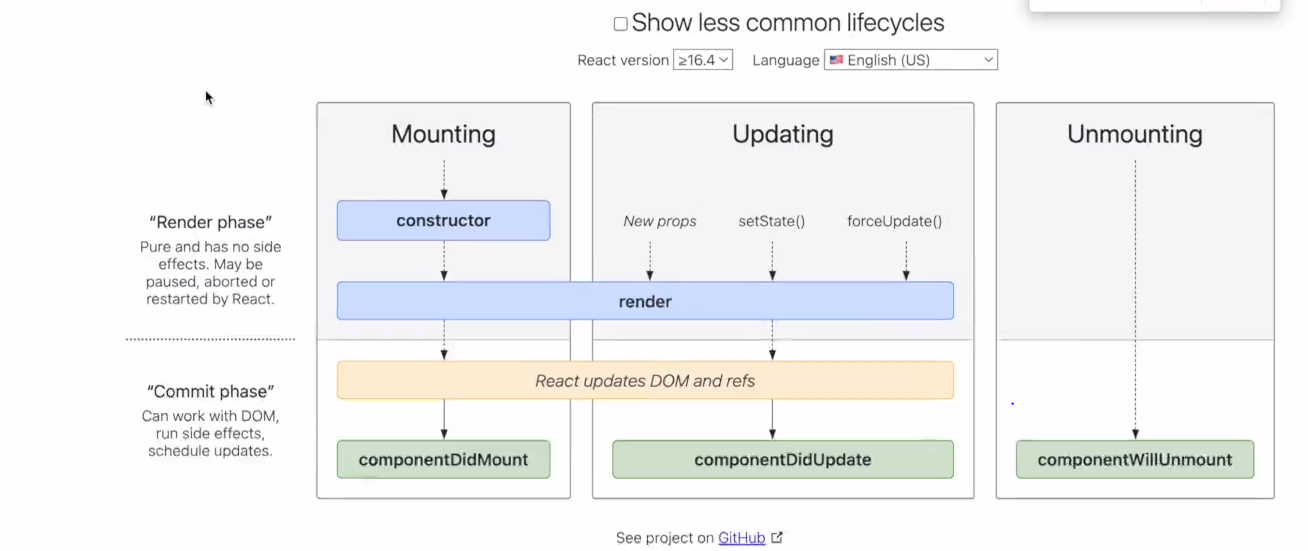


Componentdidmount()

TO make api calls we use componentdidmount. After calling apicalls then do rerender.



React lifecycle



As dom update is the costlist all the child components did munt will be collected and executed at once. It means dom wll be updated at once.

Componentdidupdate – this is will be called once update is completed.

Constructor

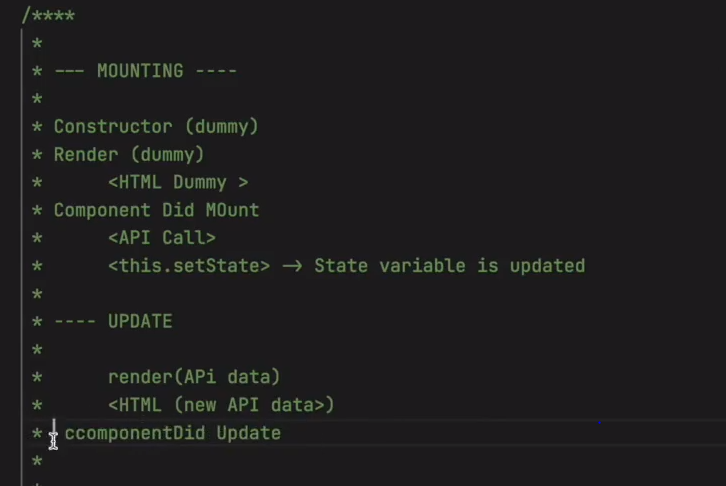
Render – UI is loaded with default data

Componentdidmount – did apicalls and update the state variables

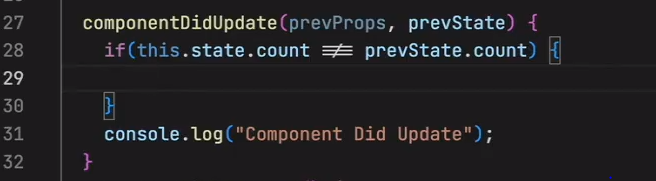
When set state is called then update process is called and render will call again with new data

HTML is loaded wwith new data.

Component did update is called.



Componentwillunmount – when component is disappeared from UI thien this wil be called.



# Optimizing the app

Custom hooks-

Hooks are like utility functions

https://react.dev/warnings/invalid-hook-call-warning

Single responsibility

Chunking

Code splitting

Dynmic bundling

Lazyloading

On demand loading

One bundle should have one functioanliy full

When we click on thefunctioanality it ill download the bundle.

# SCSS and SASS

Advanced of css.

# Styledcomponents

# TailwindCSS

PostCss

https://tailwindcss.com/docs/guides/parcel

npm install -D tailwindcss postcss

npx tailwindcss init

s parcel will bundle only css which are used

# Higher order components

It is a function that takes a component and returns a componenet.

Controlled and uncontrolled components

Lifting the stateup

PROPS DRILLING – passing data too lower level childs. Have different wayt to handle props drilling one way is react context.

React context

useContext

cannot use hooks in classbasedcomponents – use <UserContext.Consumer>

for class components

# Redux

Redux is not mandatory. Use redux only when needed.

Redux is the separate library. Redux is not react.

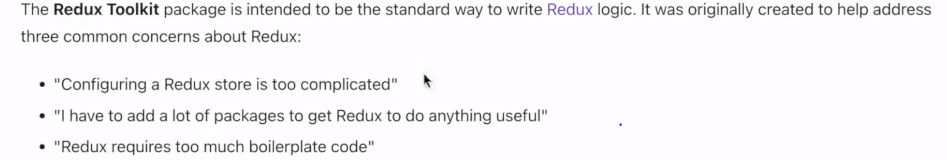
Zustand is also library like redux.

Application become easier to debug using Redux. Using redux developer tools.

There are two llibraries

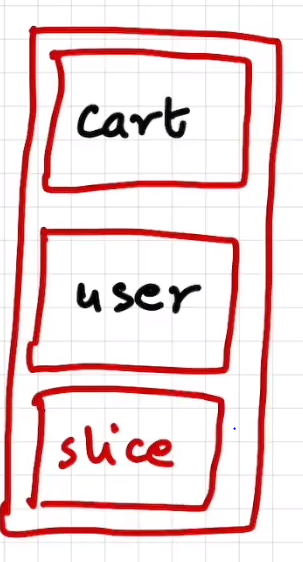
1. React redux
2. Redux toolkit.

Reduxtool is the recent way of using redux. And react redux is the bridge between redux and react.



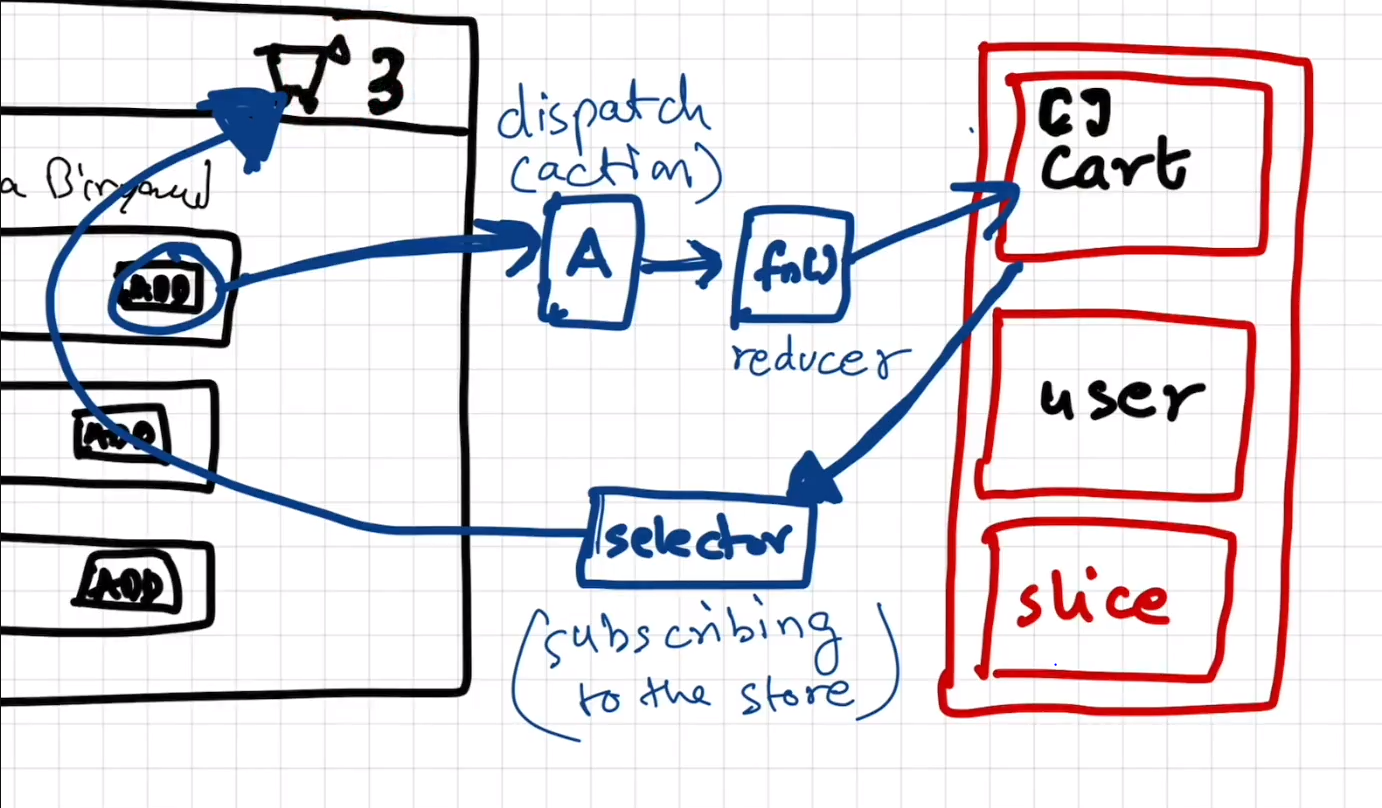
Redux store is a js object and it is placed at global center. Redux store data can be accessed anywhere in the application.

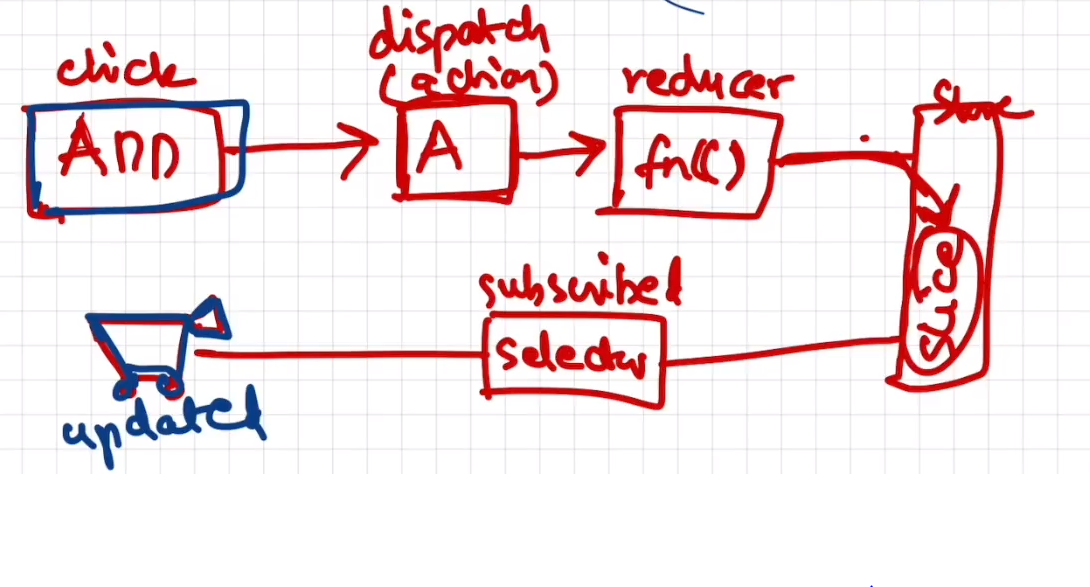
Reduxstore will have slices. For each info we will create slices. For cart one slice for userinfo one slice.



Writing datat to redux - When click on add card then it will dispatches action and it will call a function and function will update the cart. The function is called as reducer.

Reading dat from redux - we use selector tor read the data. And selector wll give the data. And is called as subscribing to the store.





Install rebux

Build store

Connect store to app

Slice(cartslice)

Dispatch

Slector

npm install @reduxjs/toolkit

npm install react-redux

const Appstore = configureStore();

this will return the redux store.

return (

        <Provider store={Appstore}>

        <UserContext.Provider value = {{loggedInUser : userName ,setUserName}}>

        <div className = "app">

            <Header></Header>

            <Outlet></Outlet>

        </div>

        </UserContext.Provider>

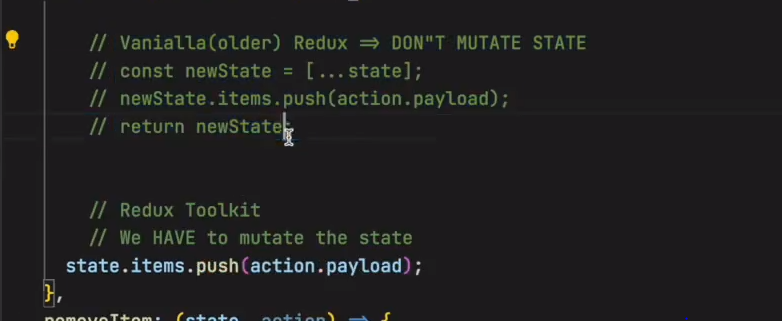
        </Provider>

    )

To use redux store warp the code which requires redux store into provider as shown above.

useSelector –

useDispatch - hook



Immer library.

RDK query

Middleware thungs.

# Testing

Types of testing

1. Unit testing
2. Integration
3. End to end

React resting library.

DOM testing library is the core library(like java). React testing library is like spring.

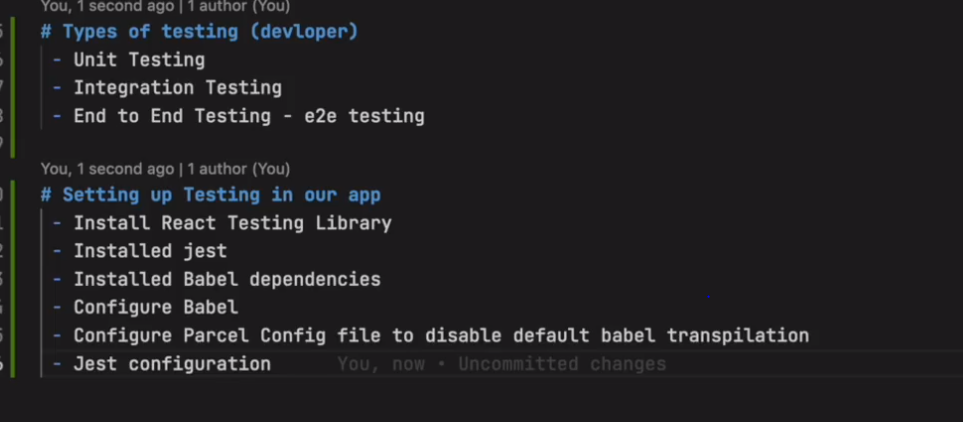
Jest javascript testing framework.

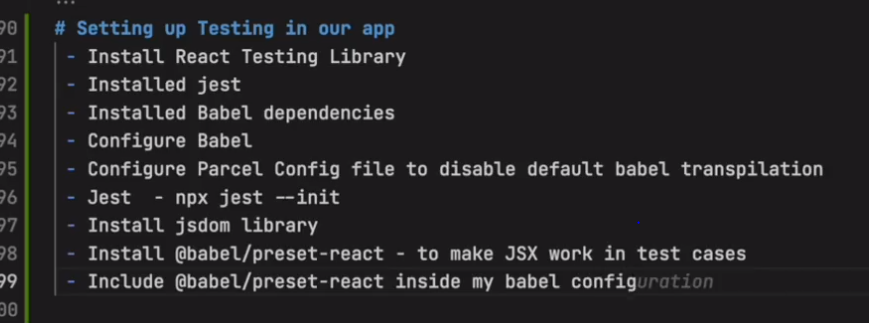
npm install -D @testing-library/react

npm install -D jest

npm install --save-dev babel-jest @babel/core @babel/preset-env

<https://parceljs.org/languages/javascript/>





Configure jest by below command.

npx jest --init

JsDOm is like a browser.

For jest 28 and above then Jsdom need yo be installed separately.

Npm install --save-dev jest-environment-jsdom

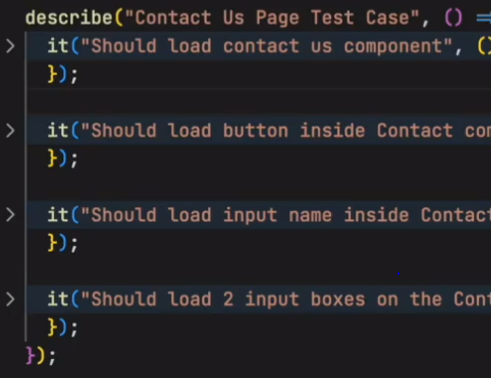
Dunder test

npm i -d @babel/preset-react

npm i -d @testing-library/jest-dom

describe for grouping the testcases

we can use it or test.



Act function