**Context Api**

Global state management

**useContext() + useReducer() = Redux**

**First you know about this topics:**

**1). Basic of Provider and consumer:**

export const Mycontext = React.createContext();

<Mycontext.Provider value={apidata}>

<User/>

</Mycontext.Provider>

<Mycontext.Consumer>

                {({data,myfunc})=>{return (

                    <div>

                        <h1>Name : {data.name}</h1>

                        <h1>Surname : {data.surname}</h1>

                        <h1>Surname : {data.marks}</h1>

                        <button onClick={myfunc}>Click Me</button>

                    </div>

                )}}

</Mycontext.Consumer>

2**). Use of Mycontext**

(When provider is only one):

import React, { Component } from 'react'

import { MyContext } from './Context'

export default class Guest extends Component {

    render() {

        return (

            <div>

                <h1>I am {this.context.data.name}</h1>

                <h1>surname: {this.context.data.surname}</h1>

                <h1>marks: {this.context.data.marks}</h1>

                <button onClick={this.context.myfunc}>Click Me</button>

            </div>

        )

    }

}

Guest.contextType = MyContext;

3). **useContext**

(when provider is more then one):

import React, { useContext } from 'react'

import { Firstname,Lastname } from './Context'

let Guest=()=> {

    const context1 = useContext(Firstname)

    const context2 = useContext(Lastname)

        return (

            <div>

                <h1>I am {context1.data.name}</h1>

                <h1>surname: {context1.data.surname}</h1>

                <h1>marks: {context1.data.marks}</h1>

                <h1>marks222: {context2.data2.marks}</h1>

                <button onClick={context1.myfunc}>Click Me</button>

            </div>

        )

    }

export default Guest

**useReducer**

**[1]. Without Api:**

import './App.css';

import { useReducer } from 'react';

const initialState = {

  "name": "Blank",

  "surname": "click button"

}

const reducer = (state, action) => {

  switch (action) {

    case 'CHANGE\_NAME':

      return {

        "name": "Pradip",

        "surname": "Kachhadiya"

      }

    default:

      return state

  }

}

function App() {

  const [state, dispatch] = useReducer(reducer, initialState)

  return (

    <div className='App'>

      <h1>my name is {state.name}</h1>

      <h3>my surname is {state.surname}</h3>

      <button onClick={() => (dispatch("CHANGE\_NAME"))}>Change name</button>

    </div>

  );

}

export default App;

**[2]. With Api:**

import './App.css';

import { useReducer } from 'react';

const initialState = {

  "name": "Click below button1",

  "name2": "Click below button2",

}

const reducer = (state, action) => {

  switch (action.type) {

    case 'CHANGE\_NAME':

      return {

        ...state,

        "name": action.payload,

      }

    case 'CHANGE\_NAME2':

      return {

        ...state,

        "name2": action.payload,

      }

    default:

      return state

  }

}

function App() {

  const [state, dispatch] = useReducer(reducer, initialState)

  const [state2, dispatch2] = useReducer(reducer, initialState)

  const url = 'https://jsonplaceholder.typicode.com/users'

  const getName = () => {

    return fetch(url)

      .then(res => res.json())

      .then(res2 => {

        (dispatch({ type: 'CHANGE\_NAME', payload: res2[1].name }))

      })

  }

  const getName2 = () => {

    return fetch(url)

      .then(res => res.json())

      .then(res2 => {

        (dispatch2({ type: 'CHANGE\_NAME2', payload: res2[2].name }))

      })

  }

  return (

    <div className='App'>

      <h2>Name1 : {state.name}</h2>

      <h2>Name2 : {state2.name2}</h2>

      <button onClick={() => getName()}>Change name1</button>

      <button onClick={() => getName2()}>Change name2</button>

    </div>

  );

}

export default App;

**Global state Management**

Use all diff. component in one component:

**App.js:**

import './App.css';

import { createContext, useContext, useReducer, useState } from 'react';

const mycontext = createContext()

const initialState = {

  "name": "Click below button1",

  "name2": "Click below button2",

}

const reducer = (state, action) => {

  switch (action.type) {

    case 'CHANGE\_NAME':

      return {

        ...state,

        "name": action.payload,

      }

    case 'CHANGE\_NAME2':

      return {

        ...state,

        "name2": action.payload,

      }

    default:

      return state

  }

}

const User1 = () => {

  const myname = useContext(mycontext)

  return (

    <div>

      <h1>{myname.state.name}</h1>

      <h1>{myname.state2.name2}</h1>

    </div>

  )

}

const User2 = () => {

  const myname = useContext(mycontext)

  return (

    <div>

      <h1>{myname.state.name}</h1>

      <h1>{myname.state2.name2}</h1>

    </div>

  )

}

const User3 = () => {

  const myname = useContext(mycontext)

  return (

    <div>

      <h1>{myname.state.name}</h1>

      <h1>{myname.state2.name2}</h1>

      <button onClick={myname.getName}>Change name1</button>

      <button onClick={myname.getName2}>Change name2</button>

    </div>

  )

}

function App() {

  const [state, dispatch] = useReducer(reducer, initialState)

  const [state2, dispatch2] = useReducer(reducer, initialState)

  const [loder, setLoder] = useState(false)

  const url = 'https://jsonplaceholder.typicode.com/users'

  const getName = () => {

    setLoder(true)

    return fetch(url)

      .then(res => res.json())

      .then(res2 => {

        (dispatch({ type: 'CHANGE\_NAME', payload: res2[1].name }))

        setLoder(false)

      })

  }

  const getName2 = () => {

    setLoder(true)

    return fetch(url)

      .then(res => res.json())

      .then(res2 => {

        (dispatch2({ type: 'CHANGE\_NAME2', payload: res2[2].name }))

        setLoder(false)

      })

  }

  return (

    <mycontext.Provider value={{ 'getName': getName, 'getName2': getName2, 'state': state, 'state2': state2 }}>

      <div className='App'>

        <h1>{loder && "Data Fetching....."}</h1>

        <h2>Name1 : {state.name}</h2>

        <h2>Name2 : {state2.name2}</h2>

        <button onClick={() => getName()}>Change name1</button>

        <button onClick={() => getName2()}>Change name2</button>

        <User1 />

        <User2 />

        <User3 />

        <br /><br />

      </div>

    </mycontext.Provider >

  );

}

export default App;