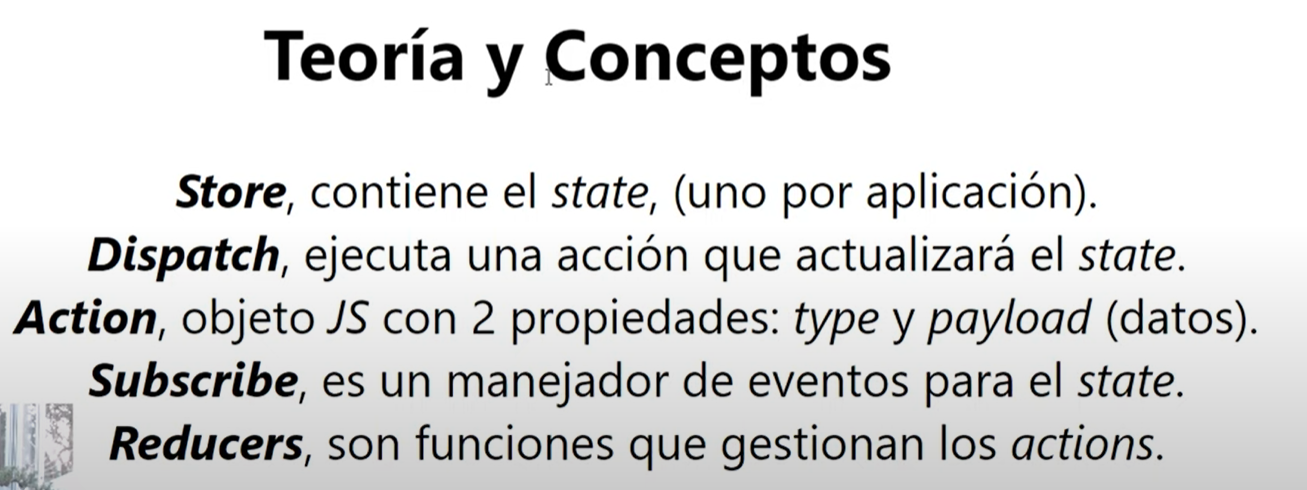
**REACT REDUX**



Creo los botones, diseno, etc.

SIN REDUX:

1. Creo los estados [counter, setCounter]

2. en el Layout, <Counter counter={counter}

3. Creo los metodos increment y decrement

4. en el Layout, <Counter counter={counter} increment={increment} decrement={decrement}/>

5. En Counter.jsx creo el onClick en los botones q ejecute increment y decrement.

**CON REDUX:**

Instalación de dependencias:

* npm i redux redux-thunk
* npm i react-redux

1. Creo carpeta src/store
2. Creo archivo store/index.js
3. Creo carpeta store/actions
4. Creo archivo store/actions/action-types.js
5. Creo archivo store/actions/index.js
6. Creo carpeta store/reducers
7. Creo archivo store/reducers/index.js
8. Creo mis action-types:

export const INC = "increment";

export const DEC = "decrement";

1. En actions/index.js

import { INC, DEC } from "./action-types";

export const increment = () => ({ type: INC });

export const decrement = () => ({ type: DEC });

1. En reducers/index.js

import { INC, DEC } from "../actions/action-types";

const initialState = { count: 0}

export default function reducer (state = initialState, { type, payload }) {

  switch (type) {

    case INC:

      return { ...state, count: state.count + 1 };

    case DEC:

      return { ...state, count: state.count - 1 };

    default:

      return state;

  }

};

1. En store/index.js

import { createStore } from "redux";

import reducer from "./reducers";

export const store = createStore(reducer);

CONEXION CON REDUX:

1. En main.jsx

import React from "react";

import ReactDOM from "react-dom/client";

import App from "./App";

import "antd/dist/antd.css";

import { Provider } from "react-redux";

import { store } from "./store";

ReactDOM.createRoot(document.getElementById("root")).render(

  <React.StrictMode>

    <Provider store={store}>

      <App />

    </Provider>

  </React.StrictMode>

);

1. En components/Counter.jsx

import { Button, Result } from "antd";

import { connect } from "react-redux";

import { increment, decrement } from "../store/actions";

function Counter({ counter, increment, decrement }) {

  return (

    <Result

      title="Este es un contador de ejemplo"

      subTitle={`El contador es ${counter}`}

      extra={[

        <Button key="inc" type="primary" onClick={() => increment()}>

          +

        </Button>,

        <Button key="dec" type="primary" onClick={() => decrement()}>

          -

        </Button>,

      ]}

    />

  );

}

const mapStateToProps = (state) => ({ counter: state.count });

const mapDispatchToProps = (dispatch) => {

  return {

    increment: () => dispatch(increment()),

    decrement: () => dispatch(decrement()),

  }

}

export default connect(mapStateToProps, mapDispatchToProps)(Counter);

1. Mi app quedaria asi:

import { Layout } from "antd";

import { useState } from "react";

import Counter from "./components/Counter";

import Header from "./components/Header";

function App() {

  return (

    <Layout>

      <Layout.Header><Header/></Layout.Header>

      <Layout.Content style={{ height: "100%", minHeight: "100vh" }}>

        <Counter/>

      </Layout.Content>

    </Layout>

  );

}

export default App;

ALTERNATIVA

La diferencia es q ahora mi component Counter recibe un estado, y mi component CounterActions solo recibe las acciones.

1. En Counter.jsx reemplazo

// const mapDispatchToProps = (dispatch) => {

//   return {

//     increment: () => dispatch(increment()),

//     decrement: () => dispatch(decrement()),

//   }

// }

// export default connect(mapStateToProps, mapDispatchToProps)(Counter);

export default connect(mapStateToProps, {increment, decrement})(Counter);

Creo Nuevo archive components/CounterActions.jsx

import { Button } from "antd";

import { connect } from "react-redux";

import { increment, decrement } from "../store/actions";

function CounterActions({ increment, decrement }) {

  return (

    <>

      <Button key="inc" type="primary" onClick={() => increment()}>

        +

      </Button>

      <Button key="dec" type="primary" onClick={() => decrement()}>

        -

      </Button>

    </>

  );

}

//connect(estado,acciones)

export default connect(null,{increment, decrement})(CounterActions)

Asi queda Counter.jsx

import { Result } from "antd";

import { connect } from "react-redux";

import CounterActions from "./CounterActions";

function Counter({ counter, increment, decrement }) {

  return (

    <Result

      title="Este es un contador de ejemplo"

      subTitle={`El contador es ${counter}`}

      extra={<CounterActions/>}

    />

  );

}

const mapStateToProps = (state) => ({ counter: state.count });

export default connect(mapStateToProps, null)(Counter);

AHORA VAMOS A VER LA MANERA MODERNA DE HACER LAS COSAS (HOOKS)

Usamos el hook useSelector y ya no usamos connect. Asi queda Counter.jsx

import { Result } from "antd";

import { useSelector } from "react-redux";

import CounterActions from "./CounterActions";

export default function Counter() {

  const counter = useSelector((state) => state.count);

  return (

    <Result

      title="Este es un contador de ejemplo"

      subTitle={`El contador es ${counter}`}

      extra={<CounterActions />}

    />

  );

}

Asi queda CounterActions.jsx

import { Button } from "antd";

import { useDispatch } from "react-redux";

import { increment, decrement } from "../store/actions";

export default function CounterActions() {

  const dispatch = useDispatch()

  return (

    <>

      <Button key="inc" type="primary" onClick={() => dispatch(increment())}>

        +

      </Button>

      <Button key="dec" type="primary" onClick={() => dispatch(decrement())}>

        -

      </Button>

    </>

  );

}

REDUX THUNK

Lo que me permite redux thunk es trabajar con acciones asíncronas.

Vamos a jsonplaceholder para traernos un usuario. https://jsonplaceholder.typicode.com/users/1

Empiezo por mis actiontypes

import { INC, DEC, SET\_USER } from "./action-types";

import axios from "axios";

export const increment = () => ({ type: INC });

export const decrement = () => ({ type: DEC });

export const getUserFromAPI = (id) => {

  return (dispatch) => {

    axios

      .get(`https://jsonplaceholder.typicode.com/users/${id}`)

      .then((response) => {

        dispatch({

          type: SET\_USER,

          payload: response.data,

        });

      });

  };

};

En reducers/index.js Creamos nuevo initial state y nuevo caso

import { INC, DEC, SET\_USER } from "../actions/action-types";

const initialState = {

  count: 0,

  user: undefined,

}

export default function reducer (state = initialState, { type, payload }) {

  switch (type) {

    case INC:

      return { ...state, count: state.count + 1 };

    case DEC:

      return { ...state, count: state.count - 1 };

    case SET\_USER:

      return { ...state, user: payload };

    default:

      return state;

  }

};

Despacho esta acción cuando se inicia mi componente header.

En Header.jsx

import { useEffect } from "react";

import { useDispatch } from "react-redux";

import { getUserFromAPI } from "../store/actions";

export default function Header() {

  const dispatch = useDispatch();

  useEffect(() => {

    dispatch(getUserFromAPI(1));

  }, []);

  return <nav>Esto es el nav</nav>;

}

En store/index.js

import { createStore, applyMiddleware } from "redux";

import thunk from "redux-thunk";

import reducer from "./reducers";

export const store = createStore(reducer, applyMiddleware(thunk));

Cambio el fondo del nav para poder ver:

import { Layout } from "antd";

import { useState } from "react";

import Counter from "./components/Counter";

import Header from "./components/Header";

function App() {

  return (

    <Layout>

      <Layout.Header style={{background: '#fff'}}><Header/></Layout.Header>

      <Layout.Content style={{ height: "100%", minHeight: "100vh" }}>

        <Counter/>

      </Layout.Content>

    </Layout>

  );

}

export default App;

Pero ahi quiero ver es el nombre de la persona, asi que en header.jsx

import { useEffect } from "react";

import { useSelector } from "react-redux";

import { useDispatch } from "react-redux";

import { getUserFromAPI } from "../store/actions";

export default function Header() {

  const dispatch = useDispatch();

  const user = useSelector((state) => state.user)//el nombre user viene del reducer

  useEffect(() => {

    dispatch(getUserFromAPI(1));

  }, []);

  return <nav>{user?.name}</nav>;

}

Ahora mi aplicación esta trabajando con Redux Thunk para conectarme a una API (llamada asíncrona).

Voy a enlazar mi counter con el id

import { useEffect } from "react";

import { useSelector } from "react-redux";

import { useDispatch } from "react-redux";

import { getUserFromAPI } from "../store/actions";

export default function Header() {

  const dispatch = useDispatch();

  const { user, count } = useSelector((state) => state)

  useEffect(() => {

    dispatch(getUserFromAPI(count));

  }, [count]);

  return <nav>{user?.name}</nav>;

}

Y como no hay ID 0 me voy a reducer y pongo el initial state en 1.

import { INC, DEC, SET\_USER } from "../actions/action-types";

const initialState = {

  count: 1,

  user: undefined,

};

export default function reducer(state = initialState, { type, payload }) {

  switch (type) {

    case INC:

      return { ...state, count: state.count + 1 };

    case DEC:

      return { ...state, count: state.count - 1 };

    case SET\_USER:

      return { ...state, user: payload };

    default:

      return state;

  }

}

Para poder usar las redux devtools tengo que en store/index.js

import { createStore, applyMiddleware, compose } from "redux";

import thunk from "redux-thunk";

import reducer from "./reducers";

const composeEnhancers = window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_COMPOSE\_\_ || compose;

export const store = createStore(reducer, composeEnhancers(applyMiddleware(thunk)));