Ejercicio

1. Crear un proyecto con el nombre miredux instalado dos componentes redux y react-redux.
   1. create-react-app miredux
   2. install redux react-redux
2. iniciar el proyecto(entrar a la carpeta miredux)
   1. npm start
3. Abrir el proyecto en el editor visual Sturio Code.
4. Verificar que se encuentran instalados en el proyecto los dos componentes redux y react.redux.
   1. En el archivo package.json del proyecto
5. Limpiar el proyecto para que únicamente muestre el texto “Hola mudo react-redux” en un h1 y como título de la página el texto “react-redux”.
6. INICIAMOS

Implementación de REDUX.

1. Crear la estrucctura de redux de la siguiente manera, en la carpeta src del proyecto, crear una carpeta que se llame redux y dentro de ella tres carpetas, store, reducers y actions, y dentro de actions, otra que se llame constants
   1. src
      1. redux
         1. store
         2. reducers
         3. actions
            1. actionincrementar
            2. actiondecrementar
         4. constants
2. Iniciaremos con la implementación de las acciones, dentro de la carpeta constants, creamos un archivo que se llame actionTypes.js
   1. export const INCREMENTAR= "INCREMENTAR";
   2. export const DECREMENTAR= "DECREMENTAR";

Hacer la implementación de las acciones, creamos un archivo que se llame actionincrementar.js dentro de la carpeta actions.js importando la constante creada en el punto anterior, la cual únicamente retornara un objeto plano con el nombre de la acción y con los valores que utilizará el reducer para actualizar el state, en su caso.

import { INCREMENTAR } from '../constants/actionTypes';

const Incrementar= () =>{

return {

type: INCREMENTAR

}

};

export default Incrementar;

1. De la misma forma hacemos la implementación de la acción decrementar.

import { DECREMENTAR } from '../constants/actionTypes';

const Decrementar= () =>{

return{

type: DECREMENTAR

}

};

export default Decrementar;

1. Hacer la implementación del reductor, dentro de la carpeta reducer crear un archivo reducer.js y dentro de el llevamos a cabo la implementación:

import { INCREMENTAR } from '../constants/actionTypes';

import { DECREMENTAR } from '../constants/actionTypes';

const stateIni= {

contador: 0

}

const Reducer= (state=stateIni, action) =>{

const stateCopy = {...state};

switch (action.type) {

case INCREMENTAR: return {

contador: ++stateCopy.contador

};

break;

case DECREMENTAR: return {

contador: --stateCopy.contador

};

break;

default: return(stateCopy);

break;

}

};

export default Reducer;

1. Continuamos con la implementación del Store, creamos un archivo store.js dentro de la carpeta store.

import { createStore } from 'redux';

import reducer from '../reducers/reducer';

const store= createStore(reducer);

export default store;

1. Hasta este momento ya tenemos implementado REDUX, el cual podemos testear, desde la consola, porque recordemos que no necesariamente debe estar conectado con REACT, para lo cual en el archivo index.js importamos el store y las dos acciones incrementar y decrementar y en el componente window agregamos estos componentes importados.

import React from 'react';

import ReactDOM from 'react-dom';

import App from './App';

import Store from './redux/store/store';

import Incrementar from './redux/actions/actionincrementar';

import Decrementar from './redux/actions/actiondecrementar';

window.store = Store;

window.incrementar= Incrementar;

window.decrementar= Decrementar;

ReactDOM.render(<App />, document.getElementById('root'));

1. Para probar la correcta implementación de REDUX, utilizamos tres comandos desde la consola,
   1. store.getState(); (muestra el estado del store),
   2. store.dispatch(incrementar()); (envía la función incrementar que corresponde a la acción INCREMENTAR),
   3. store.dispatch(decrementar()); (envía la función decrementar que corresponde a la acción DECREMENTAR)

1. Si todo está correcto veremos cómo se incrementa y decrementa el contador.

**Conectar REACT con REDUX**

1. Primeramente conectaremos el store con la aplicación, mediante un provider.

import React from 'react';

import ReactDOM from 'react-dom';

import App from './App';

import Store from './redux/store/store';

import Incrementar from './redux/actions/actionincrementar';

import Decrementar from './redux/actions/actiondecrementar';

import { Provider } from 'react-redux';

/\* Suscribe el store y las acciones en el objeto window del explorador\*/

window.store = Store;

window.incrementar= Incrementar;

window.decrementar= Decrementar;

ReactDOM.render(

/\* Con provider suscribe al componente react al store de redux, lo mete en el contecto de la App de react\*/

<Provider store={ Store }>

<App />

</Provider>, document.getElementById('root')

);

1. En el archivo App.js.
   1. Importar el connect, desde react-redux

import { connect } from 'react-redux';

* 1. Agregar la función mapStateToProps.

const mapStateToProps= (state) =>{

return{

contador: state.contador

}

}

* 1. Colocar el valor obtenido del state

<h1>{this.props.contador}</h1>

* 1. Connectar el state de REDUX, con las props del componente REACT.

const App = connect(mapStateToProps)(app);

import React, { Component } from 'react';

import { connect } from 'react-redux';

import Incrementar from './redux/actions/actionincrementar';

import Decrementar from './redux/actions/actiondecrementar';

/\* Para conectar el state con las props del componente \*/

const mapStateToProps= (state) =>{

return({

contador: state.contador

})

};

/\*Para conectar las funciones que se enviaran como acciones al generar un evento del componente react \*/

const mapDispatchToProps= (dispatch) =>{

return{

incrementar: () => dispatch(Incrementar()),

decrementar: () => dispatch(Decrementar())

}

};

class app extends Component {

constructor(props){

super(props);

this.incrementarAction = this.incrementarAction.bind(this);

this.decrementarAction = this.decrementarAction.bind(this);

}

render() {

return (

<div>

<h2>Hola mundo react-redux</h2>

{/\*Cuando el state cambia en el store, se renderiza este componete actualizando el valor del state \*/}

<h1>{this.props.contador}</h1>

{/\* Cuando se genera un evento en el componente, se envía una acción para actualizar es state\*/}

<button onClick={this.incrementarAction}>Incrementar</button>

<button onClick={this.decrementarAction}>Decrementar</button>

</div>

);

}

/\* Funciones para enviar las acciones al componete actions\*/

incrementarAction= () =>{

this.props.incrementar();

};

decrementarAction= () =>{

this.props.decrementar();

};

};

/\*Conecta el state y las acciones con el componete react App \*/

const App = connect(mapStateToProps,mapDispatchToProps)(app);

export default App;