React Redux by TechMentor Tutorials

Join us on youtube:

https://www.youtube.com/channel/UCg31rUwf-tfrpcBLe_SlZ9A

Redux is a predictable state container for javascript applications.

Hooks Example to Cover: **- 0 +**

```
<button onClick={()=>decrement()}>-</button>

{count}

<button onClick={()=>increment()}>+</button>
```

```
npx create-react-app reduxapp

npm install redux

npm install react-redux
```

1. **Action**: are javascript objects that have type field and payload and it uses for **What to Do**.

```
import { INCREMENT,DECREMENT } from "../constant"

export const increment =()=>{

   return {type:INCREMENT}

}

export const decrement =()=>{

   return {type:DECREMENT}

}
```

2. Why use Constants in react redux?

It allows you to easily find all usages of that constant across the project. It also prevents you from introducing silly bugs caused by typos -- in which case, you will get a ReferenceError immediately.

```
export const INCREMENT='INCREMENT'

export const DECREMENT='DECREMENT'
```

3. **Reducers**: are functions that take current state and and an action as arguments and return a new state as a result.

It uses for **How to Do**.

```
import { DECREMENT, INCREMENT } from "../constant";
const initialState=0
const changeCount=(state=initialState,action)=>{
  switch(action.type){
    case INCREMENT:
      return state+1
    case DECREMENT:
      return state-1
    default:
      return state
  }
}
export default changeCount
```

4. What is Root Reducer?

Redux uses a single root reducer function that accepts the current state (and an action) as input and returns a new state

The combineReducers helper function turns an object whose values are different reducing functions into a single reducing function you can pass to createStore . The resulting reducer calls every child reducer, and gathers their results into a single state object.

```
import { combineReducers } from "redux";
import CartItems from "./CartItems";
export default combineReducers({
  CartItems
})
```

5. **Store**: A store is an immutable object tree in Redux. A store is a state container which holds the application's state. Redux can have only a single store in your application

```
import {createStore} from 'redux'
import rootReducers from './reducers/index'
```

```
const store = createStore(rootReducers)

export default store;
```

Note: Redux store brings together state, actions and reducers that make up your app.

6. What is the use of provider?

to provide the data of store to react applicaiton

```
import store from './store'

import {Provider} from 'react-redux'

<Provider store={store}>

<App />

</Provider>
```

6. What is useSelector() hook?

The useSelector hook takes a function which receives as its first argument state, and then it allows you to return from that which part of the state that you want.

```
import {useSelector} from 'react-redux'

const count = useSelector((state)=>state.changeCount)
```

7. What is useDispatch() hook?

This hook returns a reference to the dispatch function from the Redux store. You may use it to dispatch actions as needed.

```
import {useDispatch} from 'react-redux'

const dispatch = useDispatch()

<button onClick={()=>dispatch(decrement())}>-</button>

{count}

<button onClick={()=>dispatch(increment())}>+</button>
```

8. What is mapStateToProps in redux?

mapStateToProps is used for selecting the part of the data from the store that the connected component needs. It's frequently referred to as just mapState for short. ... It receives the entire store state, and should return an object of data this component needs.

```
const mapStateToProps=(state)=>{

  return {count:state.changeCount}
}
```

9. What is mapDispatchToProps in redux?

mapDispatchToProps argument is responsible for enabling a component to dispatch actions.

```
const mapDispatchToProps = (dispatch) => {

   return {

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

   decrement: (num) => dispatch(decrement(num))

   }

}
```

10. What is connect() in react-redux?

The connect() function connects a React component to a Redux store. It provides its connected component with the pieces of the data it needs from the store, and the functions it can use to dispatch actions to the store.

```
import {connect} from 'react-redux'

export default connect(mapStateToProps,mapDispatchToProps)(PlusMinus)
```