# REACT-REDUX

## 🧩 useSelector

### 📌 Overview

useSelector is a hook provided by the React-Redux library. It allows you to extract data from the Redux store state using a selector function.

### 📚 Usage

* **Import the Hook**: First, import useSelector from react-redux.
* **Define a Selector Function**: The selector function takes the Redux state as an argument and returns the desired piece of state.
* **Use the Hook**: Call useSelector with the selector function inside your functional component.

### 📝 Example

import React from 'react';

import { useSelector } from 'react-redux';

const MyComponent = () => {

const myData = useSelector((state) => state.myData);

return (

<div>

<h1>{myData}</h1>

</div>

);

};

### 💡 Tips

* **Memoization**: Ensure your selector function is efficient to avoid unnecessary re-renders.
* **Reusability**: Consider defining selectors separately for reuse across components.

## 🏢 Provider

### 📌 Overview

Provider is a component from React-Redux that makes the Redux store available to any nested components that need to access the Redux state.

### 📚 Usage

* **Import the Component**: Import Provider from react-redux.
* **Wrap Your App**: Wrap your root component with Provider and pass the Redux store as a prop.

### 📝 Example

import React from 'react';

import ReactDOM from 'react-dom';

import { Provider } from 'react-redux';

import store from './store';

import App from './App';

ReactDOM.render(

<Provider store={store}>

<App />

</Provider>,

document.getElementById('root')

);

### 💡 Tips

* **Single Provider**: Typically, your application should have a single Provider at the root level.
* **Store Configuration**: Ensure your Redux store is correctly configured and passed to the Provider.

## 🚀 dispatch

### 📌 Overview

dispatch is a method available on the Redux store. It is used to send actions to the Redux store, triggering state changes.

### 📚 Usage

* **Access the Store**: You can access the dispatch method by using the useDispatch hook from react-redux.
* **Create Actions**: Define action creators that return action objects.
* **Dispatch Actions**: Call dispatch with an action object to update the state.

### 📝 Example

jsxCopy code

import React from 'react';

import { useDispatch } from 'react-redux';

const MyComponent = () => {

const dispatch = useDispatch();

const handleClick = () => {

dispatch({ type: 'MY\_ACTION\_TYPE', payload: 'myPayload' });

};

return (

<button onClick={handleClick}>

Dispatch Action

</button>

);

};

### 💡 Tips

* **Action Creators**: Use action creators to encapsulate the action logic.

When data is not store in redux

{isLoading: false, isError: false, data: Array(0)}

Array=0

But when added

Then{*isLoading*: *false*, *isError*: *false*, *data*: *Array(20)*}

Array is update by 20