Chapitre 13 : Redux-toolkit

Table des matières

I. Définition

Redux Toolkit est un ensemble d'outils qui permet de **simplifier le développement Redux**. Il comprend des utilitaires pour créer et gérer les magasins Redux, ainsi que pour écrire des actions et des réducteurs.

II. Installation

Pour utiliser Redux-toolkit, il suffit d'installer les deux packages suivants :

- npm install @reduxjs/toolkit
- npm install react-redux

III. La fonction createSlice

(Reference: https://openclassrooms.com/fr/courses/7150626-utilisez-le-state-manager-redux-pour-gerer-l-etat-de-vos-applications/7286902-unifiez-actions-et-reducers-grace-aux-slices-de-redux-toolkit)

Les slices de Redux Toolkit permettent de combiner la création des actions et la création du réducteur en un seul appel de fonction. Cela permet donc de définir les actions et le reducer qui va avec, d'un seul coup.

Pour créer un slice, on utilise la fonction createSlice de Redux-Toolkit. Cette dernière attend un objet en paramètre avec différentes propriétés :

- **name** : le nom du slice (le nom de la fonctionnalité). Ce nom est utilisé comme préfixe pour les noms des actions du slice.
- **initialState** : le state initial du slice.
- **reducers** : un objet qui définit à la fois les actions et la logique du reducer correspondant.

Exemple 1 :

```
import { createSlice } from "@reduxjs/toolkit";

const themeSlice = createSlice({
    name: 'theme',
    initialState: {show:true},
    reducers: {
        display: (state, action) => {
```

```
            return state.show ? false : true
        },
    },
})
```

Exemple 2 :

```
import { createSlice } from "@reduxjs/toolkit";

const counterSlice = createSlice ({
    name: "counter",
    initialState:{compteur : 0},
    reducers: {
        increment : (state, action) => {
            state.compteur += action.payload;
        },
        decrement : (state, action) => {
            state.compteur -= action.payload;
        },
        initialiser : (state, action) => {
            state.compteur = action.payload;
        },
    }
});
```

a. Récupérez le reducer et les actions

La fonction `createSlice` retourne un objet avec les propriétés suivantes :

- `name` : le nom du slice.
- `reducer` : le Reducer créé par Redux-Toolkit.
- `actions` : un objet contenant les action creators.

Exemple :

```
import { createSlice } from "@reduxjs/toolkit";

const counterSlice = createSlice ({
    name: "counter",
    initialState:{compteur : 0},
    reducers: {
        increment : (state, action) => {
            state.compteur += action.payload;
        },
        decrement : (state, action) => {
```

```
            state.compteur -= action.payload;
        },
        initialiser : (state, action) => {
            state.compteur = action.payload;
        },
    }
});

export const {increment, decrement, initialiser} = counterSlice.actions;

export default counterSlice.reducer;
```

IV. La fonction configureStore

Redux Toolkit a une méthode configureStore() qui simplifie le processus de configuration du magasin. Voici un exemple d'utilisation de cette méthode :

```
import counterReducer from './CounterSlice'
import { configureStore } from '@reduxjs/toolkit'
import { combineReducers } from 'redux'

const reducer = combineReducers({
    counter: counterReducer,
})
const store = configureStore({
  reducer,
})

export default store;
```