GESTION D'ÉTATS

EXEMPLE AVEC REDUX : TODOLIST

```ts
// action.ts
type NewTodo = {
  type: 'NEW';
  todo: Todo;
};
type CheckAll = {
  type: 'CHECK_ALL';
};
type Action = NewTodo | CheckAll;

const newTodo = (): NewTodo => ({
  type: 'NEW',
  todo: { checked: false, content: '' },
});
const checkAll = (): CheckAll => ({ type: 'CHECK_ALL' });
```

```ts
// store.ts
import { Reducer, Store, legacy_createStore as createStore } from 'redux';
const store: Store<State, Action> = createStore(reducer);
```

```tsx
// view.tsx
import { useDispatch, useSelector } from 'react-redux';
const Counter = () => {
  const dispatch = useDispatch();
  const todos = useSelector(todosSelector);
  return (
    <div>
      {todos.map(t => (
        <p>{t.content}</p>
      ))}
      <button onClick={() => dispatch(newTodo())} type="button"
title="+" />
    </div>
  );
};
```

```ts
// reducer.ts
type Todo = { content: string; checked: boolean };
type State = {
  todos: Array<Todo>;
};

const initialState: State = {
  todos: [],
};

const reducer: Reducer<State, Action> = (
  state: State | undefined,
  action: Action
) => {
  if (!state) return initialState;
  switch (action.type) {
    case 'NEW':
      return { ...state, todos: [...state.todos,
        action.todo] };
    case 'CHECK_ALL':
      return {
        ...state,
        todos: state.todos.map(t => {
          t.checked = true;
          return t;
        }),
      };
  }
};

export const todosSelector = (state: State) =>
  state.todos;
```

GESTION D'ÉTATS

REDUX

Remarques :

- **N'utilisez pas Redux Tools Kit** avant d'avoir complètement maîtrisé les concepts de Redux.
  RTK a été conçu pour éviter le boilerplate pour mettre en place Redux dans une application.

- Lisez la **documentation officielle** qui explique bien les concepts de Redux : https://redux.js.org

- Le lien entre Redux et React se fait notamment via la librairie react-redux. Elle apporte notamment les hooks ***useDispatch*** et ***useSelector***