# RTK Quert

可以簡化Reduce(slice)內的code

引入

|  |  |
| --- | --- |
| createApi |  |
| ferchBaseQyery |  |

流程：

1. 建立API的接點

export const todoApiService = createApi({

    // store 中要使用的 reducer 名稱

    reducerPath: 'todoApi',

    // 基礎路徑

    baseQuery: fetchBaseQuery({ baseUrl: 'https://jsonplaceholder.typicode.com' }),

    // 建立 get、post、delete function 的地方

    endpoints: (builder) => ({

        // .query 配對 baseQuery 與這個方法內的路徑

        // myType = return 回來的資料

        // string = 傳入的參數類型

        getTodoList: builder.query<myType, string>({

            query: (id) => `/todos/${id}`,

        }),

    }),

})

// 導出 Hook

export const { useGetTodoListQuery } = todoApiService

1. 在store內新增API的reducer

const reducers = combineReducers(

    {

        todoReducer,

        testReducer,

        [todoApiService.reducerPath]: todoApiService.reducer,

    }

)

// 創建store提供給Provider使用

const store = configureStore({

    reducer: reducers,

    middleware:( getCurrentMiddlewares )=>{

        return getCurrentMiddlewares()

        // .concat(loggerMiddleware)

        .concat(todoApiService.middleware) // 記得要加上concat

    }

})

// 導出store內會回傳的所有type，useSelector會自動推斷類型

export type RootState = ReturnType<typeof reducers>

export type AppDispatch = typeof store.dispatch

export default store

1. 在頁面中使用Hook

const [dataSelector, setDataSelector] = useState("")

const { data, error, isLoading } = useGetTodoListQuery(dataSelector);