

Reducer在redux中的作用为，接受一个action并且更新Redux的dataStore.

在此基础上，Reducer Enhancer是定义的一种方法，这个方法接收reducer作为参数，并且返回一个新的方法。该新方法接受两个参数(storeData, action)，该新方法返回storeData经过action过后新的状态对象。可以看出返回的方法实际上便也是一个Reducer方法，接受代表现有状态的storeData以及Action，返回代表新状态的对象。

例如：

export function customReducerEnhancer(originalReducer){ //ReducerEnhancer为一个方法，接受reducer作为参数

let initialState = null;

return (storeData, action)=> { //ReducerEnhacner最终返回一个新的方法，该方法接受两个参数storeData,以及action,并且该新方法返回storeData经过action过后新的状态对象。

if (action.type === STORE\_RESET && initialData != null){

return initialState

}

else{

const result = originalReducer(storeData, action);

if(initialState === null){

initialState = result

}

return result

}

}

}

Reducer Enhance的作用是action在达到reducer前先经过Reducer Enhancer的预处理。可以达到在Reducer Enhancer中对action进行先行拒绝，添加属性等，再分派到reducer中。

Redux的combineReducers便是Redux自带的Reducer Enhancer, 它的作用是将action分发给多个Reducer处理。 Reducer Enhancer对于Redux的dataStore来说是透明的，dataStore并不会知道是否采用了Reducer Enhancer. 它总体遵循的流程还是，action最终交付到Reducer，然后进而reducer更新dataStore.