Vuex: 集中式管理状态

核心： store

存储响应式 、 不能直接改变状态

New Vuex.Store({

State:{},

Mutations:{

Fun(state){//改变\同步函数}

},

Getters: {} // store的计算属性

Action: {

actionFunc(context){context.commit(‘Func’)}

async actionA(){await xxfunc() / commit(‘Func’, await getFunc())}

} // 提交mutation / 可异步 / 返回promise

Modules: { // store臃肿，分块

// rootState根节点状态

// rootGetters

}

Plugins: [func(){  
 // mutation{type: ‘’}

}]

})

读取： store.state.xx

改变： store.commit(‘Fun’) / store.commit({type: ‘func’, xx: ‘xx’})

异步触发：store.dispatch(‘ActionFunc’) / store.dispatch({type: ‘ActionFunc’, xx:’xx’})

1. 注入每个组件Vue.use(vuex)
2. This.$store.state.xx

mapState

mapGetters// computed

mapMutations({}) // methods

mapActions({}) // methods

namespace:

注册全局action root:true handler(函数)

registerModule

store.registerModule([‘a’, ‘b’], {}) // 嵌套模块

preserveState: true

vuex 局部状态

多个store，函数声明

不允许直接修改数据

Redux 单项数据流

1.Store: { // 单个}

State / getState /setState/ dispatch(action) / subscribe(listener) /

createStore(reducers)

2.action对象{

type: ‘’, // 必要

text: ‘’

}

异步action: thunk action

开始 reducer-state-isFetching

成功

失败

Middleware最后一个要普通actIon

嵌入框架请求到响应之中的代码

链式组合

Redux middleware 从action 开始到 reducer之前

applyMiddleware

thunkMiddleware

Action创建函数 、dispatch\ connect帮助器 => 容器组件（展示组件 + redux） => store.subscribe \ bindActionCreators绑定多个

1. Reducer 纯函数: state + action 合成 / combineReducers(object) => reducers（合成state tree）

React-redux

Provider： 将redux绑定在react

Connect: (mapStateToProps,mapDispatchToProps)

源码：

.common. 现代模块规范（commonJ s\ EcmaScript Module）

.esm 模板字符串 + .vue => js