VueX

**1.介绍:**

Vuex是为了保存组件之间共享数据而诞生的，如果组件之间需要共享数据，可以直接挂载到vuex中，而不必通过父子组件，兄弟组件间进行传参。如果组件的数据不需要共享，此时这些不需要共享的私有数据，没必要放在vuex中。只需放在该组件中的data中即可。由此可以得出一个结论:vuex是一个全局的共享数据存储区域，就相当于一个数据的仓库。

**2.vue-cli2.0脚手架中的配置步骤:**

**(1)下载安装vuex包；npm install vuex -save**

Note:以下操作在main.js中配置

**(2)在main.js中引入vuex**

Import Vuex from ‘vuex’

**(3)注册vuex到vue中**

Vue.use(Vuex)

**(4)new Vuex.store()实例，得到一个数据仓库对象。**

var store = new Vuex.store({})

**(5)将创建的store实例挂载到new Vue中,只要挂载到new Vue实例上，任何组件都可以使用store 来存储数据**

Const vm=new Vue({

El:“#app”，

Render:c=>c(APP),

store

})

**3.store中重要属性介绍**

**总概:**

(1)state(存储组件间共享数据,一般所有的ajax请求回来的数据都可用state管理)

(2)mutations(用来存储直接修改state中变量的方法)

(3)action(用来存储间接修改state中变量的方法，多为异步方法)

(4)getters(存储用来显示数据的方法,等同于computed)

(5)modules(存储不同模块下的state,mutations,getters,actions)

详细介绍

**3-1.state**

**(1)定义state**

①定义state.js，存储所有的数据变量

export default {

  count: 1,

  firstName: "jocky",

  lastName: "lou"

};

②在index.js vuex的入口文件中引入state.js文件

import defaultState from "./state/state";

③配置new Vuex.store中的state属性

state: defaultState,

**(2)组件中使用state数据(一般在计算属性中获取)**

组件中使用state中变量的方式:**this.$store.state.变量名**

**3-2.mutations**

**(1)定义mutations**

export default {

  increment(state, { c, d }) {

    state.count += c + d;

  },

};

**(2)组件中调用mutations中的方法**

方法1：this.$store.commit("increment", { c: 1, d: 2 });:

注:

①可以把mutations等价于methods，专门存储同步代码的共享方法。操作state中的数据。

②某组件如果想要修改state中的数据，必须调用mutations提供的方法，才能操作对应的数据，不推荐在组件内直接操作state中的数据，因为万一导致了数据的紊乱，不能快速定位到错误的原因，因为每个组件都有操作数据的可能。如果组件想要调用mutations中的方法，只能使用**this.$store.commit(‘方法名’,{参数2，参数3})；**

③mutations的函数参数列表中，最多支持两个参数，其中参数1:state；参数2:组件通过commit提交过来的参数，如果参数2有多个值，可以使用对象语法

**3-3.action**

**情况1：在action中定义异步方法**

**(1)定义action中的异步方法**

// 异步获取商家列表(商品首页组件)

  async getShops({ commit, state }) {

    // 发送异步ajax请求

    // 对象解构获取state中的经纬度两个变量

    const { longitude, latitude } = state;

    const result = await reqShops(longitude, latitude);

    // 提交一个mutation

    if (result.code === 0) {

      const shops = result.data;

      commit(RECEIVE\_SHOPS, { shops });

    }

  },

**(2)组件中使用action中的方法。**

this.$store.dispatch("getShops");

**情况2：在action中定义同步方法**

**(1)定义action中的同步方法**

 recordUser({ commit }, userInfo) {

    commit(RECEIVE\_USER\_INFO, { userInfo });

  },

**(2)调用action中的同步方法**

this.$store.dispatch("recordUser", user);

**3-4.getters**

**(1)定义getters**

export default {

  opCount(state) {

    return `当前的数值为:${state.count}`;

  }

};

**(2)组件中调用getters中的方法(同组件中的computed语法相同)**

computed: {

   fullName() {

      return this.$store.getters.fullName;

    }

  }

注:

①getters相当于组件中的computed,只负责对外提供数据，不负责修改数据；

②组件中使用getters中的方法时:**this.$store.getters.方法名**

**4.为vuex增加热更替的功能:**

if (module.hot) {

    module.hot.accept(

      [

        "./state/state",

        "./mutations/mutations",

        "./getters/getters",

        "./action/action"

      ],

      () => {

        const newState = require("./state/state").default;

        const newMutations = require("./mutations/mutations").default;

        const newGetters = require("./getters/getters").default;

        const newActions = require("./action/action").default;

        store.hotUpdate({

          state: newState,

          mutations: newMutations,

          getters: newGetters,

          actions: newActions

        });

      }

    );

  }

扩展:

Vuex提供了方便的方法mapState,mapMutations,mapGetters,mapAction,组件中在使用store中的共有资源时可以这样使用:

①在组件中引入上面的几个方法；

②-①如果组件中的函数名与store中定义的函数方法名相同，则可以如下操作:

…mapState([“函数名”])

②-②如果组件中的函数名与store中定义的函数方法名不相同，则可以如下操作:

…mapStore({组件中的函数名:‘store中定义的方法名’})；

或者:…mapStore({组件中的函数名：(state)=>state.count})