Vuex

Vuex是实现组件全局状态（数据）管理的一种机制，可以方便实现组件之间数据的共享

**好处：**

1. 能够在vuex中集中管理共享的数据
2. 能够高效的实现组件之间的数据共享，提高开发效率
3. 存储在vuex中的数据都是响应式的，能够实时保持数据与页面同步

一般情况下，只有组件共享的数据，才有必要存储到vuex中，对于组件中私有的数据，依旧存储在组件自身的data中

1. 基本使用
   1. 安装vuex依赖包

Npm install vuex –save

* 1. 导入vuex包

Import vuex from ‘vuex

* 1. 创建store对象

Const store = new Vuex.store({  
 state:{} //存放全局共享数据

})

* 1. 将store挂载到vue实例中

Const vue = new Vue({

Store

})

1. 核心概念概述

Vuex中主要概念如下

State,

Mutation,

Action,

Getter

注：State、Getter映射为组件的计算属性，Mutation、Action映射为组件的方法

1. **State的使用**

解释：提供唯一的公共数据，所有共享的数据都要统一放到store中进行存储，

//创建store数据源，提供唯一公共数据

Const store = new Vuex.store({  
 state:{

Count: 0

}

}

//使用组件访问state中数据有两种方式

//**第一种**

This.$store.state.全局数据名

//**第二种**（将state全局数据映射为当前组件的全局数据）

（1）从vuex中按需导入mapState函数

Import { mapState } from ‘vuex’

(2)通过导入的mapState函数将当前组件需要的全局数据映射为当 前组件的计算属性

**当前组件下**： computed:{

...mapState([‘全局数据名’])  
}

1. **Mutation**

解释:mutation用于变更store中的数据，只能通过mutation变更store数据，不可直接操作store中的数据

//定义mutation

Const store = new vuex.store({

State:{},

Mutations:{ //state固定参数必须有

Adda: function(state){}，

Addb:function(state,step){}

}

}

//触发mutation两种方式

//**第一种** (this.$store.commit方法)

在当前组件下：methods:{  
 add(){  
 this.$store.commit(‘Adda’)

}

}

//**第二种**（将mutation函数映射为当前组件的函数）

1）从vuex中按需导入mapMutations函数

Import { mapState，mapMutations } from ‘vuex’

(2)通过导入的mapMutation函数将当前组件需要的mutation函数映射为当前组件的method 方法

**当前组件下**：methods: {  
 …mapMutations([‘Adda’]

Add(){

This.Adda()

}

}

//mutation在触发时传递参数

//当前组件下 methods: {  
 add(){

This.$store.commit(‘Addb’,5) //5为参数

}

}

1. Action

解释：用于处理异步任务

如果通过异步任务操作变更数据，必须通过Action,而不能使用mutation,但在Action中还是要通过触发mutation的方式间接变更数据

//Action定义

Const store = new Vuex.store({

State:{},

Mutations:{},

Actions:{ //context固定参数必须有

addAsync(context){ //2秒后触发Adda mutation函数   
 setTimeout(()=>context.commit(‘Adda’),2000)

}

}

}

//触发Action两种方式

//**第一种** (this.$store.dispatch方法)

在当前组件下：methods:{  
 add(){  
 this.$store.dispatch(‘addAsync’)

}

}

//**第二种**（将Actions函数映射为当前组件的函数）

1）从vuex中按需导入mapActions函数

Import { mapState，mapMutation,mapActions } from ‘vuex’

(2)通过导入的mapMutation函数将当前组件需要的mutation函数映射为当前组件的method 方法

**当前组件下**：methods: {  
 …mapActions([‘addAsync’]

Add(){

This.addAsync()

}

}

//Action在触发时传递参数

//当前组件下 methods: {  
 add(){

This.$store.dispatch(‘addAsync’,5) //5为参数

}

}

1. Getter

解释：可以对store中的数据进行加工处理之后形成新的数据，类似Vue的计算属性。

Store中的数据发生变化时，Getter的数据也会跟着变化

//定义Getter

New Vuex.store({  
 state:{},

Mutations:{},

Actions:{},

Getter:{ //固定参数state必须有

Shownum(state){  
 return ‘num’+state.num

}

}

//使用组件访问state中数据有两种方式

//**第一种**

This.$store.getters.名称（例如Shownum）

//**第二种**

（1）从vuex中按需导入mapGetters函数

Import { mapGetters } from ‘vuex’

(2)通过导入的mapGetters函数将当前组件需要的数据映射为当 前组件的计算属性

**当前组件下**： computed:{

...mapGetters([‘Shownum’])  
}

1. Vuex错误弹框

This.$message.warning(‘错误信息’)