Vuex

Vuex 是一个专为 Vue.js 应用程序开发的**状态管理模式。**

调试工具：vue devtools

> Vuex就像眼镜：您自会知道什么时候需要它。

1、state

在store中定义数据，在组件中直接使用：

目录：`store/index.js`

```
export default new Vuex.Store({
    // state相当于组件中的data，专门用来存放全局的数据
    state: {
        num: 0
    },
    getters: {},
    mutations: {},
    actions: {},
    modules: {}
})
```

目录：`Home.vue`

```
<template>
  <div class="home">
    <h2>Home页面的数字：{{$store.state.num}}</h2>
  </div>
</template>

<script>
export default {

}
</script>
```

或者写为：

```
<template>
  <div class="about">
    <h2>About页面的数字：{{num}}</h2>
  </div>
</template>

<script>
export default {
  computed: {
    num(){
      return this.$store.state.num
    }
  }
```

```
}
</script>
```

2、getters

将组件中统一使用的computed都放到getters里面来操作

目录：`store/index.js`

```
export default new Vuex.Store({
    // state相当于组件中的data，专门用来存放全局的数据
    state: {
        num: 0
    },
    // getters相当于组件中的computed，getters是全局的，computed是组件内部使用的
    getters: {
        getNum(state) {
            return state.num
        }
    },
    mutations: {},
    actions: {},
    modules: {}
})
```

目录：`Home.vue`

```
<template>
  <div class="home">
    <h2>Home页面的数字：{{$store.getters.getNum}}</h2>
  </div>
</template>

<script>
export default {

}
</script>
```

3、mutations

更改 Vuex 的 store 中的状态的唯一方法是提交 mutation。

目录：`store/index.js`

```
export default new Vuex.Store({
    // state相当于组件中的data，专门用来存放全局的数据
    state: {
        num: 0
    },
    // getters相当于组件中的computed，getters是全局的，computed是组件内部使用的
    getters: {
        getNum(state) {
            return state.num
        }
    },
```

```
    // mutations相当于组件中的methods，但是它不能使用异步方法（定时器、axios）
    mutations: {
        // 让num累加
        // payload是一个形参，如果组件在commit时，有传这个参数过来，就存在，如果没有传过来，
就是undefined
        increase(state, payload){
            state.num += payload ? payload : 1;
        }
    },
    actions: {},
    modules: {}
})
```

目录：`Btn.vue`

```
<template>
    <div>
        <button @click="$store.commit('increase', 2)">点击加1</button>
    </div>
</template>
<script>
export default {
   methods: {
       /* addFn(){
           // 调用store中的mutations里的increase方法
           // 传参的话，使用payload
           this.$store.commit('increase', 2)
       } */
   }
}
</script>
```

4、actions

actions是store中专门用来处理异步的，实际修改状态值的，还是mutations

目录：`store/index.js`

```
// 在store(仓库)下的index.js这份文件，就是用来做状态管理
import Vue from 'vue'
import Vuex from 'vuex'

Vue.use(Vuex)

export default new Vuex.Store({
    // state相当于组件中的data，专门用来存放全局的数据
    state: {
        num: 0
    },
    // getters相当于组件中的computed，getters是全局的，computed是组件内部使用的
    getters: {
        getNum(state) {
            return state.num
        }
    },
    // mutations相当于组件中的methods，但是它不能使用异步方法（定时器、axios）
```

```
    mutations: {
        // 让num累加
        // payload是一个形参，如果组件在commit时，有传这个参数过来，就存在，如果没有传过来，
就是undefined
        increase(state, payload){
            state.num += payload ? payload : 1;
        },
        // 让num累减
        decrease(state){
            state.num--;
        }
    },
    // actions专门用来处理异步，实际修改状态值的，依然是mutations
    actions: {
        // 点击了“减1”按钮后，放慢一秒再执行减法
        decreaseAsync(context){
            context.commit('decrease')
        }
    },
    modules: {}
})
```

目录：`Btn.vue`

```
<template>
    <div>
        <button @click="$store.commit('increase', 2)">点击加1</button>
        <button @click="$store.dispatch('decreaseAsync')">点击延迟减1</button>
    </div>
</template>
<script>
export default {
   methods: {
       /* addFn(){
           // 调用store中的mutations里的increase方法
           // 传参的话，使用payload
           this.$store.commit('increase', 2)
       }
       reduceFn(){
           this.$store.dispatch('decreaseAsync')
       } */
   }
}
</script>
```

5、辅助函数

mapState和mapGetters在组件中都是写在computed里面

```
<template>
    <div>
        <h2>Home页面的数字：{{num}}</h2>
        <h2>About页面的数字：{{getNum}}</h2>
    </div>
</template>
```

```
<script>
import { mapState, mapGetters } from 'vuex'

export default {
  computed: {
    ...mapState(['num'])
    ...mapGetters(['getNum'])
  }
}
</script>
```

mapMutations和mapActions在组件中都是写在methods里面

```
<template>
    <div>
        <button @click="increase(2)">点击加1</button>
        <button @click="decreaseAsync()">点击延迟减1</button>
    </div>
</template>

<script>
import { mapMutations, mapActions } from 'vuex'

export default {
  methods: {
       ...mapMutations(['increase']),
       ...mapActions(['decreaseAsync'])
   }
}
</script>
```

6、拆分写法

store中的所有属性，都可以拆分成单独的js文件来书写

7、modules

我们的store可以认为是一个主模块，它下边可以分解为很多子模块，子模块都可以单独领出来写，写完再导入到主模块中。下面以 `users` 子模块举例：

将mutations中所有的方法，归纳起来。

目录：`mutations_type.js`

```
export const MUTATIONS_TYPE = {
    INCREASE: 'increase',
    DECREASE: 'decrease'
}

export default {
    // 让num累加
    // payload是一个形参，如果组件在commit时，有传这个参数过来，就存在，如果没有传过来，就是
undefined
    [MUTATIONS_TYPE.INCREASE](state, payload){
        state.num += payload ? payload : 1;
```

```
    },
    // 让num累减
    [MUTATIONS_TYPE.DECREASE](state){
        state.num--;
    }
}
```

目录：`store/index.js`

```
import mutations from './mutaions_type'

export default new Vuex.Store({
    ...
    mutations,
    ...
})
```

组件中：

```
<template>
  <div class="about">
    <h2>About页面的数字：{{getNum}}</h2>
    <button @click="increase()">About的按钮，点击加1</button>
  </div>
</template>
<script>
import { mapGetters, mapMutations } from 'vuex'
import { MUTATIONS_TYPE } from '@/store/mutaions_type.js'
export default {
  computed: {
    ...mapGetters(['getNum'])
  },
  methods: {
    // 方法一：
    ...mapMutations([MUTATIONS_TYPE.INCREASE])

    // 方法二：
    /* increase(){
      this.$store.commit(MUTATIONS_TYPE.INCREASE)
    } */
  }
}
</script>
```