**App.vue**

<template>

<div id="app">

<div id="nav">

<router-link to="/">Home</router-link> |

<router-link to="/about">任务列表</router-link>

</div>

<router-view/>

</div>

</template>

<style>

#app {

font-family: Avenir, Helvetica, Arial, sans-serif;

-webkit-font-smoothing: antialiased;

-moz-osx-font-smoothing: grayscale;

text-align: center;

color: #2c3e50;

}

#nav {

padding: 30px;

}

#nav a {

font-weight: bold;

color: #2c3e50;

}

#nav a.router-link-exact-active {

color: #42b983;

}

</style>

**Main.js**

import Vue from 'vue'

import App from './App.vue'

import router from './router'

import axios from 'axios'

import Vuex from 'vuex'

Vue.use(Vuex)

Vue.config.productionTip = false

Vue.prototype.$axios = axios;

const store = new Vuex.Store({

// vue中的单一状态树

state: {

taskLists: [{

title:"每日晨跑",completed:true

},{

title:"按时吃饭",completed:false

},{

title:"好好学习",completed:true

}]

},

// 更改 Vuex 的 store 中的状态，提交 mutation是更改 Vuex 的 store 中的状态的唯一方法

mutations: {

// setTacks(state, taskLists) {

// state.taskLists = taskLists; // 赋值

// },

add(state, data) { // 添加

state.taskLists.push(data); // 将数据push到数组中

},

del(state, index) { // 删除

state.taskLists.splice(index, 1); // index为数据下标，将数据从数组中splice

}

},

actions: {

add({commit}, data) {

commit('add', data); // context.commit 提交一个 mutation

},

del({commit}, index) {

commit('del', index); // context.commit 提交一个 mutation

}

// actions异步操作，axios获取到数据

// setTacks(context) {

// axios.get('https://jsonplaceholder.typicode.com/todos').then((response) => {

// context.commit('setTacks', response.data); // context.commit 提交一个 mutation

// })

// }

},

// store 的计算属性

getters: {

taskLists: state => {

return state.taskLists // 返回state.taskLists

}

}

})

new Vue({

router,

store,

render: h => h(App)

}).$mount('#app')

**About.vue**

<template>

<div class="about">

<h2>添加任务</h2>

<div class="addDiv">

<input type="text" class="input" v-model="inputAdd" />

<input type="button" class="button" value="添加" @click="addNew" />

</div>

<br />

<h2>任务列表</h2>

<div>

<p>总条数：{{this.$store.getters.taskLists.length}}条</p>

显示条数：

<select @change="showtaskLists($event)">

<option>10</option>

<option>50</option>

<option>100</option>

<option >200</option>

</select>

</div>

<br />

<div>

<table width="600px" border="1" >

<tr>

<th>编号</th>

<th>内容</th>

</tr>

<tr v-for="(task,index) in this.$store.getters.taskLists" :key="index" v-if="index<selectNum">

<td align="center">

{{index+1}}

</td>

<td :class="task.completed==true?'style1':'style2'" width="360px">

<div class="thisPositionMa">

<div class="thisLeft">

{{task.title}}

</div>

<div class="thisRight" @click="del(index)">

X

</div>

</div>

</td>

</tr>

</table>

<!-- <div class="thisTode" v-for="(task,index) in this.$store.getters.taskLists" :key="index" v-if="index<selectNum" :class="task.completed==true?'style1':''">

{{task.title}}

</div> -->

</div>

</div>

</template>

<style scoped="scoped">

.about {

text-align: left;

}

.addDiv {

margin-bottom: 20px;

}

.input {

width: 300px;

height: 32px;

font-size: 18px;

}

.button {

width: 80px;

height: 40px;

font-size: 18px;

}

select {

width: 70px;

height: 35px;

font-size: 18px;

}

/\* .thisTode {

width: 32%;

height: 60px;

line-height: 60px;

text-align: center;

float: left;

margin: 0 10px 10px 0;

box-sizing: border-box;

background-color: #409EFF;

overflow: hidden;

} \*/

.style1 {

background-color: #42B983;

}

.style2 {

background-color: darkgray;

}

table {

}

td {

width: 200px;

height: 50px;

}

th {

height: 50px;

text-align: center;

}

.thisPositionMa {

padding: 0 10px 0 10px;

}

.thisLeft {

float: left;

}

.thisRight {

float: right;

cursor: pointer;

}

</style>

<script>

import { mapGetters } from 'vuex' // 导入 mapGetters 辅助函数

export default {

name:'about',

data(){

return{

inputAdd: '', // 输入框内容

selectNum: 10 // 下拉选中条数

}

},

created() {

//this.$store.dispatch('setTacks'); // 在组件中发送请求的时候，需要使用 this.$store.dispatch 来分发

},

methods:{

// 添加操作，点击添加按钮执行

addNew: function(){

if(this.inputAdd=='') { // 判断输入框内容是否为空

alert('请输入Task后再点击添加！');

return; // 为空则return跳出函数

}

this.$store.dispatch( 'add', {title:this.inputAdd,completed:true});

this.inputAdd=''; // 成功后将输入框清空

// axios提交添加的数据

// this.$axios.post('https://jsonplaceholder.typicode.com/todos', this.inputAdd).then((response) => {

// alert('添加新的Task成功！');

// this.inputAdd=''; // 成功后将输入框清空

// }).catch((err) => {

// alert('添加失败！错误信息：'+err); // 错误后提示错误信息

// })

},

// 获取下拉选框选中的条数

showtaskLists: function(ele){

this.selectNum = ele.target.value; // 赋值

},

// 删除操作，点击x号删除

del: function(index) {

this.$store.dispatch( 'del', index) // 使用 this.$store.dispatch 来分发

}

},

computed:{

...mapGetters(['taskLists']) // 映射 this.taskLists 为store.state.taskLists

}

}

</script>