link: null
title: 珠峰架构师成长计划
description: 初始化项目环境
keywords: null
author: null
date: null
publisher: 珠峰架构师成长计划
stats: paragraph=60 sentences=108, words=901

vue中的状态管理

初始化项目环境

```
sudo npm install @vue/cli -g
vue ui &#x751F;&#x6210;&#x9879;&#x76EE; vue-router + vuex
```

EventBus事件车

在vue中传递数据是通过属性传递(父子关系)，子父通信是通过emit来触发父级事件。如果遇到平级组件可以通过共同的父级进行数据传递。但是在开发中，我们经常会遇到平级组件数据交互和跨组件数据交互就可以通过一个共同的实例来进行数据传递。 通过事件来共享数据（发布订阅模式）

创建bus实例

```
// lib/bus.js
import Vue from 'vue';
let $bus = new Vue();
Vue.prototype.$bus = $bus;
```

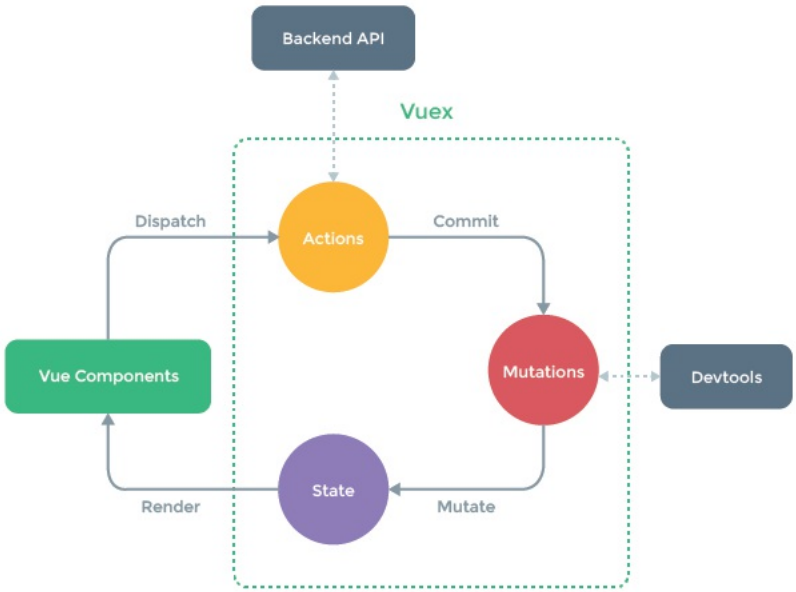
```
// main.js
import './lib/bus';
```

```
// Boy&#x7EC4;&#x4EF6; &#x53D1;&#x5C04;dinner&#x4E8B;&#x4EF6;
<template>
    <div>&#x7537;&#x5B69;
        <button @click="sayToGirl()">&#x5BF9;&#x5973;&#x5B69;&#x8BF4;&#x8BDD;</button>
    </div>
</template>
<script>
export default {
    methods: {
        sayToGirl(){
           this.$bus.$emit('dinner','你饿吗');
        }
    }
}
</script>
```

```
// Girl&#x7EC4;&#x4EF6; &#x76D1;&#x542C;dinner&#x4E8B;&#x4EF6;
<template>
    <div>
    &#x5973;&#x5B69; <span>&#x7537;&#x5B69;&#x5BF9;&#x6211;&#x8BF4;&#xFF1A; {{message}}</span>
    </div>
</template>

<script>
export default {
    data(){
        return {message:''}
    },
    mounted() {
        this.$bus.$on('dinner',(data)=>{
            this.message = data;
        })
    }
}
</script>
```

使用vuex来管理状态




```
&#x251C;&#x2500;&#x2500; actions.js
&#x251C;&#x2500;&#x2500; getters.js
&#x251C;&#x2500;&#x2500; index.js
&#x251C;&#x2500;&#x2500; modules
&#x2502;   &#x2514;&#x2500;&#x2500; teacher.js
&#x251C;&#x2500;&#x2500; mutations.js
&#x2514;&#x2500;&#x2500; state.js
```

state & getters

- state类比组件中的状态

```
// state.js
let state = {
    lesson:'&#x73E0;&#x5CF0;&#x67B6;&#x6784;'
}
export default state;

// &#x53EF;&#x4EE5;&#x5728;&#x7EC4;&#x4EF6;&#x4E2D;&#x76F4;&#x63A5;&#x901A;&#x8FC7;
this.$store.state&#x6765;&#x8BBF;&#x95EE;&#x6570;&#x636E;&#xFF0C;&#x4E5F;&#x53EF;&#x4EE5;&#x901A;&#x8FC7;&#x8BA1;&#x7B97;&#x5C5E;&#x6027;&#x7684;&#x65B9;&#x5F0F

<template>
    <div>
        &#x8BFE;&#x7A0B;&#x662F;:{{lesson}}
    </div>
</template>
<script>
export default {
    computed: {
        lesson(){
            return this.$store.state.lesson
        }
    }
}
</script>

// vue&#x63D0;&#x4F9B;&#x7684;&#x8F85;&#x52A9;&#x51FD;&#x6570;&#x5B9E;&#x73B0;
import {mapState} from 'vuex';
export default {
    computed: {
        // &#x76F4;&#x63A5;&#x53D6;&#x51FA;&#x72B6;&#x6001;&#x4E2D;&#x7684;&#x7ED3;&#x679C;
        ...mapState(['lesson']),
        // &#x7ED9;&#x72B6;&#x6001;&#x8D77;&#x540D;&#x5B57;
        ...mapState({lesson1:'lesson'}),
        // &#x901A;&#x8FC7;&#x51FD;&#x6570;&#x7684;&#x65B9;&#x5F0F;&#x83B7;&#x53D6;&#x7ED3;&#x679C;
        ...mapState({lesson2:state=>state.lesson})
    }
}

// &#x6A21;&#x5757;&#x4E2D;&#x7684;&#x72B6;&#x6001;
let teacher = {
    namespaced:true,
    state:{
        name:'&#x59DC;&#x6587;'
    }
}
export default teacher;
//
&#x53D6;&#x503C;&#x65F6;&#x9700;&#x8981;&#x901A;&#x8FC7;&#x6A21;&#x5757;&#x7684;&#x540D;&#x5B57;&#x6765;&#x83B7;&#x53D6;&#x5BF9;&#x5E94;&#x7684;&#x72B6;&#x6001;
<template>
    <div>
    teacherName: {{name}}
    </div>
</template>

<script>
import {mapState} from 'vuex';
export default {
   computed: {
       ...mapState({name:state=>state.teacher.name}),
       // 当模块中指定了namespaced:true时 可以简单写一个参数来指定需要的模块
       ...mapState('teacher',['name']),
       // state里的是teacher中的状态
       ...mapState('teacher',{teacherName:state=>state.name})
   }
}
</script>

// &#x4F7F;&#x7528;createNamespacedHelpers
import {createNamespacedHelpers} from 'vuex';
// &#x901A;&#x8FC7;createNamespacedHelpers &#x65B9;&#x6CD5;&#x6765;&#x83B7;&#x53D6;&#x5BF9;&#x5E94;&#x7684;mapstate
let {mapState} = createNamespacedHelpers('teacher');
export default {
   computed: {
    ...mapState(['name'])
   }
}
```

默认模块中的状态都是挂在在对应的模块内，并没有直接放到根状态上。像后面的getters/mutations/actions默认都会合并在根模块中

- getters类比组件中的计算属性

```
import {mapState,mapGetters} from 'vuex';
<div>
  {{getLessonName}}
</div>
export default {
  computed: {
      // getName(){
      //     return this.$store.getters.getLessonName
      // }
      ...mapGetters(['getLessonName'])
  }
}
// &#x5982;&#x679C;&#x6A21;&#x5757;&#x4E2D;&#x6709;namespaced:true
<template>
  <div>
  teacherName: {{getTeacherName}}
  </div>
</template>
<script>
import {createNamespacedHelpers} from 'vuex';
let {mapState,mapGetters} = createNamespacedHelpers('lesson');
export default {
 computed: {
  ...mapGetters(['getTeacherName'])
 }
}
</script>
```

把模块中的状态进行计算，映射出对应的结果

mutations & actions

- mutation突变，唯一修改状态的方式

```
strict:process.env.NODE_ENV !== 'production' //
&#x4E25;&#x683C;&#x6A21;&#x5F0F;&#x4FEE;&#x6539;&#x72B6;&#x6001;&#x53EA;&#x80FD;&#x901A;&#x8FC7;mutation&#x6765;&#x4FEE;&#x6539;

let mutations = {
    SET_LESSON_NAME(state,payload){ // &#x8F7D;&#x8377;
        state.lesson = payload;
        // &#x4FEE;&#x6539;&#x65F6;&#x9700;&#x8981;&#x83B7;&#x53D6;&#x5BF9;&#x5E94;&#x7684;&#x5C5E;&#x6027;
        // state.lesson = payload.name;
    }
}
export default mutations;
// &#x8F7D;&#x5728;&#x7EC4;&#x4EF6;&#x4E2D;&#x8C03;&#x7528;commit&#x65B9;&#x6CD5; &#x89E6;&#x53D1;mutation&#x5BF9;&#x5E94;&#x7684;&#x65B9;&#x6CD5;
changeName(){
    this.$store.commit('SET_LESSON_NAME','node')
    // &#x53EF;&#x4EE5;&#x5199;&#x6210;&#x5BF9;&#x8C61;&#x7684;&#x65B9;&#x5F0F;&#x4F20;&#x9012;
    // this.$store.commit({
    //     type:'SET_LESSON_NAME',
    //     name:'node'
    // });
}
```

给状态新增不存在的属性，需要通过Vue.set方法

```
Vue.set(state,'number',payload.number);
```

```
import {mapState,mapGetters,mapMutations} from 'vuex';
methods: {
    ...mapMutations(['SET_LESSON_NAME']),
    changeName(){
        this['SET_LESSON_NAME']({number:10});
    }
}
```

mutation不能操作异步逻辑

- actions就是用来处理异步的请求，异步更新状态

```
// &#x6D3E;&#x53D1;&#x52A8;&#x4F5C;&#x5230;action&#x4E2D;
this.$store.dispatch('SET_LESSON_NAME');
// &#x5728;action&#x4E2D;&#x5904;&#x7406;&#x5F02;&#x6B65;&#x903B;&#x8F91;&#x540E;&#x5C06;&#x7ED3;&#x679C;&#x63D0;&#x4EA4;&#x7ED9;mutation
import {getLesson} from  '../api/lesson'
let actions = {
    //
&#x5728;action&#x4E2D;&#x9700;&#x8981;&#x5C06;&#x6570;&#x636E;&#x63D0;&#x4EA4;&#x7ED9;mutation&#xFF0C;&#x8FD9;&#x91CC;&#x53EF;&#x4EE5;&#x505A;&#x5F02;&#x6B65;&#x
x903B;&#x8F91;
    SET_LESSON_NAME({commit},payload){
        getLesson().then(data=>{
            // data => {name:node}
            commit({type:'SET_LESSON_NAME',...data});
        })
    }
}
export default actions;

// &#x4F7F;&#x7528;mapActions
methods: {
    ...mapActions(['SET_LESSON_NAME']),
    changeName(){
        this['SET_LESSON_NAME']();
    }
}
```

action中可以做封装异步请求，多个组件相同的异步处理，可以直接调用action,action中可以多次commit,也可以在action中再次调用action

vuex进阶

自动保存到本地插件

```
// vuex&#x4E2D;&#x7684;store&#x5BB9;&#x5668;
// vuex&#x4E2D;&#x7684;store&#x5BB9;&#x5668;
export default (store)=>{
    // &#x7528;&#x65B0;&#x7684;&#x72B6;&#x6001; &#x66FF;&#x6362;&#x6389;&#x8001;&#x7684;&#x72B6;&#x6001;
    store.replaceState(JSON.parse(localStorage.getItem('state'))|| store.state);
    store.subscribe((mutation,state)=>{ // &#x8BA2;&#x9605;&#x6BCF;&#x6B21;commit&#x90FD;&#x4F1A;&#x89E6;&#x53D1;&#x6B64;&#x51FD;&#x6570;
        localStorage.setItem('state',JSON.stringify(state));
    });
}

// &#x4F7F;&#x7528;&#x63D2;&#x4EF6;
import saveLocale from  './plugins/saveLocale'
export default new Vuex.Store({
  state,
  mutations,
  getters,
  actions,
  strict:process.env.NODE_ENV !== 'production',
  modules:{
    teacher
  },
  plugins:[saveLocale]
});
```

logger插件(vuex中自己实现了这个插件)

```
import deepClone from 'lodash/cloneDeep'
export default (store)=>{
    let prevState = deepClone(store.state);
    store.subscribe((mutation,state)=>{
        console.log('prev',prevState.lesson);
        console.log(mutation);
        console.log('next',state.lesson);
        prevState = deepClone(state);
    });
}
```

vuex双向绑定,当更新数据时手动提交数据的更改

```
<input type="text" v-model="teacherName">
computed: {
    ...mapState('teacher',['name']),
    teacherName:{
        get(){
            return this.name;
        },
        set(val){
            this['SET_TEACHER_NAME'](val);
        }
    }
}
```