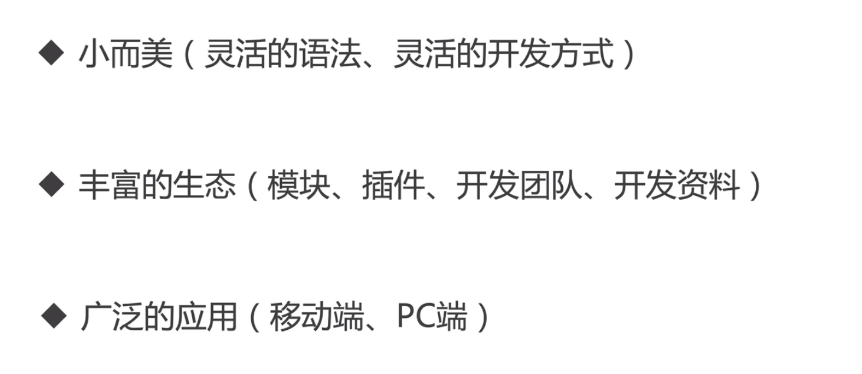
3小时速成 Vue2.x 核心技术





install nvm     //node 版本管理工具

linux 安装命令：(可以去github 官网查看安装)

curl -o- [https://xxx](https://xxx/)

使用淘宝的镜像安装 cnpm

npm install -g cnpm --registry=[https://registry.npm.taobao.org](https://registry.npm.taobao.org/)

npm --version查看npm版本

node --version 查看node.js版本

nvm：node多版本管理的最佳工具 creationix/nvm

nvm --version查看nvm版本

nvm --help查看帮助

nvm ls查看本地看装的node版本和当前使用的版本

nvm ls-remote查看远端所有的node版本

nvm install v11.0.0安装此版本的node并默认使用它

nvm use v8.12.0切换node版本为此版本

npm淘宝镜像的使用 cnpm

chrome的vue插件的使用

安装vue工程工具vue-cli：npm i  -g vue-cli

查看vue-cli版本：vue --version

安装cnpm:

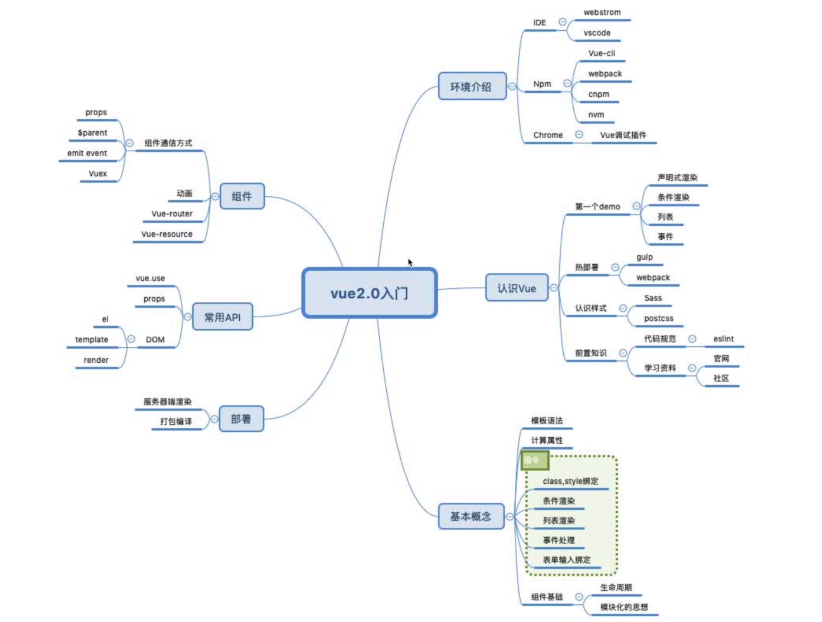
npm install -g cnpm --registry=[https://registry.npm.taobao.org](https://registry.npm.taobao.org/)

​

cnpm install

最新谷歌 vue控件安装

https://chrome.zzzmh.cn/info?token=nhdogjmejiglipccpnnnanhbledajbpd



使用CDN 引入vue.js

    对应网址： bootcdn

vue 中 el 标签：

可以绑定到 id 属性 ： 格式： el : "#xxx"

可以绑定到 class属性 格式： el : ".xxx"

如果页面上出现多个class，则优先绑定第一个class元素。 因此vue绑定元素时，尽量使用id的方式绑定。

插值表达式：{{msg}}

v-bind:绑定属性，缩写为冒号（:）

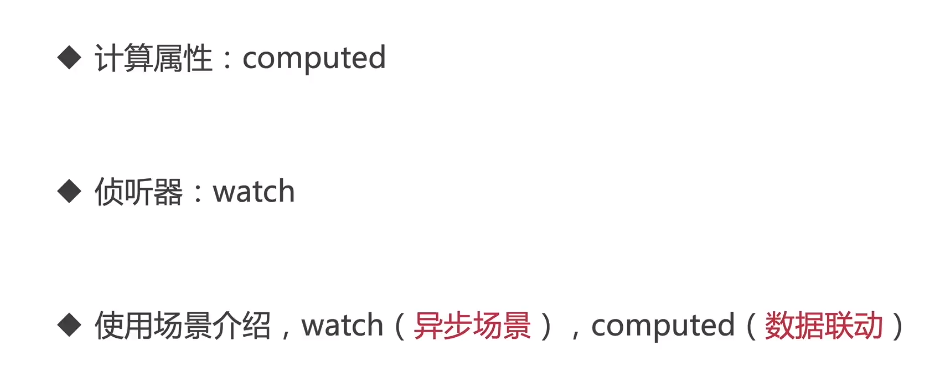
v-on:绑定事件，缩写为：@

创建一个Vue实例：

new Vue({})

el标签： 指定挂载点

v-if: 条件判断



计算属性：computed

侦听器：watch

使用场景，watch（异步场景），computed（数据联动）

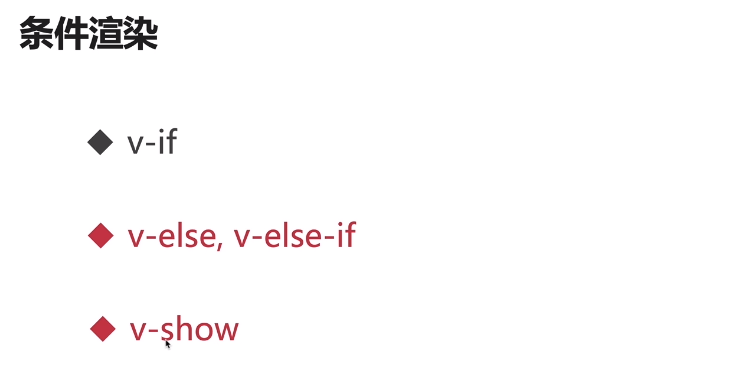
watch监听的是一个单一的变量，或是一个数组，

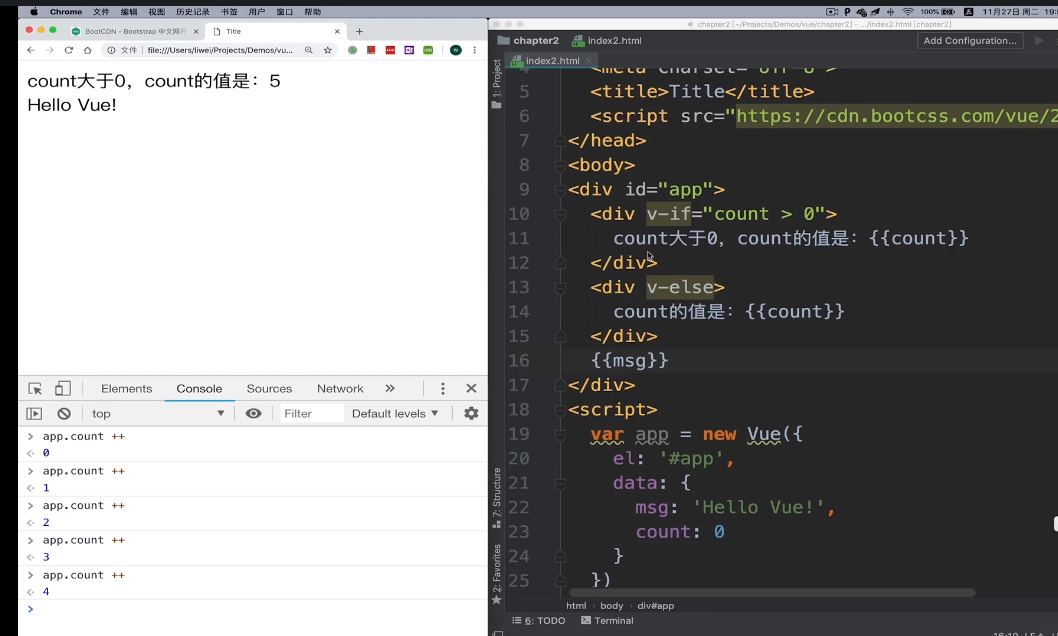
computed它可以监听很多个变量，但是这个变量一定是在vue实例里面的数据

条件渲染：v-if v-else v-else-if，v-show

列表渲染：v-for <div v-for="item in list">

style和class 的绑定：style可写成对象形式，然后通过属性绑定v-bind进行绑定；class绑定写成对象形式，键名是真是的class名称，值为布尔值或表达式，或者可以使用数组直接输出字符串，也可以用对象表达式+数组混合的方式





<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Vue条件渲染</title>

<script type="text/javascript" src="./vue.js"></script>

</head>

<body>

<div id="app">

<div v-if="count > 0">

count 大于0， 当前count的值是：{{count}}

</div>

<div v-else-if="count < 0 && count > -5">

count 介于0和-5之间

</div>

<div v-else>

count的值是：{{count}}

</div>

<br>

<div v-show="count == -1">show: {{count}}</div>

{{msg}}

</div>

<script type="text/javascript">

var app = new Vue({

el: "#app",

data: {

msg: "Hello Vue.js",

count: 10

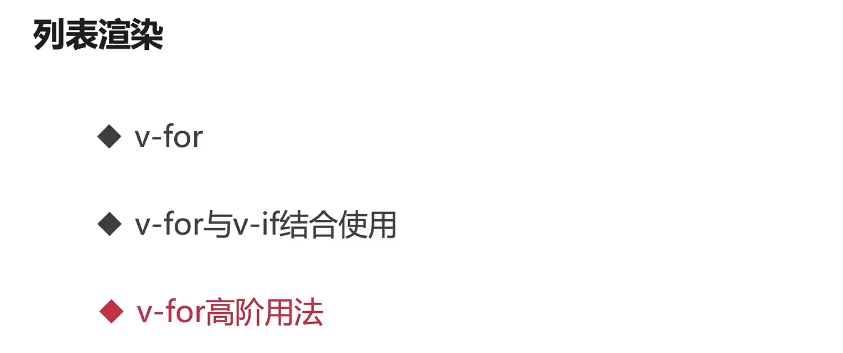
}

})

</script>

</body>

</html>



<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>列表渲染</title>

<script type="text/javascript" src="./vue.js"></script>

</head>

<body>

<div id="app">

<div v-for="item in list">{{item}}</div>

<hr color="red">

<div v-for="item1 in userList" v-bind:class="['active', 'add', 'more', {'another':item1.age < 30}]">

{{item1.name}} -- {{item1.age}}

</div>

<hr color="red">

<div v-for="item2 in userList" v-bind:class="{'active': true}">

<div v-if="item2.age > 30">

<div v-if="item2.sex == '男'">

{{item2.name}}， 你老了，你个老男人

</div>

<div v-else-if="item2.sex == '女'">

{{item2.name}}，你老了，你个老阿姨

</div>

</div>

<div v-else>

{{item2.name}},{{item2.age}}岁，好年轻

</div>

</div>

<hr color="red">

<div v-for="item3 in userList">

<div v-show="item3.age < 30" v-bind:style="style">

{{item3.name}}-{{item3.age}}-{{item3.sex}}

</div>

</div>

</div>

<script type="text/javascript">

new Vue({

el: "#app",

data: {

list: [1,2,3,4,5,6,7,8,9],

userList: [{

name: "刘豆豆",

age: 23,

sex: "男"

},{

name: "小三",

age: 18,

sex: "女"

},{

name: "老王",

age: 40,

sex: "男"

},{

name: "花花",

age: 35,

sex: "女"

}],

style: {

color: 'red',

textShadow: '0 0 5px green'

}

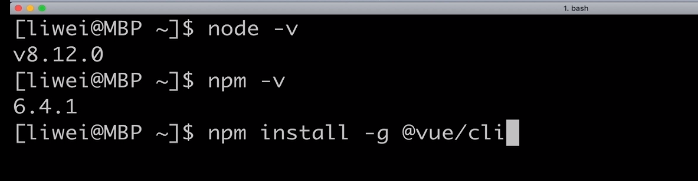
}

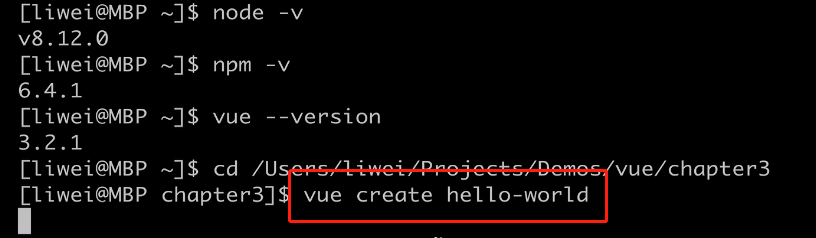
})

</script>

</body>

</html>







查询node版本：node -v

查询npm版本：npm -v

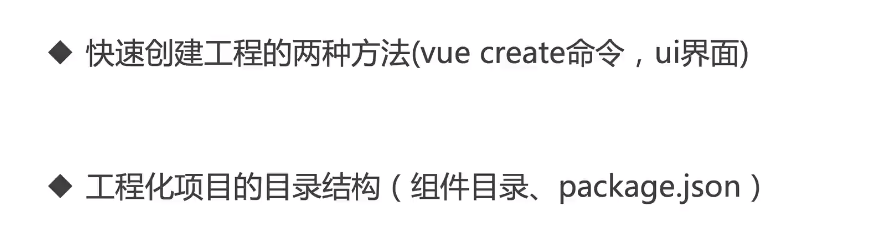
全局安装vue-cli工具：npm install -g @vue/cli

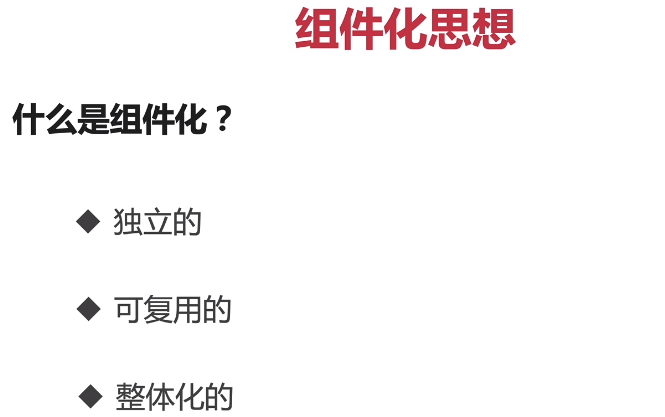
查询vue-cli版本：vue --version

进入项目目录，创建项目：vue create 项目名

启动运行项目：npm run serve

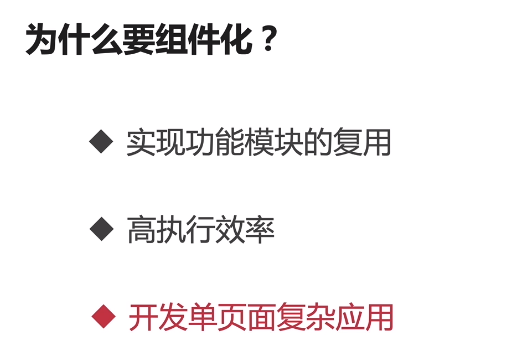
​





组件的三个特点：独立性，可复用性，完整性

解决组件化带来的问题的方式：vuex进行组件间的状态管理，vue-router组件间的路由管理；还有组件间的传参、消息、事件管理



为什么要组件化：

1. 实现功能模块的复用
2. 高效执行
3. 开发单页面复杂应用

如何进行拆分：

1.300行原则

2.复用原则

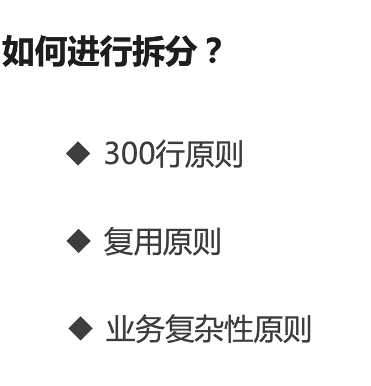
3.业务复杂性原则

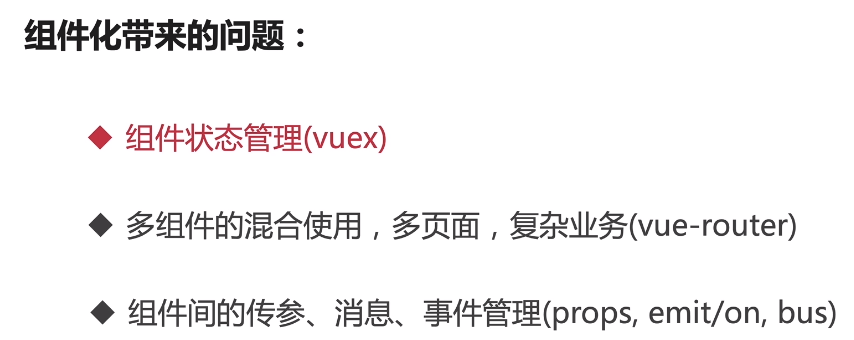
组件化问题：

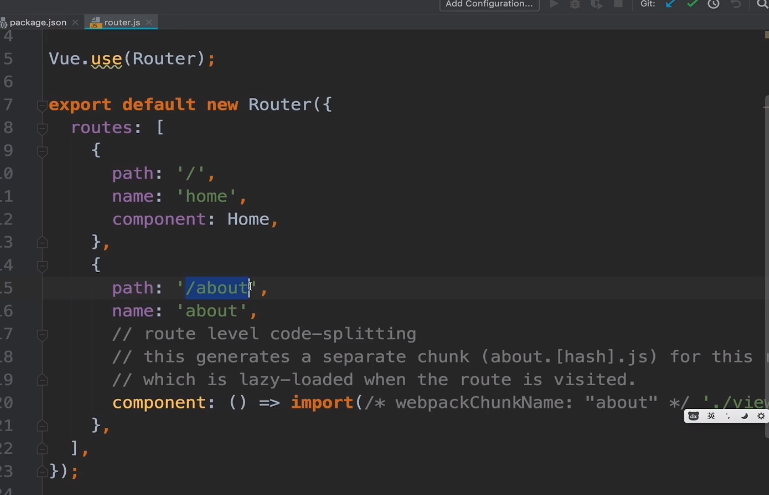
1、组件状态管理（vuex）

2、多组件的混合使用、多页面复杂业务（vue-router）

3、组件之间的传参、消息、事件管理（props,emit/on,bus）







routes对应的路由配置包含：

path为路由路径'/'为根目录；

name为路由的名称;

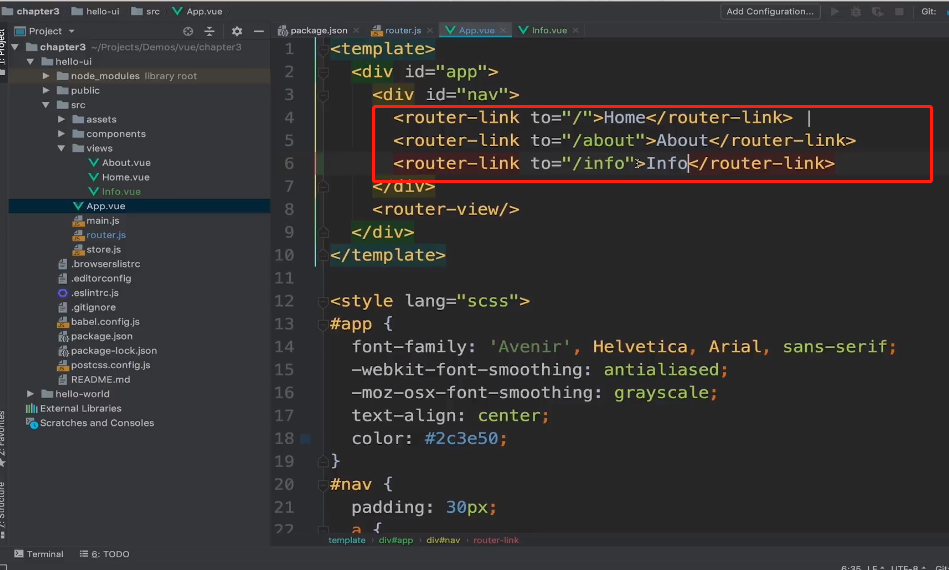
component对应的是引入的具体组件

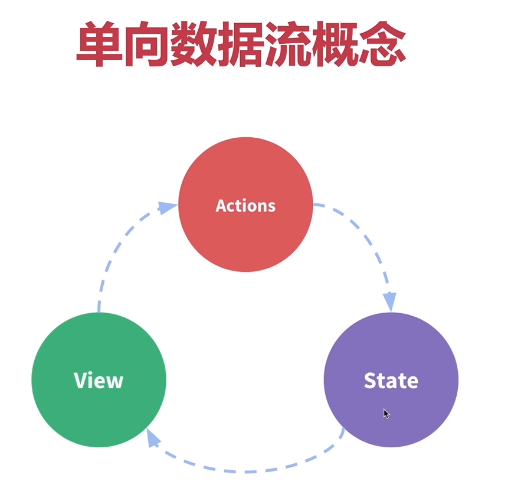
app.vue文件为项目界面结构，其中<router-link to="/">title</router-link>为路由切换标签，<router-view/>则显示当前的路由组件视图；

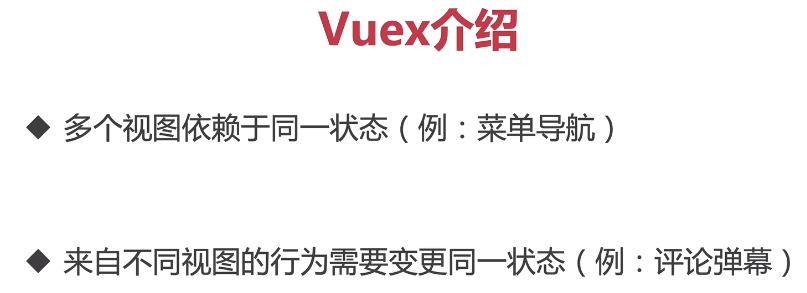
在src目录下的views下新建新的xxx.vue组件；

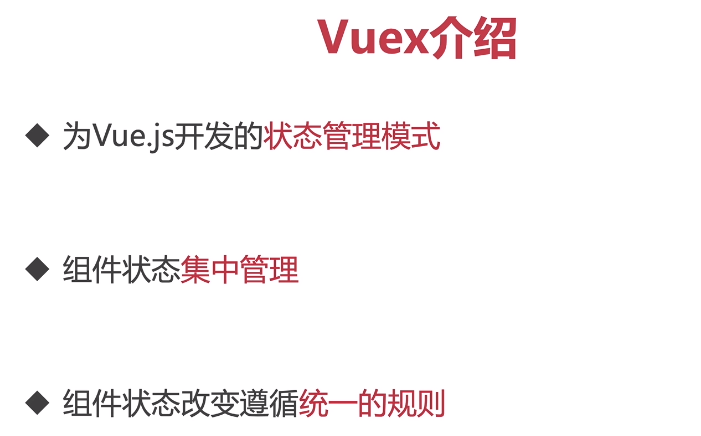
在router.js文件中添加新的xxx.vue组件信息；

在App.vue中，通过<router-link>标签来添加新的组件的link地址；

​







关于vuex

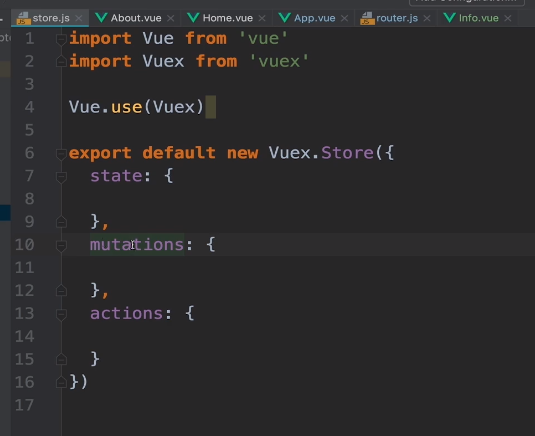
Vuex是为Vue.js开发的状态管理模式

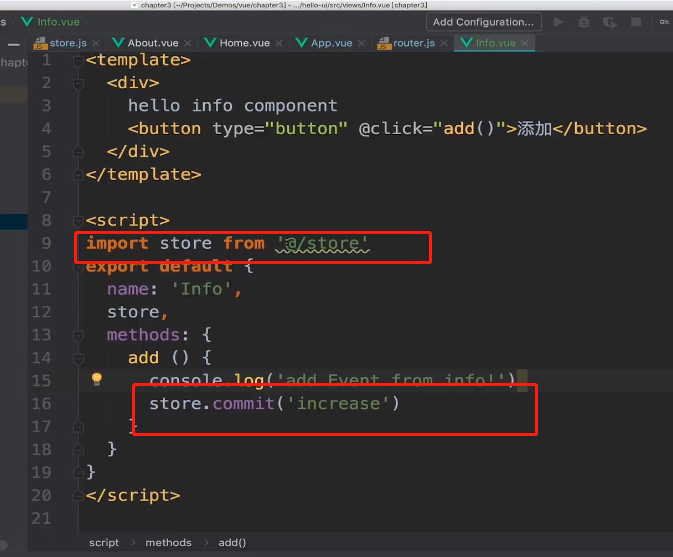
页面由视图组成，用户操作会带来状态的变化，状态的变化会导致视图的更新

1. 多个视图依赖于同一状态（例如菜单导航）
2. 来自不同视图的行为需要变更同一状态（例如评论弹幕）

---Vuex是为Vue.js开发的状态管理模式

组件状态集中管理，组件状态改变遵循统一规则





store.js中引入了vue和vuex；然后Vue.use(Vuex)

state中记录的是组件状态，进行集中管理；

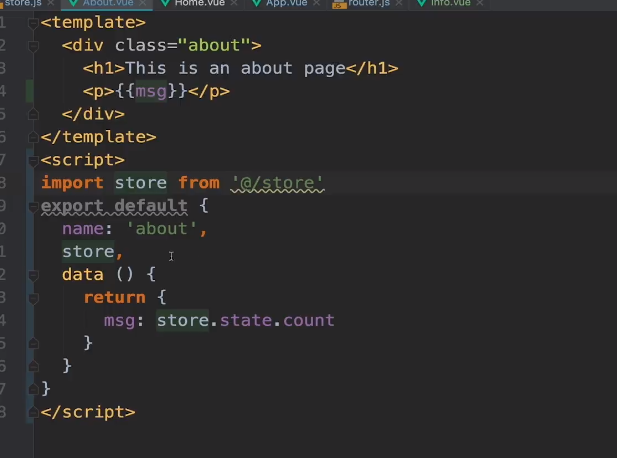
mutations是唯一可以改变集中状态的方法集的定义；

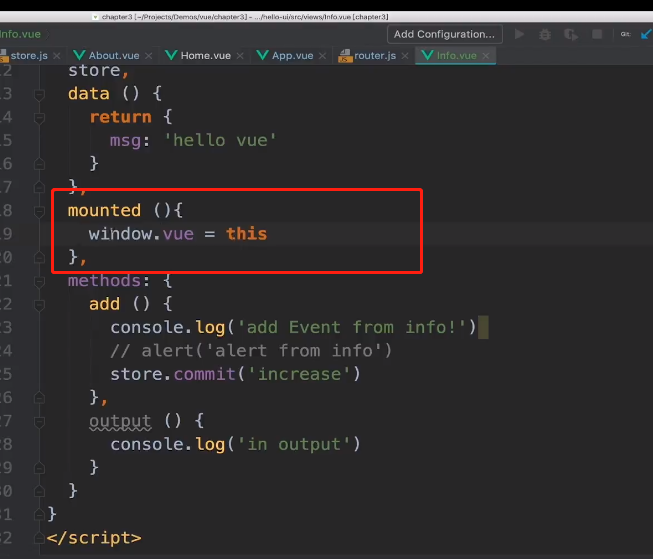
在页面xxx.vue文件中使用vuex时，需要在<scrip></script>中引入vuex，引入的方法：import store from '@/store'

在传递状态的视图组件中import store from '@/store';//@符号代表src目录

传递过去的状态名称在state中定义；在mutations中定义一个方法，对状态进行改变；在触发状态改变的视图组件中使用store.commit('mutation')进行状态改变触发

在需要接收状态改变的视图组件中先import store from '@/store';然后使用store.state.name可以直接引用被更改了的状态

​



vue调试

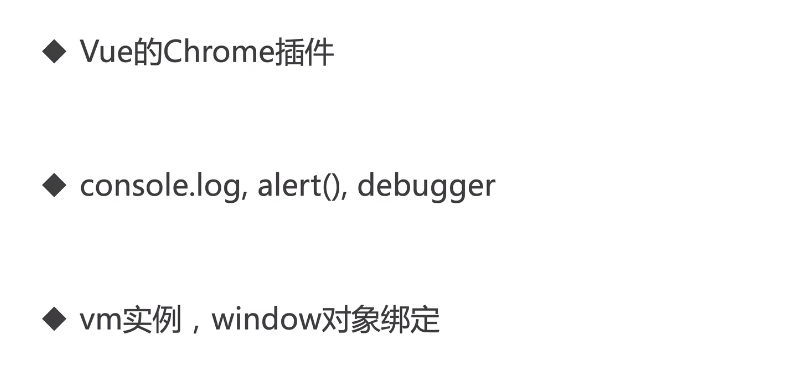
1、console.log、console.error 、alter（）

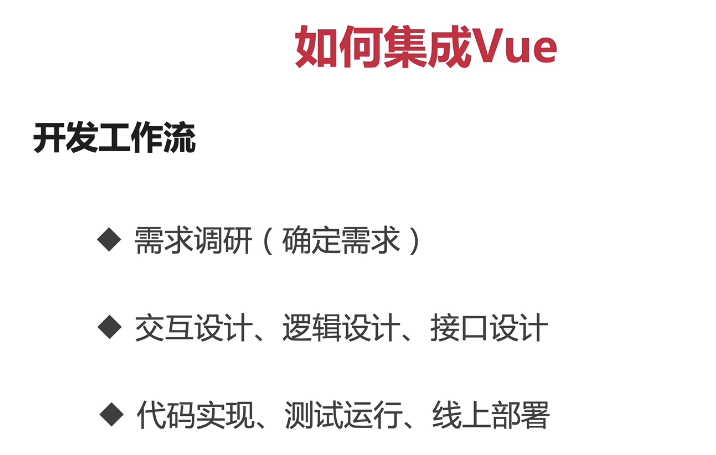
2、debugger（断点调试），当处于断点状态时，可以直接在命令行中对当前组件的属性进行调试。

在使用断点调试时，只需要在需要调试的代码处添加debugger关键词就可以了

3、在vm实例中的mounted(){}中，绑定window.vue=this;

例如调试output()。 在控制台输入window.vue.output()

​



如何集成Vue?

1. 单页面，多页面引入Vue.js
2. 复杂单页面应用Vue cli工具

开发工作流：

1. 需求调研（确定需求）
2. 交互设计、逻辑设计、接口设计
3. 代码实现、测试运行、线上部署

git简介：

1. 创建项目git clone，git init
2. 创建分支，推送分支，合并分支
3. 删除分支、回退版本

git --version  : 查看git的版本

git clone 项目地址.git   : 克隆下载代码到本地

git status：查看项目的状态信息

git branch -a：查看项目的所有分支信息

touch test.txt：创建文件

git add . ：把所有本地文件添加到待处理待提交的状态

git commit -m “提交注释”:添加本次提交的注释信息

git remote -v：查看推送到的地方，也就是你将会推送到的远程仓库地址

git push origin master : 推送到远程主干分支上

git checkout -b 新的分支： 新建并切换到新的开发分支上

git push origin dev : 推送到开发分支dev上

git branck master: 切换到master分支上

git merge dev: 将开发分支dev代码合并到master上，在合并之前要切换到master分支下

git branch -D dev: 删除开发分支dev

git push origin :dev:  删除远程dev分支。本地开发分支dev删除之后，需要把远程仓库的dev也要删除

git reset --hard head^ : 回退到上一个版本

git reset --hard head^^ : 回退到上上一个版本

git log/reflog: 查看git提交日志

git reset --hard git提交id : 回退到指定的版本

​