# 饿了么项目开发手册

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 版本 | 说明 | 作者 |
| 2018/7 | 1.0 | 初稿 | 龚宇森 |
| 2018/12 | 1.1 | 添加地理定位 | 龚宇森 |
| 2019/6 | 1.2 | 增加打包步骤 | 龚宇森 |

## 项目搭建

安装Vue.js命令行工具：

全局安装 vue-cli

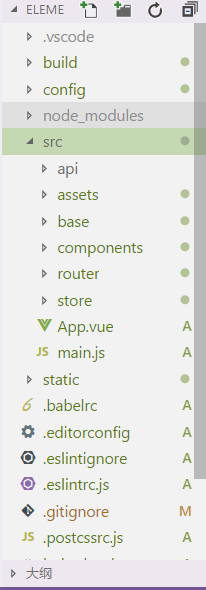
npm install -g vue-cli

创建项目

vue init webpack eleme

运行  
npm run dev

## 二．项目结构目录



**目录解释**：   
build文件夹———webpack的基本配置、开发环境配置、生产环境配置等   
config—————-配置了路径端口值等   
node\_modules—–依赖的模块   
src文件夹————项目源文件

api 文件夹 -------------- 数据请求  
assets文件夹 ——-存放bootstrap等外部插件   
 css————–公共的css文件：reset.css、fontstyle.css   
 font————–字体图标文件   
 js—————–公共的js文件

base------------------公共组件  
components———各组件文件.vue

router-----------------路由文件

store------------------Vuex

static --------------------静态文件

main.js—————入口文件，作用是初始化vue实例并使用需要的插件   
App.vue————–主组件，所有页面都是在App.vue下进行切换的。其实也可以理解为所有的路由也是App.vue的子组件。  
index.html———–文件入口

|  |
| --- |
| **assets和static文件夹的区别** assets目录中的文件会被webpack处理解析为模块依赖，只支持相对路径形式  static目录的文件不会被Webpack处理：它们会直接被复制到最终的打包目录  必须使用绝对路径引用  **static放不会变动的文件 assets放可能会变动的文件** |

## 三．组件结构

说明：  
项目只有一个页面index.html，引用了根组件App.vue，根组件下分[Home.vue和Seller.vue](http://Home.vue和Seller.vue)组件，其中Home即首页，打开首页底部菜单对应的四个页面分别对应了find、mine、order和take四个子组件。Seller.vue是商家页面，该页面底部没有菜单，所以单独做成一个组件，该页面由商品列表、评价列表、购物车等子组件构成。

## 第三方库和模块安装

需要使用的第三方静态资源文件

static目录是通过Web请求获取的静态资源文件，不会被编译打包到组件内。  
在这里放入JSON数据文件

index.html页面：

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <meta charset="utf-8">  <meta name="description" content="饿了么是中国专业的网上订餐平台，目前已覆盖北京、上海、杭州、广州等300多个城市，提供各类中式、日式、韩式、西式、下午茶、夜宵等优质美食，并提供送餐上门服务，让订餐更加轻松，叫外卖就上饿了么！">  <meta name="keywords" content="饿了么，网上订餐，外卖，快餐，外卖网">  <!-- 是否启动webapp功能 设置为yes 网站就会在满屏模式full-screen mode删除默认的苹果工具栏和菜单栏 -->  <meta content="yes" name="apple-mobile-web-app-capable" />  <!-- 开启对web app程序的支持 -->  <meta content="yes" name="apple-touch-fullscreen" />  <!-- 忽略识别手机号码和邮箱 -->  <meta content="telephone=no,email=no" name="format-detection">  <title>饿了么</title>  <!-- 饿了么专用小图标 -->  <link rel="stylesheet" href="//at.alicdn.com/t/font\_3yrixiqo86si3sor.css">  </head>  <body>  <div id="app"></div>  </body>  </html> |

需要安装的第三方模块：

npm install lib-flexible 淘宝团队处理移动端分辨率自适应的解决方案插件  
npm install axios --save 异步请求模块  
cnpm install vue-awesome-swiper –-save swiper轮播图组件  
cnpm install node-sass –-save-dev 依赖模块  
cnpm install sass-loader –save-dev 预处理模块

cnpm install fastclick --处理移动端单击300毫秒延迟问题

cnpm install qs --axios模块的POST数据请求插件

npm i vuex --save

npm install --save-dev less-loader less less模块安装

## 五．项目入口文件main.js

import fastclick from 'fastclick'

import Vue from 'vue'

import App from './App'

import router from './router'

import './assets/css/index.css'

import store from './store'

import axios from 'axios'

Import qs from 'qs'

import 'lib-flexible'

Vue.config.productionTip = false

fastclick.attach(document.body)// body内所有click 300毫秒延迟都将去掉

Vue.prototype.$http = axios

Vue.prototype.$qs = qs

new Vue({

el: '#app',

router,

components: { App },

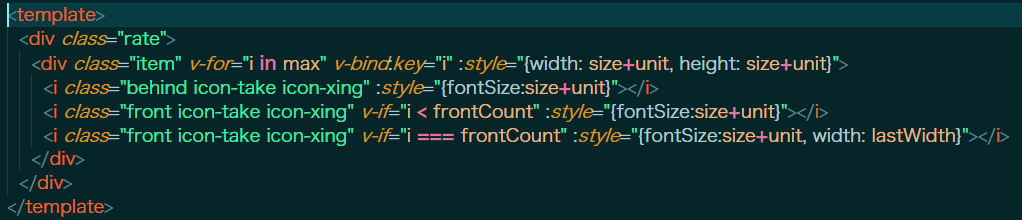
template: '<App/>',

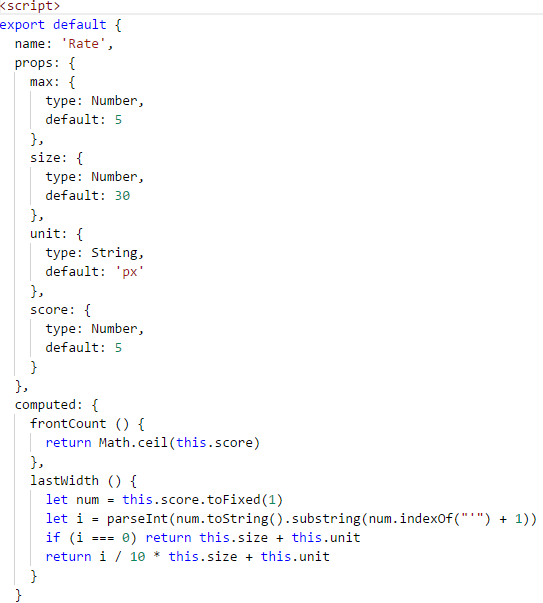
store

})

|  |
| --- |
| 图片可以使用JS进行导入：require和import均可,因为assets文件夹中的文件会被webpack解析，所以使用required。而static目录是不会被webpack处理的，最终会原样输出在dist目录中  data(){  return {  imgUrl: require('../assets/img'),  imgUrl:'static/img'  }  }  <img :src="imgUrl" /> |

## 六．Rate星星组件(base目录下)





1.**实现原理**：用两个图标字体星星绝对定位重叠，改变上面的星星的背景颜色以及宽度实现。

2.使用这个组件需要传递props，max星星的最大数量，size每个星星的大小，unit使用的单位，默认为px。score分数，根据这个分数会自动计算星星如何显示。

3.lastWidth用于计算最后一颗星星显示多少宽度，

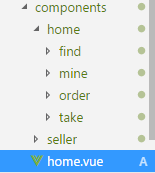
截取出小数点后的小数,如果是0，就显示一整颗，否则显示一部分

## 项目根组件App.vue



App.vue是项目的根组件，这个组件相当于一个容器，其他页面组件都是这个组件的子组件。所以，这个组件中的模板就只有一个路由标签

## Home组件



home组件的内容是底部的导航菜单，以及上面的内容部分。内容部分由四个子组件确定。这个四个子组件分别对应菜单的四个对应的页面，即这里的find（发现）、mine（我的）、order（我的）以及take（外卖）

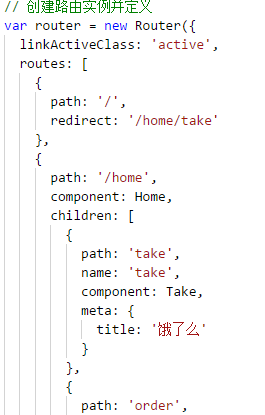


router-view标签是vue-router路由标签，会把路由对应的组件替换到这个标签的位置。  
router-link标签是用于路由跳转的标签，最终渲染出来时会替换成a标签。

注：

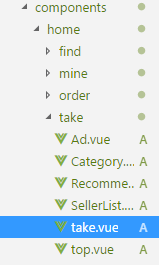
## Home组件的路由设置

在src下的router文件夹中的index.js文件是vue-router路由配置代码  


Vue.use(Router) 是Vue.js使用插件的方式，这里表示应用vue-router。  
路由的具体配置如下：  


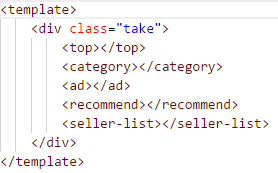
路由的配置就是创建一个Router对象。  
其中linkActiveClass属性是一个类名，表示的是当路由被激活的时候应用的是哪一个样式。  
routes属性是一个数组，数组中的每一个元素都是一个对象，代表一个路由，每一个路由对象中包含属性：  
path：路由地址  
name：路由名称  
redirect：重定向  
component：路由对应的组件名称，即该路由激活时router-view被替换的组件  
children：子路由

## 十．Take 外卖组件

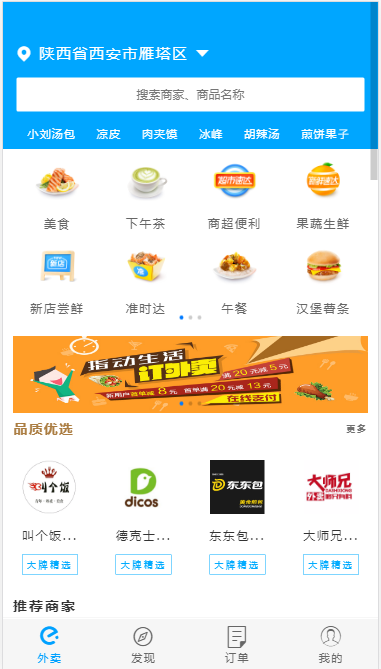


这一部分就是Take.vue组件，即外卖页面，components下是子组件

take.vue文件关键代码：



**这个页面拆分成了若干个子组件，分别对应的是components文件夹下的几个文件如下：**

**SellerList.vue**

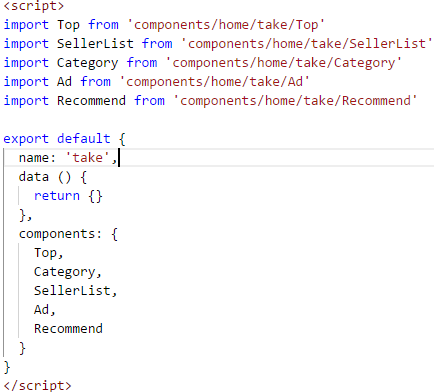
Top.vue

Category.vue

Ad.vue

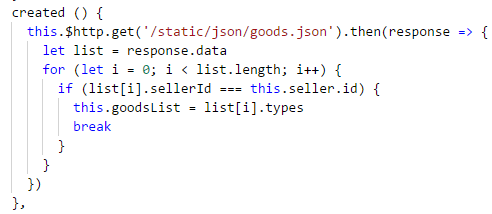
Recommend.vue

**在JS代码部分，主要就是导入相关组件并定义为子组件**

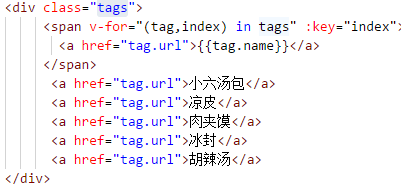


Top.vue 子组件



这部分中，下面的热卖标签是从后台数据接口获取的，代码如下：  


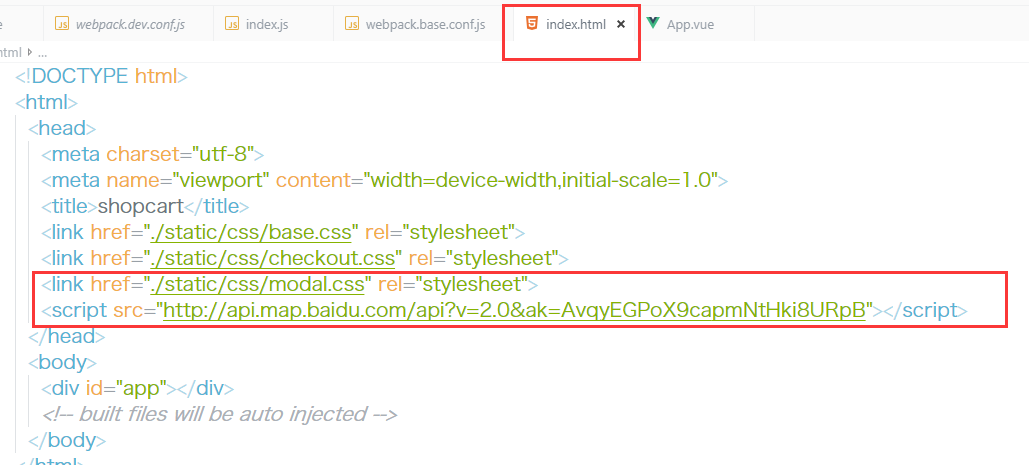
在created生命周期通过异步请求后台数据接口。



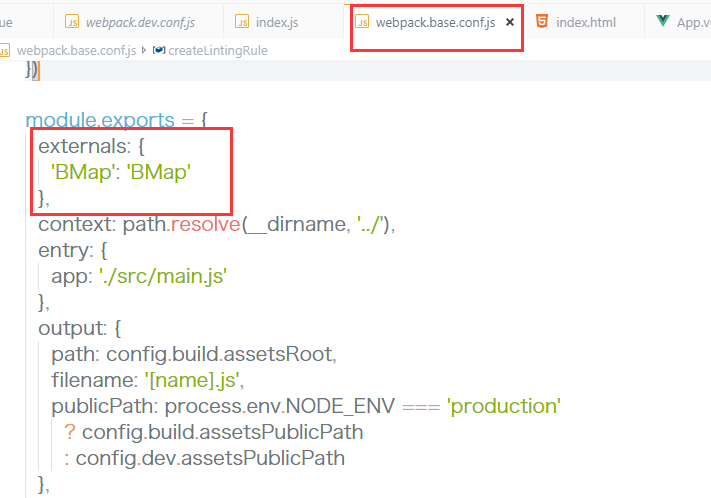
通过v-for指令遍历获取到的数据生成DOM结构。

**可使用百度接口进行定位**

1. **在index.html中引入百度JS地址**



1. **修改webpack.base.config.js文件，添加引用**



1. **编写Geo独立公共组件**



**4.在 其他容器中引入该组件即可**



<Geo></Geo>

Category.vue组件

  
**在created生命周期中使用axios的异步请求方法请求后台数据接口获取数据**。

**<swiper :*options*="swiperOption" *class*="swiper">**

**<swiper-slide *v-for*="(cates,index) in cate" *class*="cl" :*key*="index">**

**<a *class*="item" *v-for*="(cate,i) in cates" :*href*="cate.url" :*key*="i">**

**<img :*src*="cate.icon" *alt*="">**

**<p>{{cate.name}}</p>**

**</a>**

**</swiper-slide>**

**<div *class*="swiper-pagination" *slot*="pagination"></div>**

**</swiper>**

**<script>**

**import { swiper, swiperSlide } from 'vue-awesome-swiper'**

**import 'swiper/dist/css/swiper.css'**

**data () {**

**return {**

**cate: [],**

**swiperOption: {**

**direction: 'horizontal',**

**pagination: {**

**el: '.swiper-pagination',**

**type: 'bullets'**

**}**

**}**

**}**

**},**

**components: {**

**swiper,**

**swiperSlide**

**}**

**<style>**

**.category {**

**margin-top: 0.2415rem;**

**.swiper {**

**margin: 0 0.2657rem;**

**height: 4.72rem;**

**.item {**

**float: left;**

**width: 20%;**

**text-align: center;**

**img {**

**width: 1.1916rem;**

**height: 1.1916rem;**

**}**

**p {**

**color: #666;**

**}**

**}**

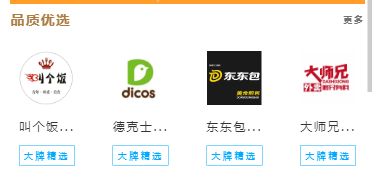
**}**

**}**

**</style>**

组件的模板部分，主要是通过v-for指令来遍 历获取的数据生成DOM结构。

Ad.vue组件

  
这个广告轮播图的实现类似分类组件，通过异步请求后台的数据接口以及轮播图插件来实现。  
Recommend.vue组件  
  
通过异步数据接口获取数据。

SellerList.vue组件（商家列表）  
  
这个列表数据通过异步数据接口获取

**使用了公共组件Rate评价组件**



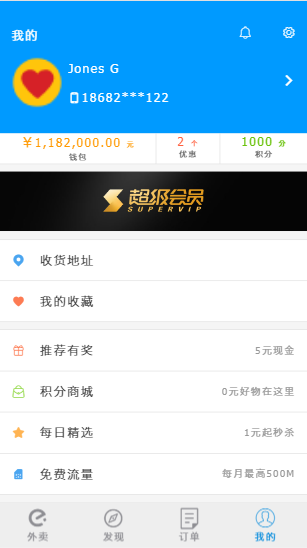
1. 一个无序列表，每个li项目就是一个商家。通过v-for指令来遍历异步获取的数据生成DOM结构。这里使用了rate组件，其中给的props是这个组件必须的属性。

## 十一.Find发现组件



该组件的实现方式与Take组件基本类似，通过axios进行异步数据接口的数据请求。然后在组件中通过vue的指令和数据绑定更新DOM中的数据以及数据的操作。

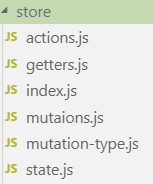
## mine我的组件



该该组件的实现方式与Take组件基本类似，通过axios进行异步数据接口的数据请求。然后在组件中通过vue的指令和数据绑定更新DOM中的数据以及数据的操作。组件中使用的图标是利用iconfont制作的

。

个人信息可以采用Vuex集中式状态存储，大致步骤如下：

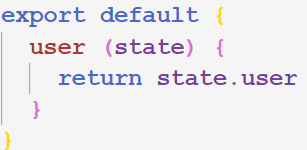
1. 搭建Vuex框架目录
2. 编写index.js入口文件



1. 在state中存储当前用户信息



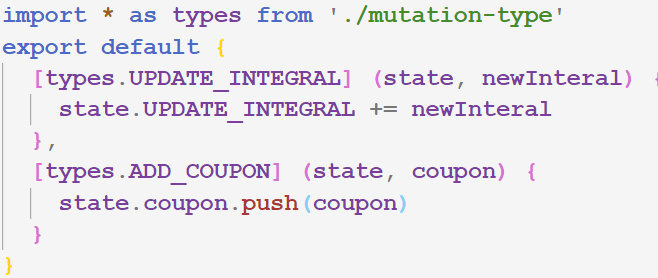
1. 添加Getters



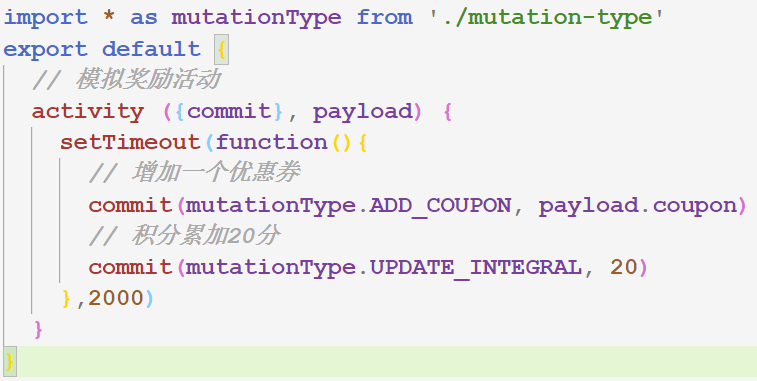
1. 添加mutation-type文件,主要作用是用常量来管理Mutaions的函数名



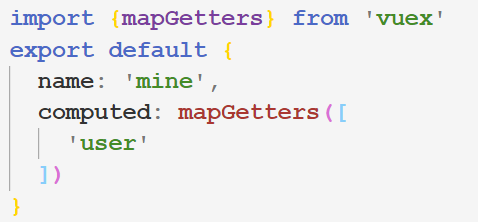
1. 添加mutations



1. 添加Actions(这里模拟出一个活动函数,让用户增加一个优惠券，积分增加)



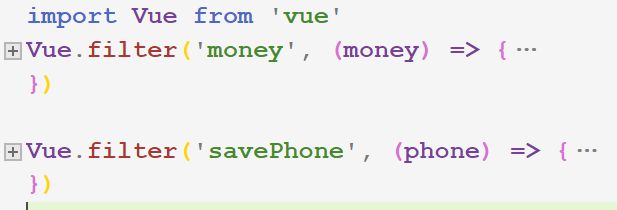
1. 在Mine.vue中得到用户信息，并展示



1. 添加两个全局过滤器（人民币格式、手机号码保护）

在base文件夹下新建filters文件夹，里面添加index.js文件

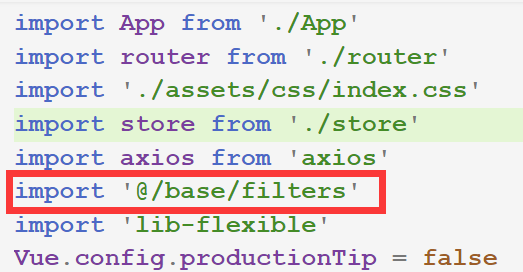
创建两个全局过滤器



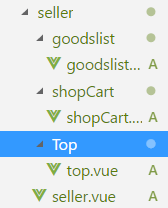
1. 在Mine组件中使用过滤器



1. 切记要在main.js中引入过滤器文件



1. Seller商家详情页组件



这个组件主要是展示商家的详情信息，以及商品列表。包含子组件：

GoodsList商品列表

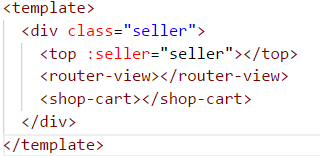
Rating评价页面

ShopCart购物车子组件

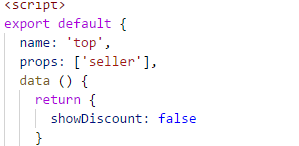
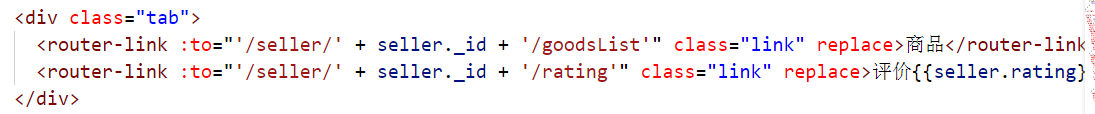
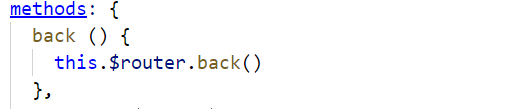
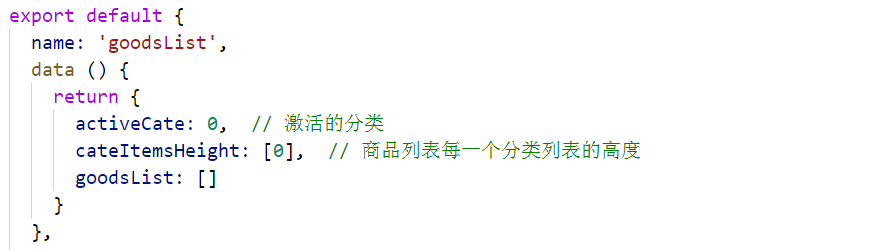
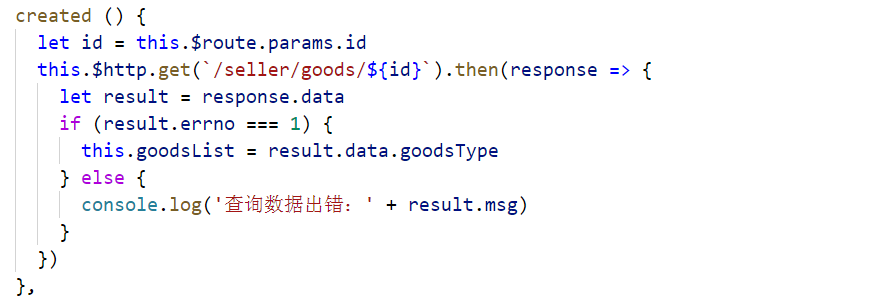
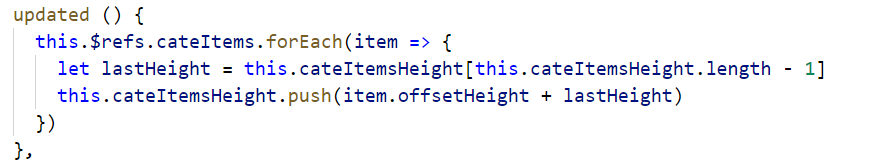
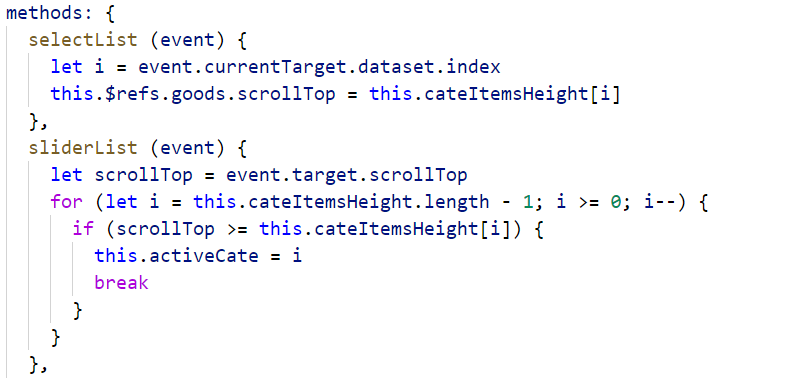
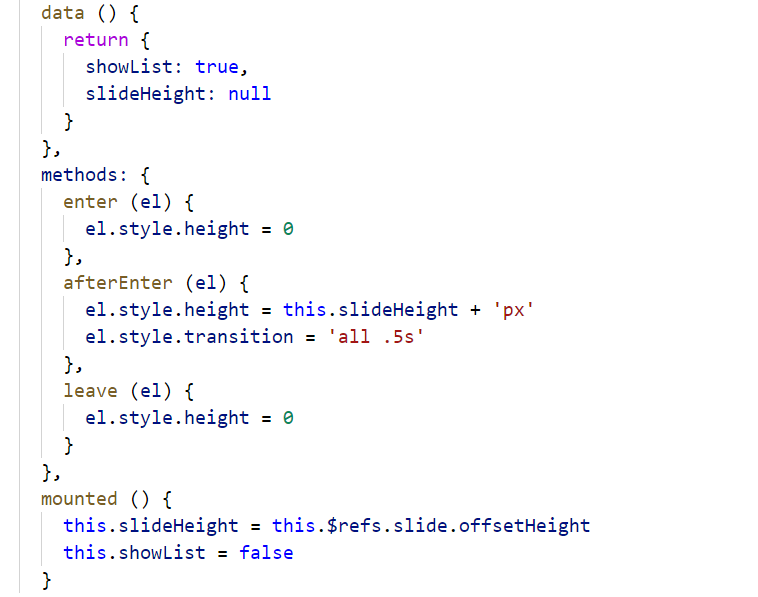
Top顶部的店名以及Logo信息

。

模板部分代码：



在created声明周期异步获取后台数据接口的商家信息数据，使用vue的数据绑定更新DOM中的数据。这个请求需要传递一个商家Id，该id在跳转到该页面的时候需要通过路由传递。  
  
路由设置：  


Top.vue子组件  
  
  
**该组件需要父组件Seller传递一个props seller，即商家的数据。**  
  
**这里是一个路由跳转，对应页面的两个tab页，商品列表和评价。其中replace属性是不在历史记录中保存这个路由地址。**  
  
  
  
GoodsList子组件  
  
这个组件中，左边的分类和右边的商品列表都是可以滑动的，这里利用了flex布局实现。  
  
定义数据属性，goodsList是从后台数据接口异步请求获取的。  
  
获取商品列表数据，通过axios的异步请求方法获取，成功之后把数据赋值给数据属性goodsList  
  
在updated生命周期中，计算右边的商品列表的每个分组的高度，用于制作分类和商品列表联动的效果。  
  
定义两个方法：  
selectList是点击左边的分类时，触发事件执行，实现点击分类右边的商品列表自动滚动到对应的分类部分。  
sliderList是滑动右边的商品列表的时候，触发事件，实现滑动到对应的分类时，左边的对应的分类自动激活。  
  
ShopCart购物车子组件  
  
  
  


这里实现了页面的返回按钮。

点击购物车时，购物车向上弹出的动画效果是通过vue的过渡效果实现。

主要是利用了vue的过渡效果的钩子函数来实现这个动画效果的。

## 十三.打包

运行命令：npm run build

如果css文件中引用了相对路径,请将config文件夹下的 index.js 的 build对象下

assetsPublicPath: '/' 改成 './'的相对路径