**Git命令**

进入文件 $ cd 文件名

返回上一层文件 $ cd ../

查看当前目录 $ ls

查看当前文件中的目录结构 $ ll-a

查看某程序版本号 $ 程序名 –v

**安装vue脚手架及项目架构简介**

$ node -v 【检测node版本（vue要求node版本在4版本以上）】

$ sudo(iOS系统加，window不需要加) npm install -g vue-cli 【安装vue-cli脚手架】

$ vue 【查看是否安装vue】

$ vue list 【查看vue中有哪些内容，里面有vue的多个模板（我们这里用webpack模板）。】

**开始新建一个项目**

$ vue init webpack my-app(项目文件名) 【初始化一个vue项目，这里是默认的，安装1.0即为1.0，安装2.0即为2.0】

$ vue init webpack#1.0 my-app(项目文件名) 【2.0版本下安装1.0时】

? project name my-app 【项目名称】

? project description (A Vue.js project) this is my first vue-app 【写入项目描述】

? Author yangxiaomiao 【项目作者】

? Use ESLint to lint your code? (Y/n) Y 【是否安装ESLint检查你的项目(ESLint为ES6的代码风格检查器)】

? Setup unit tests with Karma + Mocha? (Y/n) N 【是否使用unit，unit为前端单元测试库】

? Setup e2e tests with Nightwatch? (Y/n) N 【是否使用e2e tests】

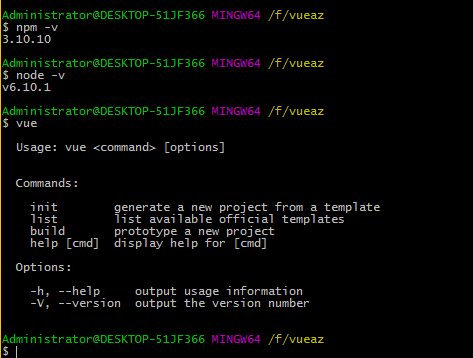
$ ls 【查看当前目录，这时就已经有了你新建的这个项目my-app】

$ cd my-app 【进入项目文件my-app】

$ ll-a 【查看当前文件中的目录结构】

$ npm install 【安装项目依赖，它会在文件中多一个node\_modules目录，里面就是依赖】

$ npm run dev 【让项目跑起来，执行项目】



npm 版本号 node 版本号 vue 版本号

**架构解释**

build 和 config 【webpack配置相关】

node\_modules 【通过npm install 安装的依赖代码库】

src 【存放项目源码，一般我们开发的所以代码都放在这个文件夹里】

static 【存放第三方静态资源】{ .gitkeep 文件：当这个目录为空，也可以把这个目录提交到git代码仓库里。}

{2.0中没有的

.babelrc 【是babel的一些配置，因为我们的代码大部分是ES6规范的，为了把ES6代码浏览器也支持】

.editorconfig 【编码器风格】

.eslintignore 【】

.eslintrc.js 【】

.gitlgnore 【】

}

index.html 【入口html文件】

package.json 【项目配置文件，描述项目】

README.md 【】

External Libraries 【】

**vue代码运行**

index.html 入口页面，向其中加入组件<app></app>

main.js 页面入口js{ 依赖vue文件 声明一个vue实例，通过components:{App}将组件插入 }

app.vue 编写页面的文件 包括template模板、script js代码、style样式三个区域块

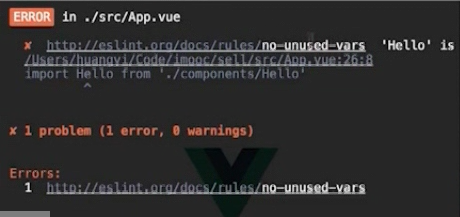
**注意：**

1. **应用组件时，必须要使用components属性进行注册，才能在上面使用组件。**
2. **代码注释前必须且只能有一个空格。**

**报错解答**

* 1. 关于ES6 的报错

当js代码中export default { components: { hello } } 这部分代码无时。报错：

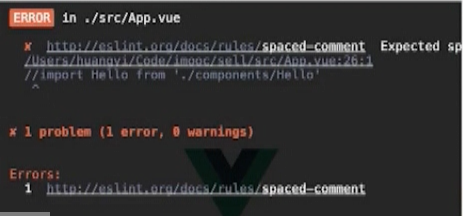


解决办法：no-unused-vars,表示你应用了一些组件，但你没有去注册它。

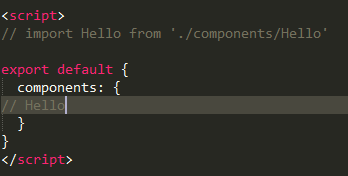
export default { components: { 组件名 } }

或注释inport 项目名from ‘./compons/组件名’

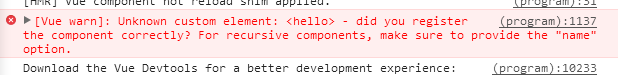
二、spaced-comment关于注释前少空格的报错（ES6的代码风格）



解决办法：spaced-comment,表示注释前必须要有一个空格。

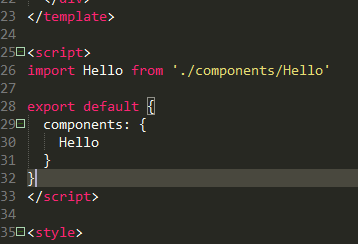


三、组件报错：不识别的一个自定义标签hello，问你有没有正确注册组件

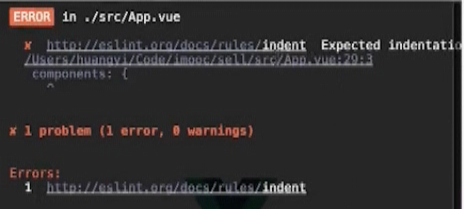


解决问题：不识别的一个自定义标签hello，问你有没有正确注册组件。所以我们必须先注册再使用。

这样注册：

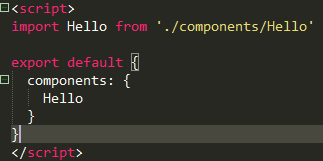


四、indent代码风格缩进问题



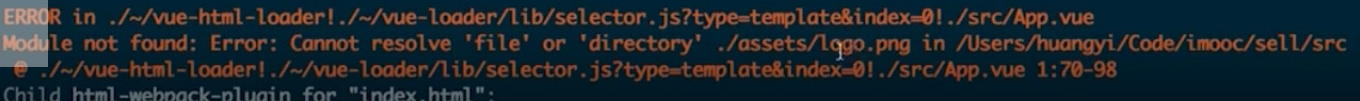
解决办法：webstorm中用Command+option+L进行代码格式化（iOS系统）

Webstorm中用control+alt+L进行代码风格化（Android）

正确代码为：****

五、not found找不到文件报错

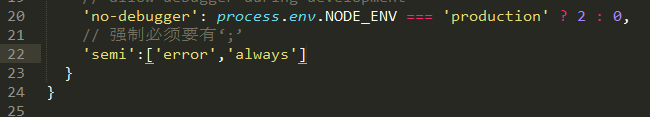
解决方法：查看路径



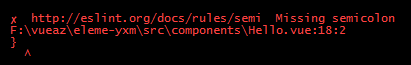
六、semi extra semicolon代码中多‘;’的报错。 

如果想要有‘;’的，可以在vue项目下的.eslintrc.js文件中改内容。

如下：



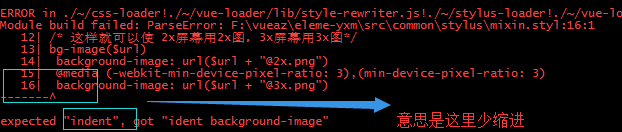
七、semi missing semicolon代码中缺少‘;’的报错。



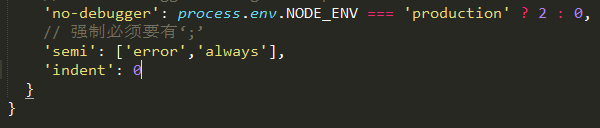
八、indent代码缩进报错 因为程序是靠缩进来确定子父级关系的，所以必须要有缩进



或者



忽略index缩进：在vue项目下的.eslintrc.js文件中改内容



九、Too many blank lines代码中有太多的空白行



Too many blank lines at the end of file. Max of 1 allowed 最后又太多空白行。

十、no-trailing-spaces trailing spaces有太多空格

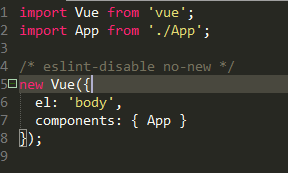


no-trailing-spaces Trailing spaces not allowed 中间有太多空格

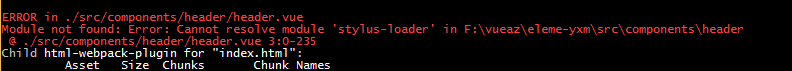
十一、app.vue文件下没有/\* eslint-disable no-new \*/时的报错



解决办法：



十二、没有stylus-loader依赖时的报错



解决办法：安装依赖 package.json文件下26行后加"stylus-loader":"^2.1.1",



之后：执行命令$ npm install 安装依赖

十三、找不到stylus Cannot find module 'stylus'



解决办法： $ npm install stylus.

十四、控制台报错：不能使用header关键字作为组件名

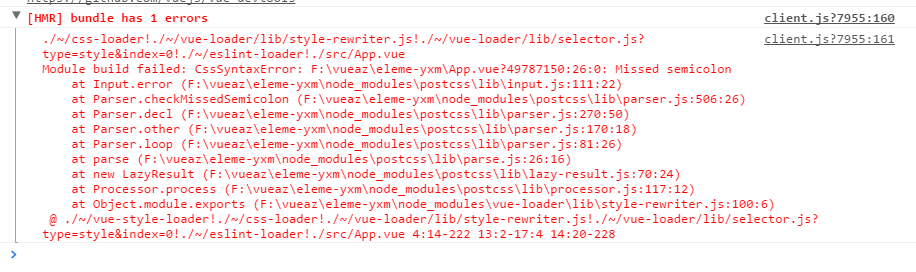


解决办法： 改两处：

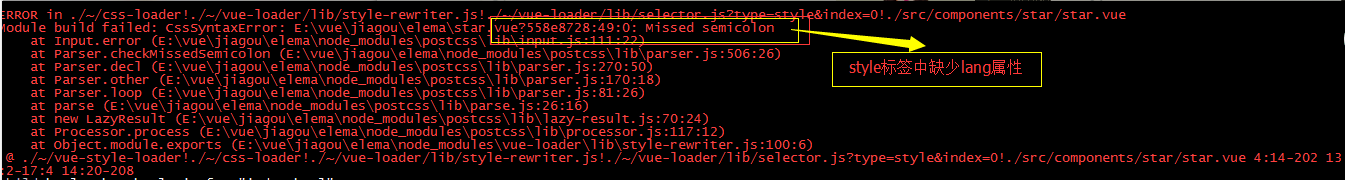
App.vue文件下 1. <header></header>改为<v-header></v-header>

App.vue文件下 2. export default { components: { 'v-header':header } };

十五、:49:0: Missed semicolon写css样式时报的错 原因：没链接lang="stylus" rel="stylesheet/stylus"文件

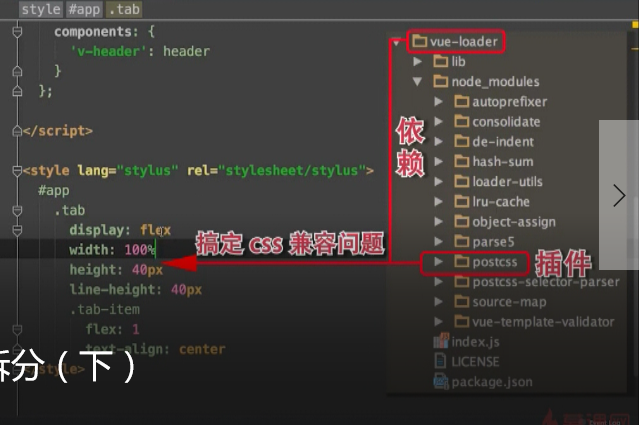


或者：

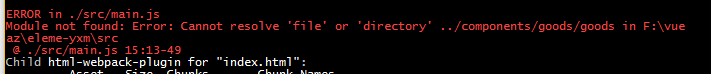


解决办法：<style lang="stylus" rel="stylesheet/stylus">

Ps附加一项：flex布局



十六、cannot resolve ‘file’ or ‘directory’路径错误



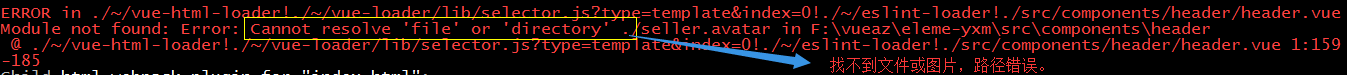
找不到文件或者图片

通过ajax请求图片：

错误：<img width="64px" height="64px" src="seller.avatar" alt="" />

正确：<img width="64px" height="64px" :src="seller.avatar" alt="" />

因为vue项目初始化的时候，图片还没请求过来，所以要加：src



十七、控制台报错 找不到#app 这是vue-router规定的写法

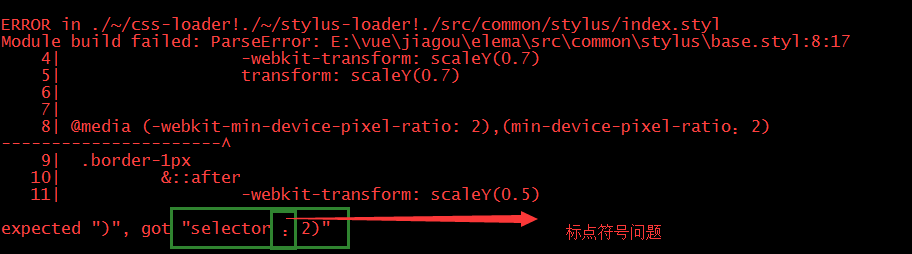
解决办法：因为index.html文件中vue的官网写的是<div id=”app”></div>。（vue-router明确必须要这样的写法。）

所以解决步骤：1. index.html文件中，将原来的<app></app>改为<div id=”app”></div>。2.app.vue文件中，将原来的最外层<div id=”app”></div>中的id=”app”删掉就好了。

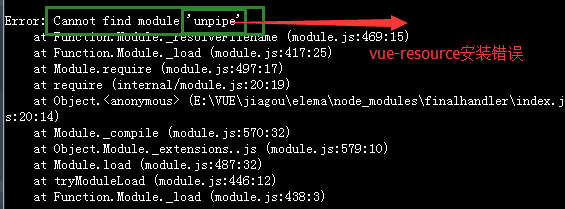
Ps附加一项：import goods from './components/goods/goods';这里为什么不需要跳出本层再去找components文件，而是直接就能找到？ 原因:在build文件下的webpack.base.conf.js中有配置，给它起了别名，如图：



十八、expected “)”,got”selector:2”标点符号‘：’报错：其中的‘:’输成了中文下的‘:’。

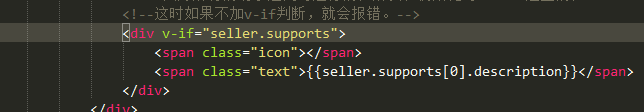


十九、vue-resource安装错误

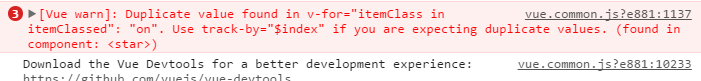


二十、vue-resource报错：因为所有的请求是同时进行的，所以在初始化时seller：{ }是空的,这时如果不加v-if判断，就会报错。

解决办法： 先用v-if判断下该内容存在不



二十一、 告诉你v-if时要用 track-by=”$index”



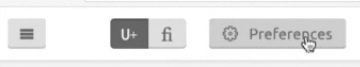
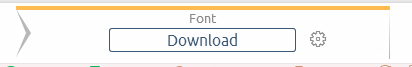
二十二、端口被占用时报错

8080端口被占用：

1. netstat -ano 查看端口进程 看是否被占用
2. 查看第二列 找到8080
3. 记住8080这一行最后面的那个数字3824
4. taskkill -f -pid 2423 杀死进程

**开 发 过 程**

一、使用图标字体过程

1. 准备图标的svg格式图片，在<https://icomoon.io/icons.html>网址上传制成图标字体。
2. 首先点击网页右上角，进入图标页面
3. 点击左上角，上传svg格式的图标。点亮需要用的图标。点击右下角，生成图标字体。点击图标左下角的<>Get Code查看使用方法。
4. 点击左上角改变Font Name。Font Name为图标的名称。
5. 最后点击下载文件和图标字体。
6. 将下载好的文件中的fonts文件放入vue项目src目录下common（得新建）下。Common文件下有js/fonts/stylus。
7. 将下载好的文件中的style.css文件放入vue项目src目录下common（得新建）下的stylus目录中。Common文件下有js/fonts/stylus。(将style.css改名为icon.styl。将css文件改为stylus语法，就是去掉所有的结束语‘;’，去掉所有的‘{}’。)

二、用自己本地的data.json文件进行模拟后台数据

1. 解析data.json文件，了解内容架构。

2.开始写ajax请求

打开build文件下的dev-server.js（开发webpack打包的入口文件）。

3.开始读取数据

var app = express()【该文件第十六行，本来就有，在这下面写ajax】

// 开始定义ajax请求

// 创建一个对象开始请求

var appData = require('../data.json');//链接json文件

// 声明变量，接收数据appData.seller中的seller为json数据中的名称

var seller = appData.seller;

var goods = appData.goods;

// 定义路由

var apiRoutes = express.Router();

// 开始写接口

apiRoutes.get('/seller',function(req,res){

res.json({

errno: 0,//用来代表数据正常

data: seller//具体的数据

});

});

apiRoutes.get('/goods',function(req,res){

res.json({

errno: 0,

data: goods

});

});

});

// 调用最上面的app，执行ajax并请求完成。

app.use('/api',apiRoutes);

//ajax加载结束

注意：可以通过<http://localhost:8080/api/seller>来看数据是否请求成功。

在谷歌浏览器中，可以用谷歌的一个jsonview插件来让数据自动格式化。

三、开始编写页面

1.组件拆分

一个app.vue大组件，再加一些小组件依赖app.vue。

2.开发时依赖的初始化css样式。

重置标签默认样式。

这里用<http://cssreset.com官网提供的标准reset>样式。

3.把该文件放到vue项目中static目录下css里面。再在index.html主页面中链接该样式。

<link rel="stylesheet" type="text/css" href="static/css/reset.css" />

4.在index.html文件<head></head>中定义移动端视口，设置初始宽高。

<meta name="viewport" conent="width=device-width,initial-scale=1.0,maximum-scale=1.0,minmum-scale=1.0,user-scalable=no" />

5.在main.js文件中写组件。

四、安装vue-router依赖就可以形成选项卡

1. 安装依赖 package.json文件里14行下加"vue-router": "^0.7.13"，再执行$ npm install安装依赖就可以了。安装完成结果会有。

2.使用方法：

（1）：// 将vue-router组件链接入项目

import VueRouter from 'vue-router';

（2）：在文件中需要选项卡的部分加<a v-link=”{path:’/goods’}”></a>，

在需要显示切换的内容的部分加<router-view></router-view>(只用一个就好)。

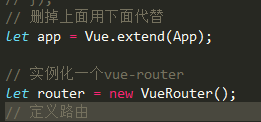


（3）：因为要用vue-router，所以要删掉原来的



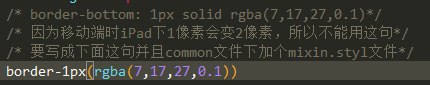
中的/\* eslint-disable no-new \*/new Vue({el: 'body',components: { App }});删掉。

用一下代码来代替：



五、1px边框（移动端布局）

1.改样式



2.加文件

（1）base.styl文件内容

（2）base.styl文件内容

（3）base.styl文件内容

六、安装vue-resource应用，渲染所有数据到页面

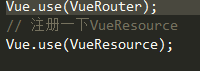
1.安装依赖 package.json文件里14行下加"vue-resource": "^1.0.1"，再执行$ npm install。

2.使用方法：

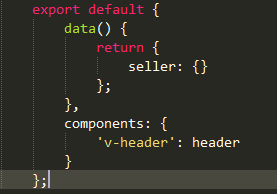
（1）main.js文件下链接



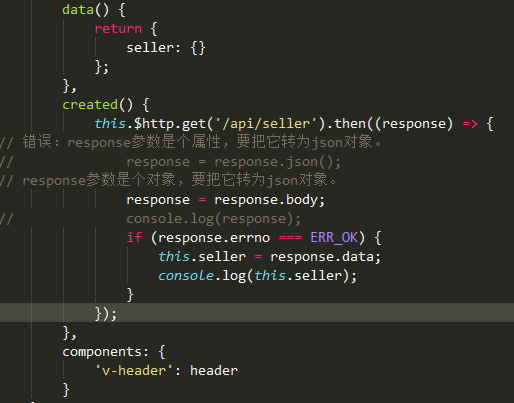
（2）main.js文件下链接



（3）app.vue文件下初始化数据对象



（4）created(){ }内部请求商家相关数据



App.vue文件下：

1. 其中有 created() { }属性，接收数据。
2. this.$http.get('/api/seller').then((response) => { }方法解析数据。
3. response = response.body;将数据解析为json对象。
4. if (response.errno === ERR\_OK) { this.seller = response.data; console.log(this.seller); }判断数据返回是否正确，正确输出数据。
5. 其中ERR\_OK为新建的常量。代码：const ERR\_OK = 0;。
6. 结束，接下来是改变dom样式把数据渲染到页面上。

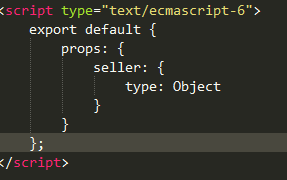
（5）渲染数据

a. 通过v-bind方法传数据

app.vue文件下： 

b. 编写header组件

header.vue文件下：



c. 编写dom结构

七、一个可以根据后台数据类型来改变图片的方法：

八、一个模糊的背景的制作方法

九、



七、处理同时应用2x、3x图片（第六章6-4外部组件6：10）

八、