一、安装Vue-cli(脚手架)工具

1. npm i –g vue-cli

2. 初始化一个vue-cli项目 版本2.XX 在版本3.0+ 用 vue create #项目名称

(1) npm install –g @vue/cli-init

(2) vue init webpack 项目名称

(3)?Project name(项目名称)

(4) ? Project description (A Vue.js project) 项目描述

(5) ? Author (Gavin1997 <15108466849@163.com>) 作者

(6) ? Vue build (Use arrow keys) #webpack配置文件 选+

(7) ? Install vue-router? (Y/n) 是否使用vue-router 选Y

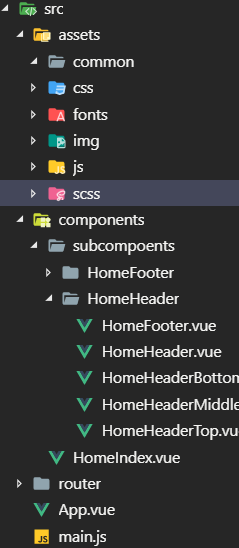
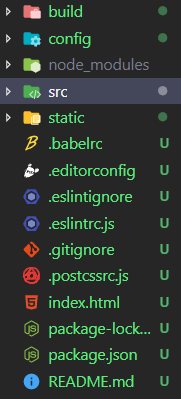
(8)? Use ESLint to lint your code? (Y/n) 是否使用Eslint检查语法 N

(9) ? Pick an ESLint preset (Use arrow keys) 选 Standard

(10) ? Set up unit tests (Y/n) 单元测试 N

(11) ? Setup e2e tests with Nightwatch? (Y/n) N

(12) ? Should we run `npm install` for you after the project has been created? (recommended) (Use arrow keys) 选use NPM



在Vue项目中使用scss 必须安装下面三个插件

1. npm install node-sass --save-dev

npm install sass-loader --save-dev

npm install style-loader --save-dev

我们想要全局引用scss 需要安装

npm install sass-resources-loader --save-dev

在build/utils.js文件中修改配置，把scss: generateLoaders('sass')改为：

return {

css: generateLoaders(),

postcss: generateLoaders(),

less: generateLoaders('less'),

sass: generateLoaders('sass', { indentedSyntax: true }),

scss: generateLoaders('sass').concat({

loader:'sass-resources-loader',

options: {

resources:path.resolve(\_\_dirname,'../src/assets/scss/\_color.scss')

}

}),

stylus: generateLoaders('stylus'),

styl: generateLoaders('stylus')

}

}

**2.图片引用的路径问题**

资源里的背景图片,不像index.html 中加载资源一样,../../assets/images/logo.png 会被相对打包成static/img,所以要保留css引用图片的正常路径

在build下的uils.js中

添加公共路径

if (options.extract) {

return ExtractTextPlugin.extract({

use: loaders,

fallback: 'vue-style-loader',

publicPath:'../../../',

})

} else {

return ['vue-style-loader'].concat(loaders)

}

}

**3.跨域访问问题**

**1. 在服务器端** 引入 cors模块 专门处理跨域问题

注意引入位置 在创建服务器之后 在监听端口之前

const prodcuts = require("./routes/products");

var app = express();

//加载处理跨域模块

const cors = require("cors");

//允许那个地址跨域访问

app.use(cors({

origin: 'http://127.0.0.1:8080',

credentials: true

}));

var server = app.listen(3000);

2. 在vue-cli中 main.js 配置axios credentials 如果用到session 因为axios默认不携带cookie,所有在 main.js中

//配置axios

*Vue*.prototype.$http = axios;

//基本访问路径

axios.defaults.baseURL = "http://127.0.0.1:3000/";

//设置axios session的credentials

axios.defaults.withCredentials = true;

//设置axios-ajax访问格式

axios.defaults.headers = {

"Content-Type": 'application/x-www-from-urlencoded'

}

**3. 在vue-cli2.x中 vue-router跳转后滚轮位置不为0保持不变的bug解决办法**

在app.vue中

export default {

name: 'App',

watch:{

'$route':function(*to*,*from*) {

document.body.scrollTop = 0;

document.documentElement.scrollTop = 0;

}

}

}

**4.在数据库mysql当中**

**当希望修改一个表里的列的字符段内容时,可用concat函数拼接**

**比如 希望在图片的src前面加上** [**http://127.0.0.1:3000/**](http://127.0.0.1:3000/)

可用[UPDATE](http://localhost/phpmyadmin/url.php?url=http://dev.mysql.com/doc/refman/5.5/en/update.html) qy\_product\_pic [set](http://localhost/phpmyadmin/url.php?url=http://dev.mysql.com/doc/refman/5.5/en/set.html) sm = CONCAT("http://127.0.0.1:3000/",sm)

**常见错误**

**1. handlers[i].call is not a function"**

处理程序 调用的不是函数

在vue中钩子函数created() { ….} 而不是created:{}

**2. 在引入子组件的路径中**

**There are multiple modules with names that only differ in casing.  
有多个模块同名仅大小写不同  
This can lead to unexpected behavior when compiling on a filesystem with other case-semantic.  
这可能导致在一些文件系统中产生不是预期的行为  
Use equal casing.   
使用唯一的写法**

原因1: improt时,文件引入的路径描述不统一,所以保留一种风格引入 尽量使用最近的路径来引入

原因2:文件名和引入的不一致,**区分大小写**

在Header组件中 引入子组件

import HomeHeaderTop from "./HomeHeaderTop"

**后面引入时也应该 ./HomeHeaderTop 切忌不要写成TOP**

**3.组件之间传参时 一定要注意参数的正确性**

**1. 当从首页的商品跳转到详情页组件时,**

getDetails(*id*){

*this*.$router.push("/product\_details/"+*id*)

}

**此时id时当前详情页的id号**

**2. 在路由中:**

{

path:'/product\_details/:tid',

name:'ProductDetails',

component:*ProductDetails*

}

**此时 path /:tid为变量 接住从首页商品跳转过来的参数值(id)**

**3. 在当前页面中 data( return{})中**

tid:*this*.$route.params.tid,

**当前页面要用的tid变量是params.后的变量 从路由传参中接到的tid**

**在axios发送请求的时候**

*this*.$http.get("details?tid="+*this*.tid).then(*res* => {