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

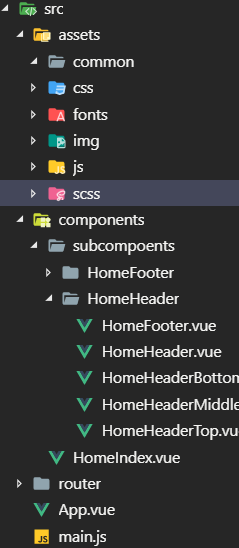
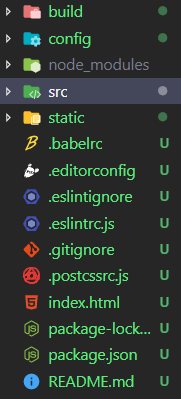
(8)? Use ESLint to lint your code? (Y/n) 是否使用Eslint检查语法 N

(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.各种问题之大坑**

**1. 跨域问题: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.当前css高亮的解决办法,不用重新刷新页面,直接写一个简单的逻辑判断**.(在同一个v-for循环中)

<li *v-for*="(c,i) of res.categories" *:class*="{booking\_active:changedActive==i}" *@click.prevent*="changeDetails(c.tid,i)">

<a>{{c.category}}</a>

</li>

自定义方法单击事件 将tid和下标传入 class属性为true的时候显示. data(reutrn{})中 changeActive默认为0;

changeDetails(*tid*,*i*) {

*console*.log(*i*)

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

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

*this*.res = res.data;

});

*this*.changedActive=*i*;

**5. 当前css高亮的解决办法,不用重新刷新页面,直接写一个简单的逻辑判断**(**不在同一个v-for循环中**)

在当前页面的data(return{})中 自定义一个数字n,默认为0;

每次点击下一页的时候 将这个n++; 上一页的时候 n--;

然后在watch中监听从后台拿到的pno(当前页面)

watch:{

pno(){

if(*this*.pno = *this*.n){

*this*.changedActive = *this*.n;

}else{

*this*.changedActive = "";

}

}

},

**解决当前页面的按钮为高亮状态问题: v-for(item,i) @click=”nowNum(i)”**

1. 在data({return{}})中设置一个全局变量 i = 0;

2. 在按钮绑定的事件 nowNum(i)中让this.i = I;

3. 在pno的监视中

pno(){

if(*this*.pno == *this*.n ){

*this*.changedActive = *this*.n;

}else if(*this*.pno==*this*.i){

*this*.changedActive = *this*.i;

}

else{

*this*.changedActive = "";

}

}

**6.实现一个简单的轮播图**

<li><img *:src*="imgArray[nowindex]"*alt*=""*style*="height:420px;width:100%; " *v-if*="ifshow"></li>

1 .img的图片列表地址保存在data(return{})中 一个数组; 设置v-if=”ifshow” 默认为true

nowindex:0; imgArry:[]

2. 在mounted里设置一个定时器

mounted() {

setInterval(() => {

*this*.nowindex++

if (*this*.nowindex === *this*.imgArray.length) {

*this*.nowindex = 0;

} else {

*this*.nowindex + 1

}

}, 2000)

},

每两秒让nowindex+1,当它等于获取图片列表数组的长度时,就代表播放完毕,让其等于0;

小圆点高亮问题:

span *v-for*="(item,index) in imgArray" *:class*="{on:index===nowindex}" *class*="carousel\_circle ml-2"*@click*="goto(index)"></span>

个数根据获得图片列表的长度来,循环得到.

calss属性是动态获取 on:当index===nowindex的时候为true;

goto(*index*) {

*this*.ifshow = true;

*this*.nowindex = index;

},

**7. 父子组件之间事件的传递**

在父组件中,引入的子组件中

<products-top *@get*="getnum"></products-top>

自定义事件名 父组件的方法名

在子组件中的methods中

自定义函数名(){

this.$emit(“get”) --@get

}

wode(){

*this*.$emit("get");

}

**8. 当需要触发键盘事件时,用@keydown**

<input *type*="text" *name*="keyword" *placeholder*="搜索国家、城市、产品" *class*="my-form-control" *list*="s\_advice" *@keydown*="search($event)" *v-model*="kw"

当一个搜索框需要搜索当按下回车搜索内容时. 先绑定 v-model 获取输入的值

在methods中 判断 e.keyCode==13时,就进行传参 kw

if(e.keyCode==13){

*this*.$router.push({path:"/products/"+*this*.kw.trim()})

}

},

Kw在DOM,BOM中的获取方式: decodeURI(location.search.split(“=”)[1]);

Kw在vue-cli中的获取方式:this.$route.params.kw

**9. 如何使用鼠标移入事件更多当前图片,小图变大图, @mouseenter;**

**不论鼠标指针穿过被选元素或其子元素，都会触发 mouseover 事件。**

**只有在鼠标指针穿过被选元素时，才会触发 mouseenter 事件**

1. 在动态获得的图片列表下 –大图

<img *:src*="res.pics[`${i}`].md" *alt*=""*style*="width:100%;height:100%;">

在data({return})中初始化i=0;

2. 在下方小图中

<li *class*="ml-1 " *v-for*="(pic,i) in res.pics" *:key*="i">

<img *:src*="pic.sm" *@mouseenter*="changeImg(i)">

</li>

3.在methods中

changeImg(*i*){

*this*.i=*i*;

}

**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* => {