Vue项目步骤第01天

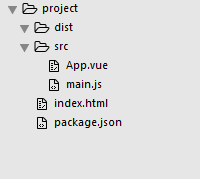
# 1、项目前的准备工作

1.1、新建文件夹project，创建package.json文件



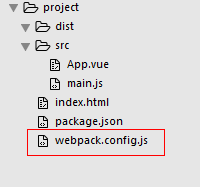
一路回车之后就能看到在project文件夹下有一个package.json文件

1.2、创建基本的目录结构



文件内容可以暂时不写

1.3、增加webpack.config.js文件



文件内容可以暂时不写

1.4、安装一些包

安装一些你目前可以想到可能需要的包，大致如下：

webpack

webpack-dev-server

html-webpack-plugin

css-loader

style-loader

file-loader

url-loader

vue

vue-loader

vue-router

vue-template-compiler

在cmd窗口中敲入命令：

cnpm i webpack webpack-dev-server html-webpack-plugin css-loader style-loader file-loader url-loader vue vue-loader vue-router vue-template-compiler --save-dev

1.5、配置webpack.config.js文件

var htmlwp = require('html-webpack-plugin');

module.exports = {

entry: \_\_dirname + "/src/main.js",

output: {

path: \_\_dirname + "/dist",

filename: "build.js"

},

module:{

loaders:[

{

test:/\.css$/,

loader:"style-loader!css-loader"

},

{

test: /\.(png|jpg|gif|ttf)$/,

loader:'url-loader?limit=20000'

},

{

test:/\.vue$/,

loader:'vue-loader'

}

]

},

plugins:[

new htmlwp({

title: '首页', //生成的页面标题

filename: 'index.html', //webpack-dev-server在内存中生成的文件名称，自动将build注入到这个页面底部，才能实现自动刷新功能

template: 'index.html' //根据index1.html这个模板来生成(这个文件请程序员自己生成)

}

)

]

}

1.6、修改package.json中的script标签

{

"name": "project",

"version": "1.0.0",

"description": "",

"main": "index.js",

"scripts": {

"dev": "webpack-dev-server --inline --hot --open --port 4009"

},

"author": "",

"license": "ISC",

"devDependencies": {

"css-loader": "^0.28.4",

"file-loader": "^0.11.2",

"html-webpack-plugin": "^2.30.1",

"style-loader": "^0.18.2",

"url-loader": "^0.5.9",

"vue": "^2.4.2",

"vue-loader": "^11.3.4",

"vue-router": "^2.7.0",

"vue-template-compiler": "^2.4.2",

"webpack": "^3.5.3",

"webpack-dev-server": "^2.7.1"

}

}

1.7、在index.html中增加一个容器div#app,并增加视口

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<title>Document</title>

</head>

<body>

<div id="app"></div>

</body>

</html>

1.8、npm run dev 测试是否正常

在cmd敲入命令：

npm run dev

看是否会报错，若不会报错则往下走。

# 2、在入口main.js文件中导入vue

2.1、导入vue

import Vue from "vue";

2.2、导入App.vue对象

import App from "./App.vue";

2.3、在vue对象中渲染App

new Vue({

el: "#app",

render: function(create) {

return create(App);

}

});

2.4、搭建App.vue基本的代码结构

<template>

<div>

</div>

</template>

<script>

export default {

}

</script>

<style scoped>

</style>

2.5、测试,在cmd中nmp run dev 看是否有错误输出

# 3、实现App.vue布局

* 1. 使用mint-ui中的header组件实现头部

**a、下载mint-ui的包**

cnpm i mint-ui --save-dev

**b、在main.js文件中导入mint-ui对象和样式**

import mintUi from "mint-ui";

import "mint-ui/lib/style.css";

**c、在main.js文件中，配置让Vue使用mintUi**

Vue.use(mintUi);

**d、在mint-ui的文档中找到header组件，在App.vue中的template进行使用**

<template>

<div>

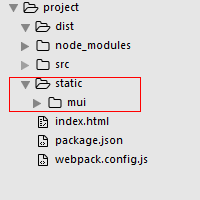
<mt-header fixed title="传智播客黑马程序员"></mt-header>

</div>

</template>

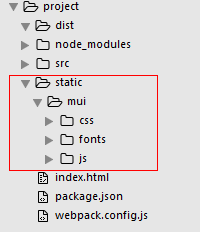
3.2、使用mui中的tabbar组件实现底部

**a、在project下建立目录，如下**



**b、下载源代码，将源代码中dist下面的文件夹放入到mui的目录中去**

源代码的路径<https://github.com/dcloudio/mui>



**c、在main.js文件中导入mui的样式**

import "../static/mui/css/mui.css";

**d、查看mui的官方案例，找到tabbar，放到App.vue的template中**

案例地址：<http://dcloud.io/hellomui/>

<template>

<div>

<mt-header fixed title="传智播客黑马程序员"></mt-header>

<nav class="mui-bar mui-bar-tab">

<a class="mui-tab-item mui-active" href="#tabbar">

<span class="mui-icon mui-icon-home"></span>

<span class="mui-tab-label">首页</span>

</a>

<a class="mui-tab-item" href="#tabbar-with-chat">

<span class="mui-icon mui-icon-email">

<span class="mui-badge">9</span>

</span>

<span class="mui-tab-label">消息</span>

</a>

<a class="mui-tab-item" href="#tabbar-with-contact">

<span class="mui-icon mui-icon-contact"></span>

<span class="mui-tab-label">通讯录</span>

</a>

<a class="mui-tab-item" href="#tabbar-with-map">

<span class="mui-icon mui-icon-gear"></span>

<span class="mui-tab-label">设置</span>

</a>

</nav>

</div>

</template>

**e、修改复制过来的代码，改变小圆点的位置**

将**<span class="mui-badge">9</span>**从第消息移动到通讯录，并且修改底部的文字，分别为首页、会员、购物车、搜索

3.3、在App.vue文件的template中使用router-view进行占位

<router-view></router-view>

这个标签应该在mt-header与nav之间

# 4、增加路由功能

4.1、在main.js文件中使用vue-router

import VueRouter from "vue-router";

Vue.use(VueRouter);

var router = new VueRouter({

routes:[

{},

{},

{},

{}

]

});

new Vue({

el:"#app",

router:router,

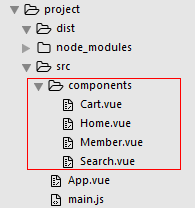
render:function(create){

return create(App);

}

});

4.2、在src目录下创建components文件夹，在components下创建Home.vue、Member.vue、Cart.vue、Search.vue，填充好对应的代码结构



代码结构为：

<template>

<div>

首页

</div>

</template>

<script>

export default {

}

</script>

<style scoped>

</style>

4.3、将App.vue底部的nav中的a标签变成router-link,href变成to，to中的值为路径，分别为/home、/member、/cart、/search， 如下

<template>

<div>

<mt-header fixed title="传智播客黑马程序员"></mt-header>

<router-view></router-view>

<nav class="mui-bar mui-bar-tab">

<router-link class="mui-tab-item mui-active" to="/home">

<span class="mui-icon mui-icon-home"></span>

<span class="mui-tab-label">首页</span>

</router-link>

<router-link class="mui-tab-item" to="/member">

<span class="mui-icon mui-icon-email">

</span>

<span class="mui-tab-label">会员</span>

</router-link>

<router-link class="mui-tab-item" to="/cart">

<span class="mui-icon mui-icon-contact">

<span class="mui-badge">9</span>

</span>

<span class="mui-tab-label">购物车</span>

</router-link>

<router-link class="mui-tab-item" to="/search">

<span class="mui-icon mui-icon-gear"></span>

<span class="mui-tab-label">搜索</span>

</router-link>

</nav>

</div>

</template>

4.4、将之前写的路由规则补全

**a、在main.js中分别把Home.vue、Member.vue、Cart.vue、Search.vue导出**

import Home from "./components/Home.vue";

import Member from "./components/Member.vue";

import Cart from "./components/Cart.vue";

import Search from "./components/Search.vue";

**b、补全规则**

var router = new VueRouter({

routes: [

{

path: "/home",

component: Home

},

{

path: "/member",

component: Member

},

{

path: "/cart",

component: Cart

},

{

path: "/search",

component: Search

}

]

});

这个步骤做完之后，点击底部按钮，可以正常路由，但是看不到文字，原因是文字被标题挡住了。需要增加全局样

4.5、增加全局样式

**a、在static目录下建立site.css文件，增加如下内容**

body {

background-color: white;

}

.templateDiv {

margin-top: 40px;

margin-bottom: 50px;

padding: 10px;

}

**b、在components目录下所有的.vue文件中，将template中的根div增加class为templateDiv**

如Home.vue

<template>

<div class="templateDiv">

首页

</div>

</template>

**c、在main.js中使用site.css**

import "../static/site.css";

4.6、完善底部栏的点击效果，在创建VueRouter对象的时候传入linkActiveClass

先去掉App.vue中首页router-link中的mui-active的类样式，让后在创建路由的时候修改linkActiveClass属性即可。

var router = new VueRouter({

linkActiveClass:"mui-active",

routes: [

{

path: "/home",

component: Home

},

{

path: "/member",

component: Member

},

{

path: "/cart",

component: Cart

},

{

path: "/search",

component: Search

}

]

});

# 5、Home.vue的实现

5.1、实现轮播图效果，使用mint-ui中的js组件,修改Home.vue中的template

<template>

<div class="templateDiv">

<mt-swipe :auto="4000">

<mt-swipe-item>1</mt-swipe-item>

<mt-swipe-item>2</mt-swipe-item>

<mt-swipe-item>3</mt-swipe-item>

</mt-swipe>

</div>

</template>

默认情况下，mt-swipe是没有高度的，需要在style中进行设置

<style scoped>

.mint-swipe{

width: 100%;

height: 300px;

background-color: red;

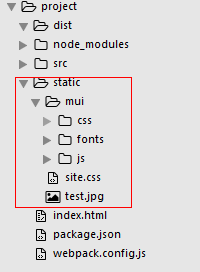
}

</style>

这样就能看到红色的块在进行轮播了

5.2、在mt-swipt-item中加入img标签，用来显示图片,并设置样式

在static目录下放一张test.jpg的测试图片



<template>

<div class="templateDiv">

<mt-swipe :auto="4000">

<mt-swipe-item>

<img src="../../static/test.jpg" alt="">

</mt-swipe-item>

<mt-swipe-item>

<img src="../../static/test.jpg" alt="">

</mt-swipe-item>

<mt-swipe-item>

<img src="../../static/test.jpg" alt="">

</mt-swipe-item>

</mt-swipe>

</div>

</template>

修改img的样式

.mint-swipe img{

width: 100%;

height: 100%;

}

5.3、通过v-for的方式来遍历Home.vue对象的数据，根据数据增加mt-swipe-item

<template>

<div class="templateDiv">

<mt-swipe :auto="4000">

<mt-swipe-item v-for="item in slideList">

<img src="../../static/test.jpg" alt="">

</mt-swipe-item>

</mt-swipe>

</div>

</template>

<script>

export default {

data:function(){

return {

slideList : [1,2,3,4,5]

};

}

}

</script>

5.4、使用ajax请求获取服务器的轮播图数据

**a、安装vue-resource包**

cnpm i vue-resource --save-dev

**b、在main.js中导入vue-resource对象**

import VueResource from "vue-resource";

**c、使用vue-resource对象**

Vue.use(VueResource);

这个话的含义就是全局的组件对象都有一个$.http对象可以使用了

**d、在Home.vue中使用$.http对象获取服务器数据**

<script>

export default {

data:function(){

return {

slideList : []

};

},

created:function(){

this.getSlideList();

},

methods:{

getSlideList:function(){

var url = "http://139.199.192.48:8888/api/getlunbo";

this.$http.get(url).then(function(response){

this.slideList = response.body.message;

console.log(response.body);

});

}

}

}

</script>

**e、修改template代码**

<mt-swipe-item v-for="item in slideList">

<img v-bind:src="item.img" alt="">

</mt-swipe-item>

5.5、实现HomeVue的9宫格效果

**a、使用mui中的9宫格效果，拷贝代码到Home.vue中的template**

<template>

<div class="templateDiv">

<mt-swipe :auto="4000">

<mt-swipe-item v-for="item in slideList">

<img v-bind:src="item.img" alt="">

</mt-swipe-item>

</mt-swipe>

<div class="mui-content">

<ul class="mui-table-view mui-grid-view mui-grid-9">

<li class="mui-table-view-cell mui-media mui-col-xs-4 mui-col-sm-3">

<a href="#">

<span class="mui-icon mui-icon-home"></span>

<div class="mui-media-body">新闻资讯</div>

</a>

</li>

<li class="mui-table-view-cell mui-media mui-col-xs-4 mui-col-sm-3">

<a href="#">

<span class="mui-icon mui-icon-email"></span>

<div class="mui-media-body">图片分享</div>

</a>

</li>

<li class="mui-table-view-cell mui-media mui-col-xs-4 mui-col-sm-3">

<a href="#">

<span class="mui-icon mui-icon-chatbubble"></span>

<div class="mui-media-body">商品购买</div>

</a>

</li>

<li class="mui-table-view-cell mui-media mui-col-xs-4 mui-col-sm-3">

<a href="#">

<span class="mui-icon mui-icon-location"></span>

<div class="mui-media-body">留言反馈</div>

</a>

</li>

<li class="mui-table-view-cell mui-media mui-col-xs-4 mui-col-sm-3">

<a href="#">

<span class="mui-icon mui-icon-search"></span>

<div class="mui-media-body">视频专区</div>

</a>

</li>

<li class="mui-table-view-cell mui-media mui-col-xs-4 mui-col-sm-3">

<a href="#">

<span class="mui-icon mui-icon-phone"></span>

<div class="mui-media-body">联系我们</div>

</a>

</li>

</ul>

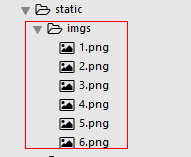
</div>

</div>

</template >

**b、配置好样式**

在static下创建imgs文件夹，在这个文件夹下放入6个按钮的图片



在Home.vue中的style进行如下配置

<style scoped>

.mint-swipe{

width: 100%;

height: 300px;

background-color: red;

}

.mint-swipe img{

width: 100%;

height: 100%;

}

.mui-grid-view.mui-grid-9{

background-color: white;

border: 0px;

}

.mui-grid-view.mui-grid-9 .mui-table-view-cell{

border: 0px;

}

.mui-icon-email:before,

.mui-icon-chatbubble:before,

.mui-icon-location:before,

.mui-icon-search:before,

.mui-icon-phone:before,

.mui-icon-home:before{

content: "";

width: 40px;

height: 40px;

display: block;

background-size: 40px 40px;

}

.mui-icon-home:before{

background-image: url("../../static/imgs/1.png");

}

.mui-icon-email:before{

background-image: url("../../static/imgs/2.png");

}

.mui-icon-chatbubble:before{

background-image: url("../../static/imgs/3.png");

}

.mui-icon-location:before{

background-image: url("../../static/imgs/4.png");

}

.mui-icon-search:before{

background-image: url("../../static/imgs/5.png");

}

.mui-icon-phone:before{

background-image: url("../../static/imgs/6.png");

}

</style>

**c、将Home.vue中的a标签变成router-link标签，将href属性改为to，跟上对应的路径**

大致规则如下：

router-link to="/news/newlist"

router-link to="/photo/photolist"

router-link to="/goods/goodslist"

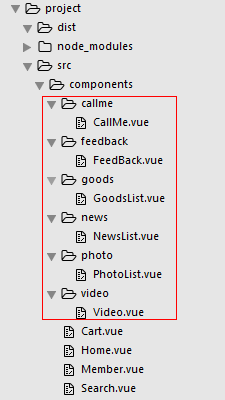
router-link to="/feekback"

router-link to="/video"

router-link to="/callme"

**d、创建上个步骤使用的vue组件，并建好vue文件的基本代码结构**

目录结构如下：



Vue代码结构如下：

<template>

<div class="templateDiv">

新闻列表

</div>

</template>

<script>

export default{

}

</script>

<style scoped>

</style>

**e、在main.js中更改路由列表**

import NewsList from "./components/news/NewsList.vue";

import PhotoList from "./components/photo/PhotoList.vue";

import Video from "./components/video/Video.vue";

import CallMe from "./components/callme/CallMe.vue";

import FeedBack from "./components/feedback/FeedBack.vue";

import GoodsList from "./components/goods/GoodsList.vue";

var router = new VueRouter({

linkActiveClass:"mui-active",

routes:[

{

path:"/home",

component:Home

},

{

path:"/member",

component:Member

},

{

path:"/cart",

component:Cart

},

{

path:"/search",

component:Search

},

{

path:"/news/newlist",

component:NewsList

},

{

path:"/photo/photolist",

component:PhotoList

},

{

path:"/goods/goodslist",

component:GoodsList

},

{

path:"/feekback",

component:FeedBack

},

{

path:"/video",

component:Video

},

{

path:"/callme",

component:CallMe

}

]

});

# 6、NewsList.vue的实现

6.1、使用$http对象来获取服务器的数据

**a、定义NewsList.vue对象中的data属性,给data中增加一个数组newsList**

<script>

export default{

data:function(){

return {

newsList:[]

};

}

}

</script>

**b、在methods中定义获取数据的方法getNewsList**

<script>

export default{

data:function(){

return {

newsList:[]

};

},

methods:{

getNewsList:function(){

}

}

}

</script>

**c、在created中调用getNewsList方法**

<script>

export default{

data:function(){

return {

newsList:[]

};

},

methods:{

getNewsList:function(){

}

},

created:function(){

this.getNewsList();

}

}

</script>

**d、获取数据成功之后，对newsList进行赋值**

<script>

export default{

data:function(){

return {

newsList:[]

};

},

methods:{

getNewsList:function(){

var url = "http://139.199.192.48:8888/api/getnewslist";

this.$http.get(url).then(function(response){

this.newsList = response.body.message;

console.log(response.body.message);

});

}

},

created:function(){

this.getNewsList();

}

}

</script>

6.2、使用mui的图文列表，对NewsList.vue中的template中的html进行填充

<template>

<div class="templateDiv">

<ul class="mui-table-view">

<li class="mui-table-view-cell mui-media">

<a href="javascript:;">

<img class="mui-media-object mui-pull-left" src="../../../static/test.jpg">

<div class="mui-media-body">

幸福

<p class='mui-ellipsis'>能和心爱的人一起睡觉，是件幸福的事情；可是，打呼噜怎么办？</p>

</div>

</a>

</li>

<li class="mui-table-view-cell mui-media">

<a href="javascript:;">

<img class="mui-media-object mui-pull-left" src="../../../static/test.jpg">

<div class="mui-media-body">

木屋

<p class='mui-ellipsis'>想要这样一间小木屋，夏天挫冰吃瓜，冬天围炉取暖.</p>

</div>

</a>

</li>

<li class="mui-table-view-cell mui-media">

<a href="javascript:;">

<img class="mui-media-object mui-pull-left" src="../../../static/test.jpg">

<div class="mui-media-body">

CBD

<p class='mui-ellipsis'>烤炉模式的城，到黄昏，如同打翻的调色盘一般.</p>

</div>

</a>

</li>

</ul>

</div>

</template>

6.3、使用v-for，让列表由newsList来填充

<template>

<div class="templateDiv">

<ul class="mui-table-view">

<li class="mui-table-view-cell mui-media" v-for="item in newsList">

<a href="javascript:;">

<img class="mui-media-object mui-pull-left" src="../../../static/test.jpg">

<div class="mui-media-body">

幸福

<p class='mui-ellipsis'>能和心爱的人一起睡觉，是件幸福的事情；可是，打呼噜怎么办？</p>

</div>

</a>

</li>

</ul>

</div>

</template>

6.4、将数据由newsList中的item进行填充

<template>

<div class="templateDiv">

<ul class="mui-table-view">

<li class="mui-table-view-cell mui-media" v-for="item in newsList">

<a href="javascript:;">

<img class="mui-media-object mui-pull-left" :src="item.img\_url">

<div class="mui-media-body">

{{item.title}}

<p class='mui-ellipsis'>{{item.zhaiyao}}</p>

</div>

</a>

</li>

</ul>

</div>

</template>

6.5、将界面进行调整

**a、列表中的图片更大一些**

<style scoped>

.mui-table-view .mui-media-object{

width: 80px;

height: 80px;

max-width: 80px;

line-height: 80px;

}

</style>

**b、在每一个列表项的底部增加时间和点击数信息**

<template>

<div class="templateDiv">

<ul class="mui-table-view">

<li class="mui-table-view-cell mui-media" v-for="item in newsList">

<a href="javascript:;">

<img class="mui-media-object mui-pull-left" :src="item.img\_url">

<div class="mui-media-body">

{{item.title}}

<p class='mui-ellipsis'>{{item.zhaiyao}}</p>

<p class='ft'>

<span>{{item.add\_time}}</span>

<span class="click-number">点击数{{item.click}}</span>

</p>

</div>

</a>

</li>

</ul>

</div>

</template>

样式设置如下：

<style scoped>

.mui-table-view .mui-media-object{

width: 80px;

height: 80px;

max-width: 80px;

line-height: 80px;

}

.ft{

font-size: 12px;

margin-top: 1em;

color:#0094ff;

}

.click-number{

margin-left: 10px;

}

</style>

6.6、使用过滤器将时间进行格式化

**a、在main.js中定义全局过滤器**

Vue.filter("dateFmt",function(input,fmtString){

return "2017-10-10";

});

**b、使用过滤器**

<span>{{item.add\_time | dateFmt('YYYY-MM-DD HH:mm:ss')}}</span>

**c、使用moment**

* + - 1. **下载moment**

cnpm i moment --save-dev

* + - 1. **在main.js文件中导入moment**

import moment from "moment";

* + - 1. **在过滤器中格式化日期的时候使用moment**

Vue.filter("dateFmt",function(input,fmtString){

return moment(input).format(fmtString);

});

# 7、NewsInfo.vue的实现

7.1、在news目录下创建NewsInfo.vue，并写好vue文件的代码结构

<template>

<div class="templateDiv">

新闻详情界面

</div>

</template>

<script>

export default {

}

</script>

<style scoped>

</style>

7.2、在main.js配置NewsInfo.vue的路由规则

import NewsInfo from "./components/news/NewsInfo.vue";

路由信息如下

{

path:"/news/newsinfo",

component:NewsInfo

}

7.3、将NewsList.vue中的a标签改为router-link，href变为to，to的值为/news/newsinfo

<template>

<div class="templateDiv">

<ul class="mui-table-view">

<li class="mui-table-view-cell mui-media" v-for="item in newsList">

<router-link to="/news/newsinfo">

<img class="mui-media-object mui-pull-left" :src="item.img\_url">

<div class="mui-media-body">

{{item.title}}

<p class='mui-ellipsis'>{{item.zhaiyao}}</p>

<p class='ft'>

<span>{{item.add\_time | dateFmt('YYYY-MM-DD HH:mm:ss')}}</span>

<span class="click-number">点击数{{item.click}}</span>

</p>

</div>

</router-link>

</li>

</ul>

</div>

</template>

7.4、考虑将NewsList.vue中用户所点击的新闻id带到NewsInfo中去

**a、在main.js中修改路由规则，增加参数**

{

path:"/news/newsinfo/:id",

component:NewsInfo

}

**b、在NewsList.vue中传递参数**

<router-link v-bind="{to:'/news/newsinfo/' + item.id}">

**c、在NewsInfo.vue中的created方法中接收参数**

<script>

export default {

created:function(){

var newsId = this.$route.params.id;

console.log(newsId);

}

}

</script>

7.5、获取新闻详情数据

<script>

export default {

created:function(){

var newsId = this.$route.params.id;

console.log(newsId);

var url = "http://139.199.192.48:8888/api/getnew/" + newsId;

this.$http.get(url).then(function(response){

console.log(response.body);

});

}

}

</script>

7.6、界面的实现

<template>

<div class="templateDiv">

<div class="title">

<h3>{{info.title}}</h3>

<p>{{info.add\_time | dateFmt("YYYY-MM-dd HH:mm:ss")}} {{info.click}}次浏览</p>

</div>

<div class="content" v-html="info.content"></div>

</div>

</template>

<script>

export default {

data:function(){

return {

info:{}

};

},

created:function(){

var newsId = this.$route.params.id;

console.log(newsId);

var url = "http://139.199.192.48:8888/api/getnew/" + newsId;

this.$http.get(url).then(function(response){

console.log(response.body);

this.info = response.body.message[0];

});

}

}

</script>

<style scoped>

.title{

color:#0094ff;

}

</style>

7.7、域名的抽取

**a、在static目录下创建一个文件夹为common.js**

export default {

install: function(Vue){

Vue.prototype.$urlPrefix = "http://139.199.192.48:8888";

}

}

**b、在main.js中将common注入到Vue中**

import common from "../static/common.js";

Vue.use(common);

**c、在vue文件中使用域名前缀**

var url = this.$urlPrefix + "/api/getnew/" + newsId;

也可以采取另外一种方案：不使用common.js，直接在main.js中

Vue.prototype.$urlPrefix = "http://139.199.192.48:8888";