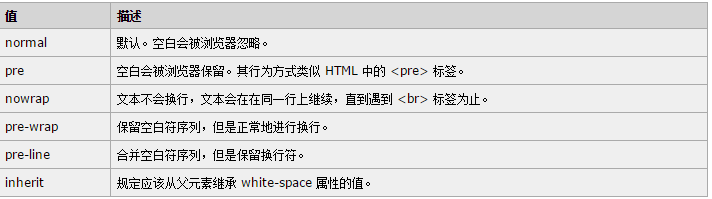
1、Jq weui地址选择器使用：

同步反显的数据不能直接vaule=”${addr}”,这样会导致cityPicker报错；

解决办法：js给目标input赋值；

## 2、空格占位实现：

Css white-space：



## 3、构建vue项目遇到的问题

1>安装vue-cli时遇到环境变量配置问题

2>npm install时报错（以管理员身份运行命令行工具）

3>初始建项目的时候不要开测试，下载缓慢，报错，并且操作麻烦

4>Npm run dev 时报错（端口号冲突，8080端口号被占用）

### 4、动态绑定img的src时，图片路径找不到

vues/index.vue static/images assets/images

图片放在assets下面 在vues/index.vue中配置变量imgSrc=’../assets/images/img.png’，

报错，找不到图片文件

图片放在static下面 在vues/index.vue中配置变量imgsrc=’/static/images.img.png’

则可以加载出图片文件

### 5、底部导航状态跟随$router.path变化

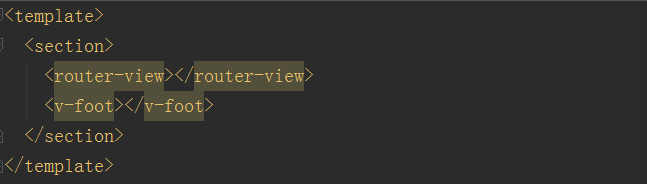
Active 状态：router-link-active

### 6、嵌套路由

路由配置：

{  
 path:'/business',  
 name:'business',  
 component:Business,  
 children:[  
 {  
 path:'',//注意：如果这里配置的为非空路径，则访问/business时页面不会渲染出 任何东西，所以配置成空路由，这样访问/business时渲染的是proList的内容  
 name:'proList',  
 component:proList  
 },  
 {  
 path:'proDetail/:id',  
 name:'proDetail',  
 component:proDetail  
 }]  
}

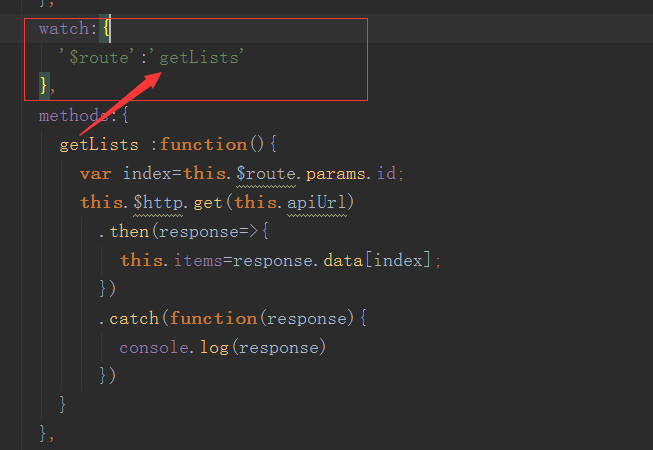
Business.vue文件入下：



此为：嵌套路由

附：以‘/’开头的嵌套路径会被当做根路径。故，使用嵌套路由时，使用嵌套的组件无需设置嵌套的路径，即children的path配置不需加‘/’

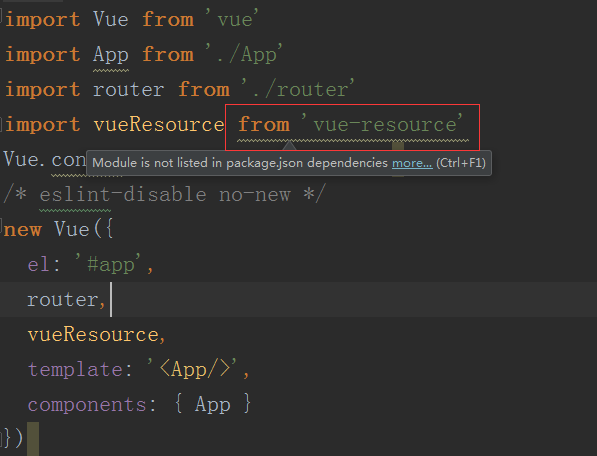
### 7、在页面链接相同页面参数不同时，想要在当前页变更参数实现数据刷新，可用watch监听$route的变化，当$route变化时重新获取数据刷新页面

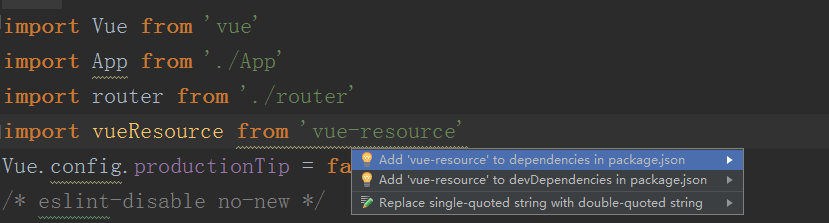


### 8、vue-resource

1)npm install vue-resource -g,仅安装，但是并没有在packge.json中的dependencies中写入，解决方法：

A、npm install vue-resource –D



B、如上图一，没有在packge.json中配置的话会有提示信息，在下划线出按alt+enter,即可出现图二中的选项，点击将其添加到packge.json文件中即可

2)vue-resource的引用(在main.js中)

Import vueResource from ‘vue-resource’，

Vue.use(vueResource)

在new vue({})中注册vueResource

在使用vue-resource的组件中import vueResource form ‘vue-resource’

（详见vues/sbInfo/children/sbInfo）

### 9、关于scss的作用域：

外部引入的文件<style scoped>@import ‘……/.scss’</style>,scoped并没有起作用

原因是scoped只对scss、less语言起作用，需要在<style lang=”scss/less”></style>

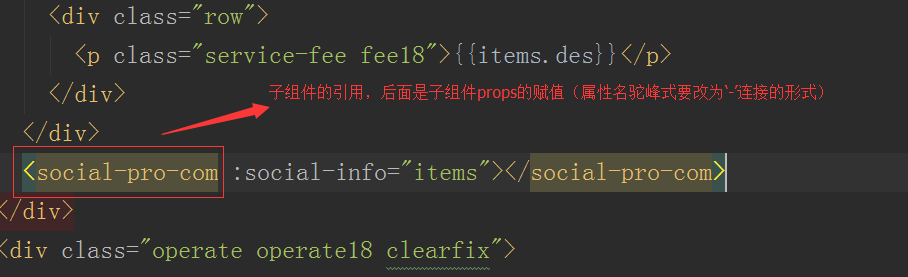
中声明，同时要能编译scss、less文件需要node-sass-loader，故安装node-sass-Loader即可，按照命令行中的提示安装需要的即可，安装python的过程中可能需要配置环境变量

### 10、父组件与子组件通信的实现[props]

子组件中声明props:[‘socialInfo’]，父组件中在调用子组件时给子组件赋值



此为在子组件中声明props------socialInfo



此为子组件在父组件中的使用(详见‘vues/pro/children/socialProCom’、vues/pro/children/shabao48)

#### 11、slot的使用

1）匿名slot

<div id=”parent”>

<child>

<p>content one</p>

<p>content two</p>

</child>

</div>

<div id=”child”>

<p>this is child content</p>

<slot></slot>

</div>

**渲染结果：**

<div id=”parent”>

<p>this is child content</p>

<p>content one</p>

<p>content two</p>

</div>

2）具名slot

<div id=”parent”>

<child>

<p slot=”ctn1”>content one</p>

<p slot=”slot2”>content two</p>

</child>

</div>

<div id=”child”>

<p>this is child content</p>

<slot name=”ctn1”>text1</slot>

<slot name=”ctn2”>text2</slot>

</div>

**渲染结果：**

<div id=”parent”>

<p>this is child content</p>

<p>content one</p>

<p>content two</p>

</div>

#### 12、vue轮播插件­——vue-awesome-swiper

使用步骤：

1. 安装 npm install vue-awesome-swiper –save (--save 就直接在package.json文件中记录了)
2. 声明
   1. 全局声明：import VueAwesomeSwiper from ‘vue-awesome-swiper’

Vue.use(VueAwesomeSwiper)

* 1. 组件内声明 import { swiper, swiperSlide } from ‘vue-awesome-swiper’

3) 调用：

<template>

<swiper :option=’swiperOption’ ref=’mySwiper’>

<swiper-slide> slide1</swiper-slide>

<swiper-slide> slide2</swiper-slide>

<swiper-slide> slide3</swiper-slide>

<div class=”swiper-pagination” slot=’pagenation’></div>//;轮播图的按钮

<div class=”swiper-pagination” slot=’button-prev’></div>//;轮播图的按钮

</swiper>

</template>

Export default{

Data(){

Return {

swiperOption:{

notNextTick:true, //组件的自有属性，设置为true，组件则不会通过

nextTick 来实例化swiper(mount 中会用到这个属性)

autoplay:3000,

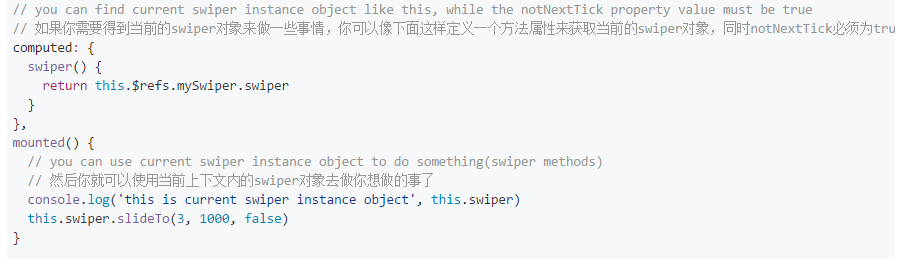
direction:horizontal,

… //(以上属性参照swiper官网介绍，属性用法完全相同)

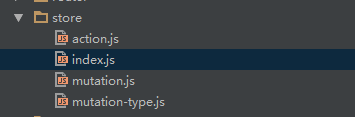
}

}

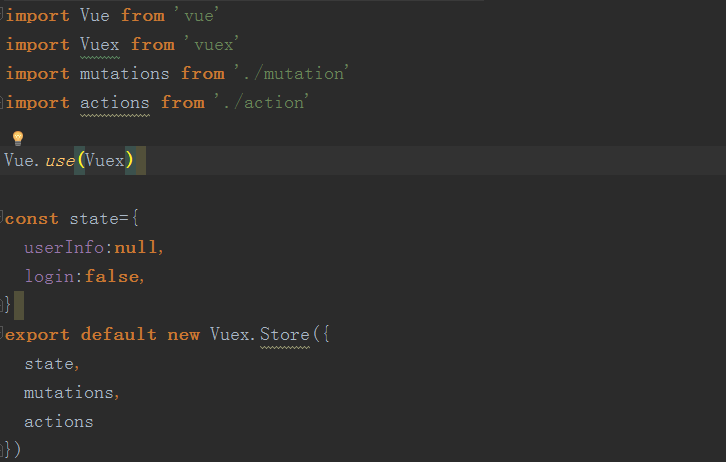
}

}

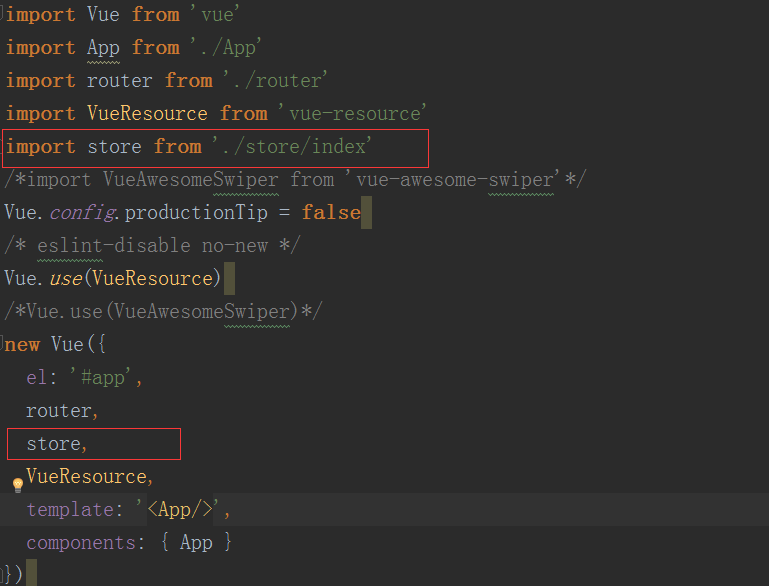
#### 13、vuex的使用



在index中声明：



在main.js文件中引入store



否则store中的state,mutation,action都无法使用

#### 14、vuex中state实际是全局变量，当刷新页面时，state会恢复“出厂设置”，如何解决

Demo中:用sessionStorage中存储登录状态，将state中的登录状态初始值赋为sessionStorage中存储的登录状态，页面加载时重新获取用户信息，并赋值给state

实际项目中: 用cookie或者sessionStorage中存储 auth\_token，根据anth\_token判断登录状态，并重新获取用户信息

15、mapState的使用：

（当一个组件需要获取到多个状态时，将这些状态都声明为计算属性会有些重复和冗余，故用mapState辅助函数帮助生成计算属性）

使用有两种：

1）当映射的计算属性名称与state的子节点的名称一致时，可以给mapState传一个字符串数组

Computed:mapState([‘userInfo’,’login’,’userDetailInfo’])

2)当需要将state的子节点重命名使用时，可以如下采用赋值的形式：

Computed:mapState({

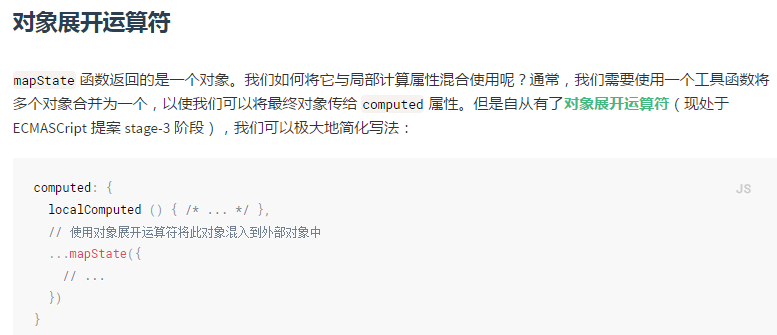
loginState:’login’,

userInfo:’info’,

})



##### 对于对象展开符的使用有些疑惑：



#### 16、Mutation的理解

Vuex中更改store中的状态的唯一方法是提交Mutation,每个Mutation都包含一个字符串的事件类型（type）和一个回调函数（handler）,回调函数用来进行状态的更改，并且接受state作为他的第一个参数

注，不能直接调用Mutation 的handler，需要以相应的type调用store.commit方法

注：**Mutation必须是同步函数**

如下代码：

Const store=new Vuex.store({

State:{

Count:1

},

Mutations:{

Increment(state){

State.count++

}

}

})

调用此回调函数：

Store.commit(‘increment’);

###### 使用常量替代Mutation事件类型

可将这些常量放在单独的文件中可对整个app包含的mutation一目了然

使用示例如下：

Export const SOME\_MUTATION=’SOME\_MUTATION’;

Import {SOME\_MUTATION} from ‘…’

Const store=new Vuex.store({

State:{….},

Mutations:{

[SOME\_MUTATION](state){

…….

}

}

})

##### 在组件中提交Mutations

Import {mapMutations} from ‘vuex’

Export default{

Methods:{

…mapMutations([

‘increment’ //映射 this。Increment()为this.$store.commit(‘increment’)

]),

…mapMutations([

Add:‘increment’ //映射 this.add() 为this.$store.commit(‘increment’)

])

}

}

#### 17、npm install … --save-dev 与 –save 的区别

--save-dev与 –save都是本地安装

--save-dev的含义代表把安装包的信息写入package.json文件的devDependencies字段中，

--save的含义代表把安装包信息写入package.json文件中的 dependencies的字段中，

如果是dependencies，那么在runtime运行时会加载这些模块，而devDependencies不会

故，（根据本人理解）--save-dev 是在安装一些编译工具插件时用到，而—save则是项目内使用的功能插件时用到

#### 18、vue-router更改页面的title

见链接：<https://juejin.im/entry/586efffcda2f600053d85b35>

微信的处理见链接：

https://github.com/deboyblog/vue-wechat-title-demo/blob/use-directive/src/main.js

##### 19、vue-resource的回调函数中this仍指向当前实例vue，而axios回调函数中，this指向发生了变化，让this指向当前vue实例，有如下方法：

a)var that=this，强行传入；

b).then(function(res){console.log(this.data)}.bind(this))

c)采用es6的箭头函数：.then((res)=>{console.log(this.data)})