# 1：路由的使用

1：cnpm i vue-router -S

2:

NPM

```
npm install vue-router
```

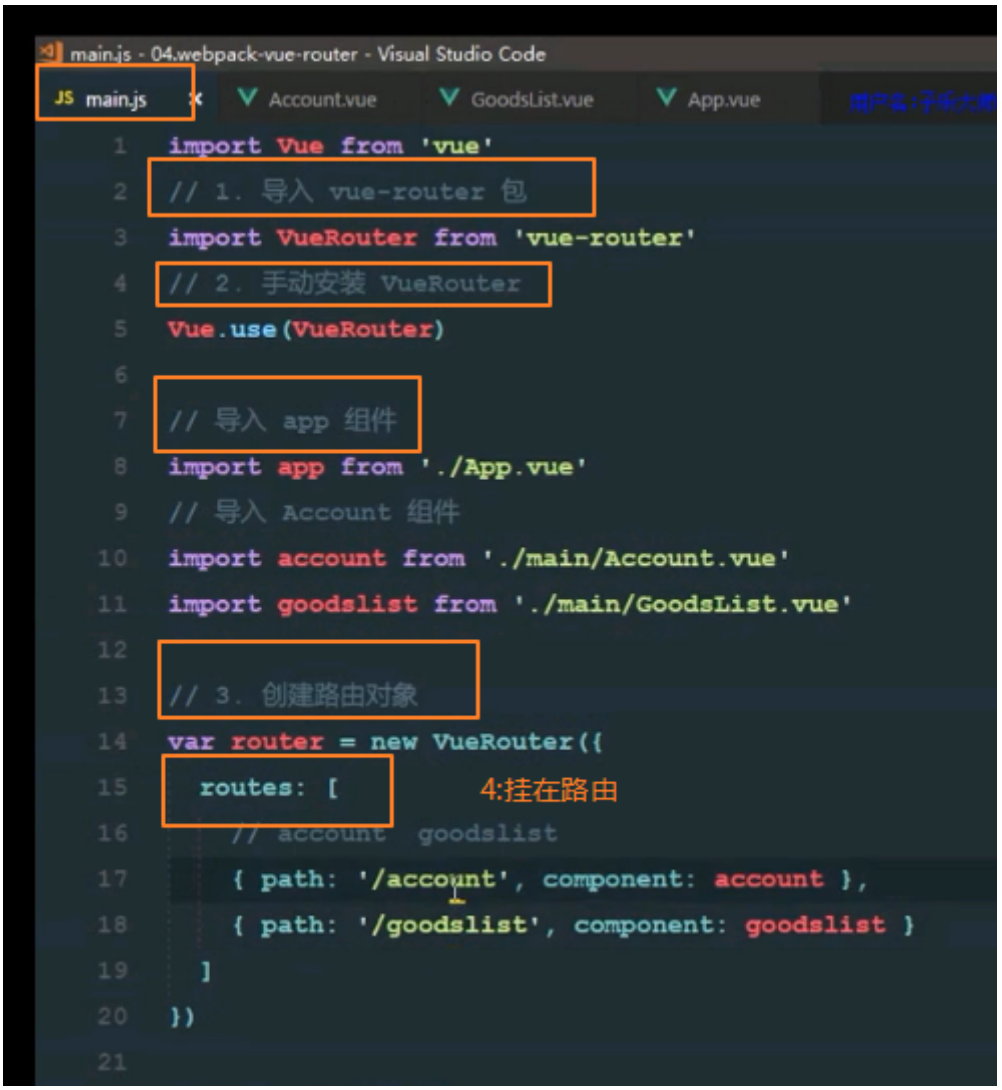
如果在一个模块化工程中使用它，必须要通过 Vue.use() 明确地安装路由功能：

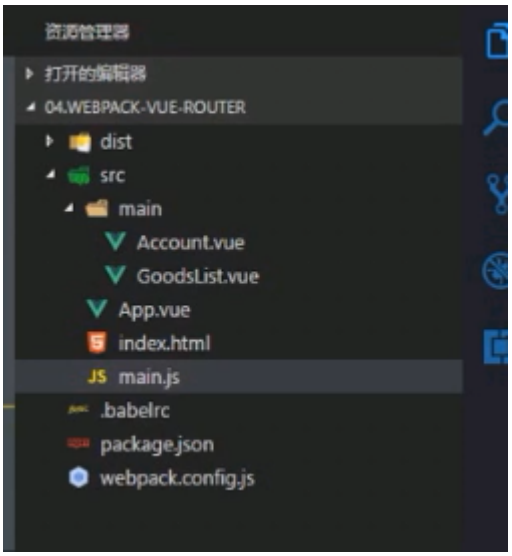
```
import Vue from 'vue'
import VueRouter from 'vue-router'

Vue.use(VueRouter)
```

如果使用全局的 script 标签，则无须如此（手动安装）。

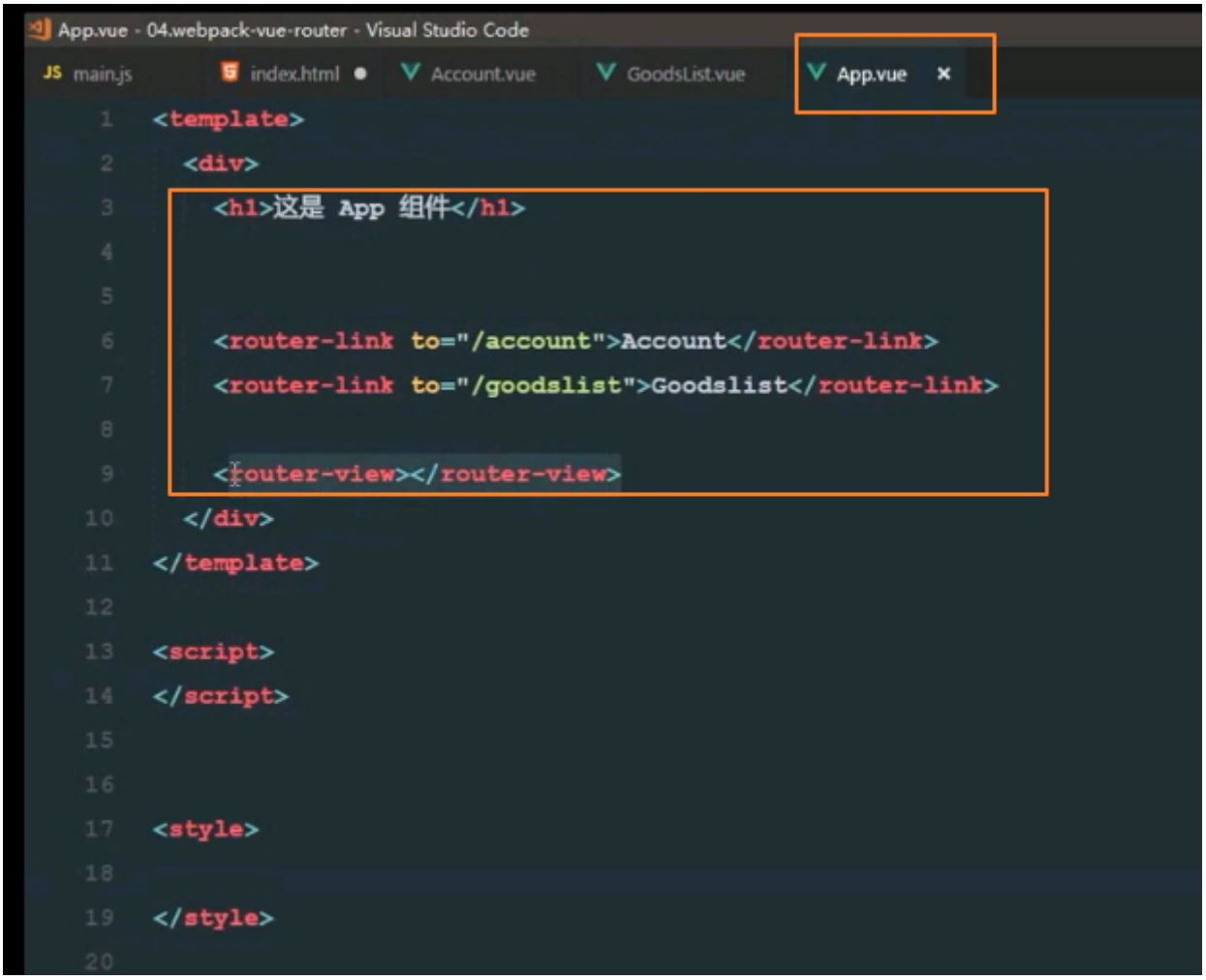
3:






需要注意的是：

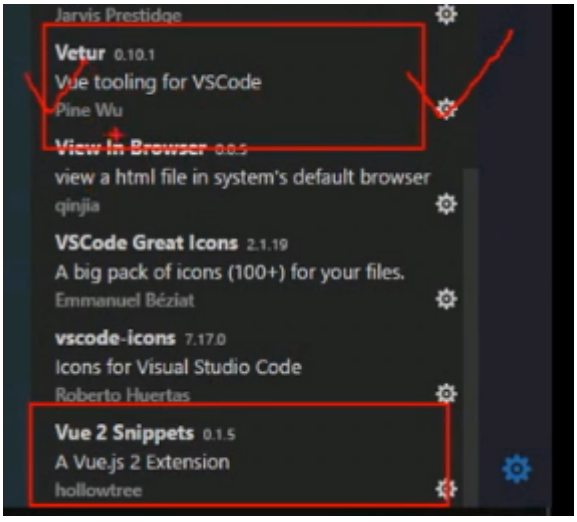
App 这个组件，是通过 VM 实例的 render 函数，渲染出来的， render 函数如果要渲染组件， 渲染出来的组件，只能放到 el:'#app' 所指定的 元素中；Account 和 GoodsList 组件， 是通过 路由匹配监听到的，所以， 这两个组件，只能展示到 属于 路由的 <router-view></router-view> 中去；



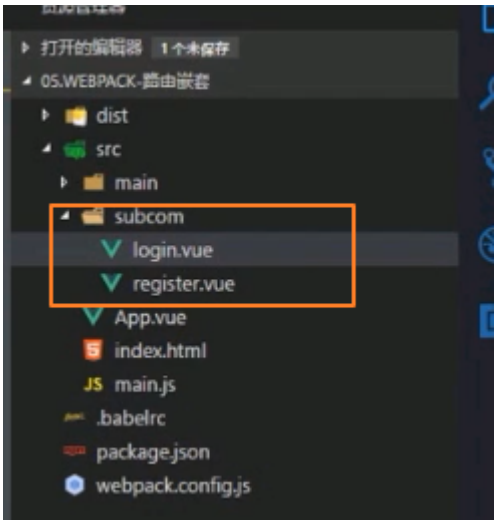
```
var vm = new Vue({
  el: '#app',
  render: c => c(app),
  // render 会把 el 指定的容器中，所有的内容都清空覆盖，所以 不要 把 路由的 router-view
  //  和 router-link 直接写到 el 所控制的元素中
  router // 4. 将路由对象挂载到 vm 上
})
```

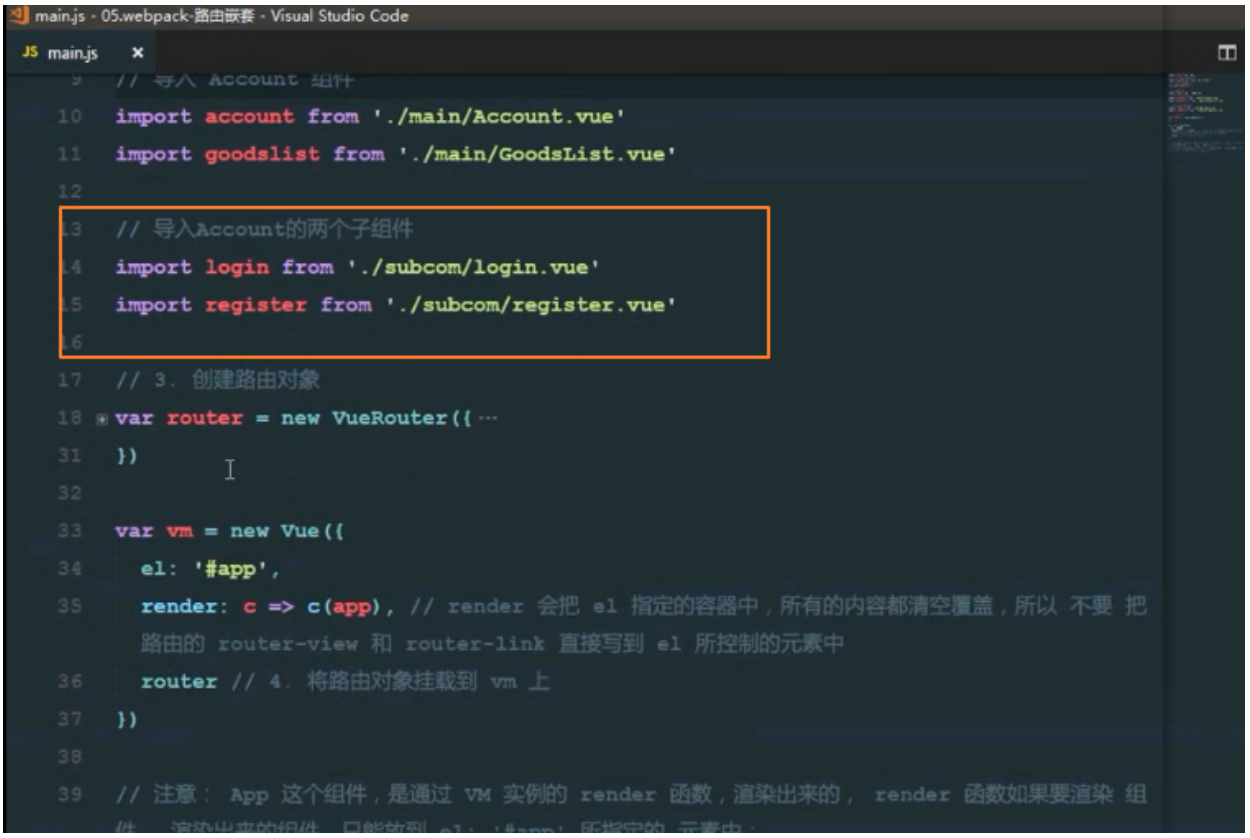
# 2：路由嵌套

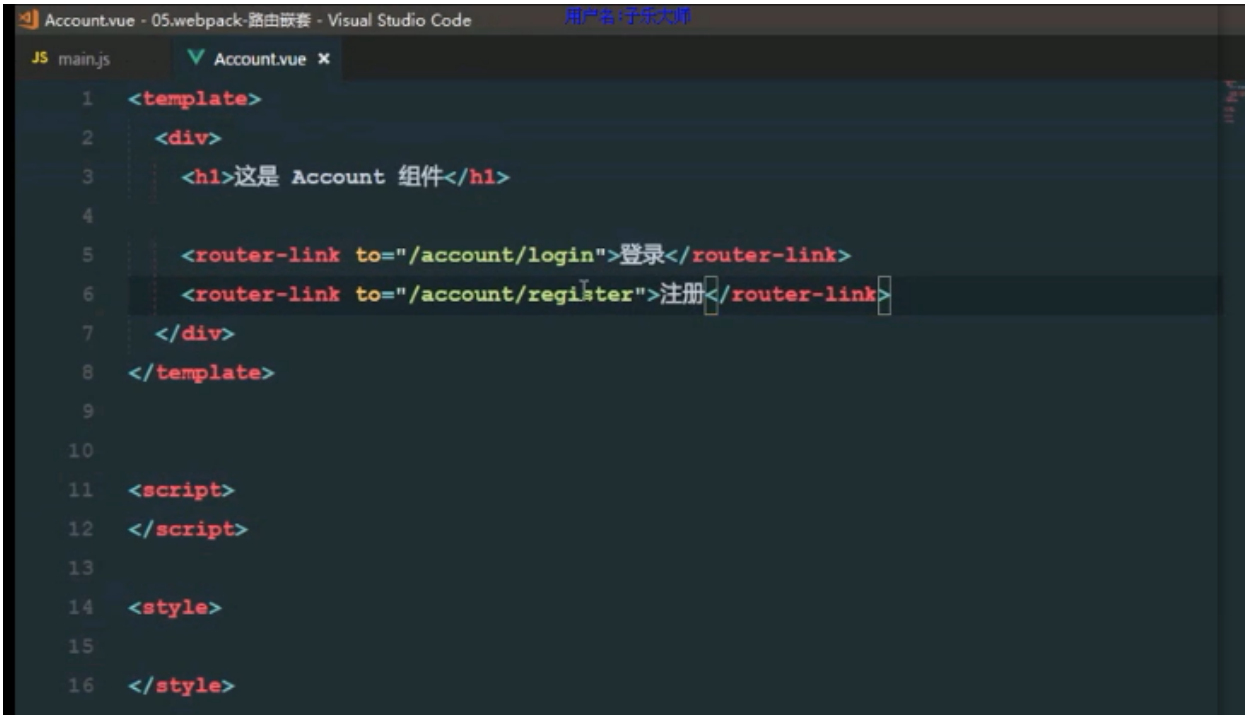
1：两个辅助的插件




2：目录结构





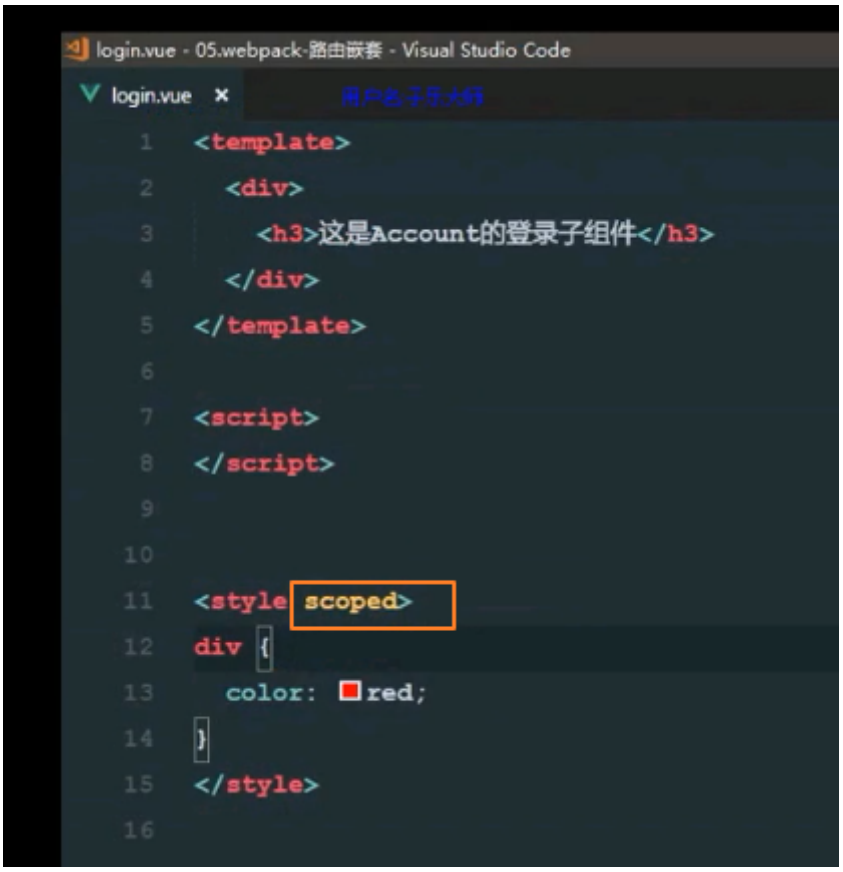



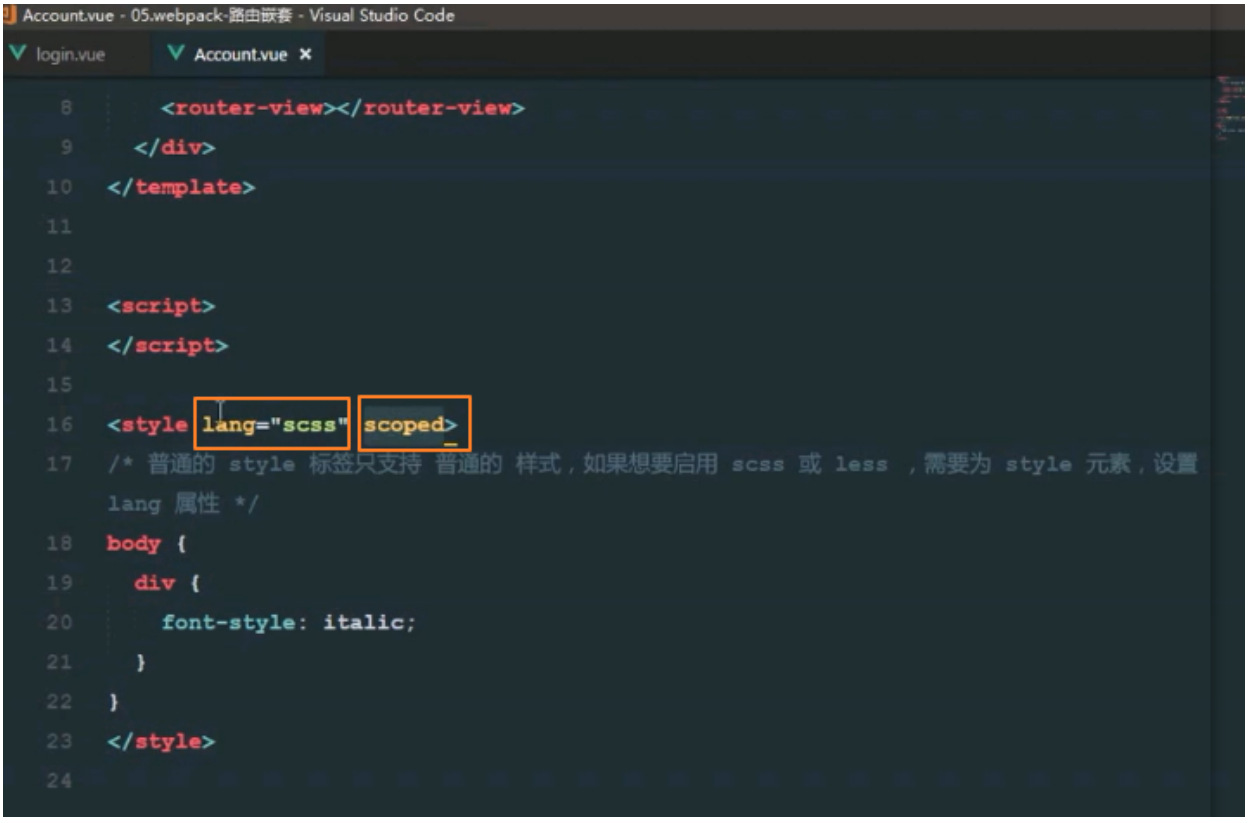

子组件样式：关键字 scoped　这样样式就只会给自己子组件加了，不然就会给所有的选择的标签去加

还有lang 属性：

普通的 style 标签只支持 普通的 样式，如果想要启用 scss 或 less ，需要为 style 元素，设置 lang 属性

只要 咱们的 style 标签， 是在 .vue 组件中定义的，那么，推荐都为 style 开启 scoped 属性

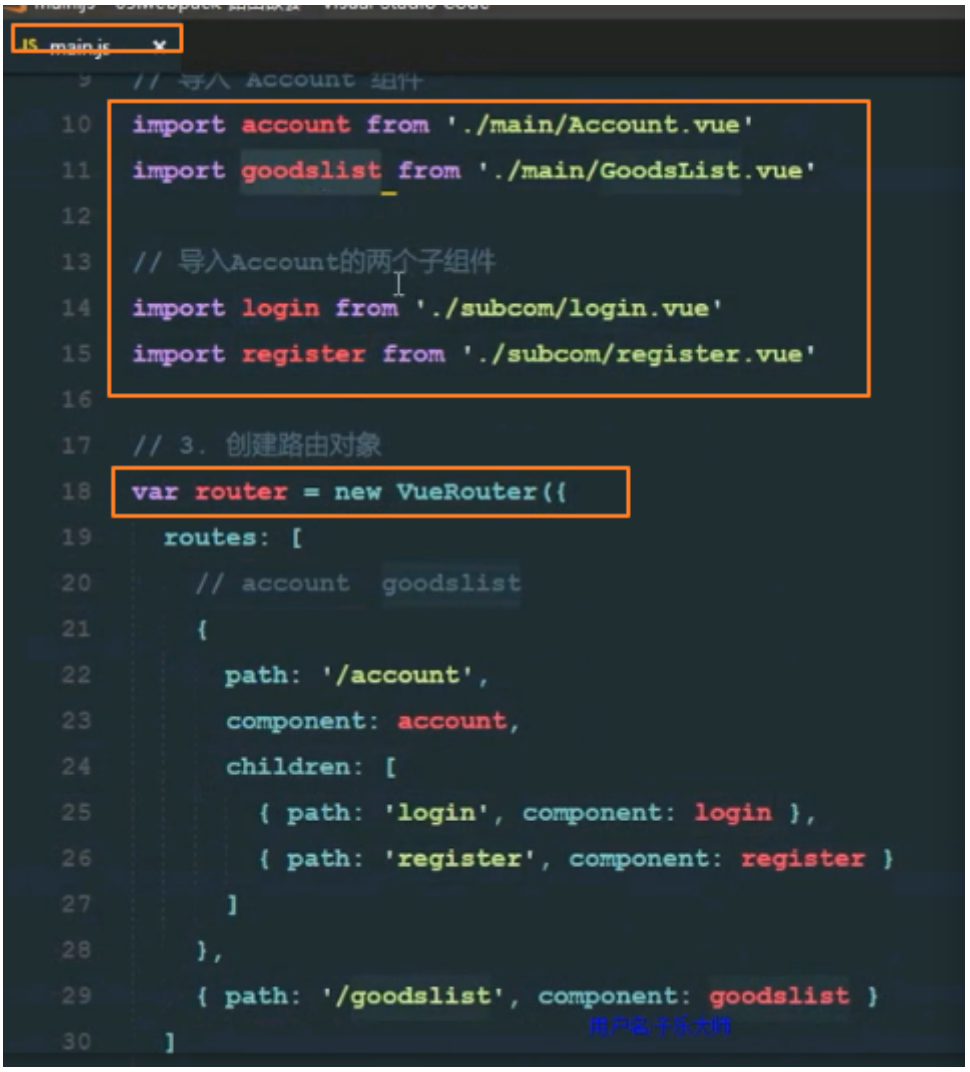


# 3：抽离路由模块

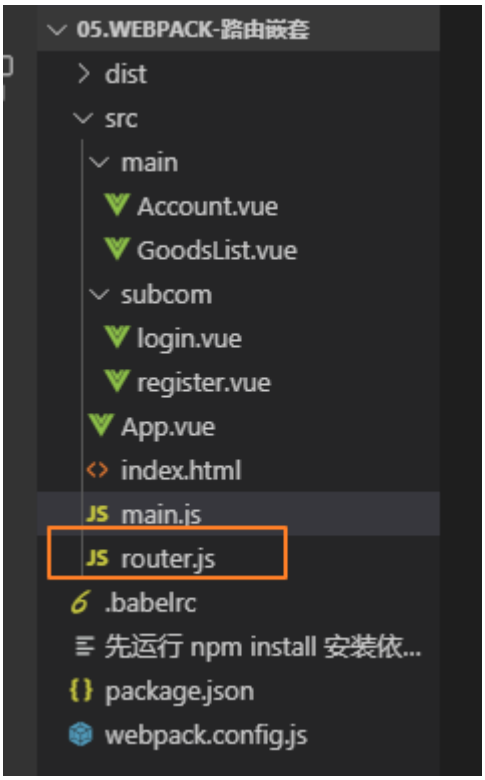
把导入需要依赖的模块

和 创建的路由对象模块抽离出去就可以




把如上的代码抽离出来，然后

当前的项目结构：




在main.js文件中


在router.js文件中


login.vue
Account.vue
register.vue
App.vue
src > JS router.js > ...
import VueRouter from 'vue-router'
// 导入 Account 组件
import account from './main/Account.vue'
import goodslist from './main/GoodsList.vue'
// 导入Account的两个子组件
import login from './subcom/login.vue'
import register from './subcom/register.vue'
// 3. 创建路由对象
var router = new VueRouter({
routes: [
// account goodslist
{
path: '/account',
component: account,
children: [
{ path: 'login', component: login },
{ path: 'register', component: register }
]
},
{ path: '/goodslist', component: goodslist }
]
})
// 把路由对象暴露出去
export default router