# 总结

1. 准备开发环境

1、使用 vue ui 配置，element插件



2、在localhost:8000端口，页面上可以配置element ui，element在 @vue/cli 脚手架中，使用插件的形式。进行右下角安装。

安装选择，配置按需导入

1. 装一下axios运行依赖

4、数据库的准备

装一下phpstudy2018，把vue-api-server/db/mydb.sql文件导入一下，密码root,数据库名字也要叫mydb，打开数据目录，在mydb目录下，能看到一些信息就导入成功。

5、post测vue-api-server服务器

（1）先vue-api-server，安装好依赖。使用cmd执行npm install

（2）node app.js，把api接口项目跑起来

（3）看电商管理后台API接口文档。

（4）测试登录的接口（post方式）

首先先复制根路径http://127.0.0.1:8888/api/private/v1，然后再接其他的参数login

参数在body中，username=admin,password=123456，选择x-www-form-urlencoded

有登录成功提示

路径如果出错的话，会是无效的token（忘了加login）。

6、现在@vue/cli项目中，初始化一下，app router.js中的内容清理一下，view删除，components下的hello.vue删除。

7、如果在创建@vue/cli项目的时候，有一些选项没有选，在后期是可以通过vue ui安装，如less-loader和less，开发依赖，它们是一起才有用的。

注意：重新运行整个项目。

8、assets放css全局样式文件，写完之后，把边距清除一下，需要在main.js中全局引入

二、流程

1、登录/退出功能

（1）登录业务流程

① 在登录页面输入用户名和密码

② 调用后台接口进行验证

③ 通过验证之后，根据后台的响应状态跳转到项目主页

（2）登录业务的相关技术点

http 是无状态的

通过 cookie 在客户端记录状态

 通过 session 在服务器端记录状态

 通过 token 方式维持状态

三、出现问题、疑问

1、element为啥是插件

答：比如vue-cli-plugin-element这个插件就依赖于vue-cli。它的作用是使这个软件在调用依赖包中的功能时更加的方便。能可以对element做一些配置

2、token跨域以及使用

有跨域问题使用token来维持状态，为什么跨域不能使用cookie和session。

它们两无法跨域共享，如果有必要，需要其他方式去操作。

每个不同的用户token是不同的，身份校验用户是否存在。

退出功能实现原理

基于 token 的方式实现退出比较简单，只需要销毁本地的 token 即可。这样，后续的请求就不会携带token ，必须重新登录生成一个新的 token 之后才可以访问页面。

3、redirect重定向是值是，路径？？？不是组件？？？

答： 是路径，官网文档中并没有看到组件

4、组件的data，不在return里边，还能写函数？？？

答：它只是定义了函数，在return的时候，把函数名称return出去了，相当于是定义了对象

5、\*清空数组，xx.length = 0跟指向空数组的效果不一样

为什么.length清零，不会马上清除？？？待后续研究。答：这个在官方文档中已经有说明，它不会动态更新的，非响应式。

6、保存服务器返回的那串图片的字符串地址是干嘛？？？

图片在服务器端没有删除吗，还是没有提交----它应该是在最终完成的时候，会一起提交上去？是这样？是的，在添加商品那里，会把图片路径传上去，但是传到哪里去，具体不知。

四、补充其他知识点

1、开发环境与生产环境

开发环境 = 生产依赖 + 开发依赖

生产环境 = 生产依赖

2、运行依赖是指你的项目在运行时候 需要用到的（比如vue）。开发依赖，是你的项目在编译构建的时候用到的（比如eslint等，辅助开发效率）

3、路由导航守卫控制访问权限----“导航”表示路由正在发生改变。

如果用户没有登录，但是直接通过URL访问特定页面，需要重新导航到登录页面。

（1）beforeEach它是一直在轮询的吗？？？

答：当一个导航触发时，全局前置守卫按照创建顺序调用。就是路由发生变化就触发一次

（2）单写next('/login')，为什么会一直执行呢？？？

答：单写next('/login')时候，会导致它跳转到login，又触发导航守卫，一直循环触发，具体它的工作原理后续继续研究？？？

注意是to.path，写to的话，会导致程序溢出。

4、居中

margin: auto

top\bottom\left\right\ 0

position:absolute

5、axios知识点

通过axios请求拦截器添加token，保证拥有获取数据的权限。

相当于是预处理这次的请求，给“请求头对象”添加 Token验证的Authorization

axios.interceptors.request.use(config => {

最后必须返回config，它的属性有请求头，他原本是没有Authorization属性的，就给它配置一下

})

config代表的是什么？？？答：做一些配置的具体也怎么回事，后续研究



6、对组件使用true或者false

如果它的元素的属性要布尔值的话，要使用v-bind才能给false，你不是用v-bind直接给false，这个false是一个字符串代表的是true。最好是直接加v-bind

7、v-if、v-else-if、v-else

<div v-if="type === 'A'">

A

</div>

<div v-else-if="type === 'B'">

B

</div>

<div v-else-if="type === 'C'">

C

</div>

<div v-else> // 这里不需要加条件

Not A/B/C

</div>

8、插槽的作用，template v-slot

在template中，一般通过插槽来获取数据，不是去请求

在渲染添加参数的切换中，把数据保存到它的数据中，通过scope再去获取，不然就同步操作。

要想获取焦点得通过原生dom对象去处理。

9、keyup.enter.native的理解。

答：在监听键盘事件时，我们经常需要检查详细的按键。Vue 允许为 v-on 在监听键盘事件时添加按键修饰符：

只有在 key 是 Enter 时调用 vm.submit()。Vue 提供了绝大多数常用的按键码的别名：

.enter

.tab

.delete（捕获删除或者“退格键”）

.esc

.space

.up

.down

.left

.right

加上native，子组件直接就能用触发这个事件，就不用使用emit去触发。通过 $emit 事件派发给父组件也是可以的

10、空字符串.split(' ')得到不是空数组

11、\*this.$nextTick(\_ => {

该方法的作用，当页面上元素被重新渲染的之后，才会执行指定回调函数中的代码，在true渲染的时候，可能没渲染出来，但是你就去获取它的对象，这样会获取不到

// 获取焦点，否则还要点击input

this.$refs.saveTagInput.$refs.input.focus()

})

12、\*存在未解决的问题，在tab参数的添加方面卡了三个小时，一直还找不到问题的所在点，待确认？？？

处理tag渲染的问题，初步找到问题的代码，理解多了一些响应式的原理，数组和对象的特别处，感觉还是有点差距。

已经搞定了，是数组里边的对象的问题，vue不能检测对象属性的动态添加，一次性加完后，再给this.xxdata就行，如果用this.xxdata那么，vue是无法对这个添加的属性进行响应的，就是在视图上没有响应，不是数组的问题，注意力放在数组身上。以为数组没有问题，里边的对象就没有问题。



13、loadsh生产依赖用来深拷贝

对象的赋值不能用函数吗----可以，是自己搞错，它本来就是字符串了

await忘记写了，导致请求不到数据

14、异步方法的使用：在echart合并对象的时候

问题：在created中去用“异步”方法去请求数据，误区：以为created执行完毕之后，数据就请求到了。

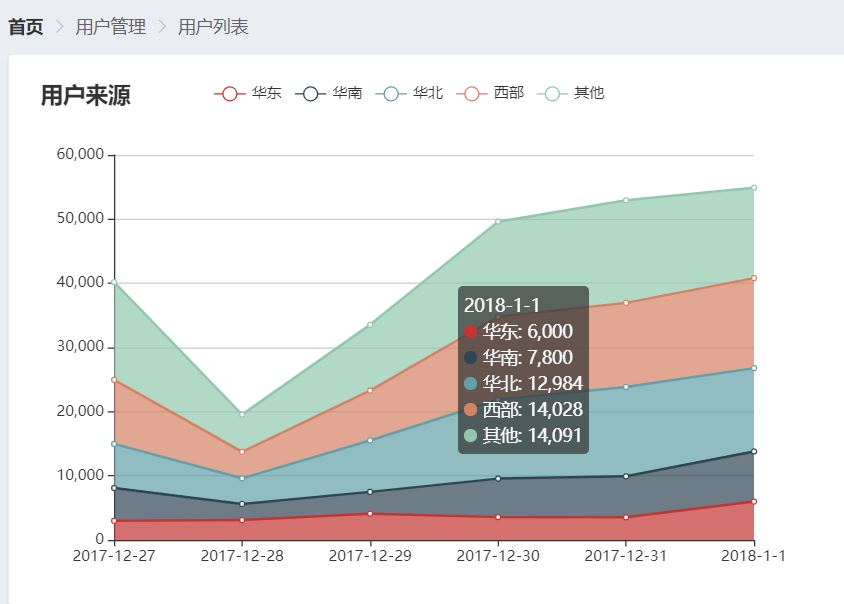
在mounted中去，把认为请求到的数据，跟合并的选项，就去合并，然后去设置echart数据。结果现象一直看不到。

解决方法：是去对了源码

异步的方法它是要在主线程执行完毕后，才轮到异步去执行，此时mounted中合并的时候，它是同步的，此时就异步的数据还是空的，合并后去设置echart数据，此时的设置是有问题的。再加了打印，看打印的顺序，验证了想法是没有问题的

解决方法，要把请求数据放到mounted中，mounted设置成异步的方法，让它内部以同步的方式去执行，await等到它数据请求到了，再一起去设置echart数据，这样就ok了

之前都没有遇到过，都是异步请求后，跟数据双向绑定的，此时不会出现这个问题。



1. element ui操作

组件样式代码，复制下来后，一些不必要的属性删除，确定没有影响在删，不确定最好别删。先看官方说明，看意思。

在plugin目录下，导入需要用到的组件。

在 Form 组件中，每一个表单域由一个 Form-Item 组件构成，表单域中可以放置各种类型的表单控件，包括 Input、Select、Checkbox、Radio、Switch、DatePicker、TimePicker

el-form-item加了定位会变小。加了之后变成行内块元素，然后由input撑开，input是有默认的宽度（没有设置情况下）。

字体图标用的是阿里的，引入字体图标的时候，不要加一点

element ui中的一些绑定的数据，最好跟它一致，不然会出现奇奇怪怪的问题。在form中没有添加：model验证是有问题的。

prop绑定的类要与el-form-item下的v-model的值相对应。我们将其做个修改便可以正常纠错了。

element ui它提供把子菜单转换成router-link的操作，我们就不用去改造，然后index就成了路由地址。服务器它给了我们这个地址。

element ui中的表单验证它是失去焦点才会触发的。就是要去点击一下，直接修改数据是不会触发的。所以每次取消操作都要清空恢复到最开始的状态。不然上次的如果不合格，它的还会显示不合格。

注意要.comfirm，Vue.prototype.$confirm = MessageBox.confirm

element ui渲染表格时候，应该是这样的，先对数组中的数据进行循环，然后把数据存到子组件的中去，这样点击时候，才知道能获取到某个组件的id。

居中的话flex似乎用的比较多

有个小问题：就是唯一的三级没有选，但是在下拉中可以看到，在对话框设置权限那就看不到？？？这个感觉有bug。

单选的话，前边的小圆点后续可以想着去掉，获取用别的版本测试，不同版本之间会有差异

六、其他注意点

在写methods，无缘无故自动加了一句话，导致浪费了很多时间。醉了

在使用axios的时候，this.$http.post中的post忘了写。

在点击左侧菜单栏的时候，会突出一小部分，要去通过样式F12看是什么引起的。是最外圈的盒子引起的，不要只看到一部分。

nprogress：在导航切换的时候，出现一个进度条的效果，在axios中request和response之间会显示，现象短暂，目前没有观察到。

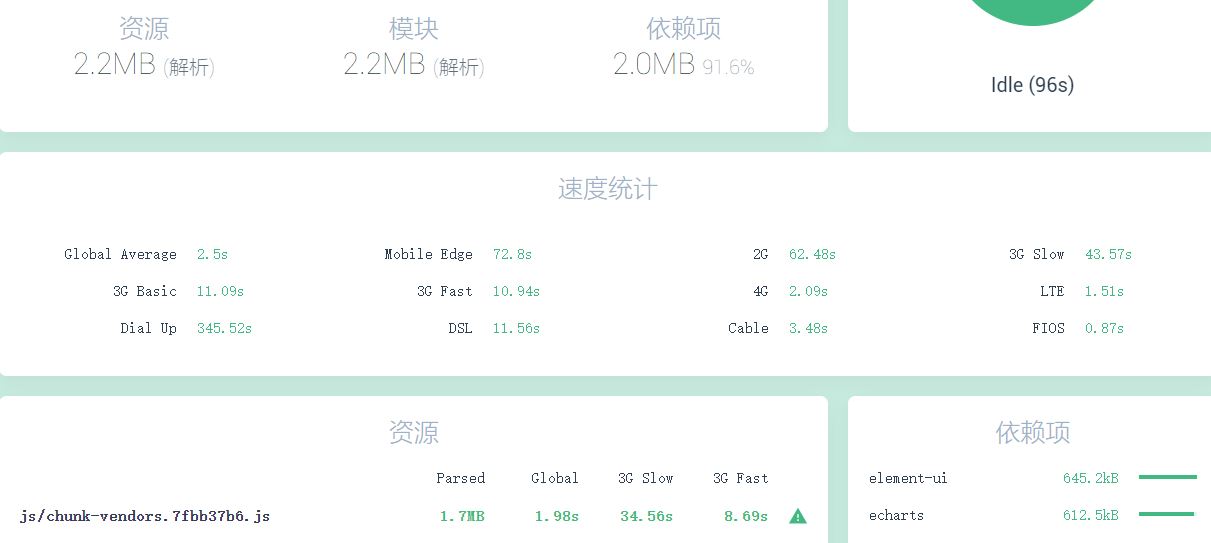
没有观察到是忘了加nprogress样式，加完后，有个蓝色的进度条，它是伪造的效果，看起来像是加载。

在build阶段，可能会出现一些console，这些是不需要，此时我们需要使用插件来处理

npm install babel-plugin-transform-remove-console --save-dev

获取 mode的值，看ui编译命令行，

打包报告：可以看到哪些文件体积大，占的比例，打开的速度，可以直观看到依赖性占了９０％



externals,引入cdn资源，然后它会去cdn上找，并不会把那些vue.js一起打包，需要在发布模式中，配置哪些需要配成cdn



在index.html中配置cdn

在本地是无法进行测试的，因为那些数据库文件是放在本地上的phpstudy中。

在node测试dist不要使用80端口

element ui 的cdn去官网使用最新的

5、ESLint语法检查规范

（1）声明但是未使用的变量会报错

（2）空行不饿能连续大于等于2

（3）在行结尾处，多余的空格不允许

（4）多余的分号，不允许

（5）字符串要使用单引号，不能使用双引号

（6）在方法名和形参列表的小括号之间，必须要有一个空格

（7）在单行注释 // 之后，必须有一个空格

（8）在每一个文件的结尾处，必须要有一个空行

（9）import语句必须放到最顶部