**一、项目搭建**

1.首先去官网下载node.js，当前该版本为12.13.1 LTS，建议不装13.1.0Current，因为没有LTS稳定

官网地址：https://nodejs.org/en/

（1）安装后检查是否安装成功：cmd输入命令 node -v ，如果输出版本号，说明我们安装node环境成功。

（2）安装该版本后，会自动装上对应的npm，版本为6.12.0

可以在cmd输入命令 npm -v

（3）npm安装插件的目录，一般不在nodejs的安装目录下，而是在用户目录下

如我的电脑是：C:\Users\1\AppData\Roaming\npm

也可以通过如下命令，检查本机npm的下载插件目录

命令行：npm config get prefix

结果：C:\Users\1\AppData\Roaming\npm

对cnpm，也可以通过

命令行：cnpm config get prefix

结果：C:\Program Files\nodejs

特别说明：如果修改错路径会无法使用npm，这个时候不用重装，只要到

C:\Users\1路径下看后缀名为.npmrc的文件，用记事本打开修改里面的路径即可

如：

prefix=C:\Program Files\nodejs\node\_global

cache=C:\Program Files\nodejs\node\_cache

2.为了提高下载效率，可以使用淘宝的镜像，从国内下载

在CMD输入命令：npm install -g cnpm –registry=https://registry.npm.taobao.org

即可安装npm镜像，以后再用到npm的地方直接用cnpm来代替就好了。

检查是否安装成功：cmd输入命令 cnpm -v ,不报错即为安装成功。

当前安装版本为6.1.0

移除cnmp：npm uninstall cnpm -g

3.全局安装webpack

命令：cnpm install webpack -g

4.安装webpack-cli（自己的系统未安装）

cnpm install webpack webpack-cli -g

5.安装vue-cli

传统版本：cnpm install vue-cli -g

6.利用vue-cli初始化vue项目

可以先设置NPM，否则速度可能很慢，甚至卡死

npm config set registry https://registry.npm.taobao.org再创建，但是后面记得切换回来，因为有些插件cnmp安装有问题

切换回：

npm config set registry https://registry.npmjs.org/

查看当前的源：

npm get registry

在CMD中先切换到想要创建项目的目录，输入命令：vue init webpack projectname

其中Projectname表示项目名，自己变更

Project description 项目描述

Author 作者

Vue build 构建模式，一般默认选择第一种

7.创建好后，可以将项目拷贝到指定目录，然后用工具打开

（1）在webstorm中，打开编辑器，选择file，选择open，打开d盘的项目文件

然后在webstorm下面的控制台中输入：npm run dev

即可启动项目

附加工具：

8.关闭命令

CTRL+C

如果有时候失效，或是命令窗口已经关闭，就要找另外的方式关闭

二、Vergil / VUE3.0+ElementUI搭建脚手架（参考https://gitee.com/bolat/codes/xq81omcy9hf46a3spt7jv49）

1.若系统中存在vue1.x或vue2.x，需要先卸载（如果安装了上述的第一步，必须要执行这步）

npm uninstall vue-cli -g

yarn global remove vue-cli

（1）说明：某次卸载的时候，出现

npm ERR! A complete log of this run can be found in:

npm ERR! C:\Users\1\AppData\Roaming\npm-cache\\_logs\2020-02-19T09\_03\_53\_915Z-debug.log

解决方法（ljy亲测可行）：清除npm缓存，使用命令

npm cache clean --force

2.卸载完完毕后，安装新版本脚手架

npm install -g @vue/cli

3. 安装yarn（网上经常忽略这步，注意安装）

npm install -g yarn

4.脚手架安装完成后创建项目

参考：<https://www.cnblogs.com/Jimc/p/10278254.html>

vue create + 项目名称

说明：

特别注意，如果电脑安装了大数据的hadoop-如ljy，会在vue create创建项目的时候失败，网上没有相应的资料，必须要在执行这个命令前先让本机安装的hadoop失效（通过改hadoop目录即可让相应的yarn指令失效，避免与vue的yarn指令冲突），安装项目后可以再将hadooop目录改回

5.选择配置项，该步骤可后续调整

默认ESlint下一步即可

6.可视化界面

vue ui

然后在浏览器输入

localhost:8000

即可打开

7.项目操作

（1）启动项目

yarn serve

（2）项目打包

yarn build

备注：打包说明

a.如果放在二级目录下，也要修改配置项

b.新建vue.config.js文件，添加配置项：publicPath: ‘./’

module.exports = {

publicPath: './',

}

（3）

8.卸载该脚手架

npm uninstall @vue/cli –g

9.同时使用两个版本的vue

官网给了一个版本之前的桥接工具

npm install -g @vue/cli-init

**二. vue项目目录讲解**

1、build：构建脚本目录

　　　　1）build.js ==> 生产环境构建脚本；

　　　　2）check-versions.js ==> 检查npm，node.js版本；

　　　　3）utils.js ==> 构建相关工具方法；

　　　　4）vue-loader.conf.js ==> 配置了css加载器以及编译css之后自动添加前缀；

　　　　5）webpack.base.conf.js ==> webpack基本配置；

　　　　6）webpack.dev.conf.js ==> webpack开发环境配置；

　　　　7）webpack.prod.conf.js ==> webpack生产环境配置；

　　2、config：项目配置

　　　　1）dev.env.js ==> 开发环境变量；

　　　　2）index.js ==> 项目配置文件；

　　　　3）prod.env.js ==> 生产环境变量；

　　 3、node\_modules：npm 加载的项目依赖模块

　　 4、src：这里是我们要开发的目录，基本上要做的事情都在这个目录里。里面包含了几个目录及文件：

　　　　1）assets：资源目录，放置一些图片或者公共js、公共css。这里的资源会被webpack构建；

　　　　2）components：组件目录，我们写的组件就放在这个目录里面；

　　　　3）router：前端路由，我们需要配置的路由路径写在index.js里面；

　　　　4）App.vue：根组件；

　　　　5）main.js：入口js文件；

　　5、static：静态资源目录，如图片、字体等。不会被webpack构建

　　6、index.html：首页入口文件，可以添加一些 meta 信息等

　　7、package.json：npm包配置文件，定义了项目的npm脚本，依赖包等信息

　　8、README.md：项目的说明文档，markdown 格式

　　9、.xxxx文件：这些是一些配置文件，包括语法配置，git配置等

**三、在项目执行的安装**

1.搭建element ui-ljy亲测（注意，网上很多资料容易出错，参考资料网址：https://www.cnblogs.com/yuyujuan/p/10267749.html）

（1）在前面两部已经创建好的项目路径下（webstrom直接在项目底下的控制台-Terminal输入下面命令）

命令：cnpm i element-ui -S

备注（卸载element ui）：npm uninstall element-ui

（2）在项目src/main.js中加上下面代码，引入element ui插件

import Element from 'element-ui'

import 'element-ui/lib/theme-chalk/index.css';

Vue.use(Element)

2：如果要进行前后端交互，可以安装工具anios

项目目录下执行：

方法1：cnpm install --save axios

方法2：cnpm install axios

（1）备注：检查一下安装的cnpm文件与npm是不是在node根目录，否则安装时会出错（ljy就在这里出错）

3.安装vuex

（1）在项目下输入命令

cnpm i vuex --save

4.安装scss

（1）在项目目录下输入下面两条命令

npm install sass-loader --save-dev

npm install node-sass --sava-dev

（2）在build文件夹下的webpack.base.conf.js的rules里面添加配置

{

test: /\.scss$/,

loaders: ['style', 'css', 'sass']

}

（3）使用scss时候在所在的style样式标签上添加lang=”scss”即可应用对应的语法，否则报错

例如：

<style lang="scss">

$color:blue;

div {color:$color;}

</style>

（4）特别说明：ljy在运行项目时，虽然scss已经不在文件中报错，但是运行时项目会出现

Module build failed: TypeError: this.getResolve is not a function at Object.loader

原因：因为当前sass的版本太高，webpack编译时出现了错误，这个时候只需要换成低版本的就行

解决办法1：找到package.json文件，里面的 "sass-loader"的版本更换掉 就行了。

例如本项目：将 "sass-loader": "^8.0.0"，更换成了 "sass-loader": "^7.3.1"。然后会提示重新npm下载，按照提示下载7.3.1即可

解决方法2：先卸载当前版本，然后安装指定的版本

卸载当前版本 npm uninstall sass-loader

安装 npm install sass-loader@7.3.1 --save-dev

5.安装ueditor（难点）-ljy根据自己项目搭建编写，因为服务端较复杂，所以能按照这个搭建好基本够用了。

参考：

https://www.cnblogs.com/ocean-sky/p/7132319.html

<https://blog.csdn.net/haochuan9421/article/details/81975966>

说明：在打包为项目后，在二级目录会出现一些小BUG，详见打包。

存在缺陷：如果网站正在网络加载（如图片），那么ueditor要加载完后才会显示

6.安装echarts

参考：https://www.jianshu.com/p/cf0a54374419

(1)安装

cnpm install echarts -S

（2）在main.js中

// 引入echarts

import echarts from 'echarts'

Vue.prototype.$echarts = echarts

7.安装Mint UI（如果需要移动端的话）

npm install mint-ui -S

// 引入全部组件

import Mint from 'mint-ui';

import 'mint-ui/lib/style.css'

Vue.use(Mint);

**四、打包**

**打包命令：**npm run build

特别注明：以下情况是建立在开发环境在一级环境，生产环境在二级环境的前提下

不过在打包以前，要先注意根据以下情况进行检查和修改。

1.打包后在项目运行，提示找不到js等文件。

解决办法：先将config/index.js中第二个assetsPublicPath: '/',改成assetsPublicPath: './',加一个点（该方法针对打包，修改后，在开发环境下并不影响运行）

步骤如下：

（1）找到config里面的index.js，找到下面这段代码：

build: {

// Template for index.html

index: path.resolve(\_\_dirname, '../dist/index.html'),

// Paths

assetsRoot: path.resolve(\_\_dirname, '../dist'),

assetsSubDirectory: 'static'

assetsPublicPath: '/',

/\*\*

\* Source Maps

\*/

productionSourceMap: true,

// https://webpack.js.org/configuration/devtool/#production

devtool: '#source-map'

（2）将assetsSubDirectory: 'static'下面的代码

assetsPublicPath: '/',

改成

assetsPublicPath: './',

（3）加入后的代码如下

build: {

// Template for index.html

index: path.resolve(\_\_dirname, '../dist/index.html'),

// Paths

assetsRoot: path.resolve(\_\_dirname, '../dist'),

assetsSubDirectory: 'static',

assetsPublicPath: './',

2.如果使用了element框架，并且用了icon图标，打包后会出现无法显示icon图标的情况

解决办法（该方法修改后，并不影响开发环境下的展示）：

在build下的utils.js文件中，加 publicPath: '../../'，步骤如下

（1）在utils.js文件中找到下述代码

// Extract CSS when that option is specified

// (which is the case during production build)

if (options.extract) {

return ExtractTextPlugin.extract({

use: loaders,

fallback: 'vue-style-loader'

})

} else {

return ['vue-style-loader'].concat(loaders)

}

}

（2）在代码行

fallback: 'vue-style-loader'

下加入代码

publicPath:'../../' //打包的时候添加，否则element ui的icon出不来

（3）最终代码如下：

// Extract CSS when that option is specified

// (which is the case during production build)

if (options.extract) {

return ExtractTextPlugin.extract({

use: loaders,

fallback: 'vue-style-loader',

publicPath:'../../' //打包的时候添加，否则element ui的icon出不来

})

} else {

return ['vue-style-loader'].concat(loaders)

}

}

3.如果使用了ueditor富文本框插件，并且在服务器是放在二级目录下（如：http://www.lcvc.cn/ebuy\_web\_vue）打包的时候要注意更改路径，否则会出现各种使用问题。

解决办法：

（1）在ueditor根目录下的ueditor.config.js中，找到下述代码

UEDITOR\_HOME\_URL: "/static/plugins/ueditor/"

然后改为

UEDITOR\_HOME\_URL: "/ebuy\_web\_vue/static/plugins/ueditor/"

说明：ebuy\_web\_vue为二级目录

特别说明（被坑了一天）：上述路径要在打包前修改好，然后执行打包命令会生成dist，将dist修改为ebuy\_web\_vue后再上传。否则上传到服务器后再改路径，容易因为缓存问题导

4.将项目打包为有项目路径的情况（并尝试去掉vue工程运行时，地址栏的#号）

参考文档：https://blog.csdn.net/qq\_34543438/article/details/90901440

（1）修改vue项目中config文件下index.js中打包配置：

如果是按照本次项目1.2节修改过，那么项目应该如下：

build: {  
 // Template for index.html  
 index: path.resolve(\_\_dirname, '../dist/index.html'),  
  
 // Paths  
 assetsRoot: path.resolve(\_\_dirname, '../dist'),  
 assetsSubDirectory: 'static',  
 //去掉#号需要  
 assetsPublicPath: './',

此时，我们把dist改为我们想打包的项目名，并做几处修改，修改后如下

build: {  
 // Template for index.html  
 index: path.resolve(\_\_dirname, '../report/index.html'),  
  
 // Paths  
 assetsRoot: path.resolve(\_\_dirname, '../report'),  
 assetsSubDirectory: 'static',  
 //去掉#号需要  
 assetsPublicPath: '/report/', // 这个地方使用绝对路径

（2）路由配置：router文件夹下index.js：

由于项目名我们统一打包为report，所以增加如下选项

mode:'history',

base:'report',

修改后的代码如下：

export default new Router({  
 mode:'history',  
 base:'report',  
 routes: [

注意：在打开项目后，如果是重新刷新页面，会出现404错误。这个要去服务端处理，个人觉得麻烦暂时不处理。

9.本项目的其他设置

（1）修改网络连接命令

在src/api/my/my\_variable.js，打包时需要修改下面的网址以适应实际需求

const baseUrl="http://127.0.0.1:8081";  
//const baseUrl="http://120.76.118.101:8081";