

前言

vue-cli 都到 3.0.3 了，所以是时候玩转一下 vue-cli 3 的新特性了。

1. vue-cli 3.0.3

以下的安装都是在 macOS 的环境下进行的，当然在 windows 和 linus 下也同理。

1.1 安装

vue cli 的包名称由 vue-cli 改成了 @vue/cli。 如果你已经全局安装了旧版本的 vue-cli (1.x 或 2.x)，你需要先通过 npm uninstall vue-cli -g 或 yarn global remove vue-cli 卸载它。

可以使用下列任一命令安装这个新 vue-cli 3.x 的包：

npm install -g @vue/cli

# OR

yarn global add @vue/cli

你还可以用这个命令来检查其版本是否正确 (3.x)：

vue --version

或者：

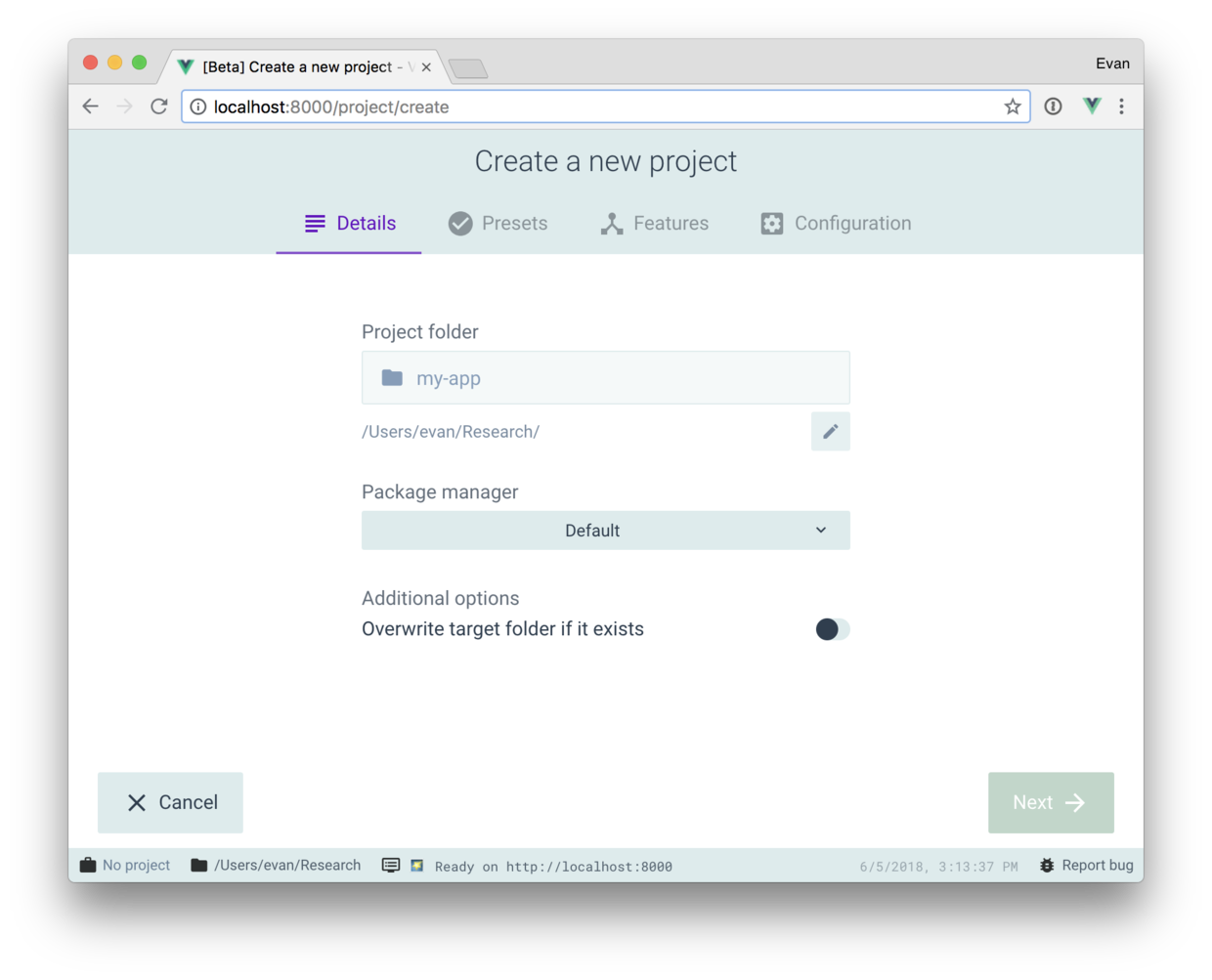
vue -V

1.2使用图形化界面

你也可以通过 vue ui 命令以图形化界面创建和管理项目：

vue ui

上述命令会打开一个浏览器窗口，并以图形化界面将你引导至项目创建的流程。



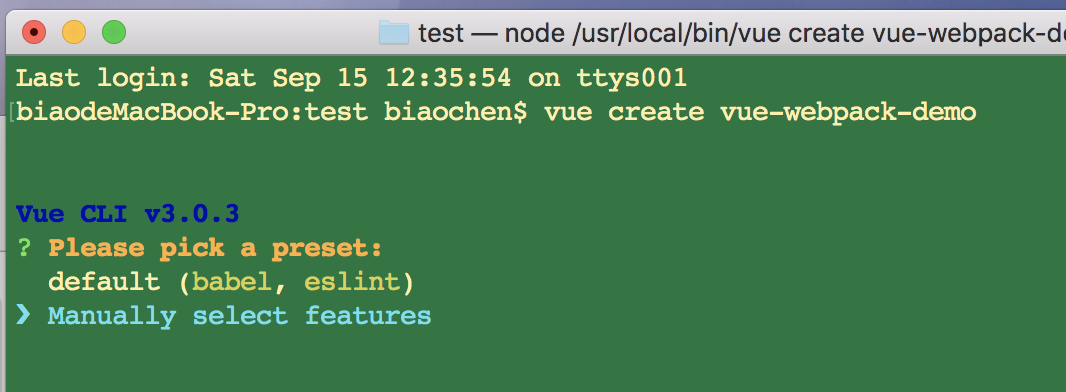
1.3 创建项目

1.3.1 默认型

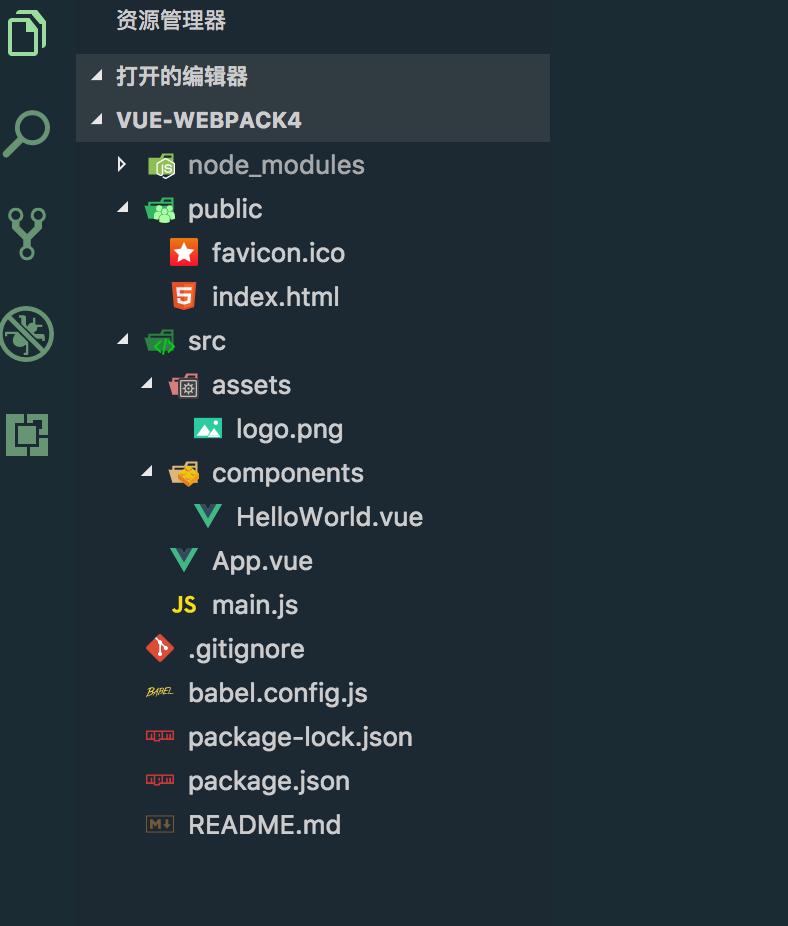
* 新建文件夹，在该文件夹下打开命令窗口，输入以下命令进行新建项目，当然我起的项目名字叫 vue-webpack-demo

vue create vue-webpack-demo

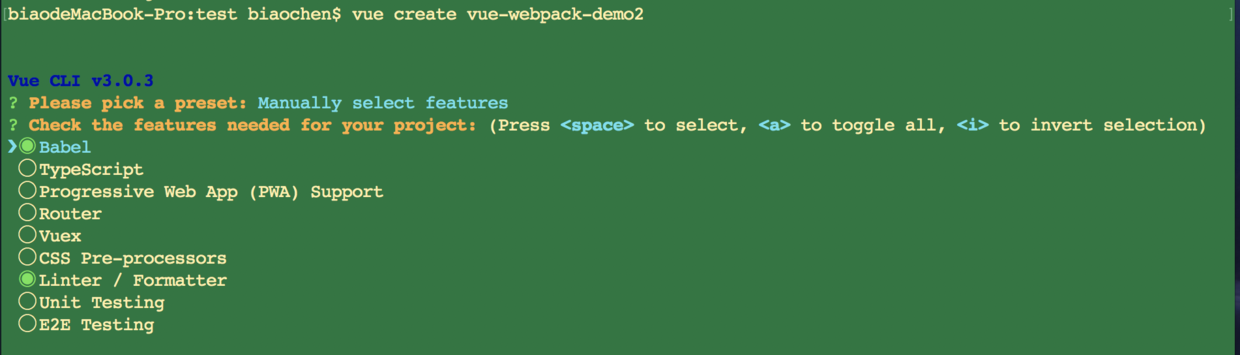
* 会让你选择默认（default）还是手动（Manually），（注：现在vue-cli3.0默认使用yarn下载）。



* 先是默认的，一路回车后的项目目录如下：



* 再来手动的，我起的项目名字叫 vue-webpack-demo2，如下图，让你选择那些选项，按 空格键 是选择单个，a 键 是全选。



* 我选择了常用的如下选项：



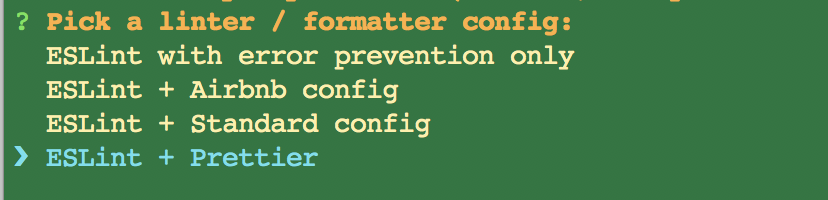
* vue-router 默认 hash 模式，所以我选择默认的，选择了 n ，而不是 history 模式：



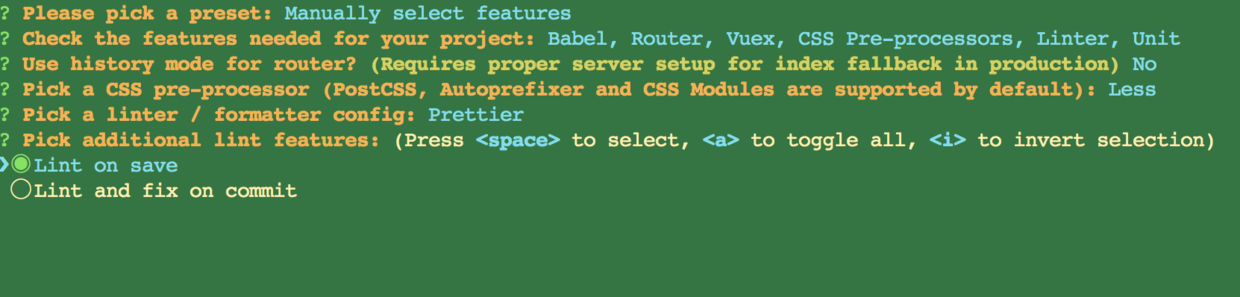
* 下一步之后问询问你安装哪一种 CSS 预处理语言，我是选择了用的 less。



* 这个是问你选择哪个自动化代码格式化检测，配合 vscode 编辑器的，Prettier - Code formatter插件，我选的随后一个。



* 第一个是保存就检测，第二个是 fix 和 commit 的时候检查。



* 选择单元测试解决方案，Mocha是流行的JavaScript测试框架之一，通过它添加和运行测试，从而保证代码质量，chai 是断言库，我两个都选择了。



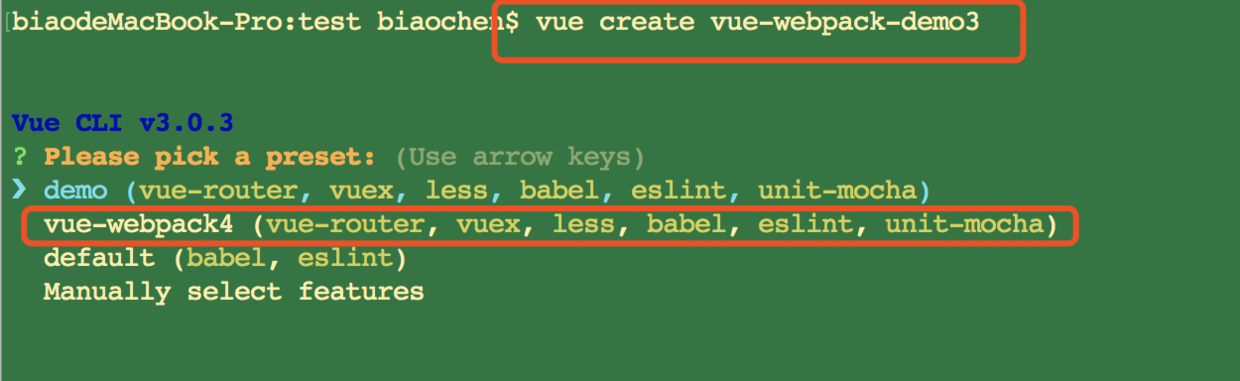
* 上边这俩意思问你像，babel, postcss, eslint 这些配置文件放哪？第一个是：放独立文件放置，第二个是：放package.json里，这里小汪选择放单独配置文件，选第一个



* 下面倒数第二行问你是否将以上这些将此保存为未来项目的预配置吗 ？选择是的时候，下次创建项目时，可以选择刚刚配置好的配置，不用再每个都配置一遍。最后一个是选择的名字，你随意选择，点击确定就开始下载模板了。



* 再创建项目的时候，刚刚配置好的选择的名字 vue-webpack4 会这样子出现：



* 启动命令

// 1. 进入项目

cd vue-webpack-demo

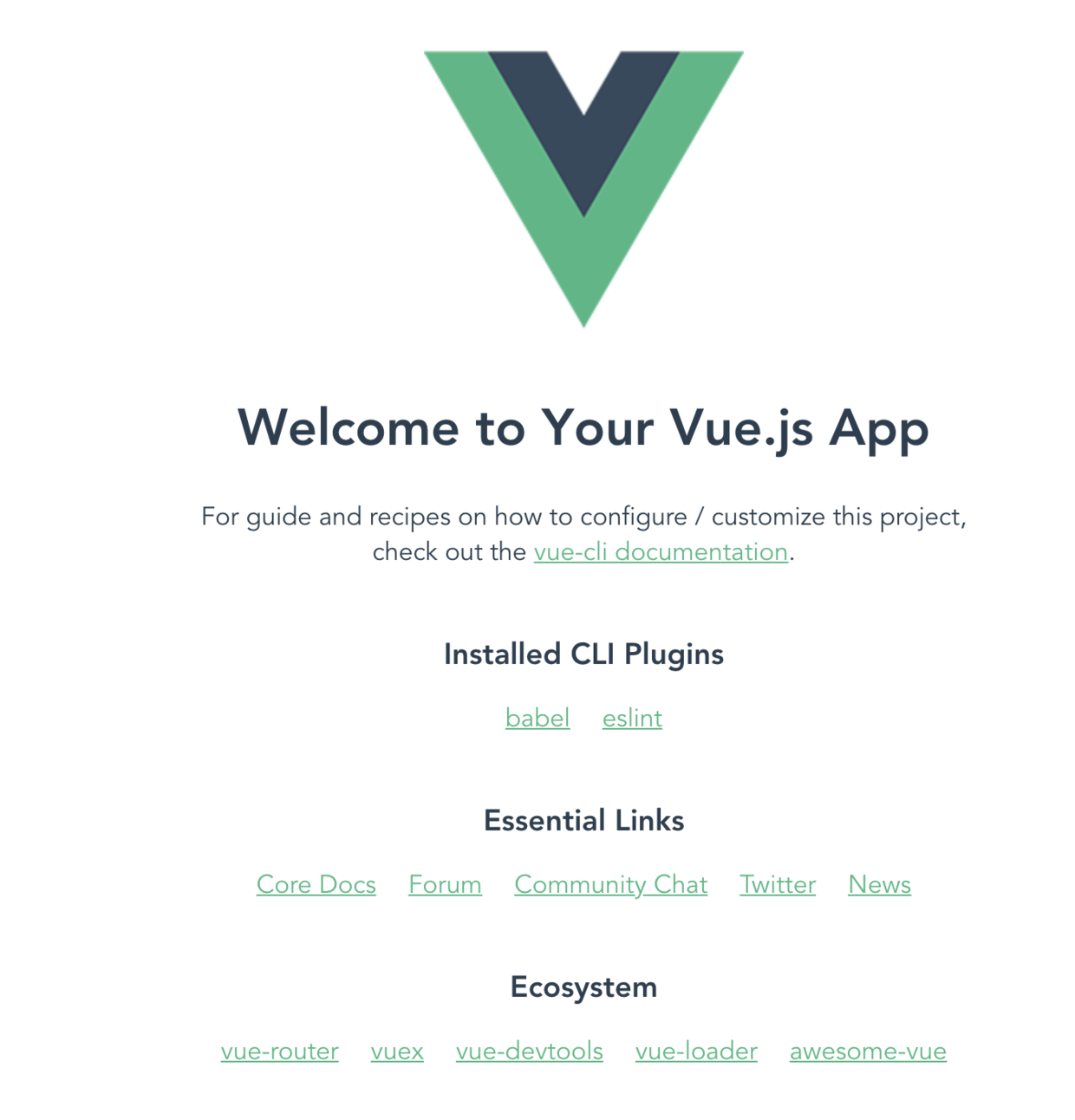
// 或者 cd vue-webpack-demo2

// 2. 安装依赖

npm i

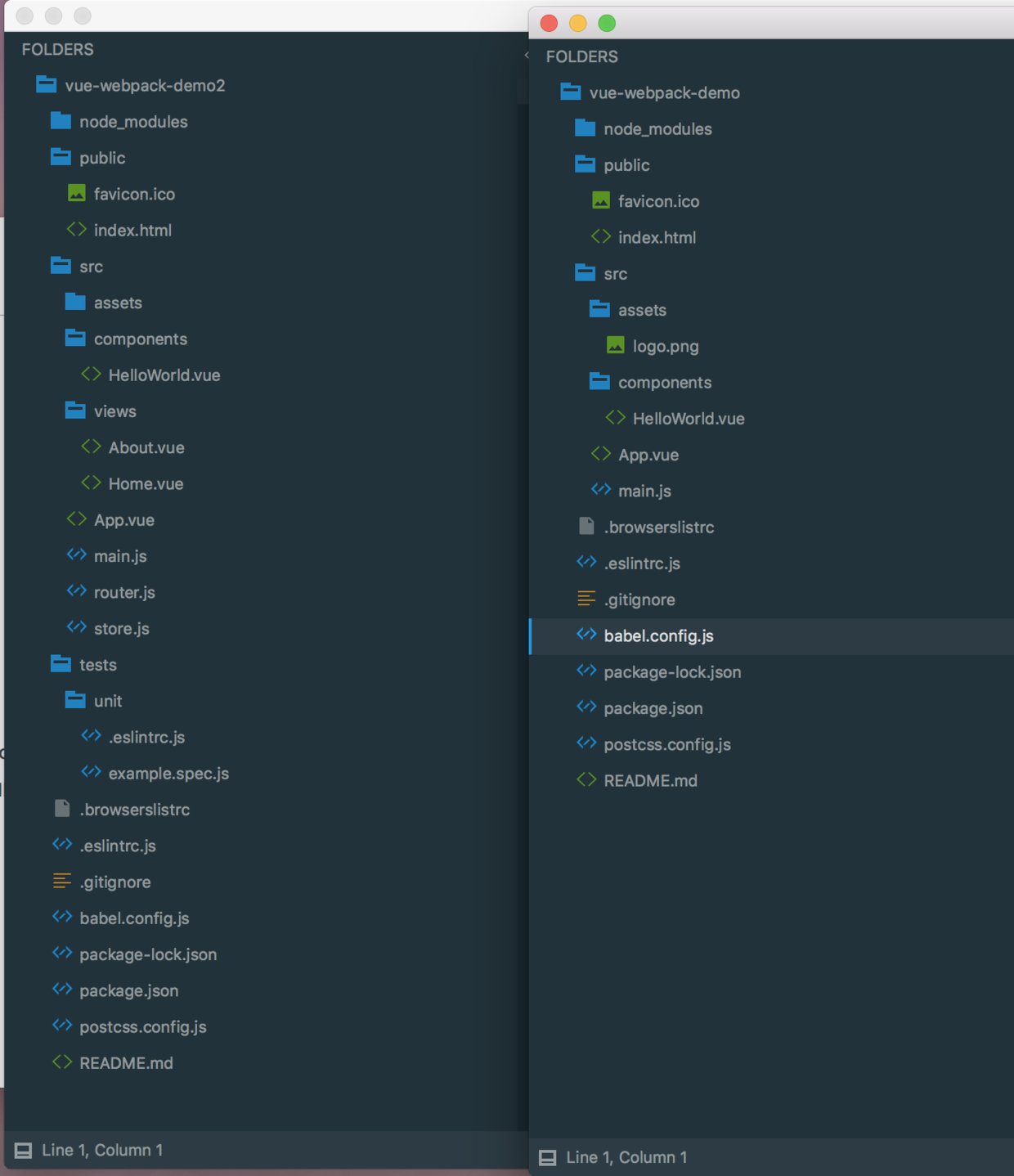
// 3. 启动

npm run serve



1.4 项目改变

1. 相比 vue-cli 2.X 创建的目录，vue-cli 3.0 创建的目录看不见 webpack 的配置



1. 启动命令行由：

npm run dev 或者 npm start

改变为：

npm run serve

1. 安装过程也发生了一些变化，配置可以保存，下次可以再用，像前面的 vue-webpack4。
2. 手动配置 webpack：在根目录下新建一个 vue.config.js 文件，进行你的配置 :

const path = require('path');

module.exports = {

// 基本路径

baseUrl: './',

// 输出文件目录

outputDir: 'dist',

// eslint-loader 是否在保存的时候检查

lintOnSave: true,

// webpack配置

// see https://github.com/vuejs/vue-cli/blob/dev/docs/webpack.md

chainWebpack: () => {},

configureWebpack: (config) => {

if (process.env.NODE\_ENV === 'production') {

// 为生产环境修改配置...

config.mode = 'production';

} else {

// 为开发环境修改配置...

config.mode = 'development';

}

Object.assign(config, {

// 开发生产共同配置

resolve: {

alias: {

'@': path.resolve(\_\_dirname, './src'),

'@c': path.resolve(\_\_dirname, './src/components')

}

}

});

},

// 生产环境是否生成 sourceMap 文件

productionSourceMap: true,

// css相关配置

css: {

// 是否使用css分离插件 ExtractTextPlugin

extract: true,

// 开启 CSS source maps?

sourceMap: false,

// css预设器配置项

loaderOptions: {},

// 启用 CSS modules for all css / pre-processor files.

modules: false

},

// use thread-loader for babel & TS in production build

// enabled by default if the machine has more than 1 cores

parallel: require('os').cpus().length > 1,

// PWA 插件相关配置

// see https://github.com/vuejs/vue-cli/tree/dev/packages/%40vue/cli-plugin-pwa

pwa: {},

// webpack-dev-server 相关配置

devServer: {

open: process.platform === 'darwin',

host: '0.0.0.0',

port: 8080,

https: false,

hotOnly: false,

// proxy: {

// // 设置代理

// // proxy all requests starting with /api to jsonplaceholder

// 'http://localhost:8080/': {

// target: 'http://baidu.com:8080', //真实请求的目标地址

// changeOrigin: true,

// pathRewrite: {

// '^http://localhost:8080/': ''

// }

// }

// },

before: (app) => {}

},

// 第三方插件配置

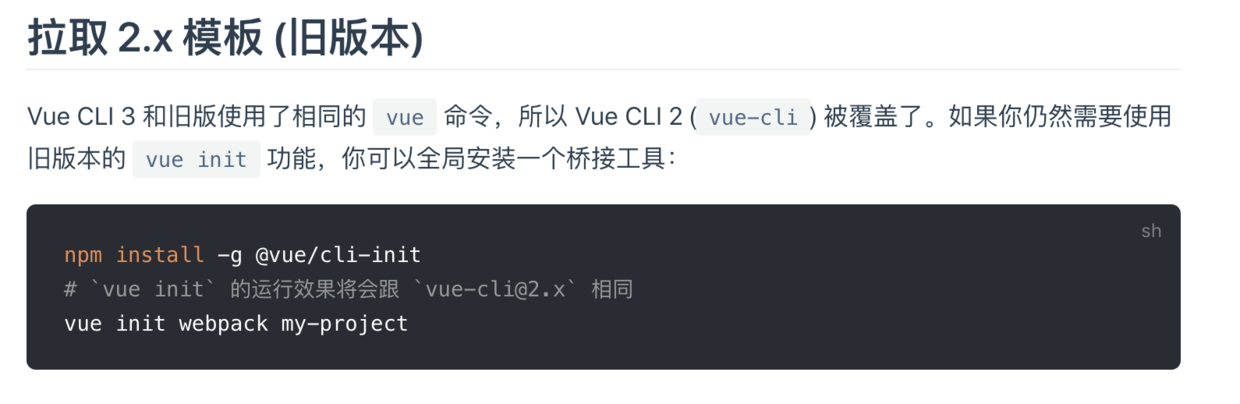
pluginOptions: {

// ...

}

};

1. 当然如果你不想用3.0的话，还是可以继续使用2.0的, 官方文档是这样说的：



具体配置看官方文档： [vue-cli 3.0](https://cli.vuejs.org/zh/guide/) [简单的配置方式](https://cli.vuejs.org/guide/webpack.html#simple-configuration)

踩坑记

1. npm 的全局路径被修改了

我都不记得在装什么包的时候修改了 mac 中 npm 的全局路径了，平时 npm 运行各种命令不报错。

全局卸载 vue-cli 命令行：

npm uninstall vue-cli -g;

但是今天全局卸载 vue-cli 的时候一直不成功，搞了一个小时，结果看了一下 npm 的全局路径，才发现路径不对！！！

如果你的 npm 的全局路径也变了，请按如下步骤修改加默认的。

方法一：

原因：npmr 的配置改变了，导致正确的 npmr 不能用。

* 打开终端，切换到根路径

cd

open .npmrc

* 文件里面修改为 prefix=/usr/local

方法二：

npm config set prefix /usr/local //是默认路径 修改了路径会出现错误。

按上面的方法修改完，再全局卸载 vue-cli 果然就成功了。