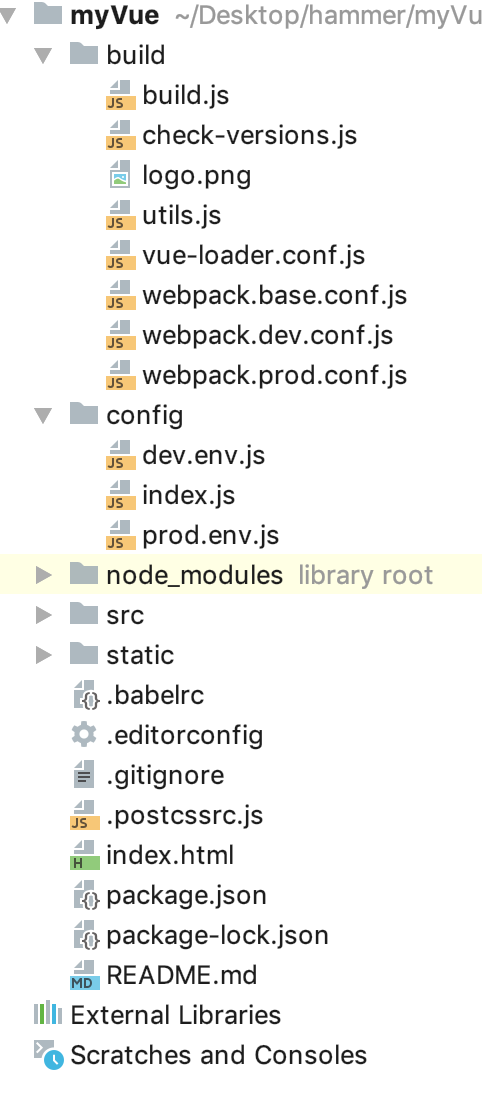
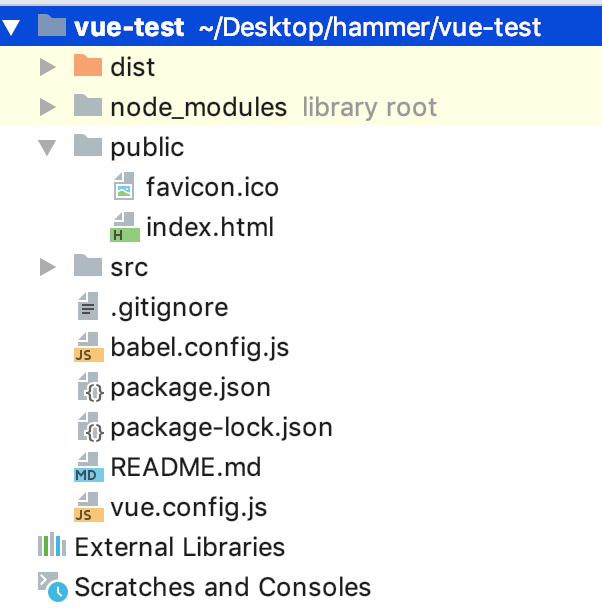
**使用Vue-cli 3.0搭建Vue项目**

参考：https://www.showdoc.cc/vuedoc?page\_id=2459992731012502

3.0目录 2.0目录



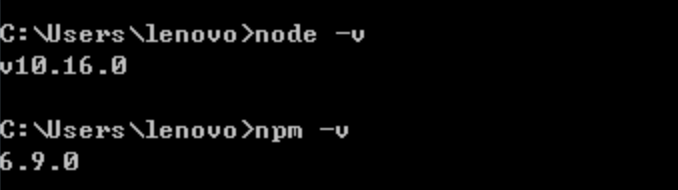
**3.0VS2.0**

vue-cli3.0默认项目目录与2.0的相比，更精简:  
1.移除的配置文件根目录下的，build和config等目录，  
2.移除了static文件夹，新增了public文件夹，并且index.html移动到public中。  
3.在src文件夹中新增了views文件夹，用于分类 视图组件 和 公共组件 。  
4.大部分配置 都集成到 vue.config.js这里,在项目根目录下

1.URL转换规则  
URL是绝对路径，如/images/foo.png,会被保留不变。  
URL以.开始，会被认为是相对模块请求，根据文档结构（folder structure）转换。  
URL以~开始，会被认为是模块请求,意味着可以在node modules里引用资源：  
  
URL以@开始，会被认为是模块请求，这很有用因为，Vue CLI对默认别名@是/src

2.public文件夹  
public文件夹下的文件并不会被Webpack处理：它们会直接被复制到最终的打包目录（文件名需指定）下。必须使用绝对路径引用这些文件，简单说就是用来存放万年不变的文件。  
在vue2.x版本类似static/ 文件夹。

**一、vue-cli搭建脚本文件**

1. 安装node.js，安装完node.js之后，npm也会自动安装  
   （1）Windows 上安装 Node.js  
   [https://nodejs.org/en/download/](https://nodejs.org/en/download/" \t "_blank)  
   （2）Mac OS 上安装  
   你可以通过以下两种方式在 Mac OS 上来安装 node：  
   a、在官方下载网站下载 pkg 安装包，直接点击安装即可。  
   b、使用 brew 命令来安装：  
   brew install node  
   查询是否安装成功的命令：  
   node -v  
   npm -v  
   
2. 使用下一命令全局安装  
   npm install -g [@vue](https://github.com/vue" \t "_blank" \o "@vue)/cli
3. 用这个命令来检查其版本  
   vue —version
4. 搭建一个项目名称为vue-test的Vue前端项目  
   vue create vue-test  
   选择Manually select features  
   选择项目需要的一些特性（此处我们选择需要Babel编译、使用Vue路由、Vue状态管理器、CSS预处理器、代码检测和格式化、以及单元测试，暂时不考虑端到端测试(E2E Testing)）  
   选择CSS预处理器语言，此处选择LESS  
   选择ESLint的代码规范，此处使用 Standard代码规范  
   选择何时进行代码检测，此处选择在保存时进行检测  
   选择单元测试解决方案，此处选择 Jest  
   选择 Babel、PostCSS、ESLint等配置文件存放位置，此处选择单独保存在各自的配置文件中  
   配置完成后等待Vue-cli完成初始化
5. npm run serve

**二、vue.config.js配置**

1. vue.config.js是一个可选的配置文件，如果项目的（和package.json同级的）根目录中存在这个文件，那么它会被[@vue](https://github.com/vue" \t "_blank" \o "@vue)/cli自动加载。你也可以使用package.json中的vue字段，但是注意这种写法需要你严格遵照JSON的格式来写。

这个文件应该导出了一个包含了选项的对象

* 1. // vue.config.js
  2. module.exports = {
  3. // 选项...
  4. }

1. 配置代理  
   如果你的前端应用和后端 API 服务器没有运行在同一个主机上，你需要在开发环境下将 API 请求代理到 API 服务器。这个问题可以通过 vue.config.js 中的 devServer.proxy 选项来配置

devServer

Type: Object  
所有webpack-dev-server的选项([https://webpack.js.org/configuration/dev-server/)都支持.注意：](https://webpack.js.org/configuration/dev-server/)%E9%83%BD%E6%94%AF%E6%8C%81.%E6%B3%A8%E6%84%8F%EF%BC%9A" \t "_blank)

有些值像host、port和https可能会被命令行参数覆写  
有些像publicPath和historyApiFallback不应该被修改，因为它们需要和开发服务器的baseUrl同步以保障正常工作  
devServer.proxy  
Type:string | object  
devServer.proxy可以是一个指向开发环境API服务器的字符串：

* 1. module.exports = {
  2. devServer: {
  3. proxy: 'http://localhost:4000'
  4. }
  5. }

这会告诉开发服务器将任何未知请求 (没有匹配到静态文件的请求) 代理到[http://localhost:4000。](http://localhost:4000%E3%80%82/" \t "_blank)

如果你想要更多的代理控制行为，也可以使用一个 path: options 成对的对象。完整的选项可以查阅 http-proxy-middleware ([https://github.com/chimurai/http-proxy-middleware#proxycontext-config)。](https://github.com/chimurai/http-proxy-middleware" \l "proxycontext-config)%E3%80%82" \t "_blank)

vue-cli2.0创建的项目的代理配置方式是修改config/index.js文件中的proxyTable:  


vue-cli3.0的代理配置，直接将proxyTable中配置copy到devServer.proxy中即可：

* 1. module.exports = {
  2. devServer: {
  3. proxy: {
  4. '/hrm/api': {
  5. //target: 'http://192.168.1.209:10751/', // Dev环境
  6. // target: 'http://192.168.1.238:10751/', // Test环境
  7. // target: 'http://192.168.1.215:10751/', // Rls环境
  8. target: 'http://192.168.1.218:10751/', // 正式环境
  9. changeOrigin: true,
  10. autoRewrite: true,
  11. cookieDomainRewrite: true,
  12. pathRewrite: {
  13. '^/hrm/api/': '/'
  14. }
  15. }
  16. }
  17. }
  18. }

**3. 配置Webpack其他选项**  
参考：webpack简单的配置方式([https://cli.vuejs.org/zh/guide/webpack.html#%E7%AE%80%E5%8D%95%E7%9A%84%E9%85%8D%E7%BD%AE%E6%96%B9%E5%BC%8F](https://cli.vuejs.org/zh/guide/webpack.html" \l "%E7%AE%80%E5%8D%95%E7%9A%84%E9%85%8D%E7%BD%AE%E6%96%B9%E5%BC%8F" \t "_blank))

调整webpack配置最简单的方式就是在vue.config.js中的configureWebpack选项提供一个对象

module.exports = {  
// 其他选项…  
configureWebpack: {  
plugins: [  
new MyAwesomeWebpackPlugin()  
//……  
]  
}  
}  
```

1. 警告

有些 webpack 选项是基于 vue.config.js 中的值设置的，所以不能直接修改。例如你应该修改 vue.config.js 中的 outputDir 选项而不是修改 output.path；你应该修改 vue.config.js 中的 baseUrl 选项而不是修改 output.publicPath。这样做是因为 vue.config.js 中的值会被用在配置里的多个地方，以确保所有的部分都能正常工作在一起。