1. build/bin里的文件

<https://segmentfault.com/a/1190000013648161>

1.node.js中的fs.writeFile方法：以异步的方式将data写入文件，文件已存在的情况下，原内容将被替换

2.postcss: 提供了一种方式用 JavaScript 代码来处理 CSS。它负责把 CSS 代码解析成抽象语法树结构（Abstract Syntax Tree，AST），再交由插件来进行处理。

3.Fs: 使用require('fs')载入fs模块，模块中所有方法都有同步和异步两种形式

fs.readFileSync:// // Sync 表示是同步方法

4. Node.js path 模块提供了一些用于处理文件路径的小工具，我们可以通过以下方式引入该模块：

var path = require("path")

// 转换为绝对路径

console.log('resolve : ' + path.resolve('main.js'));

path.resolve([from ...], to)

将 to 参数解析为绝对路径。

5.

// 遍历 components 的 key，找到相应 component 的路径，将路径保存到 componentPaths 数组

Object.keys(components).forEach(key => {

6、

里面的scripts字段是一个对象。它的每一个属性，对应一段脚本。比如，这里build命令对应的脚本是node build.js。

7. scripts对象的key是生命周期中的事件,而value是要执行的shell命令。npm 脚本的原理非常简单。每当执行npm run，就会自动新建一个 Shell，在这个 Shell 里面执行指定的脚本命令。npm 脚本的唯一要求就是可以在 Shell 执行，因此它不一定是 Node 脚本，任何可执行文件都可以写在里面。npm 脚本的退出码，也遵守 Shell 脚本规则。如果退出码不是0，npm 就认为这个脚本执行失败

8. npm会默认设置一些script的值，比如start和install，当然你可以覆盖这2个值。

1、start：

如果你在项目里根目录有一个server.js，然后你又没自己定义start script，那么npm run start，就会执行server.js

9.参考：<http://www.ruanyifeng.com/blog/2016/10/npm_scripts.html>

10.

Node.js内置的fs模块就是文件系统模块，负责读写文件。和所有其它JavaScript模块不同的是，fs模块同时提供了异步和同步的方法。

回顾一下什么是异步方法。因为JavaScript的单线程模型，执行IO操作时，JavaScript代码无需等待，而是传入回调函数后，继续执行后续JavaScript代码。比如jQuery提供的getJSON()操作

<https://www.liaoxuefeng.com/wiki/001434446689867b27157e896e74d51a89c25cc8b43bdb3000/001434501497361a4e77c055f5c4a8da2d5a1868df36ad1000>

11. 渲染内容用res.render()，将会根据views中的模板文件进行渲染。如果不想使用views文件夹，想自己设置文件夹名字，那么app.set("views","aaaa");

12. fs.writeFile(filename, data, [options], [callback(err)])以异步的方式将data写入文件，文件已存在的情况下，原内容将被替换。

13. node.js中的fs.statSync方法

同步版的 [stat()](https://www.jb51.net/article/58553.htm)。获取文件信息。fs.stat(path, [callback(err, stats)])

方法返回一个stat数组对象，包含以下信息：(以下信息为案例中读取的文件信息，非默认值)

**14.** 最近有个需求是扫描用例，用例是放在svn上，如果每次扫描都去遍历目录的话会有占用太多的io，所以想着用文件监控，有文件变化时只对该文件进行操作。

Nodejs里的 chokidar 模块可以更好的对文件进行监控，不会产生多次的事件。

**15、gulp构建方式**

例如：你可以用gulp可以网页自动刷新，和MVVM开发模式很相似，如果你对vue.js有所了解的话，那么你一定不会陌生。你也可以使用gulp对sass进行预处理、代码检测、图片优化压缩、只需要一个简单的指令就能全部完成。 由于gulp是基于node.js开发的，所以你在使用gulp之前，必须要先安装node。然后你可以打开node服务器并执行 npm install gulp -g 全局安装gulp.

然后执行npm init 创建package.json文件. 这个文件保存着你目录下的项目模块的相关信息。可以把它想象成你所用到模块儿的一个目录.

然后执行npm intall gulp --save-dev 本地安装gulp

在工程目录下手动创建gulpfile.js文件。

//最后把你建立的html文件压缩到自动创建的dist文件里;

.pipe(gulp.dest('dist'))

参考：

[**https://www.cnblogs.com/2050/p/4198792.html**](https://www.cnblogs.com/2050/p/4198792.html)

[**https://blog.csdn.net/heshuaicsdn/article/details/78926401**](https://blog.csdn.net/heshuaicsdn/article/details/78926401)

16. cp-cli是npm 的一个命令，用于复制文件内容到另一个文件里

17. cross-env NODE\_ENV=production

npm run xxx --production // app.js console.log(process.env.NODE\_ENV); production

通过NODE\_ENV可以来设置环境变量（默认值为development），NODE\_ENV是node.js的环境变量，属于node.js模块查找的一部分，如果不设置，安装全局模块后，node.js会报找不到错误

比如你们的代码有开发（development）环境，测试（test）环境和线上（production—）环境，你写代码的时候本地可能先在development上开发，然后要上到test环境，最后上线。如何控制不同环境之间的切换呢？这时候就要用到这个env了，根据终端的export NODE\_ENV=test/NODE\_ENV=development/NODE\_ENV=production来拿到不同的环境变量参数，根据这个参数做不同的配置，然后打包就可以得到你希望的生产环境。

讲得很好：

<https://blog.csdn.net/icewfz/article/details/76640319>

有空深究一下，为啥环境变量会只传到一个文件？

看看element-ui的代码里，果然是这么写的：

18. sh命令是shell命令语言解释器，执行命令从标准输入读取或从一个文件中读取。

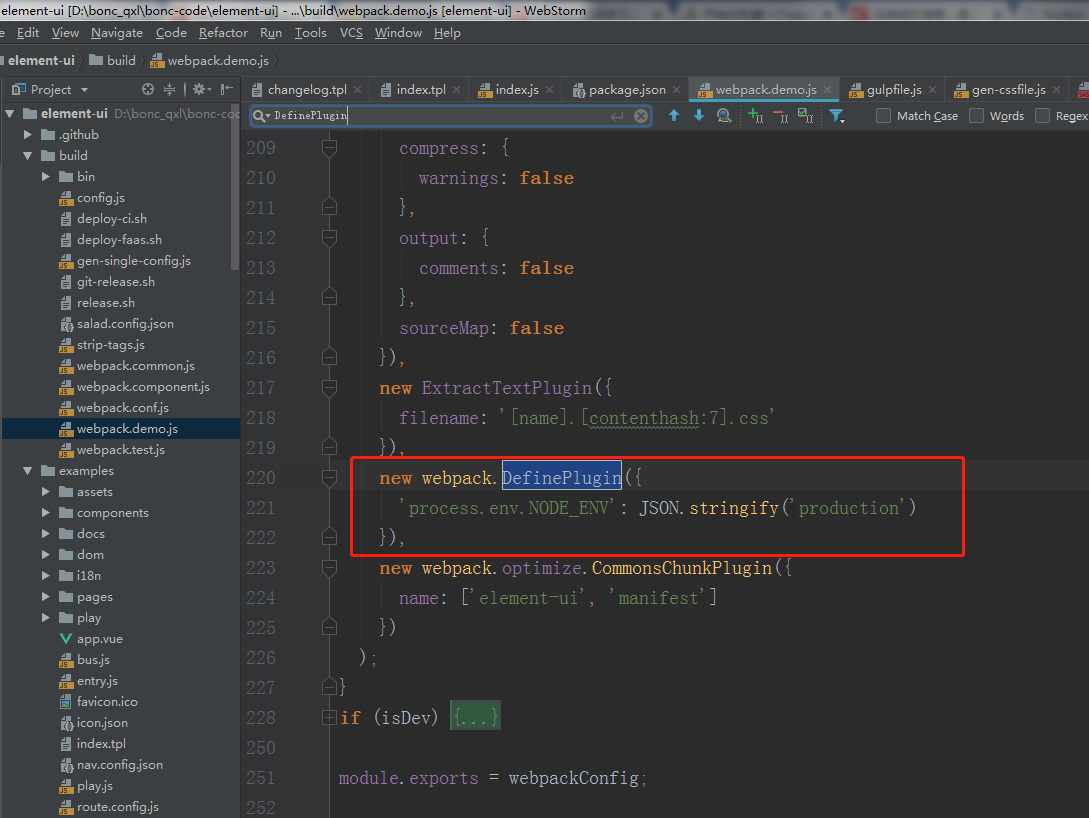
19. 上一篇文章介绍了webpack对commonjs模块的支持（如果你还没读过，建议你先阅读），这篇文章来探究一下，webpack是如何支持es模块的。

<https://segmentfault.com/a/1190000010955254>

最后有个写法需要注意： module.exports.default = module.exports; ，这里是为了兼容ESmodule，因为es6的模块 export default xxx ，在webpack中最后会变成类似于 exports.default = xxx 的形式，而 import ElementUI from 'element-ui'; 会变成 ElementUI = require('element-ui').default 的形式，为了让ESmodule识别这种commonjs的写法，就需要加上default。

20. path.join方法

21.

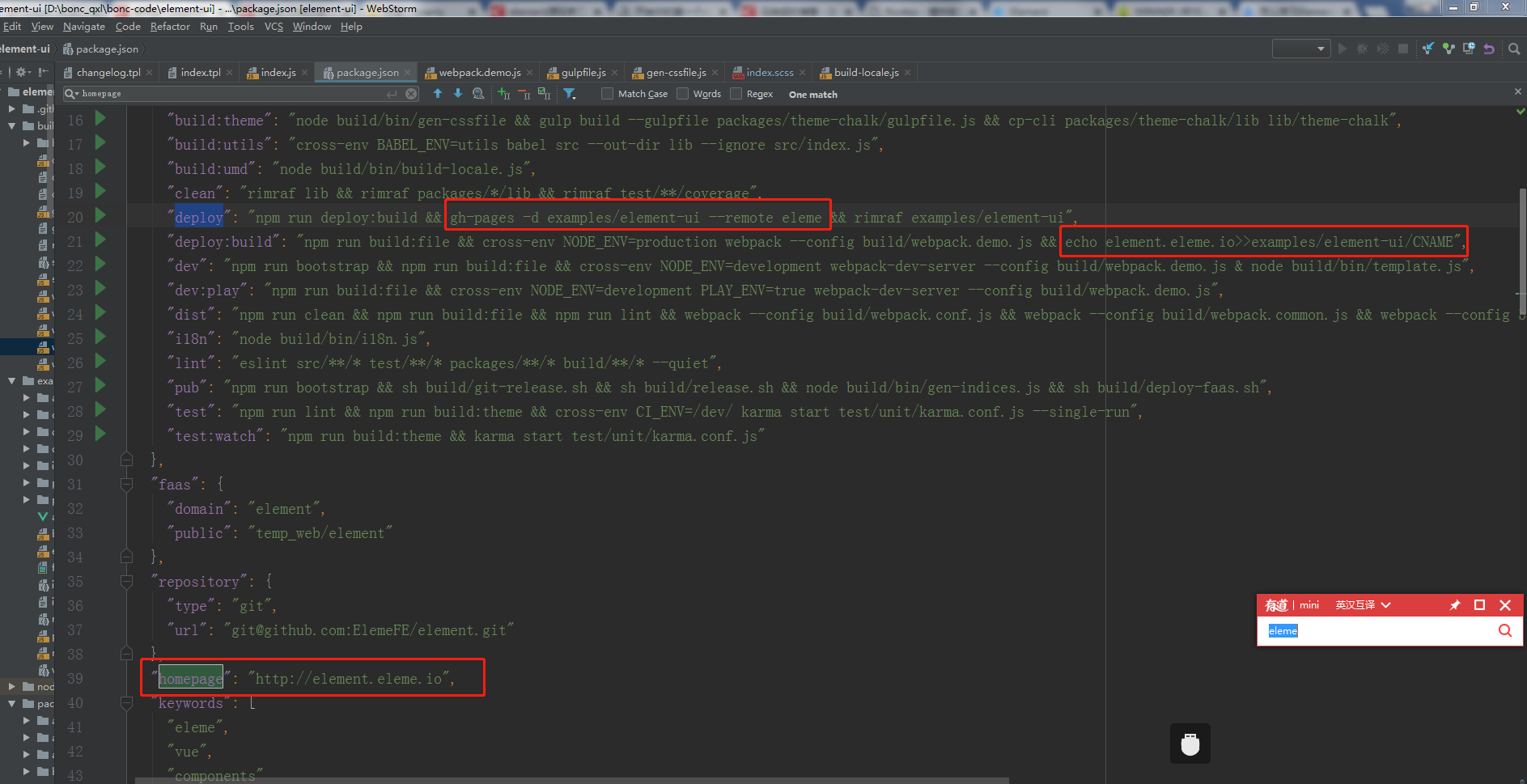


18.使用gh-pages将项目部署到github上

<https://blog.csdn.net/u011455192/article/details/81004160>

gh-pages的分支，里面的内容恰恰就是build后的内容

element的代码里也是这样写的：



19. webpack-dev-server

https://blog.csdn.net/u012045958/article/details/80544040