Element-ui源码

参考：

<https://www.codercto.com/a/26569.html>

<https://blog.csdn.net/m0_37972557>

1. 我们一般全量引入或者是部分引入组件都引用是打包以后的 lib 里边的代码

一二楼说的很正确。如果想要改源码，你的重新下载element源码。然后改掉你想要改的，然后npm run build，将改掉里面的lib替换到node\_modules中element下的lib包，只有这样才可以

1. Npm run bootstrap

Npm run build:file

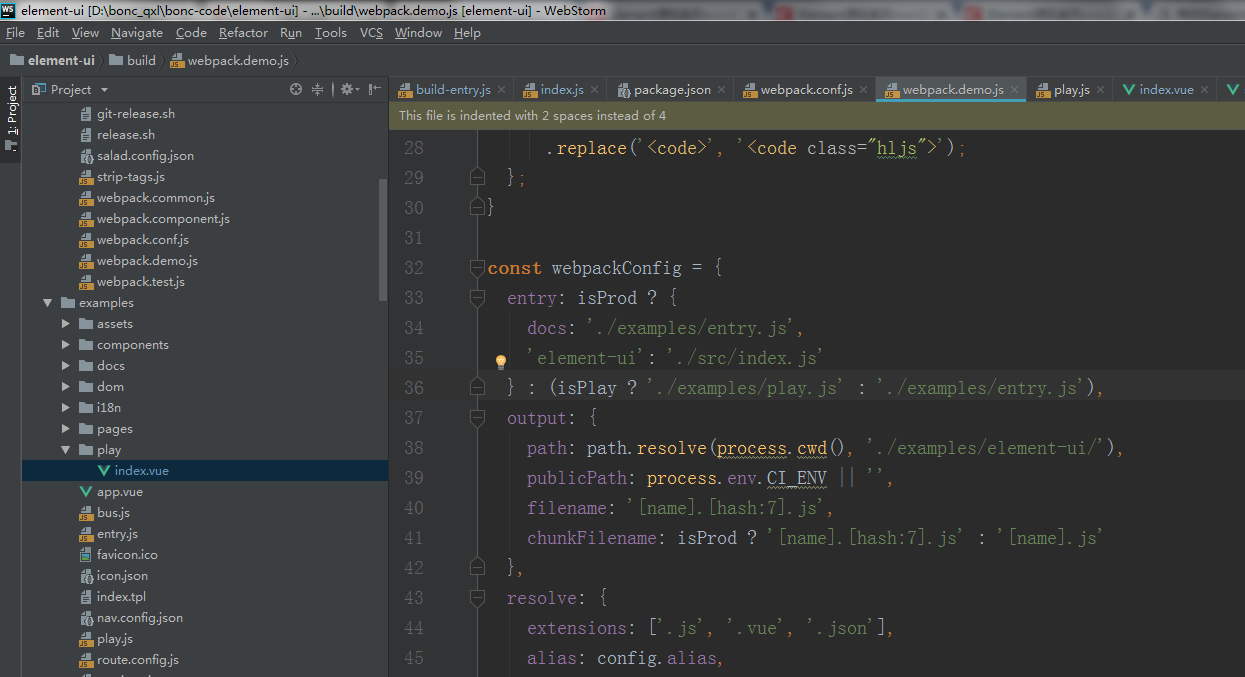
Npm run dev

效果如下：

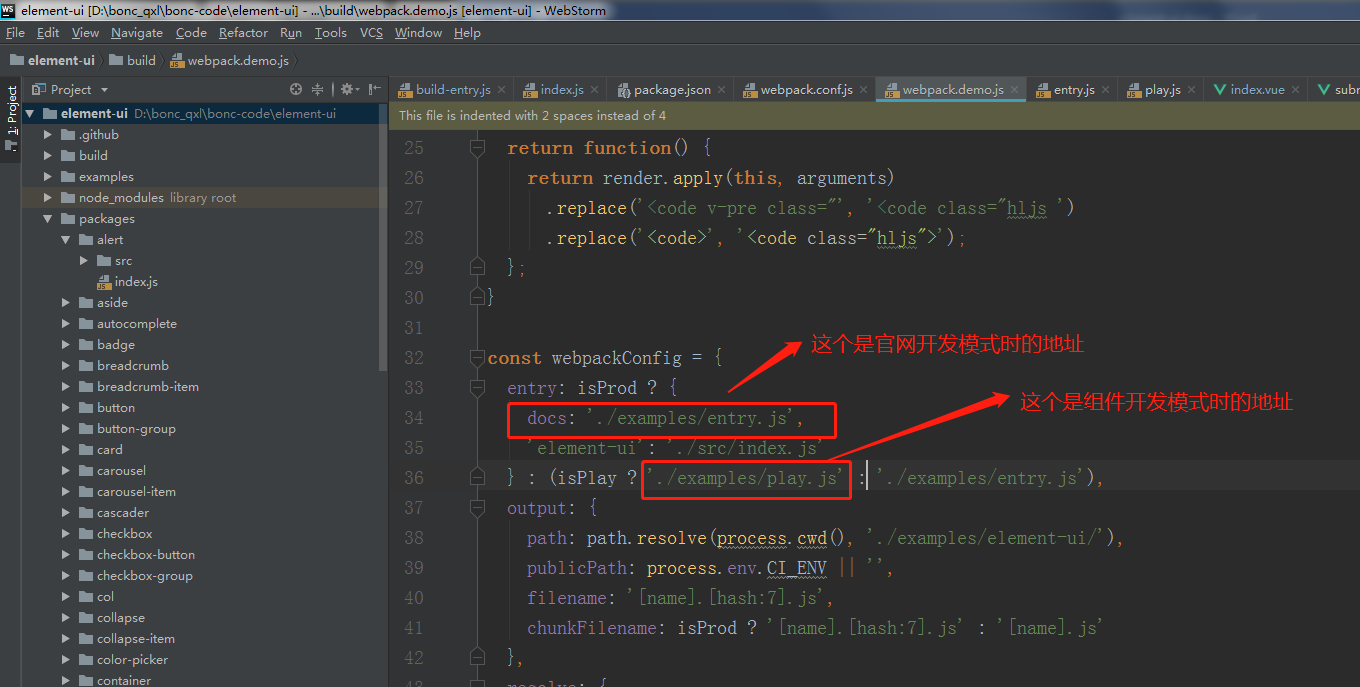


5.若是开发模式，直接在浏览器输入0.0.0:8085是不行的，要用localhost:8085才行的

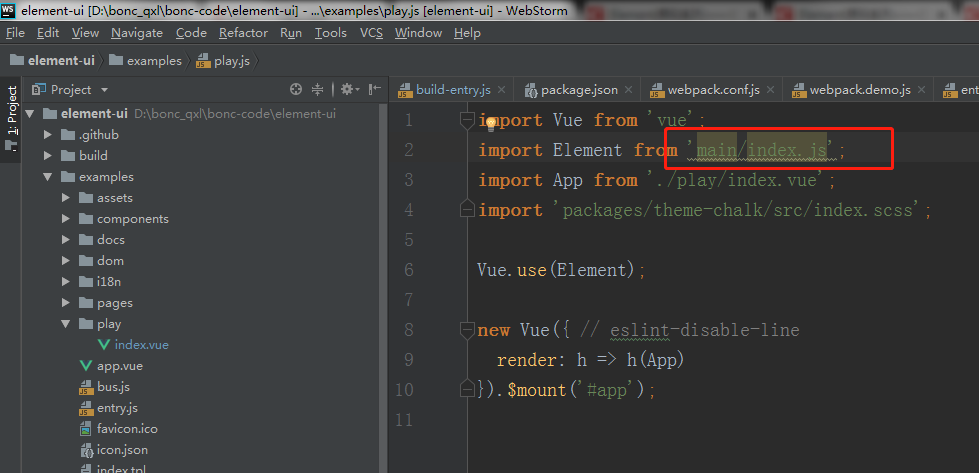
6、改动前：



(isPlay ? './examples/play.js' : './examples/entry.js'),



这个文件在哪？没找到



7、Npm脚本

npm 脚本的原理：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。因此，只要是 Shell（一般是 Bash）可以运行的命令，就可以写在 npm 脚本里面。比较特别的是，npm run新建的这个 Shell，会将当前目录的node\_modules/.bin子目录加入PATH变量，执行结束后，再将PATH变量恢复原样。

这意味着，当前目录的node\_modules/.bin子目录里面的所有脚本，都可以直接用脚本名调用，而不必加上路径。比如，当前项目的依赖里面有 Mocha，只要直接写mocha test就可以了。

8、Npm 脚本的生命周期事件

二、几个重要问题

1. .tpl文件是啥文件

以前用过的前端页面都是.html格式的，

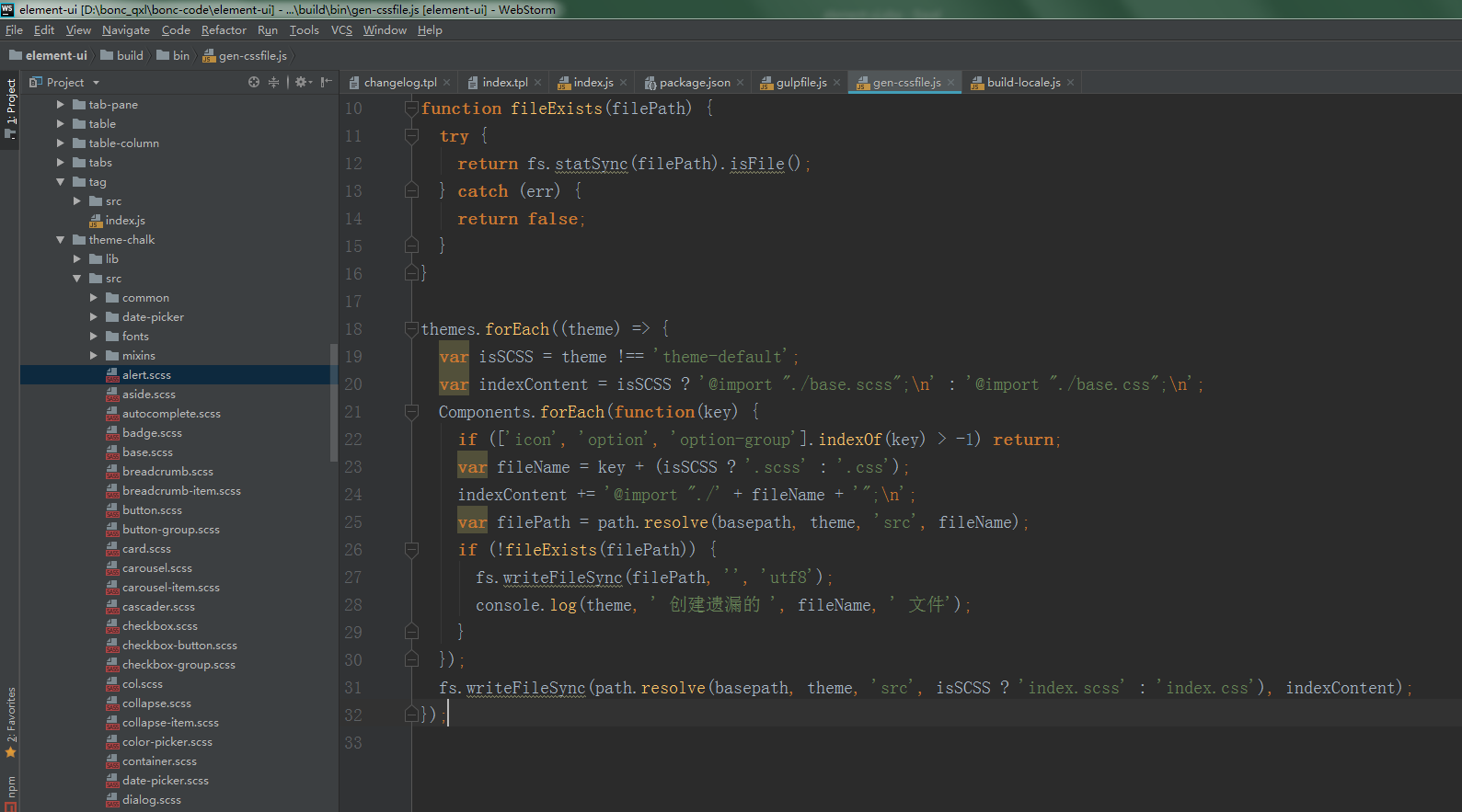
现在出现了.tpl文件，是template的缩写，发现他就是前端页面，写的也是html。

所谓的tpl就是html,

只不过里面作为模板文件(tpl)的html中添加了smarty的变量或逻辑判断.这些变量的在程序运行时由smarty替换成真正的数据

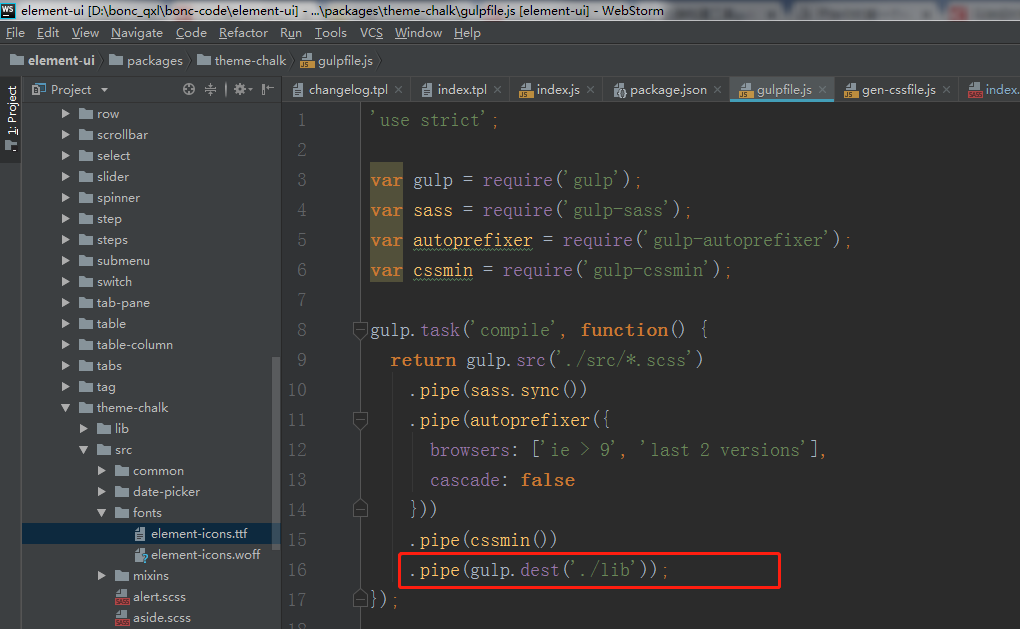
？？里面可以有变量被替换么？怎么替换的？

1. 命令行
2. "build:theme" 生成的css文件在这：



1. gulp:

最后把你建立的html文件压缩到自动创建的lib文件里：



1. 为啥有时用gulp,有时用webpack

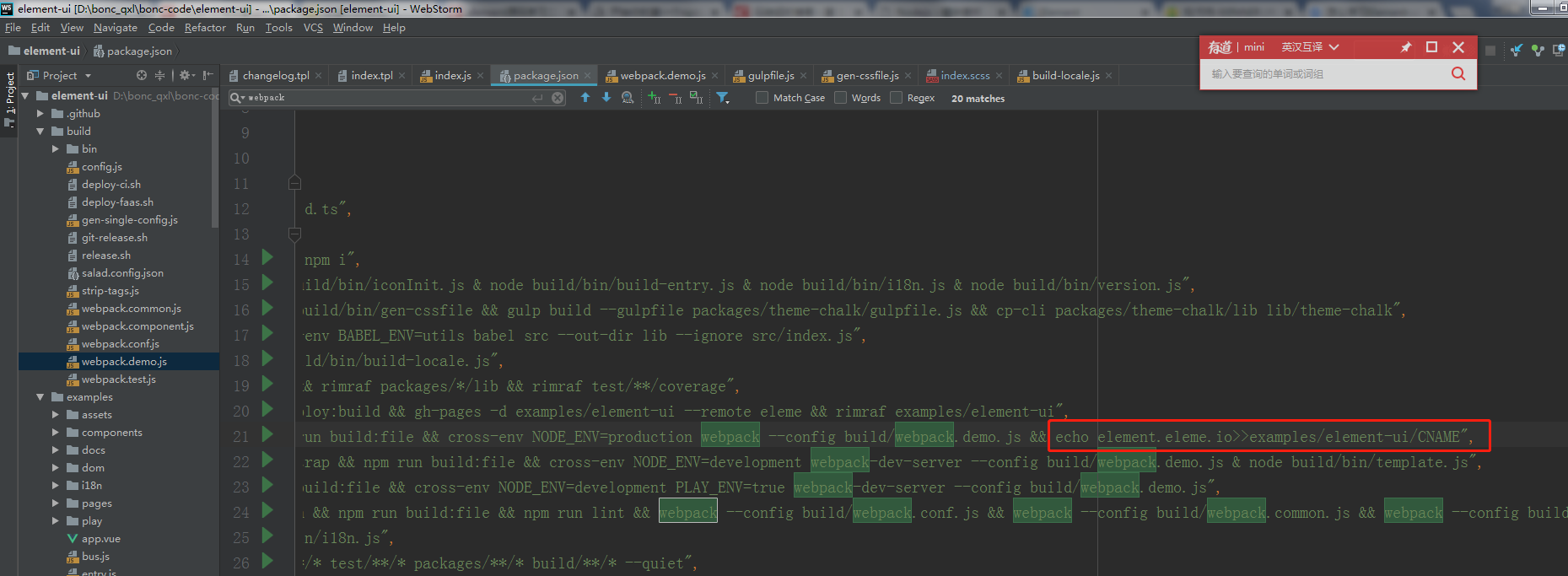
目前看，原因是gulp是用来处理了scss文件和font文件，而webpack用于处理了'.js', '.vue', '.json'文件，至于为啥，还需再看

1. pub和deploy的区别
2. md文件是咋被解析到网页中的？

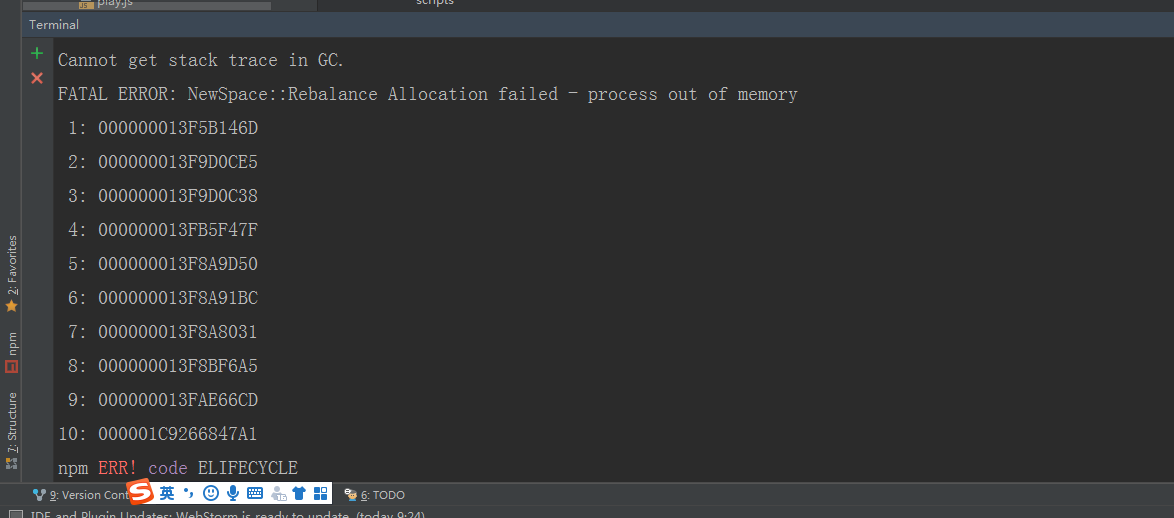
6.utils

7.mixin

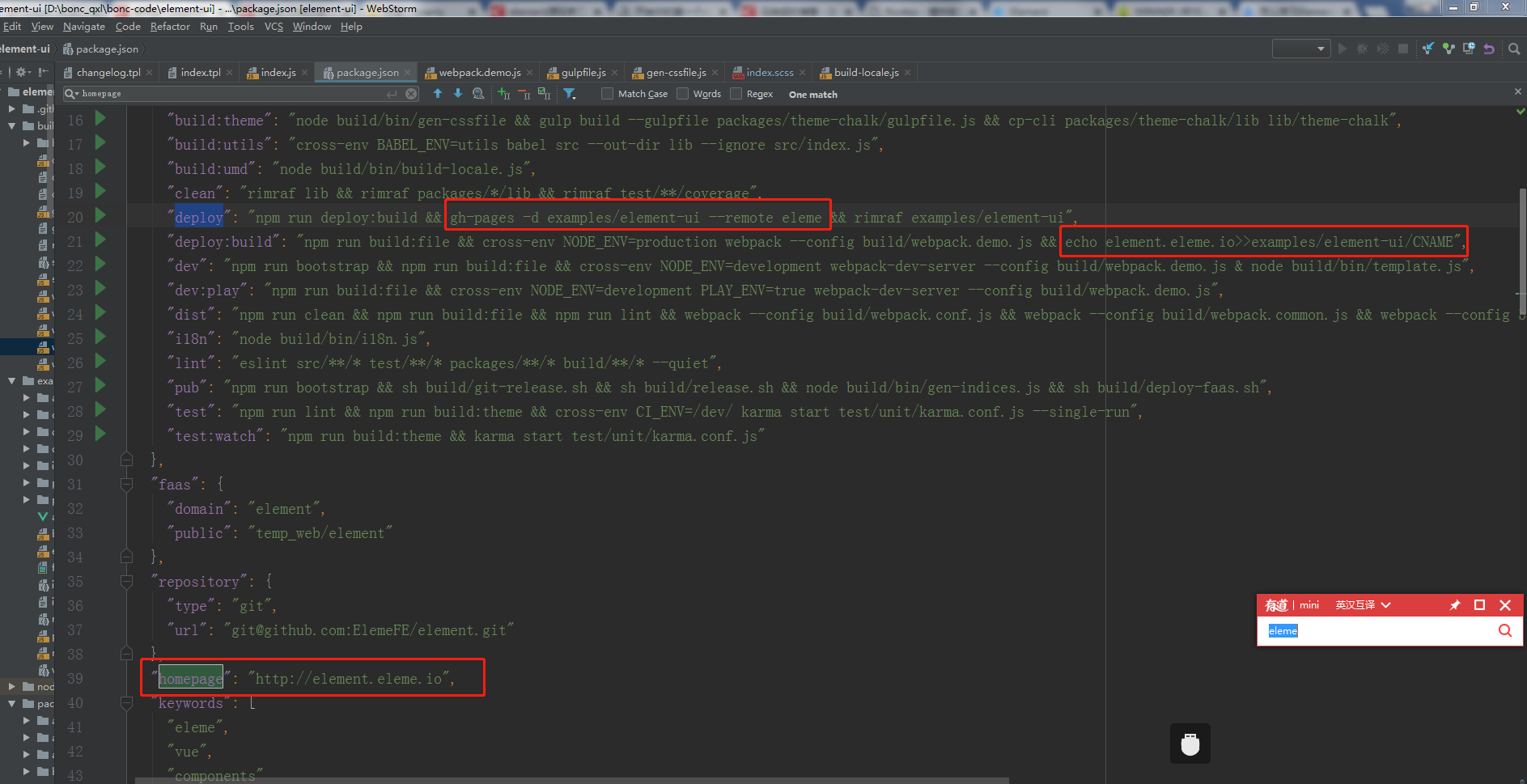
8.干嘛的？打个包试试



运行了一下，这个命令deploy:build报错了，可能是内存不够了吧：



9. “deploy”命令行：使用gh-pages将项目部署到github上

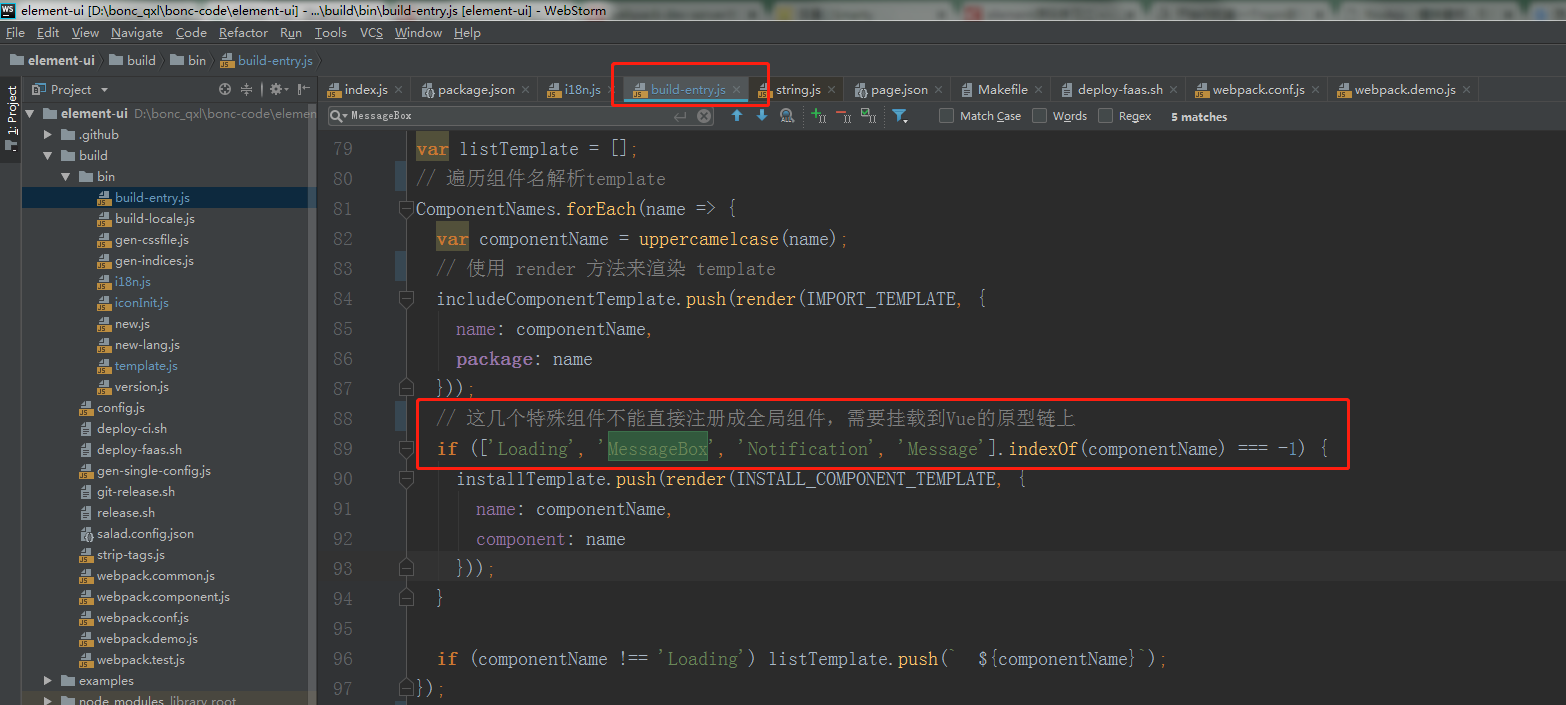


11.pub命令行的作用

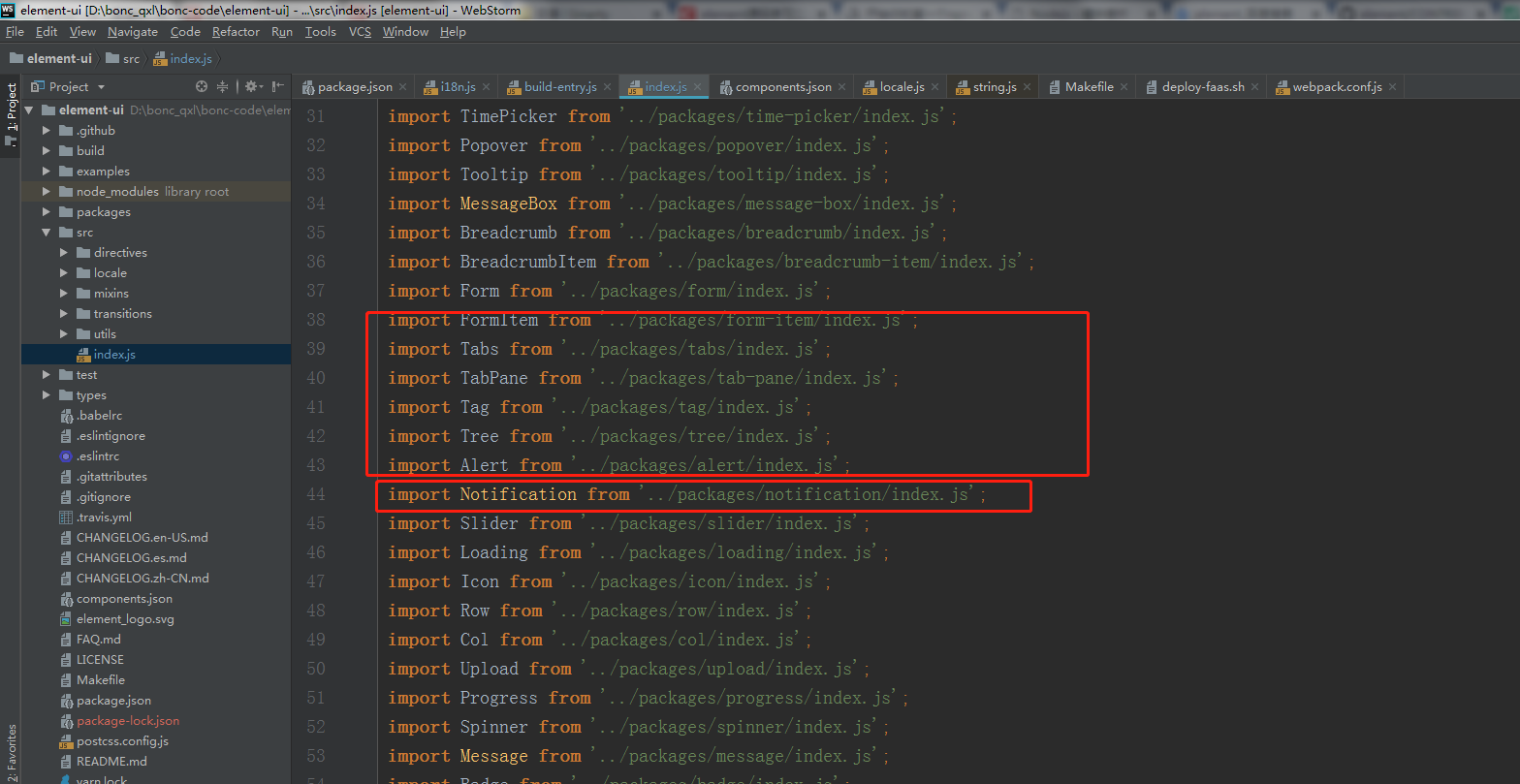
向dev分支和master分支上推送打包后的lib文件和theme，再发布到npm上

12.fs读取文件后是什么形式的？

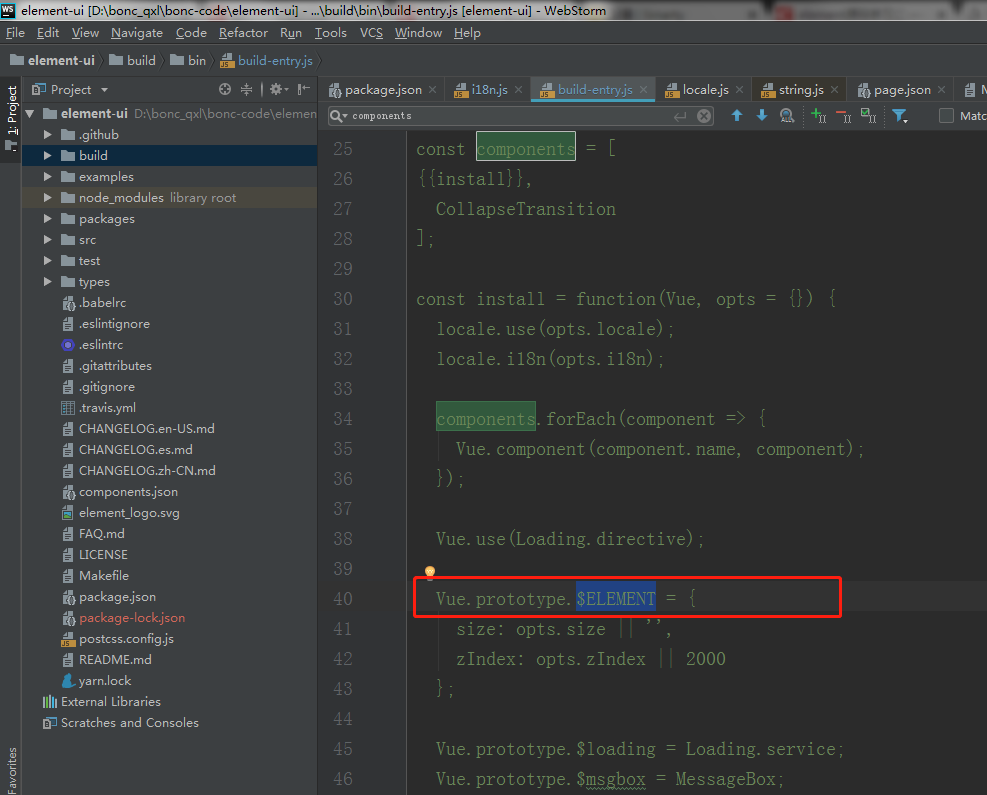
13.为啥呢？注意这个，以后可能会用到



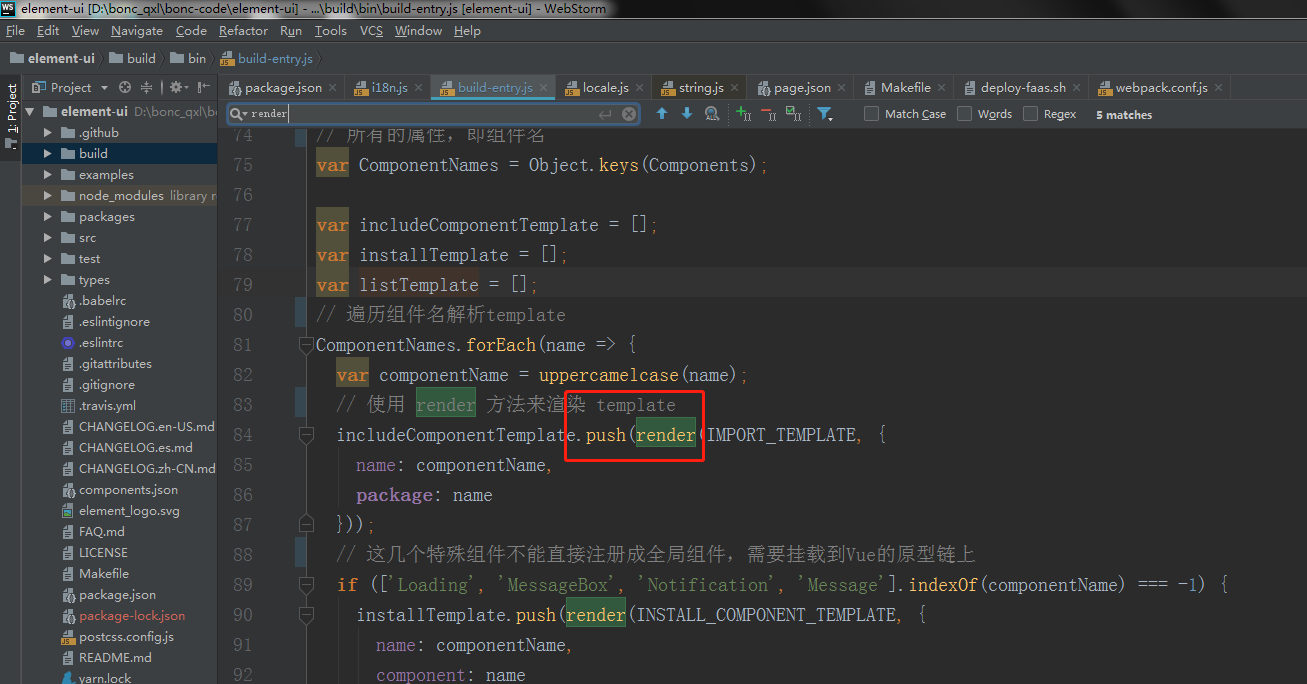
还有就是这个webstorm里能看到注册方式不同时，颜色都不一样呢：



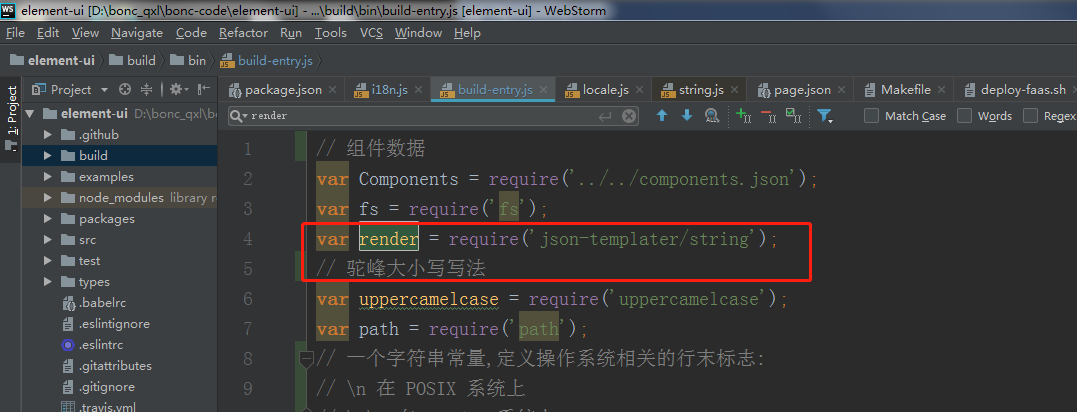
2.是啥？



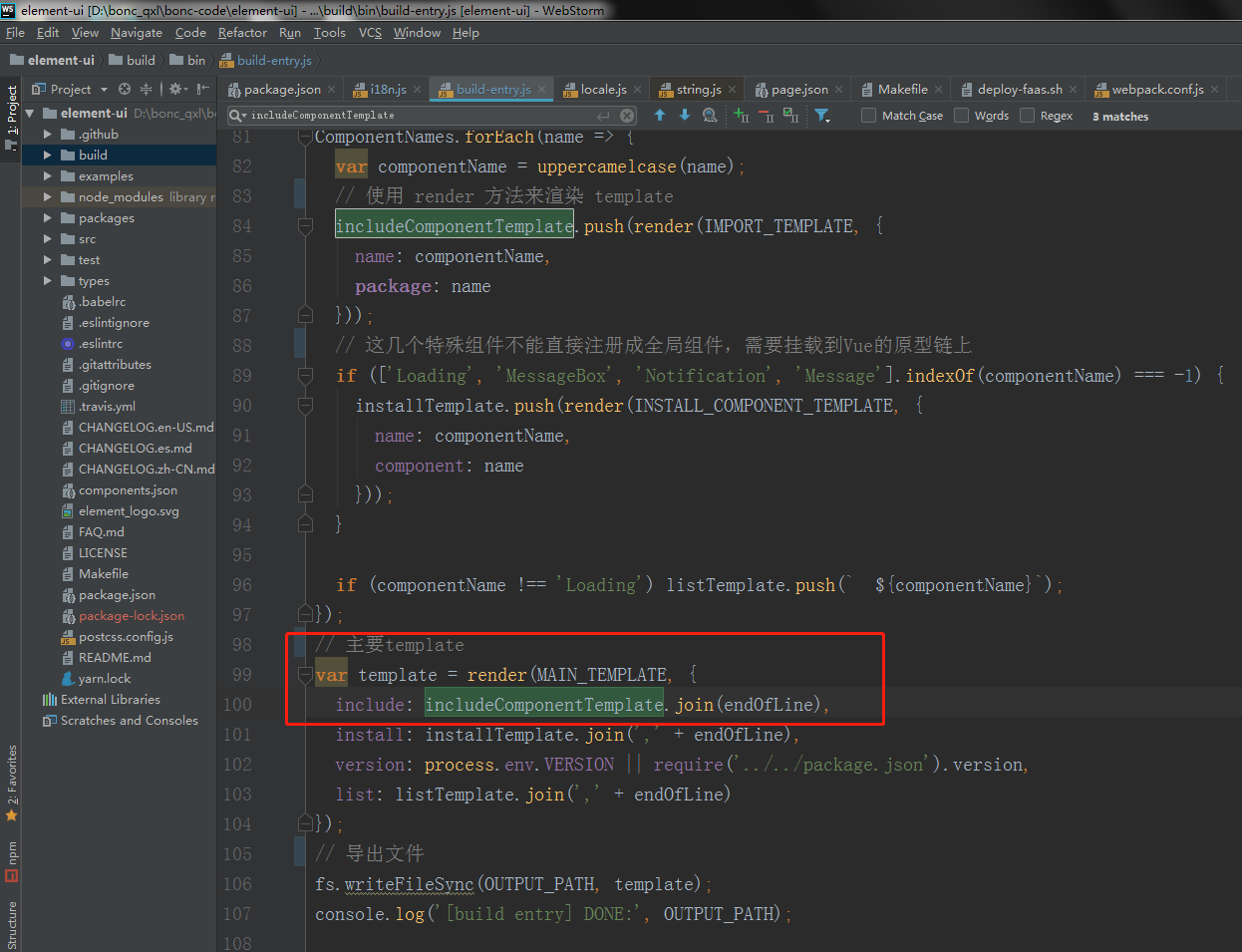
这个函数好用诶，用来替换变量，形成一个写好的模板：



定义的地方：



而且还可以把替换好的模板再替换进其他的模板里：



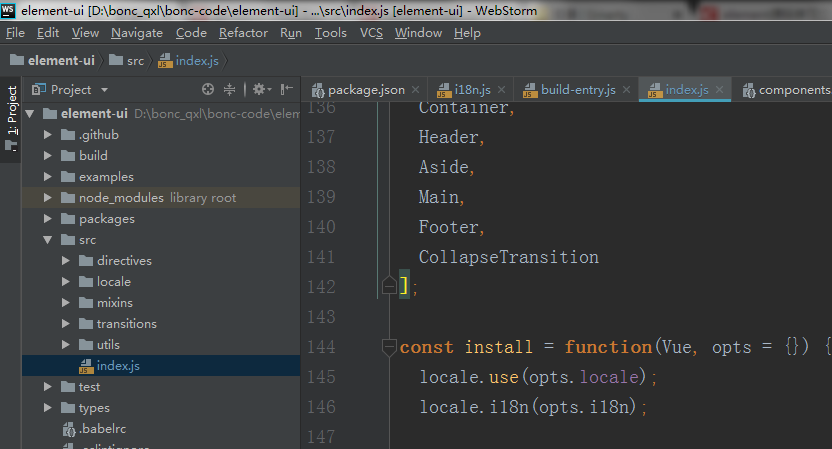
3.在build-entry.js里：

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。

exports对外暴露的install方法就是把Element组件注册会全局组件的方法。当我们使用 Vue.use 时，就会调用对外暴露的install方法。如果我们直接通过script的方式引入vue和Element，检测到Vue为全局变量时，也会调用install方法。

在module.exports对象中，除了暴露install方法外，还把所有组件进行了对外的暴露，方便引入单个组件。仔细看这个：<https://www.codercto.com/a/26569.html>

所以这个文件就是自动生成的哦：



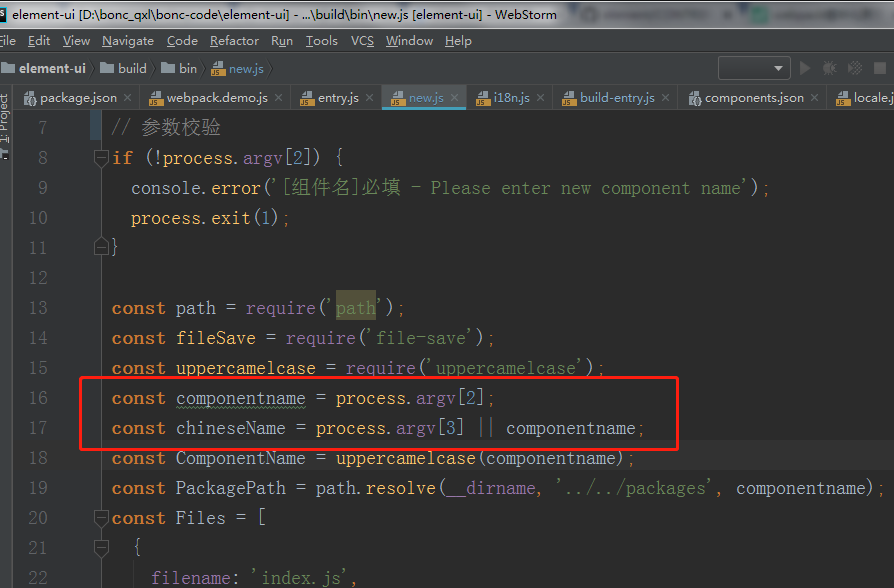
4.新建组件的命令

// 获取命令行的参数

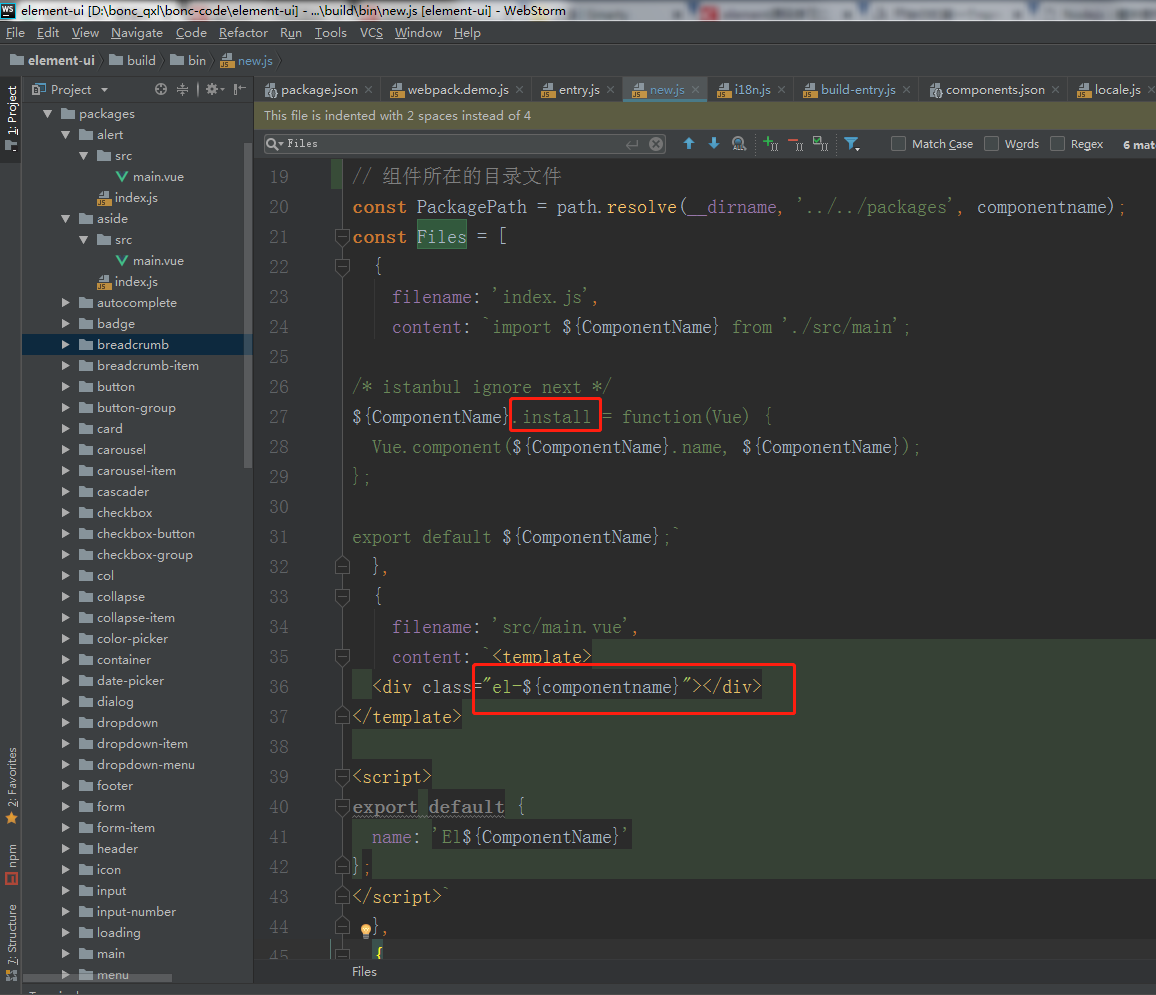
// e.g. node new.js input 输入框

// process.argv表示命令行的参数数组

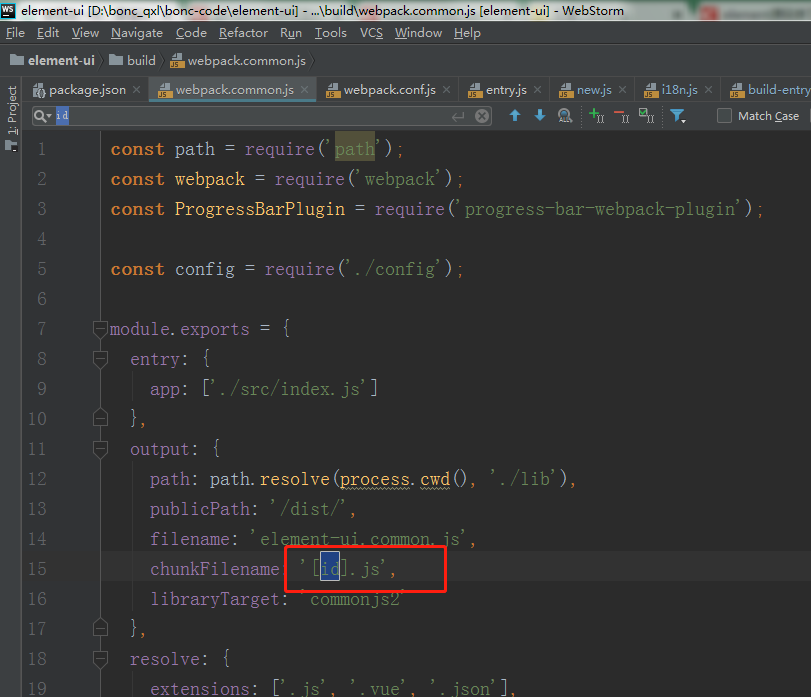
// 0是node，1是new.js，2和3就是后面两个参数



5. 每个组件都会对外单独暴露一个install方法，因为Element支持按需加载。同时，每个组件名都会加上 El 前缀。，所以我们使用Element组件时，经常是这样的 el-xxx ，这符合W3C的自定义 HTML 标签的规范（小写，并且包含一个短杠）。



2.这个是啥？

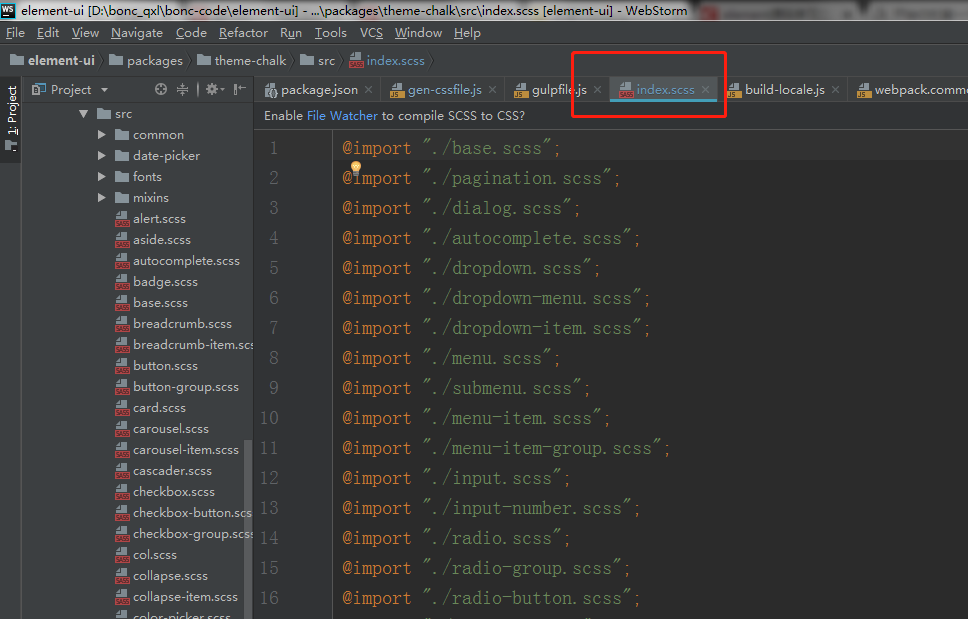


3. 当我们直接在代码中引入整个Element的时候，加载的是 webpack.common.js 打包生成的 element-ui.common.js 文件。因为我们引入npm包的时候，会根据package.json中的main字段来查找入口文件。

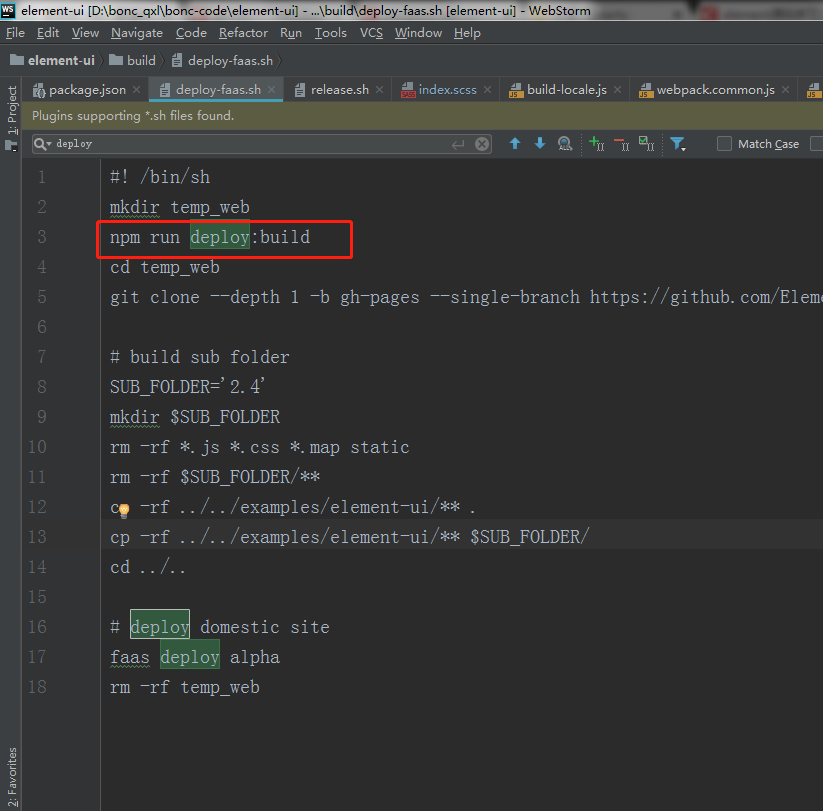
4. 在src目录下，除了index.js外，还有一些其他文件夹，这些是Element组件中经常使用的工具方法。如果你对Element的源码足够熟悉，可以直接把Element中一些工具方法拿来使用，不再需要安装其他的包。

5. build:theme

这个文件是在运行build:theme这个命令行的时候自动生成的，类似于build:file生成组件的入口文件。然后gulpfile.js将该生成的.scss文件转换为.css文件，并复制到lib里



1. 这个deploy:build命令行是在pub命令时deploy-faas.sh文件里被执行了的：



6、md文件的使用

<https://www.cnblogs.com/liugang-vip/p/6337580.html>

<https://blog.csdn.net/babylove_BaLe/article/details/76824372>

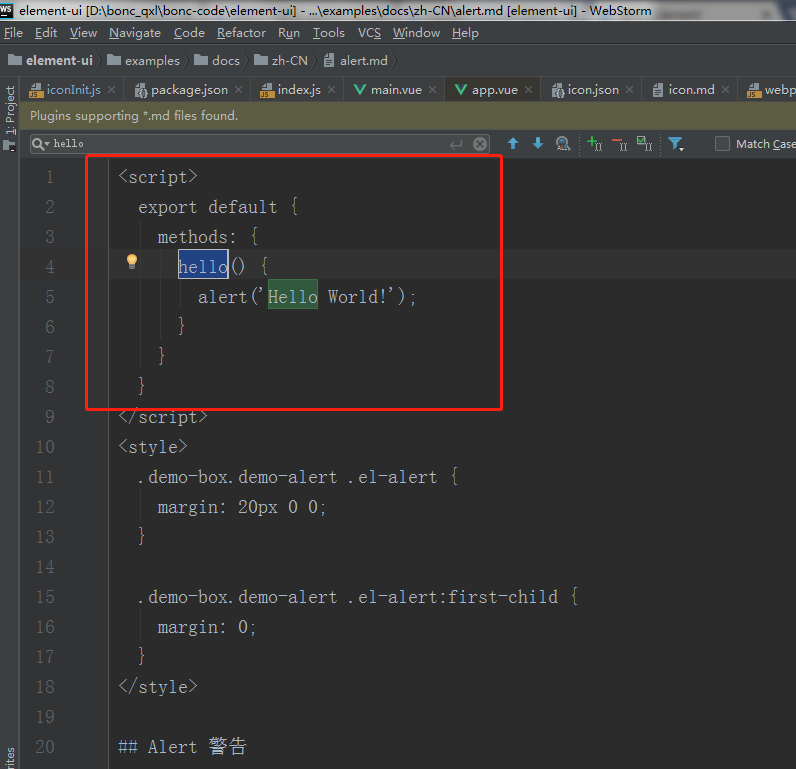
（1）``` ``` : 将要显示的代码块放在３对反引号中间

(2) :::demo 和 ::: 之间的是可以隐藏展开的

(3)问题：不知道怎么显示出代码效果的？

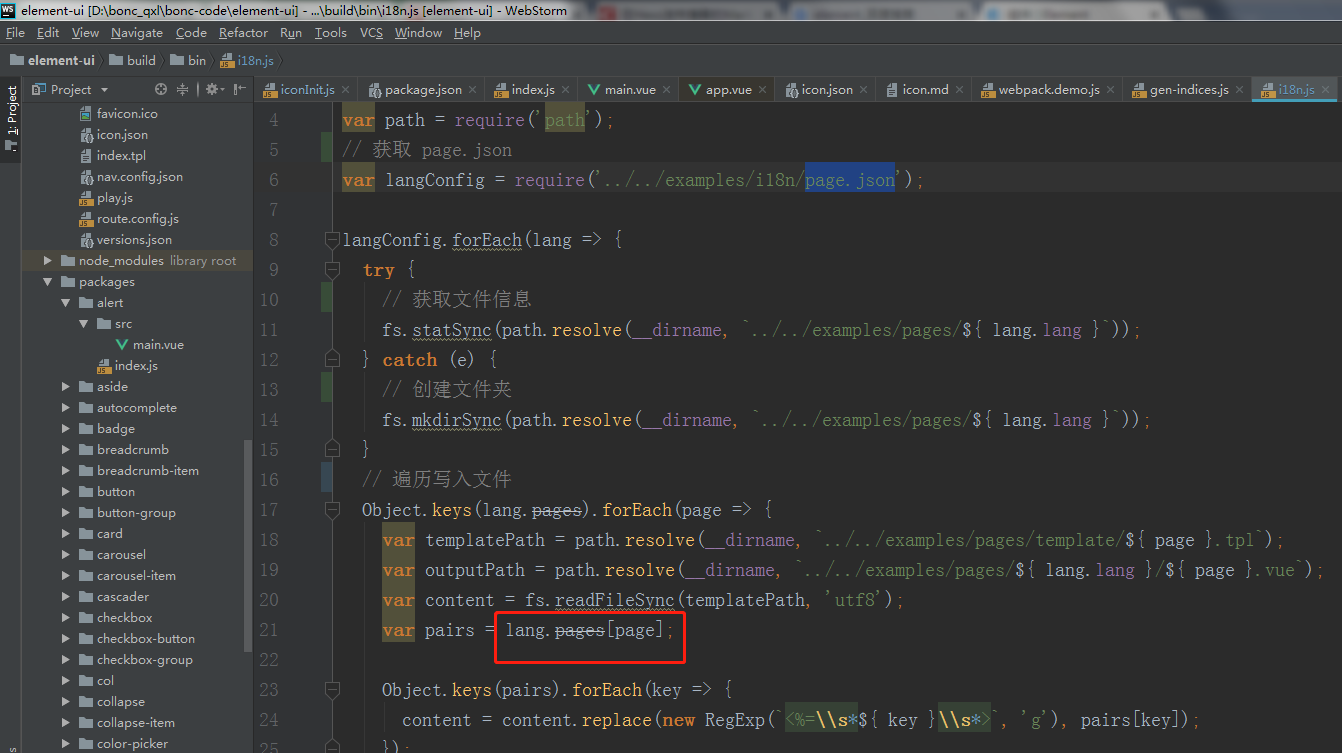
（4）为啥md文件开头和中间html都有这段代码？

（5）examples/docs/\*\*/\*.md这个目录下的文件在哪被引用了？

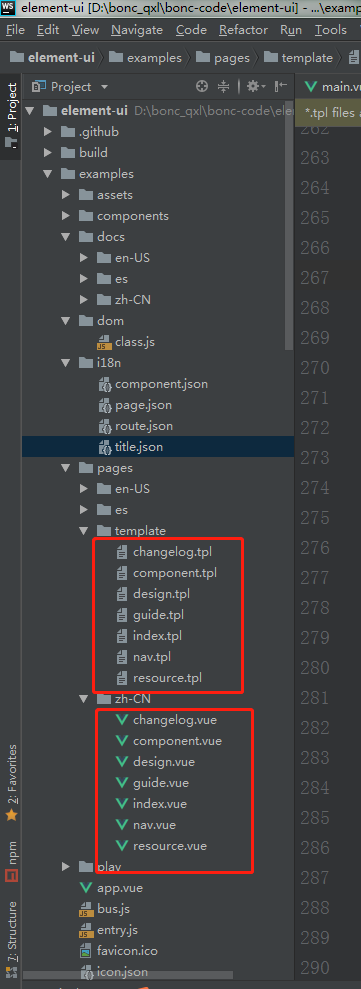


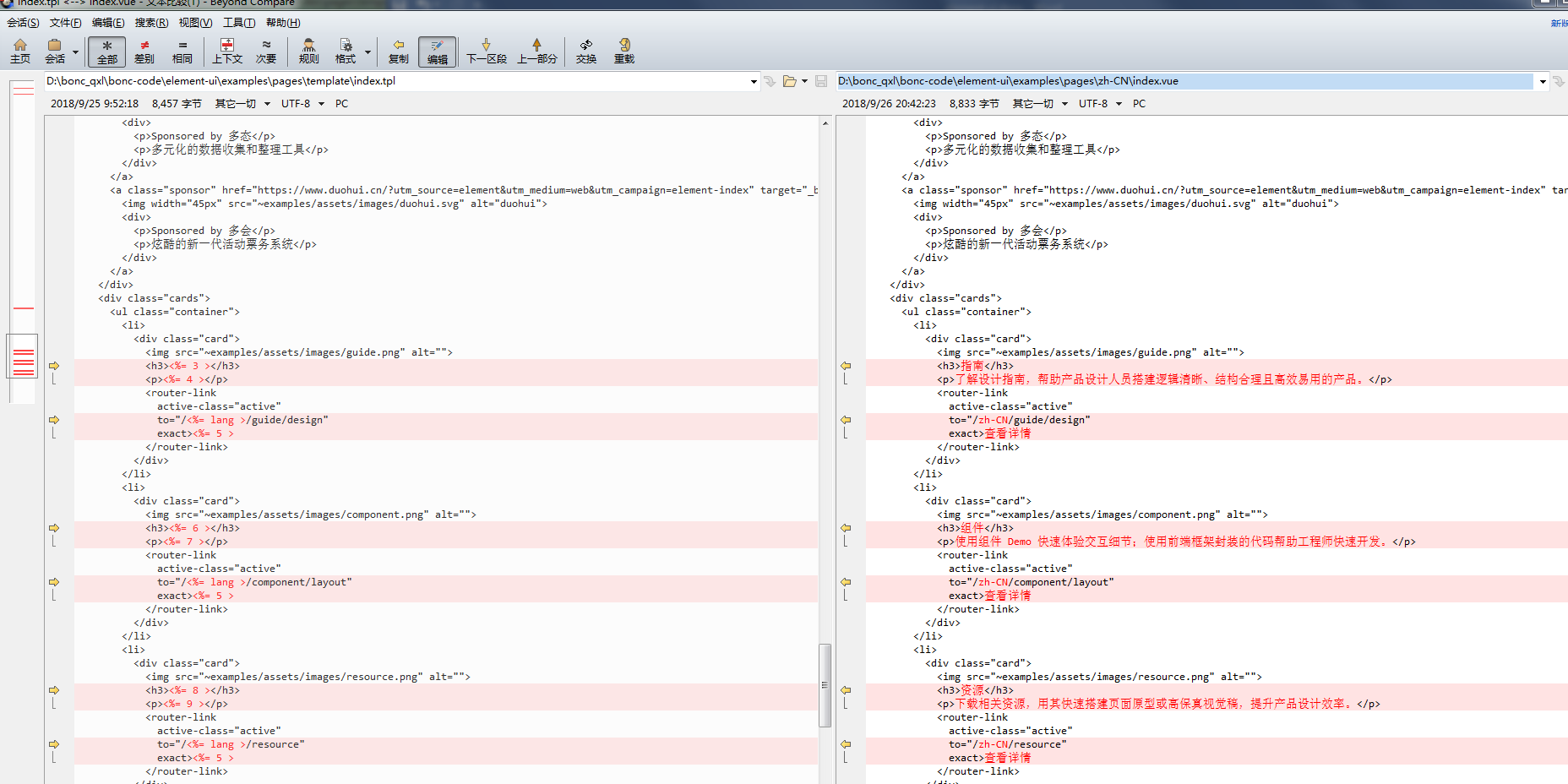


1. 这个是怎么做到的？js

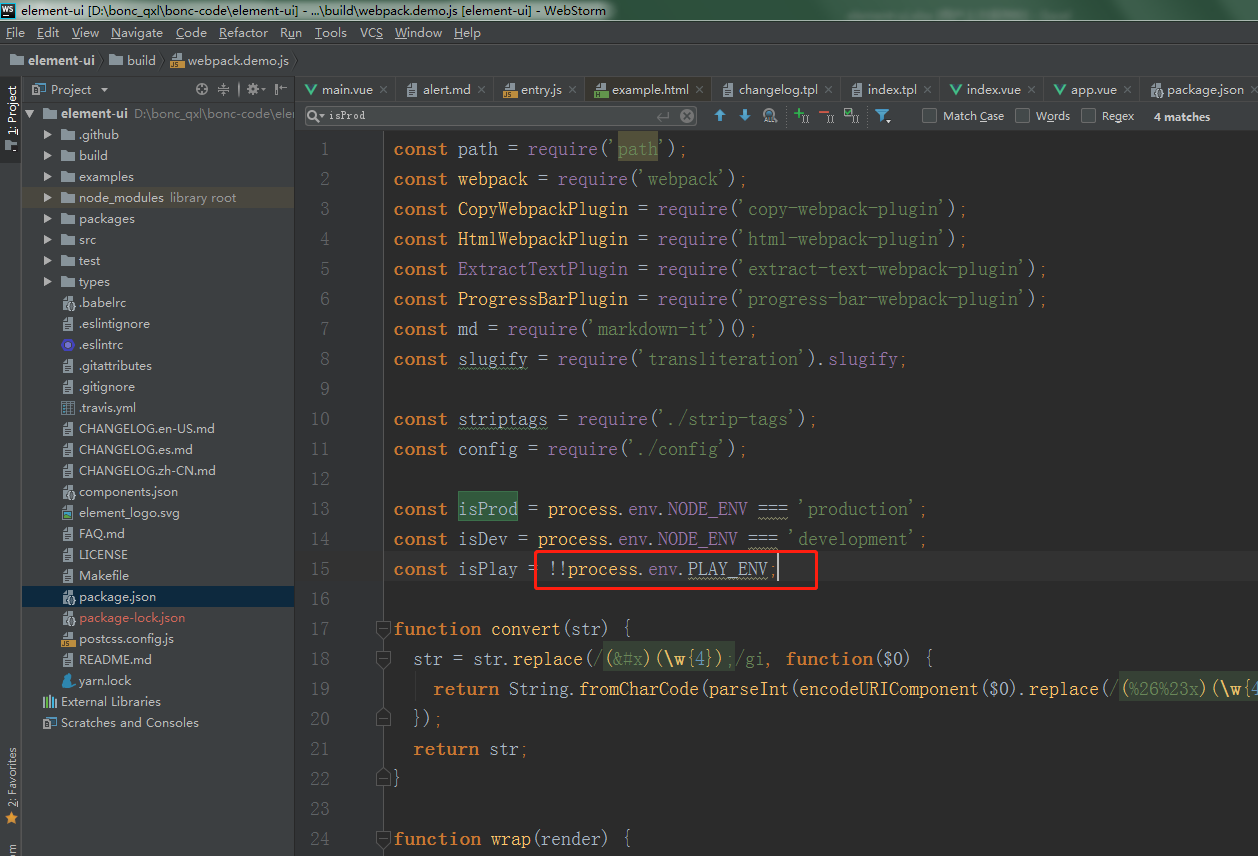


1. 这2个地方的文件是差不多的，对比其内容可知，.tpl是生成国际化的总模板：

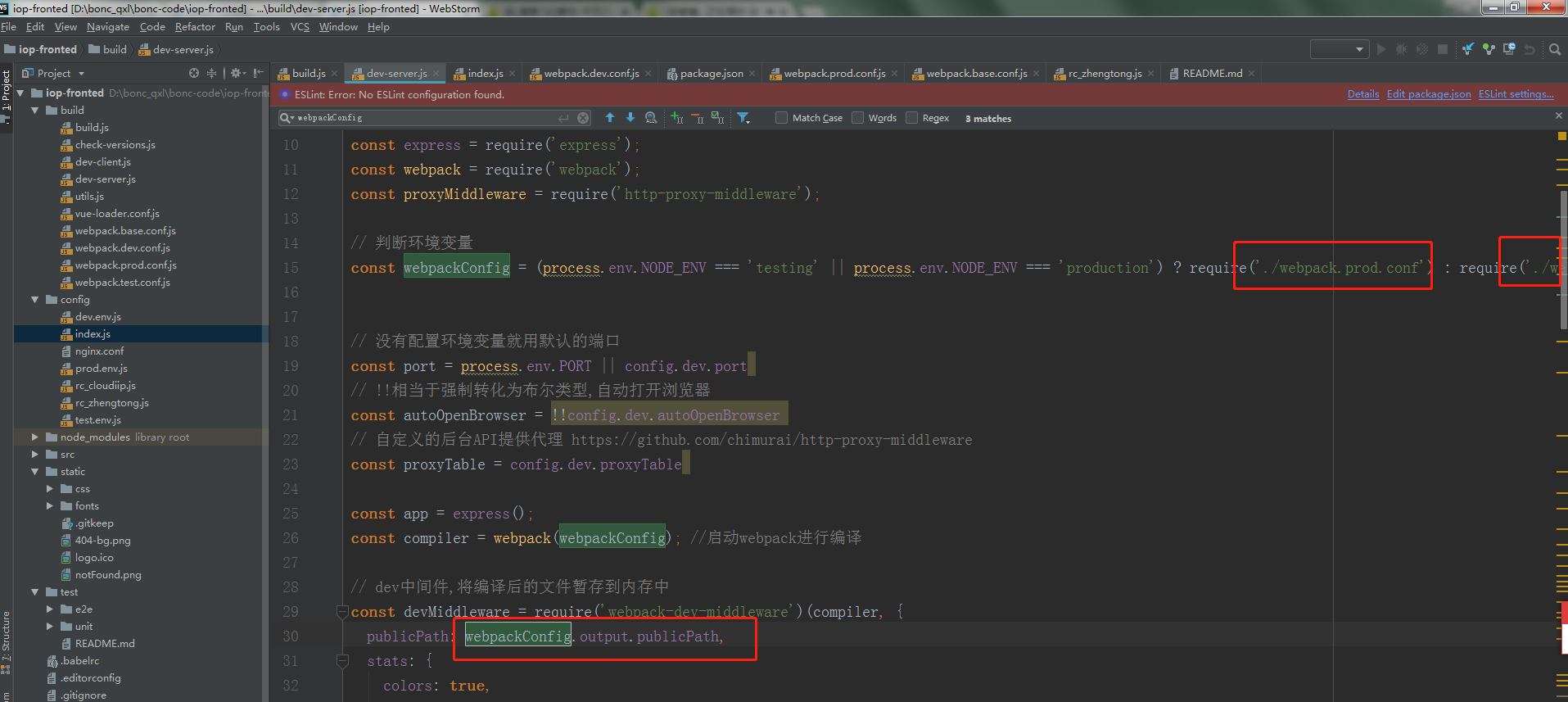




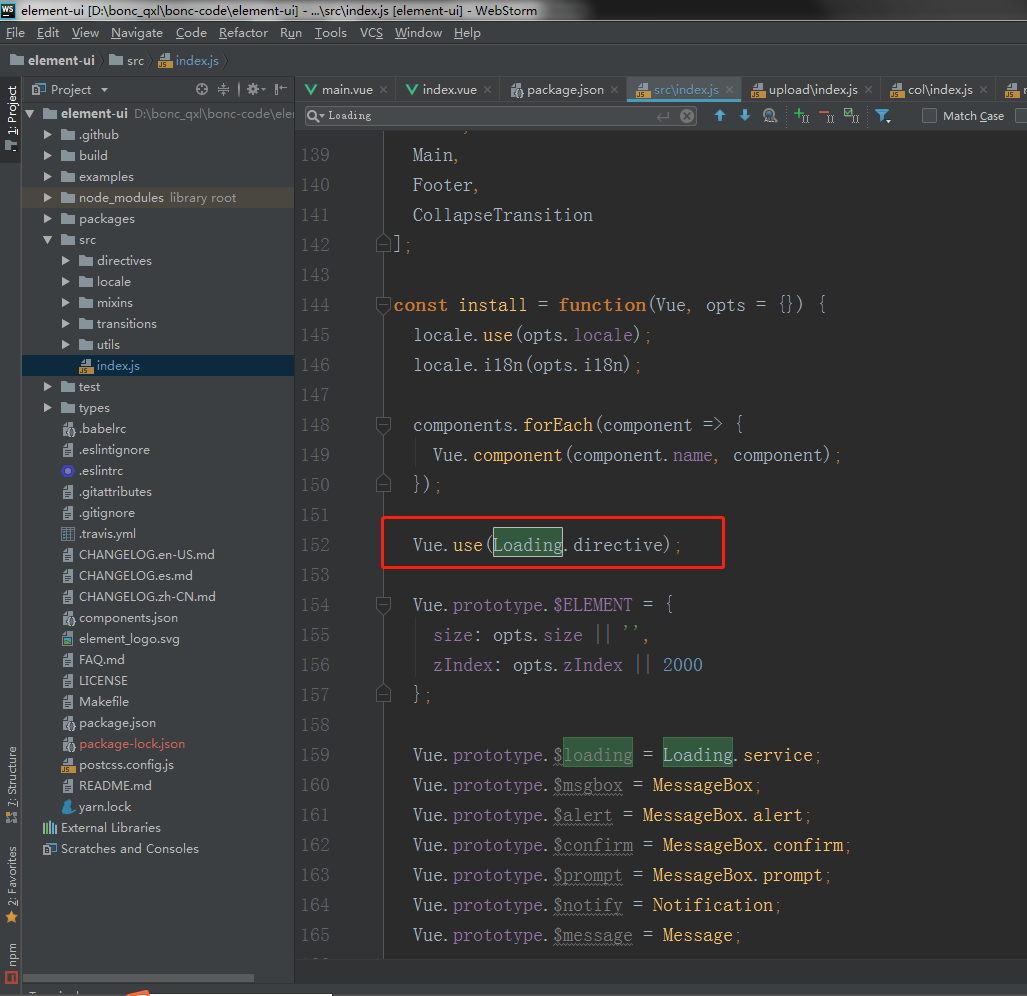
1. 这是什么意思？



1. 这2个文件里都没有这个字段啊？难道这个中间件没有起作用吗？



1. 这个为啥和其他的都不太一样呢？貌似和install有关，有空好好对比一下



1. 这种是为啥加$符号呢？

