# 什么是webpack

## 1，初识webpack

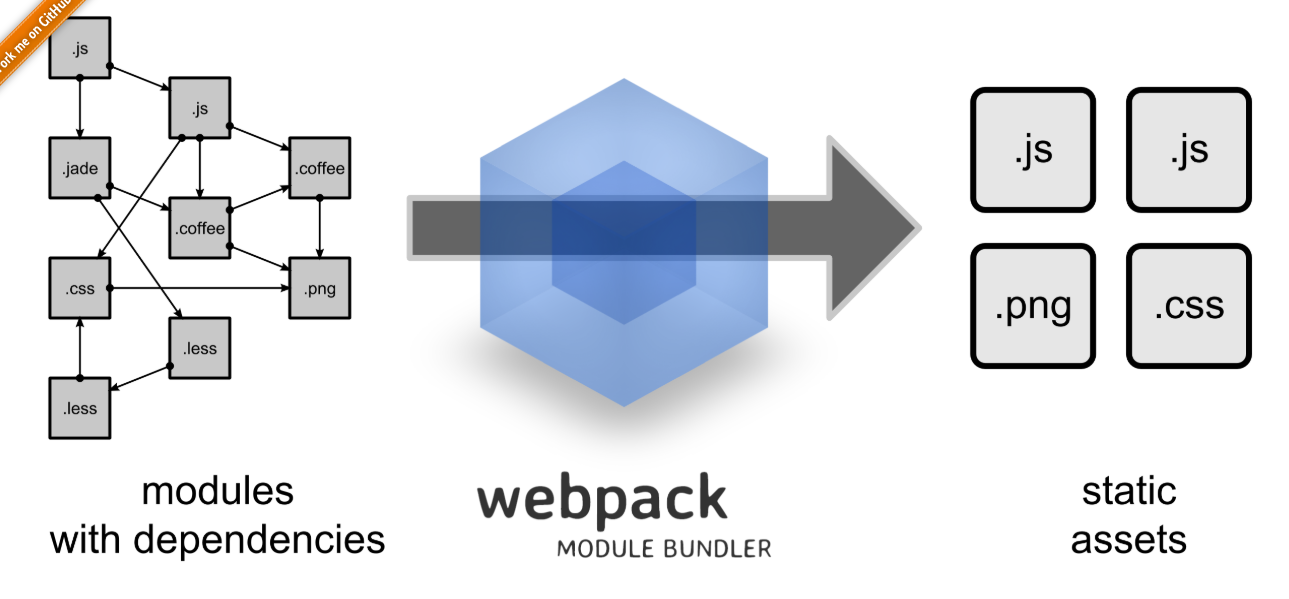
官网：<https://webpack.github.io/>

中文网：<https://www.webpackjs.com/>

Webpack是模块化管理及打包工具/模块打包机（ module bundler）。可以把我们写的项目进行打包。一般开始vue项目，都是通过脚手架创建的，创建如下：



运行项目：npm run serve，然后进行开发，这种环境叫开发环境。当项目开发完毕，上线的时候，需要项目进行打包。通过npm run build。因为项目中资源非常多，资源之间的关系也非常复杂。这个时候就需要使用webpack来解决这个问题。如下：



另外webpack可以把一些高级的语法打包成低级语法，这样浏览器就可以识别。如今的Web可以看做是功能丰富的应用，拥有复杂的js代码和众多有依赖包，为简化开发复杂度，提高开发效率，涌现了大量的实践：

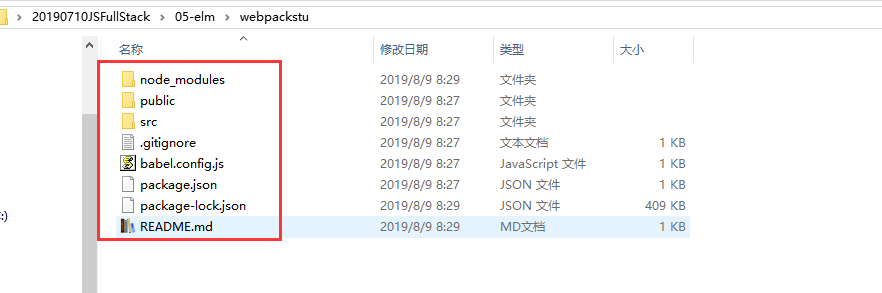
* es6 模块化
* 全新的js写法，如ES6、TypeScript、CoffeeScript
* Less、Sass等css预处理器

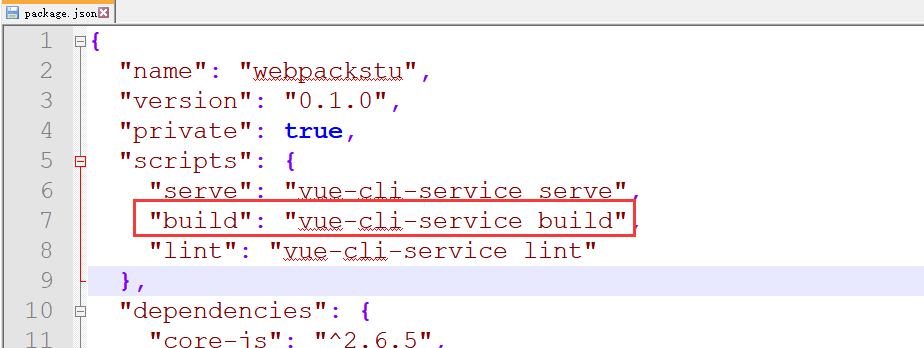
这些实践大大的提高了开发效率，但是浏览器不能解析这些代码，需要进行额外的处理才能让浏览器识别，而手动处理又非常繁琐，于是就需要一些专业的工具。

* Browserify -- 实现浏览器端的commonjs模块化，功能比较单一
* Gulp --- 优化项目的一个工具
* Grunt
* **Webpack --- 最好的模块打包机**
* Bower --- 类似于npm，管理各种包/模块的
* Yeoman -- 构建项目结构的
* Fis（百度）-- 和webpack类似

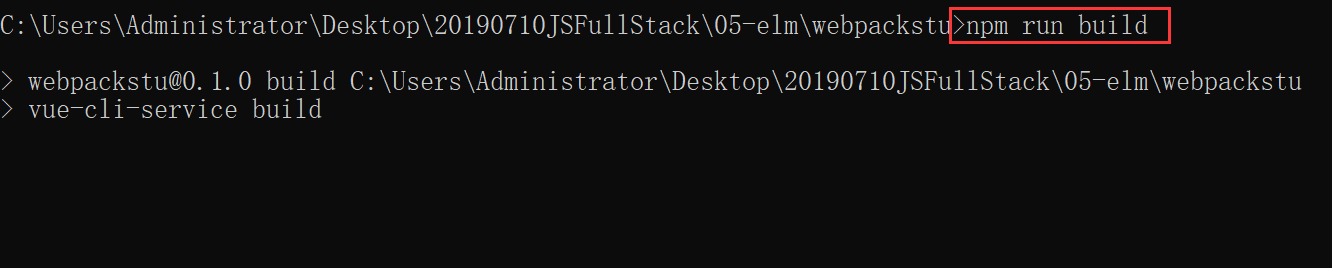
## 打包vue项目

打包之前，项目的目录结构如下：

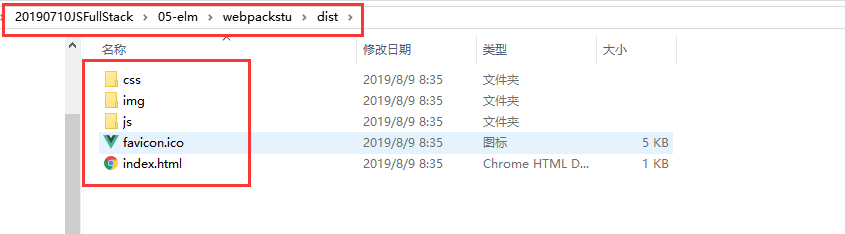




可以把上面的创建的项目进行打包，如下：



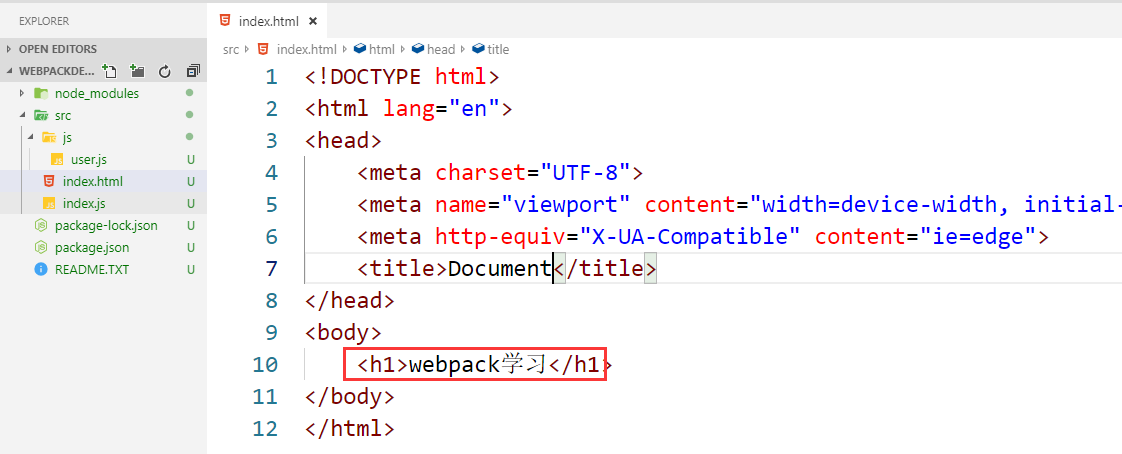
打包之后会生成一个dist目录，到时候项目要上线时，只需要把这个dist目录扔到服务器就OK了，dist目录结构如下：



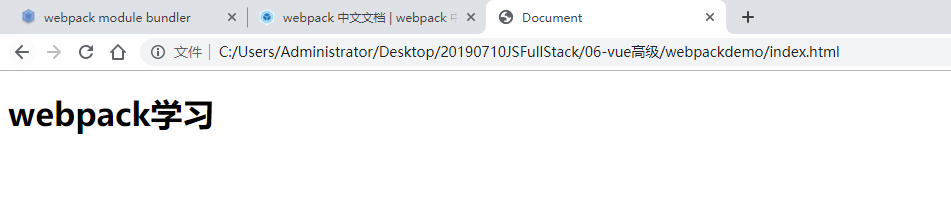
所以我们要研究它是如何打包的，如何把高级的东西，打包成低级的东西？

## 入门案例

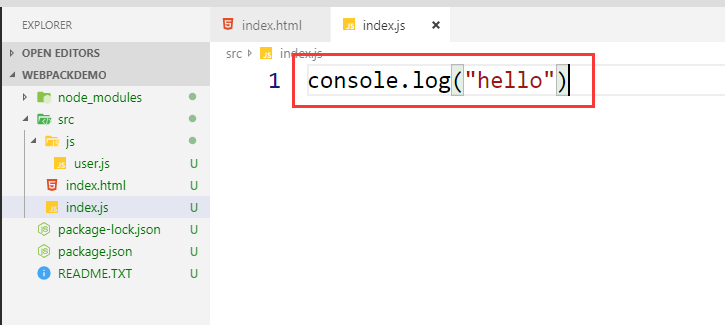
创建一个html文件如下：



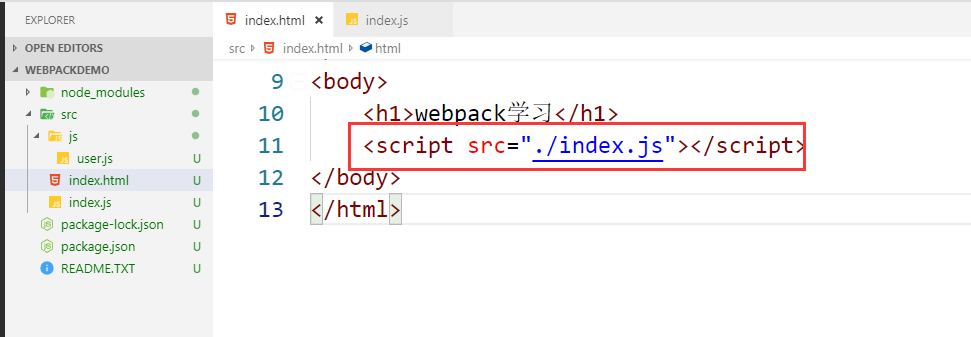
浏览器可以识别这个文件，所以可以使用浏览器直接打开这个文件，如下：



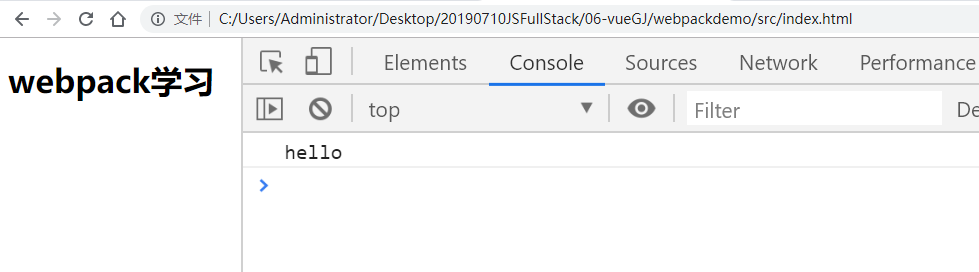
创建一个index.js文件，如下：



在index.html中引入，如下：

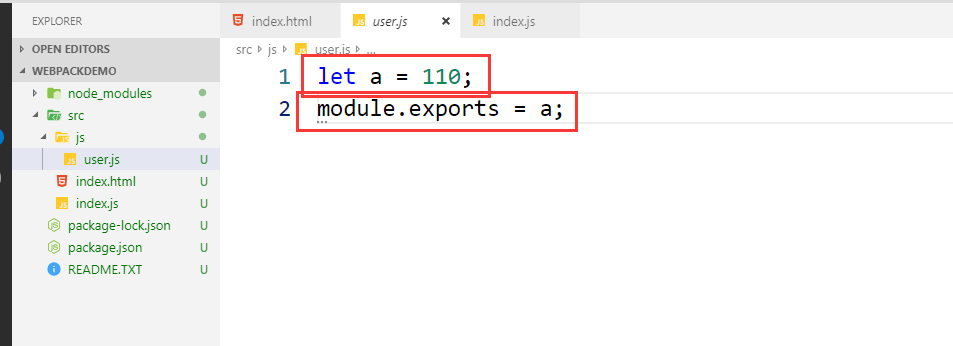


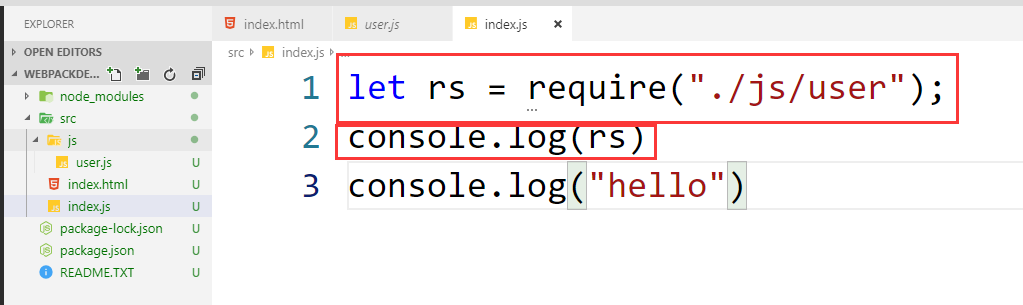
效果如下：



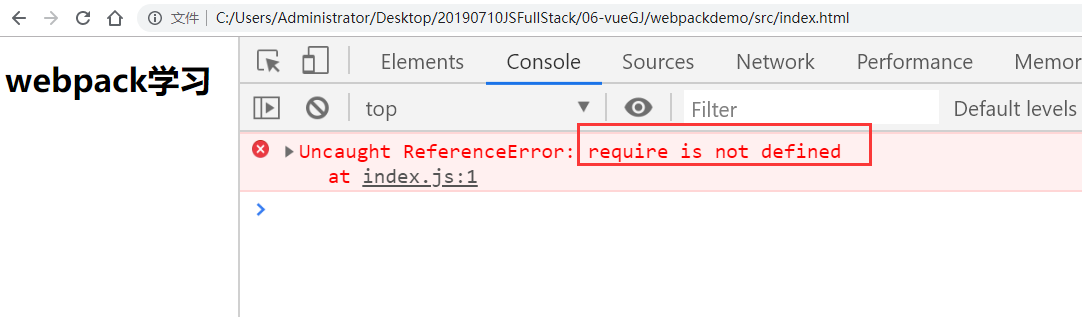
也就是浏览器默认情况也是可以识别一些低级的js的。

创建一个user.js文件，在index.js中引入这个文件，如下：





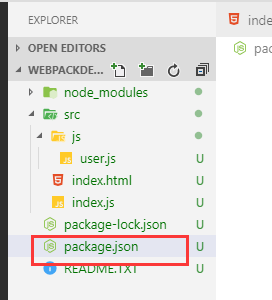
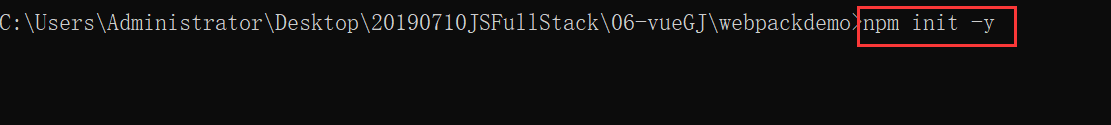
浏览器报错了，如下：



这是就需要使用webpack把浏览器不能识别的语法转化成浏览器可以识别的语法了。

## 安装webpack

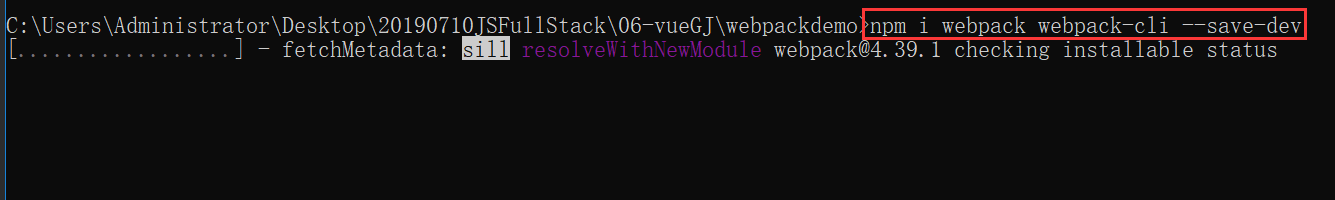
安装之前，先去npm init -y一下。先去生成一个package.json，如下：



使用webpack，需要安装一把，安装如下：



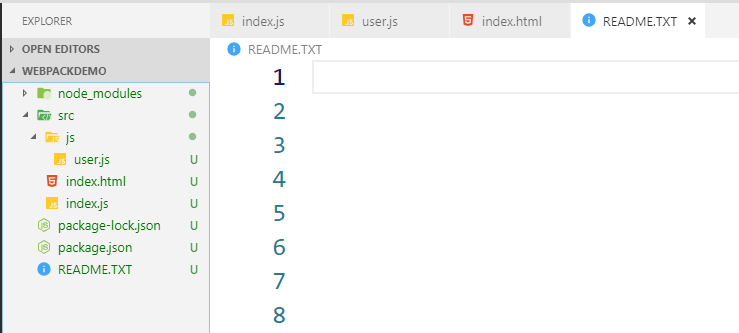
安装如下：



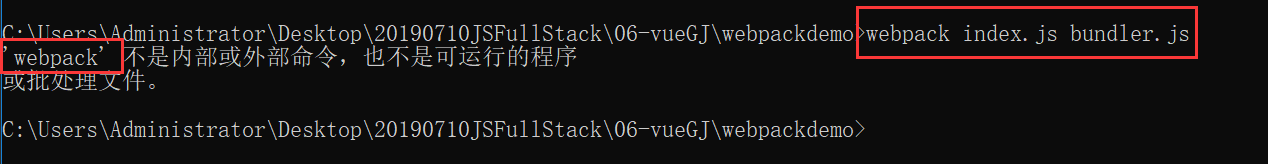
就把你安装的模块记录到package.json文件，如下：



目录结构如下：



安装完后，把index.js进行打包处理，就可以使用webpack进行打包处理了，如下：



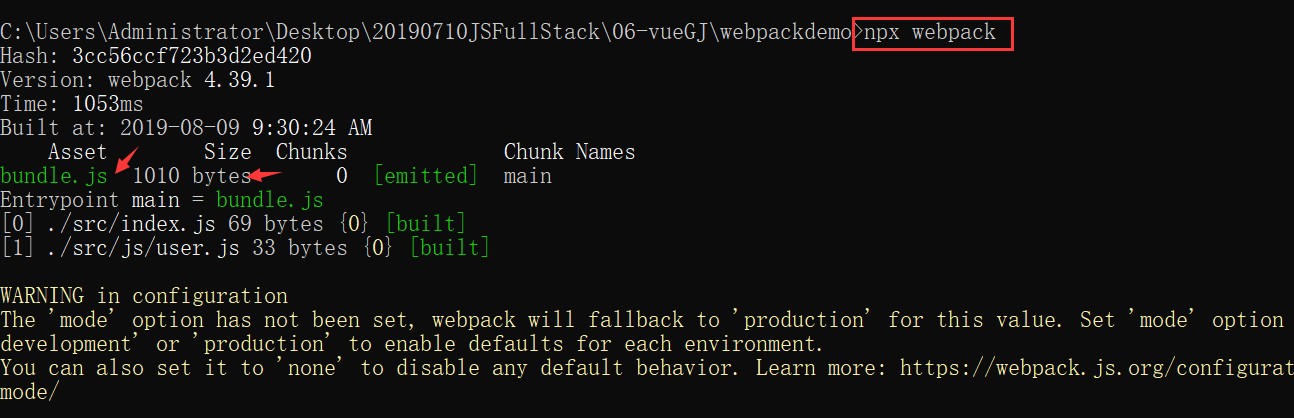
为什么是报这个的错误，你要使用webpack这个指令的话，你必须全局地安装这个webpack依赖。可以通过npx来解决这个问题。

学习webpack，就是学习如何去配置webpack。

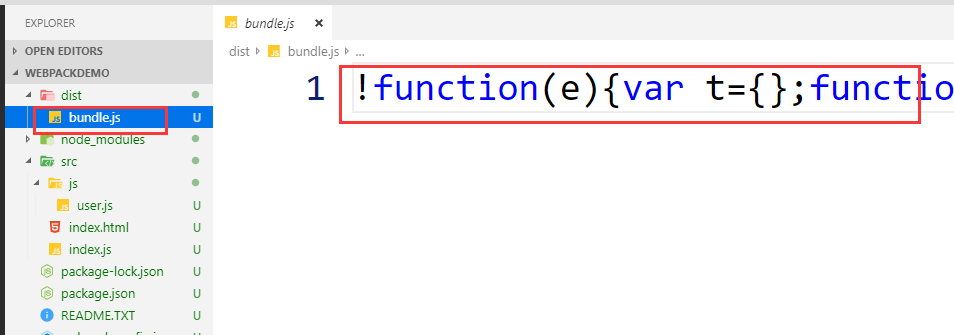
在打包处理之前，你需要创建一个配置文件，叫webpack.config.js，如下：



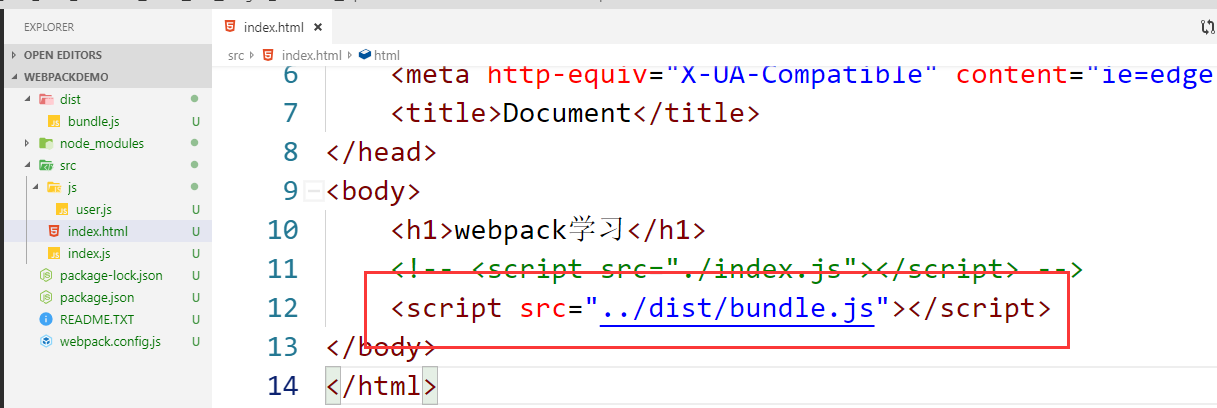
打包如下：



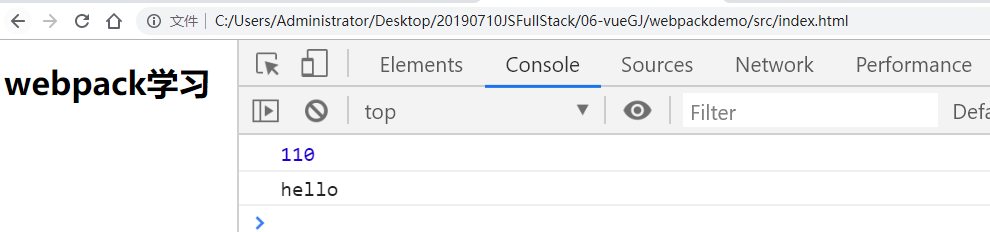
在项目中，如下：



在index.html中引入打包后的文件，如下：

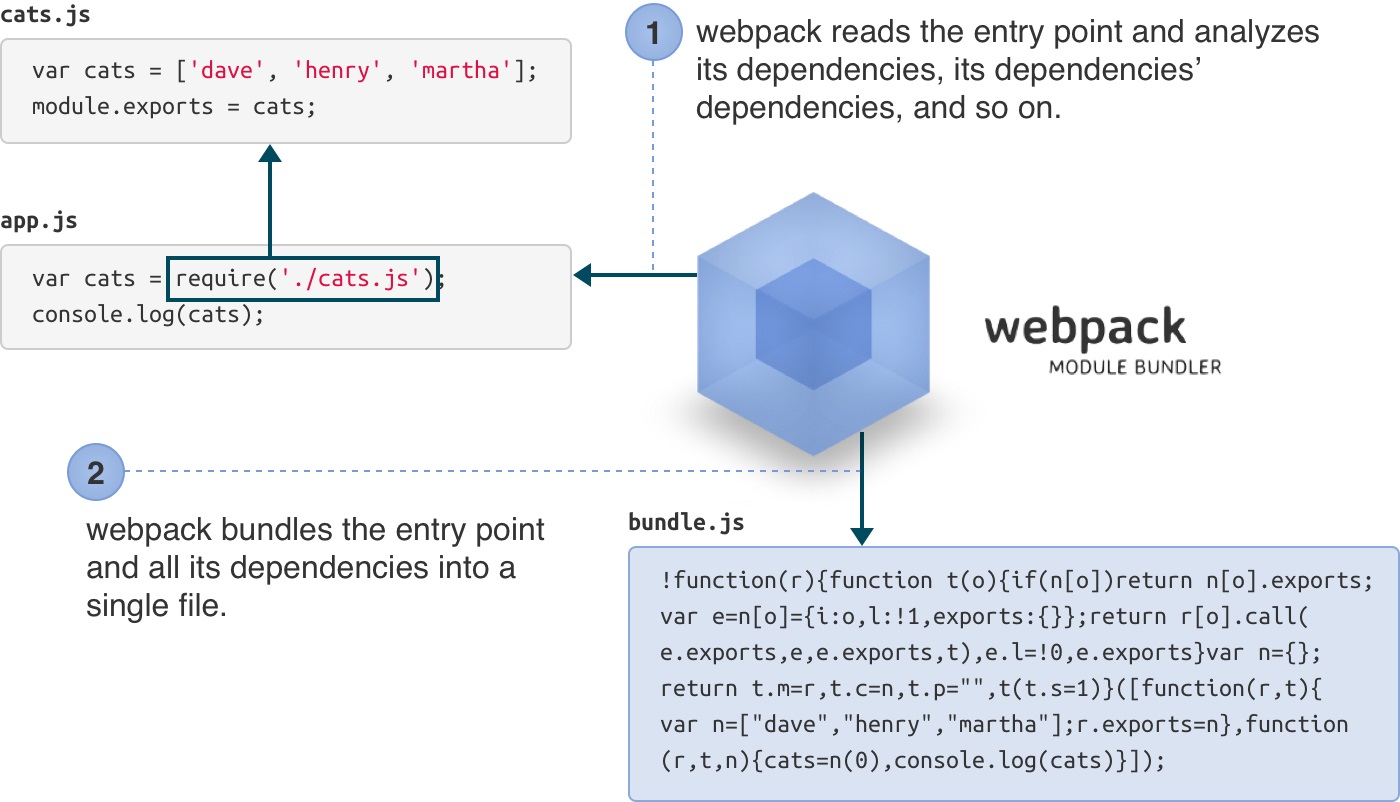


效果如下：



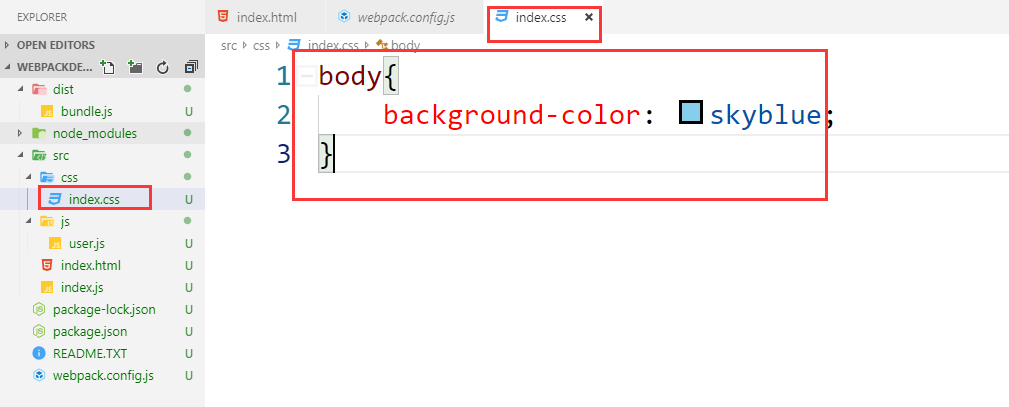
## 5，webpack是怎么工作的

查看bundle.js文件，可以发现：不仅app.js里面的内容被打包进bundle.js，app.js所依赖度cats.js也被打包了进来。webpack打包原理是边读边分析入口文件的依赖，分析入口文件依赖的文件的依赖，以此类推，只要有依赖都去分析并打包到输出文件中。

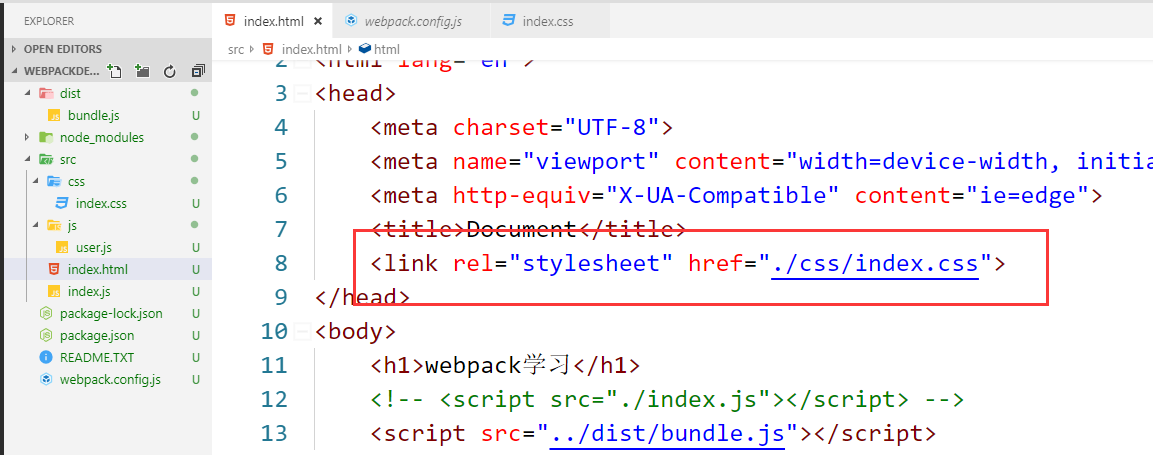


## 6，打包css文件

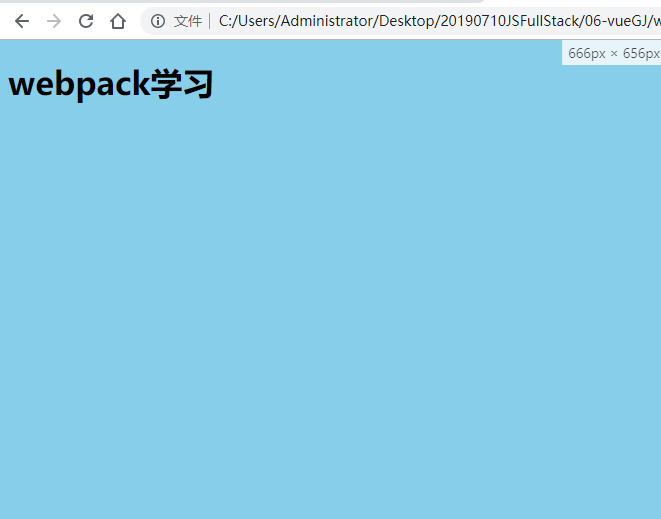
创建一个css文件，如下：



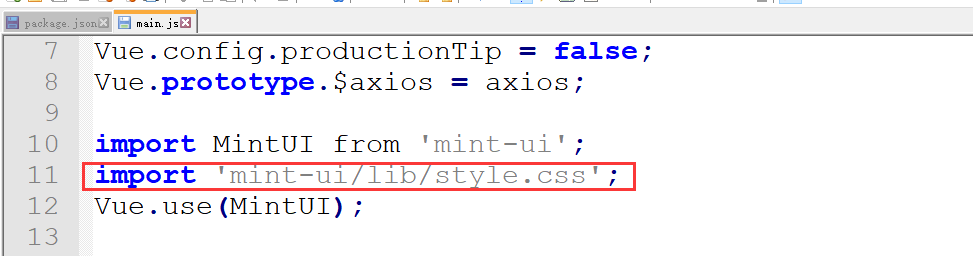
在index.html中引入，如下：



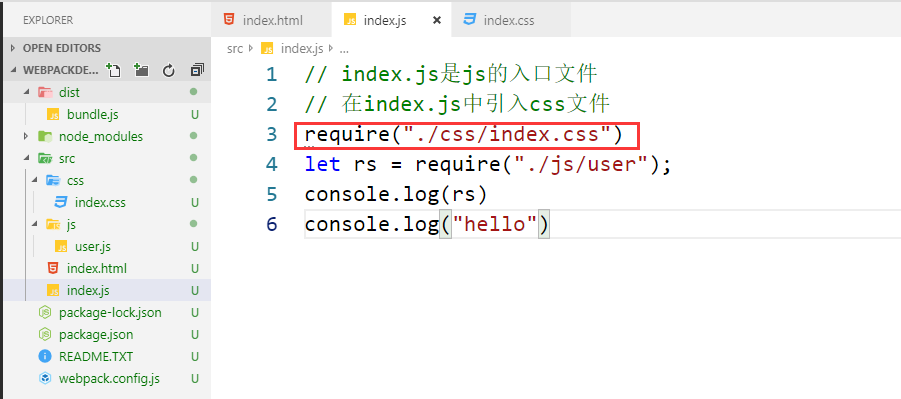
效果如下：



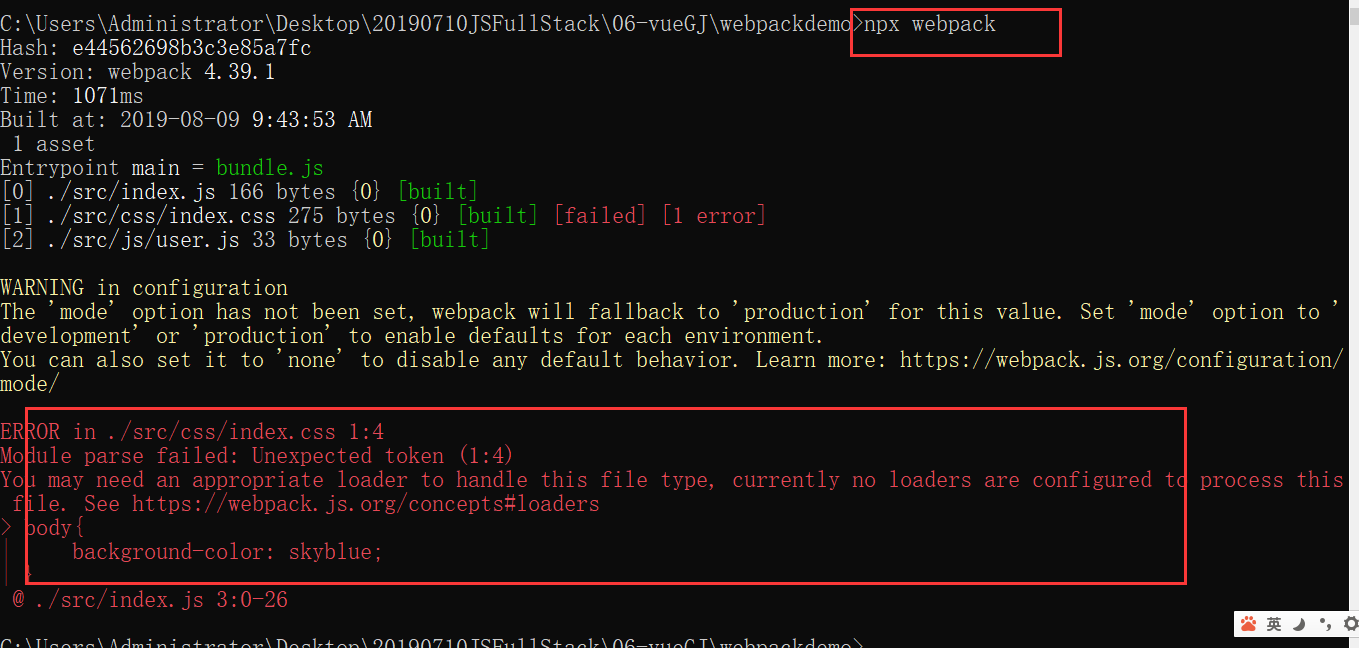
之前在写vue项目时，并没把css放到html文件中引入，引入的方式如下：



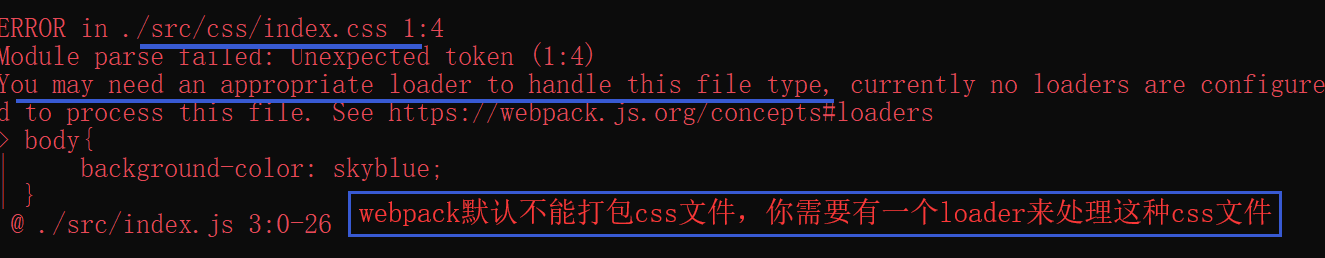
我们现在也想通过上面的方法来引入，如下：



现在进行打包，如下：

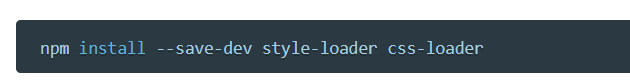


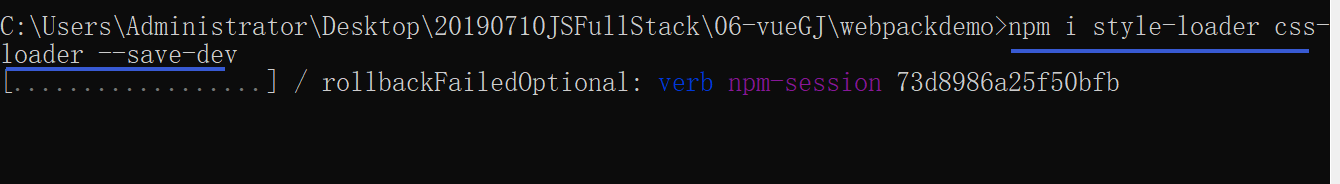
说明了，webpack默认情况下不能打包css文件。需要安装一个合适的loader来处理css文件，如下：



处理css文件，需要两个loader

* style-loader
* css-loader

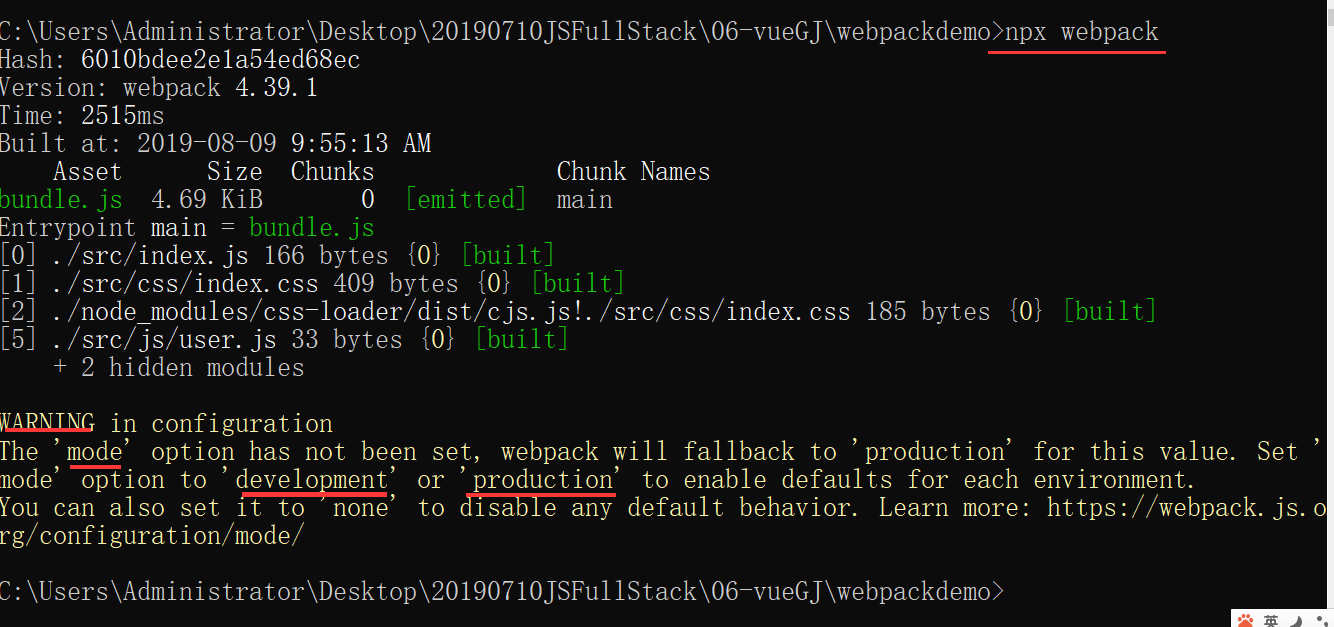




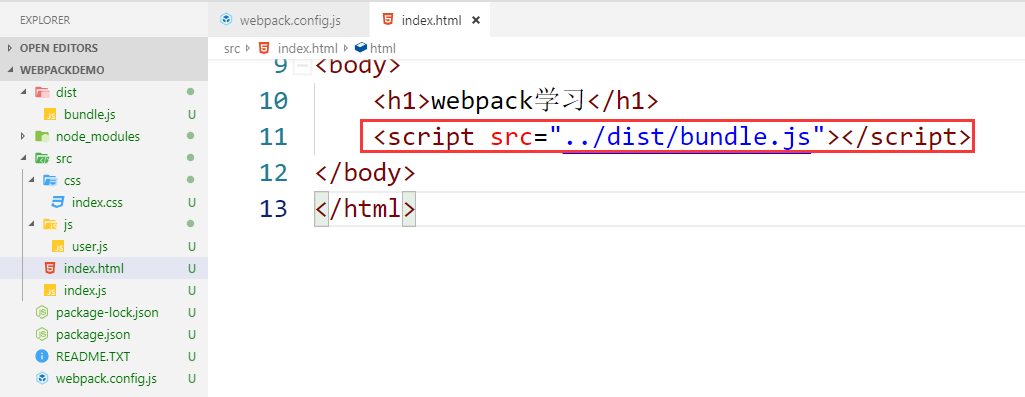
安装完后，还需要去配置，配置如下：



打包：



把打包后的文件放到index.html中，试一下，是否OK，如下：



效果如下：

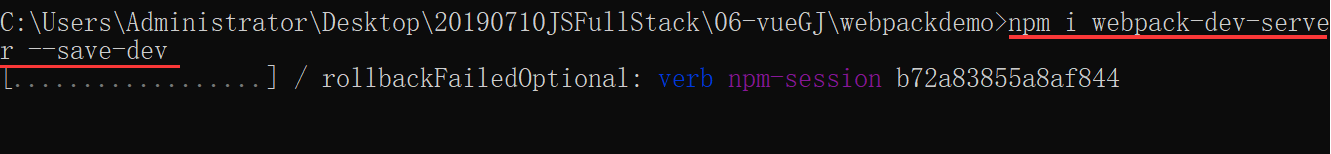


## 构建本地服务器

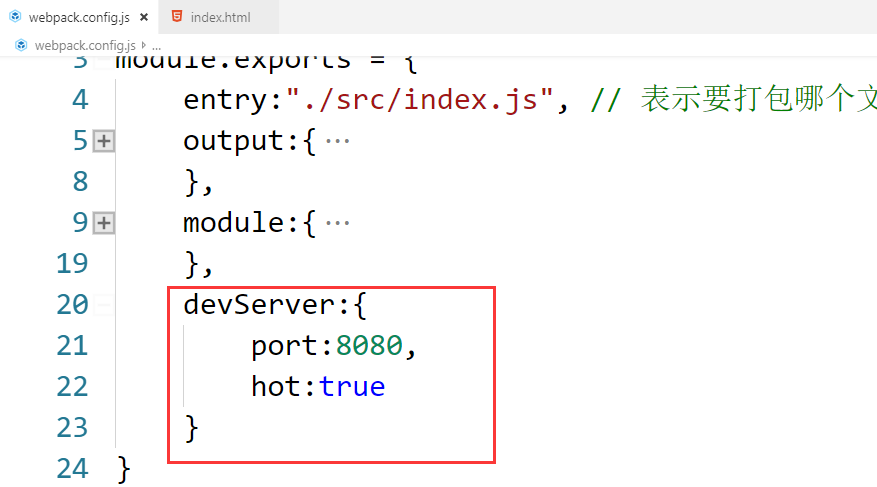
需要安装一个模块 ：

npm install --save-dev webpack-dev-server

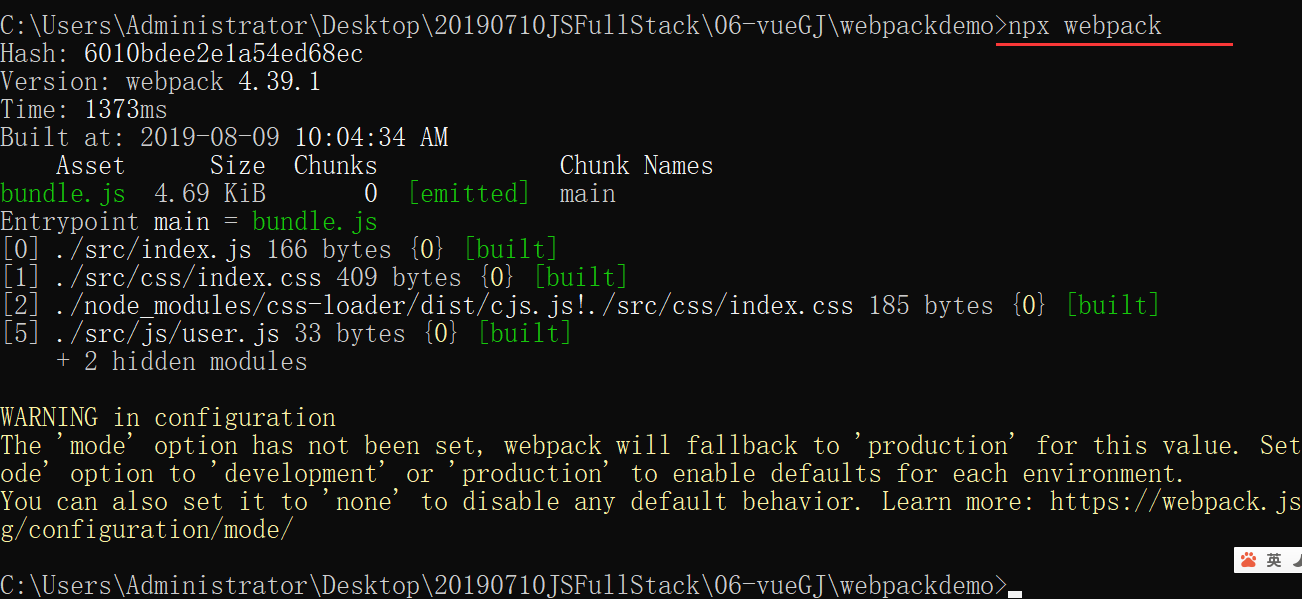
安装如下：



配置如下：



打包如下：

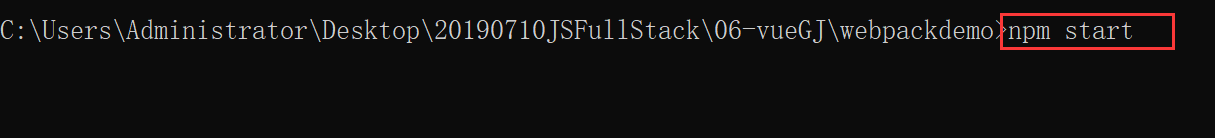


上面的写法还不行，我们最终也是需要启动一个服务器。

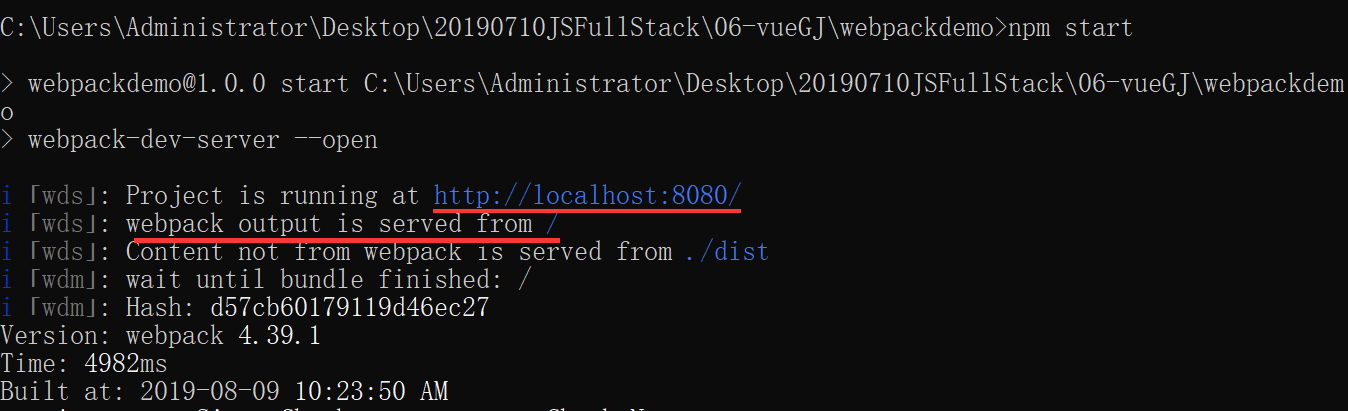
需要在package.json，配置脚本，如下：

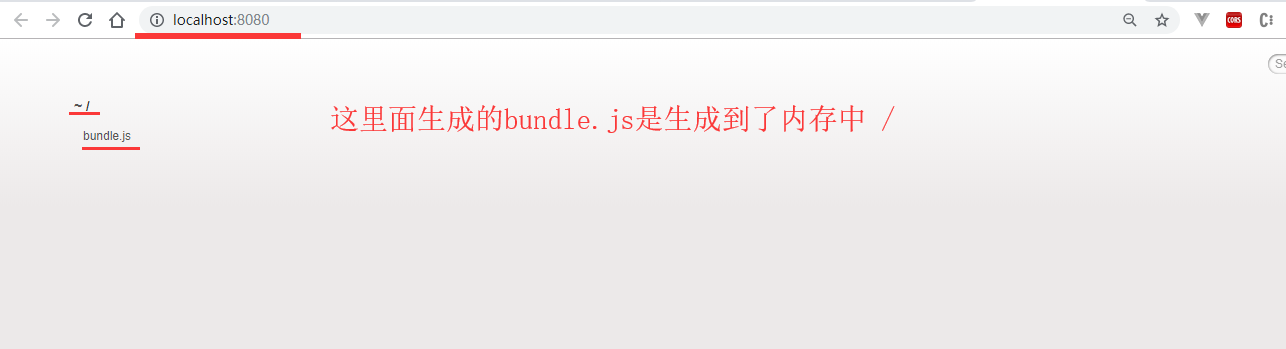


测试如下：



人家把你把包后的结果给输出了，如下：





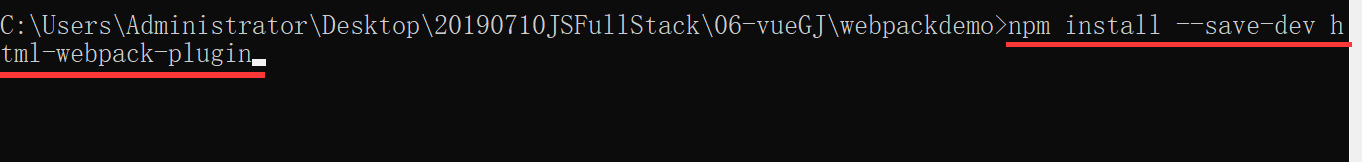
上面已经开始了一个服务器，并且可以进行打包，打包完后，生成了bundle.js这个文件了，怎么去用？

不能在index.html直接去引入bundle.js这个文件了，因为通过webpack-dev-server，人家是把打包后的文件，生成到了内存中。

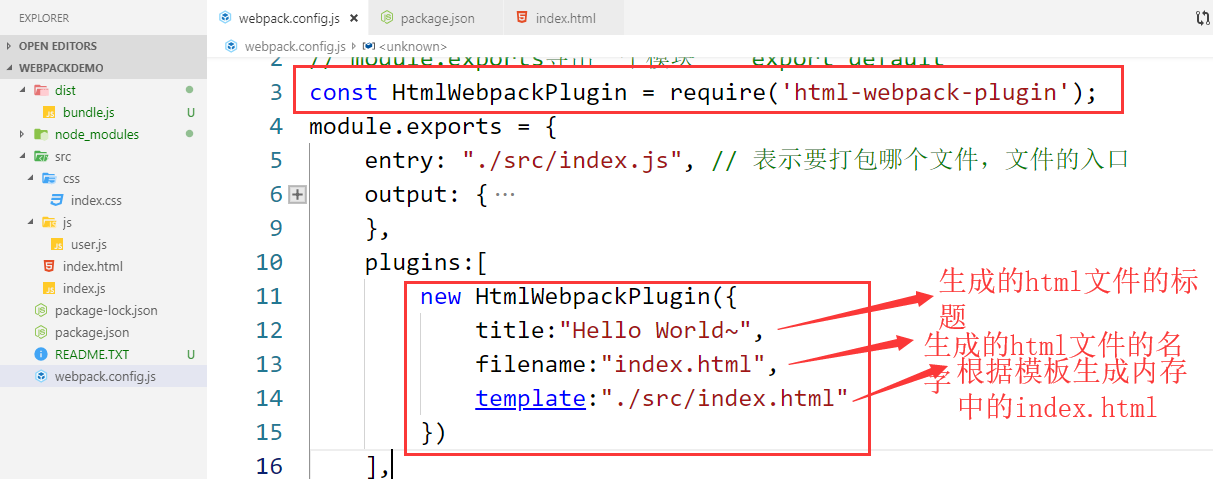
解决方案：把index.html也载入内存，在内存中index.html就可以引入bundle.js。

## HtmlWebpackPlugin

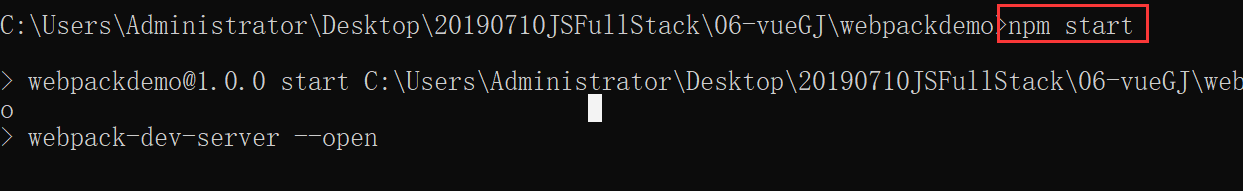
htmlwebpackplugin是一个插件，用的时候，也是需要安装的，安装如下：



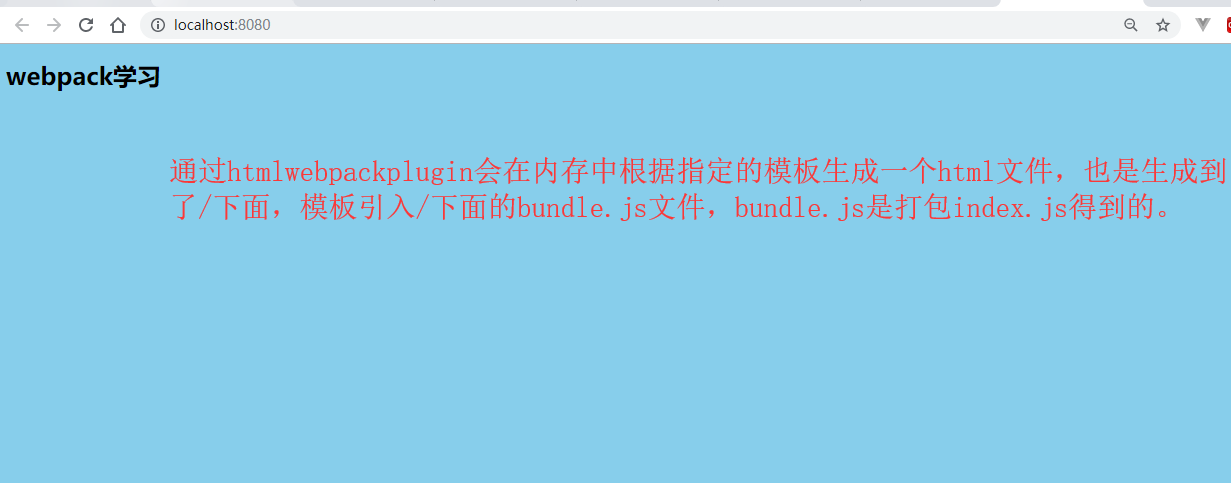
配置如下：



重启服务，如下：



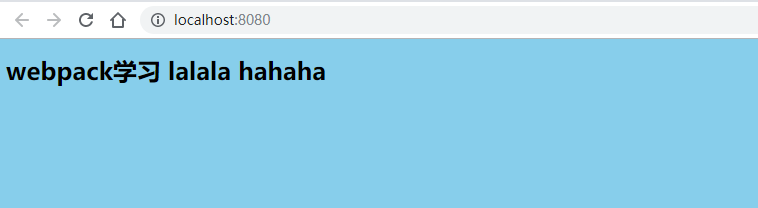
效果如下：



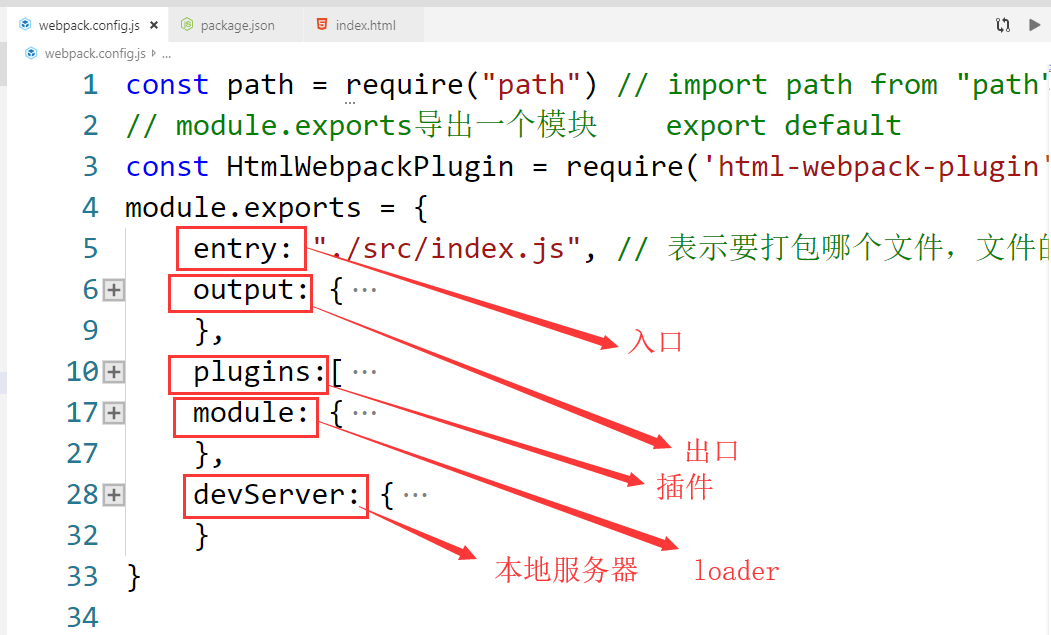
修改代码如下：



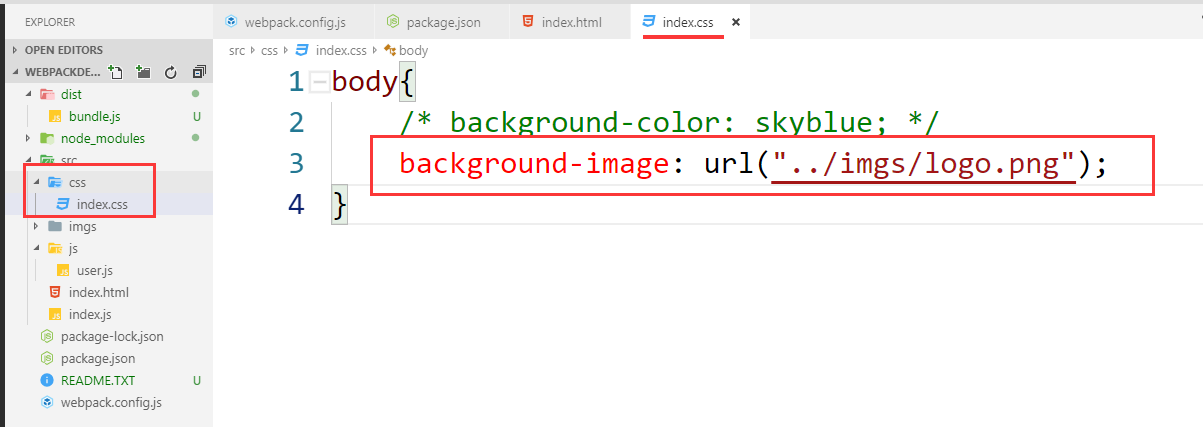
查看浏览器如下：



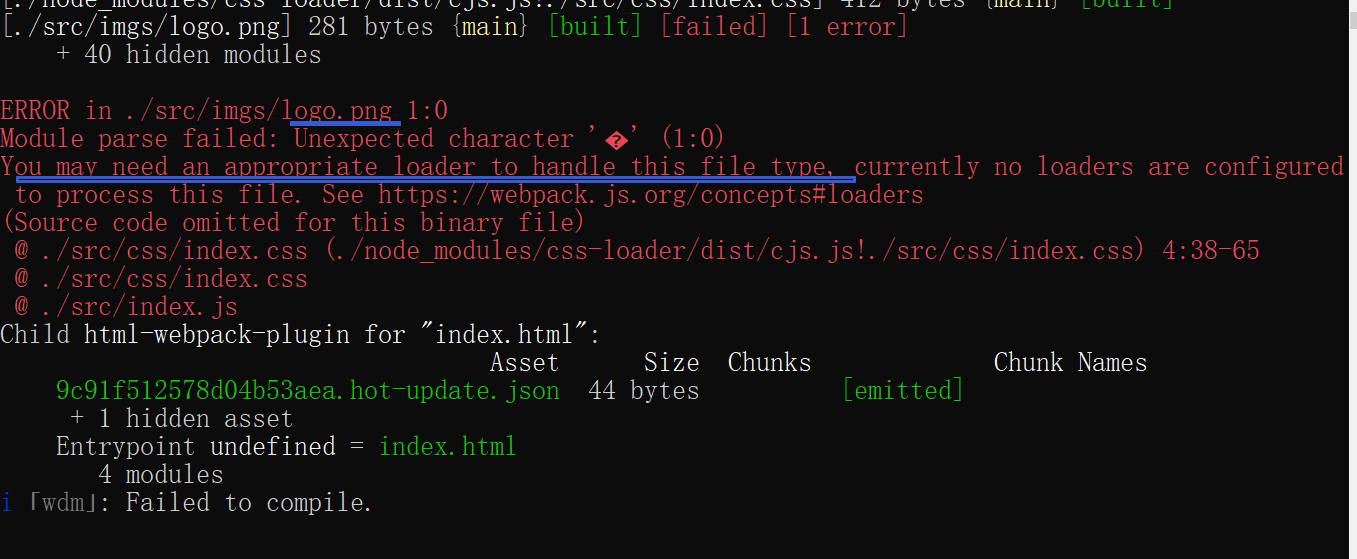
现在配置的如下：



## 打包图片

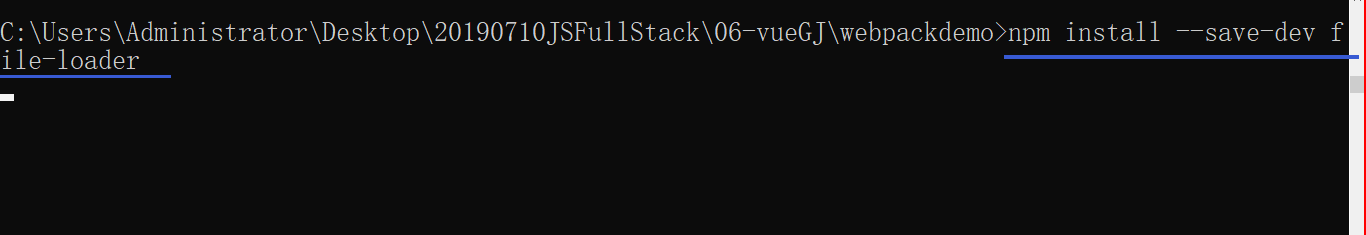


报错了，如下：



安装loader，配置loader，如下：

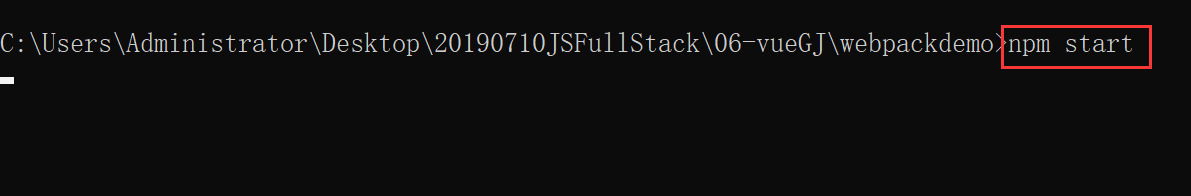




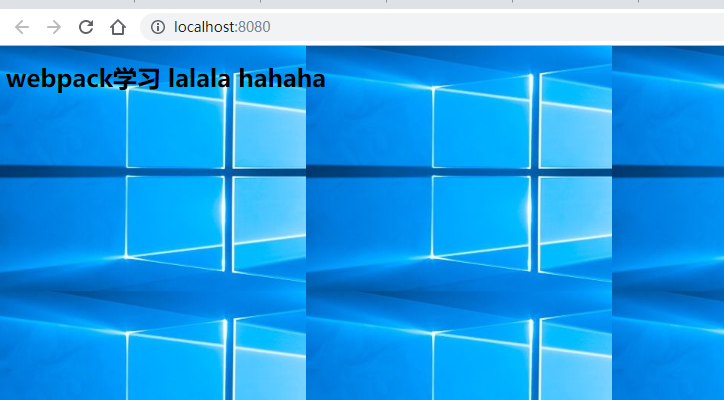
配置，如下：



重启服务，如下：

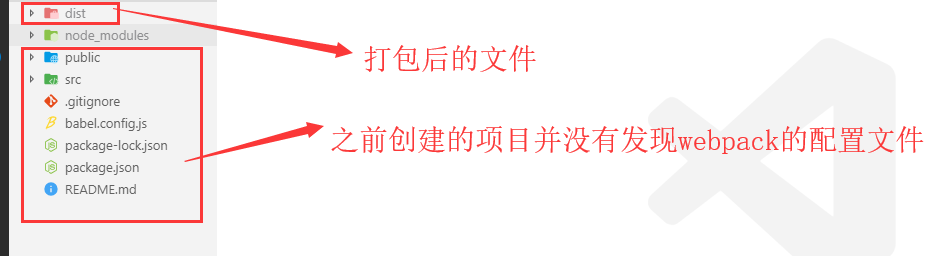


效果如下：



# vue-cli是如何配置webpack的

之前创建的项目结构如下：



默认情况下，在vue-cli3.x版本后，把webpack的配置文件给隐藏了，可以把这个配置文文件显示出来。

## 把配置文件显示出来





输入命令如下：

