# Webpack配置

### 配置package.json

$npm init

$npm init -y

### 安装 webpack 包

$cnpm i webpack webpack-cli -D //运行环境

在项目目录下创建文件夹

./src/index.js

./dist/index.html

Index.html文件下写入html代码

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <meta http-equiv="X-UA-Compatible" content="ie=edge">

    <title>Document</title>

</head>

<body>

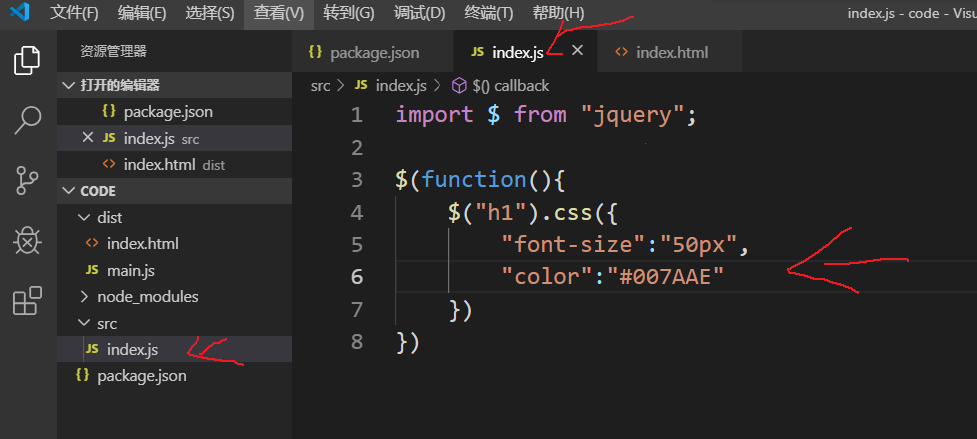
    <h1>欢迎来到webpack</h1>

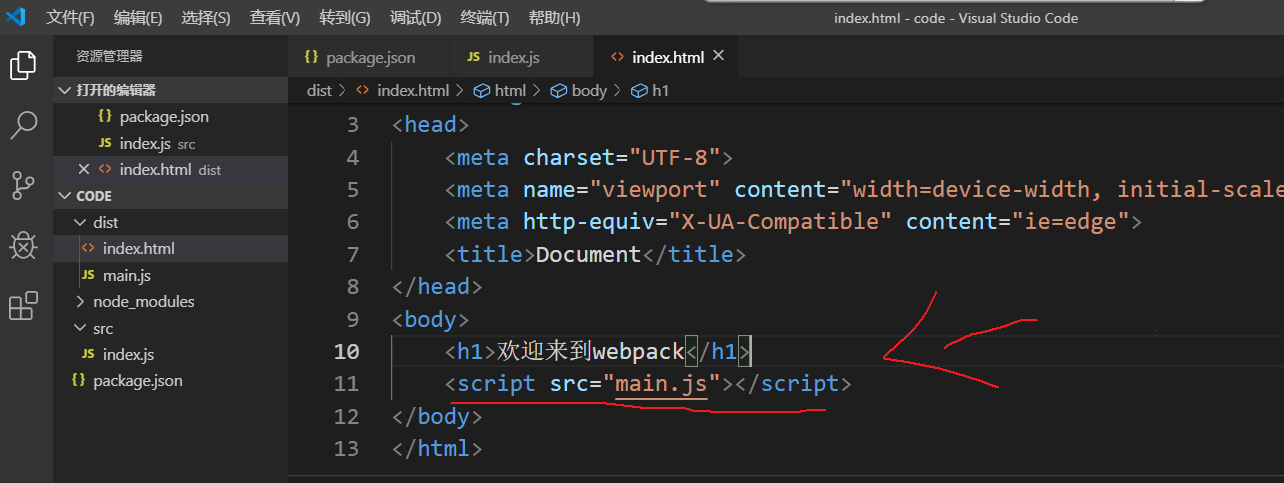
</body>

</html>

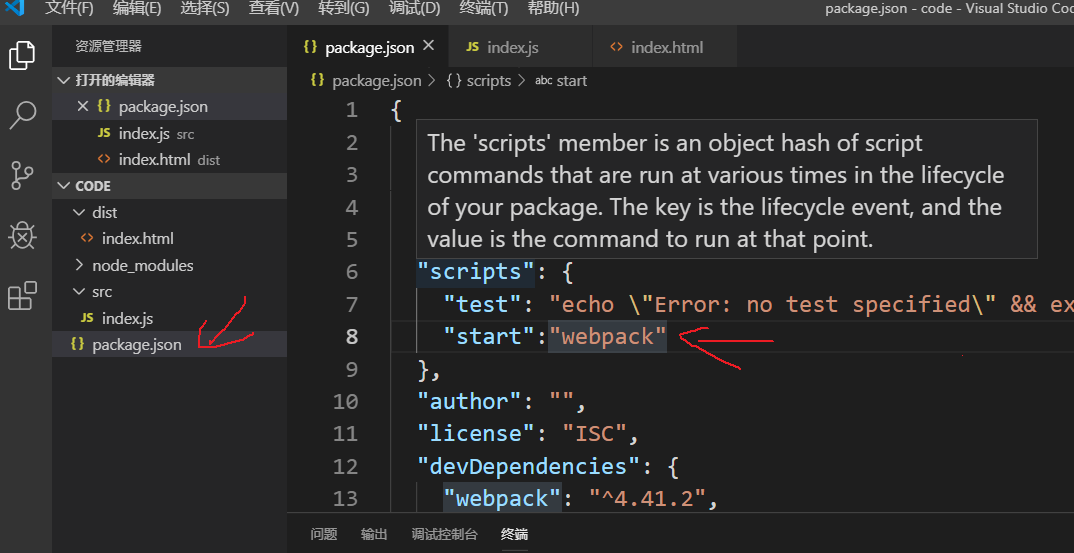
### 引入jquery包

$cnpm i jquery -S





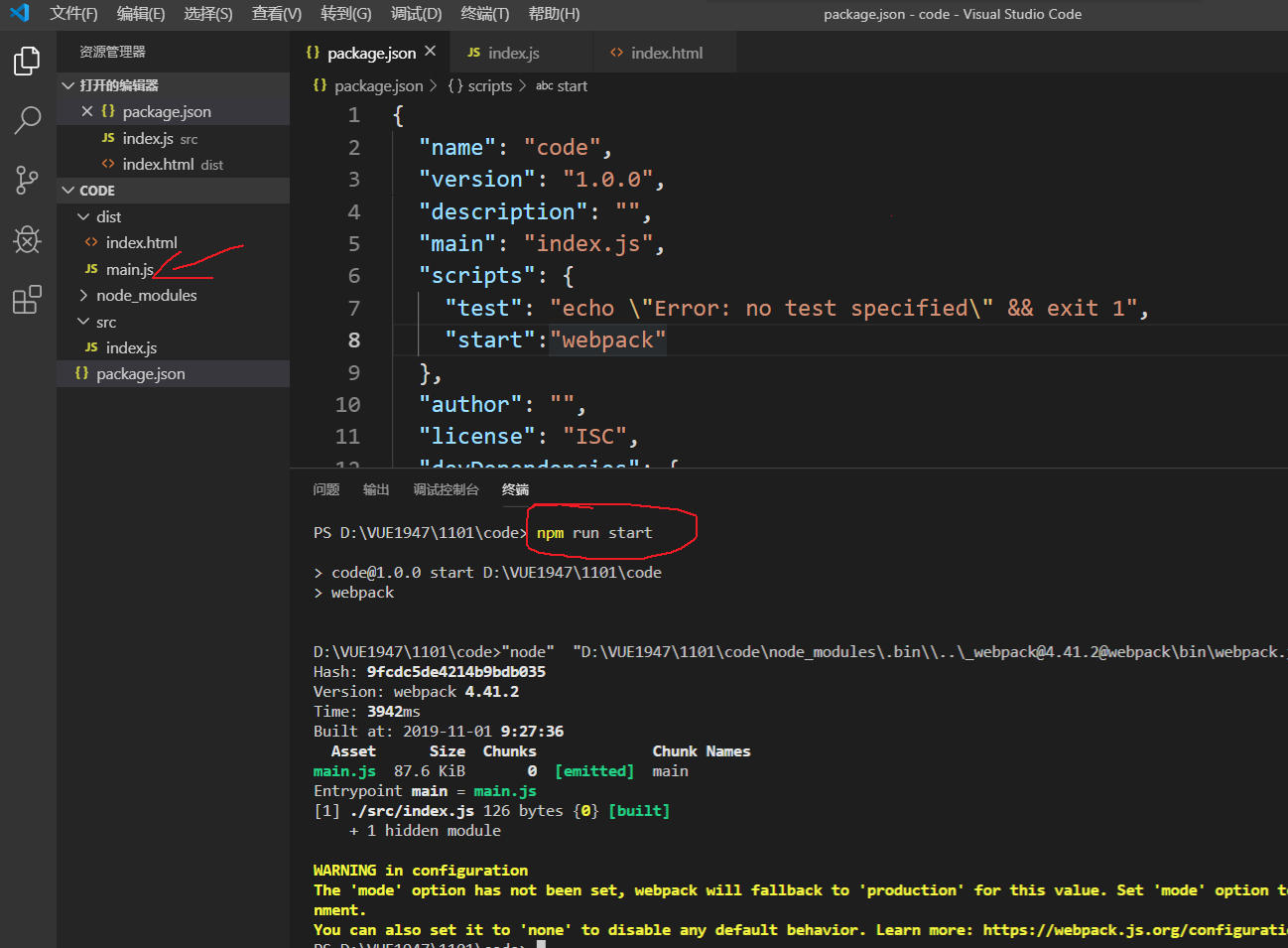
### 配置package.json 下scripts命令



### 测试

运行命令： $npm run start

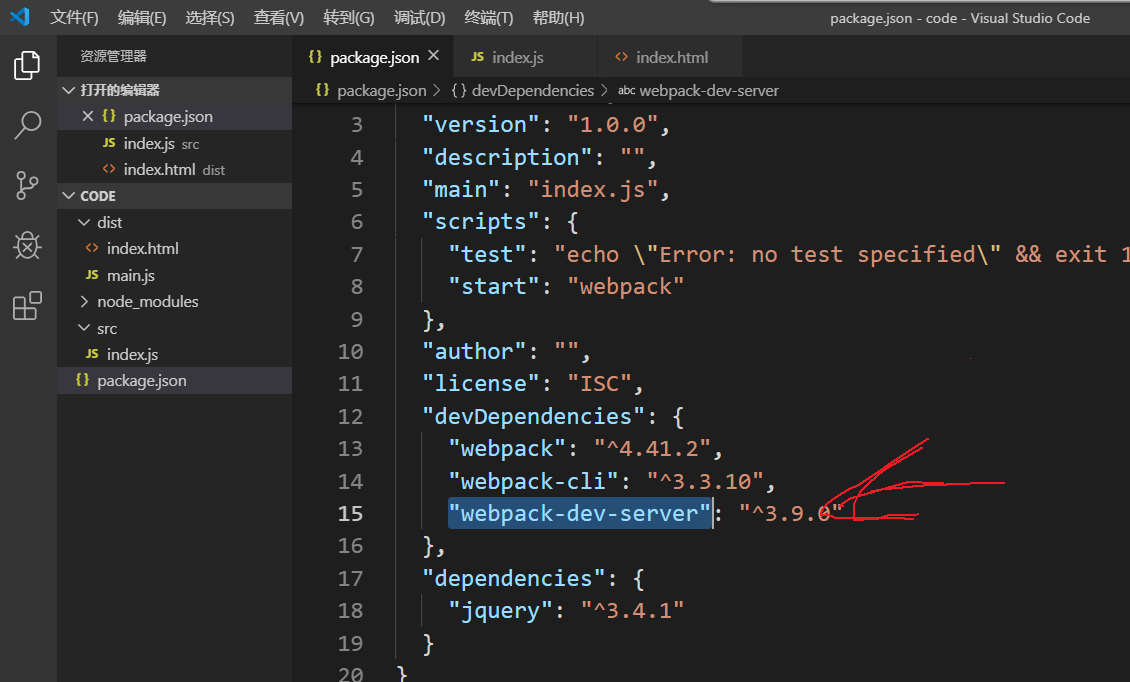
结果在./dist/main.js



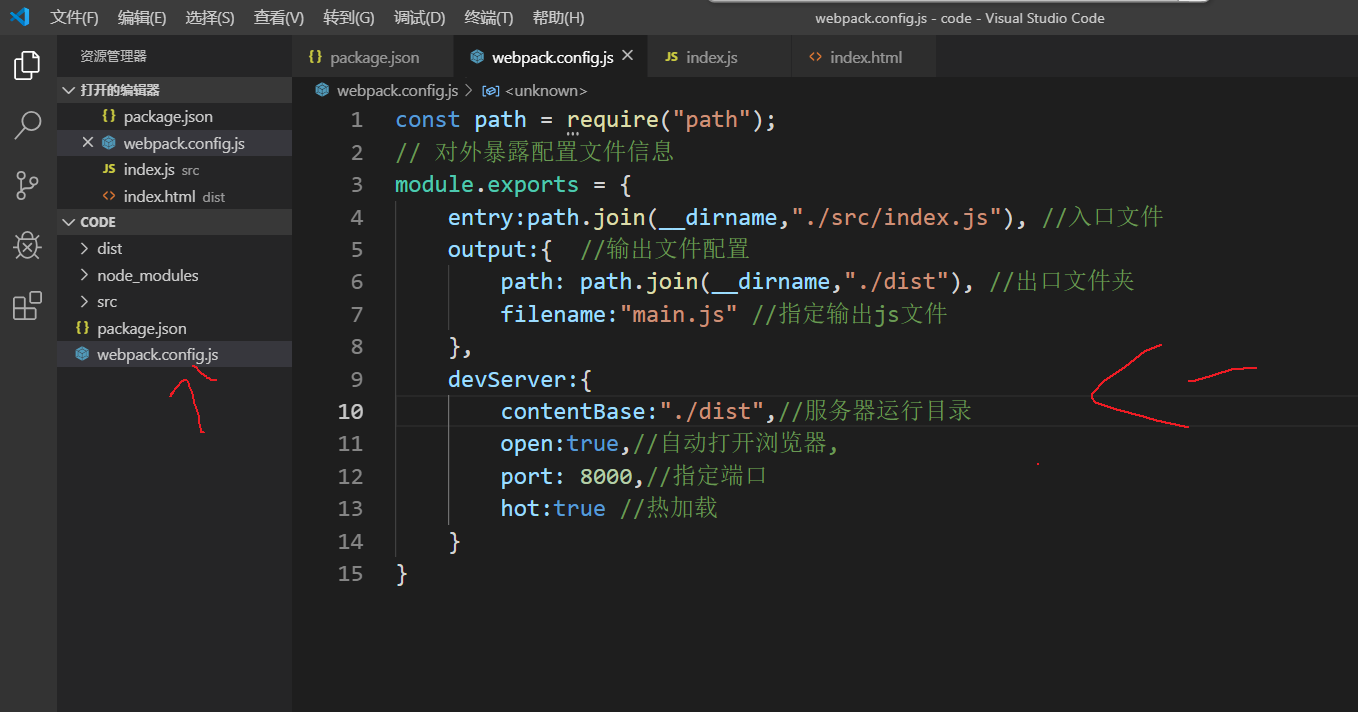
# 配置devServer

配置devserver包

$cnpm i webpack-dev-server -D



# 手动配置webpack.config.js文件



代码：

const path = require("path");

// 对外暴露配置文件信息

module.exports = {

    entry:path.join(\_\_dirname,"./src/index.js"), //入口文件

    output:{  //输出文件配置

        path: path.join(\_\_dirname,"./dist"), //出口文件夹

        filename:"main.js" //指定输出js文件

    },

    devServer:{

        contentBase:"./dist",//服务器运行目录

        open:true,//自动打开浏览器,

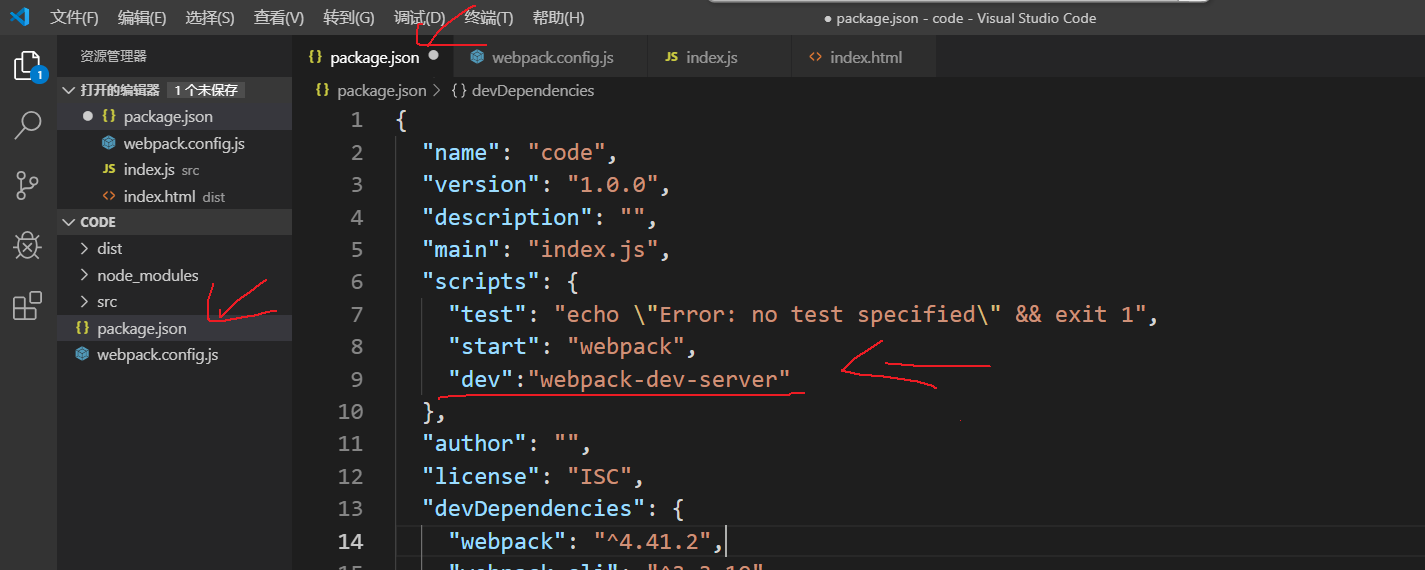
        port: 8000,//指定端口

        hot:true //热加载

    }

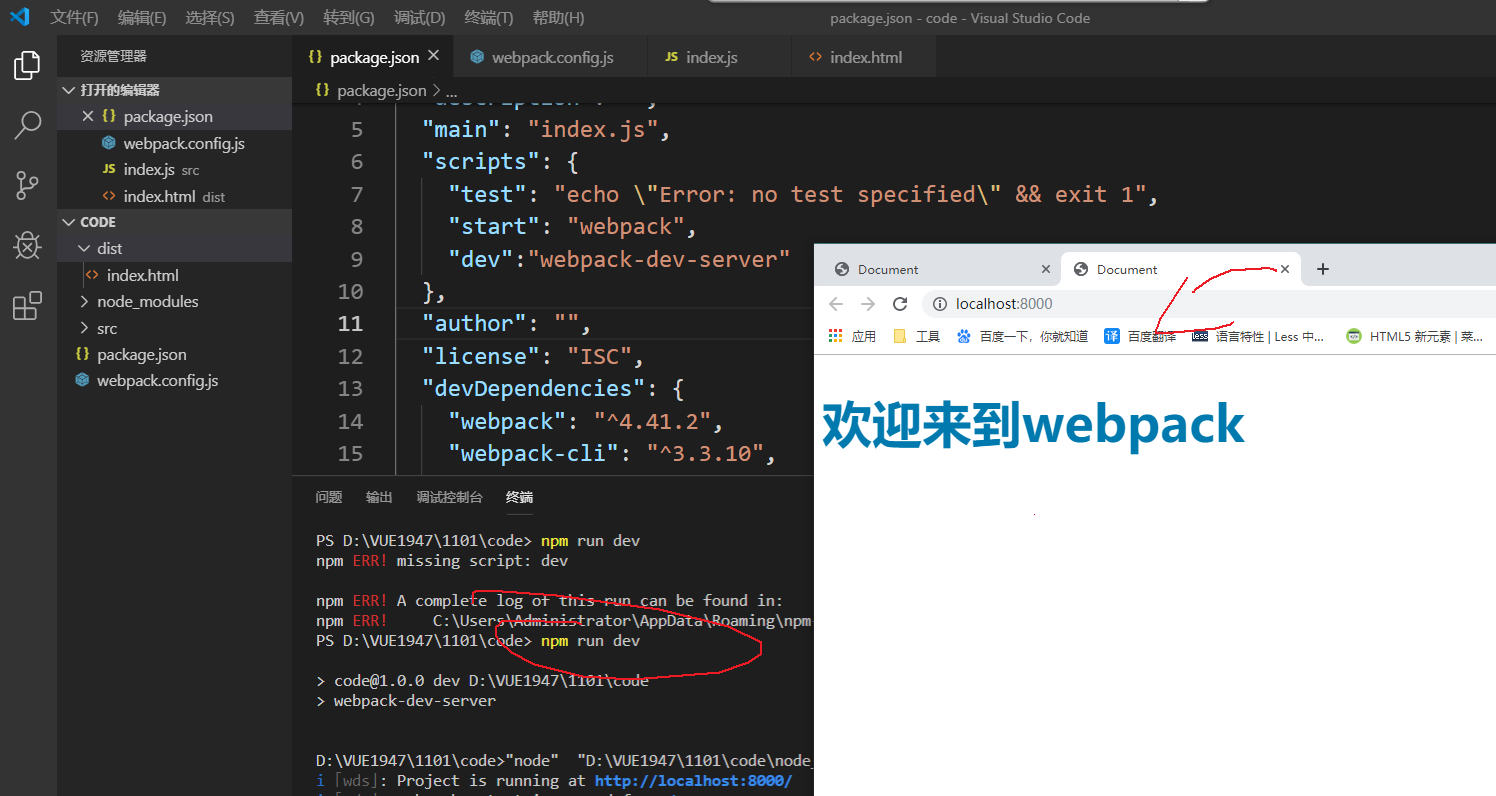
}

### 配置package.json文件devserver命令



测试：

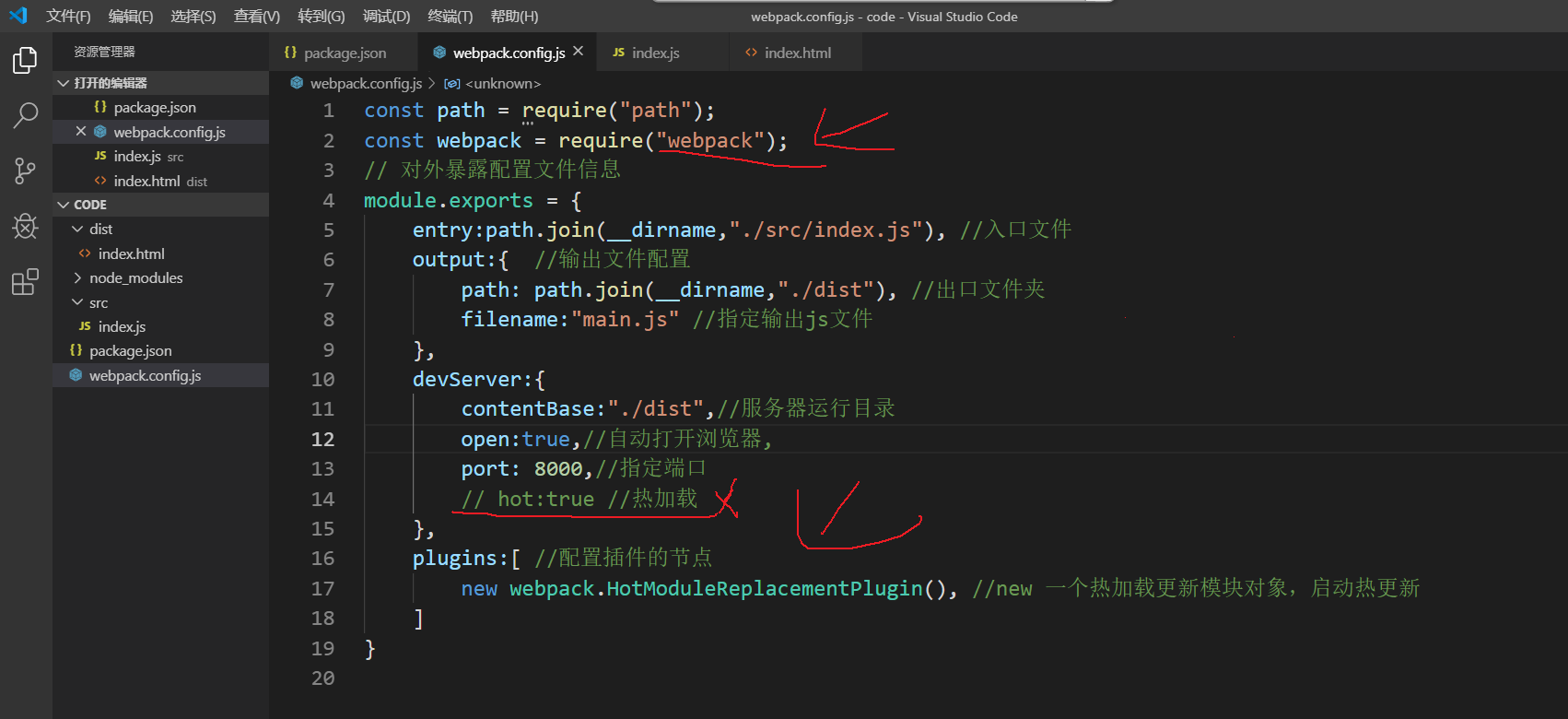
$npm run dev



# Webpack.config.js文件下插件模块

### 配置热加载模块

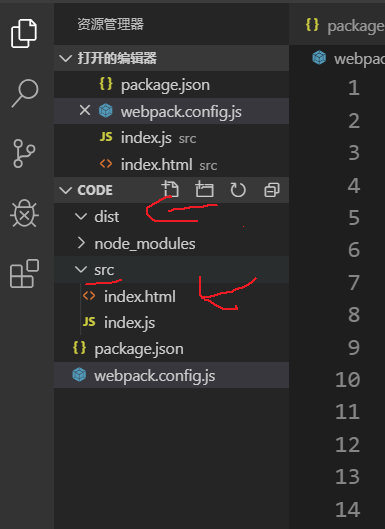
Webpack提供

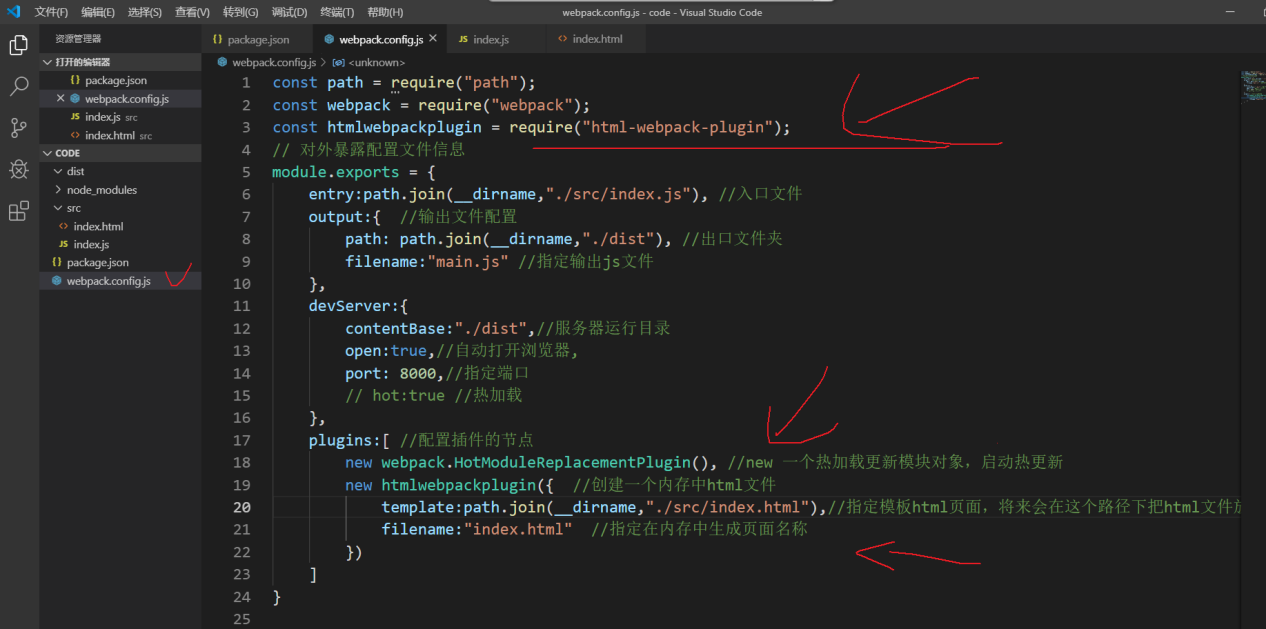


### Html静态文件加载模块

$cnpm i html-webpack-plugin -D

注意：把./dist下的index.html移动到./src/index.html 下面

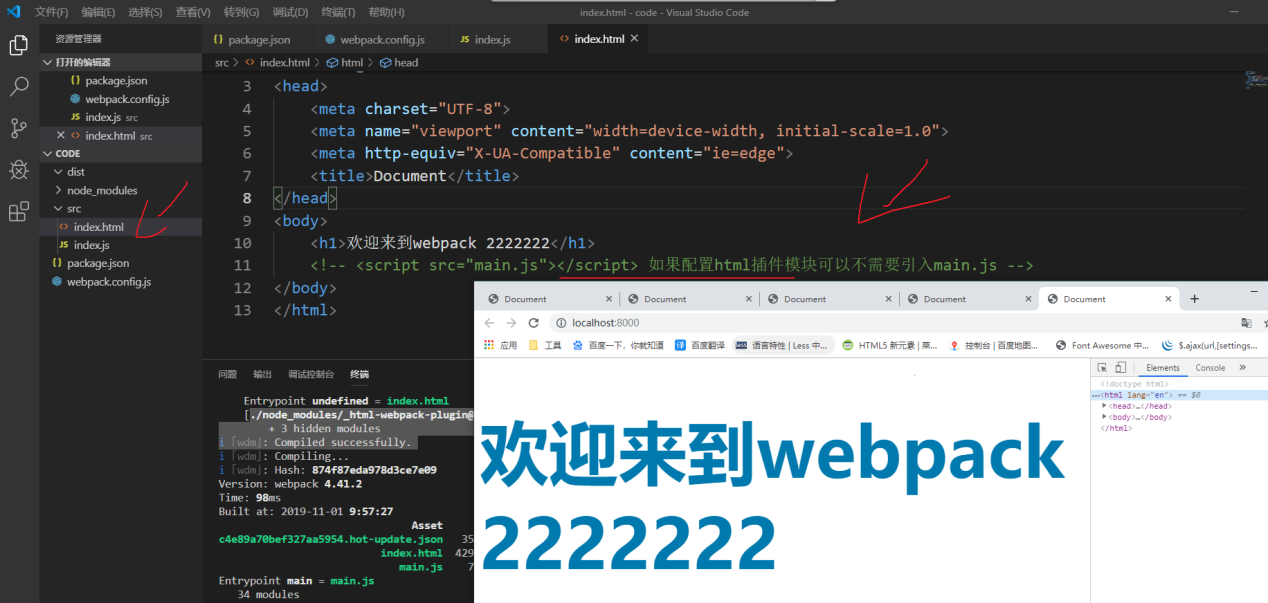




测试：

运行：$npm run dev

目的：在内存中可以运行index.html.在index.html里面可以不需要引入main.js文件，html插件模块自动引入。



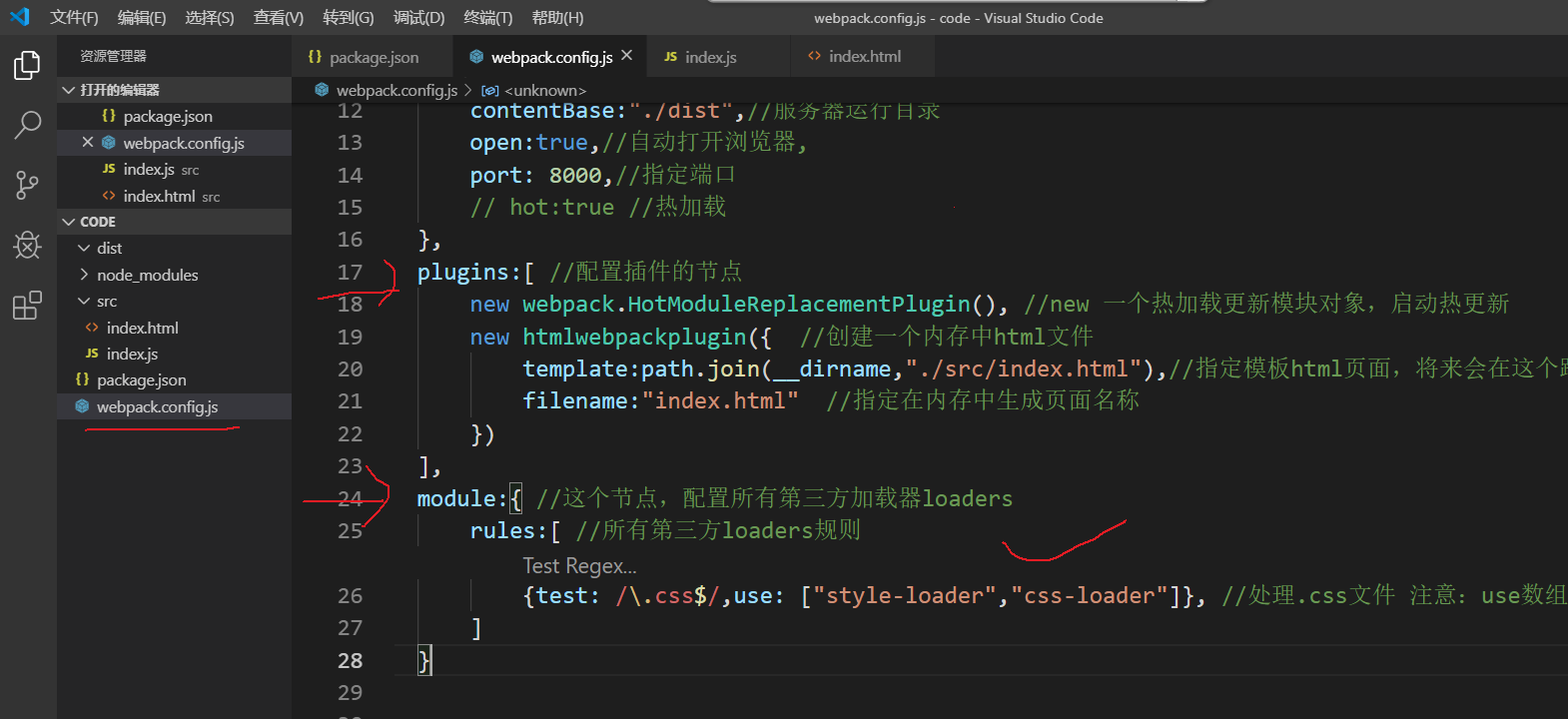
# 加载器 loader

## Css-loader css样式加载器

安装包：

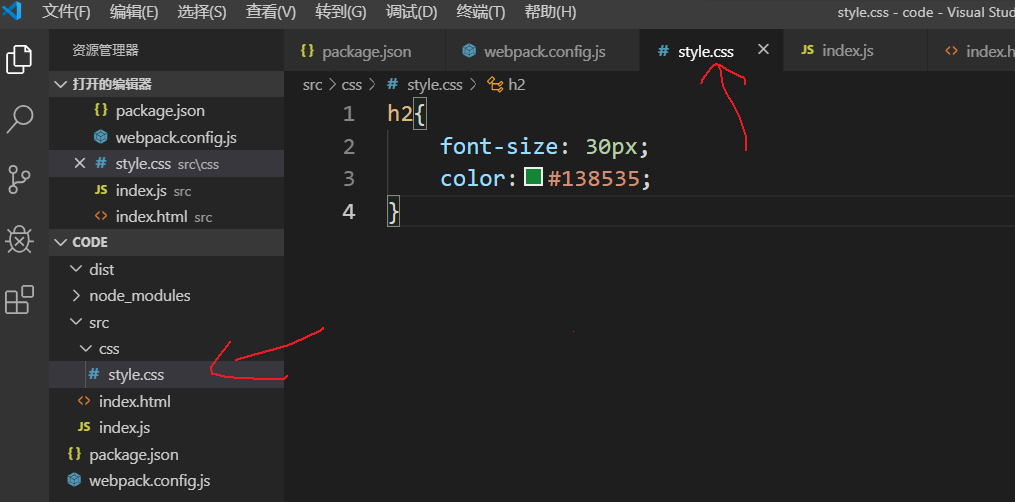
$cnpm i css-loader style-loader -S

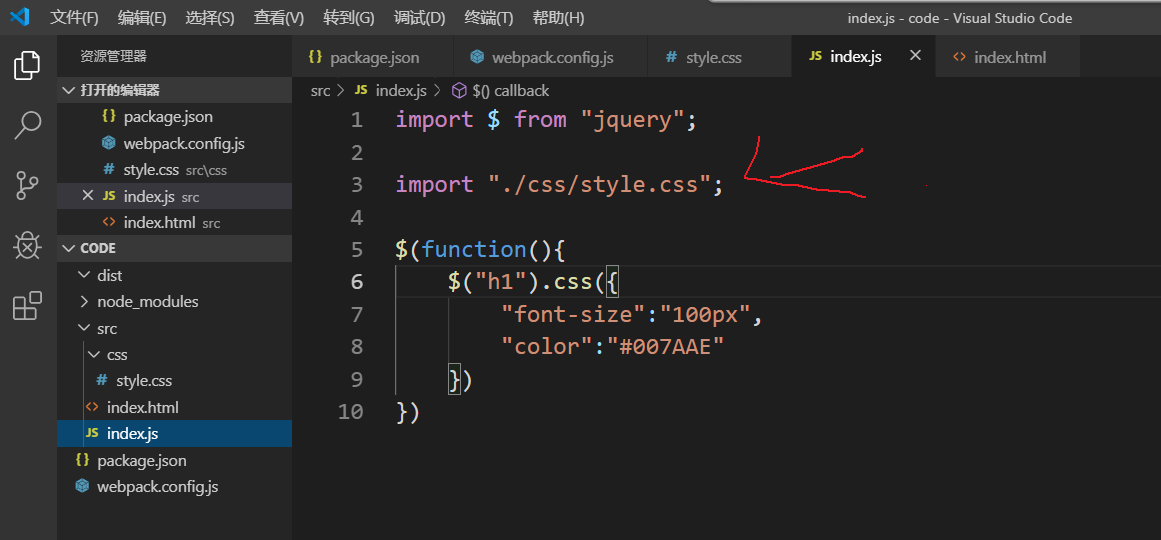
Webpack.config.js配置：

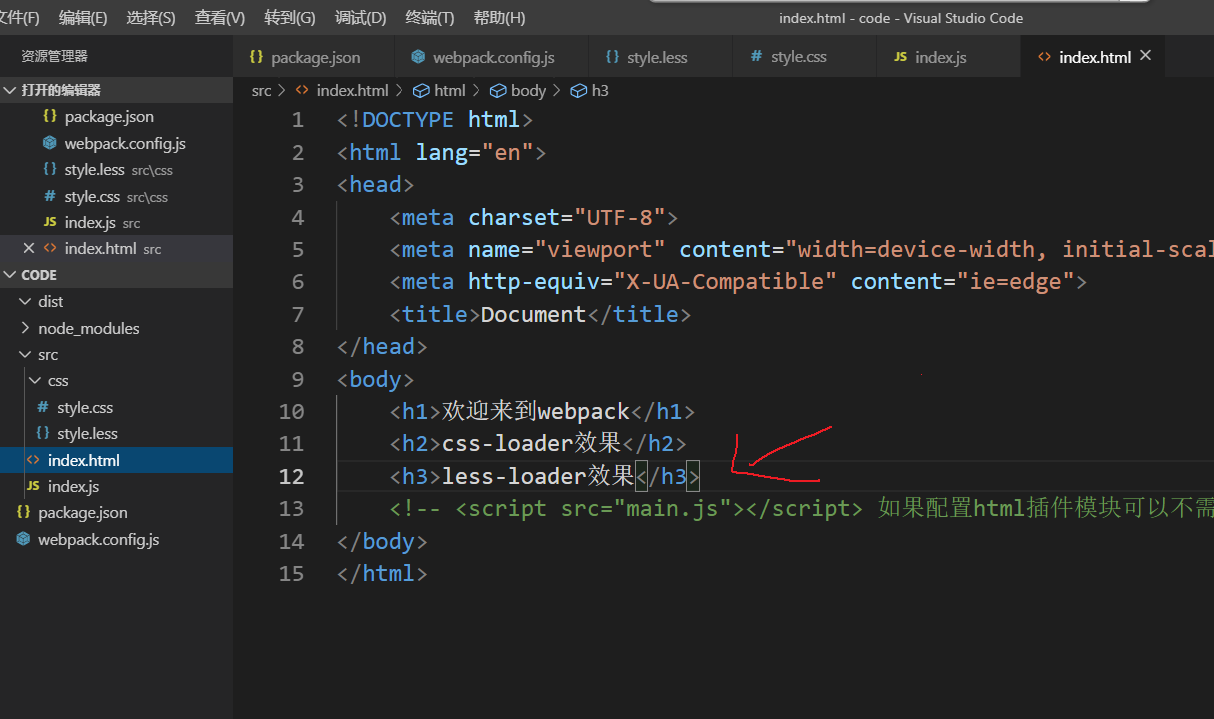


使用css样式：

创建./css/style.css

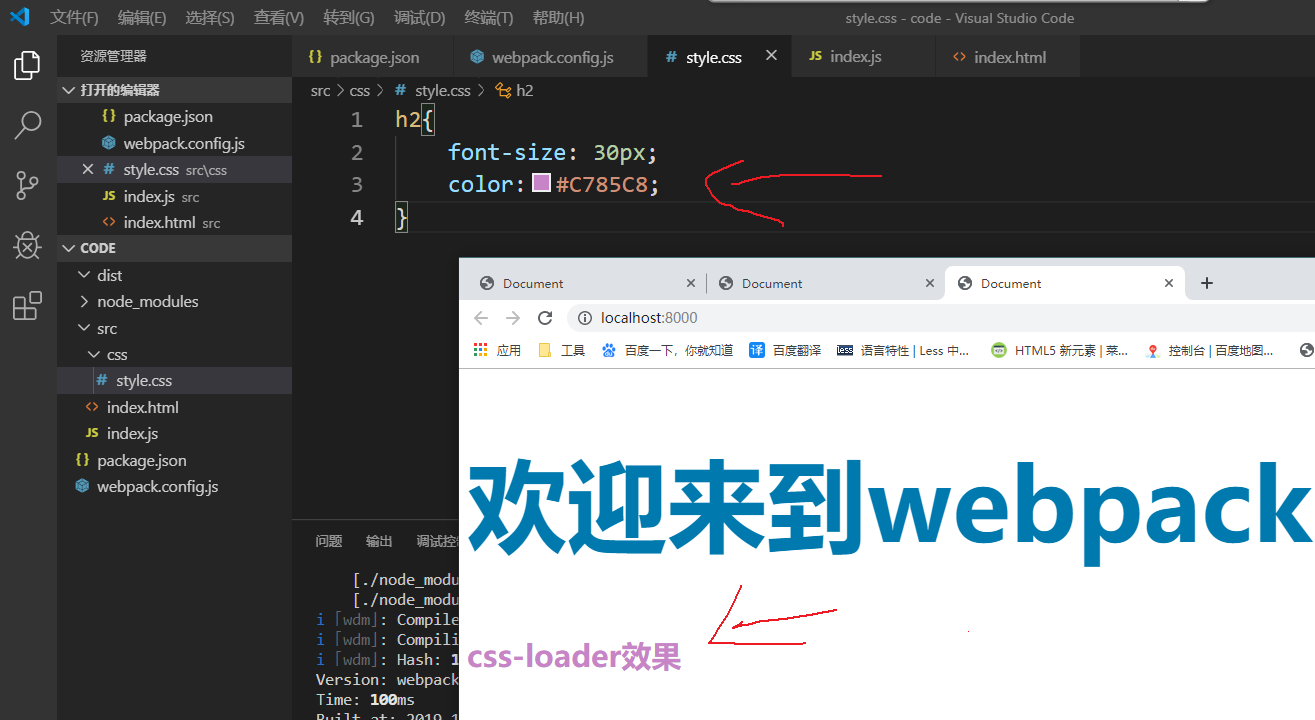






测试：

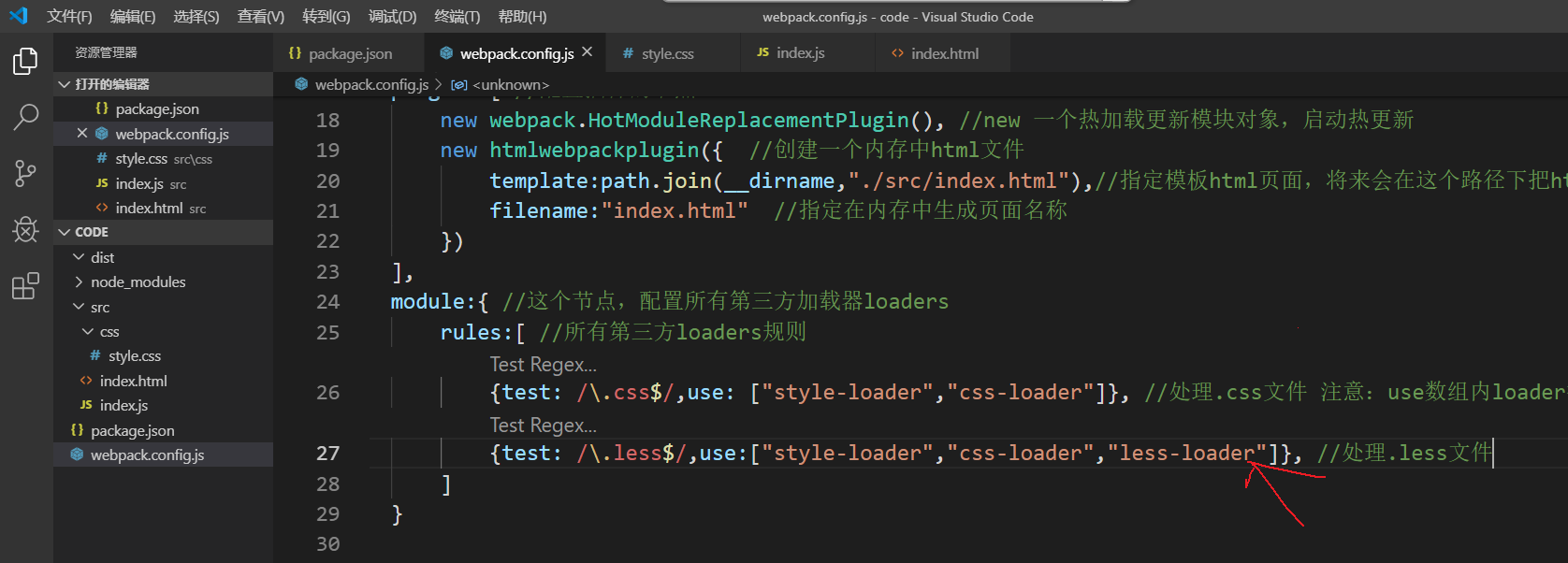
$npm run dev



## Less-loader less 加载器

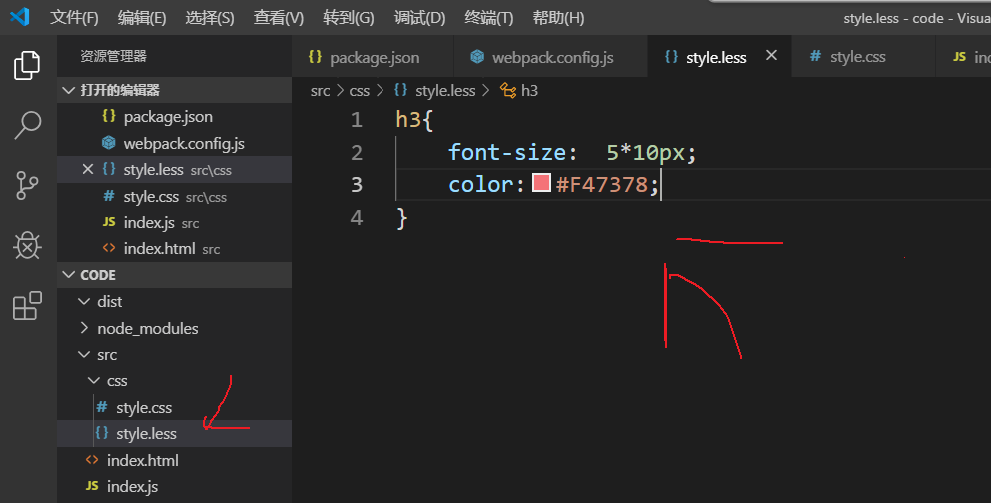
$cnpm i less-loader less -S

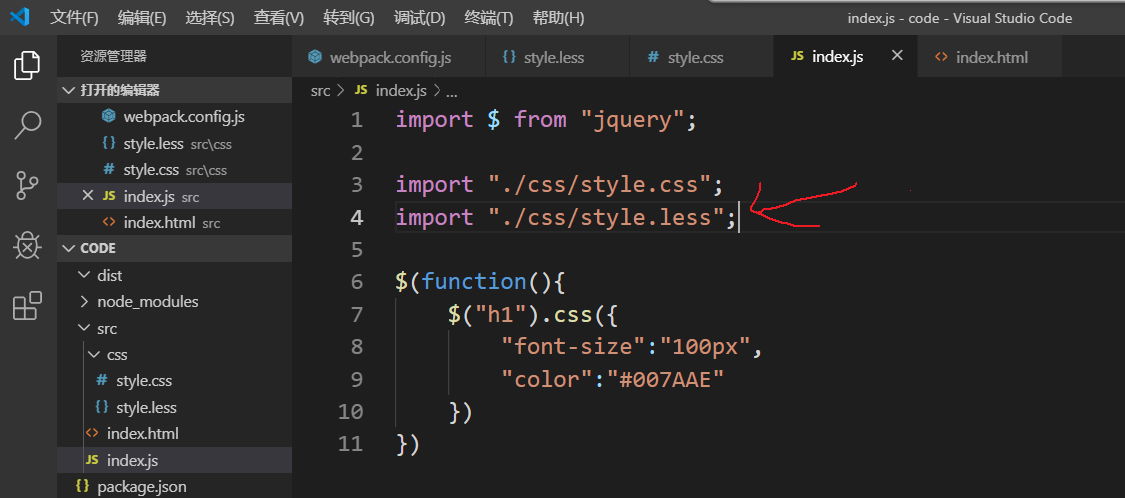
Webpack.config.js配置：

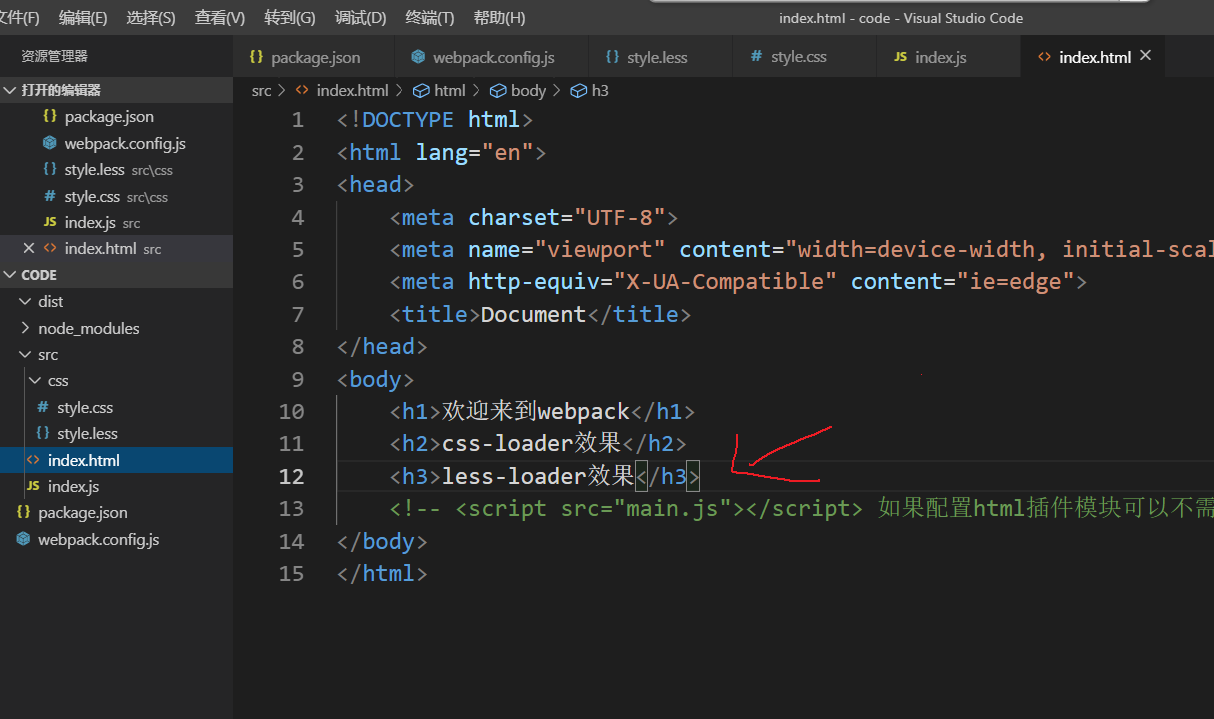


使用css样式：

创建./css/style.less







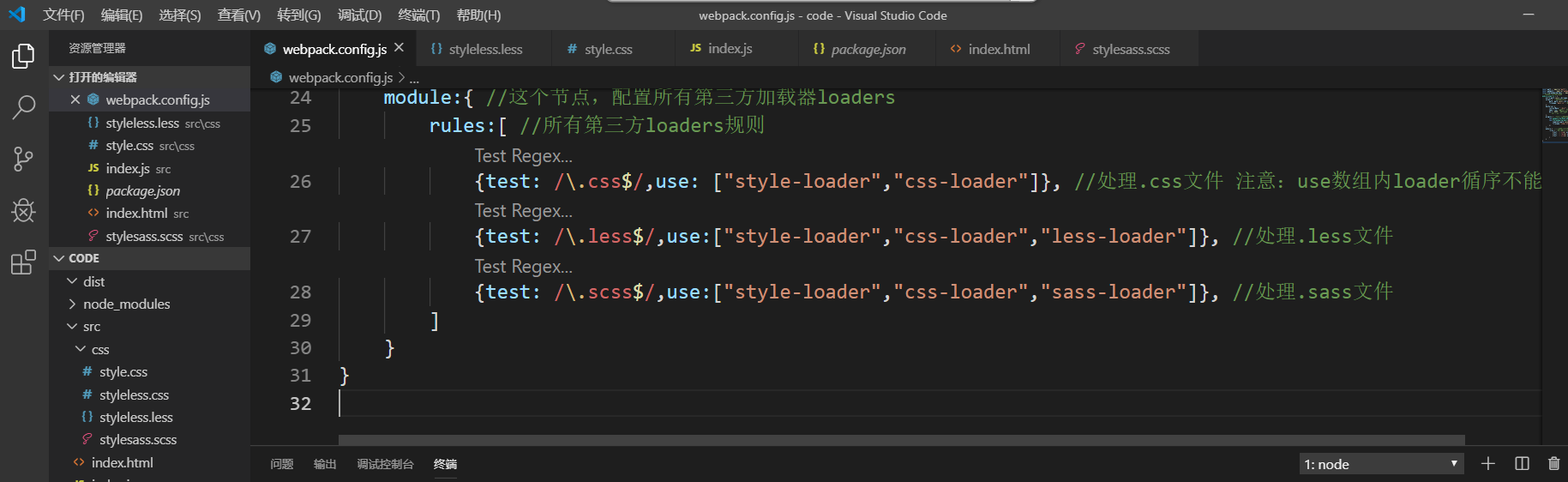
测试：

$npm run dev

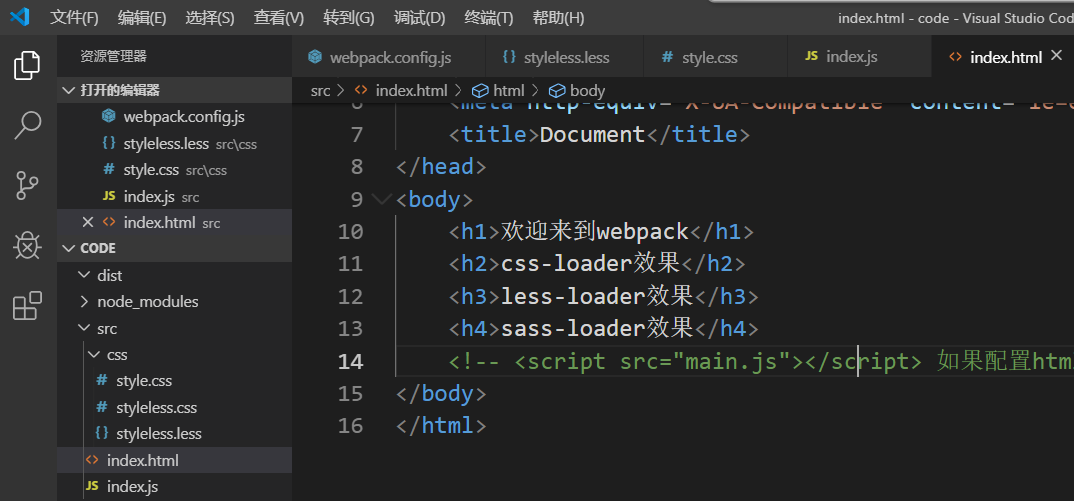
## Sass-loader sass 加载器

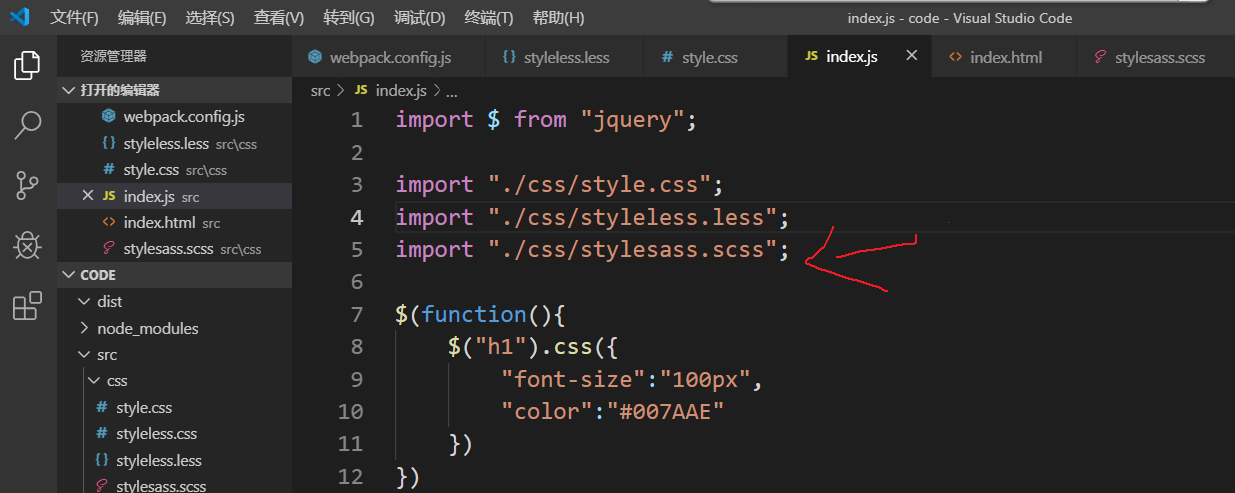
$cnpm i sass-loader node-sass -S

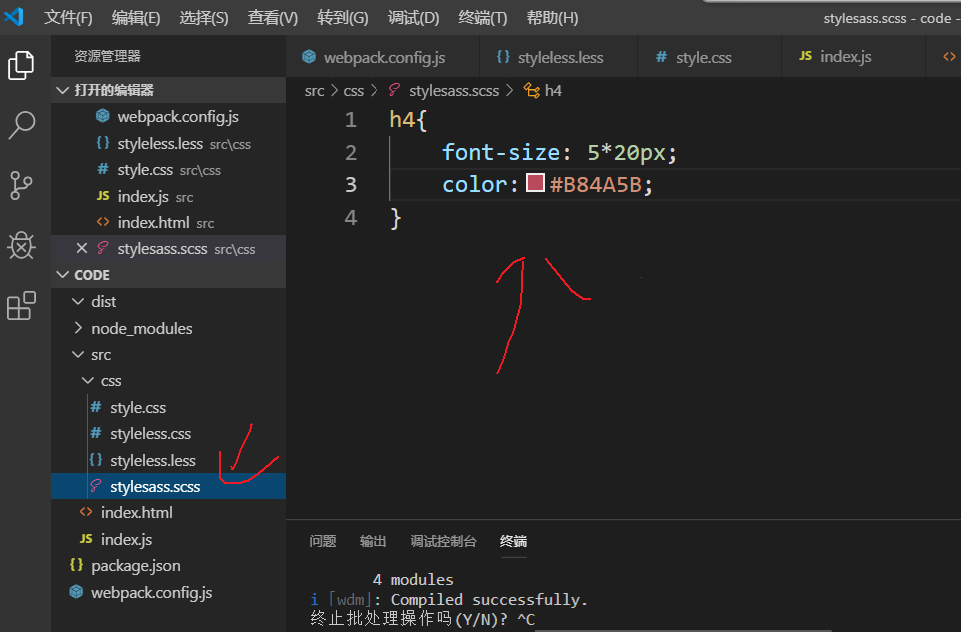
Webpack.config.js配置：



使用css样式：







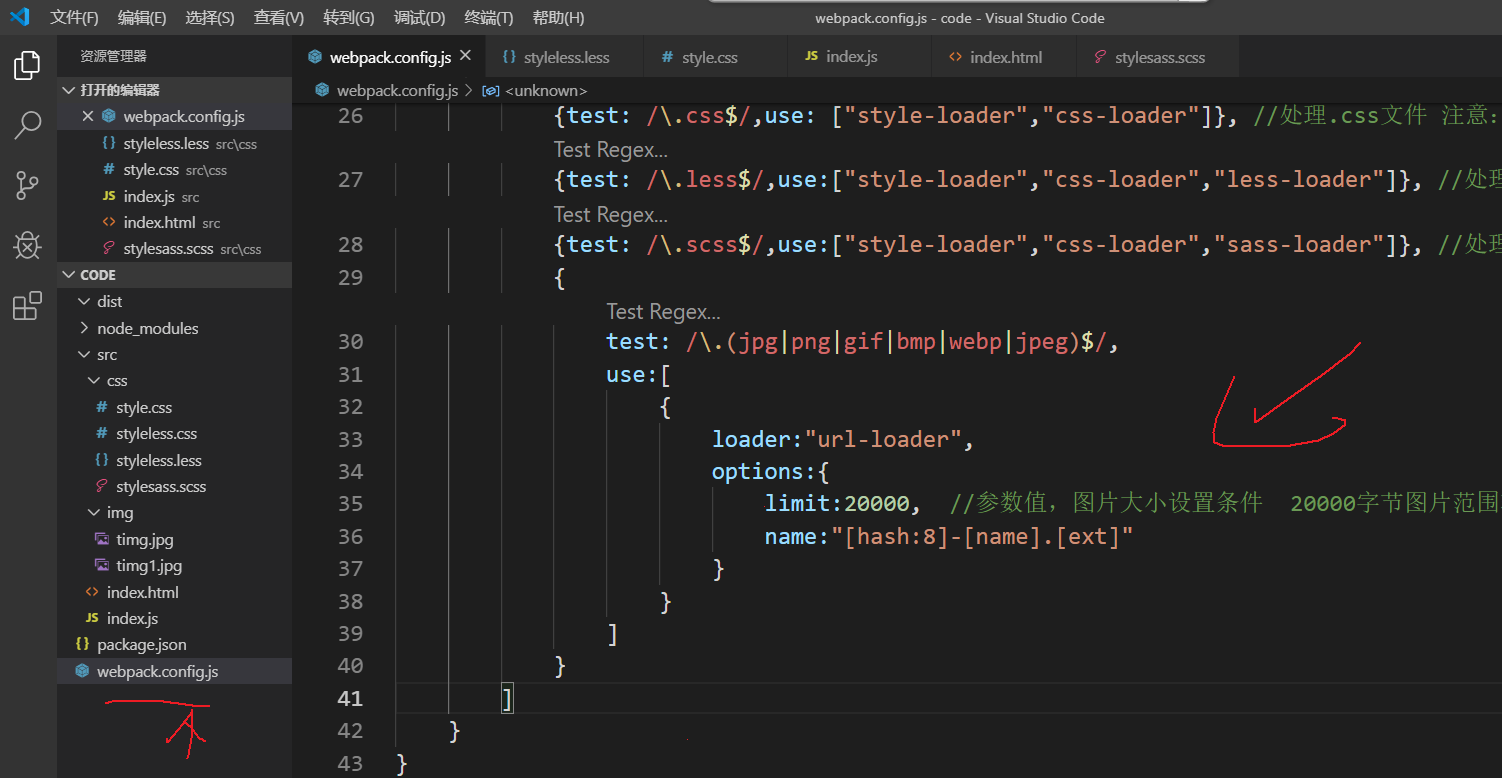
测试：

$npm run dev

## Url-loader file-loader url路径加载器

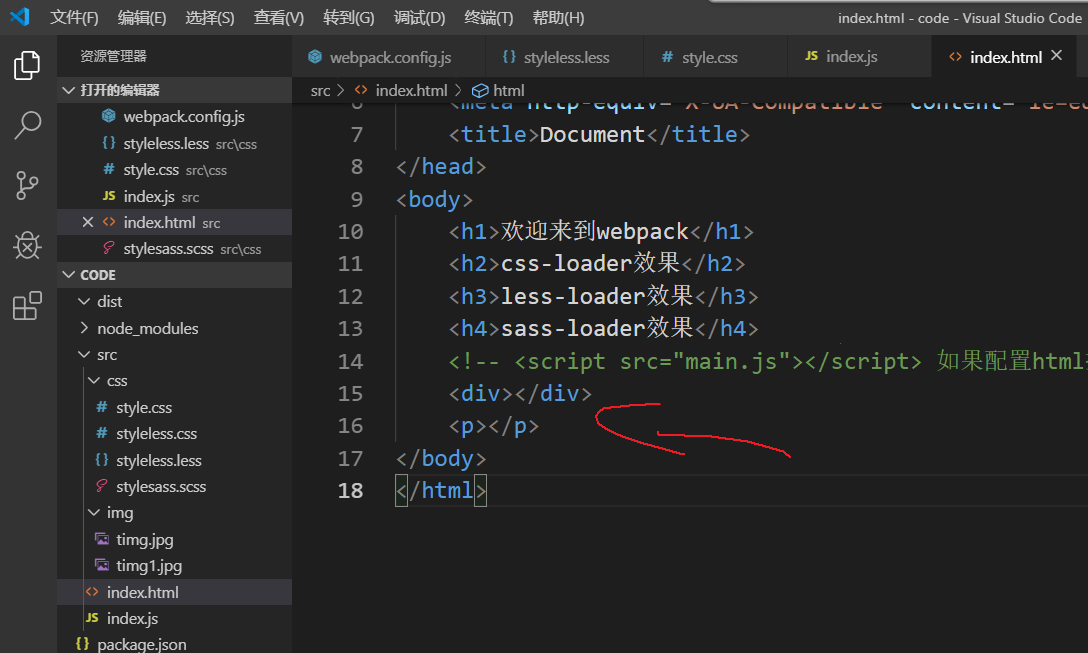
$cnpm i url-loader file-loader -S

Webpack.config.js配置文件



添加图片

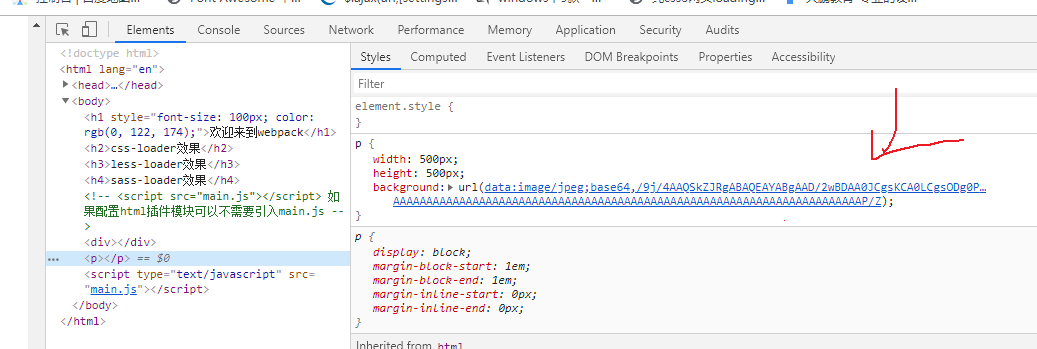
## 



测试结果

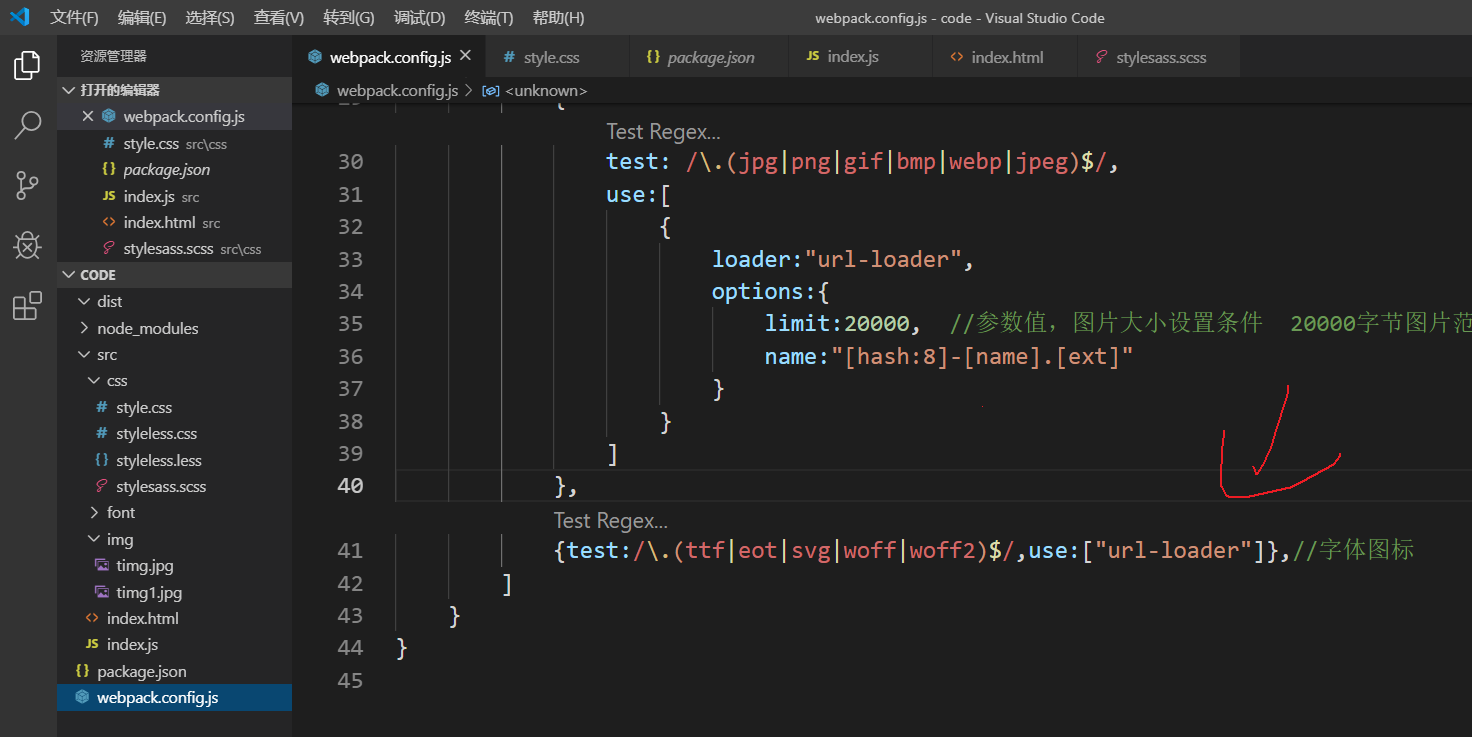
命令：$npm run dev





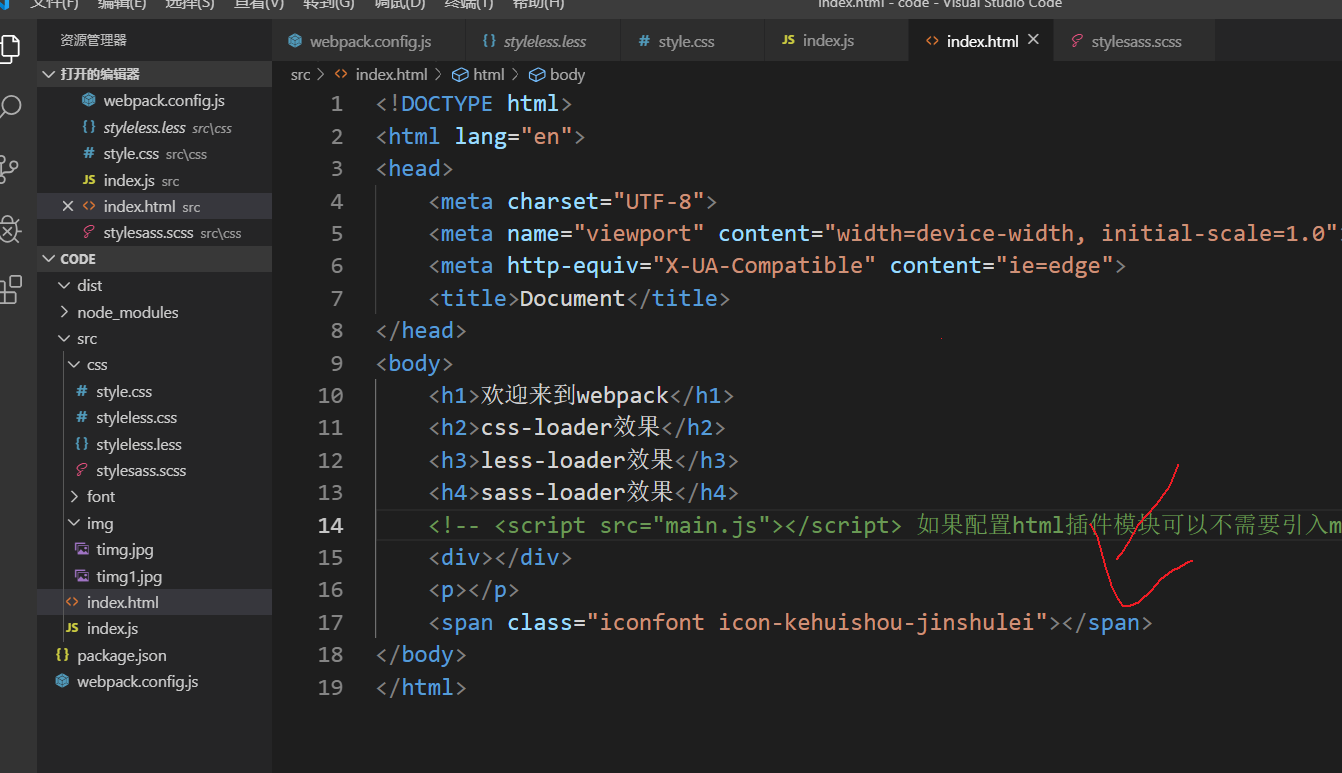
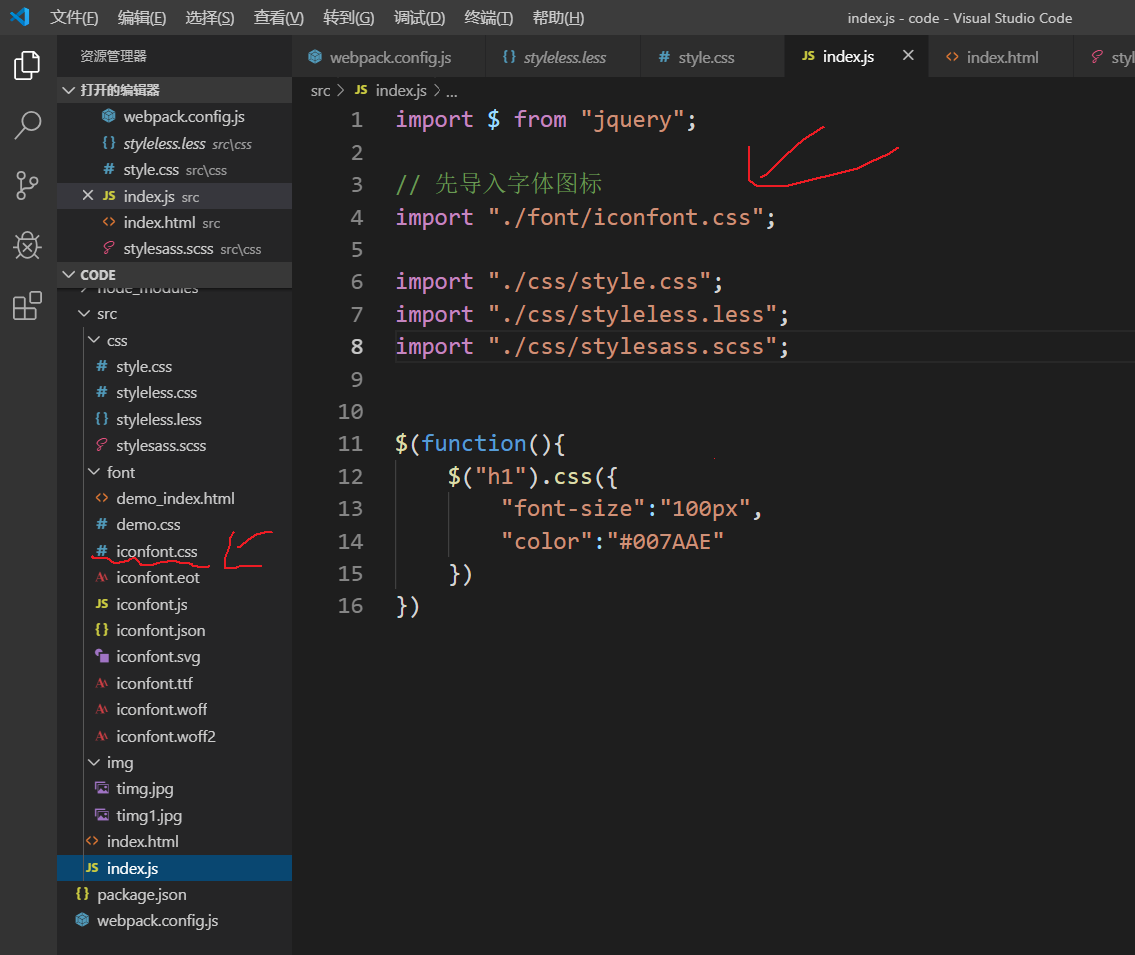
### 字体图标

Webpack.config.js配置文件



通过阿里矢量图下载文件包，放在./src下目录，通过index.js引入css文件包

<https://www.iconfont.cn/?spm=a313x.7781069.1998910419.d4d0a486a>



测试一下

$npm run dev



## Babel-loader js加载器

# **Babel 是一个 JavaScript 编译器。**

**今天就开始使用下一代的 JavaScript 语法编程吧！**

ES5->ES6->ES7

命令：

第一套安装包 环境

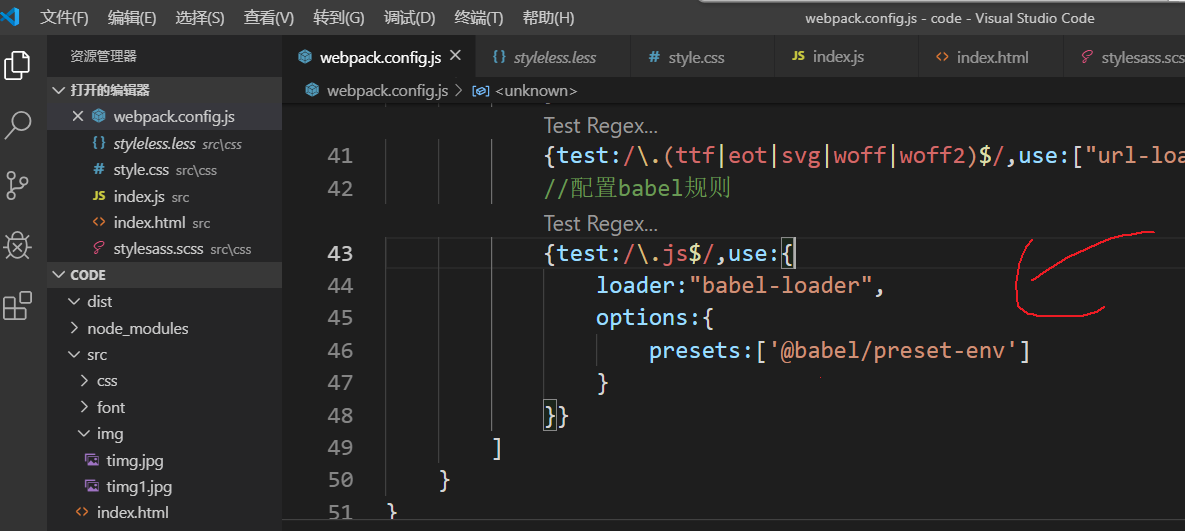
$cnpm install --save-dev @babel/core @babel/cli @babel/preset-env

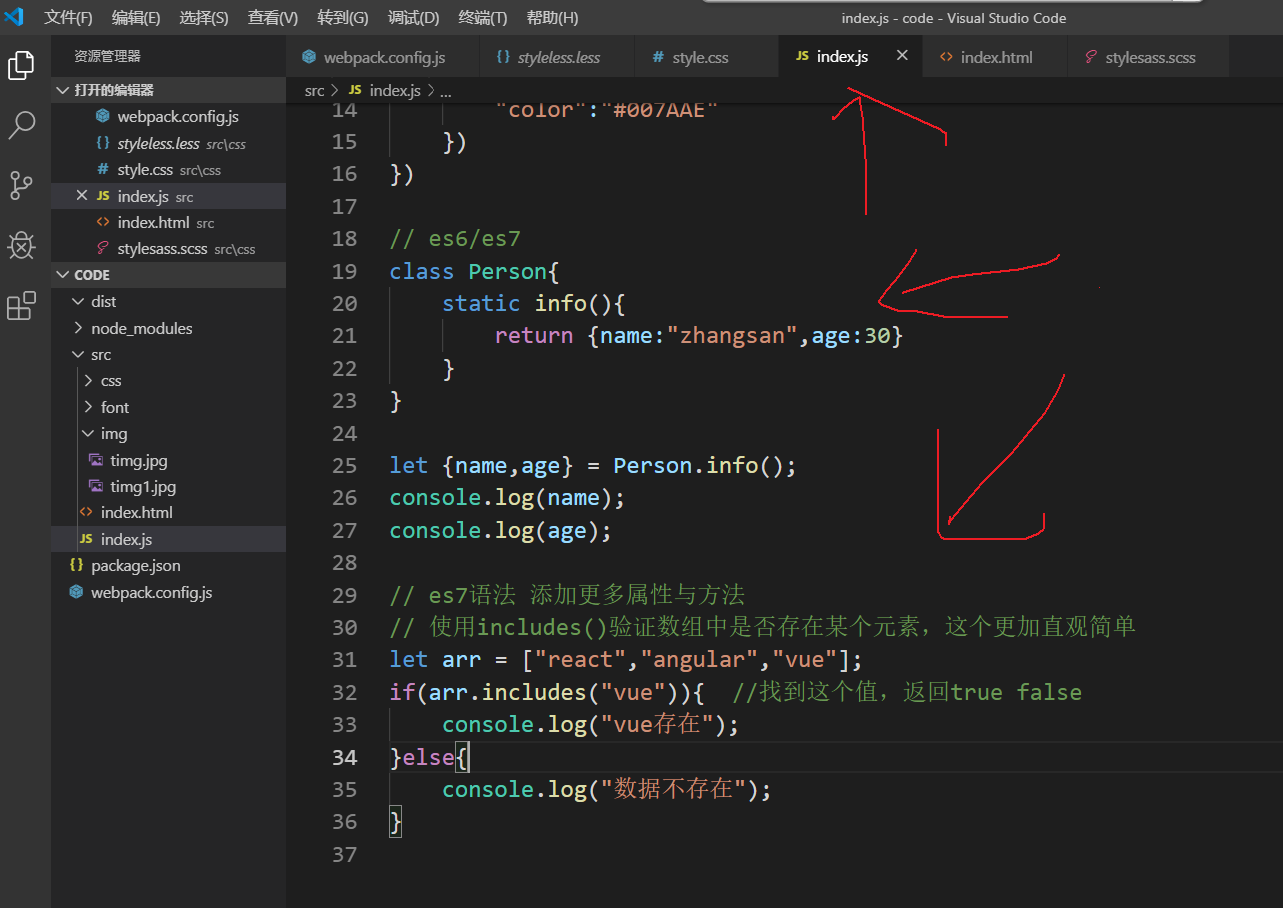
$cnpm install --save @babel/polyfill

第二套安装包 webpack

$cnpm i babel-loader babel-core babel-preset-env -S

配置webpack.config.js 文件插件





测试

$npm run dev

