### url-loader

安装url-loader和file-loader

Cnpm i url-loader file-loader -D

在配置文件中定义：

{

        test: /\.(jpg|png|gif)$/i,

        use: [

          {

            loader: "url-loader",

            options: {

              limit: 10000, *//图片大小大于8kb的时候会把图片直接输出在dist文件夹下，如果小于8kb的时候就输出为base64位编码*

              name: "images/[name].[ext]",

*//图片输出在dist文件夹下的路径*

            },

          },

        ],

      },

      {

        test: /\.(ttf|eot|svg|woff|woff2)$/,

        use: "url-loader",

*//对字体图标文件进行转化*

      },

### babel-loader

在 webpack 中，默认只能处理一部分 ES6 的新语法，一些更高级的ES6语法，或者ES7语法在 webpack 是处理不了的，需要通过第三方的loader 来帮助webpack 处理这些高级的语法

安装：

1. 第一套包：cnpm i @babel/core babel-loader @babel/plugin-transform-runtime -D

2.第二套包：cnpm i @babel/preset-env @babel/plugin-proposal-class-properties @babel/runtime -D

两套包都需要安装

解释:

@babel/core是核心

@babel/preset-env（处理es6+规范语法的插件集合）

@babel/runtime babel中的辅助函数的集合

@babel/plugin-transform-runtime 采用导入的形式使用辅助函数

@babel/plugin-proposal-class-properties 用来编译类

配置：

在webpack.config.js中配置：

{ test: /\.js$/, use: 'babel-loader', exclude: /node\_modules/}

用url-loader加载除了/node\_modules/里面文件之外其他的js文件

还需要定义一个.babelrc:

内容如下

{

"presets": ["@babel/preset-env"],

"plugins": [

"@babel/plugin-transform-runtime",

"@babel/plugin-proposal-class-properties"

]

}

进行完以上配置后，就可以在打包的时候把es6的代码转化为es5的代码了