### webpack配置文件

Webpack第三种进行打包的方式：使用配置文件

配置文件的名字必须叫做：webpack.config.js

在配置文件中定义入口文件和出口文件以及打包方式mode

**const** path=require("path");

module.exports={

    mode:"development",

    entry:{

        index:"./src/index.js"

    },

    output:{

        filename:"[name].js",

        path:path.resolve(\_\_dirname,"dist")

    }

}

### webpack热更新

安装webpack-dev-serve

Cnpm i webapck-dev-server -D

devServer: {

*//   定义热更新*

    contentBase: path.join(\_\_dirname, "dist"), *//定义热更新服务器文件夹*

    hot: true, *//是否启用热更新*

    host: "localhost", *//主机号*

    port: "8080", *//端口号*

    compress: true, *//是否对所有资源进行压缩*

    progress: true, *//是否显示进度*

    open: true, *//完成后是否自动打开浏览器*

  },

### Html-plugin

安装： cnpm i html-webpack-plugin -D

这个插件的作用就是以index.html为模板在dist文件夹底下生成一个html 并会自动引入dist下的所有js文件。

plugins: [

    new HtmlWebpackPlugin({

*//   实例化插件html-webpack-plugin*

      template: "./index.html",

*//   定义模板*

      filename: "index.html",

*//   定义输出的文件名字*

      minify: {

        removeAttributeQuotes: true, *//是否去除双引号*

        collapseWhitespace: true, *//是否变成一行*

      },

*//   对html文件进行打包*

      hash: true,

*//路径后是否需要添加hash*

    }),

  ],

**不要忘记在头部引入htmlplugin**

**const** HtmlWebpackPlugin = require("html-webpack-plugin");

### 处理css和scss文件

cnpm i css-loader style-loader -D

cnpm i sass-loader node-sass -D

 module: {*//定义在webpack里使用的加载器loaders*

    rules: [*//定义转化文件的规则，添加模块的配置规则,对文件的扩展名进行过滤*

      {

        test: /\.scss$/,

        use: [

          "style-loader", *//将js字符串生成style节点，*

          "css-loader", *//将css文件转化为commonjs的语法*

          "sass-loader", *//将scss语法转化为css语法*

        ],

      },

    ],

  },