# 一、基本概念

## 1.定义

**webpack** 是一个用于现代 JavaScript 应用程序的 *静态模块打包工具*。当 webpack 处理应用程序时，它会在内部构建一个 [依赖图(dependency graph)](https://webpack.docschina.org/concepts/dependency-graph/)，此依赖图对应映射到项目所需的每个模块，并生成一个或多个 *bundle*。

**（将复杂的项目结构最小化）**

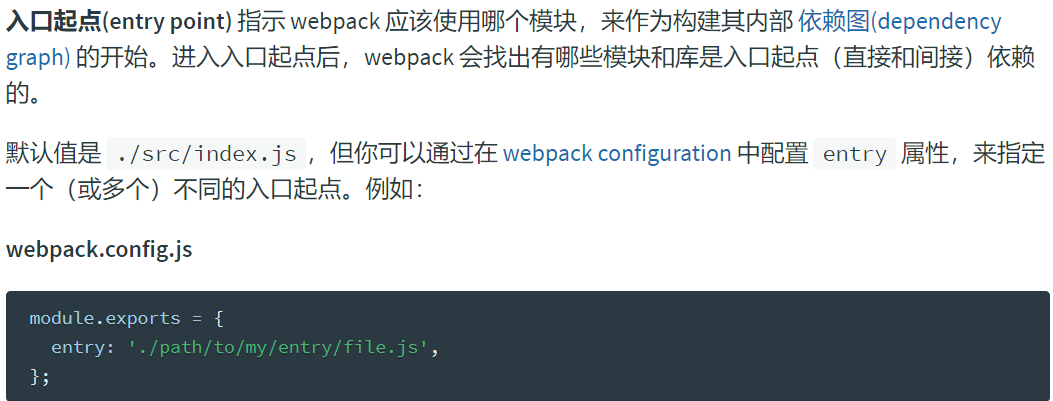
## 2.模块

打包对象：



# 二、核心概念

## 1.入口(entry)



## 2.输出(output)



## 3.loader



## 4. 插件(plugin)

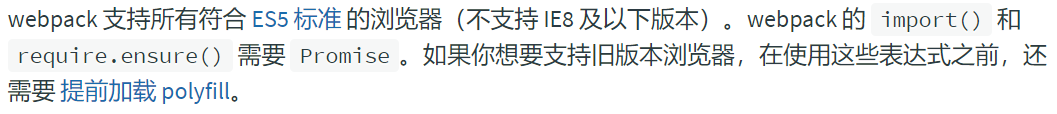


## 5.模式(Mode)





## 6.浏览器兼容性(browser compatibility)



# 三、使用

创建配置文件webpack.config.js中设置五大核心概念

## 1.打包js、json

webpack src/js/index.js -o build/js/built.js --mode=development

能够编译打包 js 和 json 文件，并且能将 es6 的模块化语法转换成

浏览器能识别的语法。

## 2.打包css

下载loader：npm i css-loader style-loader less-loader less -D

module: { //module中设置

rules：[

{

test: /\.css$/,

use: [

// use 数组中 loader 执行顺序：从右到左，从下到上 依次执行

'style-loader', //创建 style 标签，将 js 中的样式资源插入进行，添加到 head 中生效

'css-loader' // 将 css 文件变成 commonjs 模块加载 js 中，里面内容是样式字符串

]

}，

{

test: /\.less$/,

use: [

'style-loader',

'css-loader',

'less-loader'

]

}

]

}

## 3.打包HTML

下载plugin：npm install --save-dev html-webpack-plugin

const HtmlWebpackPlugin = require('html-webpack-plugin');

plugins: [

// 功能：默认会创建一个空的 HTML，自动引入打包输出的所有资源（JS/CSS）

// 需求：需要有结构的 HTML 文件

new HtmlWebpackPlugin({

// 复制 './src/index.html' 文件，并自动引入打包输出的所有资源（JS/CSS）

template: './src/index.html'

})

],

## 4.打包图片资源

下载：npm install --save-dev html-loader url-loader file-loader

module: { //module中设置

rules：[

{

。。。

}，

{

test: /\.(jpg | png | gif)$/, //js中的

loader: 'url-loader',

options: {

// 图片大小小于 8kb，就会被 base64 处理

// 优点: 减少请求数量（减轻服务器压力）

// 缺点：图片体积会更大（文件请求速度更慢）

**limit: 8 \* 1024,**

// 问题：因为 url-loader 默认使用 es6 模块化解析，而 html-loader 引入图片是 commonjs

// 解析时会出问题：[object Module]

// 解决：关闭 url-loader 的 es6 模块化，使用 commonjs 解析

**esModule: false,**

// 给图片进行重命名

// [hash:10]取图片的 hash 的前 10 位

// [ext]取文件原来扩展名

**name: '[hash:10].[ext]**

}

}，

{

test: /\.html$/,

// 处理 html 文件的 img 图片（负责引入 img，从而能被 url-loader 进行处理）

loader: 'html-loader'

}

]

}

## 5.打包其他资源

module:

{

rules: [

{

test: /\.css$/,

use: ['style-loader', 'css-loader']

},

// 打包其他资源(除了 html/js/css 资源以外的资源)

{

// 排除 css/js/html 资源

exclude: /\.(css|js|html|less)$/,

loader: 'file-loader',

options: {

name: '[hash:10].[ext]'

}

}

]

},