1.webpack的核心概念

Entry：入口，Webpack进行打包的起始点(文件)
Output：出口，webpack编译打包生成的bundle(打包文件)
Loader：模块加载(转换)器，将非js、非json模块包装成webpack能理解的js模块
Plugin：插件，在 Webpack 构建流程中的特定时机插入具有特定功能的代码
Module：模块，在 Webpack眼里一切皆模块，默认只识别js文件，如果是其它类型文件利用对应的loader转换为js模块

2.webpack配置文件的整体结构

```
module.exports = {
  entry: '', //入口
  output: {},//输出
  module: {rules: []},//配置loader
  plugins: [] //配置plugin
}
```

3.webpack模块化打包的基本流程

1. 连接：webpack从入口JS开始，递归查找出所有相关联的模块，并【连接】起来形成一个图(网)的结构
2. 编译：将JS模块中的模块化语法【编译】为浏览器可以直接运行的模块语法(当然其它类型资源也会处理)
3. 合并：将图中所有编译过的模块【合并】成一个或少量的几个bundle文件，浏览器真正运行是打包生成的bundle文

4.比较loader与plugin

1). loader：用于加载特定类型的资源文件，webpack本身只能打包js。
2). plugin：用来扩展webpack其它方面的功能，一般loader处理不了的资源、完成不了的操作交给插件处理。

5.区别live-reload（自动刷新）与hot-realod/HMR（热模替换）

相同点：
　　代码修改后都会自动重新编译打包
不同点：
　　live-reload：刷新整体页面，从而查看到最新代码的效果，页面状态全部都是新的。
　　Hot-reload：没有刷新整个页面，只是加载了修改模块的打包文件并运行，
　　从而更新页面的局部界面，整个界面的其它部分的状态还在

6.webpack常用loader与plugin汇总

loader:

1.【less-loader】:用于将less文件翻译成为css

2.【css-loader】：用于将css以CommonJs语法打包到js中

3.【style-loader】：用于动态创建一个style标签，将css引入页面

备注：上述三个loader一般配合使用，最终实现：翻译less为css，以style标签形式将样

4.【file-loader】：提取源代码图片资源，到指定位置，可修改文件名等操作。

5.【url-loader】：与file-loader功能几乎一致，优势是可以对图片进行动态转换base64编码（控制limi

备注：上述两个loader中url-loader应用比file-loader广泛。

6.【jshint-loader】：对项目中的js语法进行检查，可选的配置项有：

emitErrors：true/false

-- emitErrors为true，检查出的错误显示为 error（错误）类信息。

-- emitErrors为false，检查出的错误显示为 warning（错误）类信息。

failOnHint: true/false,

-- failOnHint为true，当jshint检查出错误时，直接打断当前的代码的编译。

-- failOnHint为false，当jshint检查出错误时，会继续编译。

esversion: 6

--告诉jshint，不再提示新语法兼容性问题（有专门的loader解决新语法问题）

reporter: function(errors) {}

--自定义一个报告错误的函数，输出想要的内容

7.【babel-loader】：将es6语法转换为es5语法

备注：该loader的使用要借助：babel-loader babel-core babel-preset-es2015

8.【postcss-loader】：用于扩展css前缀

备注：

(1).该loader需要一个postcss.config.js配置文件。

(2).该loader要配合autoprefixer库使用。

(3).该loader使用的时机为：["css-loader","postcss-loader","less-lo

pulgin:

1.【extract-text-webpack-plugin】：用于提取项目中的css,最终合并为一个单独的文件。

备注：上述插件需要配合：css-loader、less-loader两个loader使用，css-loader、

2.【html-webpack-plugin】：自动创建html文件，且自动引入外部资源。配置项如下：

```
 title:"webpack",
filename:"index.html",
template:"./src/index.html"
//用于压缩html
 minify:{
removeComments:true, //移除注释
collapseWhitespace:true}//移除换行
```

3.【clean-webpack-plugin】：清空webpack的输出目录，防止其他文件“乱入”。

4.【HotModuleReplacementPlugin】：热模替换（HMR）插件

备注：1.该模块必须配合webpack-dev-server模块使用，且webpack-dev-server中必须

2.想要让指定文件支持HMR，必须要：

(1).无论是否有插件操作过该类型的资源，最终必须交给lo

(2).必须在入口文件中声明使用。

5.【UglifyJsPlugin】：压缩js的插件，且可以生成sourceMap映射文件，用于方便排查错误。

6.【less-plugin-clean-css】：压缩css文件，在less-loader翻译less文件之后，该插件介入，开始压缩