# **模块(modules)**

在[模块化编程](https://en.wikipedia.org/wiki/Modular_programming)中，开发者将程序分解成离散功能块(discrete chunks of functionality)，并称之为*模块（一般一个js文件定义一个ES6类，一个文件即一个模块）*。

## **什么是 webpack 模块**

对比 [Node.js 模块](https://nodejs.org/api/modules.html)，webpack *模块*能够以各种方式表达它们的依赖关系，几个例子如下：

* [ES2015 import](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/import) 语句
* [CommonJS](http://www.commonjs.org/specs/modules/1.0/) require() 语句
* [AMD](https://github.com/amdjs/amdjs-api/blob/master/AMD.md) define 和 require 语句
* css/sass/less 文件中的 [@import 语句](https://developer.mozilla.org/en-US/docs/Web/CSS/@import)。
* 样式(url(...))或 HTML 文件(<img src=...>)中的图片链接(image url)

# **模块解析(module resolution)**

resolver 是一个库(library)，用于帮助找到模块的绝对路径。一个模块可以作为另一个模块的依赖模块，然后被后者引用，如下：

import foo from 'path/to/module'

// 或者

require('path/to/module')

## **webpack 中的解析规则**

webpack 能够解析三种文件路径：

### **绝对路径**

import "/home/me/file";

import "C:\\Users\\me\\file";

由于我们已经取得文件的绝对路径，因此不需要进一步再做解析。

### **相对路径**

import "../src/file1";

import "./file2";

### **模块路径**

import "file1";

import "lib/file";

注：模块路径与其他路径不同之处，模块路径没有 / 在前面

模块路径将在 [resolve.modules](https://www.webpackjs.com/configuration/resolve/" \l "resolve-modules) 中指定的所有目录内搜索文件。

module.exports = {

resolve: {

extensions: ['.js', “.jsx”,"\*",".css"],

modules:[libPath, nodeModPath, cssPath, jsPath],

alias: require('./src/pathmap.json')

}

}

Extension：指定文件可能的后缀，如果import不指定后缀，如：

import ListBtn from 'ListBtn\index';

则会按照顺序添加后缀并查看是否有该文件，如果有则应用该文件，如下：

import ListBtn from 'ListBtn\index.js';

其中，index是可以省略的，你可以在文件中这样写

import ListBtn from 'ListBtn';

Modules：模块路径目录

Alias：用于指定别名