**CommonJS并不是ES6、7，他有一套自己的独立语法，CommonJS无法在浏览器环境中运行，更多的是在nodejs中运行，我们安装的大多数npm包都是CommonJS。**

**1 概述**

Nodejs的模块系统遵循了CommonJS规范。下面，我们结合Node来深入了解CommonJS规范。

**2 模块定义**

CommonJS规范规定，一个文件就是一个模块，我们可以在文件内部使用module变量，module变量是Module类的实例。

每个模块内部，都有一个module对象，代表当前模块。它的属性如下：

* module.id 模块的识别符，通常是带有绝对路径的模块文件名。
* module.filename 模块的文件名，带有绝对路径。
* module.loaded 返回一个布尔值，表示模块是否已经完成加载。
* module.parent 返回一个对象，表示调用该模块的模块。
* module.children 返回一个数组，表示该模块要用到的其他模块。
* module.exports 初始值为一个空对象{}，表示模块对外输出的接口。

**2.1 module.exports属性**

module.exports属性表示当前模块对外输出的接口，其他文件加载该模块，实际上就是读取module.exports变量。

例如，我们在moduleA.js文件中定义funA方法，并用module.exports变量把该方法暴露出，实例代码如下：

module.exports.funcA= function(){

console.log('This is moduleA!');

}

然后，在moduleB模块中加载引入moduleA模块，便可以使用funA方法了，示例代码如下：

var a = require('./moduleA');

a.funcA();//打印'This is moduleA!'

**2.2 exports变量**

为了方便，Node为每个模块（js文件）提供一个exports变量，指向module.exports。

在对外输出模块接口时，可以向exports对象添加方法。

var funA=function(){

console.log('This is module a!');

};

exports.funA = funA; //等同于module.exports.funA=funA;

**3 模块引用**

require函数的基本功能是，读入并执行一个JavaScript文件，然后返回该模块的exports对象。当我们用require()获取module时，Node会根据module.id找到对应的module，并返回module. exports，这样就实现了模块的输出。

有三个文件：

一个是a.js（存放路径：home/a.js）,

一个是b.js（存放路径：home/user/b.js）,

一个是c.js（存放路径：home/user/c.js）。

我们在a.js文件中引用三个模块，实例代码如下：

var httpModule=require('HTTP');//用 “模块名”加载服务模块http

var b=require('./user/b');//用“相对路径”加载文件b.js

var b=require('../ home/user/c');//用“绝对路径”加载文件c.js

**4 模块标识**

模块标识就是传递给require方法的参数，必须符合小驼峰命名的字符串，或者以.、..开头的相对路径，或者绝对路径，默认文件名后缀.js。

根据参数的不同格式，require命令去不同路径寻找模块文件

举例来说，脚本/home/user/projects/foo.js执行了require('bar.js')命令，Node会依次搜索以下文件。

/usr/local/lib/node/bar.js

/home/user/projects/node\_modules/bar.js

/home/user/node\_modules/bar.js

/home/node\_modules/bar.js

/node\_modules/bar.js

这样设计的目的是，使得不同的模块可以将所依赖的模块本地化。

CommonJS是同步的，意味着你想调用模块里的方法，必须先用require加载模块。这对服务器端的Nodejs来说不是问题，因为模块的JS文件都在本地硬盘上，CPU的读取时间非常快，同步不是问题。但如果是浏览器环境，要从服务器加载模块。模块的加载将取决于网速，如果采用同步，网络情绪不稳定时，页面可能卡住,这就必须采用异步模式。