**Node.js与JavaScript的区别是什么？**

**本质：**

     Javascript是一种web**前端语言**，主要用于[web开发](https://link.jianshu.com?t=https:/www.baidu.com/s?wd=web%E5%BC%80%E5%8F%91&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YkmyNWPWmvPWPbrjbYPWnv0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHT1n1n1rjm1rHmvrjbLPjDYPs)中，由浏览器解析执行

[Node.js](https://link.jianshu.com?t=https:/www.baidu.com/s?wd=Node.js&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YkmyNWPWmvPWPbrjbYPWnv0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHT1n1n1rjm1rHmvrjbLPjDYPs)是一个可以快速构建网络服务及应用的平台，是用Javascript语言构建的服务平台，可用于后端建立服务器

**区别：**

      Javascript 主要应用前端    是[编程语言    客户端](https://link.jianshu.com?t=https:/www.baidu.com/s?wd=%E7%BC%96%E7%A8%8B%E8%AF%AD%E8%A8%80&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YznAFhnvw-rHfsuHf1n1bz0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHcYnj6LPjfvn10Ln1bzPHfkrf" \t "_blank)[编程语言(需要](https://link.jianshu.com?t=https:/www.baidu.com/s?wd=%E7%BC%96%E7%A8%8B%E8%AF%AD%E8%A8%80&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YznAFhnvw-rHfsuHf1n1bz0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHcYnj6LPjfvn10Ln1bzPHfkrf)[浏览器](https://link.jianshu.com?t=https:/www.baidu.com/s?wd=%E6%B5%8F%E8%A7%88%E5%99%A8&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YznAFhnvw-rHfsuHf1n1bz0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHcYnj6LPjfvn10Ln1bzPHfkrf)的javascript解释器进行[解释执行](https://link.jianshu.com?t=https:/www.baidu.com/s?wd=%E8%A7%A3%E9%87%8A%E6%89%A7%E8%A1%8C&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YznAFhnvw-rHfsuHf1n1bz0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHcYnj6LPjfvn10Ln1bzPHfkrf" \t "_blank))

        node.js    主要应用后端      1个平台        运行环境(一个基于Chrome JavaScript运行时建立的平台，它是对Google V8引擎进行了封装的运行环境)

简单的说node.js就是把[浏览器](https://link.jianshu.com?t=https:/www.baidu.com/s?wd=%E6%B5%8F%E8%A7%88%E5%99%A8&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YznAFhnvw-rHfsuHf1n1bz0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHcYnj6LPjfvn10Ln1bzPHfkrf" \t "_blank)的解释器封装起来作为服务器运行平台，用类似javascript的结构语法进行编程，在node.js上运行。

**Node.js的优点？Node.js的缺点？**

**优点：**

1.事件驱动

2.异步编程

3.非阻塞模式的IO

4.轻量高效

**缺点：**

1.可靠性低

2.单进程，单线程，只支持单核cpu，不能充分的利用多核cpu服务器。一旦这个进程崩掉，那么整个web服务就崩掉了。

# Node.js的基本语法

基本语法与 javascript 一样

这里我们使用exports导出一个函数,然后再另一文件中通过require导入入一个模块

例如 ：

在 a.js 中写一个函数

var hello=function(){

    console.log('hello')

}

export.hello=hello;

在 b.js 中 需要用hello这个函数

var a=require('a');

a.hello()

# 单线程

Node的设计中就是将耗时长的操作代理给操作系统或者多线程，这部分操作就是磁盘I/O和网络I/O。所以Node中异步非常常见，因为要将耗时的操作从主线程上脱离。但是这些I/O线程并不怎么耗费CPU。

无法利用多核CPU是指Node的主线程无法利用上硬件上额外的CPU。这就需要用到cluster来在机器上启动多个Node实例，将额外的CPU也使用上。

# 异步 IO

## 事件驱动

## REPL

REPL（Read-Eval-Print Loop） 中文的话有翻译成“交互式解释器”或“交互式编程环境”的

## 为什么要学习node.js

* 能够和后端程序员更加紧密的配合
* 学习前端知识需要后端知识的支撑（ajax）
* 能够扩宽自己的知识视野

## node开发需要做的事情

* 实现网站的业务逻辑
* 实现数据的增删改查

## 选择node的意义

* 使用javascript语法开发后端应用
* 一些公司要求前端工程师必须要掌握node开发
* 生态系统活跃，有大量开源库可以使用
* 前端开发工具大多都是基于node开发

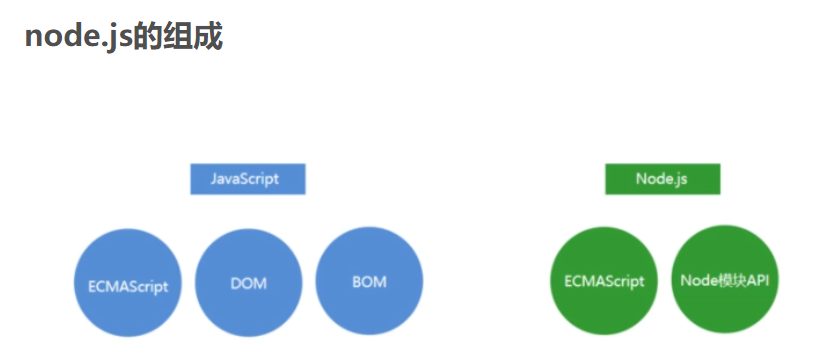
## node开发的好处

node是基于chrome v8引擎的javascript代码运行环境

运行环境

浏览器 能够运行 javascript代码，浏览器就是javascript的环境

Node 能够运行 javascript代码 node 就是



## Node如何执行js文件

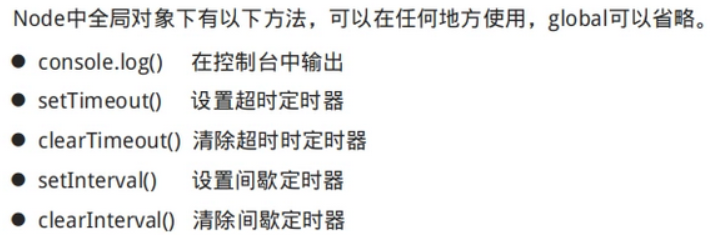
* cd:跳转目录  
  **cd …/:跳转上一级目录**
* clear:清除命令行

## Node打开js文件

点击键盘上的shift键，再次点击鼠标，打开命令行工具，即可打开js文件

## global

浏览器全局对象是window Node.js全局对象是global



global.console.log('hello Node.js');

global.setTimeout(() => {

console.log('hello Node');

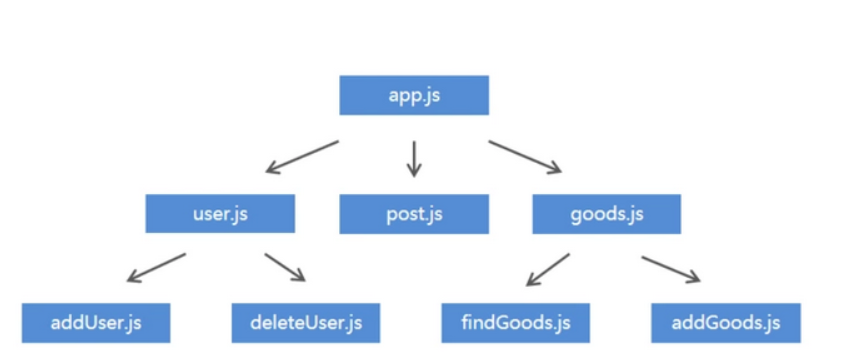
}, 2000);

# Node.js模块化开发

js在使用的时候存在两大问题：文件依赖及命名冲突，而Node模块化开发恰恰解决这个问题，文件之间处于半封闭的状态，把需要的模块暴露出来，不需要的模块则不需要暴露出来

## 软件中的模块化开发

一个功能就是一个模块，多个模块组成一个完整的应用，抽取一个模块不会影响其它模块的运行



## 模块化开发的规范



### 模块开发的第一种方法

**a模块**

const add = (n1, n2) => n1 + n2;

// 导出a模块

exports.add = add;

**b模块**

// 导入a模块

const b = require('./a');

// 使用a模块的方法

console.log(b.add(20, 50));

### 模块开发的第二种方法

使用module.exports导出模块

**a模块**

const Name = name => `hello ${name}`;

module.exports.Name = Name;

const x = 100;

// 导出模块

exports.x = x;

**b模块**

let b = require('./a');

console.log(b.Name('尧子陌'));

console.log(b.x)