

**Nodejs基础**

目录

[目录 2](#_Toc476695052)

[第1章 介绍 5](#_Toc476695053)

[1.1 宿主环境 5](#_Toc476695054)

[1.2 安装 6](#_Toc476695055)

[1.3 环境变量 6](#_Toc476695056)

[1.4 体验Nodejs 10](#_Toc476695057)

[第2章 基础 11](#_Toc476695058)

[2.1 基础语法 11](#_Toc476695059)

[2.2 ES6 12](#_Toc476695060)

[2.2.1 块级作用域 12](#_Toc476695061)

[2.2.2 let 关建字 13](#_Toc476695062)

[2.2.3 const关键字 13](#_Toc476695063)

[2.2.4 变量解构赋值 13](#_Toc476695064)

[2.2.5 字符串扩展 13](#_Toc476695065)

[2.2.6 数组扩展 14](#_Toc476695066)

[2.2.7 函数扩展 14](#_Toc476695067)

[第3章 模块 14](#_Toc476695068)

[3.1 系统模块 14](#_Toc476695069)

[3.2 文件模块 15](#_Toc476695070)

[3.2.1 用户模块 15](#_Toc476695071)

[3.2.2 第三方模块 15](#_Toc476695072)

[第4章 http服务 17](#_Toc476695074)

[4.1 搭建服务器 17](#_Toc476695075)

[4.2 处理响应 18](#_Toc476695076)

[4.3 处理请求 18](#_Toc476695077)

[4.4 静态网站 19](#_Toc476695078)

[4.4.1 路径 20](#_Toc476695079)

[4.5 动态网站 21](#_Toc476695080)

[4.5.1 路由 22](#_Toc476695081)

[4.5.2 模板引擎 22](#_Toc476695082)

[第5章 Express 23](#_Toc476695083)

[5.1 安装 23](#_Toc476695084)

[5.2 启动 23](#_Toc476695085)

[5.3 路由 23](#_Toc476695086)

[5.4 处理响应 24](#_Toc476695087)

[5.5 处理请求 24](#_Toc476695088)

[5.6 静态资源 25](#_Toc476695089)

[5.7 模板引擎 25](#_Toc476695090)

[5.8 模板布局 26](#_Toc476695091)

[5.9 中间件 26](#_Toc476695092)

[第6章 数据库软件 27](#_Toc476695093)

[6.1 安装 27](#_Toc476695094)

[6.2 数据库 27](#_Toc476695095)

[6.3 客户端 28](#_Toc476695096)

[6.3.1 CMD 28](#_Toc476695097)

[6.3.2 可视化工具 30](#_Toc476695098)

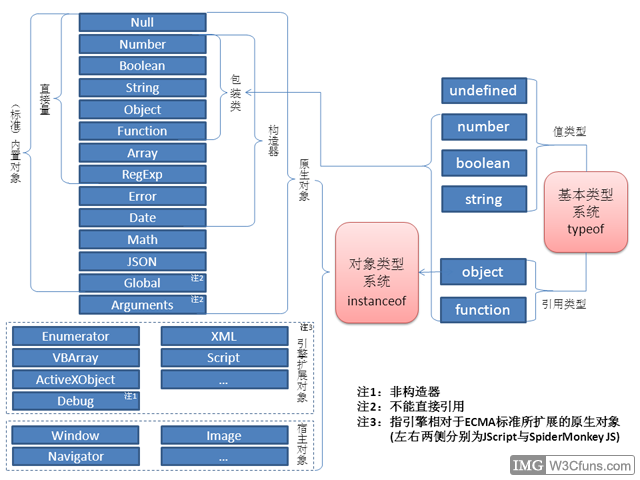
[6.3.3 Nodejs 31](#_Toc476695099)

# 介绍

Node.js是一个基于Chrome V8引擎的JavaScript运行环境，其使用了一个事件驱动、非阻塞式 I/O的模型，使其轻量又高效。

## 宿主环境

以网景的Netscape Navigator内置的JavaScript 1.1为蓝本，由ECMA-262定义的 ECMAScript 是一种脚本语言标准，它对该脚本语言的语法、类型、语句、操作符等做出明确规定。



Javascript是由ECMAScript、DOM、BOM三部分组成，浏览器实现了ECMAScript标准，使其能够运行于浏览器，我们称浏览器为ECMAScript的宿主环境。

除了可以运行于浏览器环境外，ECMAScript还可以运行于Adobe Flash环境中（称为Action Script），这时Adobe Flash也是ECMAScript的宿主环境。

我们即将学习的Nodejs就是另一种可以运行ECMAScript的宿主环境。

## 安装

Nodejs是开源、跨平台（即可以运行在不同的操作系统上）的。

Windows平台，Nodejs就是一个以.msi结尾的软件，由于Windows操作系统分成32位和64位，所以在Nodejs在Windows上也分为[32位](https://nodejs.org/dist/v6.10.0/node-v6.10.0-x86.msi)和[64位](https://nodejs.org/dist/v6.10.0/node-v6.10.0-x64.msi)两个版本。

Linux平台，根据其发行版不同，其安装方式也不同，但都可以使用源码编译安装。

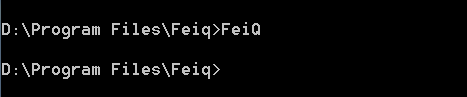
Mac平台，Nodejs是一个.pkg结尾的包，双击根据引导即可完成安装。

不论何种平台，检测Nodejs是否安装都可以通过命令 node -v，这个命令是执行结果会返回当前Nodejs的版本，以Windows平台为例，如下图：



## 环境变量

执行命令node -v时，其实是运行了一个名叫node.exe的可执行程序，操作系统自动查找并调用node.exe这个可执行程序，同样的我们也可以使用命令来执行其它可执行程序，以FeiQ.exe为例，如下图所示：

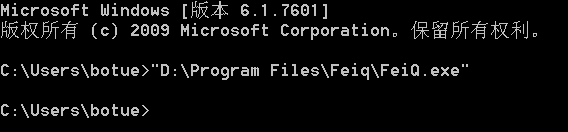


其实并不是在任意位置都可以执行feiq命令(windows平台不区分大小写)，如下图所示：



必须输入FeiQ.exe的绝对路径或将目录调整到FeiQ.exe所在目录才能被执行，

如下图所示：



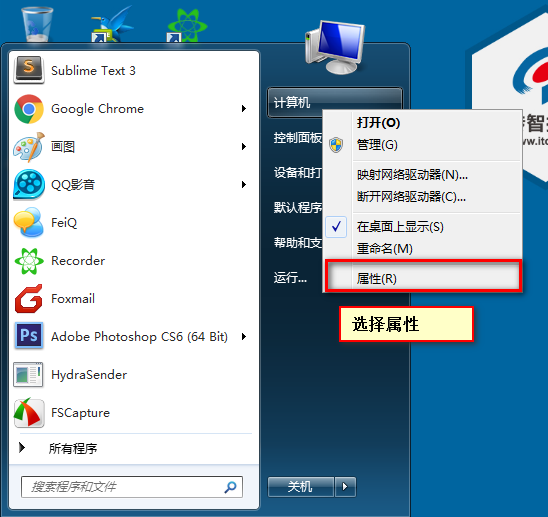
但是在执行node命令时却不需要，出现这种差别的原因在于环境变量。

环境[变量](http://baike.baidu.com/view/296689.htm)是在操作系统中一个具有特定名字的对象，它包含了一个或者多个应用[程序](http://baike.baidu.com/view/17674.htm)所将使用到的信息。

当在命令行窗口输入某个命令要求系统运行一个[程序](http://baike.baidu.com/view/17674.htm)而没有告诉它程序所在的完整路径时，操作系统会优先[当前目录](http://baike.baidu.com/view/396351.htm)下面查找此程序，找到则执行，找不到会到环境变量path中去找。

一般情况下安装Nodejs时，会自动设置path环境变量，所以安装完成立即执行node -v就能被执行，但是也有各别情况环境变量不能自动被添加，这时就需要手动添加。

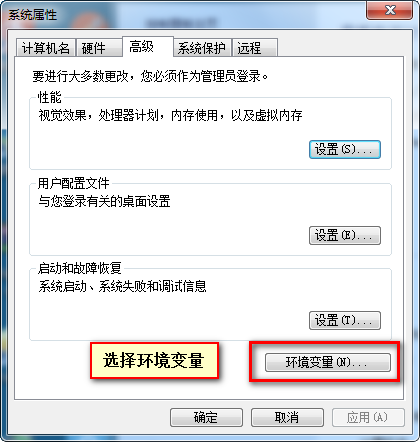
步骤1



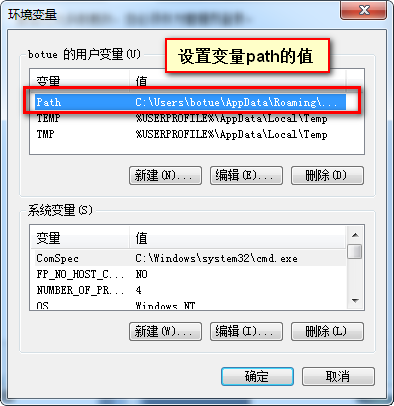
步骤2



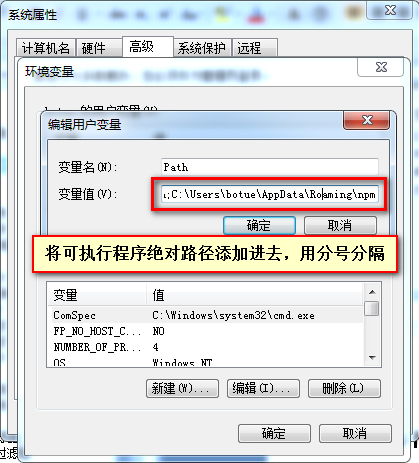
步骤3



步骤4

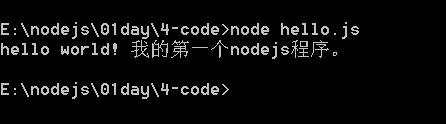


步骤5



## 体验Nodejs

宿主环境不同依据ECMAScript标准实现的脚本的执行方式也不同，例如Javascript必须通过script标签引入，然后在浏览器环境才能被执行，Nodejs则是需要通过命令行工具以命令的形式执行。



见示例代码 01hello.js

# 基础

## 基础语法

Nodejs是ECMAScript的宿主环境，与浏览器中的Javascript一奶同胞，所以其基础语法是一致的，都遵循ECMAScript 的标准规范。

见示例代码 02syntax.js

实践中不同的宿主环境又会在ECMAScript的基础上进行扩展，例如浏览器的javascript就扩展了window、location、navigator等对象，在Nodejs中则没有这些对象。

Nodejs也进行了扩展，主要体现在全局变量和全局方法两方面，如下：

1、require 用来加载并执行其它脚本文件，见示例代码03 require.js

2、module用来导出脚本文件的属性或方法，详见模块章节

3、exports 用来导出脚本文件的属性或方法，详见模块章节

4、\_\_dirname 用来获取当前文件目录，见示例代码 04\_\_dirname.js

5、\_\_filename 用来获取当前文件的绝对路径，见示例代码 0 \_\_filename.js

6、global 全局对象，类似于浏览器中的window。

a) setTimeout/clearTimeout

b) setInterval/clearInterval

c) console

见示例代码 06global.js

## ES6

Javascript在设计之初只是用来做简单的数据较验，后来发展到可以做一些网页的动态效果用来增强用户体验，再后来ECMA组织负责组织制订javascript规范的制订，将其称为ECMAScript，伴随着互联网的迅速发展，对ECMAScript提出了更高的要求，为了适应开发的需要，ECMA组织也在不断的修改完善ECMAScript的标准（如ECMAScript6简称ES6）。

然而新的标准所定义的规范在浏览器中的javascript中并不能得到很好的应用，原因在于浏览器种类较多，开发者并不能约束用户必须使用何种浏览器，导致浏览器高低版本共存是常态，为了考虑兼容，开发者不得不放弃使用一些高级特性。

Nodejs则不会出现上述情况，Nodejs由统一的团队组织维护，通过版本升级方式可以及时跟进ECMAScript的标准，而且多数情况Nodejs是运行于服务端的，开发者可以决定使用Nodejs的版本，很多高级特性可以在Nodejs中得到很好的运用。

### 块级作用域

任何一对花括号（｛｝）中的语句集都属于一个块，在块中声明的变量在代码块外都是不可访问的，称之为块级作用域，ES5以前没有块级作用域，见示例代码 07scope.js

### let 关建字

let用于声明变量（类似于var），其声明的变量只在当前代码块内有效，实际上是创建了块级作用域，见示例代码08 let.js

### const关键字

const 用于声明常量，常量于变量不同，变量声明后其值可以任意更改，而常量一旦赋值就不允许被更改，见示例代码09 const.js

### 变量解构赋值

按照某种模式，从数组和对象中提取值，对变量进行赋值这被称为解构。

a) 数组解构赋值，见示例代码 10array-structure.js

b) 对象解构赋值，见代码代码 11object-structure.js

以上介绍了解构赋值的语法规则，可以在现实开发中带来很大的便利，提升开发效率，见代码示例12 case.js

### 字符串扩展

a) 对象方法，新增一些用于处理字符串的便捷方法，见示例代码 13string-extends-1.js

b) 模板字符串，可以使用反引号（`）定义字符串，见示例代码 14string-extends-2.js

### 数组扩展

见示例代码 15array-extends.js

### 函数扩展

a) 默认参数，允许为函数指定默认参数，见示例代码16 function-extends-1.js

b) reset参数，用于获取多余的参数，将多余的参数装到一个数组中，见示例代码 17function-extends-2.js

c) 扩展运算符，它好比 rest 参数的逆运算，将一个数组转为用逗号分隔的参数序列，见示例代码 18function-extends-3.js

d) 箭头函数，使用“箭头”(=>) 定义函数，见示例代码 19function-extends.js

# 模块

Nodejs在ECMAScript的基础上扩展并封装了许多高级特性，如文件访问、网络访问等，使得Nodejs成为一个很好的Web开发平台。

基于Nodejs这个平台将Web开发常用的一些功能进行封装，称为模块。

## 系统模块

在安装Nodejs时一些模块自动被安装，这些模块大多由Nodejs官方维护，称这类模块为系统（核心）模块，如fs、path、os、http等。

模块被加载后才能使用，Nodejs提供了全局的函数require加载模块。

1、os模块 获取当前操作系统信息，见代码示例 20os.js

2、fs模块 文件读写操作，见代码示例 21fs.js

3、path模块 路径处理，见代码示例 22path.js

## 文件模块

除了系统模块外，还有以文件形式存在的模块，这些模块由一些组织或个人根据不同的业务需求开发。

注：模块也被称为包。

### 用户模块

1、Nodejs中文件模块分两种，一种是.js结尾，一种是以.json结尾，见代码示例 23cart.js和 24usrs.json

2、require加载模块路径

相对路径，如 ./ 或 ../

绝对路径，如 E:\nodejs\02day\4-code\23cart 或 / (类unix系统)

直接写模块名会被当成系统模块或加载node\_modules目录的模块

见代码示例 25case.js

3、exports和module.exports暴露模块接口，见代码示例26interface.js

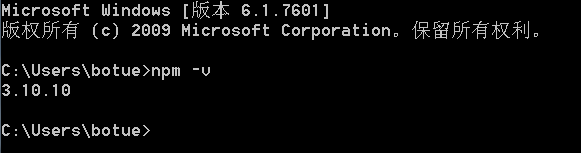
4、多文件模块（也叫包）指某一个功能模块的集合，通常由多个文件构成，统一到某一目录集中管理，并且使用package.json记录包的相关信息，例如模块名称、版本号、作者等，可使用npm init根据提示创建。

### 第三方模块

到目前为止，第三方模块数量已经相当庞大，为了能够更好的管理这些模块，Nodejs官方提供了一个工具（npm）专门用于管理这些第三方模块。



npm (Node Package Manager) 这个工具随Nodejs一起被安装，可以使用 npm -v 检测是否安装，如下图



使用npm可以安装、卸载、发布模块。

1、安装

a) 本地安装，安将到当前目录下node\_modules中

npm install 模块名称

b) 全局安装，安装到默认c盘

npm install 模块名称 -g

2、卸载

a) 本地卸载

npm uninstall 模块名称

b) 全局卸载

npm uninstall 模块名称 -g

3、发布模块

a) 注册用户，注册[npm账号](https://www.npmjs.com/signup)

b) 编写模块

新建helloitcast.js，编写简单逻辑。

c) 发布模块

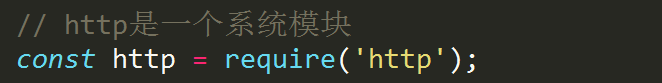
在模块目录执行 npm login，根据提示输入用户名、密码、邮箱进行登录，然后执行npm publish完成发布。

# http服务

使用Nodejs系统模块http可以创建并启动http服务器。

## 搭建服务器

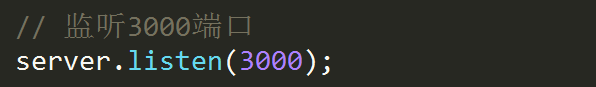
1、引入http模块



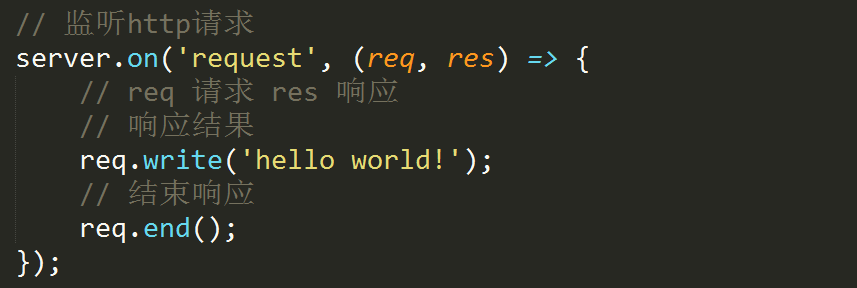
2、创建http服务实例



3、监听端口



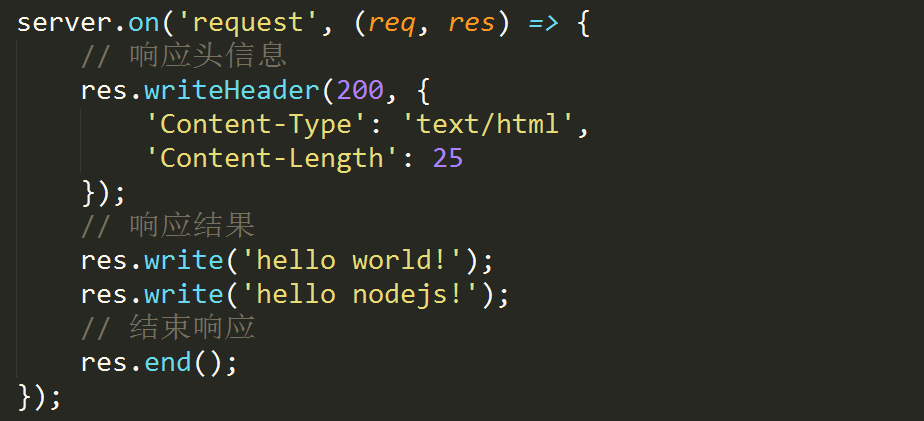
4、监听http请求



5、浏览器访问 http://localhost:3000

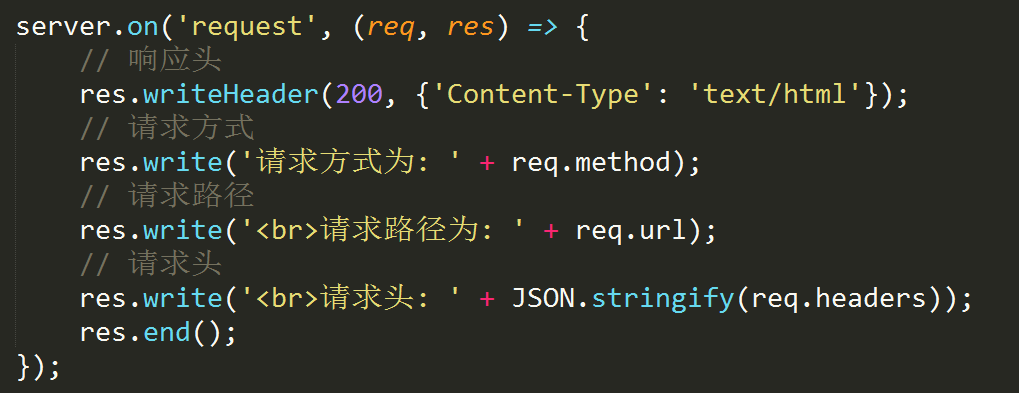
## 处理响应

通过res对象处理响应，如状态码，响应头，响应主体等。

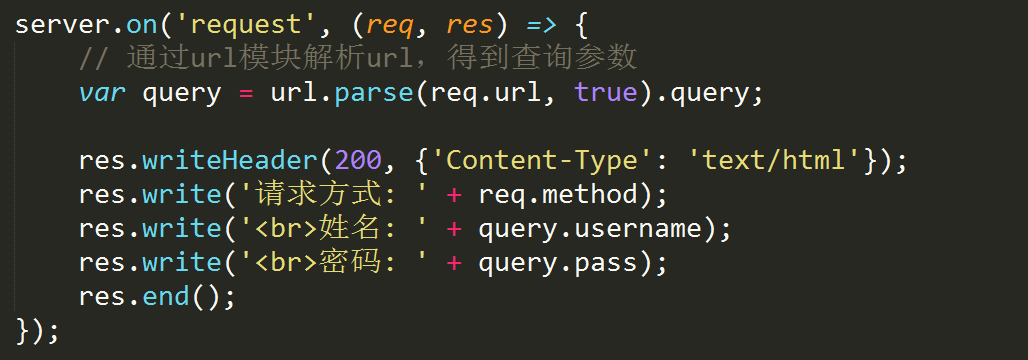


## 处理请求

使用req对象处理请求，如请求方式、请求头、查询参数、请求主体等。



get 查询参数，查询字符串做为地址的一部分，需要使用url系统模块解析得到。



post参数做为请求主体传递



## 静态网站

我们通过浏览器所能看到的网页内容实质上是服务器返回的html字符串，浏览器进行渲染输出。



浏览器解析html标签过程中，遇到img、script、link、iframe等标签时会再次向服务器发起请求，服务器也需要做出响应。



### 路径

路径是描述位置的一种方式，遵循一定规则，即路径都有一个起点称之为“根”，以“/”或“\”做为分隔符，使用“.”和“..”分别表示当前级和上一级。路径分为绝对路径和相对路径。

将以“根”开始，使用“/”或“\”分隔的路径称为绝对路径。

将以“.”或“..”开始，使用“/”或“\” 分隔的路径称为相对路径。

1、文件系统路径

操作系统上的文件都是存储在磁盘中的，使用路径可以准确确定到某个文件的位置。

文件系统中盘符（windows）或“/”（类unix）为路径的“根”，使用分隔符“/”或“\”分隔的是真实存在的目录名称和文件名。

2、网络路径

网络路径即url，用来标识互联网资源位置的一种方式，通常由协议、服务器地址、路径、文件名构成。

网络中服务器地址开始为路径的“根”（用“/”）表示，使用分隔符“/”或“\”分隔的不一定是真实的目录名称和文件名。

网络中的“根”路径可以通过服务器软件映射到文件系统中的任意位置，例如 Apache 配置documentRoot更网站根目录。

目录不等路径

根+目录+分隔符+文件名 格式

E:\asdf\sadf\aa.txt 文件系统的

http://localhost/dsaf/ads/index.html 网络系统的

判断路径位置应从路径本身出发，而不是站在目录的角度观察。

在文件系统中目录和文件都是真实存在的，路径与目录和文件的关系是一一对应的，以目录角度确定路径不出有问题。

但是网络中目录和文件不一定真实存在，路径与目录和文件不一定是一一对应的关系，以目录角度确定路径就会出问题。

## 动态网站

所谓动态网站是指网页内容由程序动态生成，通常是将数据库中的数据取出，然后拼接成html字符串，响应给浏览器进行解析。

### 路由

路由是指URL地址与程序的映射关系，更改URL地址可以改变程序的执行结果。



### 模板引擎

从数据库中取出数据后需要拼接成html字符串再响应给浏览器，这个任务通常由模板引擎完成，模版引擎的种类非常多，比如artTemplate、swig、xtpl等。



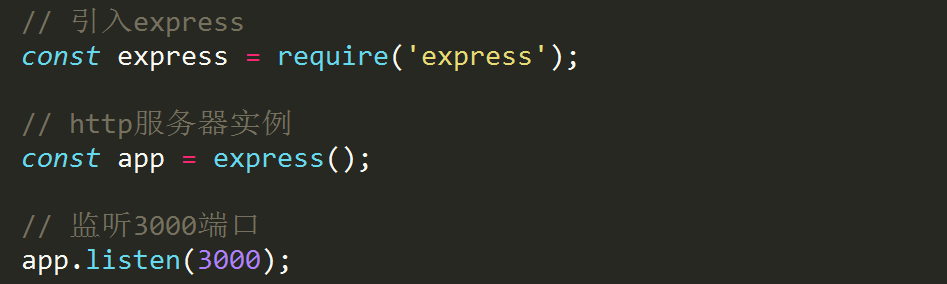
# Express

使用Nodejs可以非常快速的搭建服务器，而在真正的开发实践中，一般都会使用框架进行开发，Express框架Nodejs中非常流行的框架之一。

## 安装

Express 使用 npm install express进行本地安装。

## 启动



## 路由

Express对路由进行封装使用非常方便。



## 处理响应

Express可以响应多种格式的数据。

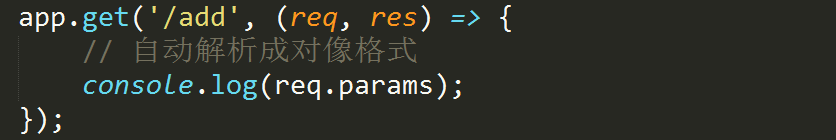


## 处理请求

a) 使用req.params或req.query接收get方式请求的参数

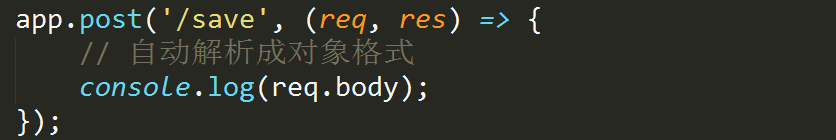
req.query接收的参数是“?”后面部分

req.params 接收的参数是通过路由 “:”定义的“形参”

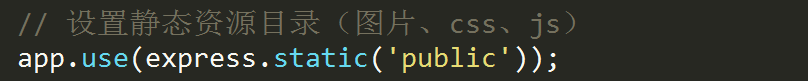


b) 使用req.body接收post方式请求的参数

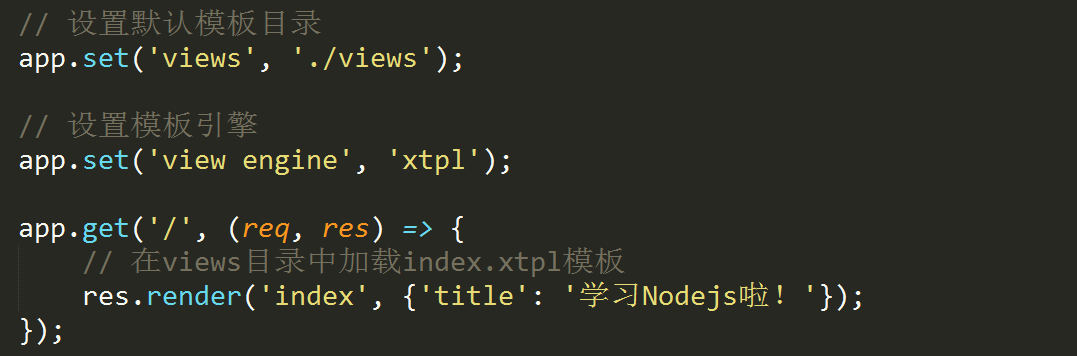
必须使用中间件后才能使用，即app.use();



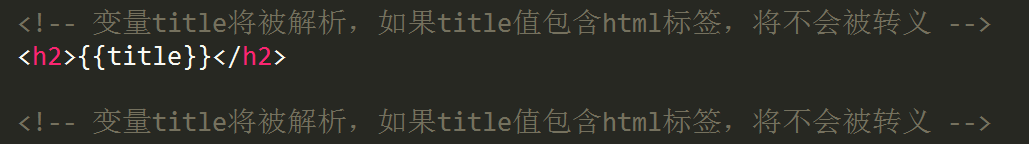
## 静态资源



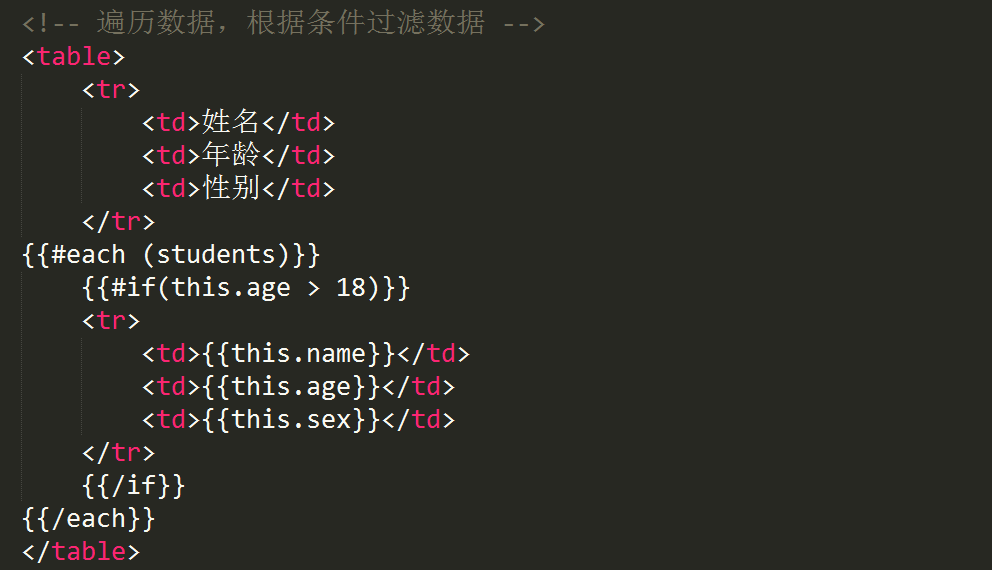
## 模板引擎



a) 使用 “{{}}”和“{{{}}}”输出变量。



b) 流程控制 “{{# 关键字}}”和 {{/关键字}}，



## 模板布局

使用模板布局可以将网页的公共部分（头部、底部）抽离出来，实现复用。

## 中间件

Express将请求与响应的过程“划分”成多个阶段，任何一个阶段都允许调用一个函数对请求或响应做出修改，这个函数就是中间件，Express使用use方法调用中间件。



还可以为不同的路由设置不同的中间件



注：中间件是按顺序执行的。

# 数据库软件

数据库是存储网站数据的一种方式，通过数据软件进行管理，常见的数据库软件有MySQL、Oracle、Sql Server等，MySQL由瑞典MySQL AB 公司开发，目前属于 [Oracle](http://baike.baidu.com/view/15020.htm) 旗下产品，MySQL 是最流行的[关系型数据库管理系统](http://baike.baidu.com/view/1450387.htm" \t "_blank)之一。

## 安装

MySQL也是一个跨平台的软件（支持多个操作系统），以Windows为例，MySQL是一个可执行程序，其安装方式与普通软件一样。

在学习Ajax阶段安装的wampserver是一个包含了Apache、PHP、MySQL的集成环境，不需要单独进行安装。

## 数据库

数据库（Database）是按照数据结构来组织、存储和管理数据的仓库，类似于EXCEL表格管理数据一样。





## 客户端

MySQL是一个C/S结构的软件系统，上述步骤安装了MySQL的服务器（即Server），通过客户端（Client）访问MySQL服务器。

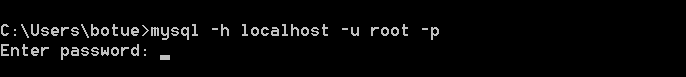
### CMD

1、登录

通过客户端访问数据库服务器，CMD是一个可以访问MySQL的客户端，使用CMD访问MySQL需要添加环境变量。

找到MySQL的安装路径D:\wamp\bin\mysql\mysql5.5.20\bin，然后添加到环境变量中。

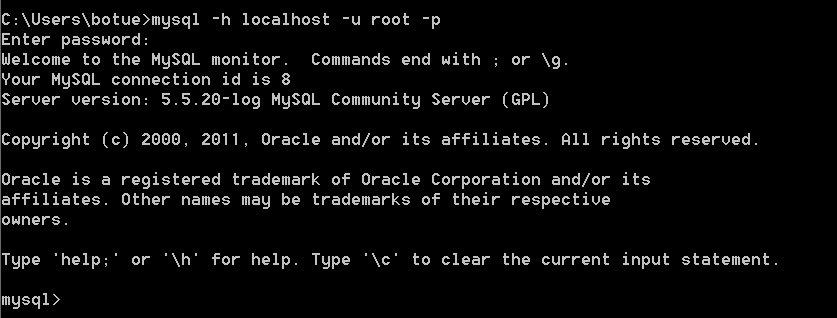
使用如下命令登录



-h 表示MySQL服务器地址

-u 表示用户名称 默认 root

-p 表单用户密码 默认 空



2、创建数据库



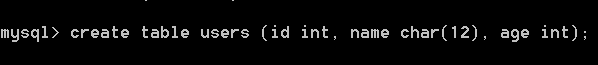
3、查看数据库



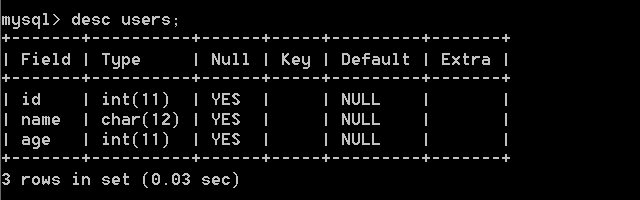
4、切换数据库



5、创建数据表



6、查看表结构



### 可视化工具

CMD做为客户端可以访问MySQL服务器，但工作效率非常低，实际开发中一般使用可视化的工具管理数据库。

可视工具种类非常多，推荐使用Navicat for MySQL，它是一个付费软件。

1、登录

登录界面操作

2、创建数据库

CREATE DATABASE 数据库名称

3、创建数据表

图形界面创建

4、插入数据

INSERT INTO 表名 (字段1, 字段2, 字段3...) VALUES(值1, 值2, 值3...);

例如INSERT INTO `users` (id, name, age, pass) VALUES(4, '赵六', 18, '8213121');

5、查询数据

SELECT 字段名1，字段2，字段名3... FROM 表名;

例如SELECT `name`, `sex`, `email`, `pass` FROM users;

6、更新数据

UPDATE 表名 set 字段名1='值1', 字段名2='值2', ....

例如 update users set pass='abcdefg', name='xiaohong', age=20

7、删除数据

DELETE FROM 表名 WHERE 条件

例如delete from users where age=20

8、筛选数据

带条件的查询

SELECT 字段名1, 字段名2, 字段名3.. WHERE 条件 AND 条年 OR 条件

例如select id, name, age from users where age<18 and sex='boy' or age>18 and sex='girl'

9、排序数据

查找数据并将数据排序

SELECT 字段名1, 字段名2.. FROM 表名 ORDER BY 排序条件 顺序

ASC 正序 DESC 倒序

例如select id, name, age from users order by age desc

10、分组数据

查找数据并分组进行统计操作

SELECT 统计操作 FROM 表名 GROUP BY 条件

count统计 sum求合 avg 平均值

select avg(age) as total from users group by sex 分组求平均年龄

select sum(age) as total from users group by sex 分组求年龄和

select count(\*) as total from users group by sex 计算分组合每组个数

11、导出导入数据

数据库是个做迁移操作的，其中包含了导入和导出操作。

12、数据类型

每个字段对应的列的数据是要有特定类型的，比如 数值、字符、时间等。

[参见文档](http://www.runoob.com/mysql/mysql-data-types.html)，此内容了解即可

13、主键

表中每一行都应该有可以唯一标识自己的一个字段，此字段称为主键（PRIMARY KEY），此字段值不允许重复，一般设为自增长（AUTO\_INCREMENT）

### Nodejs

可视化工具是辅助管理数据的一种工具，数据库应用在Web中时都是要通过程序进行操作的，例如Nodejs、PHP、JAVA都是可以操作数据库的程序。

在Nodejs中使用第三方模块获得对MySQL操作的支持。

1、安装mysql模块

npm install mysql 本地安装

2、登录

引入mysql模块，提供正确的主机地址、用户名、密码、数据名连接服务器并登录。



3、数据库操作

使用数据库实例db的query方法对数据库执行“增、删、改、查”的操作。

a) 增加

b) 删除

c) 修改

d) 查询

e) 排序

f) 分组

使用占位符“?”可以简化操作，当使用多个占位符“?”时，数据使用数组格式按占位符“?”的顺序一一对应。

# Promise

在ECMAScript规范下的Javascript和Nodejs中，执行异步操作时会大量使用回调函数，这种方式本质上没有明显的缺陷，但是回调函数会产生层层的嵌套，使得代码变的即丑陋又难以进行维护，ES6中提代了内置对象Promise来解决此类问题。

由于异步操作常常是一个“未来”的操作，执行结果具有不确定性，使用Promise可以根据异步执行的结果处理不同的逻辑，Promise译成中文是承诺的意思，即对未来的一种承诺。

