一、了解node.js

1.Node.js是一个可以让 JavaScript 运行在服务器端的平台，是一个为实时Web（Real-time Web）应用开发而诞生的平台，它从诞生之初就充分考虑了在实时响应、超大规模数据要求下架构的可扩展性。这使得它摒弃了传统平台依靠多线程来实现高并发的设计思路，而采用了**单线程、异步式I/O、事件驱动**式的程序设计模型。这些特性不仅带来了巨大的性能提升，还减少了多线程程序设计的复杂性，进而提高了开发效率。

2.Node.js 有着强大而灵活的包管理器（node package manager，npm）.

3.Node.js 是一个让 JavaScript 运行在服务端的开发平台，它让 JavaScript 成为脚本语言世界的一等公民，在服务端堪与 PHP、Python、Perl、Ruby 平起平坐。

4. Node.js 可以作为服务器向用户提供服务，与 PHP、Python、Ruby on Rails 相比，它跳过了 Apache、Nginx 等 HTTP服务器，直接面向前端开发。Node.js 的许多设计理念与经典架构（如 LAMP）有着很大的不同，可提供强大的伸缩能力，以适应21世纪10年代以后规模越来越庞大的互联网环境。

5. Node.js 的 JavaScript 引擎是 V8，来自 Google Chrome 项目。V8 号称是目前世界上最快的 JavaScript 引擎，经历了数次引擎革命，它的 JIT（Just-in-time Compilation，即时编译）执行速度已经快到了接近本地代码的执行速度。Node.js 不运行在浏览器中，所以也就不存在 JavaScript 的浏览器兼容性问题，你可以放心地使用 JavaScript 语言的所有特性。

6. Node.js 最大的特点就是采用异步式 I/O 与事件驱动的架构设计.

7. CommonJS 试图定义一套普通应用程序使用的API，从而填补 JavaScript 标准库过于简单的不足. 。CommonJS 的终极目标是制定一个像 C++ 标准库一样的规范，使得基于 CommonJS API 的应用程序可以在不同的环境下运行.

ECMAscript是JS在浏览器中的标准；浏览器之外的标准由CommonJS制定，具体实现里nodejs是最出色的一个。

二、编译（window系统）

Node.js 在 Windows 下只能通过 Microsoft Visual Studio 编译，因此你需要首先安装 Visual Studio 或者免费的 Visual Studio Express。你还需要安装 Python 2（2.5以上的版本，但要小于3.0），可以在http://python.org/取得。安装完 Python 以后请确保在PATH环境变量中添加python.exe 所在的目录，如果没有则需要手动在“系统属性”中添加。

三、编程

1.运行 Node.js 程序

node script.js

node -e "console.log('hello')"；

2. REPL模式

REPL （Read-eval-print loop），即输入—求值—输出循环

在任何时候，连续按两次 Ctrl + C 即可推出Node.js 的 REPL 模式

3. 调试小技巧supervisor（英文含义：监督）

supervisor app.js 在开发 Node.js 实现的 HTTP 应用时会发现，无论你修改了代码的哪一部份，都必须终止Node.js 再重新运行才会奏效。这是因为Node.js 只有在第一次引用到某部份时才会去解析脚本文件，以后都会直接访问内存，避免重复载入

4.npm命令

npm [install/i] [package\_name]本地模式

npm [install/i] -g [package\_name]全局模式

使用全局模式安装的包不能直接通过 require 使用，但通过 npm link命令

可以打破这一限制。如，我们已经通过 npm install -g express 安装了 express，

这时在工程的目录下运行命令：

$ npm link express

./node\_modules/express -> /usr/local/lib/node\_modules/express

我们可以在 node\_modules 子目录中发现一个指向安装到全局的包的符号链接。通过这

种方法，我们就可以把全局包当本地包来使用了。

注a: npm link 命令不支持Windows。

5.包的发布

1)创建一个文件夹（一般是包名）

2)Npm init

3)Npm adduser(根据提示填写用户名、密码、邮箱，我已经有一个wudan\_pnm pnm001 [1872776189@qq.com](mailto:1872776189@qq.com))

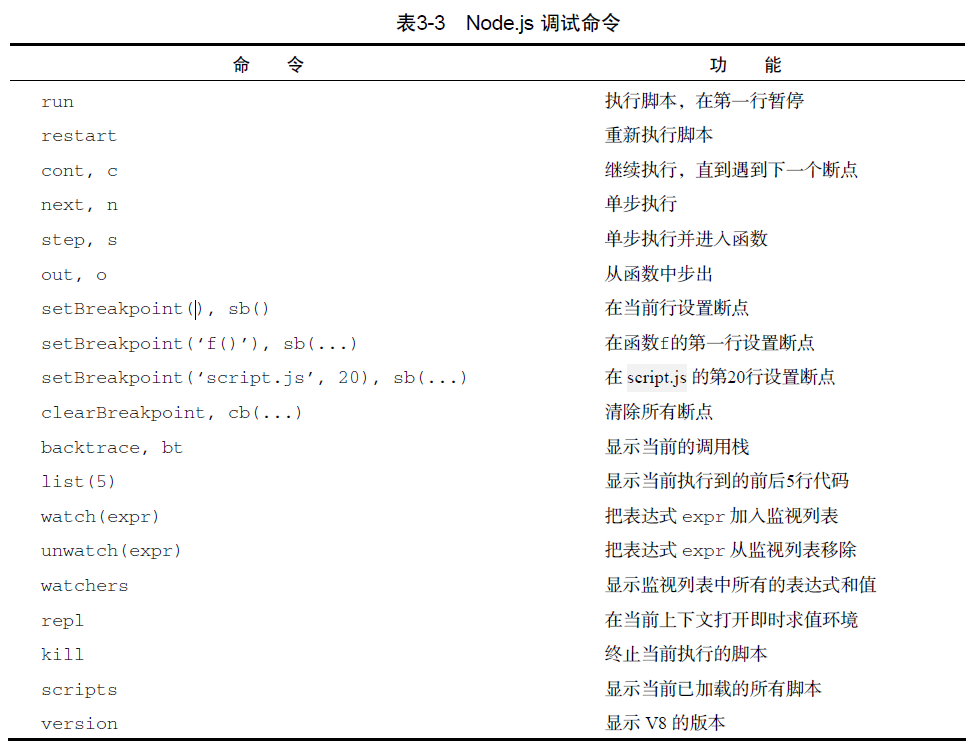
4)Npm publish

如果你的包将来有更新，只需要在 package.json 文件中修改 version 字段，然后重新使用 npm publish 命令就行了。如果你对已发布的包不满意（比如我们发布的这个毫无意义的包），可以使用 npm unpublish 命令来取消发布。

6.调试功能

在命令行下执行 node debug debug.js，将会启动调试工具，打开调试终端，下面是一些基本的命令进行调试

1)命令行调试



2）远程调试

V8 提供的调试功能是基于 TCP 协议的，因此 Node.js 可以轻松地实现远程调试。

node --debug[=port] script.js

node --debug-brk[=port] script.js

3）使用 Eclipse 调试 Node.js 程序

4）使用 node-inspector 调试 Node.js

四、核心基础

1．JavaScript 的面向对象特性是基于原型的，与常见的基于类的不同。

2.utils工具

util.inherits(constructor, superConstructor)是一个实现对象间原型继承的函数。

util.inspect(object,[showHidden],[depth],[colors])是一个将任意对象转换为字符串的方法，通常用于调试和错误输出。它至少接受一个参数 object，即要转换的对象。

四个类型测试工具：util.isArray()、util.isRegExp()、util.isDate()、util.isError()

3.事件发射器

events 模块只提供了一个对象： events.EventEmitter。EventEmitter 的核心就是事件发射与事件监听器功能的封装。

EventEmitter常用的API。

EventEmitter.on(event, listener) 为指定事件注册一个监听器，接受一个字

符串 event 和一个回调函数 listener。

EventEmitter.emit(event, [arg1], [arg2], [...]) 发射 event 事件，传

递若干可选参数到事件监听器的参数表。

EventEmitter.once(event, listener) 为指定事件注册一个单次监听器，即

监听器最多只会触发一次，触发后立刻解除该监听器。

EventEmitter.removeListener(event, listener) 移除指定事件的某个监听

器，listener 必须是该事件已经注册过的监听器。

EventEmitter.removeAllListeners([event]) 移除所有事件的所有监听器，

如果指定 event，则移除指定事件的所有监听器。

EventEmitter 定义了一个特殊的事件 error，它包含了“错误”的语义，我们在遇到

异常的时候通常会发射 error 事件。当 error 被发射时，EventEmitter 规定如果没有响应的监听器，Node.js 会把它当作异常，退出程序并打印调用栈。我们一般要为会发射 error事件的对象设置监听器，避免遇到错误后整个程序崩溃。

大多数时候我们不会直接使用 EventEmitter，而是在对象中继承它，原因有两点。首先，具有某个实体功能的对象实现事件符合语义，事件的监听和发射应该是一个对象的方法其次 JavaScript 的对象机制是基于原型的，支持部分多重继承，继承 EventEmitter 不会打乱对象原有的继承关系。

4. 文件系统 fs

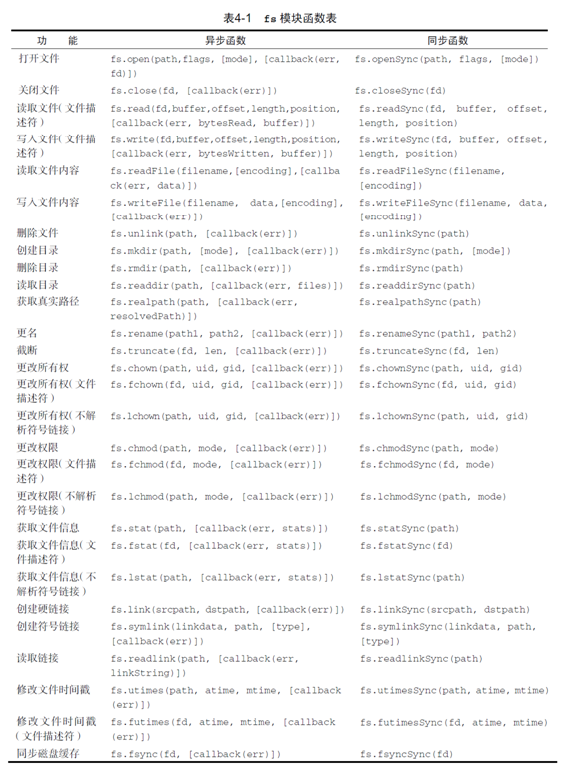
fs 模块是文件操作的封装，它提供了文件的读取、写入、更名、删除、遍历目录、链接等 POSIX 文件系统操作。与其他模块不同的是，fs 模块中所有的操作都提供了异步的和

同步的两个版本。

fs.readFile(filename,[encoding],[callback(err,data)])是最简单的读取文件的函数。（如果指定了 encoding，data 是一个解析后的字符串，否则 data 将会是以 Buffer 形式表示的二进制数据。）

fs.read的功能是从指定的文件描述符 fd 中读取数据并写入 buffer 指向的缓冲区对象.

表4-1列出了fs所有函数的定义和功能。



5.HTTP 服务器与客户端

（1）服务器端

http.Server 是 http 模块中基于事件的 HTTP 服务器对象

http.ServerRequest

http.ServerResponse有三个重要的成员函数，用于返回响应头、响应内容、结束请求。

response.writeHead(statusCode, [headers])：向请求的客户端发送响应头。

response.write(data, [encoding])：向请求的客户端发送响应内容。

response.end([data], [encoding])：结束响应，告知客户端所有发送已经完

成。

（2）客户端

http 模块提供了两个函数 http.request 和 http.get，功能是作为客户端向 HTTP

服务器发起请求。

1）http.request(options, callback) 发起 HTTP 请求。不要忘了通过 req.end() 结束请求，否则服务器将不会收到信息。

2）http.get(options, callback) 它是 http.request 的简化版，唯一的区别在于http.get自动将请求方法设为了 GET 请求，同时不需要手动调用 req.end()。

HTTP 客户端工具

五、使用node.js进行web开发

1．Express框架

它是目前最稳定、使用最广泛，而且 Node.js 官方推荐的唯一一个 Web 开发框架。

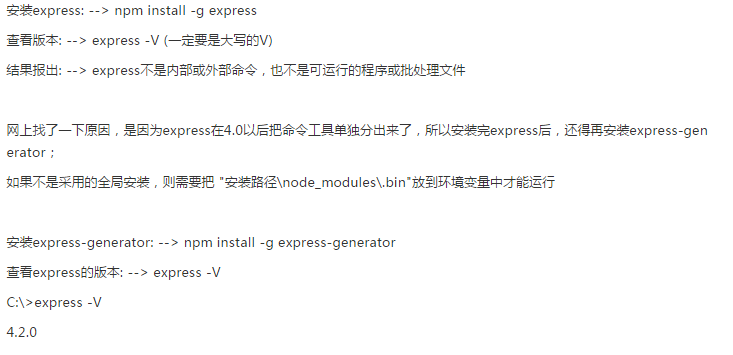
**下面用 Express 作为开发框架来开发一个网站（共三大步）：**

(1)安装express需要两步：

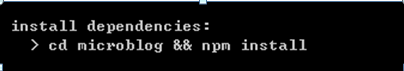
1）npm install -g express

2）npm install -g express-generator

原因如下：



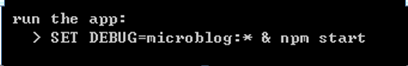
(2)建立工程express -e ejs microblog（express -t ejs microblog这种写法是老写法，现在已不适用，创建出来的view文件是jade的，因此不要用了。）



然后根据提示运行即可。

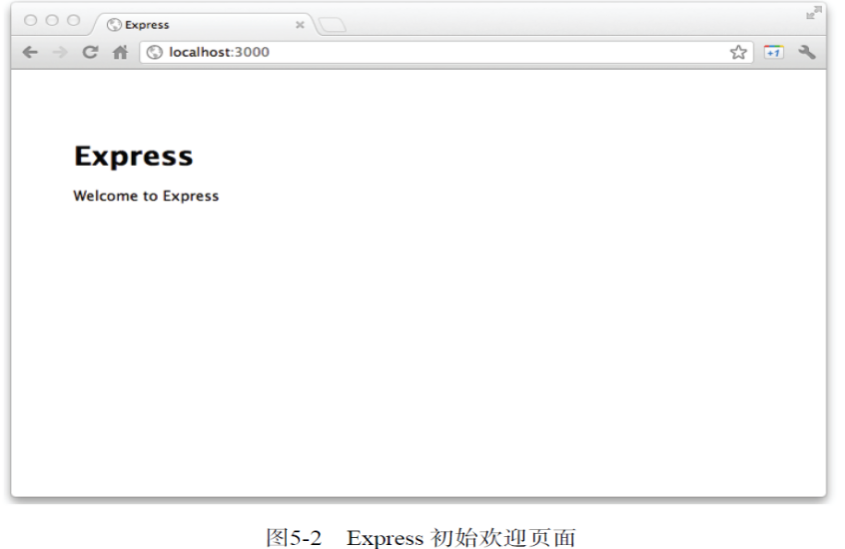
无参数的 npm install 的功能就是检查当前目录下的 package.json，并自动安装所有指定的依赖。

（3）启动服务

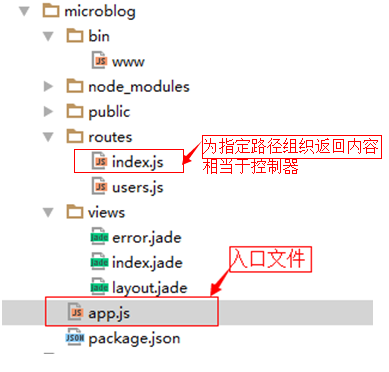


注：npm start 或者supervisor start( 通过查看package.js知道start为./bin/www，supervisor更适合调试，不用每次都关闭重启服务)

成功界面如图：

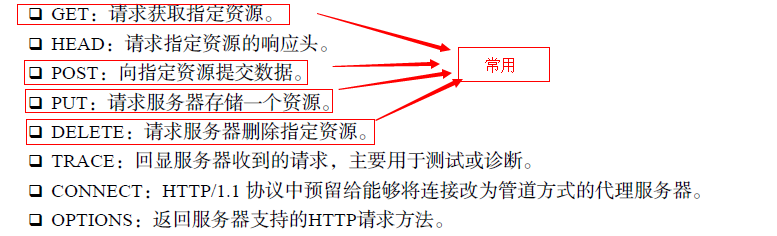
。

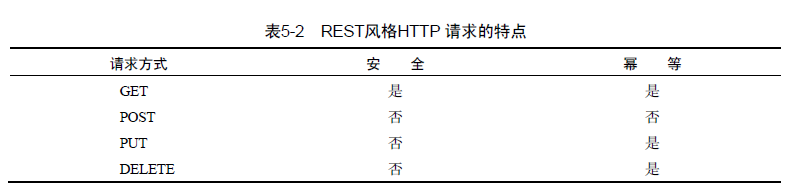
**工程结构**

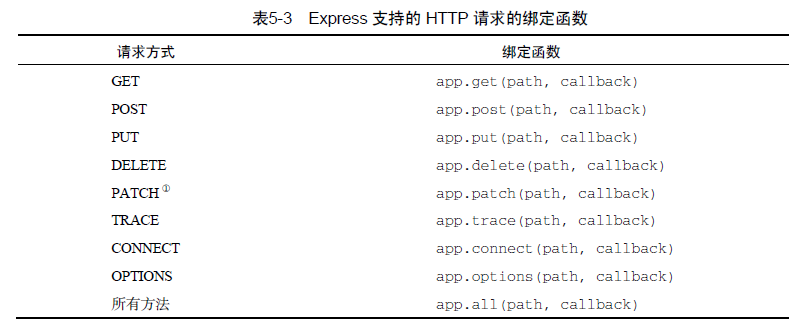


2.路由选择——rest（表征状态转移）风格，一种基于HTTP协议的网络应用的接口风格。

（1）HTTP定义了以下8中标准方法：







（2）控制权转移next()

Express支持统一路径绑定多个路由响应函数，但当访问任何两条同样的规则匹配到的路径时，会优先匹配先定义的路由规则，后面的相同规则被屏蔽。回调函数第三个参数即next会将控制权转移给后面的规则。举一个经典小例子。



（3）模板引擎

是一个从页面模板根据一定的规则生成HTML的工具。

其功能是将页面模板和要显示的数据结合起来生成HTML页面。

它既可以运行在服务器端又可以运行在客户端，大多数时候它都在服务器端直接被解析为HTML，解析完成后在传输给客户端，因此客户端甚至无法判断页面是否是模板引擎生成的。目前主流由服务器运行模板引擎。

res.render(‘index’,{title:’Express’});//功能是调用模板引擎，并将其产生的页面直接返回给客户端。第一个参数是模板的名称，即views目录下的模板文件名，不包含文件的扩展名；第二个参数是传递给模板的数据，用于模板翻译。

6．node.js进阶话题

1. 模块分为两大类：

核心模块：标准API中提供的模块，如fs、http、net、vm等

文件模块：存储为单独的文件（活文件夹）的模块，可能是JS代码、JSON或编译好的ｃ／ｃ＋＋代码。

1. 文件模块的加载有两种方式：

１）按路径加载模块（绝对路径、相对路径）

２）查找 node\_modules 文件夹。npm获取的包通常就是以这种方式加载。

1. 加载顺序

1) 如果some\_module 是一个核心模块，直接加载，结束。

2) 如果some\_module以“ / ”、“ ./ ”或“ ../ ”开头，按路径加载 some\_module，结束。

3) 假设当前目录为 current\_dir，按路径加载 current\_dir/node\_modules/some\_module。

 如果加载成功，结束。

 如果加载失败，令current\_dir为其父目录。

 重复这一过程，直到遇到根目录，抛出异常，结束。

Node.js核心：全局对象（变量）、util工具、事件、文件操作、HTTP服务

JS前台与后台的不同

1. 弱类型语言（变量），变量可以不声明，直接打点使用，即隐式定义。
2. JavaScript 的面向对象特性是基于原型的，与常见的基于类的不同。

**项目运行步骤：**

1）启动redis D:\redis64-latest>redis-server.exe redis.windows.conf

2）启动mongo D:\work\mongodb\bin>mongod.exe --dbpath=D:\work\jsgen\data

3）编译运行 F:\wudan\protect\jsgen>node app.js install

gulp

F:\wudan\protect\jsgen>node app.js recache

**第一周任务：讲一下nodejs是什么、有哪些类库、类库的特性、以及项目运行过程。**

**主要看一下英文版part2中间件的概念4——7（6.7）**

**Nodejs是什么、安装部署、基本知识（基本概念【包、模块】、核心模块【全局对象、util工具、文件系统、HTTP服务】）、Express创建的网站架构。**

https://exmail.qq.com/cgi-bin/viewfile?type=logo&domain=jdwtgroup.com

**Week 2**

Mongo命令行：

Show dbs

Show collections

db.usercollection.find()

1.流控制

串行流控制

并行流控制