# Nodejs简介

#### Node是lavaScript语言的服务器运行环境。

所谓"运行环境"有两层意思: 首先, JavaScript语言通过Node在服务器运行, 在这个意义上, Node有点像 JavaScript虚拟机, 其次, Node提供大量工具库, 使得JavaScript语言与操作系统互动(比如读写文件、新建子进 程),在这个意义上,Node又是JavaScript的工具库。

Node内部采用Google公司的V8引擎,作为JavaScript语言解释器;通过自行开发的libuv库,调用操作系统资 源。

# 二、Node.js的下载和安装

## 2.1 Node.js下载

#### node.is官网下载

官网会根据你当前的操作系统,提供给你最合适的版本夫下载。

Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient. Node.js' package ecosystem, npm, is the largest ecosystem of open source libraries in the world.

#### Download for Windows (x64)

v6.9.1 LTS **Recommended For Most Users**  v7.1.0 Current **Latest Features** 

Other Downloads | Changelog | API Docs Other Downloads | Changelog | API Docs

Or have a look at the LTS schedule.

### 2.2 安装

下载成功之后是一个msi文件,双击安装即可。安装成功后,相应的环境变量都会自动配置,不需要我们再去 手动配置。



node-v6.9. 1-x64.msi

一路next就可以安装成功。

# 2.3 测试Node.js是否安装成功

安装成功之后,可以在window控制台查看是否安装成功。

```
输入下面的命令查看node的版本。
node -v
直接输入node然后回车,就可以让node去执行我们的js代码了。
node
```

```
Microsoft Windows [版本 6.1.7601]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

C: Wsers Administrator>node
> .exit

C: Wsers Administrator>node -v
v6.2.0

C: Wsers Administrator>
```

# 2.4 使用Node.js运行JavaScript代码

新建一个nodeproject目录,新建一个js文件。01\_hello.js

```
var num1 = 10;
var num2 = 20;
console.log(num1 + num2);
```

在windows控制台中,切换目录到js文件所在目录。然后输入 node 01\_hello.js

```
E 管理员: C:\windows\system32\cmd.exe

F:\test\高级Day15_node\js>node 01_hell.js
30

F:\test\高级Day15_node\js>
```

# 三、Node.js中的一些基本概念澄清

## 3.1 Node.js不是JS应用、而是JS运行平台

看到Node.js这个名字,初学者可能会误以为这是一个Javascript应用,事实上,Node.js采用C++语言编写而成,是一个Javascript的运行环境。

既然不是Javascript应用,为何叫.js呢?因为Node.js是一个Javascript的运行环境。提到Javascript,大家首先想到的是日常使用的浏览器,现代浏览器包含了各种组件,包括渲染引擎、JavaScript引擎等,其中Javascript引擎负责解释执行网页中的Javascript代码。作为Web前端最重要的语言之一,Javascript一直是前端工程师的专利。不过,Node.js是一个后端的Javascript运行环境(支持的系统包括Linux、Windows),这意味着你可以编写系统级或者服务器端的Javascript代码,交给Node.js来解释执行,

## 3.2 Node.js与JavaScript的关系

JavaScript包括3个部分: ECMAScript-262、BOM、DOM。BOM与浏览器相关,DOM和HTML页面相关。Node.js中只是包括了ECMAScript-262。所以我们以前的一些关于BOM的操作和DOM的操作都是基于浏览器端运行的,在Node.js中是无法使用的。

## 3.3 Node.js中几个全局变量

- global:表示Node所在的全局环境,类似于浏览器的window对象。需要注意的是,如果在浏览器中声明一个全局变量,实际上是声明了一个全局对象的属性,比如 var x = 1 等同于设置 window.x = 1,但是Node不是这样,至少在模块中不是这样(REPL环境的行为与浏览器一致)。在模块文件中,声明 var x = 1,该变量不是 global 对象的属性, global.x 等于undefined。这是因为模块的全局变量都是该模块私有的,其他模块无法取到。
- process: 该对象表示Node所处的当前进程,允许开发者与该进程互动。
- console: 指向Node內置的console模块,提供命令行环境中的标准输入、标准输出功能。

## 3.4Node.js中的几个全局函数

- setTimeout(): 用于在指定毫秒之后,运行回调函数。实际的调用间隔,还取决于系统因素。间隔的毫秒数在 1毫秒到2,147,483,647毫秒(约24.8天)之间。如果超过这个范围,会被自动改为1毫秒。该方法返回一个整数,代表这个新建定时器的编号。
- clearTimeout(): 用于终止一个setTimeout方法新建的定时器。
- setInterval(): 用于每隔一定毫秒调用回调函数。由于系统因素,可能无法保证每次调用之间正好间隔指定的毫秒数,但只会多于这个间隔,而不会少于它。指定的毫秒数必须是1到2,147,483,647(大约24.8天)之间的整数,如果超过这个范围,会被自动改为1毫秒。该方法返回一个整数,代表这个新建定时器的编号。
- clearInterval():终止一个用setInterval方法新建的定时器。
- require(): 用于加载模块。
- Buffer(): 用于操作二进制数据。

## 3.5 Node.js的核心模块

如果只是在服务器运行JavaScript代码,用处并不大,因为服务器脚本语言已经有很多种了。Node.js的用处在于,它本身还提供了一系列功能模块,与操作系统互动。这些核心的功能模块,不用安装就可以使用,下面是它们的清单。

- http: 提供HTTP服务器功能。
- url: 解析URL。
- fs: 与文件系统交互。
- querystring: 解析URL的查询字符串。
- child\_process: 新建子进程。
- util: 提供一系列实用小工具。
- path: 处理文件路径。
- crypto: 提供加密和解密功能,基本上是对OpenSSL的包装。

# 三、搭建web应用

使用Node.js搭建web服务器,一般使用一些框架来帮助完成。

express 是一个开源的node.js项目框架,初学者使用express可以快速的搭建一个Web项目,express中已经集成了Web的http服务器创建、请求和文件管理以及Session的处理等功能,所以express是非常适合初学者的入门学习。

## 3.1 安装Express框架

使用node.js自带的包管理器npm安装。

1. 创建一个项目目录,Node\_Hello。进入该目录,创建一个package.json文件,文件内容如下:

```
{
  "name": "Node_Hello",
  "description": "nodejs hello world app",
  "version": "0.0.1",
  "private": true,
  "dependencies": {
      "express": "4.x"
  }
}
```

上面代码定义了项目的名称、描述、版本等,并且指定需要4.0版本以上的Express。

2. <mark>从控制台首先进入刚才的项目目录</mark>,然后输入如下命令,则会开始下载Express。

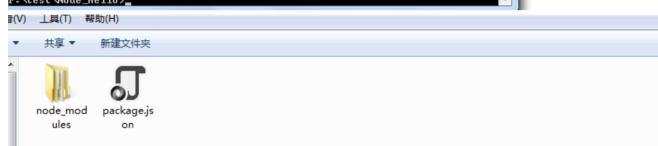
npm install



```
+-- merge-descriptorsQ1.0.1
+-- methodsQ1.1.2
+-- on-finishedQ2.3.0
| '-- ee-firstQ1.1.1
+-- parseurlQ1.3.1
+-- path-to-regexpQ0.1.7
+-- proxy-addrQ1.1.2
| +-- forwardedQ0.1.0
| '-- ipaddr.jsQ1.1.1
+-- qsQ6.2.0
+-- range-parserQ1.2.0
+-- sendQQ.14.1
| +-- destroyQ1.0.4
| +-- http-errorsQ1.5.1
| | +-- inheritsQ2.0.3
| | '-- setprototypeofQ1.0.2
| '-- mimeQ1.3.4
+-- serve-staticQ1.11.1
+-- type-isQ1.6.14
| '-- media-typerQ0.3.0
+-- utils-mergeQ1.0.0
'-- varyQ1.1.0

P:\test\Node_Hello>__

\[ \begin{align*}
\text{P:\test\Node_Hello}_\text{\text{Bb}(H)} \]
```



#### 3.2 创建启动文件

在上面的项目目录下,新建一个启动文件,名字暂叫 index.js 。书写如下代码:

```
var express = require('express');
var app = express();
app.get('/', function (req, res) {
   res.send('<h1>你好, 这是我们的第一个nodejs项目</h1>');
});
app.listen(8080);
```

# 3.3 运行index.js文件

```
node index.js
```

#### 3.4 使用浏览器访问

在浏览器输入下面的地址就可以访问我们刚刚搭建的web网站了。

```
http://127.0.0.1:8080
```