Node.js npm 详解

一、npm简介

安装npm请阅读我之前的文章Hello Node中npm安装那一部分，不过只介绍了linux平台，如果是其它平台，有前辈写了更加详细的介绍。

npm的全称：Node Package Manager.

####（1）通俗的理解

其实从字面意思就可以理解这个产品有什么作用翻译为"Node包管理器"。对，就是Node的包的一个管理工具，目前我尝试的有

1. 下载并安装包（npm install [pkg]）
2. 升级安装包（npm update [pkg]）
3. 卸载安装包（npm uninstall/rm [pkg]），可以指定卸载包的版本号 ...

其实这些命令很简单，常用的必须记住，不常用的查询即可，这才是比较好的学习知识方式。

在终端输入：

```
//查看npm拥有的全部命令
$ npm --help
$ npm help
//查看某一个npm命令的详细用法
$ npm <command> --help
$ npm help <command>
```

####（2）专业的解释

npm（Node Package Manager）是Node.js下的主流套件管理程式。它在Node.js v0.6.x版本之后，内建于Node系统。通过npm可以协助开发者安装、卸载、删除、更新Node.js套件，并且可以通过npm发布自己的插件。

二、类似的产品

其实学习一个产品，可以联系其它产品，能够更好的理解现在手头的产品。第一次学习npm我的第一反应就是，很像linux/mac平台的很多软件，依赖管理的方式可以参考maven...当然相似性可以随便联想。

接下来，举几个例子吧，当然详细了解可以查baidu && google。

1. gem
2. PyPL
3. pear
4. macPort
5. Homebrew
6. rem
7. apt-get
8. yum ...

是不是很多都很熟悉？这样对于npm的认识就不用局限于概念啦。

三、npm基础功能

（1）npmrc文件介绍

首先介绍一下npmrc文件，这个文件是npm包管理器的配置文件。

与npmrc相关的三个文件：

1. 用户配置文件：~/.npmrc
2. 全局配置文件：$PREFIX/npmrc
3. npm内部配置文件：安装npm的目录下

下面仔细看一下npm config的配置。

（2）npm获取配置的6种方式（优先级从高到低）：

1.命令行参数

```
$ --proxy http://<server>:<port>
```

2.环境变量

以"npm*config*"为前缀的环境变量将会被认为是npm的配置属性。 像Maven镜像的概念，方便通信吧。

```
$ npm_config_proxy=http://<server>:<port>
```

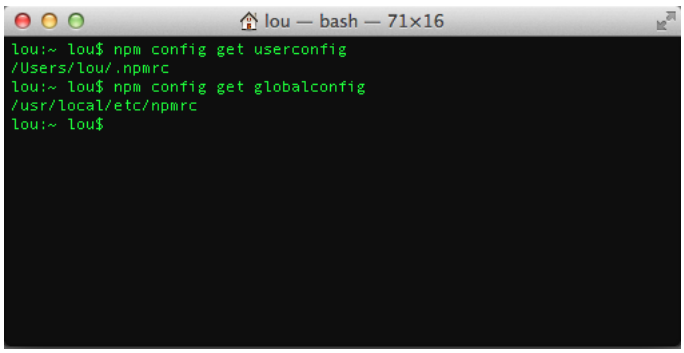
3.用户配置文件

```
//查看文件路径
$ npm config get userconfig

//mac系统默认路径
$HOME/.npmrc
```

4.全局配置文件

```
//查看文件路径
$ npm config get globalconfig

//mac系统默认路径
/usr/local/etc/npmrc
```

> 3，4中输入终端的效果如图：



5.内置配置文件

安装npm的目录下的npmrc文件。

6.默认配置

如果前5条均未设置，npm会使用默认配置参数。

（4）npm install

“安装指定包”：这个命令不难，但是也有需要注意的地方，就是安装的模式有两种，在后面会单独讲解。

> 如果不知道包的具体名称，可以在http://search.npmjs.org上进行搜索。

（5）npm uninstall

“卸载指定包”：在help的时候，会给你推荐npm rm 这个命令，uninstall会卸载掉包的依赖，rm。

（6）npm ls

查看安装的包清单，其实和linux的ls命令很像，可以跟很多参数，详情可以使用

```
$ npm help ls
```

（7）npm search

搜索包的详细信息，比如我们搜索express试试。第一次搜索，会提示建立索引，需要耐心等待片刻，大家测试的时候不要就关掉啦终端。

```
npm WARN Building the local index for the first time, please be patient
```

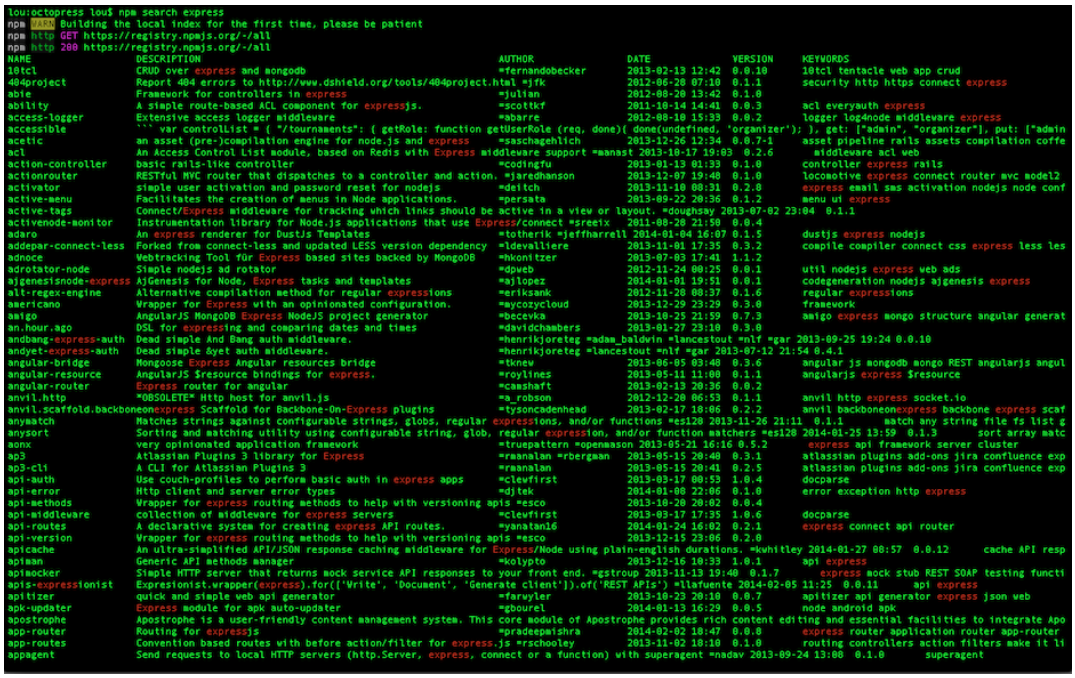

2017年4月 (112)
2017年3月 (97)
2017年2月 (41)
2017年1月 (71)
2016年12月 (74)
2016年11月 (84)
2016年10月 (130)
2016年9月 (133)
2016年8月 (58)
2016年7月 (1)
2016年6月 (1)
2016年3月 (1)

文章档案
2016年8月 (1)

积分与排名
积分 - 540915
排名 - 225

最新评论
1. Re:jQuery插件开发
感觉写的挺好 怎么没人

2. Re:理解JS中的call、
法(**********
****
**********)
有很多不严谨的说法比如
行后，当前的this指向了
前词法作用域的的this,t
s2的对象应该这样描述:
调用call方法.....



其实看上去复杂，只是东西有点大，不过主要包含以下6个部分：

1. 名称
2. 描述
3. 作者
4. 发布时间
5. 发布版本号
6. 关键字

####（8）npm update

更新安装的包

更多API可以查看官网：https://npmjs.org/doc/

四、版本号的知识。

在node.js中的package.json配置文件中，我们需要配置版本号，比如0.1.2

第一位数字：主版本号

第二位数字：子版本号

第三位数字：补丁版本号

找到一个不错的介绍软件项目版本号的文章

软件项目版本号的命名规则及格式

为什么要解释这个呢？肯定是有用，因为npm安装的时候是可以选择版本号的，有点理解会比较好吧，至少我是这么认为的。

一、前言

很久之前就想系统的学习nodejs技术了，但是由于很多事情，忙不太过来，今天晚上下定决心要入门这门技术，所以一口气看完了《Node.js开发指南》和《Node.js中文文档》，总算是真正入门了，接下来就总结一部分，剩下的只有明天再总结了。

二、Node.js简介

一句话简单说明一下node.js是什么东西。

> Node.js 是一个让 JavaScript 运行在服务端的开发平台。

三、Node.js的安装

学习了Node.js，觉得如果在window下学习这门技术的话还太成熟，第三方的支持不太好，所以只介绍linux或者mac上面的安装。node.js在window上面的安装直接下载安装包，一直下一步就可以装好，环境变量也会自动配置好，npm（node package manage）在新版本也会相应的装上。

对于mac环境安装也比较简单，介绍两种安装方式。

1.homebrew（注意一直使用admin权限吧，大多都是权限文件夹，可以在终端开始使用sudo -s命令来一直使用admin权限）

3. Re:理解JS中的call、法(**********
****
**********)

楼主是大神啊

4. Re:vue-router如何
给不同的权限

实现原理和步骤都不说下
是啥

5. Re:微信内嵌H5网页
效

楼主是解锁后请求一次?
过段时间在切回来，有效

阅读排行榜

1. vue axios全攻略(55

2. 推荐几款屏幕录制工
(30078)

3. CSS3 鲜为人知的属
ighlight-color的理解(2

4. JS 对象封装的常用方

5. 使用Atom打造无懈可
编辑器(20295)

评论排行榜

1. 彻底理解Javascript
(----------
----

2. 一道前端面试题？求

3. Javascript中的原型
__proto__的关系(8)

4. 理解JS中的call、app
****
****
********)(7)

```
$ sudo brew install node
```

这样下载完了就算安装上了，我们可以来检测一下你的安装（软件配置必须要的步骤）。

```
# 查看node版本号
$ node -v
# 查看npm版本号和清单
$ npm -v && npm list
```

如果npm没有安装上，那么可以使用以下命令安装。

```
$ sudo curl http://npmjs.org/install.sh | sh
```

2.使用源码编译的方式（其实有方便的尽量就避免麻烦的，嘿嘿）

> 下载地址：http://nodejs.org

执行典型的安装命令即可（注意到/bin目录，或者有configure文件的目录）。

```
$ ./configure && make && sudo make install
```

四、Node.js的多版本管理器n

> 下载地址：https://github.com/visionmedia/n

如上执行源码安装命令即可。

当然，如果你安装了npm，就有更方便的方式，这也是很迷人的地方。

```
$ sudo npm install -g n
```

接下来检查一下安装情况。

```
$ n --version
```

这样就可以使用自由切换使用node的版本了。用如下命令安装指定版本的node。

```
$ n version
```

下载后，会默认下载到：/usr/local/n/versions/目录

如果你安装后，再使用"$ n version"命令就是指定使用的默认版本号，也可以使用"$ n use version xxx.js"来暂时使用某个node版本执行xxx.js文件。

五、万事具备，只欠Hello Node.js

每个语言入门都可以写一个hello xxx，示好。

在node中，这个很简单，只需要进入REPL模式，暂时不要管这个模式是什么，我们先使用。这个用截图来直观说明一下。

进入REPL模式的命令：

```
$ node
```

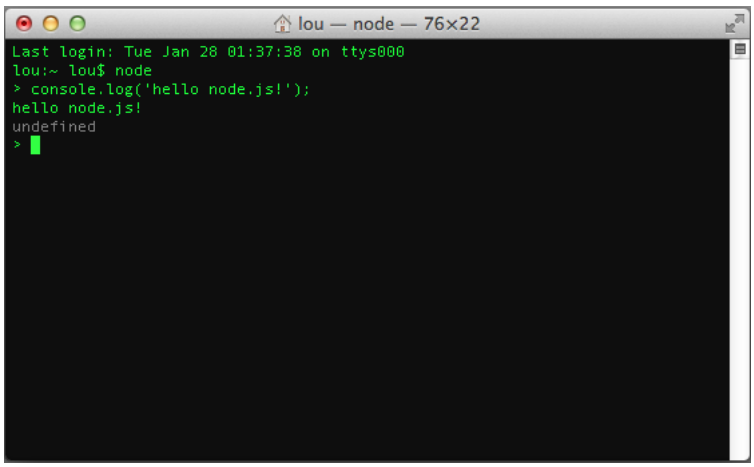
接下来直接输入：

```
console.log("hello node.js!");
```

就可以在终端打印出：

```
>hello node.js!
>undefined
```

写过js的应该对最后会输出undefined并不吃惊，先可以不管。



5. vue axios全攻略(7)

推荐排行榜

1. 一道前端面试题？求
2. 理解JS中的call、app
****
****
********)(14)
3. vue-cli中的webpacl
4. Javascript中的原型
__proto__的关系(8)
5. canvas仿芝麻信用分

下面解释一下REPL模式：

其实这个模式不用解释，经过上面的例子应该就有直观的认识啦，类似mysql、python等的shell交互的方式，可以让你输入马上就得到反馈，输出结果到屏幕上面。在node中可以连续按两次ctrl+c退出（但是python要使用exit()函数，很不舒服）。

上面是一种最简单的方式，还是介绍一种正规的方式吧，既然是js的运行平台，我们就写一个js文件：hello.js，内容很简单，如下：

```
console.log("hello node.js!");
console.log("%d\n", 60);
```

在终端执行（注意要进入hello.js所在目录哦）：

```
$ node hello.js
```

就可以在终端直接输出结果了。

> 为什么我加入了"console.log('%d\n', 60);"这一句呢？那是因为这可以测试出其实可以用c语言printf的方式来写console.log()。

六、Node.js终于来到了浏览器

前面的例子都是在控制台里面的，没什么意思，我们接下来写一个我们典型的B/S访问的方式。当然这就离不开HTTP了，查看了一下node的源码，新版本默认是http 1.1，支持http和https这两种协议。

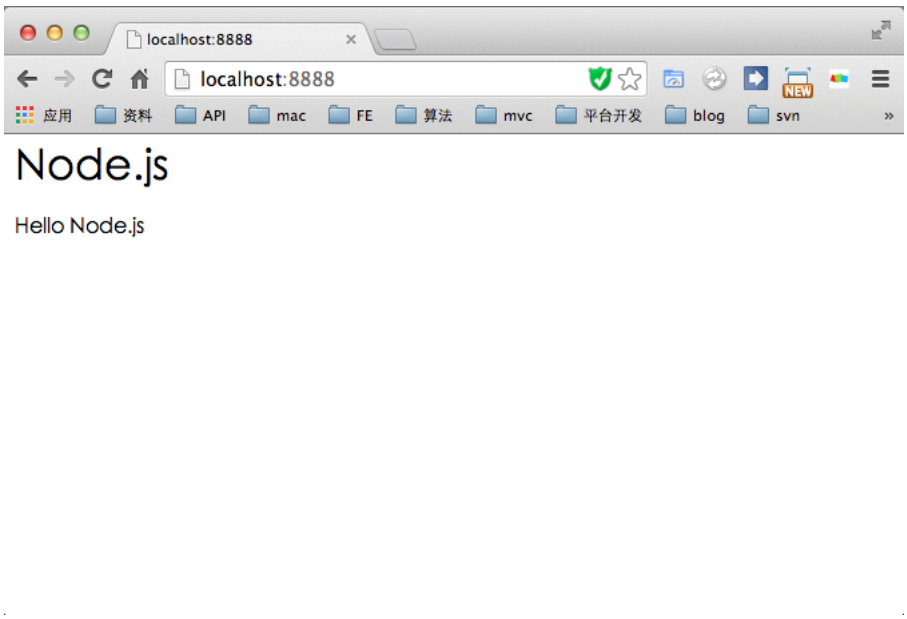
我们先建一个http.js文件，里面的内容对于后台的开发人员就很熟悉了。

```
var http = require("http");
http.createServer(function(req, res) {
    res.writeHead(200, {"Content-Type": "text/html"});
    res.write("<h1>Node.js</h1>");
    res.end("<p>Hello Node.js</p>");
}).listen(8888);
console.log("HTTP server is listening at port 8888.");
```

以上代码监听的port：8888，这其实就是建立了一个request，我们在浏览器地址栏中输入：http://localhost:8888/ 即可以访问我们打印的内容。

我们先在终端运行后台服务：

```
$ node http.js
```



我们来分析一下这个简单的程序，其实从require的加载方式我们并不陌生，这是现在模块加载在前端普遍使用的函数名，那么为什么我们的js代码就能够这样来请求和访问呢？我们来简单的看看源码的解析。

1. 首先我们先看看源码的http.js文件。

```
var Server = exports.Server = server.Server;
function createServer() { return new Server(requestListener) };
```

在这个文件中，我们知道createServer()方法其实返回的就是Server类，这个类是由server这个实例来的。

2. 接下来我们再追踪一下_http_server.js文件。

这个要问怎么找到这些文件的，其实是凭直觉和猜测，再加上模块化的调试方法，可以找到http.js文件的加载模块，就可以找到相应的文件了。

```
function requestListener() {};
function connectionListener(socket) {}
Server.prototype.setTimeout = function(msecs, callback) {};
```

再这个文件中，我们可以看出和Socket原理差不多，使用定时器来实现端口的监听，但是在源码中可以找到主定时器，这个概念来源于游戏开发，在全局设置定时器，这样大大提升了性能，不得不佩服作者的才能。

3. 那么我们node底层是c/c++来实现的，我们可以追踪一下node的启动。

我们先找到c语言的main文件，后来发现好多node_开头的，大楷就是系统和行一点的文件了，在node_main.cc文件中发现了：node::Start(argc, argv);函数，但是找不到具体实现，那么就肯定是在依赖文件中，通过头文件的声明，找到了，其实实现位于node.cc文件。

4. node.cc文件来看node启动流程。

函数大致的执行顺序为：

```
int Start(int argc, char** argv)
V8::Initialize();
    CreateEnvironment() //创建当前环境
    SetupProcessObject() //启动当前环境的进程
    Load() //加载当前环境
    context() //引用上下文
    uv_run //开始异步事件运行
    RunAtExit //删除异步事件链表
```

上面可以看出其实就是通过一系列初始化，最后达到平台的建立。

5. module的加载实质

其实我们经过上面的分析，我们大致知道的node的构成，那么我们能够在node.js（猜想：这个js文件是c和js沟通的桥梁）中，发现传入了process对象。源码结构如下：

```
(function(process) {});
//本地模块的加载接口
NativeModule.require();
```

我们来看一看根据process来绑定c语言的模块。

```
var HTTPParser = process.binding("http_parser").HTTPParser;
```

是不是就很清晰啦，接下来我们看看模块是怎么注册的。

```
//现在产生的插件会放置在node_module文件夹下应该很清晰了
//定义了一个指针结构
node_module_struct* mod
//注册自己的模块
mod->register_func(Handle<Object> target)
//为注册的模块设置一个上下文
mod->register_context_func
//最后加载到模块缓存，这里也是实现延迟加载的本质
cache->Set(module, exports)
```

在模块注册后，还需要做一些准备工作才能够真正的加载到我们的js文件。

```
//node.cc文件中执行绑定
static void Binding(const FunctionCallbackInfo<Value>& args);
//node_javascript.cc文件中会定义你的js
void DefineJavaScript(Handle<Object> target);
```

6. 在module.js文件中，得到真正的js文件的加载，平台调用。

在这里作者的module数据结构设置得比较简单，抛去其它得留下结构如下：

```
module: {
    id: "",
    parent: [],
    export: {},
    filename,
    children: []
}
```

这样我们这个http程序就能够很好得解释啦。其实知道了原理后面得知识就是看看api和写法了。太晚啦，睡觉觉啦 =_=...后续...

安装Node和npm前半部分的配置可以参考之前我的两篇文章：

1. Hello Node
2. Node npm 详解（1）

四、本地模式和全局模式

如果你了解环境变量里面的，用户变量和系统变量。可以做一个类比进行理解。当然，windows上面的环境变量概念比较好理解。

1. 本地模式

本地模式下安装包的特点

- 不会写入PATH变量（也就是环境变量，无法在全局引用该安装包，不能在终端直接使用）
- 能够在不同的node_modules目录，安装不同版本的安装包
- 能够通过require()来引入安装包

使用"npm install [@]"安装的包，默认会安装在当前目录的"node_modules"目录下（如果没有该目录，在执行命令的时候，会自动帮你创建）。

```
//专业的写法
./node_modules
```

（1）默认采用本地模式安装

```
npm install <pkg>
```

（2）信息写入package.json文件

```
npm install <pkg> --save
```

这个命令在安装包的同时，将信息写入package.json。

> @version表示指定安装包的版本号，是可选项目，默认安装最新版本。
> 项目路径中如果有package.json文件，使用npm install方法就可以根据dependencies配置安装所有的依赖包。
> 如果这样配置，当代码提交到github时，就不用提交node_modules这个文件夹。

2. 全局模式

全局模式安装包的特点

- 不需要重复安装
- 不能使用require()引入
- 会写入PATH，并建立软链接，使用命令行的方式使用
- 不方便指定特定的版本运行

（1）采用全局模式安装

```
npm install -g <pkg>
```

（3）在mac中全局的目录

```
//安装包所在目录
/usr/local/lib/node_modules/
//运行命令的软链接所在目录
/usr/local/bin
```

（4）查看安装包路径

```
//查看当前包的安装路径
npm root
//查看全局的包的安装路径
npm root -g
```

（5）设置全局模式安装目录

```
//设置后，以全局模式将会安装在此目录中，不过需要手动加入PATH，切记
npm config set prefix <global dir>
//设置npm缓存文件的存放路径
npm config set cache <cache dir>
```

（6）查看默认模式

```
//默认返回：false
$ npm get global
$ npm config get global
```

（7）设置为默认以全局模式安装，就不用每次加"-g"参数啦。

```
$ npm set global=true
$ npm config set global=true
```

> npm set / npm config set与npm get / npm config get的区别和联系单独写吧。其实不难，只是需要实验才能得出结果，这里区别很细节。
> 准备把文章拆分成几篇，写得详细了一点，这里写的话篇幅就太长了。

最骚的就是你
关注 - 12
粉丝 - 324
+加关注

0 0

« 上一篇：自己动手写一个前端路由插件
» 下一篇：认识Webpack

posted @ 2016-09-25 22:08 最骚的就是你 阅读(9461) 评论(0) 编辑 收藏