

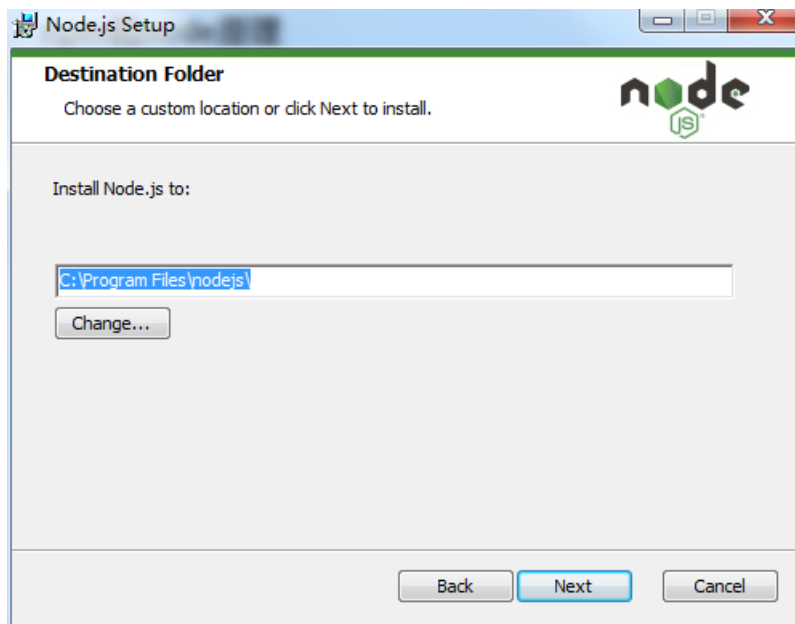
# npm以及nvm原理

## 前言

环境变量PATH（PATH环境变量是表示可执行文件的搜索路径），在命令行某个命令的时候,系统会到PATH对应的路径下查找可执行文件执行。

## npm和node原理

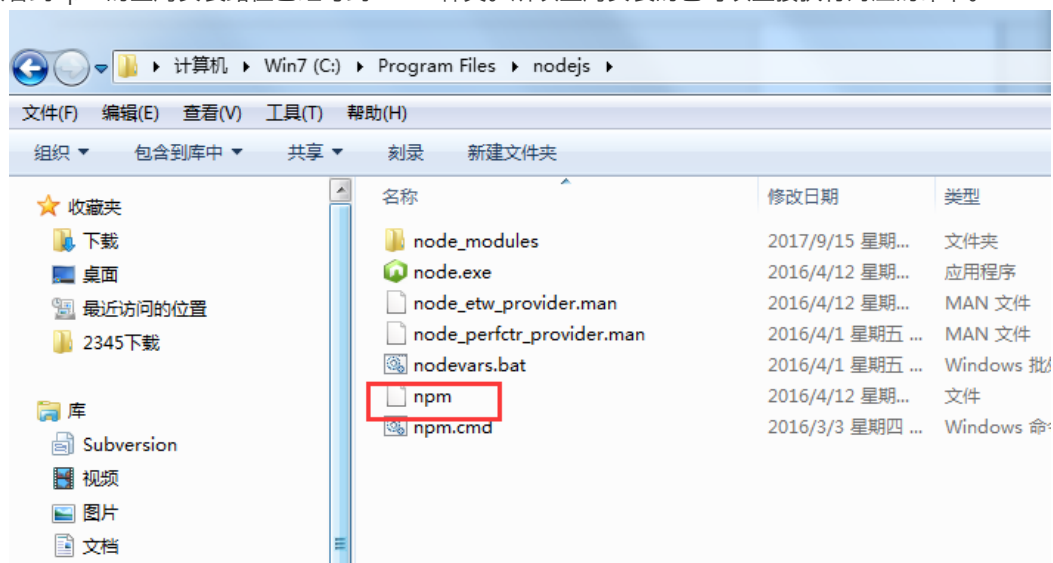
安装node的时候有一个路径配置



表示node的安装路径是C:\Program Files\nodejs\，一路默认安装，node内置了npm.

**PATH:** C:\Program Files\nodejs\; C:\Users\Administrator\AppData\Roaming\npm;

可以看到npm的全局安装路径已经写到PATH种类。所以全局安装的包可以直接执行对应的命令。



在命令行运行npm实际是执行C:\Program Files\nodejs\npm

```

Administrator@PC-20160412RGPC MINGW64 /d/workspace_preRelease/h5
$ '/c/Program Files/nodejs/npm'

Usage: npm <command>

where <command> is one of:
  access, add-user, adduser, apihelp, author, bin, bugs, c,
  cache, completion, config, ddp, dedupe, deprecate, dist-tag,
  dist-tags, docs, edit, explore, faq, find, find-dupes, get,
  help, help-search, home, i, info, init, install, issues, la,
  link, list, ll, ln, login, logout, ls, outdated, owner,
  pack, ping, prefix, prune, publish, r, rb, rebuild, remove,
  repo, restart, rm, root, run-script, s, se, search, set,
  show, shrinkwrap, star, stars, start, stop, t, tag, team,
  test, tst, un, uninstall, unlink, unpublish, unstar, up,
  update, upgrade, v, version, view, whoami

npm <cmd> -h      quick help on <cmd>
npm -l            display full usage info
npm faq           commonly asked questions
npm help <term>   search for help on <term>
npm help npm      involved overview

Specify configs in the ini-formatted file:
  C:\Users\Administrator\.npmrc
or on the command line via: npm <command> --key value
Config info can be viewed via: npm help config

npm@2.15.1 C:\Program Files\nodejs\node_modules\npm
Administrator@PC-20160412RGPC MINGW64 /d/workspace_preRelease/h5
$ '/c/Program Files/nodejs/npm'

```

这个npm最终执行的文件是使用node运行'C:\Program Files\nodejs\node\_modules\npm\bin\npm-cli.js'文件，可以做一个实验，我们将'C:\Program Files\nodejs\node\_modules\npm'文件改一个名称，比如改成'npm1'，会发现执行npm报错

```

Administrator@PC-20160412RGPC MINGW64 /d/workspace_preRelease/h5
$ npm
module.js:327
  throw err;
  ^

Error: Cannot find module 'C:\Program Files\nodejs\node_modules\npm\bin\npm-cli.js'
    at Function.Module._resolveFilename (module.js:325:15)
    at Function.Module._load (module.js:276:25)
    at Function.Module.runMain (module.js:441:10)
    at startup (node.js:139:18)
    at node.js:968:3
module.js:327
  throw err;
  ^

Error: Cannot find module 'C:\Program Files\nodejs\node_modules\npm\bin\npm-cli.js'
    at Function.Module._resolveFilename (module.js:325:15)
    at Function.Module._load (module.js:276:25)
    at Function.Module.runMain (module.js:441:10)
    at startup (node.js:139:18)
    at node.js:968:3
Administrator@PC-20160412RGPC MINGW64 /d/workspace_preRelease/h5
$

```

可以看一下npm的配置：npm config list

```

Administrator@PC-20160412RGPC MINGW64 /d/workspace_preRelease/h5
$ npm config list
; cli configs
user-agent = "npm/2.15.1 node/v4.4.3 win32 x64"

; builtin config undefined
prefix = "C:\\Users\\Administrator\\AppData\\Roaming\\npm"

; node bin location = C:\Program Files\nodejs\node.exe
; cwd = D:\workspace_preRelease\h5
; HOME = C:\Users\Administrator
; 'npm config ls -l' to show all defaults.

Administrator@PC-20160412RGPC MINGW64 /d/workspace_preRelease/h5
$

```

使用npm config -h查看配置命令帮助

其中prefix就是表示全局安装路径，可以使用来设置

```
config set <key> <value>
```

还可以通过npm config ls -l或npm config list -l命令查看所有的配置项。

npm有本地安装和全局安装两种，

## 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

## 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

//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
```

以我的机器的fis3为例。

全局安装路径是C:\Users\Administrator\AppData\Roaming\npm

我要运行本地的h5 fis3打包，本地路径是：D:\workspace\_preRelease\h5

```
Administrator@PC-20160412RGPC MINGW64 /D:/workspace_preRelease/h5
$ npm root
D:\workspace_preRelease\h5\node_modules

Administrator@PC-20160412RGPC MINGW64 /D:/workspace_preRelease/h5
$ npm root -g
C:\Users\Administrator\AppData\Roaming\npm\node_modules

Administrator@PC-20160412RGPC MINGW64 /D:/workspace_preRelease/h5
$
```

安装cnpm淘宝的npm镜像，可以加快npm包的安装

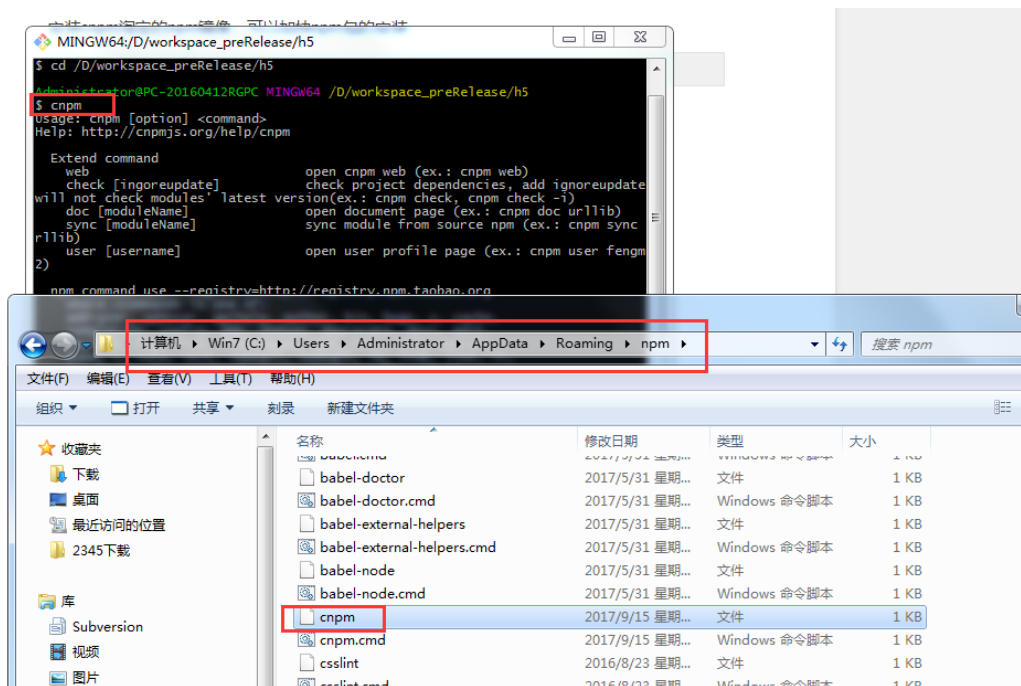
```
npm install -g cnpm --registry=https://registry.npm.taobao.org
```

需要注意的是目前cnpm需要node 6.11.2以上版本，详细查看<https://npm.taobao.org/>

同理，之所以可以执行cnpm命令，实际是执行

的C:\Users\Administrator\AppData\Roaming\npm\cnpm

npm全局安装以后之所以能识别全局安装后的包对应的命令是因为npm将全局安装路径写入了PATH，在执行npm安装的全局命令时系统会到PATH对应的路径下查找可执行文件执行。



当然真正执行是

C:\Users\Administrator\AppData\Roaming\npm\node\_modules\cnpm\bin\cnpm(下面这个测试是将C:\Users\Administrator\AppData\Roaming\npm\node\_modules\cnpm更改一下名称再执行cnpm)

```
Administrator@PC-20160412RGP MINGW64 /D:/workspace_preRelease/h5
$ cnpm
module.js:472
  throw err;
  ^
Error: Cannot find module 'C:\Users\Administrator\AppData\Roaming\npm\node_modules\cnpm\bin\cnpm'
    at Function.Module._resolveFilename (module.js:470:13)
    at Function.Module._load (module.js:418:25)
    at Module.runMain (module.js:605:10)
    at run (bootstrap_node.js:427:7)
    at startup (bootstrap_node.js:151:9)
    at bootstrap_node.js:542:3
Administrator@PC-20160412RGP MINGW64 /D:/workspace_preRelease/h5
```

使用 `npm search <module>` 命令查看安装包信息，比如 `npm search express`

```
Administrator@PC-20160412RGP MINGW64 /D:/workspace_preRelease/h5
$ npm search express
NAME | DESCRIPTION | AUTHOR | DATE | VERSION | KEYWORDS
express | Fast, ... | =hacksparrow... | 2017-08-07 | | express framework sinatra web rest restf
ul router app api
path-to-regexp | Express style path... | =dougwilson... | 2017-08-23 | | express regexp route routing
morgan | HTTP request logger... | =dougwilson... | 2017-08-24 | | express http logger middleware
express-session | Simple session... | =defunctzombie... | 2017-08-03 | |
cors | Node.js CORS... | =dougwilson... | 2017-07-13 | | cors express connect middleware
serve-favicon | favicon serving... | =dougwilson... | 2017-09-12 | | express favicon middleware
webpack-hot-middleware | Webpack hot... | =glenjamin... | 2017-09-07 | | webpack hmr hot module reloading hot-rel
loading middleware express
express-validator | Express middleware... | =gustavohenke... | 2017-09-02 | | express validator validation validate sa
nitize sanitization xss
babel-helper-optimize-call-expression | Helper function to... | =jmm... | 2017-04-07 | |
babel-plugin-syntax-do-expressions | Allow parsing of do... | =thejameskyle... | 2016-08-04 | | babel-plugin
csurf | CSRF token... | =defunctzombie... | 2016-05-27 | | csrf tokens middleware express
babel-plugin-transform-do-expressions | Compile do... | =thejameskyle... | 2017-01-20 | | babel-plugin
express-graphql | Production ready... | =wincent... | 2017-08-29 | | express restify connect http graphql mid
dware api
babel-helper-explode-assignable-expression | Helper function to... | =thejameskyle... | 2017-04-07 | |
helmet | help secure... | =adam_baldwin... | 2017-07-28 | | security headers express connect x-frame
-options x-powered-by csp hsts clickjack
escape-string-regexp | Escape RegExp... | =sindresorhus... | 2016-02-21 | | escape regex regexp re regular expressio
n string str special characters
```

## NVM原理

卸载node。

注：单独卸载node，npm的路径还在PATH环境变量中，只不过npm中安装的全局模块不能再访问了

```
MINGW64:/d/workspace_preRelease/h5
Administrator@PC-20160412RGP MINGW64 /d/workspace_preRelease/h5
$ cnpm
/c/Users/Administrator/AppData/Roaming/npm/cnpm: line 12: node: command not found
d
Administrator@PC-20160412RGP MINGW64 /d/workspace_preRelease/h5
$ npm
bash: npm: command not found
Administrator@PC-20160412RGP MINGW64 /d/workspace_preRelease/h5
$ |
```

安装nvm,一路默认路径以及环境变量如下

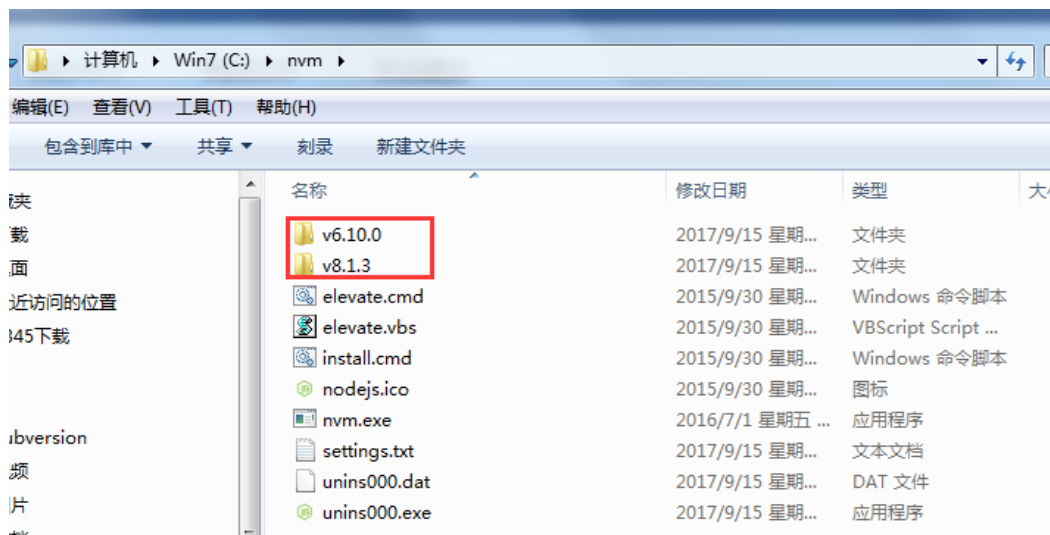
环境变量：

NVM\_HOME: C:\nvm

NVM\_SYMLINK: C:\Program Files\nodejs

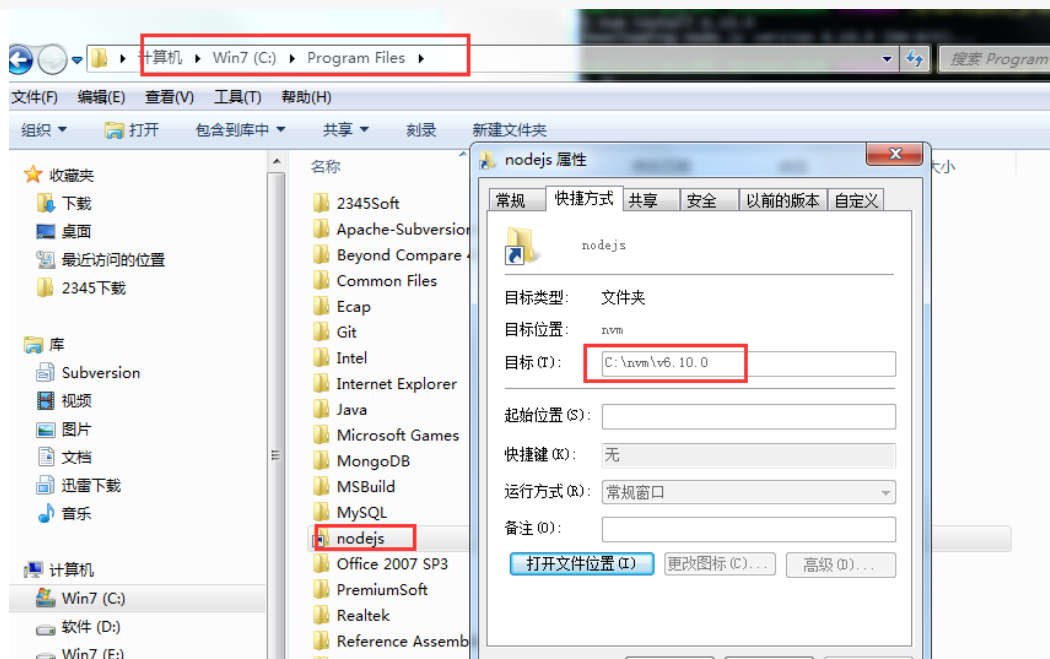
PATH: %NVM\_HOME%;%NVM\_SYMLINK%;

其中所有使用nvm安装的node版本都在C:\nvm下面能看到，比如现在我就安装了6.10.0和8.1.3两个版本

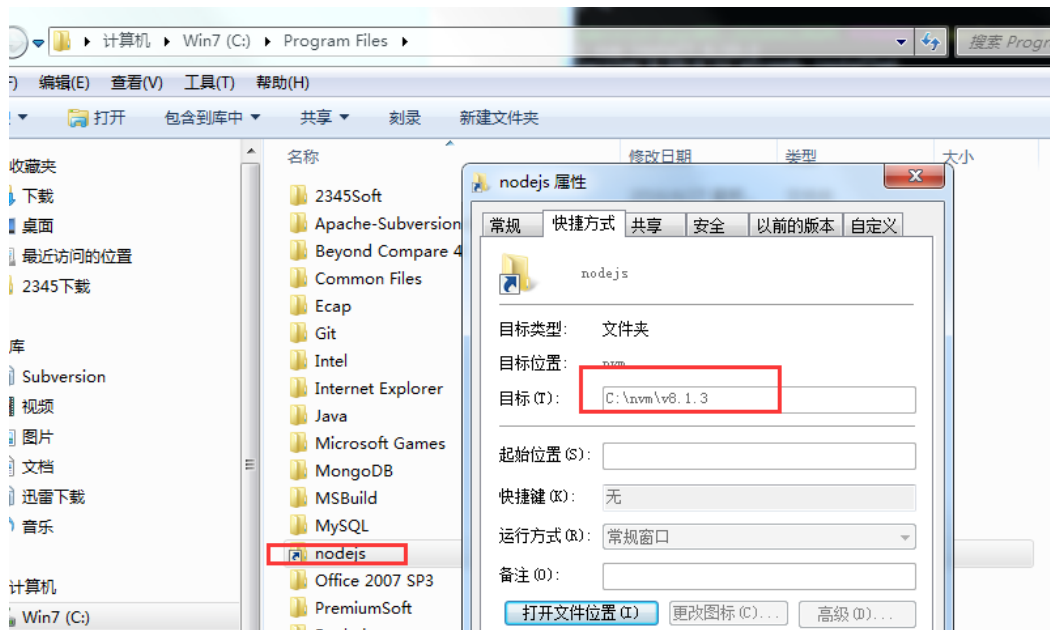


到现在为止C:\Program Files\nodejs目录都是不存在的，当启用某一个node版本时，才会创建这个文件，并将该文件映射到C:\nvm文件下对应的版本文件夹

```
$ nvm use 6.10.0
```



然后切换node版本到8.1.3



所以明白nvm切换的原理了吧。

查看npm的全局安装路径

```
Administrator@PC-20160412RGPC MINGW64 /d/workspace_preRelease/h5
$ npm config get prefix
C:\Program Files\nodejs

Administrator@PC-20160412RGPC MINGW64 /d/workspace_preRelease/h5
$
```

所以每个版本的node有自己独立的全局安装环境：**C:\nvm\<version>** 比如8.1.3版本的全局安装路径是C:\nvm\8.1.3

## 全局安装fis-parser-node-sass-nfd，但是fis3打包找不到模块

以nvm切换到node版本8.1.3为例。

之前的npm全局安装路径没有被删除：C:\Users\Administrator\AppData\Roaming\npm

所以之前全局安装fis3命令依然可用，但是现在的全局安装路径是C:\nvm\8.1.3

fis3真正的执行文件C:\Users\Administrator\AppData\Roaming\npm\node\_modules\fis3\bin\fis.js  
中查找模块有一段代码

```
process.title = this.name + ' ' + process.argv.slice(2).join(' ') + ' [' + env.cwd + '];

// 配置插件查找路径，优先查找本地项目里面的 node_modules
// 然后才是全局环境下面安装的 fis3 目录里面的 node_modules
fis.require.paths.unshift(path.join(env.cwd, 'node_modules'));
fis.require.paths.push(path.join(path.dirname(__dirname), 'node_modules'));
fis.cli.name = this.name;
fis.cli.run(argv, env);
});
```

第二句，fis将将\_\_dirname下的node\_modules文件夹作为查找模块的位置。而fis.js  
在C:\Users\Administrator\AppData\Roaming\npm下，不在现在的全局安装路径下

C:\nvm\8.1.3，so,全局安装的**fis-parser-node-sass-nfd**木有用。解决办法有三个：

- 1.在C:\Users\Administrator\AppData\Roaming\npm安装该模块
- 2.在全局安装路径下重新安装一遍fis3
- 3.本地安装该模块