nvm，nodejs和npm安装使用

summer

目　录

nvm和nodejs安装使用

Node.js下载
Node.js中文网

快速搭建 Node.js 开发环境
如果你想长期做 node 开发, 或者想快速更新 node 版本, 或者想快速切换 node 版本,
那么在非 Windows(如 osx, linux) 环境下, 请使用 nvm 来安装你的 node 开发环境, 保持系统的干净.
如果你使用 Windows 做开发, 那么你可以使用 nvmw 来替代 nvm

osx, linux 环境
如果你是 windows 环境开发, 请跳过这里, 直接查看下一章.

git clone nvm
直接从 github clone nvm 到本地, 这里假设大家都使用 ~/git 目录存放 git 项目:

```
$ cd ~/git
$ git clone https://github.com/creationix/nvm.git
```

配置终端启动时自动执行 source ~/git/nvm/nvm.sh,
在 ~/.bashrc, ~/.bash_profile, ~/.profile, 或者 ~/.zshrc 文件添加以下命令:

```
source ~/git/nvm/nvm.sh
```

重新打开你的终端, 输入 nvm

```
$ nvm

Node Version Manager

Usage:
    nvm help                    Show this message
    nvm --version               Print out the latest released version of nvm
    nvm install [-s] <version>  Download and install a <version>, [-s] from source
    nvm uninstall <version>     Uninstall a version
    nvm use <version>           Modify PATH to use <version>
    nvm run <version> [<args>]  Run <version> with <args> as arguments
    nvm current                 Display currently activated version
    nvm ls                      List installed versions
    nvm ls <version>            List versions matching a given description
    nvm ls-remote               List remote versions available for install
    nvm deactivate              Undo effects of NVM on current shell
    nvm alias [<pattern>]       Show all aliases beginning with <pattern>
    nvm alias <name> <version>  Set an alias named <name> pointing to <version>
    nvm unalias <name>          Deletes the alias named <name>
    nvm copy-packages <version> Install global NPM packages contained in <version> to c
urrent version
```

```
Example:
    nvm install v0.10.24        Install a specific version number
    nvm use 0.10                Use the latest available 0.10.x release
    nvm run 0.10.24 myApp.js    Run myApp.js using node v0.10.24
    nvm alias default 0.10.24   Set default node version on a shell

Note:
    to remove, delete or uninstall nvm - just remove ~/.nvm, ~/.npm and ~/.bower folders
```

通过 nvm 安装任意版本的 node

nvm 默认是从 http://nodejs.org/dist/ 下载的, 国外服务器, 必然很慢,

好在 nvm 以及支持从镜像服务器下载包, 于是我们可以方便地从七牛的 node dist 镜像下载:

```
$ NVM_NODEJS_ORG_MIRROR=https://npm.taobao.org/mirrors/node nvm install 4
```

于是你就会看到一段非常快速进度条:

```
######################################################################## 100.0%
Now using node v4.3.2
```

如果你不想每次都输入环境变量 NVM_NODEJS_ORG_MIRROR, 那么我建议你加入到 .bashrc 文件中:

```
# nvm
export NVM_NODEJS_ORG_MIRROR=https://npm.taobao.org/mirrors/node
source ~/git/nvm/nvm.sh
```

然后你可以继续非常方便地安装各个版本的 node 了, 你可以查看一下你当前已经安装的版本:

```
$ nvm ls
         nvm
     v0.8.26
    v0.10.26
    v0.11.11
->  v4.3.2
```

windows 环境

多版本nodejs管理工具nvm for windows

通过 nvm 安装任意版本的 node

nvm 默认是从 http://nodejs.org/dist/ 下载的, 国外服务器, 必然很慢,

npm也是从国外服务器下载的，基本下载不下来

好在 nvm 以及支持从镜像服务器下载包, 于是我们可以方便地从淘宝的 node dist 镜像下载:

```
set "NVMW_NODEJS_ORG_MIRROR=https://npm.taobao.org/mirrors/node"
set "NVMW_NPMJS_COM_MIRROR=https://npm.taobao.org/mirrors/npm"
nvm install 4.3.2
```

如果你不想每次都输入环境变量 NVMW_NODEJS_ORG_MIRROR, 那么我建议你在全局环境变量中增加它.

然后你可以继续非常方便地安装各个版本的 node 了, 你可以查看一下你当前已经安装的版本:

```
$ nvm list

    5.8.0
    5.4.0
  * 5.10.1 (Currently using 64-bit executable)
    5.0.0
```

设置npm的获取地址
`npm config set registry https://registry.npm.taobao.org`

使用 cnpm 加速 npm

如果想升级npm自身，则会遇到一点问题，因为nodejs附带了npm，因此无法全局升级npm，需要在nodejs的安装目录下局部升级npm：

```
D:
cd "nodejs"
npm update npm --registry=https://registry.npm.taobao.org
```

npm使用国内镜像安装cnpm,可以安装任何包

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

同理 nvm , npm 默认是从国外的源获取和下载包信息, 不慢才奇怪.
可以通过简单的 ---registry 参数, 使用国内的镜像 https://registry.npm.taobao.org :

```
$ npm --registry=https://registry.npm.taobao.org install koa
```

于是屏幕又哗啦哗啦地一大片输出:

```
$ npm --registry=https://registry.npm.taobao.org install koa
npm http GET https://registry.npm.taobao.org/koa
npm http 200 https://registry.npm.taobao.org/koa
...
npm http 200 https://registry.npm.taobao.org/negotiator
npm http 200 https://registry.npm.taobao.org/keygrip
koa[@0](/user/0).5.2 node_modules/koa
├── koa-compose[@2](/user/2).2.0
├── statuses[@1](/user/1).0.2
├── finished[@1](/user/1).1.1
├── escape-html[@1](/user/1).0.1
├── only[@0](/user/0).0.2
├── debug[@0](/user/0).8.0
├── fresh[@0](/user/0).2.2
├── type-is[@1](/user/1).0.1
```

```
├── delegates[@0](/user/0).0.3
├── mime[@1](/user/1).2.11
├── co[@3](/user/3).0.5
├── accepts[@1](/user/1).0.1 (negotiator[@0](/user/0).4.2)
└── cookies[@0](/user/0).4.0 (keygrip[@1](/user/1).0.0)
```

但是毕竟镜像跟官方的 npm 源还是会有一个同步时间差异, 目前 cnpm 的默认同步时间间隔是 15 分钟.
如果你是模块发布者, 或者你想马上同步一个模块, 那么推荐你安装 cnpm cli:

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

通过 cnpm 命令行, 你可以快速同步任意模块:

```
$ cnpm sync koa connect mocha
```

呃, 我就是不想安装 cnpm cli 怎么办? 哈哈, 早就想到你会这么懒了, 于是我们还有一个 web 页面:

例如我想马上同步 koa, 直接打开浏览器: https://npm.taobao.org/sync/koa

或者你是命令行控, 通过 open 命令打开:

```
$ open https://npm.taobao.org/sync/koa
```

如果你安装的模块依赖了 C++ 模块, 需要编译, 肯定会通过 node-gyp 来编译,
node-gyp 在第一次编译的时候, 需要依赖 node 源代码, 于是又会去 node dist 下载,
于是大家又会吐槽, 怎么 npm 安装这么慢...

好吧, 于是又要提到 --disturl 参数, 通过七牛的镜像来下载:

```
$ npm --registry=https://registry.npm.taobao.org --disturl=https://npm.taobao.org/mirro
rs/node install microtime
```

再次要提到 cnpm cli, 它已经默认将 --registry 和 --disturl 都配置好了, 谁用谁知道
.
写到这里, 就更快疑惑那些不想安装 cnpm cli 又吐槽 npm 慢的同学是基于什么考虑不在本地安装一个 cnpm 呢?

注意：
nvm切换nodejs的时候，要用自带的cmd切换，不要用git-bash

nodejs安装完命令无法使用，这是因为全局目录没有在环境变量里，加上就好了
`Path += C:\Users\Administrator\AppData\Roaming\npm;`