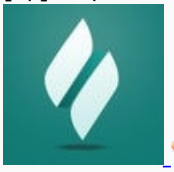
[译]10个 NPM 使用技巧 0 回复24 查看



实验楼管理员 L64
8小时前 技术分享 技术分享

对于一个项目，常用的一些npm简单命令包含的功能有：初始化一个文件夹(npm init)，下载npm模块(npm install)，创建测试(npm test) 和自定义脚本(npm run)。但是，进一步了解一些 npm 的使用技巧可以彻底改变你的日常开发任务。

注: 如果你需要关于初学npm的参考，可以参阅我们的初学者指南。如果你对 npm 和 Yarn 之间的差异感到困扰，可以参阅我们发表的文章：Yarn vs npm:你需要知道的一切

1. 获取帮助

npm 文档 和 CLI 命令行文档 是非常不错地的学习资料，但需要通过浏览器访问，这并不是很方便。因而可以通过命令行快速获取所有可选项：

```
npm help
```

此外，还能获取特定 npm 命令的使用帮助：

```
npm help <command>
```

例如：`npm help install`

另一种方式是通过下面的命令：

```
npm <command> -h
```

2. npm 命令自动完成

npm 通过bash提供了命令自动完成功能(包括 Bash for Windows 10):

```
npm completion >> ~/.bashrc
//or Z shell
npm completion >> ~/.zshrc
```

重新加载shell配置文件：

```
source ~/.bashrc
```

现在，在终端注入 npm ins，然后按下 tab键就会出现 install了，不会再浪费时间去全部输入了。

3.修复全局模块的权限

当你试图安装全部模块时，类 Linux 系统可能会抛出权限错误，可以在npm命令之前添加 sudo 来执行，但这是一个较危险的选择。一个更高的解决方式是改变 npm 默认的模块安装目录：

```
mkdir ~/.npm-global
npm config set prefix '~/.npm-global'
```

使用适当的文本编辑器将下面的一行添加到~/.bashrc或者~/.zshrc文件中：

```
export PATH="$HOME/.npm-global/bin:$PATH"
```

重新加载配置文件(source ~/.bashrc)，然后重新安装npm到用户所属路径：

```
npm install -g npm
```

这也会更新npm。

4.持续更新npm

你可以通过下面的命令显示npm当前的版本：

```
npm -v
```

如果有需要，可以通过下面的命令更新npm：

```
npm install -g npm
```

当 Node 的主版本 released 之后，你也可能需要重新构建 C++ 扩展：

```
npm rebuild
```

如果你需要管理多个版本的node.js和npm，可以考虑使用 n 或者 nvm。这有一篇关于 nvm 的文章：使用 nvm 安装多版本的 Node.js

5.定义默认的 npm init

使用 npm init初始化一个新的项目，这会提示你关于项目的更多细节，并创建一个package.json文件。

如果你厌倦了每次开始一个新的项目都需要重新输入同样的信息，可以使用-y标记表示你能接受package.json 文件的一堆默认值：

```
npm init -y
```

或者你可以设置一些语义化的默认值：

```
npm config set init.author.name <name>
npm config set init.author.email <email>
```

6.更精准的模块搜索

到目前为止，npm上已经有超过350000个模块了，并且每天还在持续增长。尽管有很多非常棒的模块，但是你还是想避免使用一些不受欢迎的、存在bug的或者无人维护的模块。在 npmjs 和 Github 上搜索npm模块是很实用但这还有一些其它选择：

npms

npms 根据一个基于项目版本、模块下载次数、最新更新日期、提交频率、测试覆盖率、文档、贡献者数量、issues数、star数、forks数和作者在社区的地位的综合测量分数进行模块排名。

npm Discover

npm Discover 定位于快速搜索和其它模块通常一起使用的模块，如 body-parser 通常和Express一起使用。

Packages by PageRank

Packages by PageRank 按照模块的谷歌排名进行搜索和排序。

Curated npm Lists

还一个选择就是利用别人的搜索结果。当需要一个健壮的解决方案时，我经常会参考 sindresorhus 的 Awesome Node.js。

7.管理你的模块

你已经安装了一些模块，看看都有啥：

```
npm list
```

(ls、la& ll可以用作 list的别名)

该命令会显示所有模块：(安装的)模块，子模块以及子模块的子模块等。可以限制输出的模块层级：

```
npm list --depth=0
```

打开一个模块的主页：

```
npm home <package>
```

这只有在你的系统能打开浏览器时有用--在服务端的系统上会失败。同样，可以打开一个模块的 Github 仓库：

```
npm repo <package>
```

或者它的文档：

```
npm docs <package>
```

或者它目前的bugs列表：

```
npm bugs <package>
```

npm list会显示和你已经安装地模块的关联模块---这些没有在package.json文件中被引用。你可以单独 npm uninstall每一个模块或者全部移除它们：

```
npm prune
```

如果安装模块时你添加了--production标记或者NODE_ENV被设置成 production，package.json文件中被指定为 devDependencies的模块也会被移除。

8.锁定依赖

默认情况下，当用 --save/-S或者 --save-dev/-D安装一个模块时，npm 通过脱字符(^)来限定所安装模块的主版本号。例如，当运行 npm update 时， ^1.5.1 允许安装版本号大于 1.5.1 但小于 2.0.0 版本的模块。

波浪号(~)字符是限定模块的次要版本。例如，当运行 npm update 时， ~1.5.1 允许安装版本号大于 1.5.1 但小于 1.6.0 版本的模块。可以将需要安装的模块版本前缀默认设置成波浪号(~):

```
npm config set save-prefix="~"
```

对于那些偏执的认为任何更新(模块的行为)会破坏系统的人，可以配置npm仅安装精确版本号的模块：

```
npm config set save-exact true
```

另一个选择是，可以在项目中使用 shrinkwrap:

```
npm shrinkwrap
```

这会生成一个 shrinkwrap.json 文件，该文件包含了你正在使用的模块的指定版本。当运行 npm install 时，该文件所指定的模块版本会覆盖 package.json 文件中所指定的版本。

9. 找出过时的模块

怎么知道一个模块已经更新了呢？我之前的方式是先列举出项目所依赖的模块(npm list --depth=0)，然后在 npmjs.com上找到该模块，手动检查该模块的版本是否已经更新。这非常费时。幸运的是，有一个更简单的方式：

```
npm outdated
```

或者 npm outdated -g 来查找全局模块。

你也可以查看一个独立模块的当前版本：

```
npm list <package>
```

也可以查看检验当前和历史版本：

```
npm view <package> versions
```

npm view <package> 会显示一个独立模块的所有信息，包括它的依赖、关键字、更新日期、贡献者、仓库地址和许可证等。

10.使用开发中的模块

当你正在开发一个模块时，会经常想在其它项目中尝试使用或者在任何一个目录运行它(如果你的应用支持)，这时没必要将其发布到 npm，并全局安装---仅需在该模块所在目录使用下面的命令：

```
npn link
```

该命令会为模块在全局目录下创建一个符号链接。可以通过下面的命令查看模块引用：

```
npm list -g --depth=0
```

或者：

```
npm outdated -g
```

现在，就可以从命令行运行模块或者通过 require 在任何项目中引入该模块。

另一个选择是，可以通过文件路径在 `package.json` 文件中声明对该模块的依赖：

```
"dependencies": {
  "myproject": "file:../myproject/"
}
```

参考

[10 Tips and Tricks That Will Make You an npm Ninja]

转载自：dwqs/blog

文章地址：https://github.com/dwqs/blog/issues/40

全部回复

还没有人回复，沙发空缺中~

登录后回复帖子

我要发帖

标签

课程相关 Linux Python 实验环境 C/C++ 技术分享 课程需求 功能建议 Java 其他 Web Hadoop SQL NodeJS PHP Shell Git 常见问题 HTML 网络 HTML5 信息安全 Android NoSQL GO Ruby 训练营 Perl

相关帖子

Javascript本地存储小结50个安卓开发者应该熟悉的Android Studio技巧和资源Google 和 Baidu 常用的搜索技巧国外最佳互联网安全博客TOP 30能使用html/css解决的问题就不要使用JSVim 新手节省时间的 10 多个小技巧2016 年 7 个顶级 JavaScript 框架全栈必备——Mysql性能调优Web 开发必备指南面向开发者的最佳 Android 库列表

×Close

公司
关于我们
联系我们
加入我们
技术博客
合作
我要投稿
教师合作
高校合作
友情链接
服务
实战训练营
会员服务
实验报告
常见问题
隐私条款
学习路径
Python学习路径
Linux学习路径
大数据学习路径
Java学习路径
PHP学习路径
全部

×Close

注意

取消 确定

×

发帖

标题

至少输入5个字

描述

- 编辑
- 预览

Markdown 语法

推荐使用 Markdown 语法，至少输入 5 个字

板块
标签

取消 提交

×

发帖

标题

[译]10个 NPM 使用技巧

描述

- 编辑
- 预览

Markdown 语法

对于一个项目，常用的一些npm简单命令包

板块
标签

取消 提交

确定删除

删除后不可恢复

取消 确定