安装

在安装 Homebrew 之前，需要将 Xcode Command Line Tools 安装完成，这样你就可以使用基于 Xcode Command Line Tools 编译的 Homebrew。

在 terminal 中复制以下命令（不包括 $），跟随指引，将完成 Hombrew 安装。

$ ruby -e "$(curl -fsSL <https://raw.githubusercontent.com/Homebrew/install/master/install>)"

紧接着，我们需要做一件事让通过 Homebrew 安装的程序的启动链接 (在 /usr/local/bin中）可以直接运行，无需将完整路径写出。通过以下命令将 /usr/local/bin 添加至 $PATH 环境变量中:

$ echo 'export PATH="/usr/local/bin:$PATH"' >> ~/.bash\_profile

Cmd+T 打开一个新的 terminal 标签页，运行以下命令，确保 brew 运行正常。

$ brew doctor

安装完成后，Homebrew 会将本地 /usr/local 初始化为 git 的工作树，并将目录所有者变更为当前所操作的用户，将来 brew 的相关操作不需要 sudo 。

Homebrew基本使用

安装一个包，可以简单的运行：

$ brew install <package\_name>

更新 Homebrew 在服务器端上的包目录：

$ brew update

查看你的包是否需要更新：

$ brew outdated

更新包：

$ brew upgrade <package\_name>

Homebrew 将会把老版本的包缓存下来，以便当你想回滚至旧版本时使用。但这是比较少使用的情况，当你想清理旧版本的包缓存时，可以运行：

$ brew cleanup

查看你安装过的包列表（包括版本号）：

$ brew list --versions

安装Homebrew-cask

$ brew install caskroom/cask/brew-cask

$ brew cask install google-chrome // 安装 Google 浏览器

$ brew update && brew upgrade brew-cask && brew cleanup // 更新

搜索

如果你想查看 cask 上是否存在你需要的 app，可以到 caskroom.io 进行搜索。

文件预览插件

有些插件可以让 Mac 上的文件预览更有效，比如语法高亮、markdown 渲染、json预览等等。

$ brew cask install qlcolorcode

$ brew cask install qlstephen

$ brew cask install qlmarkdown

$ brew cask install quicklook-json

$ brew cask install qlprettypatch

$ brew cask install quicklook-csv

$ brew cask install betterzipql

$ brew cask install webpquicklook

$ brew cask install suspicious-package

OS X 图形界面程序

$ brew cask install alfred

$ brew cask install appcleaner

$ brew cask install cheatsheet

$ brew cask install dropbox

$ brew cask install google-chrome

$ brew cask install onepassword

$ brew cask install sublime-text

$ brew cask install totalfinder

译者本人并不喜欢 brew cask 的安装方式，更倾向于到 App Store 或官方下载 OS X 图形界面程序。主要因为名字不好记忆、偶尔需要手动更新，另外当你使用 Alfred 或 Spotlight ，你将发现将程序安装在 ~/Application 会很方便。

什么是 Homebrew 与 Homebrew Cask

Homebrew 是基于 OS X 的套件管理工具，是一个开源的 Ruby 脚本，专门用于快速下载软件。更通俗地讲，Homebrew 类似于一个软件中心，你可以理解成 App Store 或者 Google Play 那样的软件商店，只不过，Homebrew 比前者以及 Mac App Store 来说有着更丰富的资源与更高效的管理，具体会在下文提及。

至于 Homebrew Cask，它是一套建立在 Homebrew 基础之上的 OS X 软件安装命令行工具，是 Homebrew 的扩展。不拿那么多拗口的术语来烦你了，简言之，你完全可以把 Homebrew Cask 当作是 Homebrew 的一部分，你只需要记住，在安装常用软件的过程中，大部分情况下我们只需要使用 Homebrew Cask 就足够了。

• 通过 Homebrew 下载安装的软件全部来自对应的软件官网，无需担心下载源的安全问题。

• 依存于系统既有的库，减少了空间占用和冗余

• 使用 Git 进行管理和更新

• 易于定制

• 安装软件 / 软件包 / 软件都在一个目录下，方便管理，

命令

brew cask install <软件名1> <软件名2> <软件名3>

通过 Homebrew 或者 Homebrew Cask 安装的软件都会集中在一个目录下面，再由 Homebrew 将相关软件软链接到相关路径。举个例子，如果当我们访问系统中的 A 文件时，系统都会将访问路径导向 B 文件对应的路径，那么 A 文件就是 B 文件的软链接，这也是 Homebrew 的原理。

准备工作

1. 配备 Intel CPU 的 Mac 电脑

确保你的 Mac 系统版本在 OS X 10.9 或以上。

2. 安装 Xcode

如果你的电脑上没有安装 Xcode，你可能需要先在 Mac App Store 下载 Xcode，如果嫌 Xcode 体积庞大的话，可以前往 Apple 开发者网站 尝试下载 Command Line Tools for Xcode 进行安装。

3. 安装 Homebrew

相信有些朋友已经在 Aria2 的文章中安装过 Homebrew 套件管理器了，上一步的 Xcode 安装也是为这一步做准备。如果你还没有安装过 Homebrew，打开终端运行下列代码：

/usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"

brew cask install 软件名

• brew cask install google-chrome 安装 Chrome 浏览器

• brew cask install alfred 安装 Alfred

• brew cask install fliqlo 安装屏保程序 Fliqlo

• brew cask install dropbox 安装 Dropbox

• brew cask install java 安装 Java 等开发环境也是可以的

• brew cask uninstall 软件名 卸载通过 Homebrew Cask 安装的软件

• brew cask search 列出所有可以被安装的软件，当然你也可以直接前往上文提供的 Homebrew Cask 搜索。

• brew cask search google 这里是查找所有与 google 有关的软件，google 关键词可以自行替换

• brew cask info 软件名 查找相关软件的信息

• brew cask cleanup 删除 Homebrew Cask 下载的包

• brew cask list 列出通过 Homebrew Cask 安装的包

• brew cask update 更新 Homebrew Cask

要注意的是，Homebrew Cask 并没有提供相关的软件更新命令，这里我们可以直接使用软件内的更新功能就可以了。

之前少数派写过增强 Mac「预览」功能（QuickLook）的教程，里面有提到很多小插件，其实你也可以直接用 Homebrew 快速下载这些插件，比如：

• brew cask install qlmarkdown 安装 Markdown 预览（QuickLook）插件

• brew cask install qlcolorcode 代码块高亮

• brew cask install qlvideo 视频预览插件

定期更新 Homebrew

更新 Homebrew

要获取最新的包的列表，首先得更新 Homebrew 自己。这可以用 brew update 办到。

brew update

完后会显示可以更新的包列表，其中打钩的是已经安装的包。输出类似下面这样：

Updated Homebrew from fe93aa3 to 6ae64c3.

Updated 1 tap (homebrew/versions).

==> Updated Formulae

awscli cmake ✔ homebrew/versions/libmongoclient-legacy

更新包 (formula)

更新之前，用 brew outdated 查看哪些包可以更新。

brew outdated

然后就可以用 brew upgrade 去更新了。Homebrew 会安装新版本的包，但旧版本仍然会保留。

brew upgrade # 更新所有的包

brew upgrade $FORMULA # 更新指定的包

清理旧版本

一般情况下，新版本安装了，旧版本就不需要了。用 brew cleanup 清理旧版本和缓存文件。Homebrew 只会清除比当前安装的包更老的版本，所以不用担心有些包没更新但被删了。

brew cleanup # 清理所有包的旧版本

brew cleanup $FORMULA # 清理指定包的旧版本

brew cleanup -n # 查看可清理的旧版本包，不执行实际操作

这样一套下来，该更新的都更新了，旧版本也被清理了。

锁定不想更新的包

如果经常更新的话，brew update 一次更新所有的包是非常方便的。但有时候会担心自动升级把一些不希望更新的包更新了。数据库就属于这一类，尤其是 PostgreSQL 跨 minor 版本升级都要迁移数据库的。希望找个时间单独处理它。这时可用 brew pin 去锁定这个包，然后 brew update 就会略过它了。

brew pin $FORMULA # 锁定某个包

brew unpin $FORMULA # 取消锁定

其他几个常用命令

brew info 可以查看包的相关信息，最有用的应该是包依赖和相应的命令。比如 Nginx 会提醒你怎么加 launchctl ，PostgreSQL 会告诉你如何迁移数据库。这些信息会在包安装完成后自动显示，如果忘了的话可以用这个命令很方便地查看。

brew info $FORMULA # 显示某个包的信息

brew info # 显示安装了包数量，文件数量，和总占用空间

brew deps 可以显示包的依赖关系，我常用它来查看已安装的包的依赖，然后判断哪些包是可以安全删除的。

brew deps --installed --tree # 查看已安装的包的依赖，树形显示

输出如下：

elixir (required dependencies)

└── :erlang

wxmac (required dependencies)

├── jpeg

├── libpng

│ └── xz

└── libtiff

└── jpeg