Mac macOS 应用 程序员 编程

关注者 7,231 被浏览 684,534

程序员用 Mac 都有哪些必备的 app?

目前我知道的好像就只有 Xcode、Textmate 和 Coderunner。 可以不拘泥于 iOS 和 Mac 开发，其他语言的开发和 web 开发...显示全部

关注问题 写回答 2 条评论 分享 邀请回答 举报 ···

99 个回答

默认排序

涛吴
编程、旅行、语言 等 5 个话题的优秀回答者

372 人赞同了该回答

- Homebrew
  方便获得各类实用工具，比如 wget
- oh-my-zsh
  自带几十个插件、自动升级的 zsh，任何开机就要开命令行的人都该尝试一下
- iTerm 2
  最新版本深度结合 tmux
- Guard livereload
  Web 开发的凶器
- Sublime Text 2
  完胜 Textmate，Vintage 模式支持部分 Vim key binding
- Divvy
- 用键盘移动 / 布局窗口
- Mou
  Markdown 编辑器
- Valgrind
  相识恨晚
- Transmit
- Doo
  文档管理工具
- SourceTree
  过得去的 Git GUI
- Less.app
- MAMP
- Alfred

... AntiRSI.

编辑于 2014-03-27

▲ 372 ▼ 17 条评论 分享 收藏 感谢 ···

郭鹏
Software Engineer

124 人赞同了该回答

竟然只有一个人提 Dash，那我再提一遍以表重要吧。

发布于 2013-09-22

▲ 124 ▼ 13 条评论 分享 收藏 感谢 ···

phodal
编程、程序员 话题的优秀回答者

相关问题

- 程序员真的需要一台 Mac 吗? 165 个回答
- Mac 程序员开发的利器是什么? 21 个回答
- 为什么那么多人推荐程序员用Mac? 26 个回答
- 关于以后当一个程序员? 22 个回答
- 做主程序员是怎样的体验? 16 个回答

相关推荐

格非讲金瓶梅 · 清华备受欢迎的一堂课
共 10 节课 ▶ 试听

零基础掌握人工智能(AI)核心语言：Python
★★★★★ 504 人参与

Android 程序设计：第 2 版
1,017 人读过 阅读

273 人赞同了该回答

推荐一个我同事写的：GitHub - macdao/ocds-guide-to-setting-up-mac: OCD's Guide to Setting up Mac《强迫症的 Mac 设置指南》

本节介绍一些常用的，跟开发没有直接关系的第三方应用及其设置。

Homebrew
包管理工具，官方称之为The missing package manager for OS X。

安装步骤见官网。

有了 brew 以后，要下载工具，比如 MySQL、Gradle、Maven、Node.js 等工具，就不需要去网上下载了，只要一行命令就能搞定：

```
brew install mysql gradle maven node
```

PS：安装 brew 的时候会自动下载和安装 Apple 的 Command Line Tools。

brew 的替代品有 MacPorts，现在基本没人用它。

Homebrew Cask
brew-cask 允许你使用命令行安装 OS X 应用。比如你可以这样安装 Chrome：brew cask install google-chrome。还有 Evernote、Skype、Sublime Text、VirtualBox 等都可以用 brew-cask 安装。

brew-cask 是社区驱动的，如果你发现 brew-cask 上的应用不是最新版本，或者缺少你某个应用，你可以自己提交 pull request。

安装步骤见官网。

应用也可以通过 App Store 安装，而且有些应用只能通过 App Store 安装，比如 Xcode 等一些 Apple 的应用。App Store 没有对应的命令行工具，还需要 Apple ID。倒是更新起来很方便。

几乎所有常用的应用都可以通过 brew-cask 安装，而且是从应用的官网上下载，所以你要安装新的应用时，建议用 brew-cask 安装。如果你不知道应用在 brew-cask 中的 ID，可以先用brew cask search命令搜索。

iTerm2
iTerm2 是最常用的终端应用，是 Terminal 应用的替代品。提供了诸如Split Panes等一群实用特性。它默认的黑色背景让我毫不犹豫的抛弃了 Terminal。

安装：

```
brew cask install iterm2
```

感谢 brew-cask，我们可以通过命令行自动安装 iTerm2 了。

在终端里，除了可以用^E等快捷键（详见其他快捷键）之外，还可以使用⌥B、⌥F等快捷键（具体可以参考这里）。前提是这样设置一下：

选择Iterm菜单 > Preferences > Profiles，选择你在使用的 Profile（默认是Default），在Keys标签页中把Left option (⌥) key acts as和Right option (⌥) key acts as都设置成+ESC。

在打开新的窗口/标签页的时候，默认情况下新窗口总是 HOME 目录，还需要我每次敲命令才能进入工作目录。如果想要这个新窗口在打开的时候就自动进入工作目录，需要如下设置：

选择Iterm菜单 > Preferences > Profiles，选择你在使用的 Profile（默认是Default），在General标签页中的Working Directory部分中选择Reuse previous seesion's directory。

至此，Terminal 应用已经出色的完成了其历史使命。后面命令行就交给 iTerm2 啦。

在 iTerm2 中双击会自动选中对应的词，三击会选中对应的整行。选中的内容会自动进入剪贴板，不需要再按⌘C复制。

GitHub - macdao/ocds-guide-to-setting-up-mac: OCD's Guide to Setting up MacOh My Zsh
默认的 Bash 是黑白的，没有色彩。而 Oh My Zsh 可以带你进入彩色时代。Oh My Zsh 同时提供一套插件和工具，可以简化命令行操作。后面我们会看到很多介绍，你会看到我爱死这家伙了。

安装方法见官网。

目前我使用的插件有：git z sublime history rbenv bundler rake

Oh My Zsh 使用了 Z shell（zsh），一个和 Bash 相似的 Shell，而非 Bash。

在 Z shell 中，~/.zshrc是最重要的配置文件。Oh My Zsh 在安装的时候会把当前环境的$PATH写入~/.zshrc中。这并不是我期望的行为，因为使用了 brew，我们基本不再需要去定制$PATH，而 Oh My Zsh 提供的默认$PATH值$HOME/bin:/usr/local/bin:$PATH是非常合适的一个值，它把$HOME/bin加入了$PATH，可以让我们把自己用的脚本放到$HOME/bin下。

所以建议把~/.zshrc重置：

```
cp ~/.oh-my-zsh/templates/zshrc.zsh-template ~/.zshrc
```

Oh My Zsh 还有很多有价值的插件。

替代品有 Oh My Fish，使用了 Fishshell 作为基础。

Stow
GNU stow 是管理符号链接（symlink）的一个小公举。主要用于 symlink 你的 dotfiles 如 emacs，git，fish/zsh 的配置文件。安装只需要

```
brew install stow
```

安装了 stow 之后，我们可以开始 symlink 一些 dotfiles 了。完整使用 stow 和 dotfiles 的流程可以参考github.com/jcouyang/dot...

当你的 dotfiles 都妥妥的 symlink 到 ~/dotfiles 后，push 到 github 上就再也不怕换电脑了。

Git 常用别名

几乎每个人都会使用一些方法比如 Git 别名来提高效率，几乎所有人都会把使用git st来代替git status。然而这需要手动设置，每个人也都不完全一样。

Oh My Zsh 提供了一套系统别名（alias），来达到相同的功能。比如gst作为git status的别名。而且 Git 插件是 Oh My Zsh 默认启用的，相当于你使用了 Oh My Zsh，你就拥有了一套高效率的别名，而且还是全球通用的。是不是棒棒哒？下面是一些我常用的别名：

AliasCommandgapagit add --patchgc!git commit -v --amendgclgit clone --recursivegcleangit reset --hard && git clean -dfxgcmgit checkout mastergcmsggit commit -mgcogit checkoutgdgit diffgdcagit diff --cachedglolagit log --graph --pretty = format:'%Cred%h%Creset -%C(yellow)%d%Creset %s %Cgreen(%cr) %C(bold blue) <%an>%Creset' --abbrev-commit --allgpgit pushgrbcgit rebase --continuegstgit statusgupgit pull --rebasegwipgit add -A; git rm $(git ls-files --deleted) 2> /dev/null; git commit -m "--wip--"
完整列表请参考：github.com/robbyrussell...

Scroll Reverser

当你在浏览一个很长的网页时，你看完了当前显示的内容，想要看后续的内容，你可以在 Trackpad 上双指上滑，或者鼠标滚轮向上滚动。这是被称作“自然”的滚动方向。

然而在 Windows 里鼠标滚动的行为是相反的：鼠标滚轮向下滚动才会让浏览器显示后续的内容，向上滚动会达到页面的顶部。你可以在 OS X 的系统偏好设置里修改（选择System Preferences > Trackpad，在Scroll & Zoom标签页中不选中Scroll direction: natural），但是这样会同时改变鼠标滚轮的方向和 Trackpad 的方向。

要想只改变鼠标滚轮的方向，而保持 Trackpad 依旧是“自然”的，我们需要 Scroll Reverser：

```
brew cask install scroll-reverser
```

PS：这货会让三指点击失效

ShiftIt
原生 OS X 下只能手动调整窗口大小，所以我们需要窗口管理工具。我用过很多窗口管理工具，可惜大部分工具都存在快捷键冲突的问题（对我来说主要是 IntelliJ IDEA）。ShiftIt 是少见的没有冲突的窗口管理工具：

```
brew cask install shiftit
```

PS：ShiftIt的旧版本需要安装 X11，最新版本已经修正了这个问题。

替代者有 SizeUp，主要快捷键和 ShiftIt 相同。

当然如果喜欢 hacking，Slate 是个不错的 hackable 的窗口管理工具。配置可以参照 thume.ca/howto/2012/11/...

Sublime Text 2
安装：

```
brew cask install sublime-text
```

在命令行中指定使用 Sublime Text 打开某文件，是一个非常常用的功能，一般我们会按照 OS X Command Line 中所说执行ln -s "/Applications/Sublime Text 2.app/Contents/SharedSupport/bin/subl" ~/bin/subl 来增加subl链接。但是如果你用 brew-cask 安装的话，恭喜你，你不需要运行这个命令，因为 brew-cask 自动帮你做了这件事情。而且你卸载 Sublime Text 的时候 brew-cask 会自动删掉这个链接。

同时 Oh My Zsh 也提供了 Sublime Text 插件，叫做sublime。参考：github.com/robbyrussell...，这个插件和通过 brew-cask 安装的 Sublime Text 完美兼容。

替代品有 Atom、TextMate、Sublime Text 3 等，跟 Sublime Text 2 一样，用 brew-cask 安装的话命令行工具会被自动加入$PATH。

MacDown
MacDown 是 Markdown 编辑器。由于 Mou 一直不支持代码高亮，我就转向了 MacDown。完美支持 GFM。

我特别喜欢 Markdown，我用 Makdown 来写文章（包括本文），写幻灯片（reveal.js）。
Markdown 可以让我专注于内容本身，而无需花精力在排版和样式上。

安装：

```
brew cask install macdown
```

z

在打开终端后，你是怎么进入项目的工作目录？是cd xxx，^R还是用别名？

z 工具可以帮你快速进入目录。比如在我的 Mac 上运行z cask就会进入/usr/local/Library/Taps/caskroom/homebrew-cask/Casks目录。

这货的安装非常方便，甚至都不需要下载任何东西，因为它已经整合在了 Oh My Zsh 中。编辑~/.zshrc文件，在plugins=(git)这行中加上z变成plugins=(git z)，然后运行source ~/.zshrc重新加载配置文件，就可以使用 z 了。

替代品有 autojump。autojump 需要使用 brew 安装。

Vimium

Vimium 是一个 Google Chrome 扩展，让你可以纯键盘操作 Chrome，把你的 Chrome 变成“黑客的浏览器”。

安装方法请参考官方网站。

其他浏览器也有类似的工具，比如 FireFox 的 KeySnail。

LastPass

LastPass 是管理密码的工具，支持二次验证，提供所有浏览器插件以及 Mac 桌面版本。

最重要的是，它提供 **命令行** 的版本，可以直接通过 brew 安装

```
brew install lastpass-cli --with-pinentry
```

之后，只需要登陆：

```
lpass login you@email.com
```

就可以拷贝密码或者集成到其他命令中了：

```
lpass show --password gmail.com -c
```

SourceTree
SourceTree 是 Atlassian 公司出品的一款优秀的 Git 图形化客户端。如果你发现命令行无法满足你的要求，可以试试 SourceTree。

安装：

```
brew cask install sourcetree
```

用 brew-cask 安装会自动增加命令行工具stree到$PATH里。在命令行中输入stree可以快速用 SourceTree 打开当前 Git 仓库。详细用法请参见stree --help。

CheatSheet
CheatSheet 能够显示当前程序的快捷键列表，默认的快捷键是长按⌘。

安装：

```
brew cask install cheatsheet
```

Alfred
Mac 用户不用鼠标键盘的必备神器，配合大量 Workflows，习惯之后可以大大减少操作时间。

上手简单，调教成本在后期自定义 Workflows，不过有大量雷锋使用者提供的现成扩展，访问这里挑选喜欢的，并可以极其简单地根据自己的需要修改。

安装：

```
brew cask install alfred
```

3. 开发工具Java
现在 OS X 都不会自带 JDK 了，所以进行 Java 开发的话，需要下载 JDK。在 brew-cask 之前，我们需要从developer.apple.com/dow... 或者 Oracle 网站上下载。还有更麻烦的－－卸载 JDK 和升级 JDK。

JDK 安装文件是 pkg 格式，卸载和.app不一样，且没有自动卸载方式。

而 brew-cask 提供了自动安装和卸载功能，能够自动从官网上下载并安装 JDK 8。

```
brew cask install java
```

如果你需要安装 JDK 7 或者 JDK 6，可以使用homebrew-cask-versions：

```
brew tap caskroom/versions
brew cask install java6
```

在 OS X 上，你可以同时安装多个版本的 JDK。你可以通过命令/usr/libexec/java_home -V来查看安装了哪几个 JDK。

那问题来了，当你运行java或者 Java 程序时使用的是哪个 JDK 呢？在 OS X 下，java也就是/usr/bin/java在默认情况下指向的是已经安装的最新版本。但是你可以设置环境变量JAVA_HOME来更改其指向：

```
$ java -version
java version "1.8.0_60"
Java(TM) SE Runtime Environment (build 1.8.0_60-b27)
Java HotSpot(TM) 64-Bit Server VM (build 25.60-b23, mixed mode)
$ JAVA_HOME=/Library/Java/JavaVirtualMachines/1.6.0.jdk/Contents/Home java -v
java version "1.6.0_65"
Java(TM) SE Runtime Environment (build 1.6.0_65-b14-466.1-11M4716)
Java HotSpot(TM) 64-Bit Server VM (build 20.65-b04-466.1, mixed mode)
```

其中JAVA_HOME=/Library/Java/JavaVirtualMachines/1.6.0.jdk/Contents/Home可以用JAVA_HOME=`/usr/libexec/java_home -v 1.6`这种更加通用的方式代替。

jEnv
也可以使用 jEnv 来管理不同版本的 JDK，这个工具跟 rbenv 类似，通过当前目录下的.java-version来决定使用哪个 JDK。jEnv 也可以用 brew 安装。不过要使用 jEnv 要有几个问题：

- 需要手动把eval "$(jenv init -)"加入 profile，没有 Oh My Zsh 插件。这点是我非常反感的。

  可以把eval "$(jenv init -)"加入~/.zlogin，这样可以避免修改~/.zshrc。
- 需要手动添加 JDK，不会自动采集系统 JDK。跟 Ruby 不同，OS X 已经提供/usr/libexec/java_home工具来管理安装的 JDK。
- 需要 jenv rehash。这个是跟 rbenv 学的。

所以我建议不要使用 jEnv。

民间使用的 Java 版本切换方法
添加以下脚本到当前 shell 配置文件中：~/.zprofile或者~/.bash_profile。

```
function setjdk() {
    export JAVA_HOME=`/usr/libexec/java_home -v $@`
}
```

这样我们就可以通过输入一条命令进行版本切换了：

```
setjdk 1.8
```

Java[OCD]
作为一个强迫症患者，每当我看到 Java 的错误写法就想纠正过来。

当指编程语言时，Java 的正确写法是首字母大写，其余小写。其他写法比如JAVA、java都是不对的。

在其他一些地方会使用小写的java：

- java命令
- 原文件Main.java
- 包名java.lang

只有在全大写的标题里使用JAVA或者环境变量JAVA_HOME。

IntelliJ IDEA
Java 开发必备工具 IntelliJ IDEA。可以安装 Ultimate Edition：

```
brew cask install intellij-idea
```

也可以安装开源免费的 Community Edition：

```
brew cask install intellij-idea-ce
```

IntelliJ IDEA 有几套内建的快捷键方案（Keymap）。其中适用于 OS X 的有Mac OS X和Mac OS X 10.5+两种。区别是:

- Mac OS X方案和其他平台上的快捷键类似，
- 而Mac OS X 10.5+更加符合 OS X 常用的快捷键。

一个团队使用不同的快捷键会严重影响效率。可以用View | Quick Switch Scheme（^ Back Quote）快速切换 Keymap。

如果可以选择的话，我建议使用Mac OS X方案。因为我经常遇到使用 Windows 的客户，而 Windows 平台上的快捷键和Mac OS X方案类似。

可以从 IDEA 的Help > Default Keymap Reference打开快捷键的参考手册。不过从这里打开的是Mac OS X 10.5+方案的，而Mac OS X方案的可以从这里找到：basrikahveci.com/static...。

rbenv
人人都需要一个 Ruby 版本管理工具。rbenv 就是这样一个轻量级工具，它可以通过 brew 安装。

安装：

```
brew install rbenv ruby-build
```

然后在~/.zshrc中加上rbenv插件。否则你需要手动添加eval "$(rbenv init -)"到~/zshrc或者~/.zprofile文件里。

有时候项目会依赖一些奇怪的版本号，比如ruby-2.1.0，这个时候你需要 rbenv-aliases 帮忙：

```
brew install rbenv-aliases
```

替代品有 RVM、chruby。因为 RVM 不能通过 brew 安装，并且安装的时候会没有节操的修改一堆文件，所以被我早早的弃用了。chruby 也是一个轻量级工具，而且可以完美的和 Oh My Zsh 集成在一起，我看到有些生产环境在用它。

Ruby 常用别名
几乎所有 Ruby 开发人员都会把bi作为bundle install的别名。Oh My Zsh 提供builder插件，这个插件提供了一套别名，比如bi、be。同时还能让你在运行一些常用 gem 的时候直接输入rspec，不需要be rspec这样了。具体包括哪些命令请参考这里。

Z shell 对于[和]符号有特殊的处理，所以在运行rake task[parameter]的时候会报错，你需要改成rake task\[parameter\]或者noglob rake task[parameter]。然而 Oh My Zsh 已经看穿这一切，自带的 rake 插件已经解决了这个问题：brake task[parameter]。

添加插件的时候注意把rake放到bundler后面，例如这样：

```
plugins=(git z sublime history rbenv bundler rake)
```

编辑于 2016-03-09

彭勇
linux/java/python/FreeSwitch/PM

77 人赞同了该回答

在github上建立了一个项目：

```
https://github.com/pubyun/macdev
```

Mac for Developer v0.1

将本人在使用Mac作为开发工具的过程中的一些体会做个记录和整理，方便以后 自己和团队的参考，提高效率。如果本文对其他开发人员也有所帮助，欢迎您给 给出反馈或者提出改进意见

本文假设您是一个开发工程师，并且是一个Mac的新手。这些步骤在OS X Mavericks 下测试通过。

欢迎大家一起改进这个项目，请Fork、Star或在Issues中提交 :)

您也可以关注我的新浪微博，以获取最新消息。

- OS X的安装
- OS X的备份和恢复
- 基本设置
- python开发环境的设置
- ruby开发环境的设置 - 整理中...
- java开发环境的设置 - 整理中...
- 推荐软件
- 配置文件 - 整理中 ...
- 常用快捷键 - 整理中 ...
- 参考文档

发布于 2014-01-21

刘鑫
程序员