

# 目录

前言	1.1
计算机编程领域	1.2
版本号命名和版本发布	1.2.1
长期支持版本	1.2.2
命令行终端	1.2.3
Help查看帮助信息	1.2.4
Next Generation	1.2.5
配置文件名为xxxfile	1.2.6
i18n和l10n	1.2.7
知识技能图谱	1.2.8
Cheat Sheet和Handout	1.2.9
编译和链接	1.2.10
嵌入式领域	1.3
上层软件领域	1.4
debounce防抖动	1.4.1
库/包的依赖管理工具	1.4.2
上层软件 - 数据库	1.4.3
schema模式	1.4.3.1
各种编程语言领域	1.5
print打印	1.5.1
log日志	1.5.2
async和wait	1.5.3
getter和setter	1.5.4
编程语言 - Python	1.5.5
adapter适配器	1.5.5.1
附录	1.6
参考资料	1.6.1

# 计算机编程通用逻辑知识概念

## 简介

计算机和编程等领域内，存在很多跨语言，跨领域的通用知识，现整理出来，供参考。

## 源码+浏览+下载

本书的各种源码、在线浏览地址、多种格式文件下载如下：

### Gitook源码

- [crifan/program\\_common\\_logic](#): 计算机编程通用逻辑知识概念

### 在线浏览

- [计算机编程通用逻辑知识概念 book.crifan.com](#)
- [计算机编程通用逻辑知识概念 crifan.github.io](#)

### 离线下载阅读

- [计算机编程通用逻辑知识概念 PDF](#)
- [计算机编程通用逻辑知识概念 ePUB](#)
- [计算机编程通用逻辑知识概念 MOBI](#)

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间：2018-01-02 14:31:24

# 计算机编程领域

此处介绍计算机编程领域内的一些通用知识、逻辑和概念。

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2018-01-02 15:16:25

# 版本号命名和版本发布

## 版本号命名规范

比如每次发版本的app，如果只是小功能更新，则只是第二位加1（或根据功能数量决定），否则大的版本的改进，则是第一位加1

[Configuration · GitBook Toolchain Documentation](#)

是有个官网推荐的做法的：

[Semantic Versioning 2.0.0 | Semantic Versioning](#)

其解释是：

Given a version number MAJOR.MINOR.PATCH, increment the:

- MAJOR version : when you make incompatible API changes,
- MINOR version : When you add functionality in a backwards-compatible manner, and
- PATCH version : when you make backwards-compatible bug fixes.

Additional labels for pre-release and build metadata are available as extensions to the MAJOR.MINOR.PATCH format.

中文版翻译是：

[语义化版本 2.0.0](#)

版本格式：主版本号.次版本号.修订号，版本号递增规则如下：

1. 主版本号：当你做了不兼容的 API 修改，
2. 次版本号：当你做了向下兼容的功能性新增，
3. 修订号：当你做了向下兼容的问题修正。

先行版本号及版本编译信息可以加到“主版本号.次版本号.修订号”的后面，作为延伸。

现在个人的常见做法是：

1. 主版本：有重大功能更新
  - 比如多加了一个大的功能模块
2. 次版本：有一些重要更新
  - 比如部分功能有重大优化
3. 补丁版本/小版本：细节的优化
  - 比如一些小功能的优化，修复了一些小bug等等

## 版本发布的一些实践和做法

对于发布新版本，去写更新日志时，其形式可以借鉴[jeesite](#)的做法：

### /// V1.0.3

Release Date: Friday, April 19, 2013

改进 问题 添加批量保存菜单排序功能

改进 实体增加@DynamicInsert和@DynamicUpdate注解，只插入或更新需要更新的字段



Opened by think-gem, issue closed on Friday, May 31, 2013

#### // 实体增加@DynamicInsert和@DynamicUpdate注解，只插入或更新需要更新的字段

改进 使用ERMaster数据建模，DbUnit数据文件将xml类型更改为xls类型，分模块存放

改进 界面改为紧凑风格

改进 增加单元测试公共基类；Hibernate升级到4.2；Hibernate Validator升级到5.0.1

改进 系统缓存调成，减少cms与sys模块的耦合。

新增 增加DataEntity数据实体，包含：备注、创建者、创建时间、更新者、更新时间、删除标记字段

新增 增加sql server驱动及建表脚本

新增 集成Activiti5.12工作流引擎，提供简单请假的例子

新增 新增数据集权限控制

新增 登录3次错误后，输入验证码

缺陷 评论模块，查看文档

其效果很醒目，且点击每条可以展开显示详情

但是建议关键字还是用个人目前常用的做法：

- 新增：增加了新功能
- 修复：修复了之前存在的bug和问题
- 优化：改进，优化了已有的东西

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间：2018-01-02 16:24:52

# 长期支持版本LTS

LTS = Long Term Support = 长期支持

## 背景

在软件领域内，很多的软件，发布的版本，随着时间的流逝，很多软件会发生很多可能，其中很大的可能是，过了一段，或许可以称为足够长的时间后，该版本的软件，已经不被原先的发布商所支持了。导致依赖于此版本的软件的其他软件，变得不可用了。这种现象，尤其是在大的软件和系统中比较常见。

## 解决办法

对此，大的软件发布商为了告诉其他人说，我发布的某某版本的某个软件，会长期支持，不会中断，保持一定程度的稳定，可以被信任，则此版本的软件，往往就叫做LTS，长期支持版本。

比如：

## Ubuntu的LTS版本

Ubuntu就发布过很多个LTS的版本，比如 Ubuntu 12.04 LTS

详见：[LTS - Ubuntu Wiki](#)

## Node.js的LTS版本

Node.js的也有[LTS版本](#)

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间： 2018-01-02 16:31:01

## 命令行终端

计算机领域内，常常会提及到终端，也叫做 命令行，往往指的是：

- Windows 中的 cmd
- Linux 中的 shell
- Mac 中的 terminal

然后还有很多第三方的扩展、工具或插件，用于增强或替换系统自带的终端，比如：

- Windows 的 Power Shell
- Linux / Mac 的 zsh
- Mac 的 iTerm

## 终端能用来做什么

答 运行各种脚本和命令，完成各种任务

而对于linux的 shell 来说，功能强大到都算作一门单独的语言： shell脚本语言

可以利用不同的语法和函数，编写复杂逻辑的代码，实现特定的功能

比如小功能：

自己的批量重命名的shell脚本

TODO: 把脚本找到 贴出来

比如，其他人用shell去给OpenWrt的极路由去写应用：

[开发极路由云插件 - OpenWrt.io](#)

比如：

在linux mac win 下运行我的python脚本BlogsToWordpress

用vi在无图形界面的centos中编辑文本文件

## 各种终端的用法 心得

下面来介绍各种不同的终端的各种用法和心得。

### Windows 中的终端

#### Windows 自带的 cmd

TODO: 把之前那个cmd心得的帖子整理过来

#### Windows 中的 Hyper Terminal

嵌入式中都有和终端有关的工具使用

嵌入式开发期间，会遇到把开发板中运行程序的输出打印显示出来，以及输入和命令执行，也需要用到终端。

有时候Windows自带的terminal也够用，但是Win7中有个更好用一点的： Hyper Terminal

详见： [【整理】如何在Win7中安装使用超级终端Hyper Terminal – 在路上](#)

## Mac 中的终端

### Mac 中自带的 terminal

#### Mac 中的 iTerm2

TODO: 整理mac的iterm2的好用之处

## Linux 的 shell

linux shell编程

## shell入门

1. 快速有个了解和概念：
  - [Shell脚本编程30分钟入门](#)
2. 完整和深入的学习
  - [Shell 教程 | 菜鸟教程](#)
  - [Linux Shell脚本教程：30分钟玩转Shell脚本编程\\_Shell中文网](#)
3. 其它还可以的资料
  - [LINUX与UNIX Shell编程指南](#)
  - [Shell编程基础 - Ubuntu中文](#)
  - [详细介绍Linux shell脚本基础学习\(一\) - 雪洁 - 博客园](#)
4. 英文不错的话，直接看英文
  - 比较系统的英文教程
    - [Advanced Bash-Scripting Guide](#)
    - [BashGuide - Greg's Wiki](#)
  - 其它更加全面的，更加系统的书籍
    - [《UNIX Shells by Example Fourth Edition》](#)
    - chm版本下载：[Prentice.Hall-Unix.Shells.By.Example,4th.Edition.chm\\_微盘下载](#)
    - [Linux Command Line and Shell Scripting Bible第三版](#)
5. 对于Shell有了基本的了解和实践之后，再去看
  - [Shell脚本编程的常识](#)
    - 这些往往是经常用到，但是各种网络上的材料都语焉不详的东西，个人认为比较有用)

## 支持多平台的 SecureCRT

当然，远程操作（CentOS等）Linux类系统，也是通过终端类等工具去操作的

比如常用的SecureCRT

TODO: 把 [【crifan推荐】支持多种协议的串口开发工具：SecureCRT](#)

[【crifan推荐】极佳的串口开发工具：SecureCRT – 在路上](#)

整理过来。

## 其他环境中的终端

## IDE工具中集成的终端

其他第三方工具，尤其是IDE，为了方便开发同时集成了终端  
比如 VSCode PyCharm  
详见 TODO: ide总结的帖子

## Android 的app去模拟终端

[【已解决】android中的shell命令行工具 – 在路上](#)

以便于我们在手机中通过终端去操作系统的资源，实现各种复杂的功能

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2018-01-02 16:51:51

## Help查看帮助信息

用help查看帮助信息

用help获取主命令和子命令的帮助信息

想要搞懂命令的基本语法，可以用：

```
some_command --help
```

或：

```
some_command help
```

甚至是命令下面的子参数，子命令的更详细的语法，可以用：

```
some_command help sub_command
```

往往可以输出对应的命令的帮助信息

举例：

## 嵌入式的uboot的语法

[U-Boot: Quick reference - CompuLab Wiki](#)

```
help - print online help
```

-> help xxx

可以输出子命令的具体的详细的语法

xxx是uboot的子命令，比如ls, cp, md等等

## iOS中包管理器Carthage的语法

[Carthage/Artifacts.md at master · Carthage/Carthage](#)

[Carthage/Carthage: A simple, decentralized dependency manager for Cocoa](#)

```
ligrifandeMacBook-Pro:QorosSales crifan$ carthage help
Available commands:

archive      Archives built frameworks into a zip that Carthage can use
bootstrap     Check out and build the project's dependencies
build        Build the project's dependencies
checkout      Check out the project's dependencies
copy-frameworks In a Run Script build phase, copies each framework specified by a SCRIPT_INPUT_FILE environment variable into the built app bundle
fetch         Clones or fetches a Git repository ahead of time
help          Display general or command-specific help
outdated      Check for compatible updates to the project's dependencies
update        Update and rebuild the project's dependencies
version       Display the current version of Carthage

ligrifandeMacBook-Pro:QorosSales crifan$ carthage help build
Build the project's dependencies

[--configuration (string)]
the Xcode configuration to build

[--platform (platform)]
```

```

the platforms to build for (one of 'all', 'Mac', 'iOS', 'watchOS', 'tvOS', or comma-separated values of the formers except
for 'all')

[--derived-data (string)]
path to the custom derived data folder

[--no-skip-current]
don't skip building the Carthage project (in addition to its dependencies)

[--color (color)]
whether to apply color and terminal formatting (one of 'auto', 'always', or 'never')

[--verbose]
print xcdebuild output inline

[--project-directory (string)]
the directory containing the Carthage project

[[[]]
the dependency names to build

```

## 苹果的Swift编译器 swiftc

之前在：

【已解决】Xcode项目编译出错：Command failed due to signal: Segmentation fault: 11

期间，注意到点苹果的swift语言的编译器swiftc，对应help结果是：

```

→ crifanLib git:(master) swiftc --help
OVERVIEW: Swift compiler

USAGE: swiftc [options] inputs

MODES:
-dump-ast           Parse and type-check input file(s) and dump AST(s)
-dump-parse          Parse input file(s) and dump AST(s)
-dump-scope-maps    expanded-or-list-of-line:column
                    Parse and type-check input file(s) and dump the scope map(s)
-dump-type-refinement-contexts
                    Type-check input file(s) and dump type refinement contexts(s)
-emit-assembly      Emit assembly file(s) (-S)
-emit-bc            Emit LLVM BC file(s)

...
-parse              Parse input file(s)
-print-ast          Parse and type-check input file(s) and pretty print AST(s)
-typecheck          Parse and type-check input file(s)

OPTIONS:
-api-diff-data-file path
                    API migration data is from path
-application-extension Restrict code to those available for App Extensions
-assert-config value Specify the assert_configuration replacement. Possible values are Debug, Release, Unsigned, DisableReplacement.

...
-whole-module-optimization
                    Optimize input files together instead of individually
-Xcc arg             Pass arg to the C/C++/Objective-C compiler
-Xlinker value       Specifies an option which should be passed to the linker

```

## sftp命令不了解，用help去查看有哪些命令

之前：

【已解决】命令行中ftp操作上传到远程服务器crifan.com

期间，对于sftp的命令有哪些不是很了解，所以可以通过help去查看：

```
sftp help
Available commands:
bye                                         Quit sftp
cd path                                       Change remote directory to 'path'
chgrp grp path                                Change group of file 'path' to 'grp'
chmod mode path                               Change permissions of file 'path' to 'mode'
chown own path                                Change owner of file 'path' to 'own'
df [-hi] [path]                                 Display statistics for current directory or
                                                filesystem containing 'path'
exit                                         Quit sftp
get [-afPpRr] remote [local]                   Download file
reget [-fPpRr] remote [local]                  Resume download file
reput [-fPpRr] [local] remote                 Resume upload file
help                                         Display this help text
lcd path                                       Change local directory to 'path'
lls [ls-options [path]]                      Display local directory listing
lmkdir path                                    Create local directory
ln [-s] oldpath newpath                      Link remote file (-s for symlink)
lpwd                                         Print local working directory
ls [-lafhlnrSt] [path]                        Display remote directory listing
lumask umask                                  Set local umask to 'umask'
mkdir path                                     Create remote directory
progress                                       Toggle display of progress meter
put [-afPpRr] local [remote]                  Upload file
pwd                                         Display remote working directory
quit                                         Quit sftp
rename oldpath newpath                      Rename remote file
rm path                                       Delete remote file
rmdir path                                    Remove remote directory
symlink oldpath newpath                     Symlink remote file
version                                       Show SFTP version
command                                       Execute 'command' in local shell
!                                            Escape to local shell
?                                            Synonym for help
sftp
```

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间： 2018-01-02 23:39:24

## Next Generation

有些软件，代码，功能，在开发了新一代，下一代，功能更好更强的版本后，有的会以 `Next Generation` 去命名。

比如之前的：

### 嵌入式的交叉编译的工具 `crosstool-ng`

`crosstool` 的下一代，新一代版本，叫做：`crosstool-ng`

### 用于科学上网的shadowsocks的Mac版本 `Shadowsocks-NG`

之前叫做`ShadowsocksX`，其新版本，下一代，叫做`Shadowsocks-NG`

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间： 2018-01-02 16:58:43

## 配置文件名为xxxfile

有很多技术、工具和框架，去给自己的配置文件命名的时候，用：`xxxfile[.yyy]`

比如：

## 前端开发

- `grunt` : `Gruntfile.js`
- `gulp` : `gulpfile.js`

## iOS开发

- `Carthage` : `Cartfile`
- `Cocoapods` : `Podfile`

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间：2018-01-05 11:30:49

## i18n和l10n

在开发软件、APP时，往往涉及到除了支持中文之外，还要支持其他国家的语言，比如英语、法语、德语等等，专业叫法叫做：多国语言（版本）或者国际化

而在谈论多国语言、国际化时，常常会遇到这个词：`i18n`

其实 `i18n == internationalization == 国际化 == 多国语言`

->之所以这么叫是由于 `internationalization` 单词太长，把中间的**18**个字母 `internationalizatio` 省略掉，简写为了 `i18n`  
和 `i18n` 的类似的含义的类似叫法还有：`l10n == localization == 本地化`

举例：

之前见过的：[python-babel/flask-babel: i18n and l10n support for Flask based on Babel and pytz](#)

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间：2018-01-04 20:33:59

## 知识技能图谱

在学习不同领域的技术时，往往你会发现，不同领域所需要学习的内容和方向差距还是很大的，感觉算是隔行如隔山，换一个技术领域和方向，很多名词甚至都没听说过。

另外，在某个领域有了深入专研之后，往往可以总结出很多该领域相关的技术和名词，整理出成系统的内容，往往就叫做该领域的技术图谱，知识图谱。

### TeamStuQ/skill-map: 程序员技能图谱

这个[TeamStuQ/skill-map: 程序员技能图谱](#)中已经整理了几十个相关领域的技能图谱了，此处就不再做重复劳动，请移步去看即可。

此处把其他一些人制作的的图谱整理如下，供参考：

### 前端开发 知识图谱

[QzhouZ/Front-End-Study: 前端开发基础知识学习](#)



[JacksonTian/fks: 前端技能汇总 Frontend Knowledge Structure]<https://github.com/JacksonTian/fks>



crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2018-01-05 21:18:15

## Cheat Sheet 和 Handout

### Cheat sheet 的含义

软件领域内的

Cheat sheet和handout的解释

- » Cheat, 中文含义为：作弊, Sheet: 中文含义为：表格, 表单, 小纸片
- » cheat sheet, 中文含义为：（用于作弊）小抄
- » 放在软件行业内，意思是：

就像考试作弊一样，把大量的内容，精简的，有机的排列在一张纸，一张图上

特点：

- 化繁为简

用途：

- 方便随时参考和查阅
- 尤其常见于，把各种软件，工具等的快捷键，主要功能等，总结后列在一张图片内，多数以图文并茂的表格，图表，图片等形式展现

概述：

- 某某（语言，工具等）的Cheat sheet == 某某（语言，工具等）快速查阅的图表型的手册

### handout 的含义

另外的 handout，中文本意：讲义；宣传册子，宣传品

- » 往往有时候也可以认为：

`cheat sheet == handout`

Cheat sheet和handout的举例：

### Cheat sheet 举例

#### vi的cheat sheet

[VI Editor / Linux Terminal Cheat Sheet \(PDF\) – Smashing Magazine](#)

- »

[VI-Help-Sheet-01-large2.jpg](#)



VI Help Sheet

Modes & Controls		Inserting Text		Other	
<b>Command Mode</b>	ESC (commands preceded by :)	<b>i</b>	Insert before cursor	<b>u</b>	Undo last change
<b>Insertion Mode</b>	Entered on insertion or change	<b>a</b>	Append after cursor	<b>J</b>	Join lines
<b>Starting VI (command line)</b>		<b>I</b>	Insert before line	<b>nJ</b>	Join next n lines
<b>vi &lt;filename&gt;</b>	Edit <i>filename</i>	<b>A</b>	Append after line	.	Repeat last command
<b>vi -r &lt;filename&gt;</b>	Edit last version of <i>filename</i> after crash	<b>o</b>	Add new line after current line	<b>U</b>	Undo all changes to line
<b>vi +n &lt;filename&gt;</b>	Edit <i>filename</i> at line n	<b>O</b>	Add new line before current line	<b>:N</b>	Open split screen
<b>vi + &lt;filename&gt;</b>	Edit <i>filename</i> at end of file	<b>r</b>	Overwrite one character	<b>v</b>	Visual mode
<b>vi +/str &lt;filename&gt;</b>	Edit <i>filename</i> at first occurrence of <i>str</i>	<b>R</b>	Overwrite many characters	<b>ctrl + c</b>	Escape insert mode
In insertion mode the following should be preceded by ESC:		<b>:r &lt;file&gt;</b>	Reads <i>file</i> and inserts it after this line		
<b>:w</b>	Save	<b>p</b>	Put after the position or line		
<b>:x</b>	Save & Exit	<b>P</b>	Put before the position or line		
<b>:q</b>	Exit if no changes made	<b>C</b>	Rewrite the whole line		
<b>:q!</b>	Exit & discard any changes				
Cursor Navigation		Deleting Text		Searching	
<b>h or ▲</b>	Cursor left	<b>x</b>	Delete character to right of cursor	<b>/string</b>	Search forward for string
<b>j or ▼</b>	Cursor down	<b>X</b>	Delete character to left of cursor	<b>?string</b>	Search backwards for string
<b>k or ▲</b>	Cursor up	<b>D</b>	Delete the rest of line	<b>n</b>	Go to next match
<b>l or ▶</b>	Cursor right	<b>dd or :d</b>	Delete current line	<b>N</b>	Go to previous match
<b>w</b>	Next word	<b>ndw</b>	Deletes the next n words	<b>:set ic</b>	Ignore case while searching
<b>W</b>	Next blank delimited word	<b>ndb</b>	Deletes the previous n words	<b>:set noic</b>	Case-sensitive searching
<b>b</b>	Start of word	<b>ndd</b>	Deletes n lines starting with current	<b>:set nu</b>	Turn on line numbers
<b>B</b>	Start of blank delimited word	<b>:x,yd</b>	Delete lines x through y	<b>:x,yg/str</b>	Search for str from line x to line y
<b>e</b>	End of word	<b>:r &lt;file&gt;</b>	Reads <i>file</i> and inserts it after this line	<b>:g/str/cmd</b>	Run cmd on lines containing str
<b>E</b>	End of blank delimited word	<b>d{nav_cmd}</b>	Overwrite many characters	*	Search for next instance of current word
<b>(</b>	Back a sentence	<b>:r &lt;file&gt;</b>	Reads <i>file</i> and inserts it after this line	#	Search for last instance of current word
<b>)</b>	Forward a sentence				
<b>{</b>	Back a paragraph				
<b>}</b>	Forward a paragraph				
<b>0</b>	Beginning of line				
<b>\$</b>	End of the line				
<b>1G</b>	Start of file				
<b>G</b>	End of file				
<b>:n</b>	nth line of file				
<b>f&lt;char&gt;</b>	Forward to <i>char</i>				
<b>F&lt;char&gt;</b>	Back to <i>char</i>				
<b>H</b>	Top of screen				
<b>M</b>	Middle of screen				
<b>L</b>	Bottom of screen				
<b>%</b>	Matching bracket				
<b>gg</b>	Start of document				
Replacing					
<b>:s/pt/str/flag</b>		Replace pattern with string			
<b>Flags</b>					
<b>g</b>	Replace all occurrences of pattern				
<b>c</b>	Confirm replaces				
<b>&amp;</b>	Repeat last :s command				

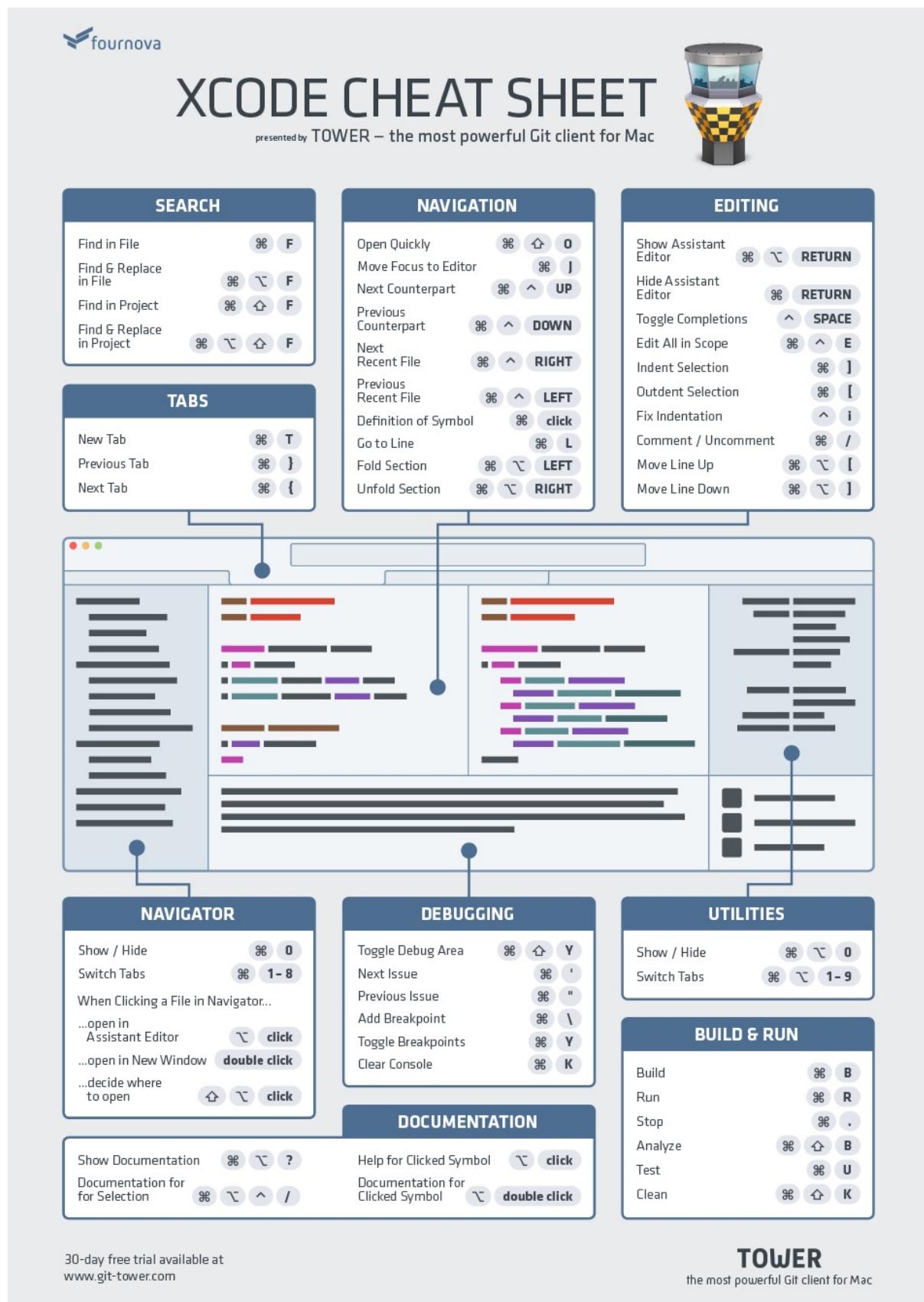
Download this Help Sheet now at [gosquared.com/liquidity](http://gosquared.com/liquidity)  
Put it on your wall Referenced from <http://www.lagmonster.org/docs/vi.html>

© 2010 Go Squared Ltd.

## Xcode的快捷键的cheat sheet

[http://www.crifan.com/xcode\\_common\\_used\\_keyboard\\_shortcut](http://www.crifan.com/xcode_common_used_keyboard_shortcut)

中的：



python的cheat sheet

[python-cheat-sheet-v1.pdf](#)[Python\\_qr.pdf](#)[python:cours:mementopython3-english.pdf](#)

Keep This Python Cheat Sheet on Hand When Learning to Code



sys Variables		String Methods		Datetime Methods	
argv	Command line args	capitalize()	lstrip()	today()	fromordinal(ordinal)
builtin_module_names	Linked C modules	center(width)	partition(sep)	now(timezoneinfo)	combine(date, time)
byteorder	Native byte order	count(sub, start, end)	replace(old, new)	utcnow()	strptime(date, format)
check_interval	Signal check frequency	decode()	rfind(sub, start, end)	fromtimestamp(timestamp)	utcfromtimestamp(timestamp)
exec_prefix	Root directory	encode()	rindex(sub, start, end)		
executable	Name of executable	endswith(sub)	rjust(width)		
exitfunc	Exit function name	expandtabs()	rpartition(sep)		
modules	Loaded modules	find(sub, start, end)	rsplit(sep)		
path	Search path	index(sub, start, end)	rstrip()		
platform	Current platform	isalnum()	split(sep)		
stdin, stdout, stderr	File objects for I/O	isalpha()	splittlines()		
version_info	Python version info	isdigit()	startswith(sub)		
winver	Version number	islower()	strip()		
<b>sys.argv for \$ python foo.py bar -c qux --h</b>		isspace()	swapcase()		
sys.argv[0]	foo.py	istitle()	title()		
sys.argv[1]	bar	isupper()	translate(table)		
sys.argv[2]	-c	join()	upper()		
sys.argv[3]	qux	ljust(width)	zfill(width)		
sys.argv[4]	--h	lower()			
<b>os Variables</b>		<b>Note</b> Methods marked * are locale dependant for 8-bit strings.		<b>Time Methods</b>	
altsep	Alternative sep			replace()	utcoffset()
curdir	Current dir string			isoformat()	dst()
defpath	Default search path			__str__()	tzname()
devnull	Path of null device			strftime(format)	
extsep	Extension separator				
linesep	Line separator				
name	Name of OS				
pardir	Parent dir string				
pathsep	Patch separator				
sep	Path separator				
<b>Note</b> Registered OS names: "posix", "nt", "mac", "os2", "ce", "java", "riscos"		<b>List Methods</b>		<b>Date Formatting (strftime and strptime)</b>	
		append(item)	pop(position)	%a Abbreviated weekday (Sun)	
		count(item)	remove(item)	%A Weekday (Sunday)	
		extend(list)	reverse()	%b Abbreviated month name (Jan)	
		index(item)	sort()	%B Month name (January)	
		insert(position, item)		%c Date and time	
		<b>File Methods</b>		%d Day (leading zeros) (01 to 31)	
		close()	readlines(size)	%H 24 hour (leading zeros) (00 to 23)	
		flush()	seek(offset)	%I 12 hour (leading zeros) (01 to 12)	
		fileno()	tell()	%j Day of year (001 to 366)	
		isatty()	truncate(size)	%m Month (01 to 12)	
		next()	write(string)	%M Minute (00 to 59)	
		read(size)	writelines(list)	%p AM or PM	
		readline(size)		%S Second (00 to 61*)	
				%U Week number <sup>1</sup> (00 to 53)	
				%w Weekday <sup>2</sup> (0 to 6)	
				%W Week number <sup>3</sup> (00 to 53)	
				%x Date	
				%X Time	
				%y Year without century (00 to 99)	
				%Y Year (2008)	
				%Z Time zone (GMT)	
				%% A literal "%" character (%)	
<b>Class Special Methods</b>				1. Sunday as start of week. All days in a new year preceding the first Sunday are considered to be in week 0.	
				2. 0 is Sunday, 6 is Saturday.	
				3. Monday as start of week. All days in a new year preceding the first Monday are considered to be in week 0.	
				4. This is not a mistake. Range takes account of leap and double-leap seconds.	
		<b>Indexes and Slices (of a=[0,1,2,3,4,5])</b>		<b>Available free from <a href="#">AddedBytes.com</a></b>	
		len(a)	6		
		a[0]	0		
		a[5]	5		
		a[-1]	5		
		a[-2]	4		
		a[1:]	[1,2,3,4,5]		
		a[:5]	[0,1,2,3,4]		
		a[:-2]	[0,1,2,3]		
		a[1:3]	[1,2]		
		a[1:-1]	[1,2,3,4]		
		b=a[:]	Shallow copy of a		

## go语言的cheat sheet

[go-lang-cheat-sheet](#)



Go Cheat Sheet		Operators																																																	
<b>Credits</b>		Arithmetic																																																	
Most example code taken from <a href="#">A Tour of Go</a> , which is an excellent introduction to Go. If you're new to Go, do that tour. Seriously.		<table> <thead> <tr> <th>Operator</th><th>Description</th><th>Operator</th><th>Description</th></tr> </thead> <tbody> <tr> <td>+</td><td>addition</td><td>==</td><td>equal</td></tr> <tr> <td>-</td><td>subtraction</td><td>!=</td><td>not equal</td></tr> <tr> <td>*</td><td>multiplication</td><td>&lt;</td><td>less than</td></tr> <tr> <td>/</td><td>quotient</td><td>&lt;=</td><td>less than or equal</td></tr> <tr> <td>%</td><td>remainder</td><td>&gt;</td><td>greater than</td></tr> <tr> <td>&amp;</td><td>bitwise AND</td><td>&gt;=</td><td>greater than or equal</td></tr> <tr> <td> </td><td>bitwise OR</td><td></td><td>Logical</td></tr> <tr> <td>^</td><td>bitwise XOR</td><td></td><td></td></tr> <tr> <td>&amp;^</td><td>bit clear (AND NOT)</td><td>&amp;&amp;</td><td>logical AND</td></tr> <tr> <td>&lt;&lt;</td><td>left shift</td><td>  </td><td>logical OR</td></tr> <tr> <td>&gt;&gt;</td><td>right shift (logical)</td><td>!</td><td>logical NOT</td></tr> </tbody> </table>		Operator	Description	Operator	Description	+	addition	==	equal	-	subtraction	!=	not equal	*	multiplication	<	less than	/	quotient	<=	less than or equal	%	remainder	>	greater than	&	bitwise AND	>=	greater than or equal		bitwise OR		Logical	^	bitwise XOR			&^	bit clear (AND NOT)	&&	logical AND	<<	left shift		logical OR	>>	right shift (logical)	!	logical NOT
Operator	Description	Operator	Description																																																
+	addition	==	equal																																																
-	subtraction	!=	not equal																																																
*	multiplication	<	less than																																																
/	quotient	<=	less than or equal																																																
%	remainder	>	greater than																																																
&	bitwise AND	>=	greater than or equal																																																
	bitwise OR		Logical																																																
^	bitwise XOR																																																		
&^	bit clear (AND NOT)	&&	logical AND																																																
<<	left shift		logical OR																																																
>>	right shift (logical)	!	logical NOT																																																
Original HTML Cheat Sheet by Ariel Mashraki (a8m): <a href="https://github.com/a8m/go-lang-cheat-sheet">https://github.com/a8m/go-lang-cheat-sheet</a>		Comparison																																																	
<b>Go in a Nutshell</b>		Other																																																	
<ul style="list-style-type: none"> <li>Imperative language</li> <li>Statically typed</li> <li>Syntax similar to Java/C/C++, but less parentheses and no semicolons</li> <li>Compiles to native code (no JVM)</li> <li>No classes, but structs with methods</li> <li>Interfaces</li> <li>No implementation inheritance. There's <a href="#">type embedding</a>, though.</li> <li>Functions are first class citizens</li> <li>Functions can return multiple values</li> <li>Has closures</li> <li>Pointers, but not pointer arithmetic</li> <li>Built-in concurrency primitives: Goroutines and Channels</li> </ul>		<table> <thead> <tr> <th>Operator</th><th>Description</th><th>Operator</th><th>Description</th></tr> </thead> <tbody> <tr> <td>&amp;</td><td>address of / create pointer</td><td>*</td><td>dereference pointer</td></tr> <tr> <td>&lt;-</td><td>send / receive operator (see 'Channels' below)</td><td></td><td></td></tr> </tbody> </table>		Operator	Description	Operator	Description	&	address of / create pointer	*	dereference pointer	<-	send / receive operator (see 'Channels' below)																																						
Operator	Description	Operator	Description																																																
&	address of / create pointer	*	dereference pointer																																																
<-	send / receive operator (see 'Channels' below)																																																		
<b>Basic Syntax</b>		<b>Declarations</b>																																																	
<b>Hello World</b>		Type goes after identifier!																																																	
File hello.go:		<pre>var foo int           // declaration without initialization var foo int = 42      // declaration with initialization var foo, bar int = 42, 1302 // declare/init multiple vars at once var foo = 42          // type omitted, will be inferred foo := 42             // shorthand, only in func bodies, implicit type const constant = "This is a constant"</pre>																																																	
package main		<b>Functions</b>																																																	
import "fmt"		<pre>// a simple function func functionName() {}</pre>																																																	
func main() {     fmt.Println("Hello Go") }		<pre>// function with parameters (again, types go after identifiers) func functionName(param1 string, param2 int) {}</pre>																																																	
\$ go run hello.go		<pre>// multiple parameters of the same type func functionName(param1, param2 int) {}</pre>																																																	
<b>Functions (cont)</b>		<b>Functions (cont)</b>																																																	
<pre>// return type declaration func functionName() int {     return 42 }  // Can return multiple values at once func returnMulti() (int, string) {     return 42, "foobar" } var x, str = returnMulti()  // Return multiple named results simply by return func returnMulti2() (n int, s string) {     n = 42     s = "foobar"     // n and s will be returned     return } var x, str = returnMulti2()</pre>		<b>Variadic Functions</b> <pre>func main() {     fmt.Println(add(1, 2, 3)) // 6     fmt.Println(add(9, 9))    // 18      nums := []int{10, 20, 30}     fmt.Println(add(nums...)) // 60 }  // Using ... before the type name of the last parameter indicates // that it takes zero or more of those parameters. // The function is invoked like any other function except we can // pass as many arguments as we want. func add(args ...int) int {     total := 0     for _, v := range args { // Iterate over all args         total += v     }     return total }</pre>																																																	
<b>Functions As Values And Closures</b>		<b>Built-in Types</b>																																																	
<pre>func main() {     // assign a function to a name     add := func(a, b int) int {         return a + b     }     // use the name to call the function     fmt.Println(add(3, 4)) }  // Closures, lexically scoped: Functions can access values that were // in scope when defining the function func scope() func() int{     outer_var := 2     foo := func() int { return outer_var }     return foo }  func another_scope() func() int{     // won't compile - outer_var and foo not defined in this scope     outer_var = 444     return foo }  // Closures: don't mutate outer vars, instead redefine them! func outer() (func(), int) {     outer_var := 2           // NOTE outer_var is outside inner's scope     inner := func() int {         outer_var += 99     // attempt to mutate outer_var         return outer_var   // =&gt; 101 (but outer_var is a newly redefined                             // variable visible only inside inner)     }     return inner, outer_var // =&gt; 101, 2 (still!) }</pre>		<p>bool string int int8 int16 int32 int64 uint uint8 uint16 uint32 uint64 uintptr byte // alias for uint8 rune // alias for int32 ~= (Unicode code point) - Very Viking float32 float64 complex64 complex128</p>																																																	
<b>Packages</b>		<b>Type Conversions</b>																																																	
		<pre>var i int = 42 var f float64 = float64(i) var u uint = uint(f)  // alternative syntax i := 42 f := float64(i) u := uint(f)</pre>																																																	
		<b>Control structures (cont)</b>																																																	

- package declaration at top of every source file
- executables are in package `main`
- convention: package name == last name of import path  
(import path `math/rand` => package `rand`)
- upper case identifier: exported (visible from other packages)
- Lower case identifier: private (not visible from other packages)

**Control structures**

```
If
func main() {
    // Basic one
    if x > 0 {
        return x
    } else {
        return -x
    }

    // You can put one statement before the condition
    if a := b + c; a < 42 {
        return a
    } else {
        return a - 42
    }

    // Type assertion inside if
    var val interface{}
    val = "foo"
    if str, ok := val.(string); ok {
        fmt.Println(str)
    }
}
```

Loops

```
// There's only 'for'. No 'while', no 'until'
for i := 1; i < 10; i++ {
}
for ; i < 10; { // while loop
}
for i < 10 { // can omit semicolons if there's only a condition
}
for { // can omit the condition ~ while (true)
}
```

**Switch**

```
// switch statement
switch operatingSystem {
case "darwin":
    fmt.Println("Mac OS Hipster")
    // cases break automatically, no fallthrough by default
case "linux":
    fmt.Println("Linux Geek")
default:
    // Windows, BSD, ...
    fmt.Println("Other")
}

// As with for and if, an assignment statement before the
// switch value is allowed
switch os := runtime.GOOS; os {
case "darwin": ...
}
```

**Arrays, Slices, Ranges**

Arrays

```
var a [10]int // int array with length 10. Length is part of type!
a[3] = 42 // set elements
i := a[3] // read elements

// declare and initialize
var a = [2]int{1, 2}
a := [2]int{1, 2} // shorthand
a := [...]int{1, 2} // ellipsis -> Compiler figures out array length
```

Slices

```
var a []int // a slice - like an array, but length is unspecified
var a = []int{1, 2, 3, 4} // declare and initialize a slice
// (backed by given array implicitly)
a := []int{1, 2, 3, 4} // shorthand
chars := [...]string{"a", "b", "c", "d"} // ["a", "b", "c"]

var b = a[lo:hi] // creates a slice (view of the array) from
// index lo to hi-1
var b = a[1:4] // slice from index 1 to 3
var b = a[:3] // missing low index implies 0
var b = a[3:] // missing high index implies len(a)

// create a slice with make
a = make([]byte, 5, 5) // first arg length, second capacity
a = make([]byte, 5) // capacity is optional
```

**Arrays, Slices, Ranges (cont)**

```
// create a slice from an array
x := [3]string{"Лайка", "Белка", "Стрелка"}
s := x[:] // a slice referencing the storage of x

Operations on Arrays and Slices
len(a) gives you the length of an array/a slice. It's a built-in function, not a attribute/method
on the array.

// loop over an array/a slice
for i, e := range a {
    // i is the index, e the element
}

// if you only need e:
for _, e := range a {
    // e is the element
}

// ...and if you only need the index
for i := range a {
}

// In Go pre-1.4, it is a compiler error to not use i and e.
// Go 1.4 introduced a variable-free form:
for range time.Tick(time.Second) {
    // do it once a sec
}
```

**Maps**

```
var m map[string]int
m = make(map[string]int)
m["key"] = 42
fmt.Println(m["key"])

delete(m, "key")

elem, ok := m["key"] // test if key "key" is present, retrieve if so

// map literal
var m = map[string]Vertex{
    "Bell Labs": {40.68433, -74.39967},
    "Google":    {37.42202, -122.08408},
}
```

**Structs**

There are no classes, only structs. Structs can have methods.

```
// A struct is a type. It's also a collection of fields

// Declaration
type Vertex struct {
    X, Y int
}

// Creating
var v = Vertex{1, 2}
var v = Vertex{X: 1, Y: 2} // Creates a struct by defining values
// with keys

// Accessing members
v.X = 4

// You can declare methods on structs. The struct you want to declare
// the method on (the receiving type) comes between the func keyword
// and the method name. The struct is copied on each method call()
func (v Vertex) Abs() float64 {
    return math.Sqrt(v.X*v.X + v.Y*v.Y)
}

// Call method
v.Abs()

// For mutating methods, you need to use a pointer (see below) to the
// struct as the type. With this, the struct value is not copied for
// the method call.
func (v *Vertex) add(n float64) {
    v.X += n
    v.Y += n
}

Anonymous structs
Cheaper and safer than using map[string]interface{}.

point := struct {
    X, Y int
}{1, 2}

Pointers
p := Vertex{1, 2} // p is a Vertex
q := &p // q is a pointer to a Vertex
r := &Vertex{1, 2} // r is also a pointer to a Vertex

// The type of a pointer to a Vertex is *Vertex
var s *Vertex = new(Vertex) // create ptr to a new struct instance
```

<p><b>Interfaces</b></p> <pre>// interface declaration type Awesomizer interface {     Awesomize() string }  // types do "not" declare to implement interfaces type Foo struct {}  // instead, types implicitly satisfy an interface if they implement // all required methods func (foo Foo) Awesomize() string {     return "Awesome!" }</pre> <p><b>Embedding</b></p> <p>There is no subclassing in Go. Instead, there is interface and struct embedding (composition).</p> <pre>// ReadWriter implementations must satisfy both Reader and Writer type ReadWriter interface {     Reader     Writer }  // Server exposes all the methods that Logger has type Server struct {     Host string     Port int     *log.Logger }  // initialize the embedded type the usual way server := &amp;Server{"localhost", 80, log.New(...)}  // methods implemented on the embedded struct are passed through server.Log(...) // calls server.Logger.Log(...)  // Field name of an embedded type is its type name ('Logger' here) var logger *log.Logger = server.Logger</pre> <p><b>Errors</b></p> <p>There is no exception handling. Functions that might produce an error just declare an additional return value of type <code>Error</code>. This is the <code>Error</code> interface:</p> <pre>type error interface {     Error() string }</pre>	<p><b>Errors (cont)</b></p> <p>A function that might return an error:</p> <pre>func doStuff() (int, error) { }  func main() {     result, error := doStuff()     if (error != nil) {         // handle error     } else {         // all is good, use result     } }</pre> <p><b>Concurrency</b></p> <p><b>Goroutines</b></p> <p>Goroutines are lightweight threads (managed by Go, not OS threads). <code>go f(a, b)</code> starts a new goroutine which runs <code>f</code> (given <code>f</code> is a function).</p> <pre>// just a function (which can be later started as a goroutine) func doStuff(s string) { }  func main() {     // using a named function in a goroutine     go doStuff("foobar")      // using an anonymous inner function in a goroutine     go func(x int) {         // function body goes here     }(42) }</pre>
<p><b>Channels</b></p> <pre>ch := make(chan int) // create a channel of type int ch &lt;- 42             // Send a value to the channel ch. v := &lt;-ch            // Receive a value from ch  // Non-buffered channels block. Read blocks when no value is // available, write blocks if a value already has been written // but not read.  // Create a buffered channel. Writing to a buffered channels does // not block if less than &lt;buffer size&gt; unread values have been // written. ch := make(chan int, 100)  close(ch) // closes the channel (only sender should close)  // Read from channel and test if it has been closed // If ok is false, channel has been closed v, ok := &lt;-ch  // Read from channel until it is closed for i := range ch {     fmt.Println(i) }  // select blocks on multiple channel operations. // If one unblocks, the corresponding case is executed func doStuff(channelOut, channelIn chan int) {     select {     case channelOut &lt;- 42:         fmt.Println("We could write to channelOut!")     case x := &lt;- channelIn:         fmt.Println("We could read from channelIn")     case &lt;time.After(time.Second * 1):         fmt.Println("timeout")     } }</pre> <p><b>Channel Axioms</b></p> <pre>// I. A send to a nil channel blocks forever var c chan string c &lt;- "Hello, World!" // fatal error: all goroutines are asleep - deadlock!  // II. A receive from a nil channel blocks forever var c chan string fmt.Println(&lt;-c) // fatal error: all goroutines are asleep - deadlock!  // III. A send to a closed channel panics var c = make(chan string, 1) c &lt;- "Hello, World!" close(c)</pre>	<p><b>Channels (cont)</b></p> <pre>c &lt;- "Hello, Panic!" // panic: send on closed channel  // IV. A receive from a close channel returns the zero value // immediately var c = make(chan int, 2) c &lt;- 1 c &lt;- 2 close(c) for i := 0; i &lt; 3; i++ {     fmt.Printf("%d ", &lt;-c) } // 1 2 0</pre> <p><b>Snippets</b></p> <p><b>HTTP Server</b></p> <pre>package main  import (     "fmt"     "net/http" )  // define a type for the response type Hello struct{}  // let that type implement the ServeHTTP method (defined in // interface http.Handler) func (h Hello) ServeHTTP(w http.ResponseWriter, r *http.Request) {     fmt.Fprintf(w, "Hello!") }  func main() {     var h Hello     http.ListenAndServe("localhost:4000", h) }  // Here's the method signature of http.ServeHTTP: // type Handler interface { //     ServeHTTP(w http.ResponseWriter, r *http.Request) // }</pre>

## HTML的cheat sheet

[htmlcheatsheet.pdf](#)

[HTML Cheat Sheet - A Simple Guide to HTML](#)

[HTML5 Cheat Sheet - WebsiteSetup.org](#)

[HTML5 Cheat Sheet - WebsiteSetup.org](#)



<b>HTML 5 NEW TAG</b>		<b>TAG NOT SUPPORTED IN HTML 5</b>	
<!--><!-- Define a comment</td> <td data-kind="ghost"></td> <td data-cs="2" data-kind="parent">&lt;comment&gt;</td> <td data-kind="ghost"></td>		<comment>	
<!DOCTYPE> Define the document type		<!DOCTYPE>	
<a> href, hreflang, media, ping, rel, target, type Define a hyperlink		<a>	
<abbr> Defines an abbreviation		<abbr>	
<acronym> Used to define an embedded acronym		<acronym>	
<address> Defines an address element		<address>	
<applet> Used to define an embedded applet		<applet>	
<area> Defines an area inside an image map alt, coords, href, hreflang, media, ping, rel, shape, target, type		<area>	
<article> Defines an article cite, pubdate		<article>	
<aside> Defines content aside from the page content		<aside>	
<audio> Defines sound content autobuffer, autoplay, controls, src		<audio>	
<b> Defines bold text		<b>	
<base> Defines a base URL for all the links in a page href, target		<base>	
<basefont> Used to define a default font-color, font-size, or font-family for all the document		<basefont>	
<bd> Defines the direction of text display dir		<bd>	
<big> Used to make text bigger		<big>	
<blockquote> Defines a long quotation cite		<blockquote>	
<body> Defines the body element		<body>	
  Inserts a single line break		 	
<button> autofocus, disabled, form, formaction, formenctype, formmethod, formnovalidate, formtarget, name, type, value		<button>	
<canvas> Defines graphics height, width		<canvas>	
<caption> Defines a table caption		<caption>	
<center> Used to center align text and content		<center>	
<cite> Defines a citation		<cite>	
<code> Defines computer code text autobuffer, autoplay, controls, src		<code>	
<col> Defines attributes for table columns		<col>	
<colgroup> Defines groups of table columns span		<colgroup>	
<command> Defines a command button checked, disabled, icon, label, radiogroup, type		<command>	
<datalist> Defines a dropdown list		<datalist>	
<dd> Defines a definition description		<dd>	
<del> Defines deleted text cite, datetime		<del>	
<details> Defines details of an element open		<details>	
<dialog> Defines a dialog (conversation)		<dialog>	
<dfn> Defines a definition term		<dfn>	
<dir> Used to define a directory list		<dir>	
<div> Defines a section in a document		<div>	
<dl> Defines a definition list		<dl>	
<dt> Defines a definition term		<dt>	
<em> Defines emphasized text		<em>	
<embed> Defines external interactive content or plugin height, src, type, width		<embed>	
<fieldset> Defines a fieldset disabled, form, name		<fieldset>	
<figure> Defines a group of media content, and their caption		<figure>	
<font> Used to define font-face, font-size, and font-color of text		<font>	
<footer> Defines a footer for a section or page		<footer>	
<form> accept=charset, action, autocomplete, enctype, method, name, novalidate, target		<form>	
<frame> Used to define one particular window (frame) within a frameset		<frame>	
<frameset> Used to define a frameset, which organizes multiple windows (frames)		<frameset>	
<h1> to <h6> Defines header 1 to header 6		<h1> to <h6>	
<head> Defines information about the document		<head>	
<header> Defines a header for a section or page		<header>	
<hgroup> Defines information about a section in a document		<hgroup>	
<hr> Defines a horizontal rule		<hr>	
<html> Defines an html document manifest, smlas		<html>	
<i> Defines italic text		<i>	
<iframe> Defines an inline subwindow height, name, sandbox, seamless, src, width		<iframe>	
<img> Defines an image alt, src, height, ismap, usemap, width		<img>	
<input> Defines a form input field accept, alt, autocomplete, autofocus, checked, disabled, form, formaction, formenctype, formmethod, formnovalidate, formtarget, list, max, minlength, min, name, pattern, placeholder, readonly, required, size, step, type, value, width		<input>	
<ins> Defines inserted text cite, datetime		<ins>	
<keygen> Defines a generated key in a form autofocus, challenge, disabled, form, keytype, name		<keygen>	
<kbd> Defines keyboard text		<kbd>	
<label> Defines an inline subwindow for, form		<label>	
<legend> Defines a title in a fieldset		<legend>	
<li> Defines a list item value		<li>	
<link> Defines a resource reference href, hreflang, media, rel, sizes, type		<link>	
<map> Defines an image map name		<map>	
<mark> Defines marked text		<mark>	
<menu> Defines a menu list label, type		<menu>	
<meta> Defines meta information charset, content, http-equiv, name		<meta>	
<meter> Defines measurement within a predefined range high, low, max, min, optimum, value		<meter>	
<nav> Defines navigation links		<nav>	
<noframes> Used to display text for browsers that do not handle frames		<noframes>	
<noscript> Defines a noscript section		<noscript>	
<object> Defines an embedded object data, form, height, name, type, usemap, width		<object>	
<ol> Defines an ordered list reversed, start		<ol>	
<optgroup> Defines an option group label, disabled		<optgroup>	
<option> Defines an option in a drop-down list disabled, label, selected, value		<option>	
<output> Defines some types of output for, form, name		<output>	
<p> Defines a paragraph		<p>	
<param> Defines a parameter for an object name, value		<param>	
<pre> Defines preformatted text		<pre>	
<progress> Defines progress of a task of any kind max, value		<progress>	
<q> Defines a short quotation cite		<q>	
<rp> Used in ruby annotations to define what to show browsers that do not support the ruby element		<rp>	
<rt> Defines explanation to ruby annotations		<rt>	
<rb> Defines ruby annotations		<rb>	
<s>, <del> Used to define strikethrough text		<s>, <del>	
<video> Defines a video autobuffer, autoplay, controls, height, loop, src, width		<video>	

**HTML5 TAG CHEAT SHEET**  
Created by WebsiteSetup.org

## Font Awesome 的Cheatsheet

React Native折腾期间，需要利用到对应字体：[fortawesome.github.io/Font-Awesome/icons/](http://fortawesome.github.io/Font-Awesome/icons/)

-»

Font Awesome Cheatsheet

就列出了所有的字体图标

Font Awesome 4.7.0 Icon, CSS Class, & Unicode

Icon	CSS Class	Unicode	Icon	CSS Class	Unicode	Icon	CSS Class	Unicode
fa-500px	[&#xf26e;]	4.4	fa-address-book	[&#xf2b9;]	4.7	fa-address-book-o	[&#xf2ba;]	4.7
fa-address-card-o	[&#xf2bc;]	4.7	fa-adjust	[&#xf042;]	4.7	fa-adn	[&#xf170;]	4.7
fa-align-justify	[&#xf039;]	4.7	fa-align-left	[&#xf036;]	4.7	fa-align-right	[&#xf038;]	4.7
fa-ambulance	[&#xf0f9;]	4.7	fa-american-sign-language-interpreting	[&#xf2a3;]	4.6	fa-anchor	[&#xf13d;]	4.2
fa-angle-double-left	[&#xf100;]	4.4	fa-angle-double-right	[&#xf101;]	4.4	fa-angellist	[&#xf209;]	4.2
fa-angle-left	[&#xf104;]	4.4	fa-angle-right	[&#xf105;]	4.4	fa-angle-up	[&#xf102;]	4.0
fa-archive	[&#xf187;]	4.0	fa-area-chart	[&#xf1fe;]	4.2	fa-arrow-circle-down	[&#xf0ab;]	4.0
fa-arrow-circle-o-down	[&#xf01a;]	4.0	fa-arrow-circle-o-left	[&#xf190;]	4.0	fa-arrow-circle-o-right	[&#xf18e;]	4.0
fa-arrow-circle-right	[&#xf0a9;]	4.0	fa-arrow-circle-up	[&#xf0aa;]	4.0	fa-arrow-down	[&#xf063;]	4.0
fa-arrow-right	[&#xf061;]	4.0	fa-arrow-up	[&#xf062;]	4.0	fa-arrows	[&#xf047;]	4.6
fa-arrows-h	[&#xf07e;]	4.0	fa-arrows-v	[&#xf07d;]	4.6	fa-arrows-alt	[&#xf102;]	4.6
fa-asterisk	[&#xf069;]	4.2	fa-at	[&#xf1fa;]	4.2	fa-assistive-listening-systems	[&#xf2a2;]	4.6
fa-backward	[&#xf04a;]	4.2	fa-balance-scale	[&#xf24e;]	4.4	fa-automobile	(alias) [&#xf1b9;]	4.1
fa-bank	(alias) [&#xf19c;]	4.1	fa-bar-chart	[&#xf080;]	4.4	fa-bandcamp	[&#xf2d5;]	4.7
fa-bars	[&#xf0c9;]	4.1	fa-bath	[&#xf2cd;]	4.7	fa-barcode	[&#xf02a;]	4.4
fa-battery-0	(alias) [&#xf244;]	4.4	fa-battery-1	(alias) [&#xf243;]	4.4	fa-battery	(alias) [&#xf240;]	4.4
fa-battery-4	(alias) [&#xf240;]	4.4	fa-battery-empty	[&#xf244;]	4.4	fa-battery-3	(alias) [&#xf241;]	4.4
fa-battery-quarter	[&#xf243;]	4.4	fa-battery-three-quarters	[&#xf241;]	4.4	fa-battery-half	[&#xf242;]	4.4
fa-be-hance	[&#xf1b4;]	4.1	fa-behance-square	[&#xf1b5;]	4.1	fa-beer	[&#xf10f;]	4.1
fa-bell-slash	[&#xf1f6;]	4.2	fa-bell-slash-o	[&#xf1f7;]	4.2	fa-bell-o	[&#xf0e2;]	4.2
fa-birthday-cake	[&#xf1fd;]	4.2	fa-bitbucket	[&#xf171;]	4.6	fa-binoculars	[&#xf1e5;]	4.2
fa-black-tie	[&#xf27e;]	4.4	fa-blind	[&#xf29d;]	4.6	fa-bitcoin	(alias) [&#xf15a;]	4.5
fa-bold	[&#xf0f2;]	4.4	fa-blush	[&#xf171;]	4.1	fa-bluetooth	[&#xf294;]	4.5

## React 的cheat sheet

### React Cheat Sheet

React Cheat Sheet

A Javascript library for building user interfaces.

DEMO: <https://s.codepen.io/ericnakagawa/debug/ALxakj>  
 GITHUB: <https://github.com/facebook/react>  
 DOCUMENTATION: <https://facebook.github.io/react/docs/>  
 CDN: <https://cdnjs.com/libraries/react>

**Hello World**

```
// Import React and ReactDOM
import React from 'react'
import ReactDOM from 'react-dom'

// Render component into the DOM - only once per app
ReactDOM.render(
  <h1>Hello, world!</h1>,
  document.getElementById('root')
);
```

**Stateless Components**

```
// Stateless React Component
const Headline = () => {
  return <h1>React Cheat Sheet</h1>
}

// Component that receives props
const Greetings = (props) => {
  return <p>You will love it {props.name}.</p>
}

// Component must only return ONE element (eg. DIV)
const Intro = () => {
  return (
    <div>
      <h1> /</h1>
      <p>Welcome to the React world!</p>
      <Greetings name="Petr" />
    </div>
  )
}

ReactDOM.render(
  <Intro />,
  document.getElementById('root')
);

// Components and Props API - http://bit.ly/react-props
// CodePen Demo: http://bit.ly/react-simple
```

**ES6 Class**

```
// use class for local state and lifecycle hooks
class App extends React.Component {

  constructor(props) {
    // fires before component is mounted
    super(props); // makes this refer to this component
    this.state = {date: new Date()}; // set state
  }

  render() {
    return (
      <h1>
        It is {this.state.date.toLocaleTimeString()}.
      </h1>
    )
  }

  componentWillMount() {
    // fires immediately before the initial render
  }

  componentDidMount() {
    // fires immediately after the initial render
  }

  componentWillReceiveProps() {
    // fires when component is receiving new props
  }

  shouldComponentUpdate() {
    // fires before rendering with new props or state
  }

  componentWillUpdate() {
    // fires immediately before rendering
    // with new props or state
  }

  componentDidUpdate() {
    // fires immediately after rendering with new P or S
  }

  componentWillUnmount() {
    // fires immediately before component is unmounted
    // from DOM (removed)
  }
}
```

**Core**

**Lifecycle Methods**

**Conditional Rendering**

```
// conditional rendering of elements and CSS class
render() {
  const {isLoggedIn, username} = this.state;
  return (
    <div className={`login ${isLoggedIn ? 'is-in' : 'is-out'}`}>
      {!isLoggedIn ?
        <p>Logged in as {username}.</p>
        :
        <p>Logged out.</p>
      }
    </div>
  )
}

// CodePen Demo: http://bit.ly/react-if-statements
```

**Tools and Resources**

```
// Create React App
http://bit.ly/react-app
// React Dev Tools for Chrome
http://bit.ly/react-dev-tools
// React Code Snippets for Visual Studio Code
http://bit.ly/react-es6-snippets-for-vscode
// Babel for Sublime Text 3
http://bit.ly/babel-sublime
```

**Free Online Course React.js 101**

The Quickest Way To Get Started With React.js

<http://bit.ly/get-react-101>

Sign Up Now!

<https://www.ihatetomatoes.net>

## Emoji的cheat sheet

Emoji cheat sheet for GitHub, Basecamp and other services

The screenshot shows a web browser window with the URL [www.webpagefx.com/tools/emoji-cheat-sheet/](http://www.webpagefx.com/tools/emoji-cheat-sheet/). The page title is "EMOJI CHEAT SHEET". Below the title, there is a paragraph of text about emoji support across various platforms. There are also some social sharing buttons for GitHub, Twitter, and Google+.

Emoji emoticons listed on this page are supported on [Campfire](#), [GitHub](#), [Basecamp](#), [Redbooth](#), [Trac](#), [Flowdock](#), [Sprintly](#), [Kandan](#), [Textbox.io](#), [Kippt](#), [Redmine](#), [JabbR](#), [Trello](#), [Hall](#), [Qiita](#), [Zendesk](#), [Ruby China](#), [Grove](#), [Idobata](#), [NodeBB Forums](#), [Slack](#), [Streamup](#), [OrganisedMinds](#), [Hackpad](#), [Cryptbin](#), [Kato](#), [Reportedly](#), [Cheerful Ghost](#), [IRCCloud](#), [Dashcube](#), [MyVideoGameList](#), [Subrosa](#), [Sococo](#), [Quip](#), [And Bang](#), [Bonusly](#), [Discourse](#), [Ello](#), [Twemoji](#), [Awesome](#), [Got Chosen](#), [Flow](#), [ReadMe.io](#), [esa](#), [DBook](#), [Groups.io](#), [TeamworkChat](#), [Damn Bugs](#), [Let's Chat](#), [Buildkite](#), [ChatGrape](#), [Dokuwiki](#), [Usersnap](#), [Discord](#), [Status Hero](#), [Morfy](#), [Bitbucket](#), [Gitter](#), [Yellow](#), [YouTube](#), [Habitica](#) and [Mattermost](#).

However some of the emoji codes are not super easy to remember, so here is a little cheat sheet.  
→ Got a modern browser or Flash enabled? Click the emoji code and it will be copied to your clipboard.

Fork 1,501 Star 5,478 Tweet 120 G+

## People

👨‍🦰 :bowtie:	😊 :smile:	😊 :simple_smile:
👩‍🦰 :laughing:	😊 :blush:	😊 :smiley:
😌 :relaxed:	😊 :smirk:	😍 :heart_eyes:
😘 :kissing_heart:	😘 :kissing_closed_eyes:	😳 :flushed:
🥰 :relieved:	🥰 :satisfied:	😁 :grin:
😜 :wink:	😜 :stuck_out_tongue_winking_eye:	😜 :stuck_out_tongue_closed_eyes:
🤗 :grinning:	😘 :kissing:	😘 :kissing_smiling_eyes:
😛 :stuck_out_tongue:	😴 :sleeping:	😟 :worried:
😦 :frowning:	😱 :anguished:	😮 :open_mouth:
😬 :grimacing:	😕 :confused:	😯 :hushed:
😐 :expressionless:	😒 :unamused:	😅 :sweat_smile:
💦 :sweat:	😢 :disappointed_relieved:	😩 :weary:
🤔 :pensive:	😢 :disappointed:	😵 :confounded:

## Python的Pandas的cheatsheet

[Pandas Cheat Sheet for Data Science in Python \(article\) - DataCamp](#)

-»

[PandasPythonForDataScience.pdf](#)

## Python For Data Science Cheat Sheet

### Pandas Basics

Learn Python for Data Science interactively at [www.DataCamp.com](http://www.DataCamp.com)



### Pandas

The Pandas library is built on NumPy and provides easy-to-use data structures and data analysis tools for the Python programming language.



Use the following import convention:

```
>>> import pandas as pd
```

### Pandas Data Structures

#### Series

A one-dimensional labeled array capable of holding any data type



```
>>> s = pd.Series([3, -5, 7, 4], index=['a', 'b', 'c', 'd'])
```

#### DataFrame

Columns	Country	Capital	Population
Index	Belgium	Brussels	11190846
	India	New Delhi	1303171035
	Brazil	Brasilia	207847528

```
>>> data = {'Country': ['Belgium', 'India', 'Brazil'],
   'Capital': ['Brussels', 'New Delhi', 'Brasilia'],
   'Population': [11190846, 1303171035, 207847528]}

>>> df = pd.DataFrame(data,
   columns=['Country', 'Capital', 'Population'])
```

### I/O

#### Read and Write to CSV

```
>>> pd.read_csv('file.csv', header=None, nrows=5)
>>> df.to_csv('myDataFrame.csv')
```

#### Read and Write to Excel

```
>>> pd.read_excel('file.xlsx')
>>> pd.to_excel('dir/myDataFrame.xlsx', sheet_name='Sheet1')
Read multiple sheets from the same file
>>> xlxs = pd.ExcelFile('file.xlsx')
>>> df = pd.read_excel(xlxs, 'Sheet1')
```

### Asking For Help

```
>>> help(pd.Series.loc)
```

#### Selection

##### Getting

```
>>> s['b']
-5
>>> df[1]
   Country    Capital  Population
1  India      New Delhi     1303171035
2  Brazil     Brasilia     207847528
```

#### Also see NumPy Arrays

##### Get one element

Get subset of a DataFrame

### Selecting, Boolean Indexing & Setting

#### By Position

```
>>> df.iloc[[0], [0]]
   Belgium
0
```

#### By Label

```
>>> df.loc[[0], ['Country']]
   Belgium
0
```

#### By Label/Position

```
>>> df.ix[2]
   Country    Capital  Population
2  Brazil     Brasilia     207847528
```

#### Boolean Indexing

```
>>> s[(s > 1) | (s < -1) | (s > 2)]
>>> df[df['Population']>1200000000]
```

#### Setting

```
>>> s['a'] = 6
Set index a of Series s to 6
```

#### Get single value by row & column

Select single value by row & column labels

#### Select single row of subset of rows

#### Select a single column of subset of columns

#### Select rows and columns

#### Series s where value is not >1 or <-1 or >2

#### s where value is <-1 or >2

#### Use filter to adjust DataFrame

### Dropping

```
>>> s.drop(['a', 'c'])
```

Drop values from rows (axis=0)

```
>>> df.drop('Country', axis=1)
```

Drop values from columns (axis=1)

### Sort & Rank

```
>>> df.sort_index()
>>> df.sort_values(by='Country')
>>> df.rank()
```

Sort by labels along an axis  
Sort by the values along an axis  
Assign ranks to entries

### Retrieving Series/DataFrame Information

#### Basic Information

>>> df.shape	(rows,columns)
>>> df.index	Describe index
>>> df.columns	Describe DataFrame columns
>>> df.info()	Info on DataFrame
>>> df.count()	Number of non-NA values

#### Summary

>>> df.sum()	Sum of values
>>> df.cumsum()	Cumulative sum of values
>>> df.max() / df.min()	Minimum/maximum values
>>> df.idxmin() / df.idxmax()	Minimum/maximum index value
>>> df.describe()	Summary statistics
>>> df.mean()	Mean of values
>>> df.median()	Median of values

#### Applying Functions

>>> f = lambda x: x**2	Apply function
>>> df.apply(f)	Apply function element-wise

### Data Alignment

#### Internal Data Alignment

NA values are introduced in the indices that don't overlap:

```
>>> s3 = pd.Series([7, -2, 3], index=['a', 'c', 'd'])
>>> s + s3
a    10.0
b    NaN
c    5.0
d    7.0
```

#### Arithmetic Operations with Fill Methods

You can also do the internal data alignment yourself with the help of the fill methods:

```
>>> s3.add(s3, fill_value=0)
a    10.0
b    -5.0
c    5.0
d    7.0
```

```
>>> s.sub(s3, fill_value=2)
a    10.0
b    -7.0
c    3.0
d    5.0
```

```
>>> s.div(s3, fill_value=4)
a    10.0
b    -5.0
c    5.0
d    7.0
```

```
>>> s.mul(s3, fill_value=3)
a    10.0
b    -6.0
c    3.0
d    5.0
```

DataCamp

Learn Python for Data Science interactively

## Kotlin的cheatsheet

Kotlin 作为 Android 开发语言相比传统 Java 有什么优势? - 知乎

Kotlin Programming Language Cheat Sheet Part 1

Kotlin Programming Language Cheat Sheet Part 2

Kotlin		
Basic syntax, functions, and expressions of the Kotlin language		
<b>Basic syntax</b>	<b>Control structures</b>	<b>Variables</b>
<code>fun main(args: Array&lt;String&gt;) { }</code> Program entry point	<code>if (a &lt; b) a else b</code> Conditional	<code>val weather = "sunny"</code> Define a final, non-null variable
<code>fun meaningOfLife() = 42</code> Define a function	<code>possiblyNull ?: "default"</code> Elvis operator for working with nullable variables	<code>var weather = "sunny"</code> Define a non-null variable
<code>println("Hello World")</code> Call a function	<code>while (a &lt; b) { }</code> While loop	<code>var weather : String = "sunny"</code> Define a variable with an explicit non-null type
<code>myList.joinToString(separator = ", ")</code> Call a function with named parameters	<code>for (i in 1..10) { println(i) }</code> Iterate from 1 to 10 (inclusive)	<code>var weather : String? = null</code> Define a nullable variable
<code>{ x: Int, y: Int -&gt; x + y }</code> Define a lambda function	<code>for(i in 10 downTo 1) { println(i) }</code> Iterate from 10 to 1 (inclusive)	
<code>"Hello \${name}"</code> Interpolate strings		
<code>department?.building?.address</code> Null-safe property access	<b>Working with collections</b>	<b>Defining collections</b>
<code>val (name, int) = myPerson</code> Destructuring classes or collections	<code>myList[5]</code> Get an element from a collection	<code>arrayOf(1, 2, ...)</code> Create an array
	<code>myList.forEach { println(it) }</code> Call a method on all entries of a collection	<code>listOf(1, 2, ...)</code> Create an immutable list
	<code>val doubled = myList.map { it * 2 }</code> Map all entries of a collection	<code>mapOf("id" to "k", "name" to "Kotlin", ...)</code> Create an immutable map
	<code>val sorted = names.sortedBy { it.length }</code> Sort a collection	<code>setOf(1, 2, ...)</code> Create an immutable set
	<code>a + b</code> Create a union of two collections a and b	
	<code>a - b</code> Create a collection of all items of a that do not occur in b	
<b>Defining classes</b>		
<code>class Person { ... }</code> Define a class		
<code>class Person(name: String) { ... }</code> Define a class with a constructor		
<code>class Person(val name: String) { ... }</code> Set a property in the constructor		
<code>class MyList : ArrayList() { ... }</code> Inherit from another class or interface		
<code>enum class Direction { UP, DOWN }</code> Define an enum		
<code>data class Person(val name: String, val age: Int)</code> Define a data class, with automatic equals(), hashCode(), and toString()		

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2018-01-02 22:37:55

# 编译和链接

此处介绍编译和链接方面的通用的知识、概念和逻辑。

之前接触过嵌入式和Linux，断断续续接触过很多编译和链接的东西，主要是命令行界面中用 `makefile`、`gcc` 等各种工具去做编译和链接的工作。

后来又接触过移动端开发，包括 `xcode` 等IDE集成开发环境

**[info] 编辑器和IDE**

详见：【整理】文本编辑器和IDE集成开发环境

发现其实IDE背后都是利用了编译和链接方面的通用知识。

对此编译和链接方面的通用的知识、逻辑、概念，去尝试总结，以供参考。

## Xcode中底层是用swiftc去编译swift代码

之前在：

[已解决] Xcode项目编译出错：Command failed due to signal: Segmentation fault: 11

期间，通过错误的log中的：

```
/Applications/Xcode.app/Contents/Developer/Toolchains/XcodeDefault.xctoolchain/usr/bin/swiftc -incremental -module-name SalesApp -Dnone -sdk /Applications/Xcode.app/Contents/Developer/Platforms/iPhoneSimulator.platform/Developer/SDKs/iPhoneSimulator9.3.sdk -target i386-apple-ios9.0 -g -module-cache-path /Users/crifan/Library/Developer/Xcode/DerivedData/ModuleCache -Xfrontend -serialize-debugging-options -enable-testing -I /Users/crifan/Library/Developer/Xcode/DerivedData/SalesApp-fxzszwvttqreanqsgzqmztnatyyjjd/Build/Products/Debug-iphonesimulator -F /Users/crifan/Library/Developer/Xcode/DerivedData/SalesApp-fxzszwvttqreanqsgzqmztnatyyjjd/Build/Products/Debug-iphonesimulator -F /Users/crifan/dev/dev_root/daryun/Projects/Qoros/QorosSales/Sourcecode/SalesAppiOS/Carthage/Build/iOS -c -j4 -I /Users/crifan/dev/dev_root/daryun/Projects/Qoros/QorosSales/Sourcecode/SalesAppiOS/SalesApp/CrifanLibHttp.swift

...
/Users/crifan/dev/dev_root/daryun/Projects/Qoros/QorosSales/Sourcecode/SalesAppiOS/SalesApp/CrifanLibDemo.swift /Users/crifan/dev/dev_root/daryun/Projects/Qoros/QorosSales/Sourcecode/SalesAppiOS/SalesApp/AddTaskViewController.swift -output-file -map /Users/crifan/Library/Developer/Xcode/DerivedData/SalesApp-fxzszwvttqreanqsgzqmztnatyyjjd/Build/Intermediates/SalesApp.build/Debug-iphonesimulator/SalesApp.build/Objects-normal/i386/SalesApp-OutputFileMap.json -parseable-output -serialize-diagnostics -emit-dependencies -emit-module -emit-module-path /Users/crifan/Library/Developer/Xcode/DerivedData/SalesApp-fxzszwvttqreanqsgzqmztnatyyjjd/Build/Intermediates/SalesApp.build/Debug-iphonesimulator/SalesApp.build/Objects-normal/i386/SalesApp.swiftmodule -Xcc -I/Users/crifan/Library/Developer/Xcode/DerivedData/SalesApp-fxzszwvttqreanqsgzqmztnatyyjjd/Build/Intermediates/SalesApp.build/Debug-iphonesimulator/SalesApp.build/swift-overrides.hmap -Xcc -Iquote -Xcc -I/Users/crifan/Library/Developer/Xcode/DerivedData/SalesApp-fxzszwvttqreanqsgzqmztnatyyjjd/Build/Intermediates/SalesApp.build/Debug-iphonesimulator/SalesApp.build/SalesApp-generated-files.hmap -Xcc -I/Users/crifan/Library/Developer/Xcode/DerivedData/SalesApp-fxzszwvttqreanqsgzqmztnatyyjjd/Build/Intermediates/SalesApp.build/Debug-iphonesimulator/SalesApp.build/SalesApp-own-target-headers.hmap

-Xcc -I/Users/crifan/Library/Developer/Xcode/DerivedData/SalesApp-fxzszwvttqreanqsgzqmztnatyyjjd/Build/Products/Debug-iphonesimulator/include -Xcc -I/Users/crifan/Library/Developer/Xcode/DerivedData/SalesApp-fxzszwvttqreanqsgzqmztnatyyjjd/Build/Intermediates/SalesApp.build/Debug-iphonesimulator/SalesApp.build/DerivedSources/i386 -Xcc -I/Users/crifan/Library/Developer/Xcode/DerivedData/SalesApp-fxzszwvttqreanqsgzqmztnatyyjjd/Build/Intermediates/SalesApp.build/Debug-iphonesimulator/SalesApp.build/Debug-iphonesimulator/SalesApp.build/DerivedSources -Xcc -DDEBUG 1 -emit-objc-header -emit-objc-header-path /Users/crifan/Library/Developer/Xcode/DerivedData/SalesApp-fxzszwvttqreanqsgzqmztnatyyjjd/Build/Intermediates/SalesApp.build/Debug-iphonesimulator/SalesApp.build/Objects-normal/i386/SalesApp-Swift.h -import-objc-header /Users/crifan/dev/dev_root/daryun/Projects/Qoros/QorosSales/Sourcecode/SalesAppiOS/SalesApp/SalesApp-Bridging-Header.h -Xcc -working-directory/Users/crifan/dev/dev_root/daryun/Projects/Qoros/QorosSales/Sourcecode/SalesAppiOS
```

而知道：

用的swiftc这个编译器，传递的参数是：

- 要编译的是：各个swift文件

- 输出是：SalesApp
- 期间利用到的各种：
  - SDK库是： -sdk /Applications/Xcode.app/Contents/Developer/Platforms/iPhoneSimulator.platform/Developer/SDKs/iPhoneSimulator9.3.sdk
  - include的头文件的路径是： -I xxx

等等

即：

对于Xcode这个IDE，你点击了编译按钮后，内部的实际过程是：

调用swiftc这个编辑器去swift代码，期间涉及到

- 引用哪些库
- 设置哪些编译参数
- 引用include哪些头文件: xxx.h
- 包含哪些路径
  - 去哪些路径中寻找头文件： -I xxx
  - 去哪些路径中寻找所要包含的库文件:-L xxx
- 设置哪些环境变量
  - 供后续过程去调用

-> 而这些都是编译和链接方面的通用知识。

-> 而其他的IDE的功能的内部过程大同小异，只不过用了不同的编译器，链接参数不同而已。

-> 而编译链接方面的知识，越是接近于底层开发，越是接触的多，理解的深入，比如嵌入式开发，Linux开发等等

-> 而对于编译和链接这方面的知识，如果之前折腾过嵌入式，Linux开发等方面的内容，则更能深切的体会到各种编译器、等内容。

## Xcode中通过设置头文件搜索路径而解决了头文件引用找不到的问题

比如：

【已解决】Xcode9代码出错：Realm/Realm.h file not found with angled include use quotes instead

期间，对于错误提示：

<Realm/Realm.h> file not found with angled include use quotes instead

如果对于编译链接知识不了解的话，则很容易被误导，而去一个个的修改（对应的几十个）swift文件，把：

<Realm/Realm.h>

改为：

Realm/Realm.h

但是却没能从根本上解决问题。实际上是，搜索头文件的路径却少了，需要去设置：

Headers Search Paths

把Realm的头文件的路径，添加进去，即可真正彻底解决头文件找不到的问题。

## Xcode中如何引入已改名的和已改变链接方式的库

之前在：

【已解决】Xcode9.2编译出错：ld library not found for -lMobClickLibrary

就遇到：

一个第三方库：友盟统计

- 之前是3.6.6的静态库：libMobClickLibrary.a
- 现在版本升级为4.0.1，且功能拆分为2个库文件了，且是动态库：
  - 统计功能的Analytics：UMMobClick.framework
  - 社交功能的Social

所以，搞清楚这点后，才知道去把Xcode中的：Build Settings->Linking->Other Linker Flags 中的：  
`-l"MobClickLibrary"` 改为：`-framework "UMMobClick"` 表示：引入新的 UMMobClick 的 framework 动态库，解决：`ld library not found for -lMobClickLibrary` 的问题。

## 苹果的swiftc编译器和 -j4 多线程编译

之前在：

【已解决】Xcode项目编译出错：Command failed due to signal: Segmentation fault: 11

期间，看到错误的log中的：

```
/Applications/Xcode.app/Contents/Developer/Toolchains/XcodeDefault.xctoolchain/usr/bin/swiftc -incremental -module-name SalesApp -Onone -sdk /Applications/Xcode.app/Contents/Developer/Platforms/iPhoneSimulator.platform/Developer/SDKs/iPhoneSimulator9.3.sdk -target i386-apple-ios9.0 -g -module-cache-path /Users/crifan/Library/Developer/Xcode/DerivedData/ModuleCache -Xfrontend -serialize-debugging-options -enable-testing -I /Users/crifan/Library/Developer/Xcode/DerivedData/SalesApp-fxz-zwvtqreanqsgzqmztnatyjjd/Build/Products/Debug-iphonesimulator -F /Users/crifan/Library/Developer/Xcode/DerivedData/SalesApp-fxzzwvtqreanqsgzqmztnatyjjd/Build/Products/Debug-iphonesimulator -F /Users/crifan/dev/dev_root/daryun/Projects/Qoros/QorosSales/Sourcecode/SalesAppiOS/Carthage/Build/ios -c -j4 /Users/crifan/dev/dev_root/daryun/Projects/Qoros/QorosSales/Sourcecode/SalesAppiOS/SalesApp/CrifanLibHttp.swift
```

其中有 `-j4`，猜测估计是类似于之前Linux中 makefile 中的 `make -j4`，估计是4个线程去并行执行的效果。

所以先去看看对应的编译器 `/Applications/Xcode.app/Contents/Developer/Toolchains/XcodeDefault.xctoolchain/usr/bin/swiftc`

结果是：

```
* crifanLib git:(master) swiftc --version
Apple Swift version 4.0.3 (swiftlang-900.0.74.1 clang-900.0.39.2)
Target: x86_64-apple-macosx10.9
```

果然是苹果的Swift语言的编译器。然后再去：

```
* crifanLib git:(master) which swiftc
/usr/bin/swiftc
```

可以看到，此处swiftc编译器是放在常见的 `/usr/bin` 目录下面的

再去通过：

```
* crifanLib git:(master) swiftc --help
...
-I value          Add directory to the import search path
-j n              Number of commands to execute in parallel
-L value          Add directory to library link search path
-l value          Specifies a library which should be linked against
...
```

```
-Xcc arg           Pass arg to the C/C++/Objective-C compiler  
-Xlinker value   Specifies an option which should be passed to the linker
```

中的： -j <n> Number of commands to execute in parallel

可以看到： -jN 表示用多少个命令并行执行

-> 和Make中的-j是类似的效果。

-> 另外，此处也说明了对于Xcode这个IDE，内部编译代码，链接库等操作，底层对应的逻辑，都是编译链接方面的知识，都是通用的。

-> 只不过用了不同的编译器，链接参数而异。

-> 而对于编译和链接这方面的知识，如果之前折腾过嵌入式，Linux开发等方面的内容，则更能深切的体会到各种编译器、库函数、参数、头文件、包含路径、环境变量等内容。

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间： 2018-01-02 23:39:20

# 嵌入式领域

此处介绍嵌入式领域内的一些通用知识、概念和逻辑。

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2018-01-02 15:18:26

# 上层软件领域

此处介绍上层软件领域内的通用知识、概念和逻辑。

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2018-01-02 15:19:08

# debounce防抖动

debounce的英文是防反跳，也叫防止误动作、防抖动

典型的用途有，键盘按键的去抖动，防抖动

类似的写前端页面时常会遇到类似的场景：

用户输入要搜索的文字而触发搜索，调用后台接口去返回数据，然后前端页面列表刷新显示

但是如果每次输入一个字符后就立刻触发后台搜索，则往往在一次性输入多个字符串时会导致中间的无效搜索，也会导致性能浪费和界面卡顿。

解决办法是：

当用户输入一个字符时，延迟触发搜索，比如延迟500ms：

- 在500ms内，如果用户没有在继续输入，再去调用接口返回搜索结果
- 在500ms内，如果用户又继续输入其他字符了，则重新触发，重新计算，再延迟500ms判断是否输入

由此可以实现：

不论用户是一次性只输入单个字符串，还是一段时间内连续输入多个字符串，最终都只是触发一次搜索，返回结果，刷新页面，使得用户体验很好。

对于上述逻辑，可以自己写代码实现：

在获得用户输入的事件时，设置一个定时器，计算倒计时。

也可以利用：

这个概念本身叫做debounce

有第三方库帮忙实现这个debounce的效果，比如lodash的debounce

比如之前写的：ReactJS中实现，延迟搜索的效果的代码：

```
import { h, Component } from 'preact';
import style from './style.less';
import PropTypes from 'prop-types';

import _ from "lodash";

export default class Search extends Component {
  state = {
    value: null
  }

  constructor(props) {
    super(props);

    this.onInput = this.onInput.bind(this);
    this.onClick = this.onClick.bind(this);
  }

  componentWillMount() {
    this.delayedOnInput = _.debounce(this.props.onInputChange, this.props.debounceTimeout);
  }

  onInput(e) {
    this.setState({value: e.target.value});

    e.persist();
    this.delayedOnInput(this.state.value);
  }
}
```

```
onClick(e) {
  if (this.props.onClick) {
    this.props.onClick(e);
  }
}

render () {
  return (
    <div class={style.cz_top_box}>
      <ul>
        <a>
          <li class={style.min_box}>
            <input
              type="search"
              value={this.state.value}
              placeholder={this.props.placeholder}
              onClick={this.onClick}
              onInput={this.onInput} />
          </li>
        </a>
      </ul>
    </div>
  );
}

Search.PropTypes = {
  debounceTimeout : PropTypes.number,
  placeholder : PropTypes.string,
  onInputChange : PropTypes.func.isRequired,
  onClick : PropTypes.func
};

Search.defaultProps = {
  debounceTimeout : 500,
  placeholder : "请输入"
};
```

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2018-01-02 22:54:24

# 库/包的依赖管理工具

在开发期间，往往需要使用别人的第三方的代码和功能。

而安装第三方库文件，理论上可以自己下载源码，自己编译得到自己要的库

但是后续的库的升级、管理依赖、维护都是个比较麻烦的事情。

所以出现了包管理工具，便于第三方库的安装和升级：

即：不同语言和环境，有不同的第三方的软件/库/包的管理工具，管理不同的版本，不同的依赖。

下面就来总结一下。

## Python

**pip**

## Javascript

**npm**

**yarn**

## iOS

**Cocoapods**

**Carthage**

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2018-01-02 23:31:09

# 上层软件 - 数据库

此处介绍上层软件领域内和数据库相关的通用知识、概念和逻辑。

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2018-01-02 15:20:22

# schema模式

之前在数据库领域内就看到和提到了schema

[MongoDB vs. Redis Comparison architecture - When to Redis? When to MongoDB? - Stack Overflow](#)

想要去搞清楚，什么是schema

什么是schema

[在数据库中， schema、catalog分别指的是什么？ - 知乎](#)

对于mysql， schema和database可以理解为等价的。

As defined in the MySQL Glossary: In MySQL, physically, a schema is synonymous with a database. You can substitute the keyword SCHEMA instead of DATABASE in MySQL SQL syntax, for example using CREATE SCHEMA instead of CREATE DATABASE.

Some other database products draw a distinction. For example, in the Oracle Database product, a schema represents only a part of a database: the tables and other objects owned by a single user.

schema就是数据库对象的集合，这个集合包含了各种对象如：表、视图、存储过程、索引等。

如果把database看作是一个仓库，仓库很多房间（schema），一个schema代表一个房间，table可以看作是每个房间中的储物柜，user是每个schema的主人，有操作数据库中每个房间的权利，就是说每个数据库映射user有每个schema（房间）的钥匙。

schema是对一个数据库的结构描述。在一个关系型数据库里面，schema定义了表、每个表的字段，还有表和字段之间的关系。catalog是由一个数据库实例的元数据组成的，包括基本表，同义词，索引，用户等等。”

[Difference Between Schema / Database in MySQL - Stack Overflow](#)

Depends on the database server. MySQL doesn't care, its basically the same thing. Oracle, DB2, and other enterprise level database solutions make a distinction. Usually a schema is a collection of tables and a Database is a collection of schemas.

[Schema（数据库中的Schema）\\_百度百科](#)

A schema is a collection of database objects (used by a user.). schema objects are the logical structures that directly refer to the database's data. A user is a name defined in the database that can connect to and access objects. schemas and users help database administrators manage database security.”

[数据库中Schema（模式）概念的理解](#)

在学习数据库时，会遇到一个让人迷糊的Schema的概念。实际上，schema就是数据库对象的集合，这个集合包含了各种对象如：表、视图、存储过程、索引等。

如果把database看作是一个仓库，仓库很多房间（schema），一个schema代表一个房间，table可以看作是每个房间中的储物柜，user是每个schema的主人，有操作数据库中每个房间的权利，就是说每个数据库映射的user有每个schema（房间）的钥匙。

默认情况下一个用户对应一个集合，用户的schema名等于用户名，并作为该用户缺省schema。所以schema集合看上去像用户名。访问一个表时，如果没有指明该表属于哪个schema，系统会自动加上缺省的schema。一个对象的完整名称为schema.object，而不属user.object。

在MySQL中创建一个Schema和创建一个Database的效果好像是一样的，但是在SQL Server和Oracle数据库中效果又是不同的。

01-02 23:27:27

# 各种编程语言领域

此处介绍各种计算机编程语言方面的通用逻辑、知识和概念。

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2018-01-02 15:22:09

## print打印

在用不同语言开发时，往往会遇到，用print类的函数去打印log输出信息，用于调试。

- 嵌入式中常见的C等语言的打印都是printf
- 写Linux驱动时，Linux内核中的打印是printk

TODO: 1.把 2.2.1. 通用函数: print -- 计算机编程语言基础知识合并进来

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2018-01-02 22:55:51

## log日志

在写代码开发期间，往往会涉及到打印Log日志。

在此总结log相关的内容。

```
日志 == log == Logger == logging
```

用来打印日志输出信息用于调试。

## log的等级

log日志有一些常见的登记，最基本的几种有：

- `info`：用于输出一些阶段性的告知用户的信息，比如用户已成功登录之类的
- `debug`：用于调试目的，而输出一些信息，比如打印某些变量的值
- `warn / warning`：用于输出一些不正常的情况，需要引起用户注意了，比如app检测存储空间快满了
- `error`：显示一些错误级别的信息，往往意味着程序也要终止运行了。需要用户处理错误，或者退出程序之类的

其他不是很常见的几种：

- `finest / fine`：表示程序正常级别输出的信息
- `trace`：类似于`debug`，用于输出一些追踪调试信息
- `verbose`：事无巨细的，最详尽的信息
- `severe`：严重的问题，类似于`error`
- `critical`：关键的问题，类似于`severe`

## log相关的库

### log4xxx系列

有个通用的log的库：log4xxx

针对于不同语言的log4xxx分别是：

- Java语言：`Log4j`
  - [log4j](#)
  - 新版本：[Log4j – Log4j 2 Guide - Apache Log4j 2](#)
- C语言：`log4c`
  - [log4c](#)
- Go语言：`log4go`
  - [alechthomas/log4go: Logging package similar to log4j for the Go programming language](#)
- 微软的.NET框架（C#等语言）：`log4net`
  - [Open Source Logging Tools in C#](#)
- Delphi语言：`Log4Delphi`
  - [Welcome to Log4Delphi](#)
- C++: `log4cpp`
  - 其他相关：
    - [log4cplus](#)
    - [log4cxx](#)

## Python

Python的log库：自带的库[logging](#)

自己也整理和封装成基本的函数：

```
def loggingInit(filename = None,
                fileLogLevel = logging.DEBUG,
                fileLogFormat = '%(asctime)s LINE %(lineno)-4d %(levelname)-7s %(message)s',
                fileLogDateFormat = '%Y/%m/%d %I:%M:%S',
                enableConsole = True,
                consoleLogLevel = logging.INFO,
                consoleLogFormat = "%(asctime)s LINE %(lineno)-4d %(levelname)-7s %(message)s",
                consoleLogDateFormat = '%Y/%m/%d %I:%M:%S',
                ):
    """
    init logging for both log to file and console
    :param logFilename: input log file name
        if not passed, use current script filename
    :return: none
    """
    logFilename = ""
    if filename:
        logFilename = filename
    else:
        logFilename = getInputFileBasenameNoSuffix() + ".log"

    logging.basicConfig(
        level = fileLogLevel,
        format = fileLogFormat,
        datefmt = fileLogDateFormat,
        filename = logFilename,
        filemode = 'w')
    if enableConsole :
        # define a Handler which writes INFO messages or higher to the sys.stderr
        console = logging.StreamHandler()
        console.setLevel(consoleLogLevel)
        # set a format which is simpler for console use
        formatter = logging.Formatter(fmt=consoleLogFormat, datefmt=consoleLogDateFormat)
        # tell the handler to use this format
        console.setFormatter(formatter)
        logging.getLogger('').addHandler(console)
```

详见：<https://github.com/crifan/crifanLib/blob/master/python/crifanLib.py>

## Flask

Python的Flask中也可以打印log

详见：

- [未解决] Flask中给app的输出到命令行中的debugger添加函数名
- [已解决] Flask中输出log日志到文件且自定义输出格式

## Swift

iOS的swift中也遇到过log库：[XCGLogger](#)

其中的log的定义是：

```
/// Enum defining our log levels
public enum Level : Int, Comparable, CustomStringConvertible {
    case verbose
    case debug
    case info
    case warning
```

```

    case error
    case severe
    case none

    ...
    public var description: String { get }
    public static let all: [XCGLogger.XCGLogger.Level]
}

```

和基本的封装

```

import XCGLogger

func initLog() {
    gLog = XCGLogger.default//XCGLogger.defaultInstance()

    //var logLevel:XCGLogger.LogLevel = XCGLogger.LogLevel.debug

    var logLevel:XCGLogger.Level = XCGLogger.Level.debug
    if LogLevelIsVerbose {
        logLevel = XCGLogger.Level.verbose
    }

    var fileLogLevel:XCGLogger.Level = XCGLogger.Level.debug
    if FileLogLevelIsVerbose {
        fileLogLevel = XCGLogger.Level.verbose
    }

    gLog.setup(
        level: logLevel,
        showThreadName: true,
        showLevel: true,
        showFileNames: true,
        showLineNumbers: true,
        writeToFile:LogFile,
        fileLevel: fileLogLevel)
}

```

之后即可正常log了：

```

gLog.info("UIScreen.mainScreen().bounds=\\"(UIScreen.main.bounds)"")
gLog.debug("forgetPasswordVC:\\"(forgetPasswordVC)"")
gLog.warning("jpush set alias \\(alias) failed")
gLog.error("上传附件失败: \\(encodingError)"")

```

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间： 2018-01-02 23:12:23

## async和wait

### Javascript中的async wait

JS的ES6/ES7? 中有async wait

### Python 3.3中也有async wait

待整理: [使用 WebSocket 和 Python 编写日志查看器 | 张吉的博客](#)

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2018-01-02 23:22:21

## 类的属性的 getter 和 setter

之前接触计算机语言，比如 c 、 c++ 等，都没有 `getter` 和 `setter` 的

后来接触到了一些语言：

- C#
- Swift
- Python

其对象==类==Class中都有了属性Property的：

- `getter`：读取属性的值
- `setter`：设置（写入）属性的值

比如：

## Python

Python的[python - 新手学习Flask引用flask-login后登入错误原因不清 - SegmentFault](#) 中的：

```
class Admin(UserMixin, db.Model):
    __tablename__ = 'admin'
    id = db.Column(db.Integer, primary_key=True)
    username = db.Column(db.String(64), unique=True)
    password_hash = db.Column(db.String(128), unique=False)

    @property
    def password(self):
        raise AttributeError('不能直接获取明文密码!')

    @password.setter
    def password(self, password):
        self.password_hash = generate_password_hash(password)
```

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间： 2018-01-10 11:40:56

# 编程语言 - Python

此处介绍各种计算机编程语言中涉及 Python 方面的通用逻辑、知识和概念。

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2018-01-02 15:26:56

## adapter适配器

Python开发期间，在使用第三方数据库等场景时，常会遇到 `apater` 这个概念。

`driver`驱动器 ≈ `adapter`适配器 ≈ `binding`粘合剂 ≈ `wrapper`封装

`adapter`/`driver`/`binding`/`wrapper`，指的是：

把一个别的功能，封装后，提供操作该功能的接口

起到了，作为粘合剂，把其他的库和你写的代码粘合起来效果

TODO:

1.把

[【软件开发基础知识】binding和wrapper – 在路上](#)

合并进来。

## Python

### Python的QT的driver

QT的库，对于Python语言来说，也有对应的driver

### Python的PostgreSQL的driver

postgre的数据库，也有对应的driver

### Python的MongoDB的driver

mongoDB也是有Python的driver

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间：2018-01-02 23:21:03

## 附录

下面列出相关参考资料。

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2017-12-14 11:21:48

## 参考资料

- 学习 shell 有什么好书推荐? - 知乎
- Cheat sheet - Wikipedia, the free encyclopedia
- 销售易的CRM PaaS平台开发手册
- log4go - GoDoc
- Python - PostgreSQL wiki
- MongoDB Drivers — MongoDB Ecosystem
- 最好用的前端开发工具之构建篇: Gulp\_HTML5|CSS3\_UDN技术社区
- StuQ - 一个新的学习方式, 提升你的IT职业技能

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2018-01-05 21:03:35