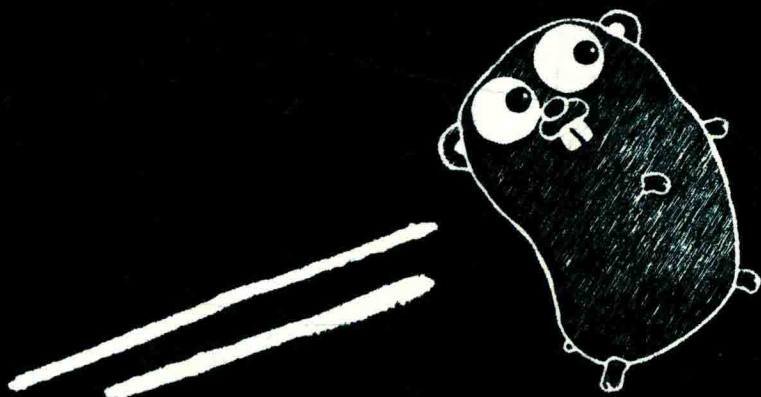


Broadview
www.broadview.com.cn

全栈开发者倾情奉献，全方位讲解Go语言的核心内容，
从源码层面了解其实现原理。



Go语言

编程入门与实战技巧

黄靖钧◎编著

体系完整：通过142个案例，囊括Go语言的主要功能

循序渐进：从基本语法到网络编程，条理清晰，逐步提高

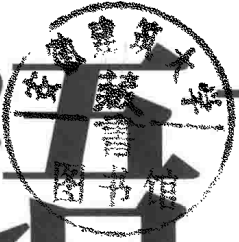
实用参考：包括18个常用标准库，深度讲解Go语言的实战技巧

原理剖析：深入理解Go语言源码实现，了解程序优化方法

 中国工信出版集团

 电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

Go语言



编程入门与实战技巧

黄靖钧◎编著

电子工业出版社
Publishing House of Electronics Industry
北京·BEIJING

内 容 简 介

本书从内容上分为三大部分，第一部分主要介绍 Go 语言的基础知识，包括 Go 语言的安装和开发工具，介绍了 Go 语言的特性与适合的场景，然后讲解了 Go 语言的程序结构和数据类型，并针对函数和一些关键字的用法与数据类型的调用原理做了阐述。第二部分介绍了 Go 语言数据结构和标准库，结合实际应用场景探讨了日常生产环境会遇到的问题与解决办法。第三部分主要介绍 Go 语言的测试工具和用法，并重点讲解了 Go 语言的内存管理机制，深入理解 Go 语言的设计哲学，了解 Go 语言底层的内存管理和并发机制，为更进一步的学习打下坚实的基础。

本书适合对计算机编程尤其是对 Go 语言编程感兴趣的新手作为入门教程阅读，还适合想在 Web 开发领域有所发展的程序员学习。

未经许可，不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有，侵权必究。

图书在版编目（CIP）数据

Go 语言编程入门与实战技巧 / 黄靖钧编著. —北京：电子工业出版社，2018.9

ISBN 978-7-121-34966-9

I. ①G… II. ①黄… III. ①程序语言—程序设计IV. ①TP312

中国版本图书馆 CIP 数据核字（2018）第 198989 号

策划编辑：张月萍

责任编辑：牛 勇 特约编辑：赵树刚

印 刷：三河市双峰印刷装订有限公司

装 订：三河市双峰印刷装订有限公司

出版发行：电子工业出版社

北京市海淀区万寿路 173 信箱 邮编：100036

开 本：787×980 1/16 印张：26.25 字数：551 千字

版 次：2018 年 9 月第 1 版

印 次：2018 年 9 月第 1 次印刷

定 价：79.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888，88258888。

质量投诉请发邮件至 zltz@phei.com.cn，盗版侵权举报请发邮件到 dbqq@phei.com.cn。

本书咨询联系方式：010-51260888-819，faq@phei.com.cn。

前言

当你打开这本书时，想必对 Go 语言是有一定兴趣的，本书希望能够让你真正喜欢上 Go 语言。本书面向的读者是对 Go 语言有一定了解但尚未入门的新手。

Go 语言是谷歌公司发布的一款开源编程语言，它对多处理器系统应用程序的编程进行了大量优化。作为新时代的代表性新生编程语言，Go 语言素有云计算时代的 C 语言之称。从 2009 年正式发布至今，Go 语言经过频繁的迭代更新走到今天，两次斩获 TIOBE 编程语言排行榜的年度语言称号（2009 年和 2016 年），已经跻身主流编程语言之列。

时至今日，Go 语言编译程序的速度可以媲美 C 或 C++ 程序，而且更加安全，支持并行进程。整个 Go 语言社区在诸多前辈们的推广与布道下，不断壮大，形成了今天富有活力的生态环境。

Go 语言在最近几年的更新中有很多富有里程碑意义的更新，例如实现了自举，Go 语言几乎完全使用 Go 语言重写了，仅保留一些基础的汇编代码。实现自举的好处有很多，一是提高了执行速度，可以避免跨语言调用或虚拟机等情况；二是提高了安全性，缩小了程序攻击面，使得整个项目更加可控；三是能够更轻易地实现跨平台编译，编写跨平台程序变得轻而易举。

除此之外，Go 语言的运行时系统和标准库等大量生产工具都得到了极大的提升，每一次版本迭代，Go 语言都会通过社区反馈权衡内存管理机制，并对调度器做出一些调整。在即将开始的 Go 语言学习过程中，相信你也能够深刻感受到 Go 语言开发团队的深思熟虑。

Go 语言还有一堆令人“爱不释手”的工具，令人惊艳的 go tool 宛如一把“瑞士军刀”，只需安装 Go 语言环境之后就可以直接使用。

重要的是，Go 语言社区生态成绩同样斐然，随着社区扩大，Go 语言包管理机制也有了很大改进，vendor 目录的启用标志着 Go 语言开发进入了一个更加规范的阶段。

整个社区里，拿得出手的杀手级应用（如 Moby、Kubernetes、Gogs、Grafana、Etc 等）喷涌而出。Go 语言累计接受了将近 4 万次的代码提交，近千名开发者参与到 Go 语言开发中。

本书的内容分为三大部分，第一部分主要介绍了 Go 语言的基础知识，包括 Go 语言的一些特性、适合的场景，以及 Go 语言的程序结构和数据类型，还讲解了函数和一些关键字的用法与数据类型的调用原理；第二部分介绍了 Go 语言常见的数据结构和常用的标准库，结合实际应用场景探讨了大部分生产环境会遇到的问题与解决办法；第三部分主要介绍了 Go 语言的测试与内存管理机制，深入理解 Go 语言的设计哲学，了解 Go 语言底层的内存管理和并发机制，为更进一步的学习打下坚实的基础。

本书的目标读者是对计算机编程尤其是对 Go 语言编程感兴趣的新手。由于作者水平和时间有限，书中难免会有一些错误和纰漏，欢迎读者指正。衷心希望通过本书的学习，能够让你对 Go 语言有一个比较全面的理解。

目录

第 1 章 认识 Go 语言	1
1.1 Go 语言简介	1
1.1.1 Go 语言简史	1
1.1.2 Go 语言特性	2
1.1.3 谁适合使用 Go 语言	2
1.2 Go 语言的开发环境部署	3
1.2.1 Go 语言环境变量	3
1.2.2 在 Linux 上安装 Go 语言环境	3
1.2.3 在 Mac OS 上安装 Go 语言环境	5
1.2.4 在 Windows 上安装 Go 语言环境	5
1.3 Go 语言的基本命令及使用	5
1.3.1 GOROOT 结构	6
1.3.2 GOPATH 结构	7
1.3.3 Go 语言命令行参数	8
1.3.4 第一个 Go 语言程序	12
1.4 Go 语言的开发工具	14
1.4.1 Code/Atom	14
1.4.2 VIM-Go	17
1.4.3 Gogland	18
1.4.4 LiteIDE	18
1.4.5 Cloud IDE	19
1.5 本章小结	23
第 2 章 程序结构	24
2.1 Go 语言程序元素	24

2.1.1	标识符.....	24
2.1.2	关键字.....	25
2.1.3	字面量.....	25
2.1.4	分隔符.....	25
2.1.5	运算符.....	26
2.1.6	注释.....	27
2.2	Go 语言基本概念.....	28
2.2.1	常量.....	28
2.2.2	变量.....	32
2.3	本章小结.....	37
第 3 章	基础数据类型.....	38
3.1	整型.....	39
3.1.1	整型的表示.....	39
3.1.2	整型的运算.....	39
3.2	浮点型.....	41
3.2.1	浮点型的表示.....	41
3.2.2	浮点型的运算.....	42
3.2.3	复数.....	42
3.3	字符与字符串.....	44
3.3.1	字符串的表示.....	44
3.3.2	操作字符串.....	46
3.3.3	字符串格式化.....	60
3.4	布尔型.....	61
3.4.1	布尔型的表示.....	61
3.4.2	布尔型的运算.....	62
3.5	基本数据类型的扩展.....	63
3.5.1	强制类型转换.....	63
3.5.2	自定义类型.....	64
3.5.3	类型别名.....	64
3.5.4	指针.....	65
3.6	本章小结.....	69
第 4 章	流程控制.....	70
4.1	条件语句.....	70

4.1.1、 if 判断	71
4.1.2 if-else 判断.....	71
4.1.3 else-if 判断.....	73
4.1.4 初始化子语句.....	74
4.2 选择语句.....	75
4.2.1 switch 语句	75
4.2.2 switch 初始化语句	78
4.2.3 select 语句	79
4.3 循环语句.....	80
4.3.1 for 的子语句	81
4.3.2 range 子语句.....	82
4.4 延迟语句.....	84
4.5 标签.....	87
4.5.1 break.....	87
4.5.2 continue.....	89
4.5.3 goto	90
4.6 本章小结.....	91
第 5 章 函数.....	92
5.1 认识函数.....	92
5.1.1 函数的声明.....	92
5.1.2 函数的参数.....	93
5.1.3 函数的返回值.....	94
5.2 函数的基础.....	95
5.2.1 多返回值	96
5.2.2 函数作为参数.....	97
5.2.3 函数作为类型.....	98
5.2.4 可变参数	99
5.2.5 匿名函数与闭包.....	102
5.2.6 递归函数	106
5.2.7 内置函数	108
5.3 函数进阶.....	108
5.3.1 参数传递机制.....	108
5.3.2 defer 与跟踪	111
5.3.3 错误与恢复.....	116
5.4 本章小结.....	122

第 6 章 复合数据类型	123
6.1 数组	123
6.1.1 声明数组	123
6.1.2 访问与修改	124
6.1.3 多维数组	126
6.1.4 将数组传递给函数	127
6.2 切片	128
6.2.1 创建数组切片	129
6.2.2 切片的使用	131
6.2.3 多维切片	137
6.2.4 将切片传递给函数	137
6.3 映射	138
6.3.1 映射的实现	138
6.3.2 映射的创建	139
6.3.3 映射的使用	139
6.3.4 将映射传递给函数	141
6.4 本章小结	142
第 7 章 包	144
7.1 包的基础	144
7.1.1 包的结构	146
7.1.2 包的导入	147
7.1.3 包的使用	153
7.1.4 Go 语言工具箱	154
7.2 自定义包	156
7.2.1 包的制作	157
7.2.2 特定平台的代码	157
7.2.3 godoc 生成文档	158
7.2.4 包的打包与发布	159
7.2.5 自定义包的导入	164
7.3 本章小结	165
第 8 章 结构体与方法	166
8.1 结构体	166
8.1.1 结构体定义	166

8.1.2	结构体使用	172
8.1.3	带标签的结构体	175
8.1.4	匿名字段和内嵌结构体	176
8.2	类型系统	179
8.2.1	用户自定义类型	179
8.2.2	值语义和引用语义	182
8.3	方法	184
8.3.1	方法声明	184
8.3.2	为类型添加方法	187
8.3.3	工厂方法创建结构体	188
8.3.4	基于指针对象的方法	190
8.3.5	方法值和方法表达式	191
8.3.6	方法和未导出字段	193
8.3.7	嵌入类型的方法和继承	194
8.4	本章小结	197
第 9 章	接口与反射	198
9.1	接口	198
9.1.1	接口是什么	199
9.1.2	接口类型与约定	204
9.1.3	接口实现	208
9.1.4	嵌套接口	210
9.1.5	接口赋值	211
9.1.6	接口查询	213
9.1.7	接口组合	214
9.2	反射	215
9.2.1	方法和类型的反射	215
9.2.2	通过反射修改设置值	216
9.2.3	反射结构	218
9.2.4	Printf 和反射	220
9.3	本章小结	222
第 10 章	并发编程	223
10.1	并发编程基础	223
10.1.1	并发与并行	223

10.1.2	指定使用核心数.....	225
10.2	协程 (goroutine)	227
10.2.1	协程基础.....	228
10.2.2	协程间通信.....	229
10.3	通道 (channel)	231
10.3.1	基本语法.....	232
10.3.2	select.....	232
10.3.3	缓冲机制.....	233
10.3.4	超时和计时器.....	234
10.3.5	channel 的传递.....	235
10.3.6	单向 channel	236
10.3.7	关闭 channel	237
10.4	并发进阶.....	237
10.4.1	多核并行化.....	237
10.4.2	协程同步.....	239
10.4.3	协程和恢复.....	242
10.5	本章小结.....	243
第 11 章	网络编程.....	244
11.1	Socket 编程.....	244
11.1.1	什么是 Socket.....	244
11.1.2	Dial()函数.....	246
11.1.3	ICMP 示例.....	247
11.1.4	TCP Socket.....	249
11.1.5	UDP Socket.....	253
11.2	HTTP 编程.....	255
11.2.1	HTTP 客户端.....	255
11.2.2	HTTP 服务器端.....	260
11.3	RPC 编程.....	262
11.3.1	Go RPC.....	263
11.3.2	HTTP RPC.....	263
11.3.3	TCP RPC.....	266
11.3.4	JSON RPC.....	268
11.3.5	RPC 接口.....	271
11.4	数据库.....	272

11.4.1	database/sql 接口	272
11.4.2	使用 MySQL 数据库	277
11.4.3	使用 SQLite 数据库	280
11.4.4	使用 PostgreSQL 数据库	282
11.4.5	NoSQL 数据库操作	286
11.5	Go 语言使用 Cookie	287
11.5.1	设置 Cookie	287
11.5.2	读取 Cookie	288
11.6	本章小结	288
第 12 章	I/O 编程	289
12.1	输入/输出	289
12.1.1	io: 基本 I/O 接口	289
12.1.2	fmt: 格式化 I/O	292
12.1.3	文本处理	298
12.2	文件系统	298
12.2.1	os: 系统功能实现	298
12.2.2	path: 兼容路径操作	299
12.3	数据结构与算法	303
12.3.1	排序	304
12.3.2	container	310
12.4	本章小结	314
第 13 章	文件处理	315
13.1	文件操作	315
13.1.1	创建文件与查看状态	316
13.1.2	重命名与移动	319
13.1.3	打开与关闭	319
13.1.4	删除与截断	321
13.1.5	读写文件	321
13.1.6	权限控制	325
13.1.7	文件链接	328
13.2	XML 处理	330
13.2.1	解析 XML	330
13.2.2	生成 XML	333

13.3	JSON 处理	336
13.3.1	解析 JSON	336
13.3.2	生成 JSON	338
13.4	日志记录	340
13.4.1	Logrus	341
13.4.2	Seelog	342
13.5	压缩	343
13.5.1	打包与解包	343
13.5.2	压缩与解压	345
13.6	本章小结	347
第 14 章	安全与测试	348
14.1	安全	348
14.1.1	安全相关的基础概念	348
14.1.2	通信安全	350
14.2	测试	354
14.2.1	单元测试	354
14.2.2	基准测试	362
14.3	本章小结	368
第 15 章	内存管理	369
15.1	内存分配	369
15.1.1	内存管理基本概念	369
15.1.2	逃逸分析	373
15.2	TCMalloc	376
15.2.1	整体结构	377
15.2.2	小内存分配	378
15.2.3	CentralCache	379
15.2.4	大内存分配	381
15.3	Mspan (内存管理器)	384
15.4	垃圾回收	398
15.4.1	标记清理算法	399
15.4.2	标记实现	403
15.4.3	清理	406
15.4.4	监控	408
15.5	本章小结	408

第 1 章

认识 Go 语言

本书开篇，先来了解 Go 语言的一些特性、历史以及优缺点；接下来根据不同系统介绍安装 Go 语言环境的方法，了解 Go 语言的基本命令以及开发 Go 语言的常用工具。

1.1 Go 语言简介

Go 语言是 Google 于 2009 年正式发布的一款开源的静态编译型编程语言。

1.1.1 Go 语言简史

Go 语言最早于 2007 年由 Rob Pike(贝尔实验室 UNIX 团队成员,曾参与 Plan 9、Inferno 和 Limbo 等项目)、Robert Griesemer (Java HotSpot 虚拟机、V8 引擎开发者之一)和 Ken Thompson(贝尔实验室 UNIX 团队成员, C 语言、UNIX 和 Plan 9 创始人之一,与 Rob Pike 共同开发了 UTF-8 字符集规范)三人在业余时间联合开发,后来还加入了 Ian Lance Taylor (GCC 核心开发人员)、Russ Cox (曾参与 Plan 9 操作系统的开发)和 Brad Fitzpatrick (memcached 的作者)等人。通过这几个人的履历就可以知道, Go 语言的创始团队对操作系统和系统编程语言有着非常深刻的理解,这对一门编程语言的开发起到至关重要的作用。

Google 于 2009 年 11 月 10 日正式发布 Go 语言,并以 BSD 协议完全开源,支持 Linux 和 Mac OS 平台,同年 11 月支持 Windows 平台。时至今日, Go 语言已经完成自举,社区生态成绩斐然,包括大量拿得出手的杀手级应用 (Moby、Kubernetes、Gogs、Grafana、Etc 等),两次夺得 TIOBE 年度编程语言称号 (2009 年和 2016 年)。Go 语言累计接受了 3 万多次代码提交,共有近 900 名开发者参与到 Go 语言项目中。

1.1.2 Go 语言特性

Go 语言作为一门静态类型的编译型语言，与当前的传统开发语言（如 Java、PHP）相比具备许多新特性。例如 Go 语言拥有自动垃圾回收功能，同时也允许开发人员干预回收操作；Go 语言有着更加丰富的内置类型，在错误处理方面语法更加精简高效。在 Go 语言中，函数支持多个返回值，而且函数也是一种值类型，可以作为参数传递。

从 `struct` 关键字就可以看出，Go 语言的类型定义参考了 C 语言中的结构（`struct`），但是 Go 语言并不像 C++ 和 Java 那样设计一个庞杂的类型系统，而是仅支持最基本的类型组合，不支持继承和重载。虽然 Go 语言没有类和继承的概念，但是它通过接口（`interface`）的概念来实现多态性。

Go 语言对多核处理器的编程进行了优化，Go 语言从程序与结构方面来实现并发编程，这是 Go 语言最重要的特性之一。

Go 语言和其他语言一样，拥有一个健全的包管理机制，同时得益于包之间的树状依赖，Go 语言的初次编译速度媲美 C/C++，甚至二次编译的速度明显快于 C/C++，同时又拥有接近 Python 等解释语言的简洁和开发效率。Go 语言在执行速度、编译速度和开发效率之间做了权衡，尽量达到了快速编译，高效执行，易于开发的目标。

最后，Go 语言支持交叉编译，可以在运行 Linux 系统的计算机上开发 Windows 下的应用程序。Go 语言源码文件格式默认都是使用 UTF-8 编码的。

1.1.3 谁适合使用 Go 语言

编程语言说到底是一种工具，不选最好的，只选最适合的，那么 Go 语言适合哪些场景？

首先使用 Go 语言，可以让 Web 服务器端的开发变得更高效，能够充分发挥多核计算机的性能，拥有更出色的网络环境兼容能力。自动垃圾回收、类型安全、依赖严格、编译快速等特点都是 Go 语言的魅力所在。很显然，Go 语言的目标就是针对服务器端的 Web 开发领域。

Go 语言凭借其出色的并发能力，在高性能分布式系统领域如鱼得水，像集群系统、游戏服务器端等场景都可以把 Go 语言作为首选开发语言。但是，Go 语言并不适合开发强实时性的软件，垃圾回收和自动内存分配等因素导致 Go 语言在实时性上有些力不从心。

1.2 Go 语言的开发环境部署

目前 Go 语言支持 Linux、FreeBSD、Mac OS 和 Windows 平台，安装包可以从 <https://golang.org/dl> 下载。

1.2.1 Go 语言环境变量

与 Java 等编程语言一样，安装 Go 语言开发环境需要设置全局的操作系统环境变量（除非使用包管理工具直接安装）。

主要的系统级别的环境变量有如下这些。

- \$GOROOT：表示 Go 语言环境在计算机上的安装位置，它的值可以是任意你喜欢的位置（方便管理为原则），这个变量只有一个值，值的内容必须是绝对路径。
- \$GOPATH：这是 Go 语言的工作目录，可以有多个，类似于工作空间的概念。一般不建议将 \$GOPATH 和 \$GOROOT 设置为同一个目录。

1.2.2 在 Linux 上安装 Go 语言环境

首先下载 Linux 的安装包，如图 1-1 所示，选择 Linux 安装包。

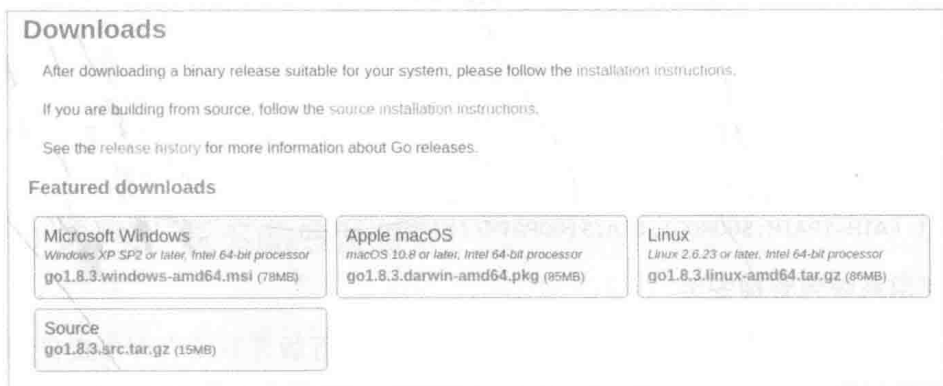


图 1-1 下载界面

下载之后，本地解压：

```
$ tar xf go1.8.3.linux-amd64.tar.gz
```


1. 当前用户

解压之后可以看到一个名为 go 的文件夹，把这个文件夹移动到你喜欢的方便管理的位置，然后编辑 \$HOME/.profile 文件：

```
$ vim $HOME/.profile
```

在文件末尾加入如下内容：

```
export GOROOT=$HOME/Applications/go/  
export GOPATH=$HOME/Workspace/Go/:$HOME/other/Go/  
export PATH=$PATH:$GOROOT/bin:${GOPATH//://bin:}/bin
```

上面的例子中，把 Go 语言环境放在 \$HOME/Applications/go/ 目录中，同时把 \$HOME/Workspace/Go/ 和 \$HOME/other/Go/ 设置为 Go 语言的工作目录。虽然 \$GOPATH 可以设置多个，但以后包管理安装时默认使用第一个 \$GOPATH 的值作为下载目录，建议把 \$GOPATH 第一个值设置为常用的全局工作目录，例如以后安装一些 Go 语言工具时，它们的 bin 执行文件可以在这里找到。

第三句是把前两个变量输出到系统 \$PATH 变量中，这样在当前用户环境中就可以随时使用 Go 语言环境了。

2. 所有用户

如果想让所有用户都能全局使用这个 Go 环境变量，可以把解压出来的 go 目录移动到相应的目录，并设置变量，例如：

```
$ mv go /usr/local/  
$ sudo vim /etc/profile  
export GOROOT=/usr/local/go/  
export GOPATH=$HOME/Workspace/Go/:$HOME/other/Go/  
export PATH=$PATH:$GOROOT/bin:${GOPATH//://bin:}/bin
```

3. 使用系统包管理安装

除了上面两种安装方式之外，还可以使用 Linux 发行版在软件仓库中提供的 Go 语言环境安装包，例如 Debian、Ubuntu 等发行版：

```
$ sudo apt install golang-go
```

其他发行版类似，一般软件包的名称不是 golang 就是 go，或者组合词。