CMP BOOKS 机工IT

详细讲述Go语言规范与语法细节及开发中常见的误区，通过研读标准库等经典代码设计模式，启发读者深刻理解Go语言的核心思维，进入Go语言开发的更高阶段。

信息科学与技术丛书

深入学习Go语言

李晓钧 编著

机械工业出版社
CHINA MACHINE PRESS

信息科学与技术丛书

深入学习 Go 语言

李晓钧　编著

机械工业出版社

Go 语言适合用来进行服务器编程与网络编程，包括 Web 应用编程等。本书详细讲解了 Go 语言数据类型、关键字、字面量、基本语法等基础概念及 Go 项目的工程构建、测试、编译与运行等；深入讲解了协程（goroutine）和通道（channel）等与并发编程有关的概念；还介绍了系统标准库、网络编程和第三方包。读者掌握本书内容后，可以顺利进行实际项目开发。

本书适合 Go 语言初学者和有一定经验的程序员阅读。

书中代码可免费下载（扫描封底二维码）。

图书在版编目（CIP）数据

深入学习 Go 语言 / 李晓钧编著. —北京：机械工业出版社，2019.8
（信息科学与技术丛书）
ISBN 978-7-111-63072-2

Ⅰ. ①深… Ⅱ. ①李… Ⅲ. ①程序语言—程序设计 Ⅳ. ①TP312

中国版本图书馆 CIP 数据核字（2019）第 126072 号

机械工业出版社（北京市百万庄大街 22 号 邮政编码 100037）
策划编辑：车 忱 责任编辑：车 忱
责任印制：张 博 责任校对：张艳霞

三河市国英印务有限公司印刷

2019 年 8 月第 1 版 • 第 1 次印刷
184mm×260mm • 16.75 印张 • 415 千字
0001—3000 册
标准书号：ISBN 978-7-111-63072-2
定价：69.00 元

电话服务 网络服务
客服电话：010-88361066 机 工 官 网：www.cmpbook.com
010-88379833 机 工 官 博：weibo.com/cmp1952
010-68326294 金 书 网：www.golden-book.com
封底无防伪标均为盗版 机工教育服务网：www.cmpedu.com

出 版 说 明

随着信息科学与技术的迅速发展，人类每时每刻都会面对层出不穷的新技术和新概念。毫无疑问，在节奏越来越快的工作和生活中，人们需要通过阅读和学习大量信息丰富、具备实践指导意义的图书来获取新知识和新技能，从而不断提高自身素质，紧跟信息化时代发展的步伐。

众所周知，在计算机硬件方面，高性价比的解决方案和新型技术的应用一直备受青睐；在软件技术方面，随着计算机软件的规模和复杂性与日俱增，软件技术不断地受到挑战，人们一直在为寻求更先进的软件技术而奋斗不止。目前，计算机和互联网在社会生活中日益普及，掌握计算机网络技术和理论已成为大众的文化需求。由于信息科学与技术在电工、电子、通信、工业控制、智能建筑、工业产品设计与制造等专业领域中已经得到充分、广泛的应用，所以这些专业领域中的研究人员和工程技术人员越来越迫切需要汲取自身领域信息化所带来的新理念和新方法。

针对人们了解和掌握新知识、新技能的热切期待，以及由此促成的人们对语言简洁、内容充实、融合实践经验的图书迫切需要的现状，机械工业出版社适时推出了“信息科学与技术丛书”。这套丛书涉及计算机软件、硬件、网络和工程应用等内容，注重理论与实践的结合，内容实用、层次分明、语言流畅，是信息科学与技术领域专业人员不可或缺的参考书。

目前，信息科学与技术的发展可谓一日千里，机械工业出版社欢迎从事信息技术方面工作的科研人员、工程技术人员积极参与我们的工作，为推进我国的信息化建设做出贡献。

机械工业出版社

前　言

现在市面上与 Go 语言相关的书籍较少，大部分书籍针对的是中高级开发人员，而从基础知识讲解，进而到初步应用开发的指导性书籍更少。

针对以上情况，本书详细讲解了 Go 语言基础知识点，并联系实际指出其可能存在的陷阱，帮助读者加深学习时的理解。本书还结合流行度较高的开源第三方包，引导读者进行更高级的实际项目开发。

本书非常适合 Go 语言新手细细阅读。有一定经验的开发人员，也可以根据自己的情况，选择一些章节来看。

第 1～4 章为基础部分，主要讲解 Go 语言的基础知识，包括 Go 语言的安装、基本语法、标识符、关键字、运算符、标点符号、字面量等，以及 Go 项目的工程构建、编译与运行等。

第 5～8 章为中级部分，主要讲解 Go 语言的复合数据类型，包括数组（array）、切片（slice）、字典（map）、结构体（struct）、指针（pointer）、函数（function）、接口（interface）和通道（channel）类型等。利用灵活的 type 关键字，可以自定义各种需要的数据类型。函数提供了更直接的数据处理能力，而通过 panic，recover，defer 处理错误的方式，也是 Go 语言的典型特征。

第 9～13 章为高级部分，主要讲解结构体、接口和方法，它们是 Go 语言简单与组合思维的基础。非常友好地支持并发是 Go 语言天然具有的典型特征，协程（gorountine）和通道（channel）配合，加上 sync 包提供的系列功能，使我们可以很方便地编写支持高并发的代码。

第 14～16 章为拓展部分，主要介绍 Go 语言提供的官方标准库，包括 OS 操作、文件 I/O、网络传输处理、指针相关操作、代码反射、日志记录等。这些包可以让我们快速进入实际开发。另外对 MySQL 数据库以及 LevelDB、BoltDB 数据库的操作有简单介绍。

第 17、18 章为应用部分，主要以网络爬虫和 Web 框架为例，进入实际开发。网络爬虫是互联网服务中比较重要的功能，通过互联网抓取、分析、保存资料是程序员的一项基本能力，读者可以看到 Go 语言在此方面也是游刃有余。而利用 Gin 这款轻量级的 Web 框架，可以很方便地搭建各种 Web 服务。

自 2009 年 Go 语言面世以来，已经有越来越多的公司转向 Go 语言开发。而 Go 语言以语法简单、学习门槛低、上手快著称，但入门后很多人发现要写出地道的、遵循 Go 语言思维的代码却实属不易。

我作为 Go 语言的爱好者，在阅读系统标准库源代码或其他知名开源包源代码时，发现大牛对这门语言的了解之深入，代码实现之巧妙优美，除了膜拜还是膜拜。所以我建议你有时间多多阅读这些代码，网上说 Go 大神的标准是“能理解简洁和可组合性哲学”。的确，Go 语言追求代码简洁到极致，而组合思想可谓借助于结构体和接口而成为 Go 的灵魂。

function、method、interface、type 等名词是程序员们接触比较多的关键字，但在 Go 语言中，你会发现，它们有更强大、更灵活的用法。当你彻底理解了 Go 语言相关基本概念，以及对其特点有了深入的认知（当然这也是这本书的目的），再假以时日多练习和实践，我相信你很快就能真正掌握这门语言，成为一名出色的 Gopher。

本书最早通过网络发布，有不少关注 Go 语言的朋友通过各种途径给了不少建议，这里要感谢网友 Joyboo、林远鹏、Mr_RSI、magic-joker 等。

本书最终得以出版，需要感谢李岩兄的鼓励和帮助，以及其他各位朋友和老师们，感谢你们的鼓励和帮助，感谢你们的支持！

最后，希望更多的人了解和使用 Go 语言，也希望阅读本书的朋友们多多交流。虽然本书中的例子都经过实际运行，但难免会有错误和不足之处，烦请您指出。书中其他疏漏之处也恳请各位读者斧正。作者联系邮箱：roteman@163.com。

祝各位 Gopher 工作开心，编码愉快！

李晓钧

目　录

第1章 Go语言简介

“Go 让我体验到了从未有过的开发效率。”谷歌资深工程师罗伯·派克（Rob Pike）这样说。他表示，“使用它可以进行快速开发，同时它还是一个真正的编译语言，我们之所以现在将其开源，原因是我们认为它已经非常有用和强大。”

1.1 为什么要学Go语言

Go 语言是一门全新的静态类型开发语言，具有自动垃圾回收、丰富的内置类型、函数多返回值、错误处理、匿名函数、并发编程、反射、defer 等关键特征，并具有简洁、安全、并行、开源等特性。从语言层面支持并发，可以充分利用 CPU 多核，Go 语言编译的程序可以媲美 C 或 C++代码的速度，而且更加安全、支持并行进程。系统标准库功能完备，尤其是强大的网络库使建立 Web 服务成为再简单不过的事情。Go 语言内置运行时，支持继承、对象等，开发工具丰富，例如 gofmt 工具能自动格式化代码，让团队代码风格完美统一。同时 Go 非常适合用来进行服务器编程、网络编程（包括 Web 应用、API 应用）和分布式编程等。

自 2009 年 Go 语言面世以来，已经有越来越多的公司转向 Go 语言开发，例如腾讯、百度、阿里、京东、小米、猎豹移动以及 360，而七牛云的技术栈基本上完全采用 Go 语言来开发。还有像今日头条、Uber 这样的公司，也使用 Go 语言对自己的业务进行了彻底重构。在全球范围内 Go 语言的使用不断增长，尤其是在云计算领域，用 Go 语言编写的几个主要云基础项目如 Docker 和 Kubernetes，都取得了巨大成功。除此之外，还有多种有名的项目如 etcd、consul、flannel 等，均使用 Go 语言实现。

Go 语言有“两快”，一是编译运行快，二是学习上手快。Go 语言的学习曲线并不陡峭，无论是刚开始接触编程的朋友，还是有其他语言开发经验而打算学习 Go 语言的朋友，大家都可以放心大胆来学习和了解 Go 语言，“它值得拥有！”

让我们开始 Go 语言学习之旅吧！

1.2 Go语言安装

要用 Go 语言来进行开发，需要先搭建开发环境。Go 语言支持以下系统：

- Linux
- FreeBSD
- Mac OS X（也称为 Darwin）
- Windows

首先需要下载 Go 语言安装包，下载地址为https://golang.org/dl/，国内下载地址是https://golang.google.cn/dl/。

1. 源码编译安装

Go 语言是谷歌在 2009 年发布的第二款开源编程语言。经过几年的版本更迭，目前 Go 已经

发布了 1.11 版本，UNIX/Linux/Mac OS X 和 FreeBSD 系统下可使用如下源码安装方法。

（1）下载源码包。链接是 https://golang.google.cn/dl/go1.11.1.linux-amd64.tar.gz。

（2）将下载的源码包解压至 /usr/local 目录：

tar -C /usr/local -xzf go1.11.1.linux-amd64.tar.gz

（3）将 /usr/local/go/bin 目录添加至 PATH 环境变量：

export PATH=$PATH:/usr/local/go/bin

（4）设置 GOPATH、GOROOT 环境变量。

GOPATH 是工作目录，GOROOT 是 Go 的安装目录，这里为/usr/local/go/。

> Mac 系统下可以使用以.pkg 为扩展名的安装包直接双击来完成安装，安装目录在 /usr/local/go/ 下。

2. Windows 系统下安装

在 Windows 系统下一般采用直接安装，下载 go 1.11.1.windows-amd64.zip 版本，直接解压到安装目录，如 D:\Go，然后把 D:\Go\bin 目录添加到 PATH 环境变量中。

另外，还需要设置两个重要环境变量：

GOPATH=D:\goproject

GOROOT=D:\Go\

以上环境变量设置好后，就可以使用 Go 语言来开发了。

Windows 系统也可以选择 go1.11.1.windows-amd64.msi，双击运行程序，根据提示来操作。

> GOPATH 是工作目录，可以有多个，用分号隔开。
> GOROOT 是安装目录。

按 Win+R 键打开命令行（注意：设置环境变量后需要重新打开命令行），输入 go，出现如下显示，说明 Go 语言运行环境已经安装成功。

```
D:\goproject\src>go
Go is a tool for managing Go source code.

Usage:

        go <command> [arguments]

The commands are:

        bug         start a bug report
        build       compile packages and dependencies
        clean       remove object files and cached files
        doc         show documentation for package or symbol
        env         print Go environment information
        fix         update packages to use new APIs
        fmt         gofmt (reformat) package sources
        generate    generate Go files by processing source
        get         download and install packages and dependencies
        install     compile and install packages and dependencies
        list        list packages or modules
        mod         module maintenance
```

```
        run             compile and run Go program
        test            test packages
        tool            run specified go tool
        version         print Go version
        vet             report likely mistakes in packages

Use "go help <command>" for more information about a command.

Additional help topics:

        buildmode       build modes
        c               calling between Go and C
        cache           build and test caching
        environment environment variables
        filetype        file types
        go.mod          the go.mod file
        gopath          GOPATH environment variable
        gopath-get      legacy GOPATH go get
        goproxy         module proxy protocol
        importpath      import path syntax
        modules         modules, module versions, and more
        module-get      module-aware go get
        packages        package lists and patterns
        testflag        testing flags
        testfunc        testing functions

Use "go help <topic>" for more information about that topic.
```

另外，输入 go version，可看到安装的 Go 版本信息，如图 1-1 所示。

```
D:\goproject\src>go version
go version go1.11 windows/amd64

D:\goproject\src>go env
set GOARCH=amd64
set GOBIN=
set GOCACHE=C:\Users\hawki\AppData\Local\go-build
set GOEXE=.exe
set GOFLAGS=
set GOHOSTARCH=amd64
set GOHOSTOS=windows
set GOOS=windows
set GOPATH=D:\goproject
set GOPROXY=
set GORACE=
set GOROOT=D:\Go
set GOTMPDIR=
set GOTOOLDIR=D:\Go\pkg\tool\windows_amd64
set GCCGO=gccgo
set CC=gcc
set CXX=g++
set CGO_ENABLED=1
set GOMOD=
set CGO_CFLAGS=-g -O2
set CGO_CPPFLAGS=
set CGO_CXXFLAGS=-g -O2
set CGO_FFLAGS=-g -O2
set CGO_LDFLAGS=-g -O2
set PKG_CONFIG=pkg-config
```

图 1-1　go version 和 go env 命令

图 1-1 中，输入 go env 命令可以看到用户自己的相关 Go 环境变量，这里重点关注 GOPATH 和 GOROOT，其他变量暂不用深入了解。

在本书中，所有代码和标准库的讲解都基于 Go 1.11 版本，还没有升级的用户请及时升级。

$GOPATH 允许有多个目录，当有多个目录时，请注意分隔符，Windows 中的分隔符是分号“;”。当有多个$GOPATH 时默认将 go get 命令获取的包存放在第一个目录下。

$GOPATH 目录下约定有三个子目录。

- src 存放源代码（如.go, .c, .h, .s 等文件）。按照 Go 默认约定，src 目录是 go run，go install 等命令的当前工作路径（即在此路径下执行上述命令）。src 也是用户代码存放的主要目录，所有的源码都放在这个目录下面，一般一个项目和一个目录对应。
- pkg 存放编译时生成的中间文件（比如：.a）。
- bin 存放编译后生成的可执行文件。

接下来就可以试试代码编译运行了。

在本书中，所有示例代码都放在$GOPATH 目录下的 src\go42 目录中，本书的第一个例子文件名为 test.go，代码如下：

```
//GOPATH\src\go42\chapter-1\1.2\1\ test.go

package main

import "fmt"

func main() {
    fmt.Println("Hello, World!")
}
```

使用 go run 命令执行以上代码，程序输出如下：

```
GOPATH\src\go42\chapter-1\1.2\1>go run test.go

Hello, World!
```

1.3 Go 语言开发工具

本书推荐的 Go 语言开发工具是 LiteIDE，这是一个开源、跨平台的轻量级 Go 语言集成开发环境（IDE）。在安装 LiteIDE 之前，一定要先安装 Go 语言环境。LiteIDE 支持以下的操作系统：

- Windows x86 (32-bit 或 64-bit)
- Linux x86 (32-bit 或 64-bit)

LiteIDE 可以通过以下途径下载。

可执行文件地址：https://sourceforge.net/projects/liteide/files/

源码地址：https://github.com/visualfc/liteide

golang 中国也可以下载（https://www.golangtc.com/download/liteide）。下载速度可能会快一些，但版本更新较慢，建议还是选择官方地址下载。

1. 可执行文件系统下直接安装

Windows 下选择 liteidex35.1.windows-qt5.9.5.zip，下载之后解压，在 liteide\bin 文件夹下找

到 liteide.exe，双击运行。

如无意外，将会出现 LiteIDE 的运行界面，如图 1-2 所示。

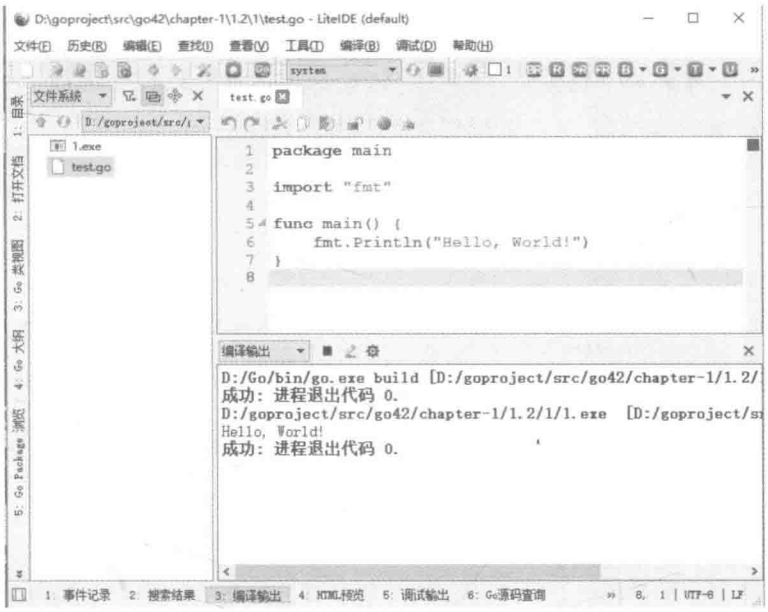

图 1-2　LiteIDE 运行界面

LiteIDE 的使用相对来说比较简单，很容易上手，就不在此细说了。

2．源码编译安装

下载 LiteIDE 源码后，需要使用 Qt4/Qt5 来编译源代码，Qt 库可以从 https://qt-project.org/downloads 上获取。Mac OS X 用户可以不从源代码编译 Qt，直接在终端中运行 brew update && brew install qt，节省大量时间。

有关 LiteIDE 安装的更多说明请访问 http://liteide.org/cn/doc/install/。

其他的开发工具还有 Eclipse、Sublime 等，可以根据个人喜好选择使用。

现在 Go 语言和开发工具都已经安装完成，接下来开始学习 Go 的基础知识，并实际使用它们来进行练习和开发。

第 2 章　Go 语言编程基础

程序语言的数据类型是学习时必须了解的基础知识，Go 语言也一样。本章就从 Go 语言的数据类型开始。

2.1　数据类型

2.1.1　基础数据类型

Go 语言数据类型可用于参数和变量声明，按类别大致有以下几种数据类型：

（1）布尔类型：布尔型的值只可以是常量 true 或者 false，例如 var b bool = true。

（2）数字类型：整型 int 和浮点型 float32、float64。Go 语言支持整型和浮点型数字，并且原生支持复数，其中位的运算采用补码。

（3）字符串类型：字符串就是一串固定长度的字符连接起来的字符序列。Go 语言的字符串是由单个字节连接起来的，字节使用 UTF-8 编码标识 Unicode 文本。

（4）复合（派生）类型：包括指针类型（pointer）、数组类型（array）、结构类型（struct）、通道类型（channel）、函数类型（function）、切片类型（slice）、接口类型（interface）和字典类型（map）。

（5）错误类型（error）：error 类型是 Go 语言预定义类型。

如：

```
type error interface {
    Error() string
}
```

错误类型是接口，其 nil 值表示无错误。例如，定义从文件读取数据的函数，其返回值就有一个错误类型：

```
func Read(f *File, b []byte) (n int, err error)
```

Go 语言也有基于架构的数据类型，如 int、uint 和 uintptr 等，这些数据类型的长度都是根据运行程序时所在的操作系统类型决定的。

根据每种具体的基础数据类型，整理了相关的详细说明，详情如下。

整型见表 2-1。

表 2-1　整型数据

关　键　字	是否有符号	长度（范围）
uint8	无符号	8 位整型（0～255）
uint16	无符号	16 位整型（0～65535）
uint32	无符号	32 位整型（0～4294967295）
uint64	无符号	64 位整型（0～18446744073709551615）

（续）

关 键 字	是否有符号	长度（范围）
int8	有符号	8 位整型（-128～127）
int16	有符号	16 位整型（-32768～32767）
int32	有符号	32 位整型（-2147483648～2147483647）
int64	有符号	64 位整型（-9223372036854775808～9223372036854775807）

其他整型见表 2-2。

表 2-2　其他整型

关 键 字	说　明
byte	类似 uint8，8 位
rune	类似 int32，32 位
uint	32 或 64 位无符号整型，与系统有关
int	32 或 64 位有符号整型，与系统有关
uintptr	无符号整型，用于存放一个指针

浮点型见表 2-3。

表 2-3　浮点型

关 键 字	说　明
float32	IEEE-754　32 位浮点型数
float64	IEEE-754　64 位浮点型数

浮点数可细分为 float32 和 float64 两种。浮点数能够表示的范围可以从很小到很大，这个极限值范围可以在 math 包中获取，math.MaxFloat32 表示 float32 的最大值，大约是 3.4e38，math.MaxFloat64 大约是 1.8e308，两个类型最小的非负值大约是 1.4e-45 和 4.9e-324。

float32 大约可以提供小数点后 6 位的精度，而 float64 可以提供小数点后 15 位的精度。通常情况应该优先选择 float64，因为 float32 的精度较低，在累积计算时误差扩散很快，而且因为浮点数和整数的底层解释方式完全不同，float32 能精确表达的最小正整数并不大。

字符串在 Go 语言中是只读的 Unicode 字节序列，Go 语言使用 UTF-8 格式编码 Unicode 字符，每个字符对应一个 rune 类型。一旦字符串变量赋值之后，内部的字符就不能修改。

在一定条件下，字符串可以转为数字：

```
int, err := strconv.Atoi(string)                    // string 转 int
int64, err := strconv.ParseInt(string, 10, 64)      // string 转 int64
```

数字也可转为字符串：

```
string := strconv.Itoa(int)                         // int 转 string
string := strconv.FormatInt(int64, 10)              // int64 转 string
```

在 Go 语言中，对字符串使用 range 循环会在每次迭代时，解码一个 UTF-8 编码的字符。每次循环时，循环的索引是当前文字的起始位置，它的值（rune）是 Unicode 代码点。

使用 range 迭代字符串时，需要注意，range 迭代的是 Unicode 而不是字节。返回的两个值，第一个是被迭代的字符的 UTF-8 编码的第一个字节在字符串中的索引，第二个是对应的字

符且类型为 rune（实际就是表示 Unicode 值的整型数据）。例如：

```
//GOPATH\src\go42\chapter-2\2.1\1\main.go

package main

import "fmt"

const s = "Go 语言"

func main() {

    for i, r := range s {
        fmt.Printf("%#U  :  %d\n", r, i)
    }
}
```

程序输出：

```
U+0047 'G'  :  0
U+006F 'o'  :  1
U+8BED '语'  :  2
U+8A00 '言'  :  5
```

复数类型见表 2-4。

表 2-4　复数类型

关 键 字	说　明
complex64	用 32 位浮点数构造复数
complex128	用 64 位浮点数构造复数

复数类型在实际中相对使用较少，主要用于数学专业。它分为两种类型：complex64 和 complex128。

复数使用 re+imi 来表示，其中 re 代表实数部分，im 代表虚数部分，i 代表 $\sqrt{-1}$。示例如下：

```
var c complex128 = 1.0 + 10i
fmt.Printf("The value is: %v", c)     // 输出：The value is: (1+10i)
```

如果 re 和 im 的类型均为 float32，那么下面的 cc 是类型为 complex64 的复数：

```
cc:= complex(re, im)
```

函数 real(c)和 imag(c)可以分别获得相应的实数和虚数部分。例如：

```
c := complex(5.67, 99.09)
fmt.Println("re:  ", real(c), " im:  ", imag(c))
```

2.1.2　复合数据类型

Go 语言中有多种复合数据类型：数组（array）、切片（slice）、字典（map）、结构体（struct）、指针（pointer）、函数（function）、接口（interface）和通道（channel）。

数组和结构体都是聚合类型，长度固定。而切片和字典都是动态数据结构，长度可变。有关复合数据类型，后面会详细解释。