



Go语言编程实战

◎ 强彦 王军红 主 编

清华大学出版社

Go 语言编程实战

◎强彦 王军红 主编



清华大学出版社

北京

内 容 简 介

本书从初学者的角度出发,通过通俗易懂的语言、丰富实用的案例,详细介绍了使用 Go 语言进行程序开发需要掌握的知识。全书分为 16 章,包括为什么要使用 Go 语言,Go 语言开发环境,“Hello World”程序实现,流程控制,数组、切片和映射,string 操作,函数,指针,结构体和方法,接口,并发,文件操作,错误处理与日志,创建自己的 go 包,Go 语言编码、数据库编程等。书中所有知识都结合具体实例进行介绍,设计程序代码给出了详细注释,可以使读者轻松领会 Go 程序开发的精髓,快速提高开发技能。另外,本书还有配套的 PPT 和视频讲解。

本书适合作为 Go 语言开发入门者的自学用书,也适合作为高等院校相关专业的教学参考书,还可供开发人员查阅、参考。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

Go 语言编程实战/强彦,王军红主编. —北京:清华大学出版社,2019

ISBN 978-7-302-52301-7

I. ①G… II. ①强… ②王… 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 029205 号

责任编辑:王 芳 李 晔

封面设计:台禹微

责任校对:梁 毅

责任印制:沈 露

出版发行:清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址:北京清华大学学研大厦 A 座 邮 编:100084

社总机:010-62770175 邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质量反馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

课件下载: <http://www.tup.com.cn>,010-62795954

印 装 者:三河市国英印务有限公司

经 销:全国新华书店

开 本:185mm×260mm 印 张:12.25

字 数:299 千字

版 次:2019 年 8 月第 1 版

印 次:2019 年 8 月第 1 次印刷

定 价:59.00 元

产品编号:082137-01



本书作为 Go 编程语言的入门级书籍,分为基础篇、核心篇、提高篇、应用篇四篇。

第 1 篇为基础篇。本篇介绍 Go 语言的产生背景、特点以及 Go 开发环境的安装,旨在一步一步地引领读者走进 Go 的世界。其次,从“Hello World”程序开始,引出常量、变量、运算符等 Go 语言中最基础的知识。

第 2 篇为核心篇。本篇分为八章,讲解流程控制、数组、切片、字典、字符串操作、函数、指针、接口、结构体、方法和并发等 Go 语言中的重要概念,并配有大量的具体案例以供读者参考学习。

第 3 篇为提高篇。本篇主要介绍编写程序时经常遇到的三种需求,分别是文件 I/O 操作、错误处理、自定义 package。其中前两种需求均可以调用 Go 语言标准库中的方法解决,良好的错误处理可以让一个程序的稳定性得到很大提高;自定义 package 可以提高程序代码的复用性,使读者更高效地编写程序。

第 4 篇为应用篇。本篇主要介绍了 Go 语言支持的五种常用编码方式以及 Go 语言操纵 MySQL 数据库的方法,共两章:第 15 章 Go 语言编码,介绍了 Base64 编码、十六进制编码、JSON 编码、XML 编码、CSV 编码,并结合实例介绍了 Go 语言如何对数据进行编码、解码;第 16 章数据库编程,介绍如何使用 Go 语言操作数据库,主要包括 MySQL 数据库安装和 Go 语言连接、查询 MySQL 数据库的方法。

编者

2018 年 11 月



第 1 篇 基 础 篇

第 1 章 为什么要使用 Go 语言	3
1.1 Go 语言的产生背景	3
1.2 Go 语言的主要特点以及使用 Go 语言开发的优势	4
1.2.1 Go 语言的优势	4
1.2.2 Go 语言的设计原则	4
1.2.3 Go 语言的特点	5
本章小结	8
课后练习	8
第 2 章 Go 语言开发环境	9
2.1 Go 安装	9
2.2 开发工具	11
2.2.1 LiteIDE	11
2.2.2 Goland	11
2.2.3 Eclipse	12
2.2.4 Sublime Text	13
2.3 Go 语言中的开发包	14
2.3.1 Go 语言标准库	14
2.3.2 常用包介绍	15
2.3.3 其他包	15
本章小结	16
课后练习	16

第3章 “Hello World”程序实现	17
3.1 Go语言开发的基本规则	17
3.1.1 第一个Go程序	17
3.1.2 包及其导入	18
3.1.3 变量	19
3.1.4 常量	21
3.1.5 注释	23
3.2 基本数据类型	23
3.2.1 布尔类型	23
3.2.2 整型类型	24
3.2.3 浮点类型	25
3.2.4 复数类型	26
3.2.5 字符串类型	26
3.3 派生数据类型	26
3.4 运算符	28
3.4.1 算术运算符	28
3.4.2 关系运算符	28
3.4.3 逻辑运算符	29
3.4.4 按位运算符	29
3.4.5 赋值运算符	30
3.4.6 其他(杂项)运算符	31
3.5 类型转换与类型别名	31
本章小结	32
课后练习	32

第2篇 核心篇

第4章 流程控制	35
4.1 选择结构	35
4.1.1 条件语句	35
4.1.2 switch语句	37
4.2 循环结构	39
4.2.1 for语句	39
4.2.2 range	40
4.3 跳转语句	41

4.3.1	break 和 continue	41
4.3.2	goto	42
	本章小结	43
	课后练习	43
第 5 章	数组、切片和映射	45
5.1	数组	45
5.1.1	声明与初始化	46
5.1.2	元素访问	47
5.1.3	值类型	48
5.2	切片	48
5.2.1	创建与初始化	49
5.2.2	使用切片	50
5.3	映射	57
5.3.1	创建和初始化	58
5.3.2	使用映射	58
	本章小结	61
	课后练习	61
第 6 章	string 操作	63
6.1	string 介绍	63
6.2	连接字符串	64
6.2.1	字符串的连接方式	64
6.2.2	连接方式性能比较	67
6.3	解析字符串	67
6.3.1	遍历字符串	67
6.3.2	字符串操作	69
6.4	检查字符串长度	71
6.4.1	调用 bytes.Count() 函数	72
6.4.2	调用 strings.Count() 函数	72
6.4.3	调用 utf8.RuneCountInString() 函数	73
6.5	数据复制	73
	本章小结	74
	课后练习	74
第 7 章	函数	76
7.1	创建一个简单函数	76

7.2 复杂函数	77
7.2.1 带参数的函数	77
7.2.2 含返回值的函数	78
7.2.3 含多个返回值的函数	79
7.2.4 含多个参数的函数	79
7.3 匿名函数和闭包	81
7.4 递归函数	82
本章小结	83
课后练习	83
第 8 章 指针	86
8.1 指针的定义	86
8.2 Go 语言中的指针	87
8.2.1 Go 语言指针基本操作	87
8.2.2 Go 语言 new 函数	88
8.2.3 Go 语言指针数组	89
8.2.4 Go 语言指针作为函数参数	90
本章小结	90
课后练习	91
第 9 章 结构体和方法	93
9.1 结构体	93
9.1.1 什么是结构体	93
9.1.2 创建一个结构体	94
9.1.3 嵌入式结构体	100
9.2 方法	102
9.2.1 什么是方法	102
9.2.2 如何创建一个方法	103
9.2.3 方法与封装	105
9.2.4 嵌入式结构体中的方法	106
本章小结	107
课后练习	107
第 10 章 接口	109
10.1 接口的定义	109
10.2 接口的实现	110

10.3	空接口	112
10.4	类型断言	113
10.5	类型查询	115
	本章小结	115
	课后练习	115
第 11 章	并发	117
11.1	协程	117
11.1.1	协程简单应用	118
11.1.2	协程与阻塞	118
11.1.3	NewTimer 与 NewTicker	119
11.2	同步协程	121
11.2.1	WaitGroup	121
11.2.2	Cond	122
11.2.3	Once	123
11.3	通道	124
11.3.1	通道定义	124
11.3.2	通道的缓冲机制	125
11.3.3	通道的 close	127
11.3.4	select	127
11.3.5	协程与通道结合	131
	本章小结	133
	课后练习	133

第 3 篇 提 高 篇

第 12 章	文件操作	139
12.1	写数据到文件	139
12.2	从文件中读取数据	141
12.2.1	创建文件	141
12.2.2	打开文件	141
12.2.3	读文件	142
12.3	文件的复制	144
12.3.1	使用 Go 语言内建的 Copy() 函数	144
12.3.2	使用 Go 语言内建的 CopyN() 函数	145
12.3.3	文件的读入与写出	146

本章小结	147
课后练习	147
第 13 章 错误处理与日志	149
13.1 错误处理	149
13.1.1 定义错误	149
13.1.2 打印错误	150
13.2 defer()、panic()、recover()函数	151
13.2.1 defer()函数	151
13.2.2 panic()函数	153
13.2.3 recover()函数	153
13.3 日志	154
13.4 举例结合使用错误处理方法和日志	157
本章小结	158
课后练习	158
第 14 章 创建自己的 go 包	162
14.1 创建一个 go 包	162
14.2 go 包的导入方式	164
14.2.1 相对路径导入	164
14.2.2 绝对路径导入	164
本章小结	164
课后练习	164

第 4 篇 应 用 篇

第 15 章 Go 语言编码	167
15.1 Base64 编码	167
15.2 十六进制编码	169
15.3 JSON 编码	170
15.4 XML 编解码	171
15.4.1 XML 编码	171
15.4.2 XML 解码	172
15.5 CSV 编码	174
本章小结	175
课后练习	175

第 16 章 数据库编程	177
16.1 Go 语言与数据库	177
16.2 安装 MySQL	177
16.3 MySQL 连接	179
16.4 连接测试	180
16.5 数据查询	180
16.5.1 已知数据表结构时查询数据	181
16.5.2 未知数据表结构时查询数据	182
本章小结	183
课后练习	183

第 1 篇 基 础 篇

Go 语言是 Google 公司在 2009 年发布的第二款开源编程语言。Go 语言专门针对多处理器系统应用程序的编程进行了优化,使用 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。

本篇不会过于深入地涉及 Go 语言的核心内容,主要是对本书全篇的各种概念做简要介绍,更加深入的内容会在核心篇详述。本篇旨在让读者在 Go 语言运行环境的安装和入门程序运行的过程中逐步发现 Go 语言的新特性以及它和其他编程语言的不同之处。

本篇主要介绍 Go 的背景、安装、常用 IDE、package 及各种数据类型。

第 1 章主要介绍 Go 语言的产生背景及 Go 语言的特点、优势。

第 2 章主要介绍 Go 语言开发过程中常用的集成开发环境 (IDE) 及 Go 语言的安装方法,还会介绍 Go 语言的抽象语法树、标准库与第三方库。

第 3 章主要通过“Hello World”的简单案例引出 Go 语言的各种数据类型及包的概念,为核心篇做好铺垫。

为什么要使用Go语言

Go语言作为一门新语言,在 Hacker News 于 2018 年 7 月发布的编程语言招聘趋势 TOP10 中,Python 稳居冠军宝座,而 Go 语言逆袭进入前三,Go 语言在编程语言中之所以能有这样的地位,主要在于它区别于其他语言的一些优势。比如部署简单、并发性好、有良好的程序设计风格、执行性能好等。

Go 语言的学习非常简单,不但可以通过同步方式轻松实现高并发,并且代码简洁、格式统一、阅读方便、性能强劲,开发效率又不差于 Python 等动态语言。

总而言之,从工程的角度上来看,对于大多数场景来说,选择 Go 语言是极为明智的选择。这样可以轻松地兼顾运行性能、开发效率及维护难度这三大难点。

本章要点:

- 了解 Go 语言的产生背景。
- 了解 Go 语言的执行性能和开发效率。
- 熟悉 Go 语言的设计规则。
- 熟知 Go 语言的九个主要特点。

1.1 Go 语言的产生背景

最近十年来,C/C++在计算机领域没有得到很好的发展,也没有新的系统编程语言出现,开发程度和系统效率在很多情况下不能兼得。要么执行效率高,开发和编译效率低,如 C++; 要么执行效率低,但编译效率高,如 NET、Java。所以需要一种拥有较高效的执行速度、编译速度和开发速度的编程语言,Go 由此产生。

Go 是由 Google 公司推出的一个开源项目(系统开发语言),是一个编译型、静态类型的语言,具备垃圾收集(garbage collection)、限定性结构类型(structural typing)、内存安全

(memory safety)以及 CSP 样式的并发编程(concurrent programming)等功能特性。

Go 最初的设计由 Robert Griesemer、Rob Pike 和 Ken Thompson 在 2007 年 9 月开始,官方的发布是在 2009 年 11 月。2010 年 5 月由 Rob Pike 公开将其运用于 Google 内部的一个后台系统。目前 google App Engine 也支持 Go 语言(目前仅支持三种:Java、Python 和 Go)。

Go 可以运行在 Linux、Mac OS X、FreeBSD、OpenBSD、Plan 9 和 Microsoft Windows 系统上,同时也支持多种处理器架构,如 I386、AMD64 和 ARM。

1.2 Go 语言的主要特点以及使用 Go 语言开发的优势

1.2.1 Go 语言的优势

选择使用 Go 语言,主要是基于以下两方面的考虑。

1. 执行性能

缩短 API 的响应时长,解决批量请求访问超时的问题。在 Uwork 的业务场景下,一次 API 批量请求往往会涉及对另外接口服务的多次调用,而在之前的 PHP 实现模式下,要做到并行调用是非常困难的,串行处理却不能从根本上提高处理性能。而 Go 语言不一样,通过协程可以方便地实现 API 的并行处理,达到处理效率的最大化。依赖 Go 语言的高性能 HTTP Server,提升系统吞吐能力,由 PHP 的数百级别提升到数千级别甚至过万级别。

2. 开发效率

Go 语言使用起来简单、代码执行效率高、编码规范统一、上手快。通过少量的代码,即可实现框架的标准化,并以统一的规范快速构建 API 业务逻辑。能快速构建各种通用组件和公共类库,进一步提升开发效率。

1.2.2 Go 语言的设计原则

1. Go 程序设计规则

- (1) Go 编程的风格,可以以组为单位进行变量和常量声明,以及加载包;
- (2) Go 语言支持简单的函数、条件和循环风格,把括号都省略了;
- (3) 大写字母开头的变量是可导出的,也就是其他包可以读取的,是公有变量;小写字母开头的变量就是不可导出的,是私有变量;
- (4) 大写字母开头的函数相当于 class 中的带 public 关键词的公有函数;小写字母开头的函数就是带 private 关键词的私有函数;
- (5) Go 语言和 Python 一样不需要以分号结尾;
- (6) Go 语言支持函数返回多个值。

2. Go 语言常用量

表 1-1 列出了 Go 语言的常用量。

表 1-1 Go 语言常用量

常用量	说明
var	用来创建变量
const	用来创建常量
iota	用来声明枚举类型 enum,它默认开始值是 0,每调用一次加 1
map	读取和设置类似 slice,通过 key 来操作,只是 slice 的 index 只能是 int 类型,而 map 多了很多类型,可以是 int,也可以是 string
make	用于内建类型(map,slice 和 channel)的内存分配
new	用于各种类型的内存分配
goto	用来跳转到必须在当前函数内定义的标签
func	用来声明一个函数 funcName
defer	用于延迟执行代码,类似于析构函数
panic	用于中断原有的控制流程
recover	用于恢复中断的函数
import	用于导入包文件

注意: Go 中有两个保留的函数,即 `init()` 函数(能够应用于所有的 package)和 `main()` 函数(只能应用于 package main)。这两个函数在定义时不能有任何参数和返回值。

Go 程序会自动调用 `init()` 和 `main()` 函数,所以不需要在任何地方调用这两个函数。每个 package 中的 `init()` 函数都是可选的,但 package main 必须包含一个 `main()` 函数。

1.2.3 Go 语言的特点

Go 语言以最直接、简单、高效、稳定的方式来解决问題,其关键特性主要包括以下几方面。

1. 并发与协程(goroutine)

Go 在语言级别支持协程(也称为微线程,比线程更轻量、开销更小、性能更高)并发,并且实现起来非常简单。对比 Java 的多线程和 Go 的协程实现,Go 是以简单、高效的方式解决问题。

Java 多线程并发实例如下所示:

```
public class MyThread implements Runnable {
    String arg;
    public MyThread (String a){
        arg = a;
    }
    public void run(){
        //...
    }
    public static void main(String[] args){
        new Thread(new MyThread("test")).start();
        //...
    }
}
```


Go 协程并发实例如下所示：

```
func run(arg string){
    //...
}
func main(){
    go run("test")
    ...
}
```

Go 语言与其他语言的不同之处是，它的语言级别支持协程(goroutine)并发，操作起来非常简单，语言级别提供关键字(Go)用于启动协程，并且在同一台机器上可以启动成千上万个协程。

2. 基于消息传递的通信方式

通道(channel)是 Go 在语言级别提供给进程内的协程的通信方式，简单易用，线程安全。

在异步的并发编程过程中，只能方便、快速地启动协程还不够。协程之间的消息通信也非常重要，否则各个协程就无法控制。在 Go 语言中，使用基于消息传递的通信方式进行协程间通信，并且将消息通道作为基本的数据类型，使用类型关键字 chan 进行定义，并发操作时线程安全。可见，Go 语言会用最实用、最有利于解决问题的方法，以最简单、直接的形式向用户提供服务。

通道并不仅仅只是用于简单的消息通信，还可以引申出很多非常实用，而实现起来又非常方便的功能。比如，实现 TCP 连接池、限流等等，这些在其他语言中实现起来并不容易，但 Go 语言可以轻易做到。

3. 丰富实用的内建数据类型

Go 语言作为编译型语言，在数据类型上也非常全面，除了传统的整型、浮点型、字符型、数组、结构体等类型，从实用性上考虑，也对字符串类型、切片类型(可变长数组)、字典类型、复数类型、错误类型、通道类型，甚至任意类型(interface{})进行了原生支持，并且用起来非常方便。比如字符串、切片类型，操作简便性和 Python 类似。表 1-2 列出了 Go 语言的常用内置数据类型。

表 1-2 常用内建数据类型

常用内建数据类型	说 明
string	字符串类型
slice	切片类型，即可变长序列
map	字典类型，Key-Value 形式
complex64, complex128	复数类型，支持复数运算
error	错误类型，通常用于函数返回，显示表明逻辑执行的正常性
interface{}	Any 类型，类似于 Java 中的 Object 基类，非常灵活
chan	Channel 类型，用于协程间的消息通信

另外，Go 语言将错误类型(error)作为基本的数据类型，并且在语言级别不再支持 try...catch 的用法。Go 的开发者认为在编程过程中，要保证程序的健壮性和稳定性，对异常的精