

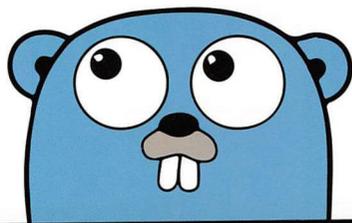
资深程序员、慕课网特邀讲师分享多年的Go语言开发经验
详解Go语法及开发技巧，深度剖析开源网络库cellnet的设计和架构思想
100分钟配套教学视频、72个开发实例精讲

GO语言

从入门到进阶实战

(视频教学版)

徐波◎编著



- 用浅显易懂的语言讲解，不让读者有云山雾罩的感觉
- 用大量实例带领读者学习，让读者在实际动手中提高编程水平
- 给出了大量的“避坑”技巧，让读者在实际开发中少走弯路
- 实例来自于作者多年的口述教学和技术分享会，广受业界好评
- 每个实例程序都精心设计，让Go语言学习者大呼过瘾



机械工业出版社
China Machine Press



内容简介

本书采用“理论+实例”的编写形式，通过大量的实例，并结合作者多年的一线开发实战经验，全面介绍了Go语言的语法及使用方法。全书秉承方便学习、易于理解、便于查询的理念编写而成，无论是想系统学习Go语言基础知识的初学者，还是想进阶提高的有经验开发人员，都能通过本书迅速掌握Go语言的各种基础语法和开发技巧。本书作者曾经与慕课网合作录制过相关视频课程，有丰富的视频制作经验，所以特意将本书重点内容精心录制了配套教学视频，这将极大地提升读者的学习效率，取得比同类图书更好的学习效果。另外，本书还免费提供了书中涉及的实例源代码，以方便读者学习。

本书分为13章，主要介绍了Go语言的特性与环境搭建、基本语法与使用、容器（存储和组织数据的方式）、流程控制、函数、结构体、接口、包、并发、反射、编译与工具、Go程序开发技巧；最后的实战演练部分剖析了作者的开源网络库cellnet的架构及设计思想，并且实现了Socket聊天功能。本书对于Go语言的并发特色功能有全面、深入的讲解，需要读者重点学习。

本书结构清晰，内容通俗易懂，案例丰富，实用性强，特别适合Go语言初学者和进阶读者阅读。另外，本书也适合作为相关培训学校和各大院校的教材或教学参考书。

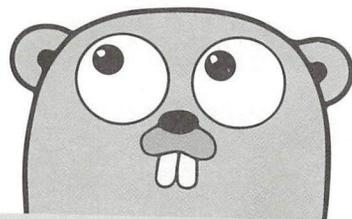


GO语言

从入门到进阶实战

(视频教学版)

徐波◎编著



Let's
Go!



机械工业出版社
China Machine Press



图书在版编目 (CIP) 数据

Go语言从入门到进阶实战: 视频教学版/徐波编著. —北京: 机械工业出版社, 2018.5

ISBN 978-7-111-59824-4

I. G… II. 徐… III. 程序语言—程序设计IV. TP312

中国版本图书馆CIP数据核字 (2018) 第086983号

Go 语言从入门到进阶实战 (视频教学版)

出版发行: 机械工业出版社 (北京市西城区百万庄大街 22 号 邮政编码: 100037)

责任编辑: 欧振旭 李华君

责任校对: 姚志娟

印刷: 中国电影出版社印刷厂

版次: 2018 年 6 月第 1 版第 1 次印刷

开本: 186mm × 240mm 1/16

印张: 26.25

书号: ISBN 978-7-111-59824-4

定价: 99.00 元

凡购本书, 如有缺页、倒页、脱页, 由本社发行部调换

客服热线: (010) 88379426 88361066

投稿热线: (010) 88379604

购书热线: (010) 68326294 88379649 68995259

读者信箱: hzit@hzbook.com

版权所有·侵权必究

封底无防伪标均为盗版

本书法律顾问: 北京大成律师事务所 韩光/邹晓东



配套学习资源

本书提供了配套教学视频等超值学习资料，下面将分别介绍。

1. 同步配套教学视频

作者为本书中的一些重点例子录制了同步配套教学视频，以帮助读者高效学习，如图 1 所示。



图 1 本书教学视频

2. 书中案例源文件

本书提供了书中涉及的案例源代码文件，如图 2 所示。读者可以边看视频边参考提供的源代码自己动手实现书中的例子，这样能迅速学会每个 Go 语言的语法特性，也能迅速



配套学习资源

提高自己的编程能力。



图2 本书案例源文件

3. 配套学习资源获取方式

本书提供的配套学习资源需要读者自行下载。请登录机械工业出版社华章公司的网站 www.hzbook.com，然后搜索到本书页面，单击页面上的资料下载按钮即可下载。



前言

现今，多核 CPU 已经成为服务器的标配。但是对多核的运算能力挖掘一直由程序员人工设计算法及框架来完成。这个过程需要开发人员具有一定的并发设计及框架设计能力。虽然一些编程语言的框架在不断地提高多核资源使用效率，如 Java 的 Netty 等，但仍然需要开发人员花费大量的时间和精力搞懂这些框架的运行原理后才能熟练掌握。

Go 语言在多核并发上拥有原生的设计优势。Go 语言从 2009 年 11 月开源，2012 年发布 Go 1.0 稳定版本以来，已经拥有活跃的社区和全球众多开发者，并且与苹果公司的 Swift 一样，成为当前非常流行的开发语言之一。很多公司，特别是中国的互联网公司，即将或者已经完成了使用 Go 语言改造旧系统的过程。经过 Go 语言重构的系统能使用更少的硬件资源而有更高的并发和 I/O 吞吐表现。

Go 语言简单易学，学习曲线平缓，不需要像 C/C++ 语言动辄需要两到三年的学习期。Go 语言被称为“互联网时代的 C 语言”。互联网的短、频、快特性在 Go 语言中体现得淋漓尽致。一个熟练的开发者只需要短短的一周时间就可以从学习阶段转到开发阶段，并完成一个高并发的服务器开发。

面对 Go 语言的普及和学习热潮，本书使用浅显易懂的语言，介绍了 GO 语言从基础的语法知识到并发和接口等新特性知识，从而带领读者迅速熟悉这门新时代的编程语言。

本书特色

1. 提供同步配套的教学视频

为了让读者更好地学习本书，作者为书中的重点内容录制了配套教学视频，借助这些视频，读者可以更轻松地学习。

作者曾经为慕课网的专业视频制作提供指导，并在慕课网做过多期 Go 语言、Unity 3D 游戏引擎和 Cocos 游戏引擎等网络教学培训，受到众多开发者的青睐及好评。希望读者能够通过作者录制的视频轻松地学习 Go 语言。

2. 来自一线的开发经验及实战例子

本书中的大多数例子及代码都来自于作者多年的口述教学和技术分享会等实践，受到



了众多开发者的一致好评。同时，作者本人也是一名开源爱好者，编写了业内著名的 cellnet 网络库。本书将为读者介绍 cellnet 的架构和设计思想，以帮助读者剖析 cellnet 内部的运行机制，从而让读者能方便地使用 cellnet 快速实现业务逻辑。

3. 浅显易懂的语言、触类旁通的讲解、循序渐进的知识体系

本书在内容编排上尽量做到通俗易懂；在讲解一些常见编程语言特性时，将 Go 语言和其他多种语言的特性进行对比，让掌握多种编程语言的开发者能迅速理解 Go 语言的特性。无论是初学者，还是久经“沙场”的老程序员，都能通过本书快速学习 Go 语言的精华。

4. 内容全面，实用性强

本书详细介绍了作者精心挑选的多个实用性很强的例子，如 JSON 串行化、有限状态机（FSM）、TCP 粘包处理、Echo 服务器和事件系统等。读者既可以从例子中学习并理解 Go 语言的知识点，还可以将这些例子应用于实际开发中。

本书内容

第1章 初识Go语言

本章主要介绍了以下内容：

- (1) Go 语言的特性；
- (2) 使用 Go 语言的开源项目；
- (3) 安装 Go 语言开发包和搭建其开发环境。

第2章 Go语言基本语法与使用

本章主要介绍了 Go 语言的基本语法，如变量、各种常见数据类型及常量，此外还介绍了 Go 1.9 版本中新添加的特性，即类型别名。

第3章 容器：存储和组织数据的方式

本章介绍了 Go 语言编程算法中常用的容器，如数组、切片、映射，以及列表的创建、设置、获取、查询和遍历等操作。

第4章 流程控制

本章主要介绍了常见的条件判断、循环和分支语句，包括以下内容：

- (1) 条件判断（if）；
- (2) 条件循环（for）；



- (3) 键值循环 (for range) ;
- (4) 分支选择 (switch) ;
- (5) 跳转语句 (goto) ;
- (6) 跳出循环 (break) 和继续循环 (continue) 。

第5章 函数 (function)

本章首先介绍了 Go 语言中较为基础的函数声明格式及命名返回值特性；然后介绍了 Go 语言中较为灵活的特性，即函数变量和匿名函数；还介绍了一个展示操作与数据分离的示例：字符串的链式处理，从而引出函数闭包概念；之后介绍了 Go 语言中最具特色的如下几个功能：

- (1) 延迟执行语句 (defer) ——将语句延迟到函数退出时执行；
- (2) 宕机 (panic) ——终止程序运行；
- (3) 宕机恢复 (recover) ——让程序从宕机中恢复。

第6章 结构体 (struct)

本章介绍了 Go 语言中最重要的概念：结构体。首先讲解了结构体多种灵活的实例化和成员初始化方法；接着使用面向对象和面向过程等思想，逐步介绍了 Go 语言中的方法及新的概念接收器；然后使用游戏中经典的位置移动例子，展现了结构体的实际使用方法；最后，使用大量例子介绍了结构体内嵌和类型内嵌内容，并在 JSON 数据的分离实例中体验 Go 语言的内嵌结构体的强大功能。

本章中的经典例子：使用事件系统实现事件的响应和处理——展现 Go 语言的方法与函数的统一调用过程。

第7章 接口 (interface)

本章介绍了 Go 语言接口的如下几个知识点：

- (1) 声明接口；
- (2) 实现接口的条件；
- (3) 接口的嵌套组合；
- (4) 接口的转换；
- (5) 接口类型断言和类型分支——判断接口的类型；
- (6) 空接口。

本章中涉及的例子有：便于扩展输出方式的日志系统；使用接口进行数据的排序；使用空接口实现可以保存任意值的字典及实现有限状态机 (FSM) 等。

第8章 包 (package)

本章首先介绍了构建工程的基础概念 GOPATH，接着介绍了包 (package) 的创建和



导入过程与方法，以及能控制访问权限的导出包内标识符的方法。

本章给出了一个典型的工厂模式自动注册示例，介绍了多个包的定义和使用方法。

第9章 并发

本章讲解了 Go 语言中并发的两个重要概念：轻量级线程(goroutine)和通道(channel)。首先介绍了 goroutine 的创建方法及一些和并发相关的概念，便于读者理解 goroutine 和线程的区别与联系；然后介绍了通道的声明、创建和使用方法，使用 3 个示例，即模拟远程过程调用 (RPC)、使用通道响应计时器事件和 Telnet 回音服务器，来展示通道的实际使用方法；最后介绍了在并发环境下的同步处理方法，如使用互斥锁和等待组等，以及使用竞态检测提前发现并发问题。

第10章 反射

本章按照反射的类别分为两部分：反射类型对象 (reflect.Type) 和反射值对象 (reflect.Value)。首先介绍了反射类型的获取及遍历方法，同时介绍了反射类型对象获取结构体标签的方法；接着介绍了反射值对象获取及修改值和遍历值等；最后通过使用反射将结构体串行化为 JSON 格式字符串的示例，介绍了反射在实际中的运用。

第11章 编译与工具

本章介绍了 Go 语言中常用的编译及工具指令，例如：

- (1) go build/go install——编译及安装源码；
- (2) go get——远程获取并安装源码；
- (3) go test——单元测试和基准测试框架；
- (4) go pprof——性能分析工具。

第12章 “避坑”与技巧

本章首先介绍了作者多年使用 Go 语言的开发经验和技巧总结，以及一些使用 Go 语言中可能发生的错误及优化建议，例如合理使用并发、在性能与灵活性中做出取舍后再使用反射等；接着介绍了 Go 语言中一个不为人知的特性——map 的多键索引，利用该特性可以方便地对数据进行多个条件的索引；最后介绍了使用 Go 语言的 Socket 处理 TCP 粘包问题。

第13章 实战演练——剖析cellnet网络库设计并实现Socket聊天功能

本章介绍了 cellnet 网络库的基本特性、流程、架构及如下几个关键概念：

- (1) 连接管理；
- (2) 会话收发数据流程；
- (3) 事件队列；

- (4) 消息编码;
- (5) 消息元信息;
- (6) 接收和发送封包。

本章使用 cellnet 网络库实现了带有聊天功能的客户端和服务端。

本书读者对象

- Go 语言初学者;
- Go 语言进阶读者;
- 编程初学者;
- 后端程序初学者;
- 前端转后端的开发人员;
- 熟悉 C/C++、Java 和 C#语言, 想了解和学习 Go 语言的编程爱好者;
- 想用 Go 语言快速学习编写服务器端程序的开发者;
- 相关培训学员;
- 各大院校的学生。

关于作者

本书由徐波编写, 郭聪和张锐参与审核和校对。感谢我的妻子和家人在我写书期间的大力支持。

另外, 在本书编写期间, 得到了吴宏伟先生的耐心指导, 他一丝不苟、细致入微地对书稿进行了审核和校对, 这让本书的条理更加清晰, 语言更加通俗易懂。在此表示深深的感谢!

虽然我们对书中所述内容都尽量核实, 并多次进行了文字校对, 但因时间所限, 加之水平所限, 书中的疏漏和错误在所难免, 敬请广大读者批评指正。联系我们请发 E-mail 到 hzbook2017@163.com。

徐波

目录

配套学习资源

前言

第 1 章 初识 Go 语言	1
1.1 Go 语言特性	1
1.2 使用 Go 语言的项目	9
1.3 怎样安装 Go 语言开发包	10
1.3.1 Windows 版安装	11
1.3.2 Linux 版安装	13
1.4 搭建开发环境	14
1.4.1 集成开发环境——Jetbrains GoLand	14
1.4.2 方便定义功能的编辑器——Visual Studio Code	15
第 2 章 Go 语言基本语法与使用	19
2.1 变量	19
2.1.1 声明变量	19
2.1.2 初始化变量	20
2.1.3 多个变量同时赋值	23
2.1.4 匿名变量——没有名字的变量	24
2.2 数据类型	24
2.2.1 整型	25
2.2.2 浮点型	25
2.2.3 示例：输出正弦函数（Sin）图像	26
2.2.4 布尔型	28
2.2.5 字符串	29
2.2.6 字符	31
2.2.7 切片——能动态分配的空间	32
2.3 转换不同的数据类型	33
2.4 指针	34
2.4.1 认识指针地址和指针类型	35
2.4.2 从指针获取指针指向的值	36
2.4.3 使用指针修改值	37

2.4.4	示例：使用指针变量获取命令行的输入信息	39
2.4.5	创建指针的另一种方法——new()函数	40
2.5	变量生命期——变量能够使用的代码范围	40
2.5.1	什么是栈	41
2.5.2	什么是堆	42
2.5.3	变量逃逸（Escape Analysis）——自动决定变量分配方式，提高运行效率	43
2.6	字符串应用	46
2.6.1	计算字符串长度	46
2.6.2	遍历字符串——获取每一个字符串元素	47
2.6.3	获取字符串的某一段字符	48
2.6.4	修改字符串	49
2.6.5	连接字符串	49
2.6.6	格式化	50
2.6.7	示例：Base64 编码——电子邮件的基础编码格式	51
2.6.8	示例：从 INI 配置文件中查询需要的值	52
2.7	常量——恒定不变的值	57
2.7.1	枚举——一组常量值	58
2.7.2	将枚举值转换为字符串	59
2.8	类型别名（Type Alias）	60
2.8.1	区分类型别名与类型定义	61
2.8.2	非本地类型不能定义方法	62
2.8.3	在结构体成员嵌入时使用别名	63
第 3 章	容器：存储和组织数据的方式	65
3.1	数组——固定大小的连续空间	65
3.1.1	声明数组	66
3.1.2	初始化数组	66
3.1.3	遍历数组——访问每一个数组元素	67
3.2	切片（slice）——动态分配大小的连续空间	67
3.2.1	从数组或切片生成新的切片	68
3.2.2	声明切片	70
3.2.3	使用 make()函数构造切片	71
3.2.4	使用 append()函数为切片添加元素	71
3.2.5	复制切片元素到另一个切片	73
3.2.6	从切片中删除元素	74
3.3	映射（map）——建立事物关联的容器	76
3.3.1	添加关联到 map 并访问关联和数据	76
3.3.2	遍历 map 的“键值对”——访问每一个 map 中的关联关系	77
3.3.3	使用 delete()函数从 map 中删除键值对	79
3.3.4	清空 map 中的所有元素	79
3.3.5	能够在并发环境中使用的 map——sync.Map	79

3.4 列表 (list) ——可以快速增删的非连续空间的容器	81
3.4.1 初始化列表	83
3.4.2 在列表中插入元素	83
3.4.3 从列表中删除元素	84
3.4.4 遍历列表——访问列表的每一个元素	85
第 4 章 流程控制	87
4.1 条件判断 (if)	87
4.2 构建循环 (for)	88
4.2.1 for 中的初始语句——开始循环时执行的语句	89
4.2.2 for 中的条件表达式——控制是否循环的开关	89
4.2.3 for 中的结束语句——每次循环结束时执行的语句	90
4.3 示例：九九乘法表	90
4.4 键值循环 (for range) ——直接获得对象的索引和数据	91
4.4.1 遍历数组、切片——获得索引和元素	92
4.4.2 遍历字符串——获得字符	92
4.4.3 遍历 map——获得 map 的键和值	92
4.4.4 遍历通道 (channel) ——接收通道数据	93
4.4.5 在遍历中选择希望获得的变量	93
4.5 分支选择 (switch) ——拥有多个条件分支的判断	94
4.5.1 基本写法	95
4.5.2 跨越 case 的 fallthrough——兼容 C 语言的 case 设计	96
4.6 跳转到指定代码标签 (goto)	96
4.6.1 使用 goto 退出多层循环	96
4.6.2 使用 goto 集中处理错误	97
4.6.3 统一错误处理	98
4.7 跳出指定循环 (break) ——可以跳出多层循环	99
4.8 继续下一次循环 (continue)	100
第 5 章 函数 (function)	101
5.1 声明函数	101
5.1.1 普通函数的声明形式	101
5.1.2 参数类型的简写	102
5.1.3 函数的返回值	102
5.1.4 调用函数	104
5.1.5 示例：将“秒”解析为时间单位	104
5.1.6 示例：函数中的参数传递效果测试	105
5.2 函数变量——把函数作为值保存到变量中	108
5.3 示例：字符串的链式处理——操作与数据分离的设计技巧	109
5.4 匿名函数——没有函数名字的函数	112
5.4.1 定义一个匿名函数	112

5.4.2	匿名函数用作回调函数	113
5.4.3	使用匿名函数实现操作封装	113
5.5	函数类型实现接口——把函数作为接口来调用	115
5.5.1	结构体实现接口	115
5.5.2	函数体实现接口	116
5.5.3	HTTP 包中的例子	117
5.6	闭包 (Closure) ——引用了外部变量的匿名函数	118
5.6.1	在闭包内部修改引用的变量	119
5.6.2	示例：闭包的记忆效应	119
5.6.3	示例：闭包实现生成器	121
5.7	可变参数——参数数量不固定的函数形式	122
5.7.1	fmt 包中的例子	122
5.7.2	遍历可变参数列表——获取每一个参数的值	123
5.7.3	获得可变参数类型——获得每一个参数的类型	124
5.7.4	在多个可变参数函数中传递参数	125
5.8	延迟执行语句 (defer)	127
5.8.1	多个延迟执行语句的处理顺序	127
5.8.2	使用延迟执行语句在函数退出时释放资源	127
5.9	处理运行时发生的错误	131
5.9.1	net 包中的例子	131
5.9.2	错误接口的定义格式	132
5.9.3	自定义一个错误	132
5.9.4	示例：在解析中使用自定义错误	133
5.10	宕机 (panic) ——程序终止运行	135
5.10.1	手动触发宕机	135
5.10.2	在运行依赖的必备资源缺失时主动触发宕机	136
5.10.3	在宕机时触发延迟执行语句	136
5.11	宕机恢复 (recover) ——防止程序崩溃	137
5.11.1	让程序在崩溃时继续执行	137
5.11.2	panic 和 recover 的关系	139
第 6 章	结构体 (struct)	141
6.1	定义结构体	141
6.2	实例化结构体——为结构体分配内存并初始化	142
6.2.1	基本的实例化形式	142
6.2.2	创建指针类型的结构体	143
6.2.3	取结构体的地址实例化	143
6.3	初始化结构体的成员变量	144
6.3.1	使用“键值对”初始化结构体	145
6.3.2	使用多个值的列表初始化结构体	146
6.3.3	初始化匿名结构体	147

6.4	构造函数——结构体和类型的一系列初始化操作的函数封装	148
6.4.1	多种方式创建和初始化结构体——模拟构造函数重载	149
6.4.2	带有父子关系的结构体的构造和初始化——模拟父级构造调用	149
6.5	方法	150
6.5.1	为结构体添加方法	151
6.5.2	接收器——方法作用的目标	152
6.5.3	示例：二维矢量模拟玩家移动	155
6.5.4	为类型添加方法	160
6.5.5	示例：使用事件系统实现事件的响应和处理	165
6.6	类型内嵌和结构体内嵌	170
6.6.1	声明结构体内嵌	170
6.6.2	结构内嵌特性	172
6.6.3	使用组合思想描述对象特性	173
6.6.4	初始化结构体内嵌	174
6.6.5	初始化内嵌匿名结构体	175
6.6.6	成员名字冲突	177
6.7	示例：使用匿名结构体分离 JSON 数据	178
第 7 章	接口 (interface)	181
7.1	声明接口	181
7.1.1	接口声明的格式	181
7.1.2	开发中常见的接口及写法	182
7.2	实现接口的条件	182
7.2.1	接口被实现的条件一：接口的方法与实现接口的类型方法格式一致	182
7.2.2	条件二：接口中所有方法均被实现	185
7.3	理解类型与接口的关系	186
7.3.1	一个类型可以实现多个接口	186
7.3.2	多个类型可以实现相同的接口	187
7.4	示例：便于扩展输出方式的日志系统	189
7.5	示例：使用接口进行数据的排序	195
7.5.1	使用 <code>sort.Interface</code> 接口进行排序	195
7.5.2	常见类型的便捷排序	197
7.5.3	对结构体数据进行排序	199
7.6	接口的嵌套组合——将多个接口放在一个接口内	202
7.7	在接口和类型间转换	205
7.7.1	类型断言的格式	205
7.7.2	将接口转换为其他接口	205
7.7.3	将接口转换为其他类型	208
7.8	空接口类型 (<code>interface {}</code>)——能保存所有值的类型	208
7.8.1	将值保存到空接口	209
7.8.2	从空接口获取值	209

7.8.3	空接口的值比较	210
7.9	示例：使用空接口实现可以保存任意值的字典	211
7.10	类型分支——批量判断空接口中变量的类型	214
7.10.1	类型断言的书写格式	214
7.10.2	使用类型分支判断基本类型	215
7.10.3	使用类型分支判断接口类型	215
7.11	示例：实现有限状态机（FSM）	217
第 8 章	包（package）	227
8.1	工作目录（GOPATH）	227
8.1.1	使用命令行查看 GOPATH 信息	227
8.1.2	使用 GOPATH 的工程结构	228
8.1.3	设置和使用 GOPATH	229
8.1.4	在多项目工程中使用 GOPATH	230
8.2	创建包 package——编写自己的代码扩展	231
8.3	导出标识符——让外部访问包的类型和值	231
8.3.1	导出包内标识符	231
8.3.2	导出结构体及接口成员	232
8.4	导入包（import）——在代码中使用其他的代码	232
8.4.1	默认导入的写法	233
8.4.2	导入包后自定义引用的包名	234
8.4.3	匿名导入包——只导入包但不使用包内类型和数值	235
8.4.4	包在程序启动前的初始化入口：init	235
8.4.5	理解包导入后的 init()函数初始化顺序	235
8.5	示例：工厂模式自动注册——管理多个包的结构体	237
第 9 章	并发	241
9.1	轻量级线程（goroutine）——根据需要随时创建的“线程”	241
9.1.1	使用普通函数创建 goroutine	241
9.1.2	使用匿名函数创建 goroutine	244
9.1.3	调整并发的运行性能（GOMAXPROCS）	245
9.1.4	理解并发和并行	245
9.1.5	Go 语言的协作程序（goroutine）和普通的协作程序（coroutine）	246
9.2	通道（channel）——在多个 goroutine 间通信的管道	246
9.2.1	通道的特性	247
9.2.2	声明通道类型	247
9.2.3	创建通道	248
9.2.4	使用通道发送数据	248
9.2.5	使用通道接收数据	249
9.2.6	示例：并发打印	252
9.2.7	单向通道——通道中的单行道	254