O'REILLY®

Go语言并发之道

Concurrency in Go

Katherine Cox-Buday 著
于畅 马鑫 赵晨光 译

中国电力出版社

Go语言并发之道

Katherine Cox-Buday 著
于畅 马鑫 赵晨光 译

Beijing • Boston • Farnham • Sebastopol • Tokyo O'REILLY®
O'Reilly Media, Inc. 授权中国电力出版社出版

中国电力出版社

图书在版编目（CIP）数据

Go语言并发之道 /（美）凯瑟琳（Katherine Cox-Buday）著；于畅，马鑫，赵晨光译.
—北京：中国电力出版社，2018.11

书名原文：Concurrency in Go

ISBN 978-7-5198-2494-5

I. ①G… II. ①凯… ②于… ③马… ④赵… 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249617号

北京市版权局著作权合同登记 图字：01-2018-3791号

出版发行：中国电力出版社
地　　址：北京市东城区北京站西街19号（邮政编码100005）
网　　址：http://www.cepp.sgcc.com.cn
责任编辑：刘 炽（liuchi1030@163.com）
责任校对：黄蓓，常燕昆
装帧设计：Karen Montgomery，张 健
责任印制：杨晓东

印　　刷：北京天宇星印刷厂
版　　次：2018年11月第一版
印　　次：2018年11月北京第一次印刷
开　　本：750毫米×980毫米 16开本
印　　张：16.25
字　　数：304千字
印　　数：0001—3000册
定　　价：58.00元

O'Reilly Media, Inc.介绍

O'Reilly Media通过图书、杂志、在线服务、调查研究和会议等方式传播创新知识。自1978年开始，O'Reilly一直都是前沿发展的见证者和推动者。超级极客们正在开创着未来，而我们关注真正重要的技术趋势——通过放大那些“细微的信号”来刺激社会对新科技的应用。作为技术社区中活跃的参与者，O'Reilly的发展充满了对创新的倡导、创造和发扬光大。

O'Reilly为软件开发人员带来革命性的“动物书”；创建第一个商业网站（GNN）；组织了影响深远的开放源代码峰会，以至于开源软件运动以此命名；创立了《Make》杂志，从而成为DIY革命的主要先锋；公司一如既往地通过多种形式缔结信息与人的纽带。O'Reilly的会议和峰会集聚了众多超级极客和高瞻远瞩的商业领袖，共同描绘出开创新产业的革命性思想。作为技术人士获取信息的选择，O'Reilly现在还将先锋专家的知识传递给普通的计算机用户。无论是通过书籍出版，在线服务或者面授课程，每一项O'Reilly的产品都反映了公司不可动摇的理念——信息是激发创新的力量。

业界评论

“O'Reilly Radar博客有口皆碑。”

——Wired

“O'Reilly凭借一系列（真希望当初我也想到了）非凡想法建立了数百万美元的业务。”

——Business 2.0

“O'Reilly Conference是聚集关键思想领袖的绝对典范。”

——CRN

“一本O'Reilly的书就代表一个有用、有前途、需要学习的主题。”

——Irish Times

“Tim是位特立独行的商人，他不光放眼于最长远、最广阔的视野并且切实地按照Yogi Berra的建议去做了：‘如果你在路上遇到岔路口，走小路（岔路）。’回顾过去Tim似乎每一次都选择了小路，而且有几次都是一闪即逝的机会，尽管大路也不错。”

——Linux Journal

献给让这本书得以成文的L.和N.，
你们是我生命中最美好的，我爱你们。

目录

前言

嘿，欢迎阅读本书！很高兴你已经拿起这本书开始阅读，非常期待在接下来的 6 章中和你一起探索关于 Go 语言并发编程的主题。

Go 语言是一种美妙的语言。当它被创造并首次公开的时候，我带着极大的兴趣探索它：简洁、编译速度飞快、运行稳定、支持鸭子类型（duck typing），让我高兴的是，它原生支持并发。当我第一次使用“go 关键字”创建一个 goroutine 的时候，（我保证）我开心得只剩傻笑了。我曾经用其他一些编程语言写过并发程序，但我从未用过像 Go 语言这样这么容易实现并发的语言（我并不是说其他有这种特性的语言不存在，只是我没用过）。我已经找到了我的 Go 语言最佳实践。

在过去的几年里，我用 Go 语言写个人的脚本和项目，直到发现自己已经可以在成千上万行代码的项目中畅游。随着语言的不断发展和社区的不断壮大，我们一起找到了 Go 语言并发编程的最佳实践。一些人就他们找到的模式进行讨论，但在社区里还没有如何使用 Go 语言并发编程的综合指南。

正是考虑到这一点，我才决定写这本书。我希望可以让社区了解到关于 Go 语言并发编程的一些全面且高质量的信息：如何使用，最佳实践，以及如何将它集成到你的系统中，还有它背后的工作原理。我竭尽全力均衡这些关注点。

我希望这是一本对你有益的书。

本书的读者对象

这本书适合已经了解 Go 语言，并有一些开发经验的人。我没有解释 Go 语言的基本语法。最好了解一些其他语言的并发编程，当然这并不是必须的。

通过本书，我们将会讨论整个 Go 语言并发的技术栈：常见的并发陷阱，Go 语言并发设计原理，Go 语言并发原语中的基础语法，常见的并发模式，并发模式的设计，各种工具的使用。

由于主题涵盖的范围较广，本书适合各个使用方向的读者。下面一节将会帮助你快速找到你想要了解的内容。

本书内容

当我阅读技术书籍的时候，通常会先读能引起我兴趣的地方。或者，当我努力研究工作上所需的新技术的时候，我会先看与我工作相关的内容。无论你的出发点是什么，这里有一份关于本书的线路图，它能帮助你找到你需要的内容。

第 1 章 并发概述

本章将提供一个广泛的历史视角来说明并发的重要性，还会讨论一些并发中难以纠正的问题。它还简要介绍了 Go 语言如何帮助你解决这些问题。

如果你有并发的相关知识，或者只是想了解如何使用 Go 语言的并发原语，可以跨过此章节。

第 2 章 对你的代码建模：通信顺序进程

本章论述了推动 Go 语言设计的一些激励因素。这将帮助你在社区中与其他人顺畅交流，并理解为什么要按照 Go 语言的设计模式来编程。

第 3 章 Go 语言并发组件

在这里我们将开始深入 Go 语言并发原语的语法。我们还会介绍控制内存

访问同步的 sync 包。如果之前你从未使用 Go 语言做过并发编程，并且你正在研究如何正确地使用它，那么你应该从这里开始阅读。

通过基础的 Go 语言并发代码片段，与其他语言的并发模型做比较。严格地说，这些知识并不是必需的，但是这些概念有助于你完全理解 Go 语言的并发实现。

第 4 章 Go 语言的并发模式

在本章中，我们将开始研究如何将 Go 语言原生函数构造成合理的模式。这些模式可以解决并避免我们在组合原生函数时可能遇到的问题。

如果你已经开始写 Go 语言的并发代码了，那么这一章还是有点用的。

第 5 章 大规模并发

在这一章中，我们将组合之前学到模式，设计更合理的模型，应用到大型程序、服务和分布式系统中。

第 6 章 goroutine 和 Go 语言运行时

本章描述 Go 语言运行时如何处理调度 goroutine。这些内容主要是给那些想了解 Go 语言运行时内部构造的人学习的。

在线资源

Go 语言拥有一个非常活跃且富有激情的社区！对于那些 Go 语言的初学者，请放心，这个社区很容易找到友好、乐于助人的朋友来帮助你走上 Go 语言编程之路。下面是我最喜欢的一些面向社区的用来阅读、获得帮助，以及和你的 Gopher 伙伴进行互助的资源。

- *https://golang.org/*
- *https://golang.org/play*
- *https://go.googlesource.com/go*

- *https://groups.google.com/group/golang-nuts*
- *https://github.com/golang/go/wiki*

本书约定

本书使用以下排版约定：

斜体

表示新术语、URL、电子邮件地址、文件名和文件扩展名。

等宽字体（`constant width`）

表示程序片段，以及正文中出现的变量、函数名、数据库、数据类型、环境变量、语句和关键字等。

加粗等宽字体（**`constant width bold`**）

表示需要由用户输入的命令或其他文本。

等宽斜体（*`Constant width italic`*）

表示需要由用户输入的值或根据上下文确定的值替换的部分。

表示提示、建议或一般注意事项。

表示警示或告诫。

使用代码示例

本书中所包含的代码都可以在 *http://katherine.cox-buday.com/concurrency-in-go* 页面找到。所有代码都在 MIT 许可下发布的，并且可以在许可的条件下使用。

O'Reilly Safari

Safari（前身为 Safari Books Online）是一个会员制的培训、参考网站，服务于企业、政府、教育者和个人。

会员可以访问数千书籍、培训视频、学习路径、交互教程和超过 250 家出版商的企划列表，包括 O'Reilly Media、Harvard Business Review、Prentice Hall Professional、 Addison-Wesley Professional、Microsoft Press、Sams、Que、Peachpit Press、Adobe、Focal Press、Cisco Press、John Wiley & Sons、Syngress、Morgan Kaufmann、IBM Redbooks、Packt、Adobe Press、FT Press、Apress、Manning、New Riders、McGraw-Hill、Jones & Bartlett、Course Technology 等。

更多信息请访问 *http://oreilly.com/safari*。

联系我们

请把对本书的评价和问题发给出版社。

美国：

O'Reilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472

中国：

北京市西城区西直门南大街2号成铭大厦C座807室（100035）
奥莱利技术咨询（北京）有限公司

我们为本书准备了一个网页，用于陈列勘误、示例代码，以及其他信息。本书的网址是：*http://bit.ly/concurrency-in-go*。

对于本书的评论和技术性问题，请发送电子邮件到：*bookquestions@oreilly.com*。

要了解更多 O'Reilly 图书、培训课程、会议和新闻的信息，请访问以下网站：*http://www.oreilly.com*。

我们的 Facebook：*http://facebook.com/oreilly*。

我们的 Twitter：*http://twitter.com/oreillymedia*。

我们的 YouTube：*http://www.youtube.com/oreillymedia*。

致谢

编写本书是一个充满挑战且令人畏惧的工作。如果没有一群人支持我，审查内容，编写工具和回答问题，本书是不可能完成的。我非常感谢所有提供帮助的人，我们共同完成了本书！

一燕不成夏……

——Proverb

- Alan Donovan，他帮助我完成最初的提议，并帮助我踏上征程。
- Andrew Wilkins，我非常有幸可以在 Canonical 和他一起工作，他的洞察力、专业精神和智慧影响了这本书，他的评论使本书变得更好。
- Ara Pulido，帮助我用一个新 gopher 的视角审视本书。
- Dawn Schanafelt，我的编辑。在使本书读起来尽可能清晰的工作中起到了非常大的作用。我特别感谢她（和 O'Reilly），我写这本书时在生活的道路上遇到一些困难时给予我的耐心。

- Francesc Campoy，经常提醒我要总是注意新的 gopher。
- Ivan Daniluk，他对细节的关注和并发的兴趣确保这是一本全面而有用的书。
- Yasushi Shoji，`org-asciidoc` 的作者，`org-asciidoc` 是我用来将 Org 文档转化为 AsciiDoc 文档的工具，他并不知道他在帮助我写一本书，但是他对 bug 修复以及回答问题十分热情。
- Go 语言维护者：感谢你们的奉献。
- Org 模式的维护者，本书是使用 GNU Emacs 模式编写的，我这辈子都在使用 org，感谢大家。
- GNU Emasc 的维护者，编写本书所使用的文本编辑器，我想象不到一个对于我的生命更有意义的工具了。
- 圣路易斯公共图书馆，进行本书大部分内容写作的地方。

第 1 章

并发概述

并发是一个有趣的词，在我们的认知中，不同的人对它有不同的理解。除了“并发”之外，你可能已经听到了被到处传播的“异步”“并行”或“线程”这几个词。有些人认为这些词表示同一件事，有些人则非常具体地描绘出这些词之间的区别。如果我们要花一整本书的时间讨论并发，那么有必要先花些时间讨论一下我们所说的“并发”究竟是什么。

第 2 章我们将花些时间讲并发的哲学，但现在先给并发一个较为实际的定义，它将作为我们理解的基础。

当大多数人使用“并发”这个词时，通常指的是与一个或多个进程同时发生的过程。通常，这也意味着所有这些进程都在同一时间取得进展。理解这个定义的一个简单方法是想象成人。你目前正在阅读这本书，而世界上的其他人同时也在生活。他们与你同时存在。

并发是计算机科学中的一个宽泛的话题，从这个定义中可以学到各种各样的主题：理论，建模的方法，逻辑的正确性，实际的问题，甚至是理论物理！在本书中，我们将讨论一些辅助主题，但我们将主要基于 Go 语言环境讨论一些并发的实际问题，具体来说：如何选择模型并发，从这个模型中产生哪些问题，以及我们如何在这个模型中组合原语来解决问题。

在本章中，我们将深入了解并发成为计算机科学中的一个重要话题的原因，为什么并发是困难的，并且值得仔细研究，最重要的是，尽管这些想法富有挑战性，Go 语言可以通过其并发原语使程序更加清晰和迅捷。