

## 23 | 测试的基本规则和流程（上）

2018-10-03 郝林



你好，我是郝林，今天我分享的主题是：测试的基本规则和流程（上）。

你很棒，已经学完了本专栏最大的一个模块！这涉及了Go语言的所有内建数据类型，以及非常有特色的那些流程和语句。

你已经完全可以去独立编写各种各样的Go程序了。如果忘了什么，回到之前的文章再复习一下就好了。

在接下来的日子里，我将带你去学习在Go语言编程进阶的道路上必须掌握的附加知识，比如：Go程序测试、程序监测，以及Go语言标准库中各种常用代码包的正确用法。

从上个世纪到今日今时，程序员们，尤其是国内的程序员们，都对编写程序乐此不疲，甚至废寝忘食（比如我自己就是一个例子）。

因为这是我们普通人训练自我、改变生活、甚至改变世界的一种特有的途径。不过，同样是程序，我们却往往对编写用于测试的程序敬而远之。这是为什么呢？

我个人感觉，从人的本性来讲，我们都或多或少会否定“对自我的否定”。我们不愿意看到我们编写的程序有Bug（即程序错误或缺陷），尤其是刚刚倾注心血编写的，并且信心满满交付的程序。

不过，我想说的是，人是否会进步以及进步得有多快，依赖的恰恰就是对自我的否定，这包括否

定的深刻与否，以及否定自我的频率如何。这其实就是“不破不立”这个词表达的含义。

对于程序和软件来讲，尽早发现问题、修正问题其实非常重要。在这个网络互联的大背景下，我们所做的程序、工具或者软件产品往往可以被散布得更快、更远。但是，与此同时，它们的错误和缺陷也会是这样，并且可能在短时间内就会影响到成千上万甚至更多的用户。

你可能会说：“在开源模式下这就是优势啊，我就是要让更多的人帮我发现错误甚至修正错误，我们还可以一起协作、共同维护程序。”但这其实是两码事，协作者往往是由早期或核心的用户转换过来的，但绝对不能说程序的用户就肯定会成为协作者。

当有很多用户开始对程序抱怨的时候，很可能就预示着你对此的人设要崩塌了。你会发现，或者总有一天会发现，越是人们关注和喜爱的程序，它的测试（尤其是自动化的测试）做得就越充分，测试流程就越规范。

即使你想众人拾柴火焰高，那也得先让别人喜欢上你的程序。况且，对于优良的程序和软件来说，测试必然是非常受重视的一个环节。所以，尽快用测试为你的程序建起堡垒吧！

对于程序或软件的测试也分很多种，比如：单元测试、**API**测试、集成测试、灰度测试，等等。我在本模块会主要针对单元测试进行讲解。

我们来说一下单元测试，它又称程序员测试。顾名思义，这就是程序员们本该做的自我检查工作之一。

**Go**语言的缔造者们从一开始就非常重视程序测试，并且为**Go**程序的开发者们提供了丰富的**API**和工具。利用这些**API**和工具，我们可以创建测试源码文件，并为命令源码文件和库源码文件中的程序实体，编写测试用例。

在**Go**语言中，一个测试用例往往会由一个或多个测试函数来代表，不过在大多数情况下，每个测试用例仅用一个测试函数就足够了。测试函数往往用于描述和保障某个程序实体的某方面功能，比如，该功能在正常情况下会因什么样的输入，产生什么样的输出，又比如，该功能会在什么情况下报错或表现异常，等等。

我们可以为**Go**程序编写三类测试，即：功能测试（**test**）、基准测试（**benchmark**，也称性能测试），以及示例测试（**example**）。

对于前两类测试，从名称上你就应该可以猜到它们的用途。而示例测试严格来讲也是一种功能测试，只不过它更关注程序打印出来的内容。

一般情况下，一个测试源码文件只会针对于某个命令源码文件，或库源码文件（以下简称被测源码文件）做测试，所以我们总会（并且应该）把它们放在同一个代码包内。

测试源码文件的主名称应该以被测源码文件的主名称为前导，并且必须以“**\_test**”为后缀。例如，

如果被测源码文件的名称为`demo52.go`，那么针对它的测试源码文件的名称就应该是`demo52_test.go`。

每个测试源码文件都必须至少包含一个测试函数。并且，从语法上讲，每个测试源码文件中，都可以包含用来做任何一类测试的测试函数，即使把这三类测试函数都塞进去也没有问题。我通常就是这么做的，只要把控好测试函数的分组和数量就可以了。

我们可以依据这些测试函数针对的不同程序实体，把它们分成不同的逻辑组，并且，利用注释以及帮助类的变量或函数来做分割。同时，我们还可以依据被测源码文件中程序实体的先后顺序，来安排测试源码文件中测试函数的顺序。

此外，不仅仅对测试源码文件的名称，对于测试函数的名称和签名，Go语言也是有明文规定的。你知道这个规定的内容吗？

所以，我们今天的问题就是：**Go语言对测试函数的名称和签名都有哪些规定？**

这里我给出的典型回答是下面三个内容。

对于功能测试函数来说，其名称必须以`Test`为前缀，并且参数列表中只应有一个`*testing.T`类型的参数声明。

对于性能测试函数来说，其名称必须以`Benchmark`为前缀，并且唯一参数的类型必须是`*testing.B`类型的。

对于示例测试函数来说，其名称必须以`Example`为前缀，但对函数的参数列表没有强制规定。

## 问题解析

我问这个问题的目的一般有两个。第一个目的当然是考察Go程序测试的基本规则。如果你经常编写测试源码文件，那么这道题应该是很容易回答的。

第二个目的是作为一个引子，引出第二个问题，即：`go test`命令执行的主要测试流程是什么？不过在这里我就不问你了，我直接说一下答案。

我们首先需要记住一点，只有测试源码文件的名称对了，测试函数的名称和签名也对了，当我们运行`go test`命令的时候，其中的测试代码才有可能被运行。

`go test`命令在开始运行时，会先做一些准备工作，比如，确定内部需要用到的命令，检查我们指定的代码包或源码文件的有效性，以及判断我们给予的标记是否合法，等等。

在准备工作顺利完成之后，`go test`命令就会针对每个被测代码包，依次地进行构建、执行包中符合要求的测试函数，清理临时文件，打印测试结果。这就是通常情况下的主要测试流程。

请注意上述的“依次”二字。对于每个被测代码包，`go test`命令会串行地执行测试流程中的每个步骤。

但是，为了加快测试速度，它通常会并发地对多个被测代码包进行功能测试，只不过，在最后打印测试结果的时候，它会依照我们给定的顺序逐个进行，这会让我们感觉到它是在完全串行地执行测试流程。

另一方面，由于并发的测试会让性能测试的结果存在偏差，所以性能测试一般都是串行进行的。更具体地说，只有在所有构建步骤都做完之后，`go test`命令才会真正地开始进行性能测试。

并且，下一个代码包性能测试的进行，总会等到上一个代码包性能测试的结果打印完成才会开始，而且性能测试函数的执行也都会是串行的。

一旦清楚了Go程序测试的具体过程，我们的一些疑惑就自然有了答案。比如，那个名叫TestIntroduce的测试函数为什么没执行，又比如，为什么即使是简单的性能测试，执行起来也会比功能测试慢，等等。

## 总结

在本篇文章的一开始，我就试图向你阐释程序测试的重要性。在我经历的公司中起码有一半都不重视程序测试，或者说没有精力去做程序测试。

尤其是中小型的公司，他们往往完全依靠软件质量保障团队，甚至真正的用户去帮他们测试。在这些情况下，软件错误或缺陷的发现、反馈和修复的周期通常会很长，成本也会很大，也许还会造成很不好的影响。

Go语言是一门很重视程序测试的编程语言，它不但自带了testing包，还有专用于程序测试的命令`go test`。我们要想真正用好一个工具，就需要先了解它的核心逻辑。所以，我今天问你的第一个问题就是关于`go test`命令的基本规则和主要流程的。在知道这些之后，也许你对Go程序测试就会进入更深层次的了解。

## 思考题

除了本文中提到的，你还知道或用过testing.T类型和testing.B类型的哪些方法？它们都是做什么用的？你可以给我留言，我们一起讨论。

感谢你的收听，我们下次再见。

[戳此查看Go语言专栏文章配套详细代码。](#)

# GO语言核心36讲

3个月带你通关 GO 语言

郝林

《Go 并发编程实战》作者  
GoHackers 技术社群发起人  
前轻松筹大数据负责人

