Go语言入门到精通

江洲老师云课堂

主讲：江洲老师

生产者消费者模型

目 录

单向channel

1.单向channel

单向channel

默认情况下，通道channel是双向的，也就是，既可以往里面发送数据也可以同里面接收数据。

但是，我们经常见一个通道作为参数进行传递而值希望对方是单向使用的，要么只让它发送数据，要么只让它接收数据，这时候我们可以指定通道的方向。

单向channel变量的声明如下：


```
var ch1 chan int          // ch1是一个正常的channel，是双向的
var ch2 chan<- float64    // ch2是单向channel，只用于写float64数据
var ch3 <-chan int        // ch3是单向channel，只用于读int数据
```

注：

- chan<- 表示数据进入管道，要把数据写进管道，对于调用者就是输出。
- <-chan 表示数据从管道出来，对于调用者就是得到管道的数据，当然就是输入。
- 可以将 channel 隐式转换为单向队列，只收或只发，不能将单向 channel 转换为普通 channel

生产者消费者模型

1.生产者消费者模型

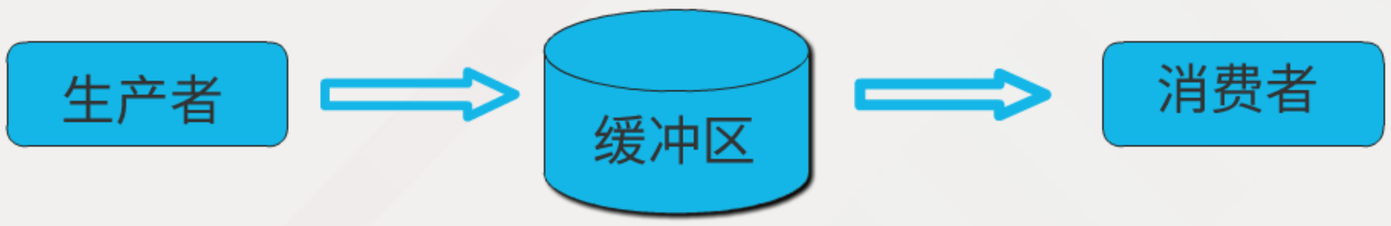
生产者消费者模型

单向channel最典型的应用是“**生产者消费者模型**”。

所谓“生产者消费者模型”：某个模块（函数等）负责产生数据，这些数据由另一个模块来负责处理（此处的模块是广义的，可以是类、函数、协程、线程、进程等）。
产生数据的模块，就形象地称为生产者；而处理数据的模块，就称为消费者。

单单抽象出生产者和消费者，还够不上是生产者／消费者模型，该模式还需要有一个缓冲区处于生产者和消费者之间，作为一个中介。生产者把数据放入缓冲区，而消费者从缓冲区取出数据。

大概的结构如下图：



简单说明：
首先创建一个双向的channel，然后开启一个新的goroutine，把双向通道作为参数传递到producer方法中，同时转成只写通道。

子协程开始执行循环，向只写通道中添加数据，这就是生产者。
主协程，直接调用consumer方法，该方法将双向通道转成只读通道，通过循环每次从通道中读取数据，这就是消费者。

注意：channel作为参数传递，是引用传递。

生产者消费者模型

在生产者消费者模型中，缓冲区有什么作用呢?

1：解耦

假设生产者和消费者分别是两个类，如果让生产者直接调用消费者的某个方法，那么生产者对于消费者就会产生依赖（也就是耦合），将来如果消费者的代码发生变化，可能会直接影响到生产者。

而如果两者都依赖于某个缓冲区，两者之间不直接依赖，耦合度就相应降低了。

2：处理并发

生产者直接调用消费者的某个方法还有另一个弊端：由于函数调用是同步的（或者叫阻塞的），在消费者的方法没有返回之前，生产者只好一直等在那边，而万一消费者处理数据很慢，生产者只能无端浪费时间。

使用了生产者／消费者模式之后，生产者和消费者可以是两个独立的并发主体，生产者把制造出来的数据往缓冲区一丢，就可以再去生产下一个数据，基本上不用依赖消费者的处理速度。

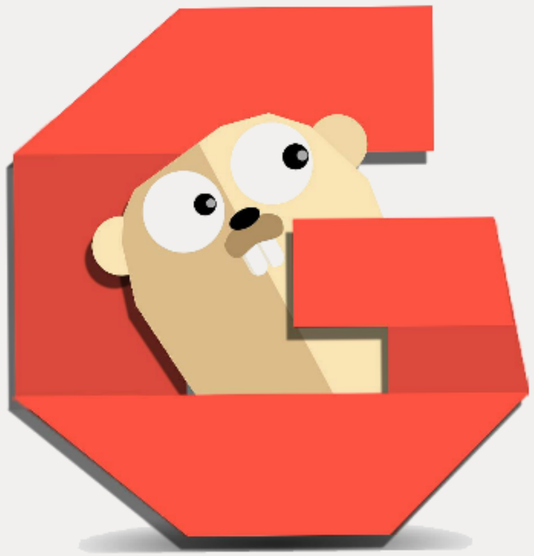
其实最当初这个**生产者消费者模式，主要就是用来处理并发问题的。**

3：缓存

如果生产者制造数据的速度时快时慢，缓冲区的好处就体现出来了。

当数据制造快的时候，消费者来不及处理，未处理的数据可以暂时存在缓冲区中，等生产者的制造速度慢下来，消费者再慢慢处理掉。

实例演示



Go语言入门到精通

江洲老师云课堂

主讲：江洲老师

感谢您的聆听和观看