|  |
| --- |
| <https://www.cnblogs.com/zyf-zhaoyafei/p/5919421.html> |
| <https://blog.csdn.net/kongdefei5000/article/details/75209005> |
| <https://blog.csdn.net/zhonglinzhang/article/details/45913443> |
|  |

管道channel

Golang的协程进行通信时 ， 通过channel 进行

Channel可认为类似OS体系中的消息队列

Golang channel 区分为 有缓冲 与 无缓冲

无缓冲 是同步的 c1 := make(chan int)

有缓冲 是非同步的 c2 := make(chan int, 1)

c1 <- 1

无缓冲 在像c1通道放1 ， 而且一直要有别的协程 <-c1 接手了这个参数，那么c1 <- 1才会继续下去，否则会一直阻塞

c2 <- 1 则不会阻塞 ，因为缓冲大小是 1，只有当放第二个值的时候，第一个还未被取走，此时才会阻塞

即：

无缓冲 一个送信人到你家门口送信，你不在家他不走，只有你接下信，他才走；无缓冲保证信送入到手上

有缓冲 一个送信人到你家送信，将信仍进你家信箱，转身就走；除非你的信箱满了，他必须等信箱空下来，有缓冲能保证信进入你家信箱

Channel 关注点：

1. 可否探测队列是满或空？或者说是否可以不阻塞的尝试读写？

|  |
| --- |
| ch := make (chan **int**, 1)  ch <- 1  **select** {  **case** ch <- 2:  **default**:  fmt.Println("channel is full !")  } |

1. 读阻塞和写阻塞时关闭会怎样？
2. 关闭后未读取的消息会被抛弃？
3. 往关闭的channel发送数据或读取数据会怎么样？
4. 怎样探测channel的关闭？
5. 两个地方读或写阻塞同一个channel , 有没有优先级？
6. 是否可以设置阻塞的超时时间？

|  |
| --- |
| timeout := make (chan **bool**, 1)  go func() {  time.Sleep(1e9) // sleep one second  timeout <- **true**  }()  ch := make (chan **int**)  **select** {  **case** <- ch:  **case** <- timeout:  fmt.Println("timeout!")  } |

1. 阻塞时怎样可以被弹出来？比如某些信号？

答案：

1. 不能不阻塞的尝试读写（可模拟）
2. 关闭会导致推出阻塞
3. 可以探测关闭
4. Channel本身不能设定超时

重点关注：

1. 出基本读写方式外，还有哪些特别的读写方式？在阻塞、关闭、超时方面又有什么不同？发现了select range关键字
2. 推荐的多通道读
3. 推荐的同步方法
4. 推荐的超时方法