链滴

初识 golang 资源竞争

作者：selfjt

原文链接：https://ld246.com/article/1573019528110

来源网站：链滴

go语言的的资源竞争问题.使用原子函数保证原子性操作.可以解决.

一、概述

这里利用原子方法进行标记(StoreInt64) 然后LoadInt64取的标记值，这个一系列操作是安全访问的动了两个 goroutine，并完成一些工作。在各自循环的每次迭代之后， goroutine 会使用LoadInt64 来检查 shutdown 变量的值。这个函数会安全地返回shutdown 变量的一个副本。如果这个副本的为 1， goroutine 就会跳出循环并终止。

main 函数使用 StoreInt64 函数来安全地修改 shutdown 变量的值。如果哪个 doWork goroutine 图在 main 函数调用 StoreInt64 的同时调用 LoadInt64 函数，那么原子函数会将这些调用互相同步保证这些操作都是安全的，不会进入竞争状态.

二、代码

创建变量

```
var (
    //定义一个shutdown是通知正在执行的goroutine停止工作标注
    shutdown int64
    //定义一wg4等待程序结束
    wg4 sync.WaitGroup
)
```

创建doWork

这个是用来模拟工作的goroutine

检查之前的shutdown标志来确定是否提前终止

```
func doWorks(name string) {
    defer wg4.Done()
    for {
        fmt.Printf("Doing %s Work\n", name)
        time.Sleep(250 * time.Microsecond)
        //判断是否要停止工作
        if atomic.LoadInt64(&shutdown) == 1 {
            fmt.Printf("Doing %s Down\n", name)
            //跳出循环
            break
        }
    }
}
```

创建main

```
func main() {
    //建立两个goroutine 计数器2
    wg4.Add(2)
    //使用两个协程 goroutine
    go doWorks("A")
    go doWorks("B")
    //给goroutine的运行时间
    time.Sleep(1 * time.Second)
    //该停止工作了,安全设置shutdown标志
    fmt.Println("Shutdown Now")
    atomic.StoreInt64(&shutdown, 1)
    //等待goroutine结束
    wg4.Wait()
}
```

记得导入包.

参考

- GO 源码