

Go语言入门到精通

江洲老师云课堂

主讲: 江洲老师

Go并发2





PART 01 runtime包



runtime包

1.runtime包

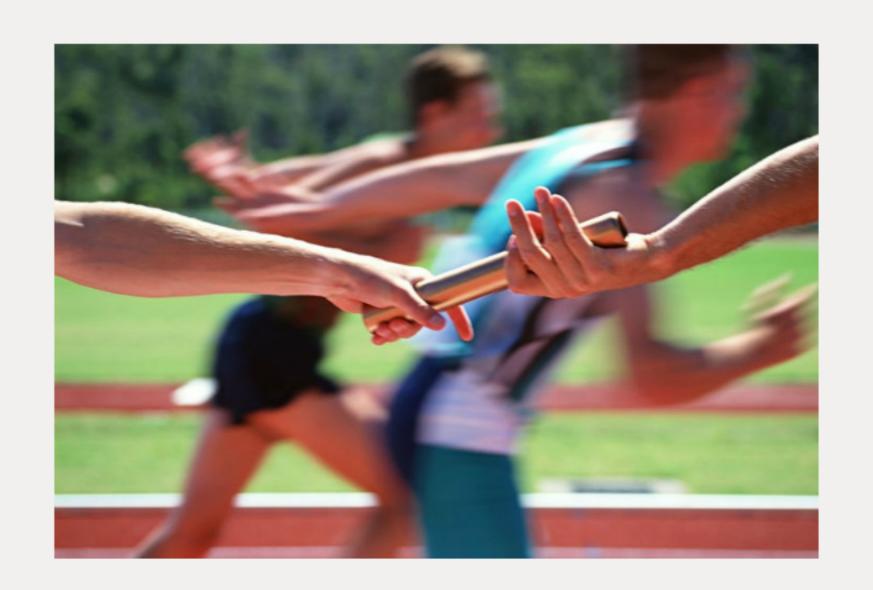
Gosched

runtime.Gosched() (go schedule)用于让出CPU时间片,让出当前goroutine的执行权限,调度器安排其他等待的任务运行,并在下次再获得cpu时间轮片的时候,从该出让cpu的位置恢复执行。

有点像跑接力赛,A跑了一会碰到代码runtime.Gosched()就把接力棒交给B了,A歇着了,B继续跑。

程序的执行过程:

主协程进入main()函数,进行代码的执行。 当执行到go func()匿名函数时,创建一个新的协程, 开始执行匿名函数中的代码,主协程继续向下执行, 执行到runtime.Gosched()时会暂停向下执行,直到 其它协程执行完后,再回到该位置,主协程继续向下 执行。





Goexit

调用 runtime.Goexit() 将立即终止当前 goroutine 执行,调度器确保所有已注册 defer 延迟调用被执行。

```
package main
import (
"fmt"
"runtime"
func main() {
 go func() {
   defer fmt.Println("A.defer")
   func() {
                                                              程序运行结果:
     defer fmt.Println("B.defer")
                                                                                 B.defer
     runtime.Goexit() // 终止当前 goroutine, import "runtime"
    fmt.Println("B") // 不会执行
                                                                                 A.defer
   }()
   fmt.Println("A") // 不会执行
 }() //不要忘记()
 //死循环,目的不让主goroutine结束
 for {
```





GOMAXPROCS

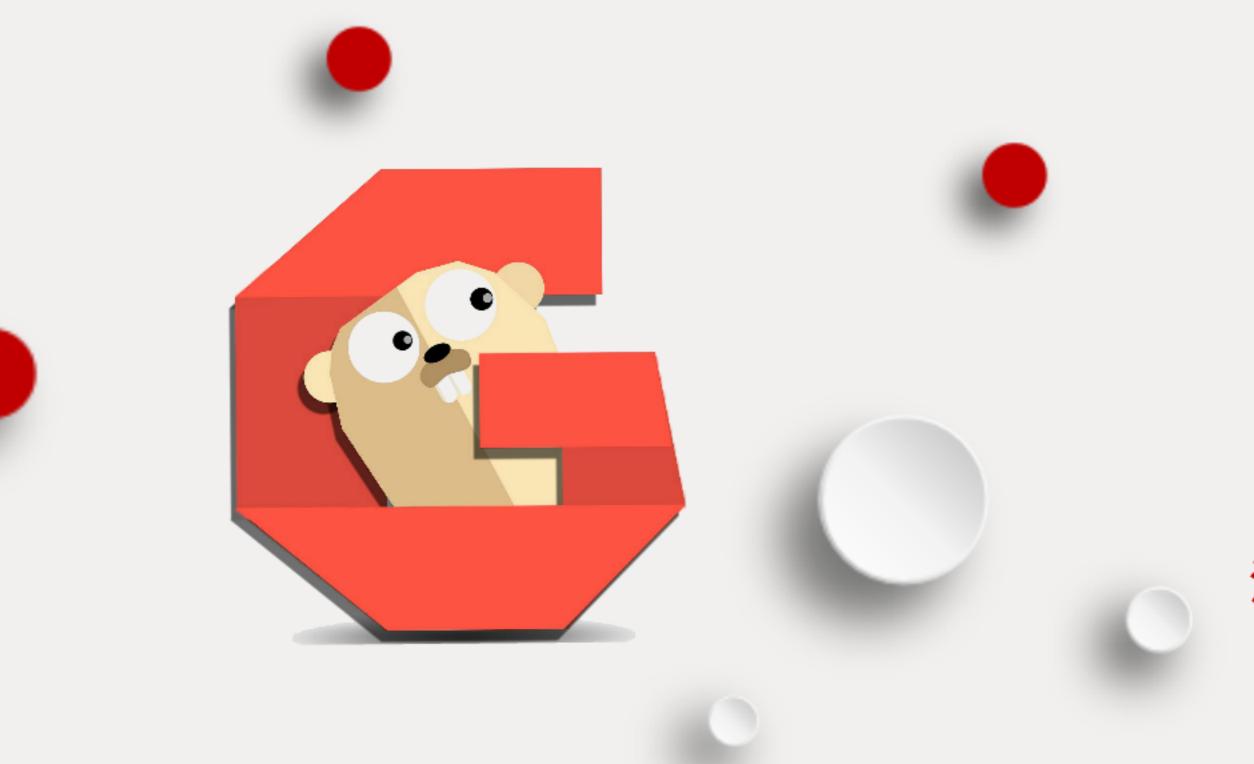
调用 runtime.GOMAXPROCS() 用来设置可以并行计算的CPU核数的最大值,并返回之前的值。

```
package main
                   在第一次执行runtime.GOMAXPROCS(1)时,最多同时只能有一个goroutine被执
import (
                   行。所以会打印很多1。
 "fmt"
                   过了一段时间后,GO调度器会将其置为休眠,并唤醒另一个goroutine,这时候
                   就开始打印很多0了,在打印的时候,goroutine是被调度到操作系统线程上的。
func main() {
//n := runtime.GOMAXPROCS(1)   // 第一次 测试
n := runtime.GOMAXPROCS(2)  // 第二次 测试
fmt.Printf("n = %d\n", n)
 for {
                   在第二次执行runtime.GOMAXPROCS(2) 时, 我们使用了两个CPU,所以两个
  go fmt.Print(0)
                   goroutine可以一起被执行,以同样的频率交替打印0和1。
  fmt.Print(1)
```



实例演示





江洲老师云课堂

— 主讲: 江洲老师

感谢您的聆听和观看