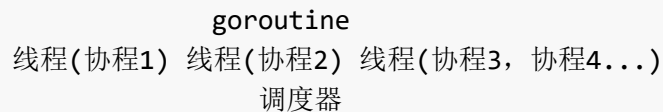


goroutine

goroutine的优点

1: 非抢占式多任务处理,由协程主动交出控制权 2: 编译器/解释器/虚拟机层面的多任务 3: 多个协程可能在一个或多个线程上运行(由调度器决定的)

goroutine的图示



```
graph TD
    G[goroutine] --- T1[线程(协程1)]
    G --- T2[线程(协程2)]
    G --- T3[线程(协程3, 协程4...)]
    T1 --- S[调度器]
    T2 --- S
    T3 --- S
```

1:调度器在合适的点进行切换 2:可以使用-race来检测冲突的点

goroutine的可能的切换点

1: I/O,select 2: channel 3: 等待锁 4: 函数调用(有时) 5: runtime.Gosched()-----用于让出控制权使别的协程去工作。只是参考，具体的切换还是需要调度器。

例子

```
1:
func main(){
    for i:=0;i<10;i++){
        go func(i int){
            fmt.Println(i)
        }()
    }
    time.Sleep(time.MIN)
}
```