main.go

**func** init() {

fmt.Println("我是init函数")

fmt.Println("在main函数执行之前的init函数")

}

**func** main() {

fmt.Println("我是main函数")

}

init函数在这个文件执行之前，默认执行

===============================================================================

    //添加记录CPU使用率的文件--------------------

    cpuf, err1 := os.Create("cpu\_profile")

    if err1 != nil {

        log.Fatal(err1)

    }

    pprof.StartCPUProfile(cpuf)

    defer pprof.StopCPUProfile()

    //添加记录CPU使用率的文件--------------------

添加记录cpu时间代码

===============================================================================

go分析程序性能

go tool pprof server cpu\_profile

peek 函数名

list 函数名

查看进程信息

ps -aux|grep server

USER PID %CPU %MEM VSZ RSS TTY STAT START TIME COMMAND

USER: 行程拥有者

PID: pid

%CPU: 占用的 CPU 使用率

%MEM: 占用的记忆体使用率

VSZ: 占用的虚拟记忆体大小

RSS: 占用的记忆体大小

TTY: 终端的次要装置号码 (minor device number of tty)

STAT: 该行程的状态:

D: 无法中断的休眠状态 (通常 IO 的进程)

R: 正在执行中

S: 静止状态

T: 暂停执行

Z: 不存在但暂时无法消除

W: 没有足够的记忆体分页可分配

<: 高优先序的行程

N: 低优先序的行程

L: 有记忆体分页分配并锁在记忆体内 (实时系统或捱A I/O)

START: 行程开始时间

TIME: 执行的时间

COMMAND:所执行的指令

在vscode里面的testing，显示打印信息。

"go.testFlags": ["-v"]





