前情回顾

http协议：就是一种信息发送请求接收的固定格式。

io input output

在内存中存在数据交换的操作都可以认为是IO操作

和终端交互：input print

和磁盘交互： read write

和网络交互： recv send

IO密集型程序：在程序执行过程中存在大量IO操作，而cpu运算操作较少。消耗CPU较少，运行效率较低

计算密集型程序（CPU密集型程序）：在程序执行中CPU运算较多，IO操作相对较少。消耗CPU大，运行书读快。 （算法优化。）

IO分类

阻塞IO 非阻塞IO IO多路复用

阻塞IO、

阻塞IO是IO的默认形态，是效率较低的一种IO情形。

阻塞情况

\*因为某种条件没有达成造成的阻塞

e.g accept input recv

\*处理IO数据传输时间较长形成的阻塞

e.g 网络传输过程，文件读写过程

非阻塞IO

通过修改IO时间的属性，使其变为非阻塞状态。（让一些条件阻塞函数不再阻塞）

\*非阻塞IO往往和循环判断一起使用

s.setblocking(False)

将套接字设置为非阻塞状态。

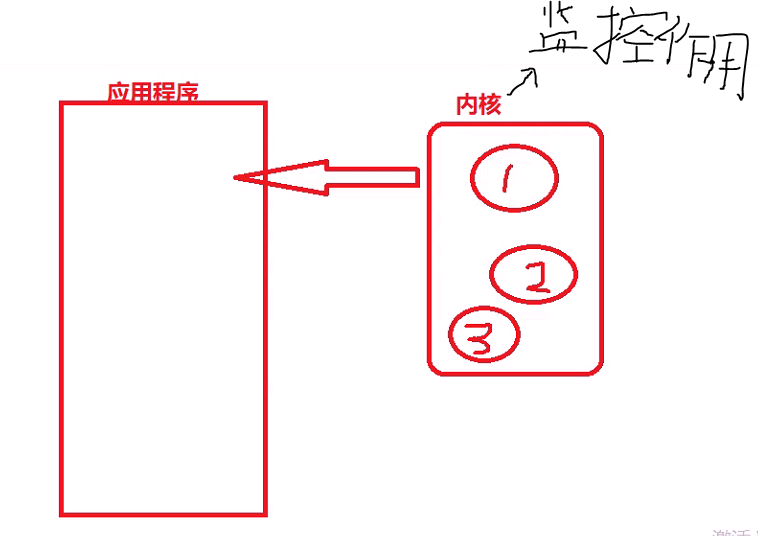
超时检测

将原本阻塞的函数设置一个最长阻塞时间。如果时间内条件达成则正常运行，如果任然阻塞则视为超时，继续向下运行或产生异常。

s.settimeout(sec)

设置套接字的超时时间

IO多路复用



定义：同时监控多个IO事件，当哪一个IO事件准备就绪就执行哪个IO事件，以此形成可用同时操作多个IO的并法行为，避免一个IO阻塞，造成所有IO都无法执行。

IO准备就绪： 是一种IO必然要发生的临界状态

IO多路复用的 编程实现

1. 将IP设置为关注IO
2. 将关注IO提交给内核监测
3. 处理内核给我们反馈的准备就绪 的IO

具体方案：

selec 🡪 windows linux unix

poll 🡪 linux unix

epoll 🡪 linux unix

import select

rs,ws,,xs = select(rlist,wlist,xlist[,timeout])

功能： 监控IO事件，阻塞等待IO事件发生

参数：

rlist 列表 存放我们监控等待处理的IO事件

wlist 列表 存放我们要主动操作的IO事件

xlist 列表 我们要关注出错处理的IO事件

timeout 超时时间

返回值： rs 列表 rlist中准备就绪的IO

ws 列表 wlist中准备就绪的IO

xs 列表 xlist中准备就绪的IO

注意：1、wlist中如果有IO事件则select立即会返回为ws

2、在处理IO过程中不要处理一个客户端长期占有服务端使服务端无法运行到select的情况

3、IO多路复用占用计算机资源少，IO效率高

位运算：

整数按照二进制位进行运算

& 按位与 | 按位或 ^按位异或

<<左移 >>右移

14和11

即：1110和1011

& 1010 一0则0

| 1111 一1则1

^ 0101 相同为0不同为1

11<<2 101100

14>>2 11

poll

1. 创建poll对象

p = select.poll()

1. 添加注册事件

p.register(s)

p.register(s，POLLIN |POLLERR)

POLLIN POLLOUT POLLERR POLLHUP POLLNVAL

rlisr wlist xlist 断开 无效数据

p.unregister(s) 从关注事件中移除

1. 阻塞等待IO发送

events = p.poll()

功能：阻塞等待IO发生

返回值：events是一个列表，列表中给每一个元素都是一个元组，代表一个发生的IO事件

[(fileno, event),(),()…..]

就绪IO的 文件描述符 具体就绪事件

\*需要通过文件描述符（fileno）找到对应的IO对象

{s.fileno() : s}

4、处理具体的IO