套接字传输注意事项

1. 监听套接字存在客户端即可发起连接，但是最终连接的处理需要accept进行处理。
2. 如果连接的另外一端退出，则recv会立即返回空字符串不在阻塞。
3. 当连接的另一端退出时，在试图send发送就会产生。Brokenpipeerror.

网络收发缓冲区

缓冲区作用：协调收发（处理）速度

减少交互次数

Send和recv实际上是和缓冲区进行交互，发送缓冲区满时就无法发送，接收缓冲区满时recv才阻塞

TCP粘包

产生的原因：

TCP套接字以字节流方式传输，没有消息边界，发送和接收并不能保证每次发送都及时的被接收。

影响：

如果每次发送内容表达一个独立的含义此时可能需要处理粘包防止产生歧义。

处理方法：

1. 每次发送的消息添加结尾标志（人为增加消息边界）
2. 发送数据结构体
3. 协调收发速度，每次发送后都预留接收时间

基于udp套接字的服务端

1. 创建数据报套接字

Sockfd = sockfd(SF\_INET,SOCK\_DGRAM)

1. 绑定地址

Sockfd.bind(addr)

1. 消息的收发

Data,addr = sockfd.recvfrom(buffersize)

功能 ： 接收UDP消息

参数 ： 每次最多接收多大的消息

返回值：data 接收到的数据

addr消息发送端的地址

\*一次接收一个数据报，如果数据报 大小大于buffersize则会丢失部分消息。

sockfd.sendto(data,addr)

功能 ：发送udp消息

参数 ：data发送的消息bytes格式

addr目标地址

返回值：发送的字节数

1. 关闭套接字

sockfd.close()

udp客户端

1. 创建套接字

socket（AF\_INET,SOCK\_DGRAM）

1. 消息收发

recvfrom/sendto

1. 关闭套接字

close()

cookie

1、sys.argv属性

功能 ： 获取命令行参数，得到一个列表

命令本身是argv[0]

后面的参数从argv[1]开始，默认以空格分隔，使用引号引起来的内容算作一个整体，

命令行参数都以字符串放入列表

2、在程序的第一行加

#！/usr/bin/env python3

添加程序的执行权限

chmod 755 file.py

修改后即可通过 ./file.py运行程序

TCP套接字编程和udp套接字 编程的区别

1. 流式套接字使用字节流的方式传输，数据报套接字以数据报形式传输数据
2. TCP会有粘包现象，udp有消息边界不会想成粘包
3. TCP可以保证数据传输完整性，udp则不保证
4. TCP需要进行listen accept操作，udp不需要
5. TCP收发消息使用新的套接字，recv send。udp使用recvfrom,sendto.

补充函数

sendall （）

功能 ： 发送TCP消息

参数 ： 要发送的内容，bytes格式

返回值 ： 发送成功返回none，失败产生异常

套接字对象

s代表一个套接字

s.family ： 获取套接字地址族类型 (AF\_INET)

s.type ： 获取套接字类型(SOCK\_ATREAM)

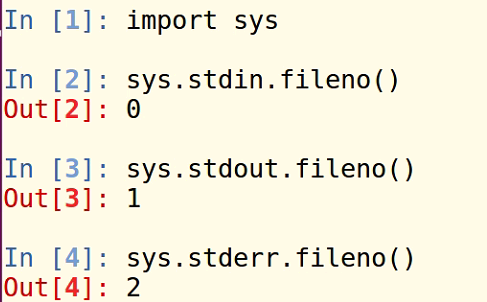
s.getsockname() ： 获取套接字的绑定地址

s.fileno() ： 获取套接字的文件描述符

文件描述符：每一个io事件操作系统都会分配一个不同 的正整数作为编号，

改正整数即为这个io的文件描述符。

\*文件描述符是操作系统识别io的唯一标志



s.getpeername(): 获取客户端连接套接字的对应地址

s.setsockopt(level，option，value) ：

功能：设置套接字选项 ，丰富或者修改套接字属性功能

参数： level选项类别 SOL\_SOCKET

option具体选项

value 选项值

s．getsockopt(level，option，value)

功能：获取套接字选项值

参数： level选项类别SOL\_SOCKET

option具体选项

返回值： 选项值

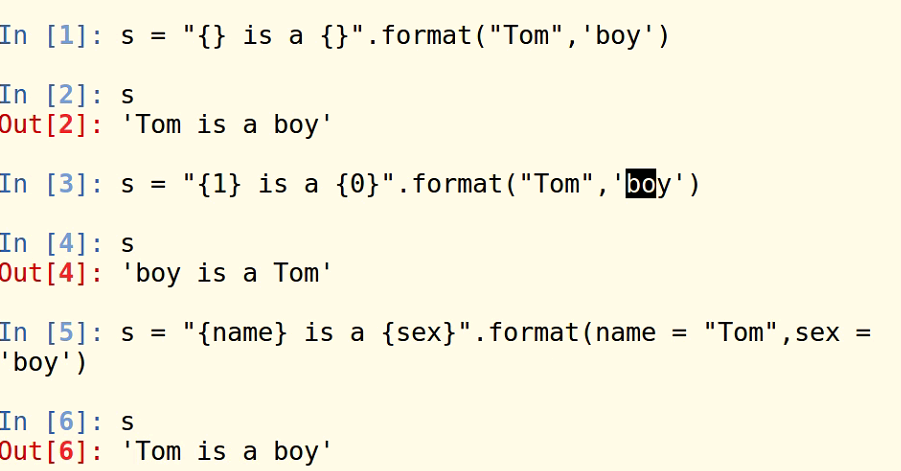
\*如果要设置套接字选项，最好在创建套接字之后立即设置

udp应用之广播

广播：一点发送，多点接收

广播地址：一个网段内有一个指定的广播地址，是该网段的最大地址、

176.233.4.255



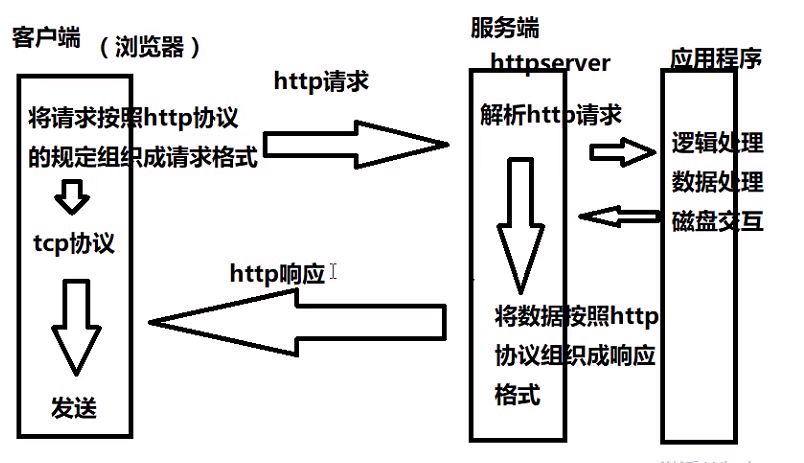
广播风暴：一个网络中有大量的广播就会产生广播风暴 占用大量带宽，影响正常访问速度。

TCP应用之http传输

http协议：超文本传输协议，是一个应用层协议

用途 ：网页数据的传输

数据传输方法



特点：

1. 应用层协议，传输层使用TCP服务
2. 简单，灵活，多种语言都有HTTP相关操作接口
3. 无状态的协议，即不记录用户传输的信息
4. http1.1支持持久连接，
5. 一端通过http请求的格式发送具体请求内容，另一端接收http其扭曲，按照协议格式解析。获取真

实请求后按照http协议响应格式组织回复内容，回发给请求方，完成一次交互。

http请求（request）、

请求格式：

请求行：具体的请求类别和请求 内容

格式：GET / HTTP/1.1

请求类别 请求内容 协议版本

请求类别 ： 表示请求的种类

GET 获取网络资源

POST 提交一定的附加 信息，得到返回结果

HEAD 获取响应头

PUT 更新服务器资源

DELETE 删除服务器 资源

CONNECT

TRACE 用于测试

OPTIONS 获取服务器性能信息

请求头 ： 对请求内容的具体描述信息

空行

请求体 ： 请求参数或者是提交内容

http响应：（response）

响应格式

响应行：反馈响应的情况

格式： http/1.1 200 ok

协议版本 响应码 附加信息

响应码： 响应的具体情况

1xx: 提示信息，表示请求成功

2xx: 响应成功

3xx：响应需要重定向

4xx：客户端错误

5xx：服务端错误

常见响应码： 200 成功

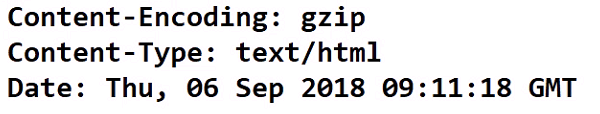
404 请求内容不存在

401 没有访问权限

500 服务器发生未知错误

503 暂时无法执行

响应头 ： 对相应内容的具体描述



空行

响应体: 返回给请求端的 具体内容

要求 ：

1. 知道什么是http协议
2. 知道请求的格式和每一部分的功能
3. 响应的格式和每一部分功能
4. 常见的请求类型和响应码代表什么

