## 网络编程

Socket(套接字)：打开一个Socket需要知道目标主机的IP地址和端口号，再制定协议类型。

Socket：提供了标准的BSD Sockets API

SocketServer：提供了服务器中心类，可以简化网络服务器的开发

模块功能：

1. Socket类型

套接字格式：socket(family,type[,protocal]),使用给定的地址族、套接字类型

|  |  |
| --- | --- |
| Socket类型 | 描 述 |
| socket.AF\_UNIX | 只能够用于单一的Unix系统进程间通信 |
| socket.AF\_INET | 服务器之间网络通信 |
| socket.AF\_INET6 | IPv6 |
| socket.SOCK\_STREAM | 流式socket，用于TCP |
| socket.SOCK\_DGRAM | 数据报式socket，用于UDP |
| socket.SOCK\_RAW | 原始套接字，普通的套接字无法处理ICMP、IGMP等网络报文，而SOCK\_RAW可以；其次，SOCK\_RAW也可以处理特殊的IPv4报文；此外，利用原始套接字，可以通过IP\_HDRINCL套接字选项由用户构造IP头。 |
| socket.SOCK\_SEQPACKET | 可靠的连续数据包服务 |
| 创建TCP Socket | S=socket.socket(socket.AF\_INET,socket.SOCK\_STREAM) |
| 创建UDP Socket | S=socket.socket(socket.AF\_INET,socket.SOCK\_DGRAM) |

Socket函数



### TCP编程

网络编程包括两个部分：服务端和客户端

TCP是一种面向连接的通信方式，主动发起连接的叫客户端，被动响应连接的叫服务端。

创建和运行TCP服务端的五个步骤：

1. 创建Socket，绑定Socket到本地IP与端口
2. 开始监听连接
3. 进入循环，不断接收客户端的连接请求
4. 接收传来的数据，并发送给对方数据
5. 传输完毕后，关闭Socket

示例：

import socket  
import threading  
import time  
  
  
def dealClient(sock,addr):  
 # 第四步：接收传来的数据，并发送给对方数据  
 print('Accept new connection from %s:%s...' % addr)  
 sock.send(b'Hello,I am pjsServer!')  
 while True:  
 data = sock.recv(1024)  
 time.sleep(1)  
 if not data or data.decode('utf-8') == 'exit':  
 break  
 print('--->>>%s!' % data.decode('utf-8'))  
 sock.send(('Loop\_Msg: %s!' % data.decode('utf-8')).encode('utf-8'))  
  
 # 第五步：关闭Socket  
 sock.close()  
 print('Connection from %s:%s closed.' % addr)  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 # 第一步：创建一个基于IPv4和TCP协议的Socket  
 # Socket绑定的IP（127.0.0.1为本机IP）与端口  
 s = socket.socket(socket.AF\_INET,socket.SOCK\_STREAM)  
 s.bind(('127.0.0.1',9999))  
  
 # 第二步：监听连接  
 s.listen(5)  
 print('Waiting for connection...')  
 while True:  
 # 第三步：接收一个新连接  
 sock,addr = s.accept()  
 # 创建新线程来处理TCP连接  
 t = threading.Thread(target=dealClient,args=(sock,addr))  
 t.start()

接着编写客户端，与服务端进行交互。需要三个步骤

1. 创建Socket，连接远端地址
2. 连接后发送数据和接收数据
3. 传输完毕后，关闭Socket

代码如下：

import socket  
  
#初始化Socket  
s = socket.socket(socket.AF\_INET,socket.SOCK\_STREAM)  
# 连接目标Ip和端口  
s.connect(('127.0.0.1',9999))  
#接收消息  
print('---->>' + s.recv(1024).decode('utf-8'))  
#发送消息  
s.send(b'Hello,I am a pjsClient')  
print('---->>' + s.recv(1024).decode('utf-8'))  
s.send(b'exit')  
#关闭Socket  
s.close()