# TÀI LIỆU CHO NHÀ PHÁT TRIỂN

MỤC LỤC

[TÀI LIỆU CHO NHÀ PHÁT TRIỂN 1](#_Toc487882238)

[A.Giới thiệu 1](#_Toc487882239)

[I. Module Socket 2](#_Toc487882240)

[B.Chương trình 5](#_Toc487882241)

[I. Lớp xử lý Server(Server\_BLL.py) 5](#_Toc487882242)

[II.Lớp xử lý dữ liệu chung(Process\_BLL.py) 9](#_Toc487882243)

[III. Lớp xử lý client (Client\_BLL.py) 10](#_Toc487882244)

## A.Giới thiệu

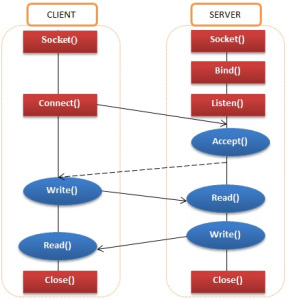
### I. Module Socket

Socket là một cổng logic mà một chương trình sử dụng để kết nối với một chương trình khác chạy trên một máy tính khác trên Internet. Chương trình mạng có thể sử dụng nhiều Socket cùng một lúc, nhờ đó nhiều chương trình có thể sử dụng Internet cùng một lúc. có 2 loại socket thường thấy đó là dùng giao thức TCP hoặc UDP

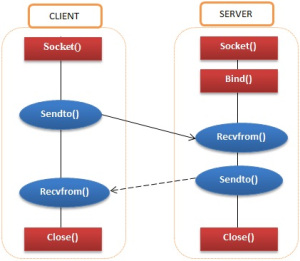
1. Stream Socket: Dựa trên giao thức TCP( Tranmission Control Protocol) việc truyền dữ liệu chỉ thực hiện giữa 2 quá trình **đã thiết lập kết nối**. Giao thức này đảm bảo dữ liệu được truyền đến nơi nhận một cách đáng tin cậy, đúng thứ tự nhờ vào cơ chế quản lý luồng lưu thông trên mạng và cơ chế chống tắc nghẽn.
2. Datagram Socket: Dựa trên giao thức UDP( User Datagram Protocol) việc truyền dữ liệu **không yêu cầu có sự thiết lập kết nối giữa 2 quá trình**. Ngược lại với giao thức TCP thì dữ liệu được truyền theo giao thức UDP không được tin cậy, có thế không đúng trình tự và lặp lại. Tuy nhiên vì nó không yêu cầu thiết lập kết nối không phải có những cơ chế phức tạp nên tốc độ nhanh…ứng dụng cho các ứng dụng truyền dữ liệu nhanh như chat, game…..

Một TCP/IP Socket gồm một địa chỉ IP kết hợp với một port xác định duy nhất một tiến trình (process ) trên mạng.Hay nói cách khác luồng thông tin trên mạng dựa vào IP là để xác định một máy trên mạng còn port xác định 1 tiến trình trên 1 máy.

**Cơ chế gọi hàm trong lập trình socket**

[](https://c6h0st.files.wordpress.com/2015/10/tcp2.jpg)

*Lập trình Socket với TCP*

[](https://c6h0st.files.wordpress.com/2015/10/udp2.jpg)

**Python Socket module**

|  |  |
| --- | --- |
| **Term** | **Description** |
| domain | Họ của các giao thức dùng để truyền dữ liệu: AF\_INET, PF\_INET, PF\_UNIX, PF\_X25, ….  Ở đây chúng ta thường dùng AF\_INET (IPv4) |
| type | Kiểu của socket: SOCK\_STREAM (TCP), SOCK\_DGRAM (UDP) |
| protocol | Dùng để xác định các biến thể của các giao thức, mặc định là 0 |
| hostname | Dùng để xác định hostname, có thể là:   * Chuỗi * Địa chỉ IPv4 nếu dùng AF\_INET * Địa chỉ IPv6 nếu dùng AF\_INET6 |
| port | Xác định port của socket |

Để tạo 1 socket ta làm như sau:

s = socket.socket (socket\_family, socket\_type)

Với socket\_family, socket\_type đã được giới thiệu bên trên (Domain, type)

Sau khi khởi tạo được 1 đối tượng socket, tiếp đến ta có thể khởi tạo chương trình cho Client hoặc Server

**Server Socket Methods**

|  |  |
| --- | --- |
| **Method** | **Description** |
| s.bind() | Kết nối (hostname, port number) đến socket. |
| s.listen() | Cho Server lắng nghe các Client |
| s.accept() | Thiết lập kết nối giữa Client-Server |

**Client Socket Methods**

|  |  |
| --- | --- |
| **Method** | **Description** |
| s.connect() | Kết nối đến Server |

**General Socket Methods**

|  |  |
| --- | --- |
| **Method** | **Description** |
| s.recv() | Nhận dữ liệu trong giao thức TCP |
| s.send() | Truyền dữ liệu trong giao thức TCP |
| s.recvfrom() | Nhận dữ liệu trong giao thức UDP |
| s.sendto() | Truyền dữ liệu trong giao thức UDP |
| s.close() | Đóng kết nối |
| socket.gethostname() | Trả về tên của host |

**Đây là 1 đoạn code TCP Socket đơn giản**

**Server side**

import socket # Import socket module

s = socket.socket() # Tạo đối tượng socket (mặc định là TCP trên nền IPv4)

host = socket.gethostname() # Xác định tên localhost

port = 12345 # Xác định port muốn dùng

s.bind((host, port)) # Kết nối đến socket

s.listen(5) # Server lắng nghe 5 Client cùng 1 lúc

while True:

c, addr = s.accept() # Thiết lập kết nối

print 'Got connection from', addr

c.send('Thank you for connecting') #Gửi dữ liệu đến Client

c.close() # Đóng kết nối

**Client side**

import socket # Import socket module

s = socket.socket() # Tạo đối tượng socket

host = socket.gethostname() # Xác định tên localhost

port = 12345 # Xác định port

s.connect((host, port)) #Kết nối đến Server

print s.recv(1024) #Nhận dữ liệu từ Server với Buffer là 1024 bytes

s.close() # Đóng kết nối

Chạy 2 script trên và đây là kết quả:

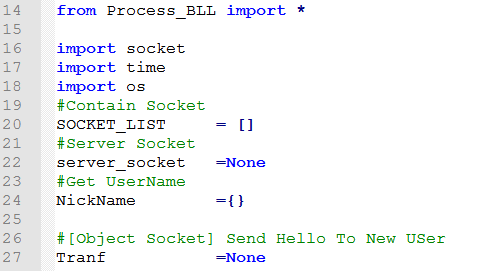
Got connection from ('127.0.0.1', 48437) #Server side

Thank you for connecting #Client side

## B.Chương trình

### Lớp xử lý Server(Server\_BLL.py)

#### 1.KHởi tạo một số biến



Trong đó:

Nạp module vào chương trình

import socket

import time

import os

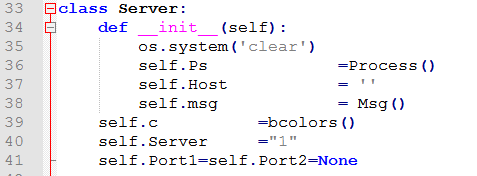
**Một số biến:**

SOCKET\_LIST: chứa danh sách các socket

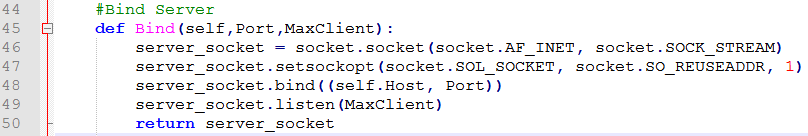
server\_socket: chứa địa chỉ của server

NickName: Chứa tên username của client

#### 2.Lớp khởi tạo Server



#### 3.Hàm Khởi tạo server: khởi tạo & trả về địa chỉ của server

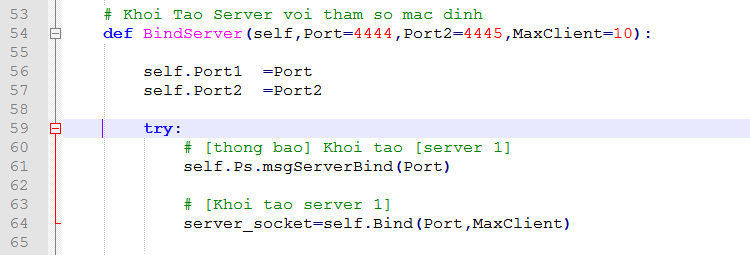


#### 4.Hàm xử lý khi tạo server: Trả về địa chỉ server

Server1: khởi tại ở cổng 8888

Server2: khởi tại ở cổng 8889

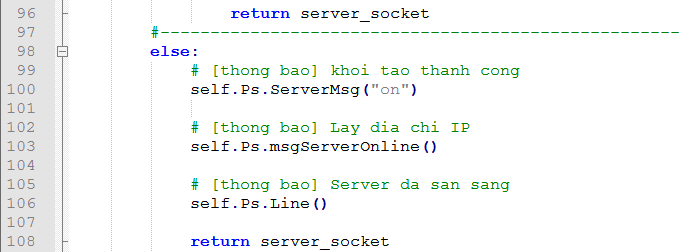
Nếu server 1 bận thì server 2 sẽ được mở



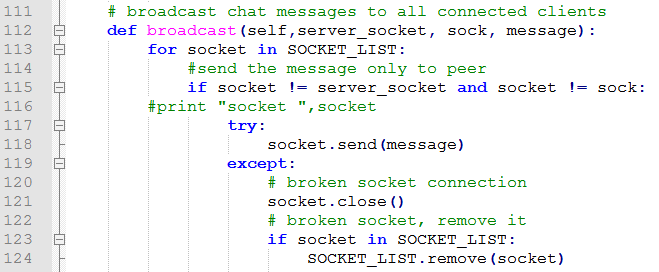
.

.

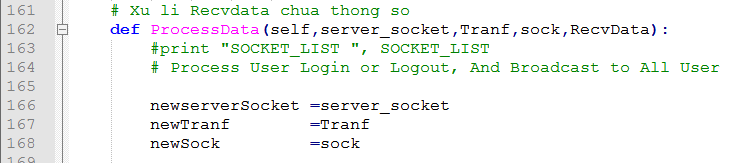
.



#### 5.Hàm gửi dữ liệu đến tất cả các user:

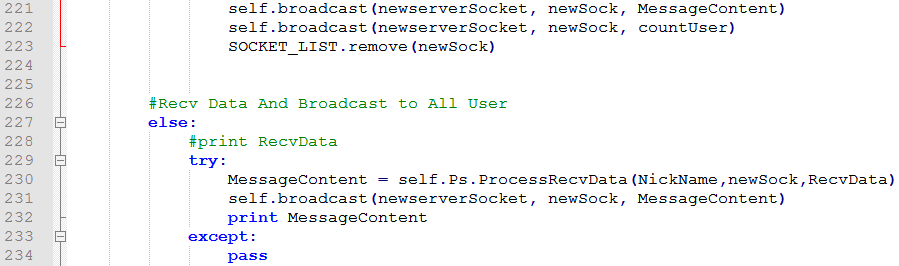


#### 6.Hàm xử lý dữ liệu giữa các user-client



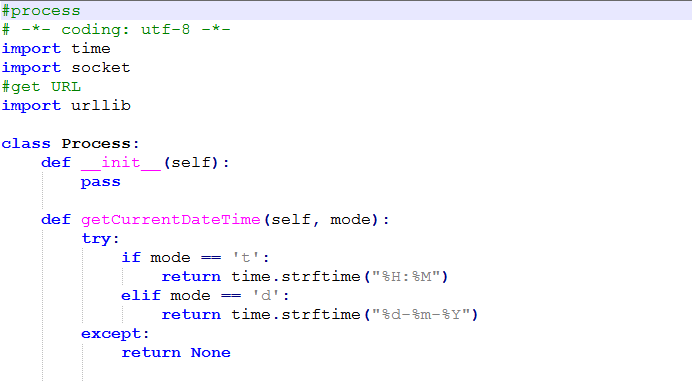
.

.

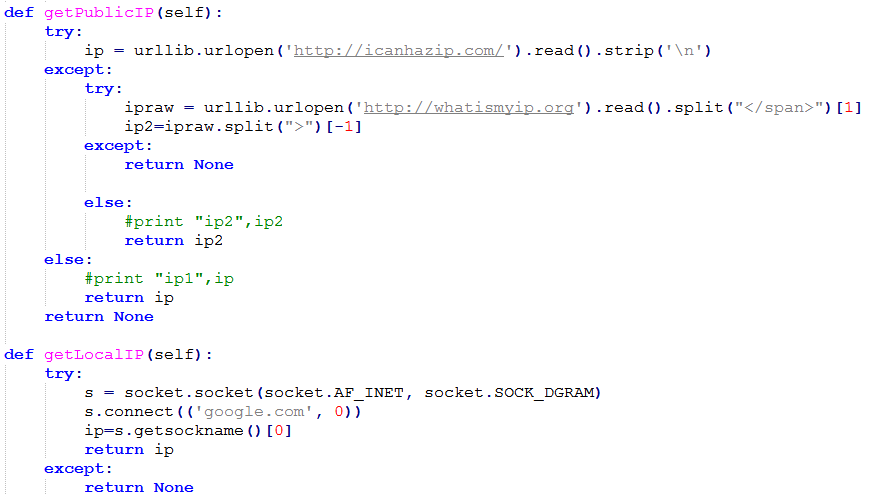
. 

### II.Lớp xử lý dữ liệu chung(Process\_BLL.py)

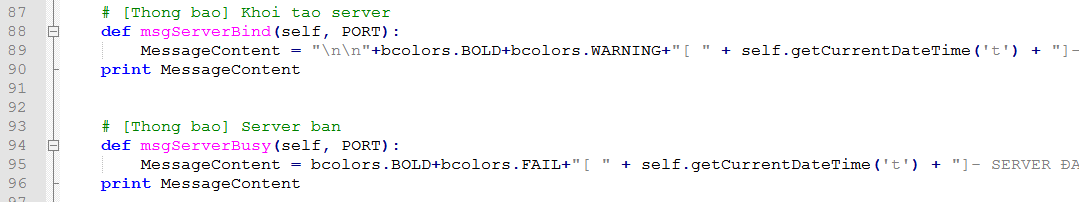
#### 1.Hàm lấy thông tin thời gian

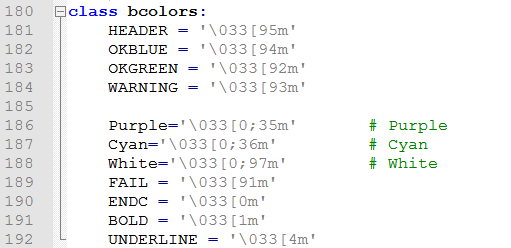


#### 2.Hàm lấy thông tin IP



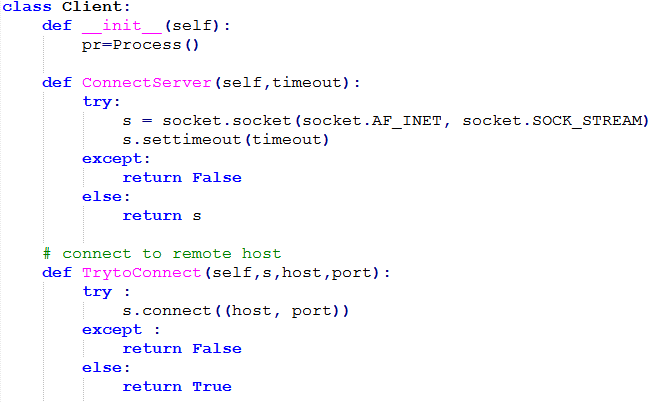
#### 3.Một số hàm xử lý dữ liệu màu sắc, nội dung





### III. Lớp xử lý client (Client\_BLL.py)

#### 1.Lớp khởi tạo client : Kết nối đến server và trả về địa chỉ socket



#### 2.Hàm xử lý dữ liệu nhận về và dữ liệu gửi đi

