socket

파이썬의 소켓 프로그래밍을 위한 라이브러리

예제

클라이언트

다음 예제는 google.com의 ip 주소를 알아낸 후 google.com 서버에 80 포트로 접속한다. 그리고 간단히 메시지를 주고 받는다. 웹브라우저 역시 이 예제와 동일한 방식으로 동작한다.

client.py

```
#Socket client example in python

import socket   #for sockets
import sys  #for exit

#create an INET, STREAMing socket
try:
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
except socket.error:
    print 'Failed to create socket'
    sys.exit()

print 'Socket Created'

host = 'www.google.com';
port = 80;

try:
    remote_ip = socket.gethostbyname( host )

except socket.gaierror:
    #could not resolve
    print 'Hostname could not be resolved. Exiting'
    sys.exit()

#Connect to remote server
s.connect((remote_ip , port))

print 'Socket Connected to ' + host + ' on ip ' + remote_ip

#Send some data to remote server
message = "GET / HTTP/1.1\r\n\r\n"

try :
    #Set the whole string
    s.sendall(message)
except socket.error:
    #Send failed
    print 'Send failed'
    sys.exit()

print 'Message send successfully'

#Now receive data
reply = s.recv(4096)

print reply

#Close the socket
s.close()
```

서버

다음 예제는 8888포트로 바인딩 후 client와의 연결을 기다리고 연결되면 accept 함수 실행 후 핸들링을 위한 쓰레드를 생성한다. 무한루프를 이용하여 계속해서 client 연결 대기와 핸들링을 수행한다.

server.py

```
import socket
import sys
from thread import *

HOST = ''   # Symbolic name meaning all available interfaces
PORT = 8888 # Arbitrary non-privileged port

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
print 'Socket created'

#Bind socket to local host and port
try:
    s.bind((HOST, PORT))
except socket.error , msg:
    print 'Bind failed. Error Code : ' + str(msg[0]) + ' Message ' +
msg[1]
    sys.exit()

print 'Socket bind complete'

#Start listening on socket
s.listen(10)
print 'Socket now listening'

```

```
#Function for handling connections. This will be used to create threads
def clientthread(conn):
    #Sending message to connected client
    conn.send('Welcome to the server. Type something and hit enter\n')
#send only takes string

    #infinite loop so that function do not terminate and thread do not
end.
    while True:

        #Receiving from client
        data = conn.recv(1024)
        reply = 'OK...' + data
        if not data:
            break

        conn.sendall(reply)

    #came out of loop
    conn.close()

#now keep talking with the client
while 1:
    #wait to accept a connection - blocking call
    conn, addr = s.accept()
    print 'Connected with ' + addr[0] + ':' + str(addr[1])

    #start new thread takes 1st argument as a function name to be run,
second is the tuple of arguments to the function.
    start_new_thread(clientthread ,(conn,))

s.close()
```

참고

- Python socket – network programming tutorial