**Server Side:**

#!/usr/bin/python # This is server.py file

import socket # Import socket module

s = socket.socket() # Create a socket object

host = socket.gethostname() # Get local machine name

port = 1234 # Reserve a port for your service.

s.bind((host, port)) # Bind to the port

s.listen(5) # Now wait for client connection.

#while True:

c, addr = s.accept() # Establish connection with client.

print('Got connection from', addr)

while(True):

#print("Receiving.....")

str=c.recv(1024)

print("Receiving.....",str)

if str==b"Bye":

print("Server Breaking connection\n")

break

str=input("enter ur message")

if not str :

break

c.send(str.encode())

#print("Received from client.....")

#print("Receiving from client....",c.recv(1024))

c.send(b"Bye")

c.close() # Close the connection

**Client Side:**

#!/usr/bin/python # This is client.py file

import socket # Import socket module

s = socket.socket() # Create a socket object

host = socket.gethostname() # Get local machine name

port = 1234 # Reserve a port for your service.

s.connect((host, port))

while(True):

str=input("Enter your message")

if not str :

break

s.send(str.encode())

print("Receiving.....")

str=s.recv(1024)

print(str)

if str==b"Bye":

print("Client Breaking connection\n")

break

s.send(b"Bye")

s.close # Close the socket when done

**Outputs :**

**Server Output :**

PS C:\programs\python> python chatserver.py

Got connection from ('192.168.29.219', 50092)

Receiving..... b' Checking 001'

enter ur message Checked 001

Receiving..... b' Checking 002'

enter ur message checked 002 .. bye

Receiving..... b''

**Client Output :**

PS C:\programs\python> python chatclient.py

Enter your message Checking 001

Receiving.....

b' Checked 001'

Enter your message Checking 002

Receiving.....

b' checked 002 .. bye

**Conclusion:**

From above experiment we learn to establish communication between two processes by implementing socket programming in python (client and server communication).

**Postlabs:**

1. What is a socket?

A socket is a core endpoint of a two-way communication link between tow programs running on the network. A socket is bound to a port number so that the TCP layer can identify the application that data is destined to be sent to. An endpoint is a combination of an IP Address and a port number.

1. What is a stream socket?

In computer operating systems, a stream socket is a type of inter-process communications socket or network socket which provides a connection-oriented, sequenced, and unique flow of data without record boundaries, with well-defined mechanisms for creating and destroying connections and for detecting errors.

1. Differentiate between datagram and stream socket

|  |  |
| --- | --- |
| Datagram Socket | Stream Socket |
| Datagram socket is a type of network socket which provides connection-less point for sending and receiving packets | Stream socket is a type of inter-process communications socket or network socket which provides a connection-oriented, sequenced, and unique flow of data |
| Every packet is individually routed and delivered | It provides sequenced delivery of packets |
| It is an unreliable mechanism with no error detection or prevention mechanisms | It is a reliable mechanism with fixed steps to creating and destroying connections and detecting errors |
| It uses UDP protocol for transmission | It uses TCP protocol for transmission |