1 way chat

Aim

To create a client program and server program. Client sends messages to server continuesly

Theory

BufferedReader br=new BufferedReader(new InputStreamReader(s.getInputStream())); is the instruction for creating object,which is used to read data from the socket s.
msg=br.readLine(); instruction reads data from the socket.
ServerSocket soc=new ServerSocket(8000); creates a server socket object, which can be used for accepting sockets from the client.
s=new Socket("localhost",8000); in client side program we can use this instructionsto create a socket for communicating with server
PrintWriter pw= new PrintWriter(s.getOutputStream(),true); is an instruction to create object pw, Using this pw we can write data to the socket using the instruction pw.println(msg);.

Algorithm

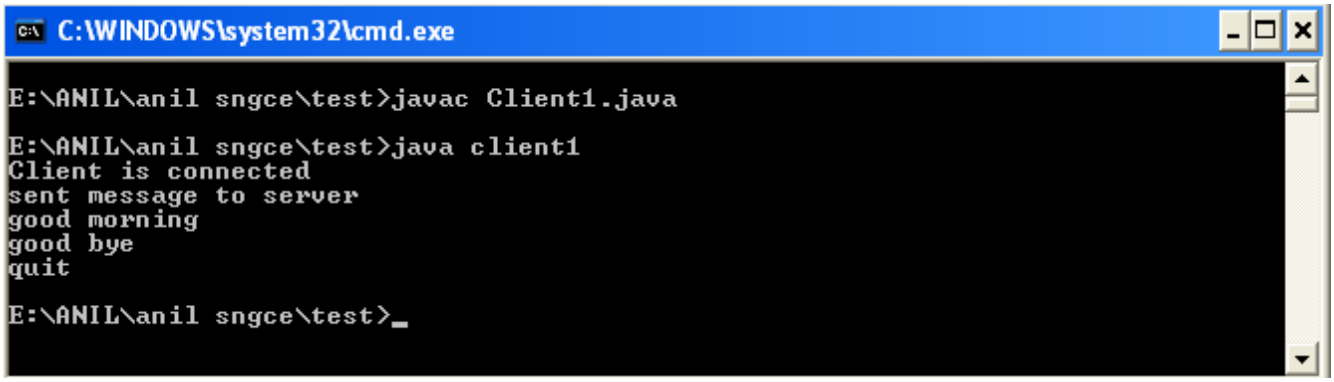
Create a class client1
main function
create a socket
create a printwriter object, which is used write data into socket
create a BufferedReader object, which is used to read data from the keyboard
do
read data from the keyboard
write data to the socket
end do if data is equal to 'quit'
end of main function
end of class client1

Create a class server1
main function
create a socket and serversocket
accept the client socket using serversocket object
create a BufferedReader object, which is used to read data from the socket
do
read data from the socket
write data to the screen
end do
end of main function
end of class server1

Test cases

Case1:
run one client program and one server program. Send message from client to server.
Case2:
run more than one client program and one server program. Send messages from clients to server.

Output


C:\WINDOWS\system32\cmd.exe
E:\ANIL\anil sngce\test>javac Client1.java
E:\ANIL\anil sngce\test>java client1
Client is connected
sent message to server
good morning
good bye
quit
E:\ANIL\anil sngce\test>


C:\WINDOWS\system32\cmd.exe
E:\ANIL\anil sngce\test>javac Server1.java
E:\ANIL\anil sngce\test>java Server1
Waiting for the clients to connect
good morning
good bye
quit
E:\ANIL\anil sngce\test>

MORE THAN ONE CLIENT TO SERVER DATA TRANSFER

Aim

To create a client program and server program. Clients send messages to server continuesly

Program description

Run server program on one machine. After that run client program on more than one machines. Enter data on the client window. Checks whether that data is displayed on the server window. In server machine we create a server class and thread class t1, for each client server class creates one instance of thread class t1. That thread reads data from the client and prints into the screen of the server.

Theory

BufferedReader br=new BufferedReader(new InputStreamReader(s.getInputStream())); is the instruction for creating object,which is used to read data from the socket s.
msg=br.readLine(); instruction reads data from the socket.
ServerSocket soc=new ServerSocket(8000); creates a server socket object, which can be used for accepting sockets from the client.
s=new Socket("localhost",8000); in client side program we can use this instructionsto create a socket for communicating with server
PrintWriter pw= new PrintWriter(s.getOutputStream(),true); is an instruction to create object pw, Using this pw we can write data to the socket using the instruction pw.println(msg);.

Algorithm

Create a class client1
main function
create a socket
create a printwriter object, which is used write data into socket
create a BufferedReader object, which is used to read data from the keyboard
do
read data from the keyboard
write data to the socket
end do
end of main function
end of class client1

Create a class server1
main function
create a socket and serversocket
create a BufferedReader object, which is used to read data from the socket
do
accept the client socket using serversocket object
start a thread for each client connection
end do
end of main function
end server1
create a class t1 which implements runnable interface
in run function
do
read data from the socket
write data to the screen
end do
end of class t1

Test cases

Case1:
run one client program and one server program. Send message from client to server.
Case2:
Run more than one instance of client program and one server program. Send messages from clients to server.

Output