Hacettepe University Computer Engineering Department		
Course	:	BBM342 Operating Systems
Experiment No	:	3
Subject	:	Socket Programming: Simple IRC
Advisers	:	Dr. Ahmet Burak CAN RA Kazim SARIKAYA
Due Date	:	7 Haz 2013

INTRODUCTION

The client–server model is a distributed application model in computing. This model distributes tasks or workloads between servers and clients. While servers provide resources or services, clients provide request services as shown in Figure 1. Often clients and servers communicate over a computer network on separate hardware using TCP/IP protocol stack. A server is a host that is running one or more server programs to share resources with clients. A client does not share any of its resources, but requests a server's content or service function. Clients therefore initiate communication sessions with servers, which await incoming requests. The client–server model was developed at Xerox PARC during the 1970s. It is now prevalent in computer networks. E-mail, World Wide Web, network printing are typical examples of this model.

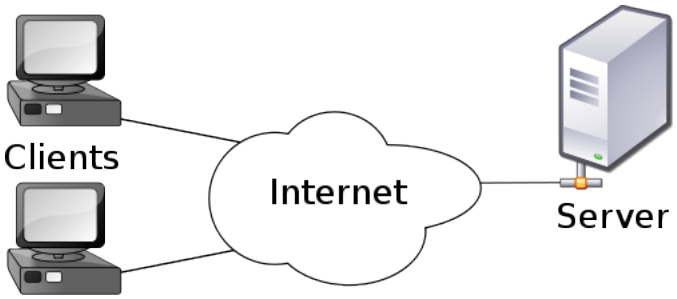


Figure 1: Server Client Model

IRC is a chat system on internet. IRC clients connect to an IRC server. An IRC server provides channels for users. After connecting to server, users (clients) join channels and send text messages public or private. Also, users can send files to each other. In Figure 2, you can see a chat window of a IRC client.

In this experiment you should implement a simple IRC server and client

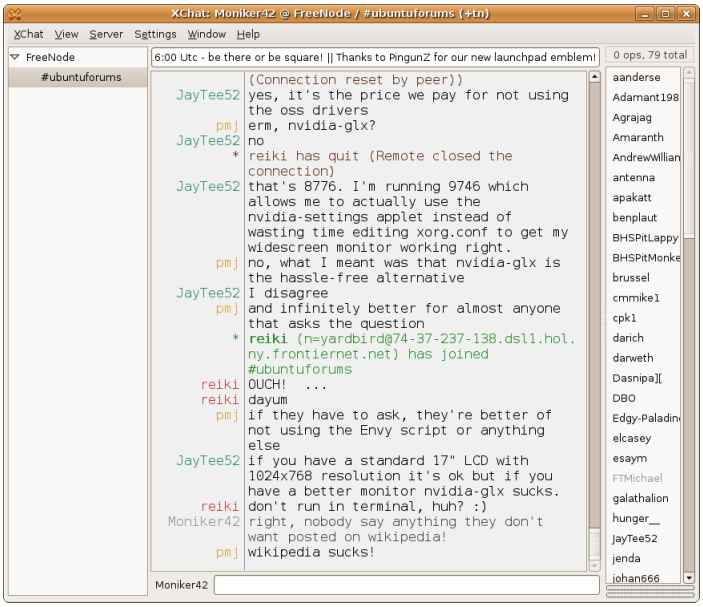


Figure 2: An IRC Client Window

using java programming language. The aim of the experiment is understanding server and client sockets, sending text messages between a server and clients, using threads with sockets.

EXPERIMENT DESCRIPTION

In the experiment, you should implement two Java programs.

IRC Server

You should implement a multithreaded TCP Server for a simple IRC system. The server program accepts two arguments from the command line. The first argument should be a port number which the server socket should bind and listen for client connections. Second argument is a channel file that contains a channel name in each line. Server manages channels and users on each channel. When a user connects to the server, the user may ask list of channels and the server should provide a channel list. A user should join only one channel. When a user joins to a channel, the server should add the user to the channel's user list. When a user sends a message to the channel, the server should send the message to all users in the channel. When a user disconnects or leaves the channel, the server should remove the user from channel's user list. When a user asks users joined the channel, server should send channel's user list to the client.

IRC Client

The client program accepts a nickname from the command line. The nickname will represent the client in IRC channels. You should implement a multithreaded TCP Client with gui for a simple IRC system. The gui should consist of two text areas. The first text area is to print commands, command results and messages. If a printed text originated from client, it should start with < to denote client to server. If a printed text originated from server, it should start with > to denote server to client. The second area is for writing commands and messages. A command should be started with the character /, messages should be started without any prefix. When a user types a command or message and pressed the **Enter** key, the command or message should be sent to the server. A server should process the command or message and then return the answer to the corresponding clients. A sample client screen is demostrated at Figure 3.

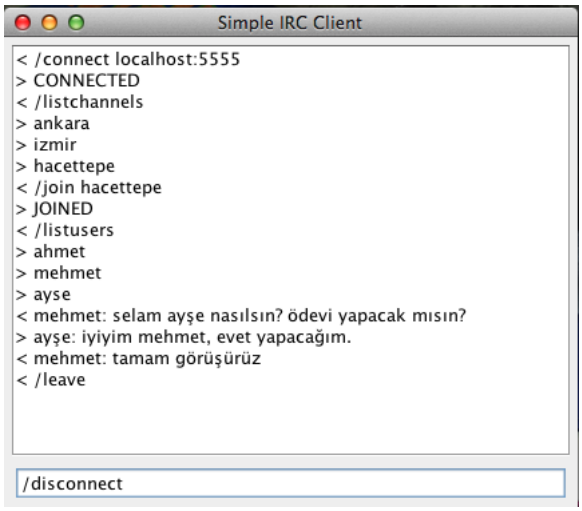


Figure 3: Sample Client of User Ahmet

Protocol Commands

- **/connect <server:port>**: connect command should connect client to the *server* at given *port*. Before this command, any other commands should throw an exception and *THERE IS NOT ANY CONNECTION* message should be printed in the first text area.
- **/disconnect**: disconnects from the connected server.
- **/listchannels**: list channels.
- **/join <channel>**: joins to user to the given *channel*. When a user joins to a channel, user should be appended to the channel's user list.

If the channel does not exist, the exception *THERE IS NOT SUCH A CHANNEL*.

- **/listusers**: After joining to a channel, this command send the channel's user list. If a user did not join to a channel, the exception *THERE IS NOT A JOINED CHANNEL* should be thrown.
- **/leave**: leaves from the joined channel. If user did not join a channel before this command, the exception *THERE IS NOT A JOINED CHANNEL* should be thrown.

In Java, you can create server and client sockets with *ServerSocket* and *Socket* classes respectively. This classes provide input and output functionality with stream classes. You can send and receive strings to/from sockets using *PrintWriter* and *Scanner* classes. A server can establish connections with more than one client. For providing connection and data transfer simultaneously, a server implementation should use multithreading. You can create threads with Java by using *Runnable* interface. While implementing a multithreaded application, you should implement thread-safe classes.

Messages

Each messages should be start with direction character (< or >). Then the username of sender, semicolon and the message should be followed at line.

REPORT

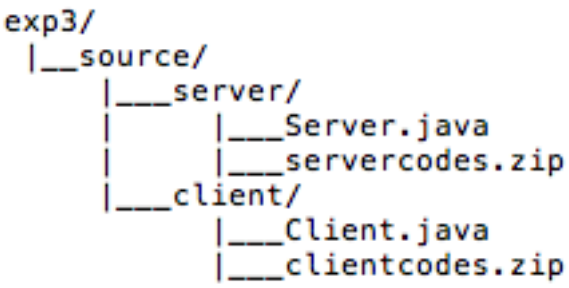
You should not write report for this experiment.

SUBMIT

You should submit your codes on https://submit.cs.hacettepe.edu.tr system. Your server main class should be Server.java (don't include package) and client main class should be Client.java (don't include package). You should create a server folder for server side and client folder for client side, then you should zip both folders and put into source directory.

NOTES

- SAVE all your work until the experiment is graded.
- The assignment must be original, INDIVIDUAL work. Downloaded or modified source codes will be considered as cheating. Also the students who share their works will be punished in the same way.



- You can ask your question via course's news group.