IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

MESSENGER USING SOCKET PROGRAMMING IN JAVA

Vasuki S,Anjuka Bharathi T, Gobinath B, Kalaivani M, Dhanush P

Department of Computer science engineering, SNS COLLEGE OF TECHNOLOGY, Sathyroad, SNS kalvi nagar, Vazhiyampalayam, Coimbatore, Tamil Nadu, 641035.

ABSTRACT:

The world is emerging with technologies that help in communicating with people they like people they are unknown people they are not near them. so, we have built a Client Server Messenger to provide an advanced communication platform so that multiple people can communicate with each other at the same time. All you need is just an internet connection. You can connect to your group of friends wherever internet connectivity is there.

INTRODUCTION:

Chatting Application is a desktop-based application.

This client server chat application is based on java swing and used socket package. it's simple and easy and require only core java knowledge. I have taken this program from internet and modified a little bit to make it simpler and more elegant

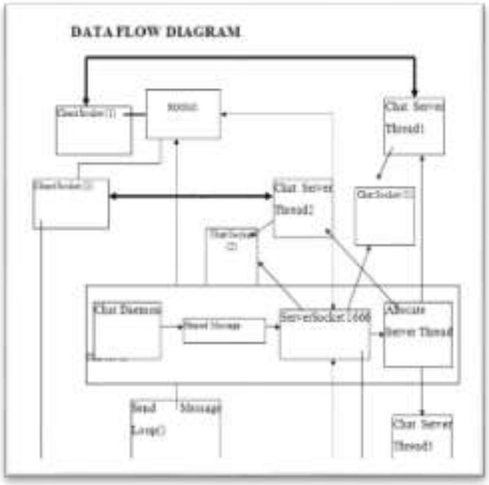
This application/program is a good example of using java.io, java.net package to create a chat application. A beginner of java language, who is familiar with this package can able, be beneficiate.

Chatting is a method of using technology to bring people and ideas "together" despite of the geographical barriers. The technology has been available for years but the acceptance it was quit recent. Our project is an example of a multiple client chat server.

It is made up of 2 applications the client application, which runs on the user's Pc and server application, which runs on any Pc on the network. To start chatting client should get connected to server. We will focus on TCP and UDP socket connections which are a fundamental part of socket programming.

Keywords: sockets, client-server, Java network programming-socket functions, Multicasting etc.

Client Server Messenger Data Flow Diagram



MAIN OBJECTIVE:

The aim of this project is to express how we can implement a simple chat application between a server and a client? The application is a desktop-based application and is implemented using Swing and awt. The project is developed in Java SE language executed on a single stand-alone java across a network using loop back address concept.

Application consists of two programs:

Server

Client

Server:

The server module of the application waits for the client to connect to it. Then if connection is granted a client interacts communicates and connects to the server, it can mutually communicate with the server. The duty of the server is to let clients exchange the messages.

Client:

The client module is the one that utilizer sends requests to the server. Utilizer utilizes the client as the means to connect to the server. Once he establishes the connection, he can communicate to the connected server.

Server Socket

This class is used by the server to declare a Server Socket object which the server needs to listen to connection requests from clients

Project Scope:

This project can be mainly divided into two modules:

1. **Server**

2. **Client**

This project is mainly depended on client/server model. The client requests the server and server responses by granting the client's request. The proposed system should provide both of the above features along with the followed ones:

Server

A server is a computer program that provides services to other computer programs (and their users) in the same or other computers. The computer that a server program runs in is also frequently referred to as a server. That machine may be a dedicated server or used for other purposes as well. Example Server, Database, Dedicated, Fileserver, Proxy Server, Web Server. The server is always waiting for client's requests. The client come and go down but the server remains the same.

A server application normally listens to a specific port waiting for connection requests from a client. When a connection request arrives, the client and the server establish a dedicated connection over which they can communicate. During the connection process, the client is assigned a local port number, and binds a socket to it. The client talks to the server by writing to the socket and gets information from the server by reading from it. Similarly, the server gets a new local port number (it needs a new port number so that it can continue to listen for connection requests on the original port). The server also binds a socket to its local port and communicates with the client by reading from and writing to it. The client and the server must agree on a protocol that is, they must agree on the language of the information transferred back and forth through the socket. Normally, a server runs on a specific computer and has a socket that is bound to a specific port number. The server just waits, listening to the socket for a client to make a connection request.

Client

On the client site the client knows the hostname of the machine on which the server is running and the port number on which the server is listening.

To make a connection request, the client tries to rendezvous with the server on the server's machine and port. The client also needs to identify itself to the server so it binds to a local port number that it will use during this connection. This is usually assigned by the system.

The model used for this project is the single server –

Single client models.

The following specifications must be implemented:

1. The server and client are two separate programs.

REQUIREMENTS:

External Interface Requirements

User Interface

The user interface required to be developed for the system should be user friendly and attractive.

There are two sets of Java APIs for graphics programming:

AWT (Abstract Windowing Toolkit) and Swing.

- AWT API was introduced in JDK 1.0. Most of the AWT components have become obsolete and should be replaced by newer Swing components.
- Swing API, a much more comprehensive set of graphics libraries that enhances the AWT, was introduced as part of Java Foundation Classes (JFC) after the release of JDK 1.1. JFC consists of Swing, Java2D, Accessibility, Internationalization, and Pluggable Look-and-Feel Support APIs. JFC was an add-on to JDK 1.1 but has been integrated into core Java since JDK 1.2.

Software Interfaces:

Programming Language - Java

AND SOCKET PROGRAMMING

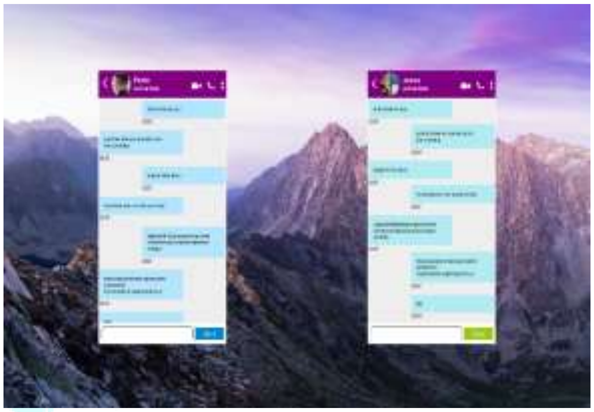
Operational Concepts and Scenarios:

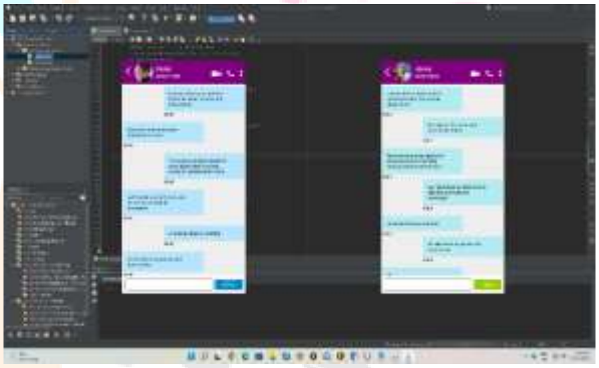
Operation of the application based on the inputs given by the user:

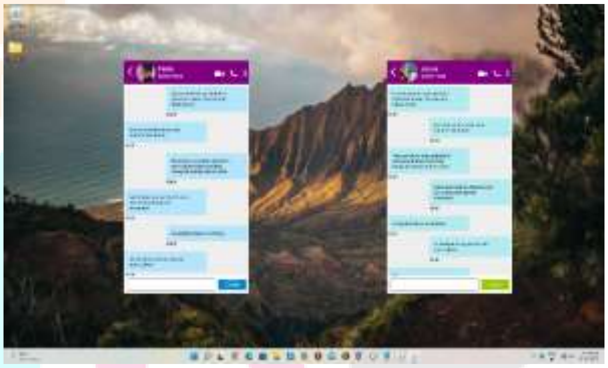
When the run button is clicked then the chat form is initialized with a connection between the host and the client machine.

Note: server must be started at first before a client start. Contains a rich textbox which send messages from one user to another Contains a textbox for messages to be written that is sent across the network. Contains a Send button When the sent button is clicked, in the background, the text in the textbox is encoded and sent as a packet over the network to the client machine. Here this message is decoded and is shown in the rich textbox.

IMAGE:







Future work

There is always a room for improvements in any software package, however good and efficient it may be done. But the most important thing should be flexible to accept further modification. Right now we are just dealing with text communication. In future this software may be extended to include features such as:

- Files transfer: this will enable the user to send files of different formats to others via the chat· application.
- Voice chat: this will enhance the application to a higher level where communication will be· possible via voice calling as in telephone.
- Video chat: this will further enhance the feature of calling into video communication.

CONCLUSION:

The chat application provides a better and flexible system for chatting. It is developed with recent advanced technologies is a way to provide a reliable system. Main advantages of the system are instant messaging, real-world connectivity, adding security, group chat, etc. This application can find better need in the market for most of the organizations aim at having private applications for them. Additional features will also be included in the system based on the public need which includes conference call, video chat. Location share, etc. based on the need

REFERENCES:

1. “Multiusers communicating system in client/server mode based on www”,by Shen Ruiming , Computer Engineering , 1998(2):53-58 2. ”The design of instant communicating system in server side”, by Huan Kai,Tao Hongcai,Journal of Chengdu University of Information Technology,2006(4):535-538

3. “The data visual development and application based on three layers structure”, by Li Zuohong,Luo Zhijia, Microcomputer Information,2006(21):182-185

4.” Implementation of enterprise instant communicating system based on application layer with java programming”, by Lin Jianbing, Zou Jinan, vol.27, no. 6, pp. 56-61,July,2015.

5. The Java Tutorials, “Lesson: All about Sockets”.

6. “Predicting Defects for Eclipse,” by T. Zimmermann, R. Premraj, and A. Zeller, in Proceedings of the Third International Workshop on Predictor Models in Software Engineering, Washington, DC, USA, 2007, p.

7. “Private information retrieval,” by B. Chor, E. Kushilevitz, O. Goldreich, and M. SudanJ. ACM, vol. 45, no. 6, pp. 965–981, 1998