INTERNATIONAL JOURNAL OF PURE AND APPLIED RESEARCH IN ENGINEERING AND TECHNOLOGY

A PATH FOR HORIZING YOUR INNOVATIVE WORK

AN IMPLEMENTATION OF CLIENT AND SERVER DATA TRANSFER USING SOCKET

SHWETA KOLTE, KAJAL THORAT, PRATIKSHA DESHMUKH, PROF. Z. I. KHAN

Dept. of Computer Science& Engg., DRGITR, Amravati.

Accepted Date: 15/03/2016; Published Date: 01/05/2016

Abstract*:* In this project, the client server architecture is used to develop a data transfer & chat application. Firstly a data transfer & chat application is created for both Client and Server which is based on Transmission Control Protocol where TCP is connection oriented protocol and is a reliable connection protocol. A client-server data transfer application consists of a Client and a Server and there exists a two way communication between them. Client Manager is used to maintain the clients list which the sender and receiver at both sides use to interact with each other.

Keywords: Server Data Transfer, Socket

Corresponding Author: MISS. SHWETA KOLTE

Access Online On:

www.ijpret.com

How to Cite This Article:

Shweta Kolte, IJPRET, 2016; Volume 4(9): 1531-1538

PAPER-QR CODE

INTRODUCTION

In general, the server process will start on some computer system; in fact, the server should be executed before the client. Server usually initializes itself, and then goes to wait state or sleep state where it will wait for a client request. After that, a client process can start on either the same machine or on some other machine. Whenever the client wants some service from the server, it will send a request to the server and the server will accept the request and process it. After the server has finished providing its service to the client, the server will again go back to sleep, that is, waiting for the next client request to arrive. This process is repeated as long as the server processes is running. Whenever such request comes, the server can immediately serve the client and again go back to the waiting state for the next request to arrive. Client server model is the standard model which has been accepted by many for developing network applications. In this model, there is a notion of client and notion of server. As the name implies, a server is a process (or a computer in which the process is running) that is offering some services to other entities which are called clients. A client on the other hand is process (which is running) on the same computer or other computer that is requesting the services provided by the server. A data transfer is basically a combination of two sides transfer:

1. Server Side

2 .Client Side

Server application runs on the server computer and client application runs on the client computer In this data transfer a client can send data to anyone who is connected to the server.Java application programming interface (API) provides the classes for creating sockets to facilitate program communications over the network. Sockets are the endpoints of logical connections between two hosts and can be used to send and receive data. Java treats socket communications much as it treat input and output operations thus programs can read from or write to sockets as easily as they can read from or write to files. To establish a server connection, a server socket needs to be created and attached to a port, which is where the server listens for connections. The port recognizes the Transmission Control Protocol service on the socket. For instance, the email server runs on port 25, and the web server usually runs on port 80.

1.1 Server Execution: At server the side, a thread is created which receives numerous clients requests. It also contains a list in which Client's name and IP addresses are stored. After that, it broadcast the list to all the users who are currently in chat room and when a client logs out then

server deletes that particular client from the list, update the list and then broadcast the list to all available clients.

1.2 Client Execution: A client firstly must have to register itself by sending username to the server and should have to start the thread so that system can get the list of all available clients. Then any of two registered clients can communicate with each other.[2]

2. LITERATURE REVIEW

The socket programming concept has benefited many areas of computer networks today. The socket application developed in this work is based on the concept of distributed computing. This is relevant as to that it allows for resources to be distributed among several node in the network [1]. Socket programming helps to implement the bottom level of network communication, using Application Programming Interface (API). A similar application is built by other researchers as presented in [2], [3] and [4]. This application is also using a multicast datagram socket class which is useful for sending and receiving IP multicast packets. Furthermore, a multicast socket is also a UDP datagram socket which is capable for joining "group". This additional functionality is an added advantage for using multicast socket [5].

In Computer Engineering, client-server is a software architecture model consisting of two parts, client systems and server systems both communicating over a computer network or on the same computer. A client-server application is a distributed system consisting of both client and server software. The client process always initiates a connection to the server, while the server process always waits for requests from any client. When both the client process and server process are running on the same computer, this is called a single seat setup. The client-server relationship describes the relation between the clients and how it makes a service request from the server and how the server can accept these requests, process them and return the requested information to the client.

2.1 Client-Server and Other Models

The client-server model according to Hou Meng-bo (May 2008) was originally developed to allow more users to share access to database applications. Compared to the mainframe approach, client-server offers improved scalability because connections can be made as needed rather than being fixed.Client-server is just one approach to managing network applications. The primary alternative, peer-to-peer networking, models all devices as having equivalent capability rather than specialized client or server roles. Compared to client-server,a peer to peer network offers

some advantages such as more flexibility in growing the system to handle large number of clients. Client-server networks generally offer advantages in keeping data secure.

2.2 Client-Server Communication

According to Fisher Price Et al (2011) at a basic level, network-based systems consist of a server, client and a media for communication. A computer running a program that makes a request for services is called client machine. A computer running a program that offers requested services from one or more clients is called server machine. The media for communication can be wired or wireless network.Generally, programs running on client machines make requests to a program (often called as server program) running on a server machine.[4]

TCP is a connection-oriented protocol that provides a reliable flow of data between two computers. Example applications that use such services are HTTP, FTP, and Telnet. UDP is a protocol that sends independent packets of data, called datagram, from one computer to another with no guarantees about arrival and sequencing. [2]

3. PROPOSEDOBJECTIVE

The aim of this project is to develop a reliable and secure network programming (Client-Server chat model) which can perform a multithreaded server client data tranfer based on Java socket programming using Transport Control Protocol (TCP). As security is the key factor while communicating over a network. became the choice for the implementation of this application based on its scalability and flexibility, high performance, high availability, strong data protection, web and data warehouse strengths, management ease, lowest total cost of ownership and open source freedom.

4. IMPLEMENTATION

1. Software Requirement

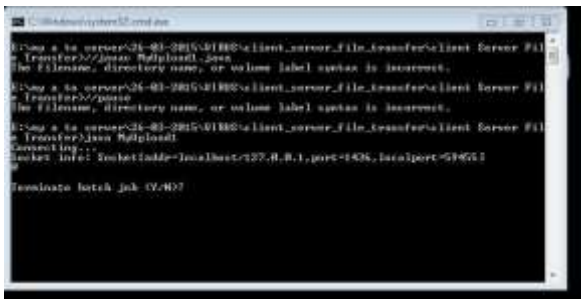
Java can be used easily to develop network applications. It comes with a very powerful class library for networking as part of java package. The java.net package will have to be included at the beginning of the program for network input and output methods operations. By using TCP, the server runs first and then waits for client's requests and sever has ability to either accept or deny the client request. A server has a socket which is bound to a specific port. Client programs send connection request to the server program and if everything is well then server accepts the connection. After accepting the client request the server new socket bound to a different port, it

requires a new socket bound because the original socket has to listen to other connection requests.

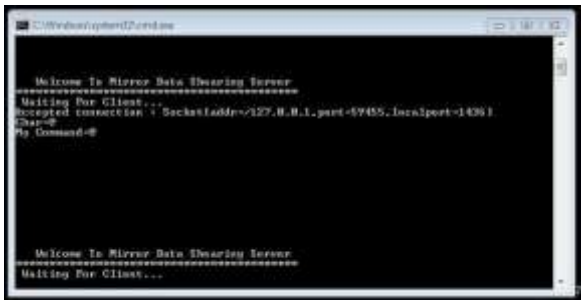
2. Java became the chosen programming language for the client-server data transfer because Java application programming interfaces (API) provides the classes for creating sockets to facilitate program communications over the network. Sockets are the endpoints of logical connections between two hosts and can be used to send and receive data. Java treats socket communications much as it treat input and output operations; thus programs can read from or write to sockets as easily as they can read from or write to files. [1]

3. Once a datagram packet arrives at the host at the specified local port number at which the socket is opened, the receiver program will proceed further – extract the bytes stored in the datagram packet and prints the contents as a String. The local port number at which the receiver should open its datagram socket is passed as an input command- line parameter and this port number should also be known to the sender so that the message can be sent to the same port number. The datagram sender program creates a DatagramPacket object and sets its destination IP address to the IP address of the remote host, the port number at which the message is expected to receive and the actual message. The sender program has nothing much to do after sending the message and hence the socket is closed. the socket at the receiver side is also closed after receiving and printing the message.[3]

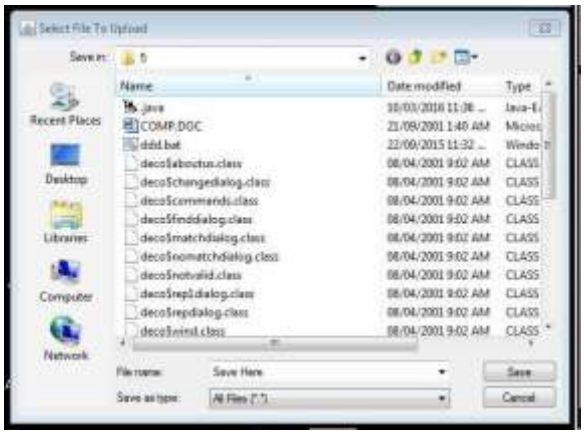
Screenshot uploading file



Screenshot server side



Screenshot waiting for client



Screenshot brawsing file

5. ADVANTAGES

- Centralization: Servers help in administering the whole set-up.
- Proper Management: management of files becomes easy.
- Back-up and Recovery possible:

- Upgradation and Scalability:
- Accessibility: From various platforms in the network, server can be accessed remotely.
- Security

6. LIMITATIONS

- Required server start first.
- As the server receives as many request from clients so there is chance that server can become congested and overloaded.[1]
- In case of server fails then the users also suffers. [2]
- Any unauthorized client can hack the client account and can change the data.

7. FUTURE SCOPE

The project shall consider among other things the following issues:

1. To provide a better understanding of how network programming in java works.

2. Develop a reliable network communication for a Client-Server data transfer & chat application.

3. Analyses of network programming in java (Multithreaded Client-Server Chat applications) for better understanding of the solutions.

4. Conduct an experimental result in order to establish the parameter of the problem. In conclusion, suggest ways the problems can be eliminated and recommends how the problems can be prevented.

8. CONCLUSION

The project entitled "IMPLEMENTATION CLIENT-SERVER-DATA TRANSER USING SOCKET ALGORITHM" is developed using JAVA as front end and No Need Of Any Backend computerize the process of check in and checkout system. This project covers all the basic features required for transfer and receive data.

9. REFERENCES

1. "A design and implementation of active network socket programming." Computer Communications and Networks. 78 - 38 , 41 – 41

2. Malhotra, A., Sharma, V., Gandhi, P., Purohit, N. "UDP based chat application." Computer Engineering and Technology (ICCET), 2010 2nd International Conference vol.6: pp.V6-374-V6-377.

3. Kaur, D., Dhanda, P., Mirchandani, M. 2000. "Development of a real time chat application on intelligent network based on fuzzy logic," Circuits and Systems, 2000." Proceedings of the 43rd IEEE Midwest Symposium vol.3, pp.1376- 1380.

4. Shirali-Shahreza, M.H., Shirali-Shahreza, M. 2007. "Text Steganography in chat." Internet. ICI. 3rd IEEE/IFIP International Conference in Central Asia pp.1-5, 26-28.

5. Wang, Ho Leung, Tsuhan Chen, Hendriks, F., Xiping Wang, and Zon-Yin Shae. "eMeeting: a multimedia application for interactive meeting and seminar." Global Telecommunications Conference, GLOBECOM. IEEE 3, pp. 2994- 2998, 17-21.

6. Multithreaded Client/Server Application [http://www.ase.md/~aursu/ClientServerThreads.html]

7. Client-server Chat [http://jerome.jouvie.free.fr/Java/Network/Tutorials/Tutorial2.php]

8. D. E. Comer, "Computer Networks and Internets," 5th Edition, Prentice Hall, 2008.