Game Server   
with   
Socket Programming

Queens College

Professor Md Mahbubur Rahman

Group Members

Ashikur Rahman | James Park | Asif Talukder

**-> Introduction**

Socket programming is a method of sending data across a network between two computers. A connection-oriented protocol or a connectionless protocol can be used to establish a connection. TCP/IP, a connection-oriented protocol, will be used in our case.

Computers must establish a connection for connection-oriented protocols before transferring data. The sole connectionless protocol choice is UDP (User Datagram Protocol). The fundamental difference between the two is that UDP is connectionless, which means there is no session between the client and the server, and TCP is connection-oriented, that means that communication must first be established between the client and the server.

The Client/Server architecture will be used to further demonstrate sockets. Sending to and retrieving from the socket connection is how the client and server communicate. A socket is a type of communication endpoint that interconnects devices on a network. It has a port number that the TCP/IP layer can use to identify the receiving application. A port number and an IP address are generally included in an endpoint.

The Transmission Control System (TCP) is a widely used data transmission protocol that enables client/server end points on a network.

Two categories of Sockets:

* A server socket - It awaits a request from a client.
* A client socket - It establishes communication between client and server.

The client has to know two things about the server: The server’s IP address and The port number.

**-> The project idea**

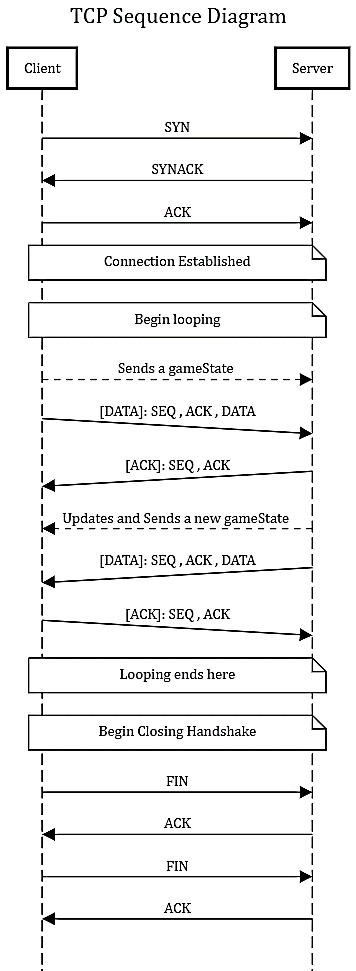
Our project is about making a client-server architecture that serves a purpose of communicating states between them. The client plays a console game and communicates the state with the server. The server saves the state and acts as a bot opponent to play against the client. We opted to console interface to make our project focused on socket programming. So we did not use any GUI.

-> tools used --- development envirnment and IDE

Our development environment:

* OS: Windows 10 and partially MAC
* SDK : Java version "17.0.1" 2021-10-19 LTS
* Java(TM) SE Runtime Environment (build 17.0.1+12-LTS-39)
* IDE : Visual Studio Code & partially JetBrains Intelli-J

**-> Diagram for client server connection**



**-> Libraries Used:**

-> java.net.ServerSocket

A server runs on a specific computer on the network and has a socket that's bound to a specific port number. We use the same computer as the client, and start the server on port 4040

        final int PORT = 4040;

        ServerSocket serverSocket = new ServerSocket(PORT);

The server just waits, listening to the socket for a client to make a connection request. Next we accept incoming connections:

            Socket clientSocket = serverSocket.accept();

When the server code encounters the accept method, it blocks until a client makes a connection request to it.

-> java.net.Socket

The client must know the hostname or IP 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:

        final String HOST = "127.0.0.1";

       final int PORT = 4040;

Socket socket = new Socket(HOST, PORT);

-> Other Libraries:

- java.io.PrintWriter : To write on socket output stream

- java.util.Random : To generate random numbers that helps mimic a bot opponent

- java.util.Scanner : To get user inputs

- java.io.IOException : To handle exceptions

**-> Contributions of each group member in percentage**

-> Server (33.3%) :

James worked on server and handling incoming stream; and threading new clients.

-> Client (33.3%) :

Asif worked on client and handling game instance that sends output streams to the socket.

-> StateManager + Game (33.3%) :

Ashikur worked on the console Game and the state manager.

**-> References**

We would like to honor some walkthrough tutorials on Java socket libraries that helped us understand and plan this project.

* <https://youtu.be/vCDrGJWqR8w>
* <https://www.tutorialspoint.com/java/java_networking.htm>
* <https://www.geeksforgeeks.org/socket-programming-in-java/>
* <https://www.section.io/engineering-education/socket-programming-in-java/>