

## Exercise 01

**Name:** Jose Juan Sandoval

**Link to Project:** <https://github.com/Juanchiselo/CS380/tree/master/Exercise%2001>

### Java Code

#### EchoServer.java

```
package Exercise01;

import java.io.IOException;
import java.net.ServerSocket;

public final class EchoServer
{
    public static void main(String[] args) throws IOException
    {
        int portNumber = 22222;
        boolean listening = true;

        try (ServerSocket serverSocket = new ServerSocket(portNumber))
        {
            while (listening)
            {
                // Creates and starts a new thread for the newly connected client.
                new ServerThread(serverSocket.accept()).start();
            }
        }
        catch (IOException e)
        {
            System.err.println("ERROR: Could not listen on port " + portNumber + ".");
            System.exit(-1);
        }
    }
}
```

#### ServerThread.java

```
package Exercise01;

import java.io.*;
import java.net.Socket;

public class ServerThread extends Thread
{
    private Socket socket = null;

    public ServerThread(Socket socket)
    {
        super("ServerThread for "
            + socket.getInetAddress().getHostAddress());
        this.socket = socket;
    }

    /**
     * The overridden run() function belonging to the Thread class.
     * This is what handles the communication between the server and the client.
     */
    public void run()
    {
        try
        {
            String clientAddress = socket.getInetAddress().getHostAddress();

            // Objects needed for receiving and reading the client's messages.
```

```

String message;
InputStream inputStream = socket.getInputStream();
InputStreamReader inputStreamReader = new InputStreamReader(inputStream, "UTF-8");
BufferedReader in = new BufferedReader(inputStreamReader);

// Objects needed for sending messages to the client.
OutputStream outputStream = socket.getOutputStream();
PrintStream out = new PrintStream(outputStream, true, "UTF-8");

System.out.println("Client connected: " + clientAddress);

// Welcomes the client.
// NOTE: This is important because the client is waiting to receive
// a message in order to be able to send a message to the server.
out.println("Hi " + clientAddress + ", thanks for connecting!"
    + " If you would like to disconnect just type \"EXIT\".");

// The main loop of execution.
// It only executes when the client has sent a message.
while((message = in.readLine()) != null)
{
    if(message.toUpperCase().equals("EXIT"))
        break;

    // Echoes the message back to the client.
    out.println(message);
}
socket.close();
System.out.println("Client disconnected: " + clientAddress);
}
catch (IOException e)
{
    System.err.println("ERROR: Connection lost with client "
        + socket.getInetAddress().getHostAddress());
}
}
}

```

## EchoClient.java

```

package Exercise01;

import java.io.*;
import java.net.Socket;
import java.net.UnknownHostException;
import java.util.Scanner;

public final class EchoClient
{
    public static void main(String[] args) throws IOException
    {
        String hostName = "localhost";
        int portNumber = 22222;

        try (Socket socket = new Socket(hostName, portNumber))
        {
            // Objects needed for receiving and reading the server's messages.
            String message;
            InputStream inputStream = socket.getInputStream();
            InputStreamReader inputStreamReader = new InputStreamReader(inputStream, "UTF-8");
            BufferedReader in = new BufferedReader(inputStreamReader);

            // Objects needed for sending messages to the server.
            Scanner scanner = new Scanner(System.in);
            OutputStream outputStream = socket.getOutputStream();
            PrintStream out = new PrintStream(outputStream, false, "UTF-8");

            // The main loop of execution.
            // It only executes when the server has sent a message.

```

```
while((message = in.readLine()) != null)
{
    System.out.println("Server> " + message);
    System.out.print("Client> ");
    message = scanner.nextLine();

    // Sends the message to the server.
    out.println(message);

    if(message.toUpperCase().equals("EXIT"))
        break;
}
socket.close();
}
catch (UnknownHostException e)
{
    System.err.println("ERROR: Unknown host " + hostName + ".");
}
catch (Exception e)
{
    System.err.println("ERROR: Could not connect to " + hostName + ".");
}
finally
{
    System.exit(1);
}
}
```