Echo Client/Server

1. Introduction

This tutorial introduces a simple echo client/server and how to use it. The client sends a message to the server, and then the server replies with the same message to the client.

2. The Server

The server starts by listening for a TCP connection on the local PC (loop back is the same as localhost) and on a specified port.

```
// set port number
int port = 8080;
// create a local listener
TcpListener listener = new TcpListener(IPAddress.Loopback, port);
// start listening
listener.Start();
```

The server then creates a TcpClient so that it can connect to a new client contacting the server. It then gets the stream from the client so that it can read and write to/from the client.

```
// accept a client
TcpClient client = listener.AcceptTcpClient();
// get the stream
NetworkStream stream = client.GetStream();
// create a writer and reader
StreamWriter writer = new StreamWriter(stream, Encoding.ASCII) { AutoFlush = true };
StreamReader reader = new StreamReader(stream, Encoding.ASCII);
```

The server then loops waiting for a message from the client. When a message arrives from the client, it sends the message back to the client and displays the message on its console.

```
// read from the client and echo back
while (true)
{
    string inputLine = "";
    while (inputLine != null)
    {
        // read from the client
        inputLine = reader.ReadLine();
        // echo back to the client
        writer.WriteLine("Echoing string: " + inputLine);
        // print to the console
        Console.WriteLine("Echoing string: " + inputLine);
    }
    // close the connection
    Console.WriteLine("Server saw disconnect from client.");
}
```

3. The Client

The client starts by creating for a TCP connection on the local PC (localhost) and on a specified port. It then gets the stream from the connection so that it can read and write to/from the server.

```
// set port number
int port = 8080;
// create a client
TcpClient client = new TcpClient("localhost", port);
// get the stream
NetworkStream stream = client.GetStream();
// create a writer and reader
StreamReader reader = new StreamReader(stream);
StreamWriter writer = new StreamWriter(stream) { AutoFlush = true };
```

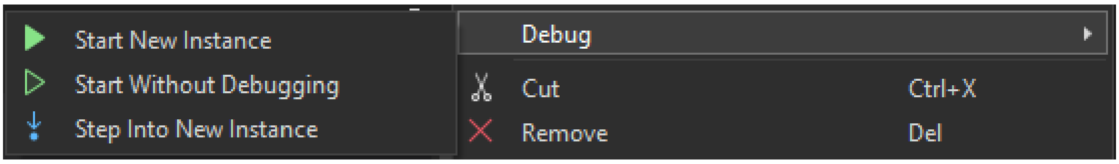
The client then loops waiting for a message from the user and thrn sends that message to the server. When the message arrives back from the server, it displays the message on its console.

```
while (true)
{
    // read from the console
    Console.Write("Enter text to send: ");
    string lineToSend = Console.ReadLine();
    // send to the server
    Console.WriteLine("Sending to server: " + lineToSend);
    writer.WriteLine(lineToSend);
    // read from the server and print to the console
    string lineReceived = reader.ReadLine();
    Console.WriteLine("Received from server: " + lineReceived);
}
```

4. Exercises

Please attempt these exercises, but if you get stuck or you are confused then ask for help during the scheduled lab times.

1. Start the Server. You do this by starting it without debugging, otherwise the debugger will not allow you to start the client. It is suggested to right-click on the Server project and select Debug -> Start Without Debugging.



2. Start the Client in the same manner as you did for the Server. You can now send a message to/from the Server.

3. Consider how you can use this code for sending a player’s initials and score from a client to a server. This could be used to store highscores on the server.