Fall-2020

NETWORK PROGRAMMING

INSTRUCTOR MISBAH ANWER

STUDENT: BUSHRA (63759)

Contents

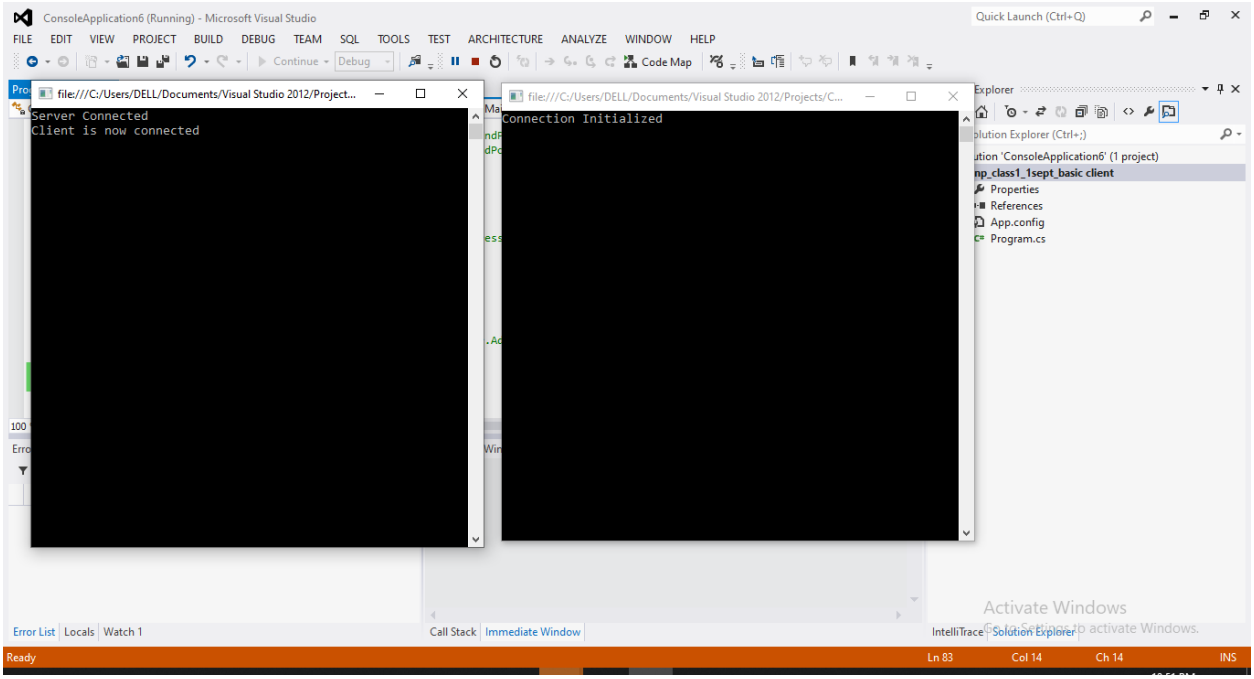
2nd Lecture – Establishing Connection Client/Server

Server Code

```
        IPAddress ip = IPAddress.Loopback;

          IPEndPoint ep = new IPEndPoint(ip, 2000);
    Socket sk = new Socket(ip.AddressFamily, SocketType.Stream, ProtocolType.Tcp);

          sk.Bind(ep);
          Console.WriteLine("Server Connected ");
          sk.Listen(1);
          Socket cl = sk.Accept();
          Console.WriteLine("Client is now connected");
```

Client Code

```
IPAddress ip = IPAddress.Loopback;
          IPEndPoint ep = new IPEndPoint(ip, 2000);
        Socket sk = new Socket(ip.AddressFamily, SocketType.Stream, ProtocolType.Tcp);

          sk.Connect(ep);
          Console.WriteLine("Connection Initialized");
          Console.ReadLine();
```



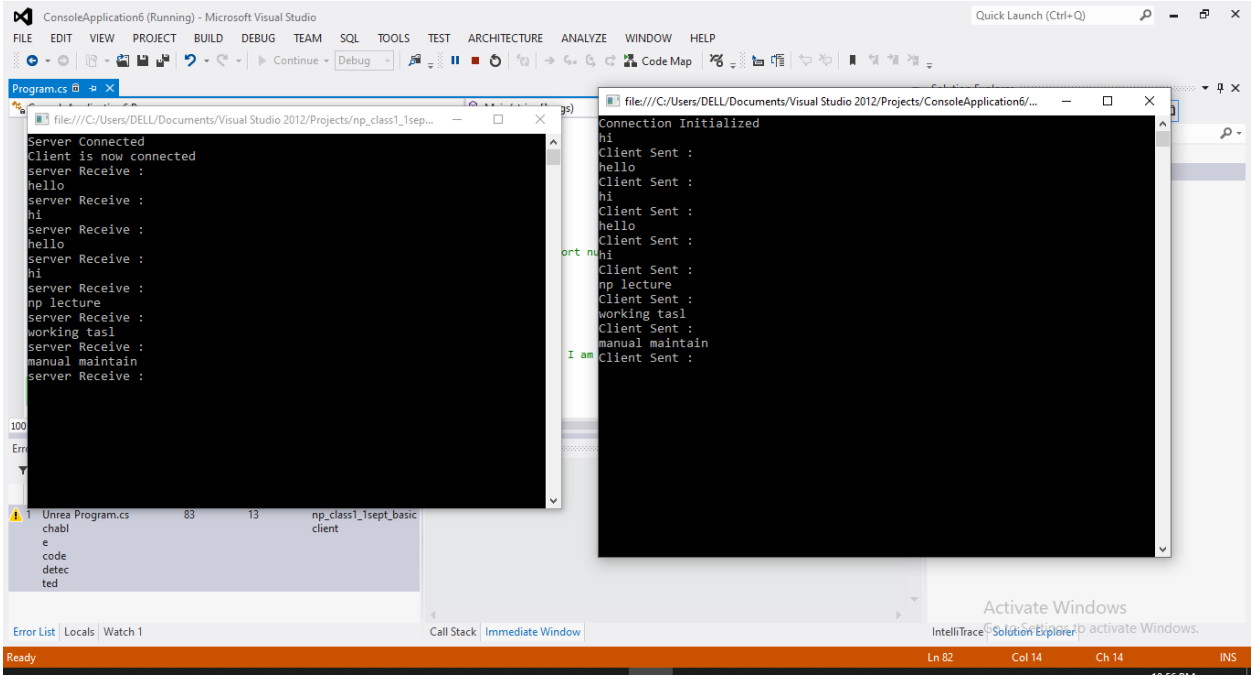
One-Way Communication

Client Code:

```
IPAddress ip = IPAddress.Loopback;
            IPEndPoint ep = new IPEndPoint(ip, 2000);
          Socket sk = new Socket(ip.AddressFamily, SocketType.Stream, ProtocolType.Tcp);

            sk.Connect(ep);
            Console.WriteLine("Connection Initialized");
            Console.ReadLine();


            while (true)
            {
                byte[] ar = new byte[50];

                Console.WriteLine("Client Sent : ");
                string str = Console.ReadLine();
                sk.Send(Encoding.ASCII.GetBytes(str));

           }

            Console.ReadKey();
```

Server Code

```
          IPAddress ip = IPAddress.Loopback;

            IPEndPoint ep = new IPEndPoint(ip, 2000);
     Socket sk = new Socket(ip.AddressFamily, SocketType.Stream, ProtocolType.Tcp);

            sk.Bind(ep);
            Console.WriteLine("Server Connected ");
            sk.Listen(1);
            Socket cl = sk.Accept();
            Console.WriteLine("Client is now connected");


            while (true)
            {
                byte[] ar = new byte[50];

                Console.WriteLine("server Receive :");
              cl.Receive(ar);
             Console.WriteLine(Encoding.ASCII.GetString(ar));


        }
            Console.ReadKey();
```


ConsoleApplication6 (Running) - Microsoft Visual Studio
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM SQL TOOLS TEST ARCHITECTURE ANALYZE WINDOW HELP
Program.cs
file:///C:/Users/DELL/Documents/Visual Studio 2012/Projects/np_class1_1sep...
Server Connected
Client is now connected
server Receive :
hello
server Receive :
hi
server Receive :
hello
server Receive :
hi
server Receive :
np lecture
server Receive :
working tasl
server Receive :
manual maintain
server Receive :
file:///C:/Users/DELL/Documents/Visual Studio 2012/Projects/ConsoleApplication6/...
Connection Initialized
hi
Client Sent :
hello
Client Sent :
hi
Client Sent :
hello
Client Sent :
hi
Client Sent :
np lecture
Client Sent :
working tasl
Client Sent :
manual maintain
Client Sent :
Error List Locals Watch 1
Call Stack Immediate Window
IntelliTrace Solution Explorer
Ready
Ln 82 Col 14 Ch 14 INS

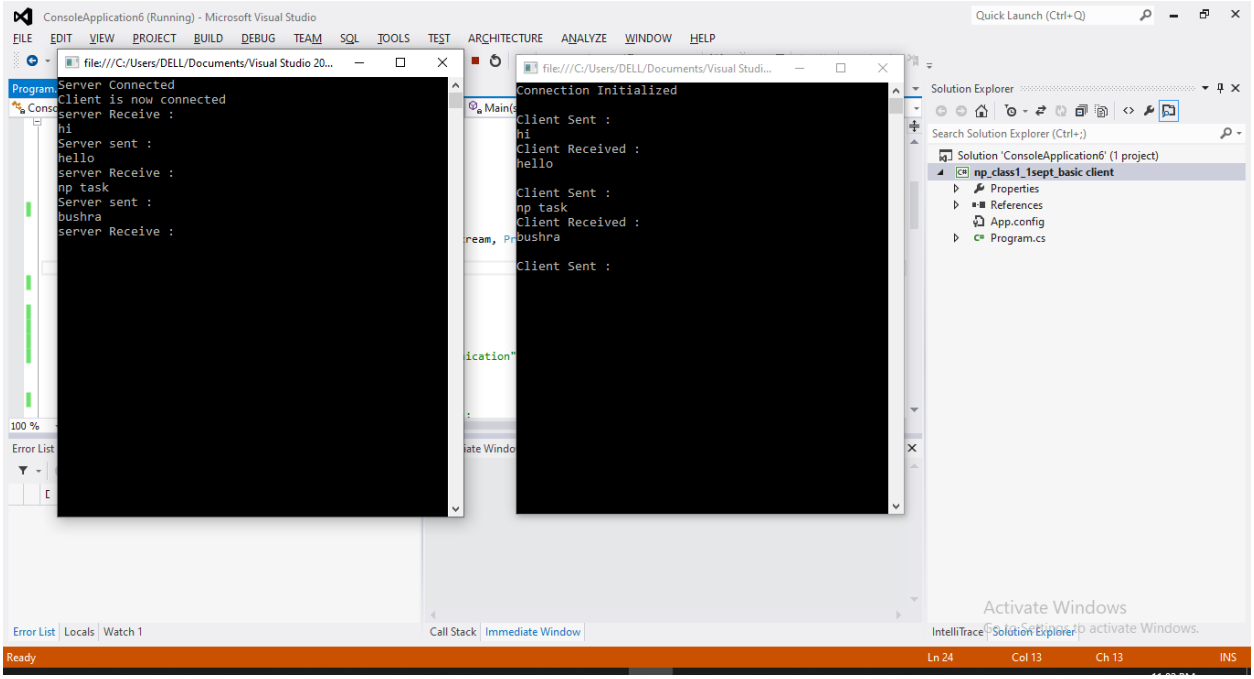
Two-Way Communication

Server Code

```
IPAddress ip = IPAddress.Loopback;

        IPEndPoint ep = new IPEndPoint(ip, 2000);
        Socket sk = new Socket(ip.AddressFamily, SocketType.Stream, ProtocolType.Tcp);

            sk.Bind(ep);
            Console.WriteLine("Server Connected ");
            sk.Listen(1);
            Socket cl = sk.Accept();
            Console.WriteLine("Client is now connected");

            while (true)
            {
                byte[] ar = new byte[50];

                Console.WriteLine("server Receive :");
              cl.Receive(ar);
             Console.WriteLine(Encoding.ASCII.GetString(ar));

                Console.WriteLine("Server sent : ");
              string str = Console.ReadLine();
             cl.Send(Encoding.ASCII.GetBytes(str));


            }
            Console.ReadKey();
```

Client Code

```
  IPAddress ip = IPAddress.Loopback;
            IPEndPoint ep = new IPEndPoint(ip, 2000);
   Socket sk = new Socket(ip.AddressFamily, SocketType.Stream, ProtocolType.Tcp);

            sk.Connect(ep);
            Console.WriteLine("Connection Initialized");
            Console.ReadLine();


            while (true)
            {
                byte[] ar = new byte[50];

                Console.WriteLine("Client Sent : ");
                string str = Console.ReadLine();
                sk.Send(Encoding.ASCII.GetBytes(str));
                Console.WriteLine("Client Received : ");
              sk.Receive(ar);
              Console.WriteLine(Encoding.ASCII.GetString(ar));
           }

            Console.ReadKey();
```


ConsoleApplication6 (Running) - Microsoft Visual Studio
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM SQL TOOLS TEST ARCHITECTURE ANALYZE WINDOW HELP
file:///C:/Users/DELL/Documents/Visual Studio 20...
Server Connected
Client is now connected
server Receive :
hi
Server sent :
hello
server Receive :
np task
Server sent :
bushra
server Receive :
file:///C:/Users/DELL/Documents/Visual Studi...
Connection Initialized
Client Sent :
hi
Client Received :
hello
Client Sent :
np task
Client Received :
bushra
Client Sent :
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'ConsoleApplication6' (1 project)
np_class1_1sept_basic client
Properties
References
App.config
Program.cs
Error List
Locals
Watch 1
Call Stack
Immediate Window
IntelliTrace
Solution Explorer
Ready
Ln 24
Col 13
Ch 13
INS

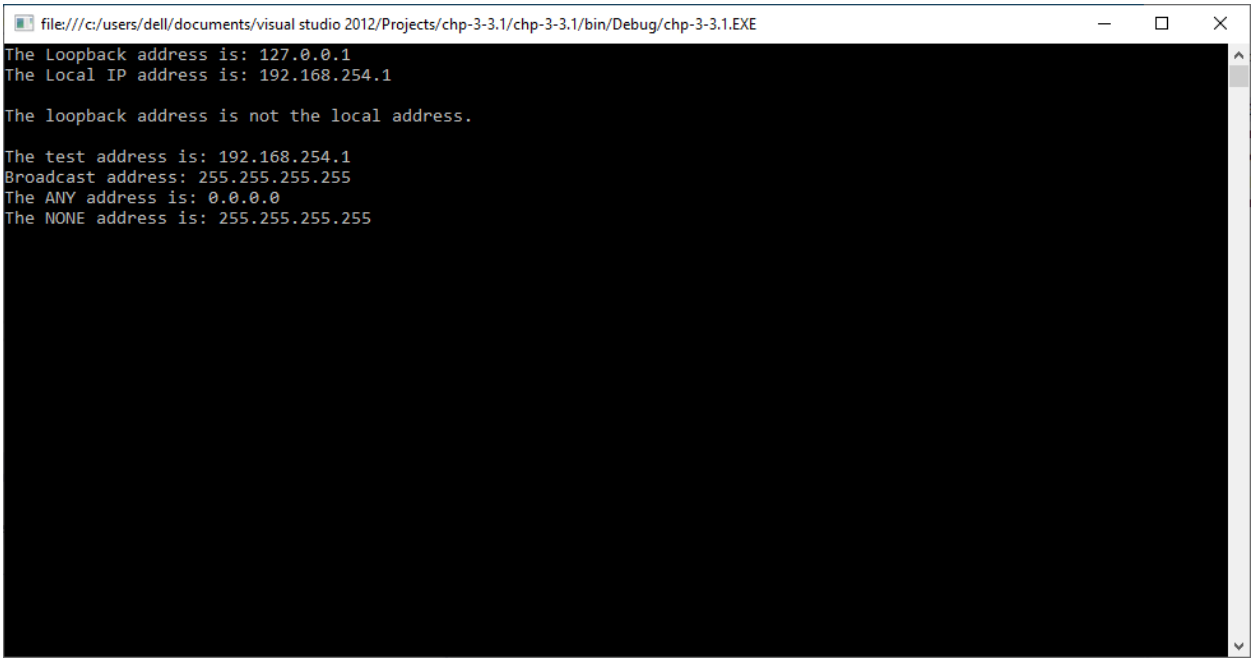
Exercise - 3.1 (Book)

(22nd September)

CODE:

```
IPAddress test1 = IPAddress.Parse("192.168.254.1");
     IPAddress test2 = IPAddress.Loopback;
     IPAddress test3 = IPAddress.Broadcast;
     IPAddress test4 = IPAddress.Any;
     IPAddress test5 = IPAddress.None;
     IPHostEntry ihe = Dns.GetHostByName(Dns.GetHostName());
     IPAddress myself = ihe.AddressList[0];
     if (IPAddress.IsLoopback(test2))
         Console.WriteLine("The Loopback address is: {0}", test2.ToString());
     else
         Console.WriteLine("Error obtaining the loopback address");
     Console.WriteLine("The Local IP address is: {0}\n",
                myself.ToString());
     if (myself == test2)
         Console.WriteLine("The loopback address is the Â same as local
address.\n");
     else

         Console.WriteLine("The loopback address is not the local address.\n");
     Console.WriteLine("The test address is: {0}", test1.ToString());
     Console.WriteLine("Broadcast address: {0}", test3.ToString());
     Console.WriteLine("The ANY address is: {0}", test4.ToString());
     Console.WriteLine("The NONE address is: {0}", test5.ToString());
     Console.ReadLine();
```

OUTPUT:

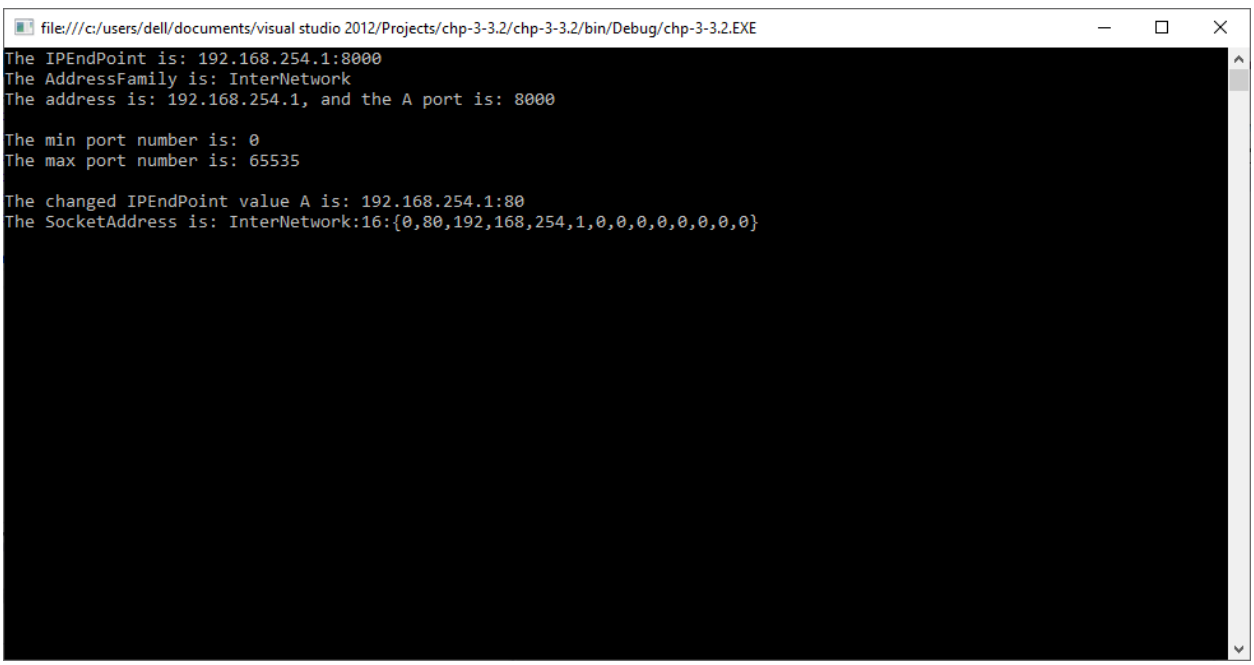


Exercise - 3.2 (Book)

Code:

```
IPAddress test1 = IPAddress.Parse("192.168.254.1");
            IPEndPoint ie = new IPEndPoint(test1, 8000);
            Console.WriteLine("The IPEndPoint is: {0}", ie.ToString());
            Console.WriteLine("The AddressFamily is: {0}", ie.AddressFamily);
            Console.WriteLine("The address is: {0}, and the Â port is: {1}\n",
ie.Address, ie.Port);
            Console.WriteLine("The min port number is: {0}",
            IPEndPoint.MinPort);
            Console.WriteLine("The max port number is: {0}\n", IPEndPoint.MaxPort);
            ie.Port = 80;
            Console.WriteLine("The changed IPEndPoint value Â is: {0}", ie.ToString());
            SocketAddress sa = ie.Serialize();
            Console.WriteLine("The SocketAddress is: {0}", sa.ToString());
            Console.ReadLine();
```

OUTPUT:



Exercise - 3.3 (Book)

CODE:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Net;
using System.Threading.Tasks;
using System.Net.Sockets;

namespace chp_3_3._3
{
    class Program
    {
        static void Main(string[] args)
        {
            //3.3
            IPAddress ia = IPAddress.Parse("127.0.0.1");
            IPEndPoint ie = new IPEndPoint(ia, 8000);
            Socket test = new Socket(AddressFamily.InterNetwork, SocketType.Stream,
ProtocolType.Tcp);
            Console.WriteLine("AddressFamily: {0}", test.AddressFamily);
            Console.WriteLine("SocketType: {0}", test.SocketType);
            Console.WriteLine("ProtocolType: {0}", test.ProtocolType);
            Console.WriteLine("Blocking: {0}", test.Blocking);
            test.Blocking = false;
            Console.WriteLine("new Blocking: {0}", test.Blocking);
            Console.WriteLine("Connected: {0}", test.Connected);
            test.Bind(ie);
            IPEndPoint iep = (IPEndPoint)test.LocalEndPoint;
            Console.WriteLine("Local EndPoint: {0}", iep.ToString());
            Console.ReadLine();
            test.Close();
        }
    }
}
```

OUTPUT:

file:///c:/users/dell/documents/visual studio 2012/Projects/chp-3-3.3/chp-3-3.3/bin/Debug/chp-3-3.3.EXE

```
AddressFamily: InterNetwork
SocketType: Stream
ProtocolType: Tcp
Blocking: True
new Blocking: False
Connected: False
Local EndPoint: 127.0.0.1:8000
```

Exercise - 3.4 (Book)

CODE:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Net;
using System.Net.Sockets;
using System.Threading.Tasks;

namespace chp_3_3._4
{
    class Program
    {
        static void Main(string[] args)
        {
            IPAddress host = IPAddress.Parse("192.168.254.1");
            IPEndPoint hostep = new IPEndPoint(host, 8000);
            Socket sock = new Socket(AddressFamily.InterNetwork, SocketType.Stream,
ProtocolType.Tcp);
            try
            {
                sock.Connect(hostep);
            }
            catch (SocketException e)
            {
                Console.WriteLine("Problem connecting to host");
                Console.WriteLine(e.ToString());

                sock.Close();
                return;
            }
            try
            {
                sock.Send(Encoding.ASCII.GetBytes("testing"));
            }
            catch (SocketException e)
            {
                Console.WriteLine("Problem sending data");
                Console.WriteLine(e.ToString());

                sock.Close();
                return;
            }

            sock.Close();
        }
    }
}
```

OUTPUT:

1a)

Client /server Communication

Using loopback

Client:

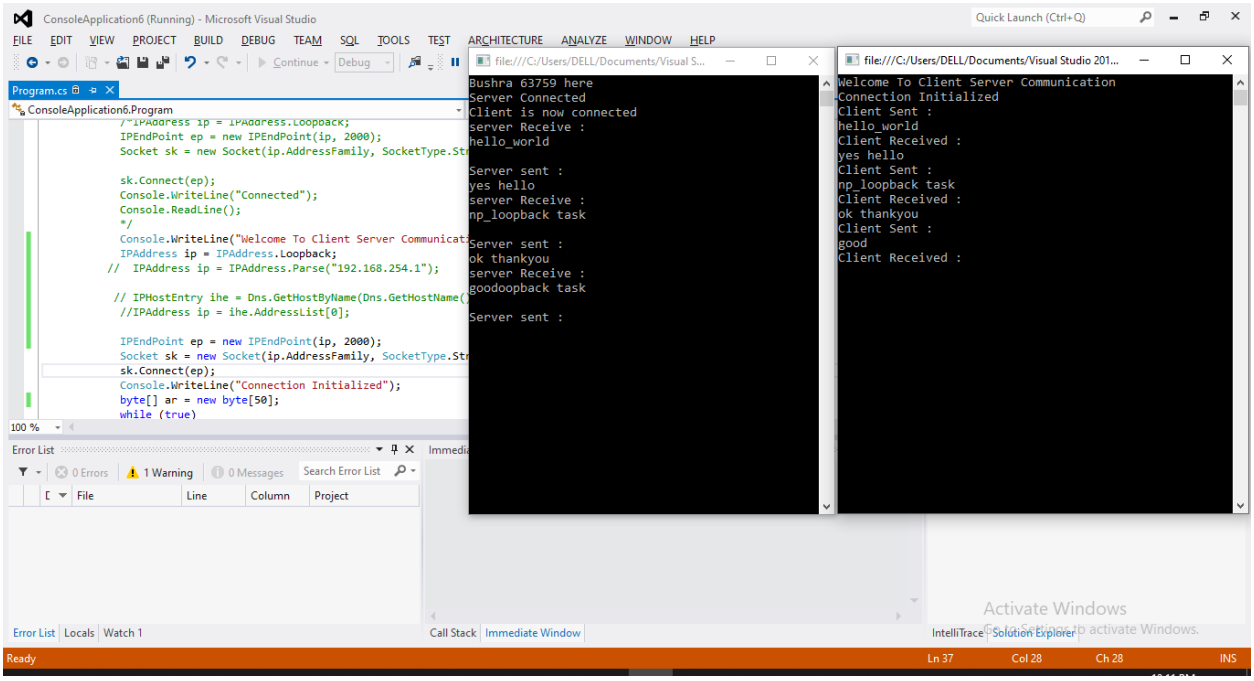
```
            //client

             Console.WriteLine("Welcome To Client Server Communication");
             IPAddress ip = IPAddress.Loopback;

             IPEndPoint ep = new IPEndPoint(ip, 2000);
             Socket sk = new Socket(ip.AddressFamily, SocketType.Stream,
ProtocolType.Tcp);
             sk.Connect(ep);
             Console.WriteLine("Connection Initialized");
             byte[] ar = new byte[50];
             while (true)
             {

                 Console.WriteLine("Client Sent : ");
                 string str = Console.ReadLine();
                 sk.Send(Encoding.ASCII.GetBytes(str));
                 Console.WriteLine("Client Received : ");
                 sk.Receive(ar);
                 Console.WriteLine(Encoding.ASCII.GetString(ar));
            }
             Console.ReadKey();
```

Server:

```
IPAddress ip = IPAddress.Loopback;
             IPEndPoint ep = new IPEndPoint(ip, 2000);
             Socket sk = new Socket(ip.AddressFamily, SocketType.Stream,
ProtocolType.Tcp);
             sk.Bind(ep);
             Console.WriteLine("Server Connected ");
             sk.Listen(1);
             Socket cl = sk.Accept();
             Console.WriteLine("Client is now connected");
             byte[] ar = new byte[50];
             while (true)
             {

                 Console.WriteLine("server Receive :");
                 cl.Receive(ar);
                 Console.WriteLine(Encoding.ASCII.GetString(ar));

                 Console.WriteLine("Server sent : ");
                 string str = Console.ReadLine();
                 cl.Send(Encoding.ASCII.GetBytes(str));

             }
```

```
Console.ReadKey();
```

Output:



1b)Client /server Communication

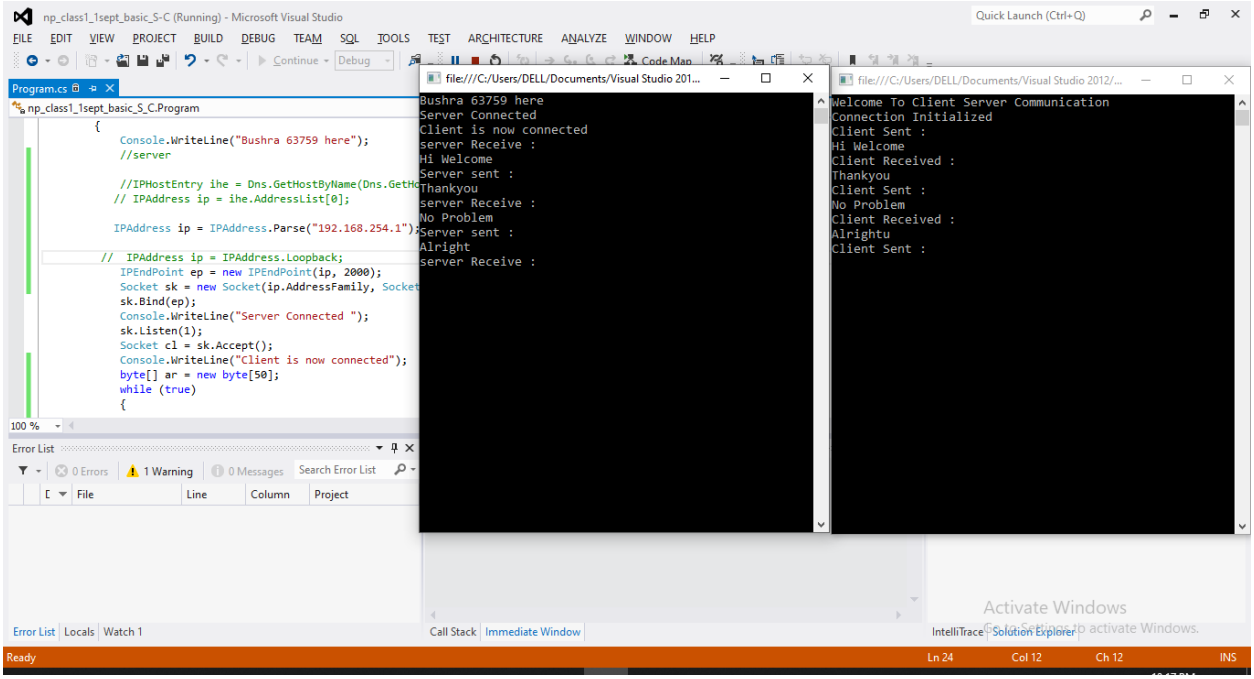
Using Parse

Client:

```
          //client

           Console.WriteLine("Welcome To Client Server Communication");
        IPAddress ip = IPAddress.Parse("192.168.254.1");


           IPEndPoint ep = new IPEndPoint(ip, 2000);
           Socket sk = new Socket(ip.AddressFamily, SocketType.Stream,
ProtocolType.Tcp);
           sk.Connect(ep);
           Console.WriteLine("Connection Initialized");
           byte[] ar = new byte[50];
           while (true)
           {

               Console.WriteLine("Client Sent : ");
               string str = Console.ReadLine();
               sk.Send(Encoding.ASCII.GetBytes(str));
               Console.WriteLine("Client Received : ");
               sk.Receive(ar);
               Console.WriteLine(Encoding.ASCII.GetString(ar));
          }
           Console.ReadKey();
```

Server:

```
IPAddress ip = IPAddress.Parse("192.168.254.1");

           IPEndPoint ep = new IPEndPoint(ip, 2000);
           Socket sk = new Socket(ip.AddressFamily, SocketType.Stream,
ProtocolType.Tcp);
           sk.Bind(ep);
           Console.WriteLine("Server Connected ");
           sk.Listen(1);
           Socket cl = sk.Accept();
           Console.WriteLine("Client is now connected");
           byte[] ar = new byte[50];
           while (true)
           {

               Console.WriteLine("server Receive :");
               cl.Receive(ar);
               Console.WriteLine(Encoding.ASCII.GetString(ar));

               Console.WriteLine("Server sent : ");
               string str = Console.ReadLine();
               cl.Send(Encoding.ASCII.GetBytes(str));

           }
           Console.ReadKey();
```


np_class1_1sept_basic_S-C (Running) - Microsoft Visual Studio
Program.cs
np_class1_1sept_basic_S_C.Program
Console.WriteLine("Bushra 63759 here");
//server
//IPHostEntry ihe = Dns.GetHostByName(Dns.GetH
// IPAddress ip = ihe.AddressList[0];
IPAddress ip = IPAddress.Parse("192.168.254.1");
// IPAddress ip = IPAddress.Loopback;
IPEndPoint ep = new IPEndPoint(ip, 2000);
Socket sk = new Socket(ip.AddressFamily, Socket
sk.Bind(ep);
Console.WriteLine("Server Connected ");
sk.Listen(1);
Socket cl = sk.Accept();
Console.WriteLine("Client is now connected");
byte[] ar = new byte[50];
while (true)
{
file:///C:/Users/DELL/Documents/Visual Studio 201...
Bushra 63759 here
Server Connected
Client is now connected
server Receive :
Hi Welcome
Server sent :
Thankyou
server Receive :
No Problem
Server sent :
Alright
server Receive :
file:///C:/Users/DELL/Documents/Visual Studio 2012/...
Welcome To Client Server Communication
Connection Initialized
Client Sent :
Hi Welcome
Client Received :
Thankyou
Client Sent :
No Problem
Client Received :
Alrightu
Client Sent :
Error List
0 Errors
1 Warning
0 Messages
Search Error List
File
Line
Column
Project
Activate Windows
Ready
Ln 24
Col 12
Ch 12
INS

1c)Client /server Communication

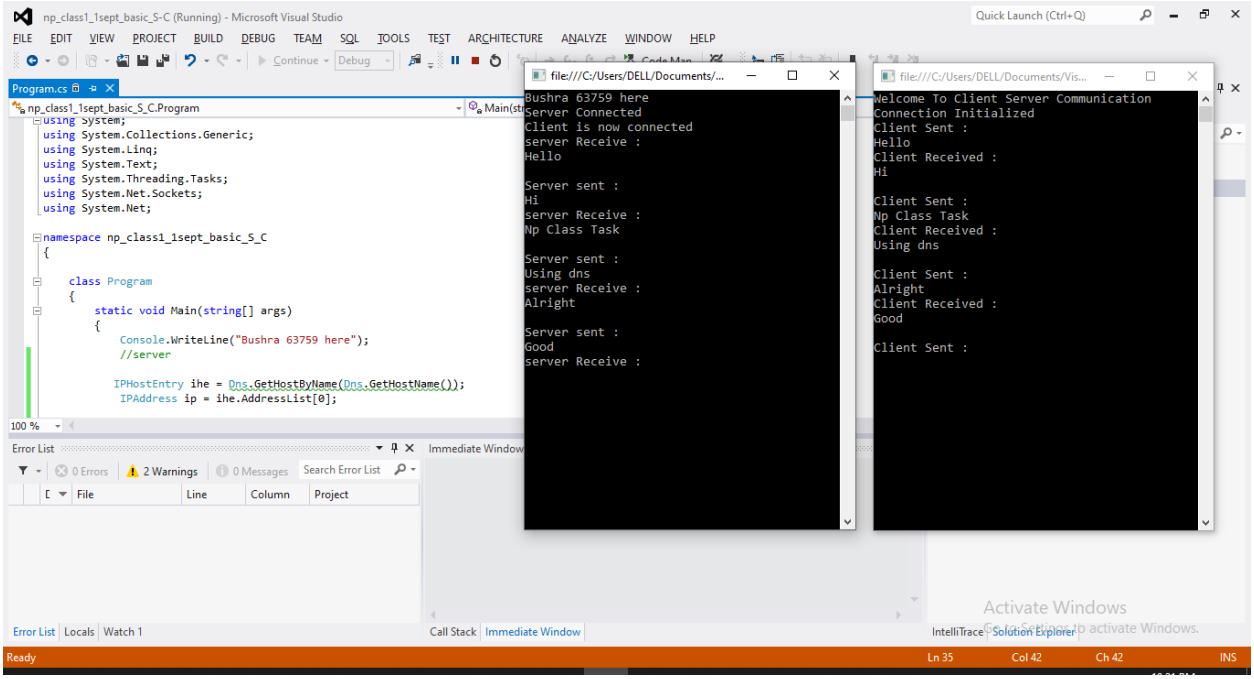
Using DNS

Client:

```
            //client

             Console.WriteLine("Welcome To Client Server Communication");

            IPHostEntry ihe = Dns.GetHostByName(Dns.GetHostName());
             IPAddress ip = ihe.AddressList[0];
 IPEndPoint ep = new IPEndPoint(ip, 2000);
             Socket sk = new Socket(ip.AddressFamily, SocketType.Stream,
ProtocolType.Tcp);
             sk.Connect(ep);
             Console.WriteLine("Connection Initialized");
             byte[] ar = new byte[50];
             while (true)
             {

                 Console.WriteLine("Client Sent : ");
                 string str = Console.ReadLine();
                 sk.Send(Encoding.ASCII.GetBytes(str));
                 Console.WriteLine("Client Received : ");
                 sk.Receive(ar);
                 Console.WriteLine(Encoding.ASCII.GetString(ar));
            }
             Console.ReadKey();
```

Server:

```
            IPHostEntry ihe = Dns.GetHostByName(Dns.GetHostName());
             IPAddress ip = ihe.AddressList[0];
           IPEndPoint ep = new IPEndPoint(ip, 2000);
             Socket sk = new Socket(ip.AddressFamily, SocketType.Stream,
ProtocolType.Tcp);
             sk.Bind(ep);
             Console.WriteLine("Server Connected ");
             sk.Listen(1);
             Socket cl = sk.Accept();
             Console.WriteLine("Client is now connected");
             byte[] ar = new byte[50];
             while (true)
             {

                 Console.WriteLine("server Receive :");
                 cl.Receive(ar);
                 Console.WriteLine(Encoding.ASCII.GetString(ar));

                 Console.WriteLine("Server sent : ");
                 string str = Console.ReadLine();
                 cl.Send(Encoding.ASCII.GetBytes(str));

             }
             Console.ReadKey();
```


np_class1_1sept_basic_S-C (Running) - Microsoft Visual Studio
Program.cs
np_class1_1sept_basic_S_C.Program
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Net.Sockets;
using System.Net;
namespace np_class1_1sept_basic_S_C
{
class Program
{
static void Main(string[] args)
{
Console.WriteLine("Bushra 63759 here");
//server
IPHostEntry ihe = Dns.GetHostByName(Dns.GetHostName());
IPAddress ip = ihe.AddressList[0];
file:///C:/Users/DELL/Documents/...
Bushra 63759 here
Server Connected
Client is now connected
server Receive :
Hello
Server sent :
Hi
server Receive :
Np Class Task
Server sent :
Using dns
server Receive :
Alright
Server sent :
Good
server Receive :
file:///C:/Users/DELL/Documents/Vis...
Welcome To Client Server Communication
Connection Initialized
Client Sent :
Hello
Client Received :
Hi
Client Sent :
Np Class Task
Client Received :
Using dns
Client Sent :
Alright
Client Received :
Good
Client Sent :
Error List
Immediate Window
Ready

Helper Class (29th September)

TCPClient/TCPListener

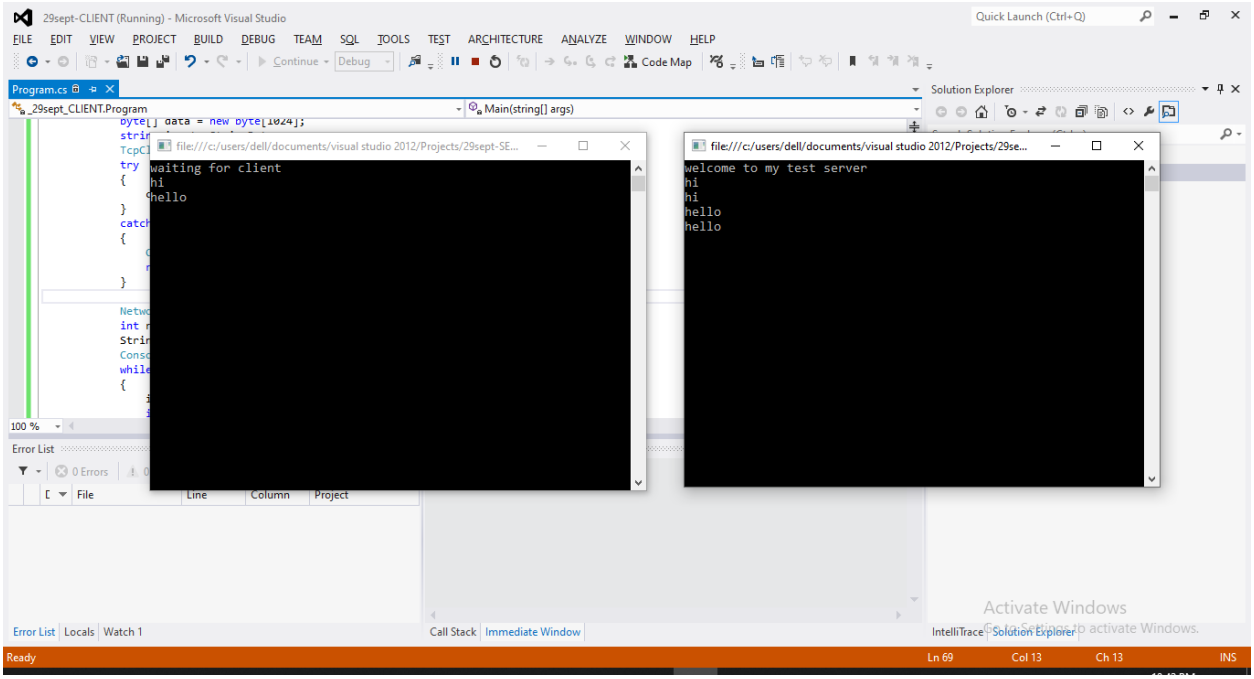
Server Code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Net.Sockets;
using System.Text;
using System.Threading.Tasks;
using System.IO;

namespace _29sept_SERVER
{
    class Program
    {
        static void Main(string[] args)
        {

            int recv;
            byte[] data = new byte[1024];
            //  Console.WriteLine("Hello World!");
            TcpListener tp = new TcpListener(9050);
            tp.Start();
            Console.WriteLine("waiting for client");
            //Socket cl=sk.Accept();
            TcpClient tc = tp.AcceptTcpClient();
            NetworkStream ns = tc.GetStream();
            string welcome = "welcome to my test server";
            data = Encoding.ASCII.GetBytes(welcome);

            ns.Write(data, 0, data.Length);

            while (true)
            {
                data = new byte[1024];
                recv = ns.Read(data, 0, data.Length);
                if (recv == 0)
                    break;

                Console.WriteLine(Encoding.ASCII.GetString(data, 0, recv));
                ns.Write(data, 0, recv);
            }
            ns.Close();
            tc.Close();
            tp.Stop();

        }
    }
}
```

Client Code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Net;
using System.Net.Sockets;
using System.IO;

namespace _29sept_CLIENT
{
    class Program
    {
        static void Main(string[] args)
        {
            byte[] data = new byte[1024];
            string input, StringData;
            TcpClient cli;
            try
            {
                cli = new TcpClient("127.0.0.1", 9050);
            }
            catch (SocketException)
            {
                Console.WriteLine("unable to connect");
                return;
            }

            NetworkStream ns = cli.GetStream();
            int recv = ns.Read(data, 0, data.Length);
            StringData = Encoding.ASCII.GetString(data, 0, recv);
            Console.WriteLine(StringData);
            while (true)
            {
                input = Console.ReadLine();
                if (input == "exit")
                    break;
                ns.Write(Encoding.ASCII.GetBytes(input), 0, input.Length);
                ns.Flush();
                data = new byte[1024];
                recv = ns.Read(data, 0, data.Length);
                StringData = Encoding.ASCII.GetString(data, 0, recv);
                Console.WriteLine(StringData);


            }
            Console.WriteLine("disconnected");
            ns.Close();
            cli.Close();
            Console.ReadLine();

            Console.ReadKey();
        }
    }
}
```

Output:


29sept-CLIENT (Running) - Microsoft Visual Studio
Program.cs
_29sept_CLIENT.Program
Main(string[] args)
file:///c:/users/dell/documents/visual studio 2012/Projects/29sept-SE...
waiting for client
hi
hello
file:///c:/users/dell/documents/visual studio 2012/Projects/29se...
welcome to my test server
hi
hi
hello
hello
Solution Explorer
Error List
Ready

Chapter # 05 Connection Oriented Sockets

Simple TCP Server and TCP Client (5.1 AND 5.2)

Dated October 6, 2020

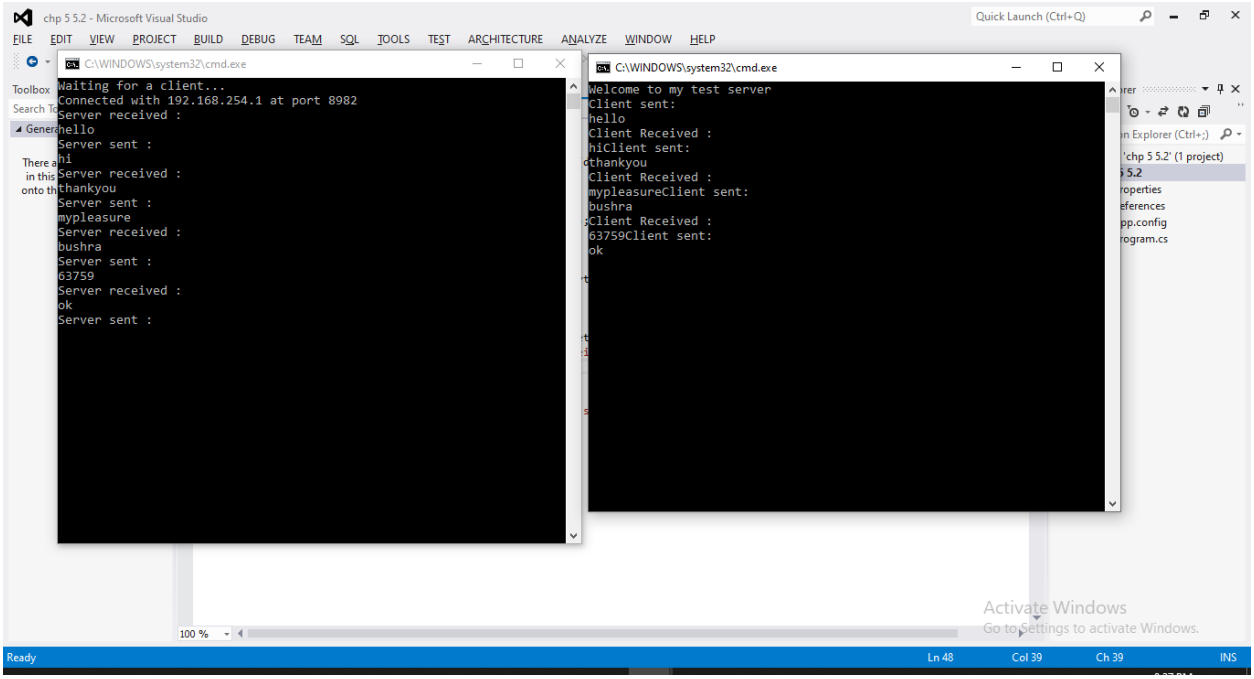
Server Code:

```
using System.Net;
using System.Net.Sockets;
namespace chp_5_5._1
{
    class Program
    {
        static void Main(string[] args)
        {
            int recv;
            byte[] data = new byte[1024];
            string input;
            IPHostEntry ihe = Dns.GetHostByName(Dns.GetHostName());
                IPAddress ip = ihe.AddressList[0];
              IPEndPoint ipep = new IPEndPoint(ip,2000);

                Socket newsock = new
Socket(AddressFamily.InterNetwork,SocketType.Stream, ProtocolType.Tcp);
                newsock.Bind(ipep);
                newsock.Listen(10);
                Console.WriteLine("Waiting for a client...");
                Socket client = newsock.Accept();
                IPEndPoint clientep =(IPEndPoint)client.RemoteEndPoint;
Console.WriteLine("Connected with {0} at port {1}",clientep.Address, clientep.Port);
                string welcome = "Welcome to my test server";
                data = Encoding.ASCII.GetBytes(welcome);
                client.Send(data, data.Length,SocketFlags.None);
              while (true)
              {
                  data = new byte[1024];
                  Console.WriteLine("Server received : ");

                        recv = client.Receive(data);
                       if (recv == 0)
                           break;
    Console.WriteLine(Encoding.ASCII.GetString(data, 0, recv));
                       Console.WriteLine("Server sent :");
                           input = Console.ReadLine();
                           if (input == "exit")
                               break;

                           client.Send(Encoding.ASCII.GetBytes(input));
                                      }
                            Console.WriteLine("Disconnected from {0}",clientep.Address);
                          client.Close();
                      newsock.Close();

  }
}

        }
```

Client Code

```
using System.Net;
using System.Net.Sockets;

namespace chp_5_5._2
{
    class Program
    {
        static void Main(string[] args)
        {
                 byte[] data = new byte[1024];
                string input, stringData;

                IPHostEntry ihe = Dns.GetHostByName(Dns.GetHostName());
                IPAddress ip = ihe.AddressList[0];
                IPEndPoint ipep = new IPEndPoint(ip, 2000);
Socket server = new Socket(AddressFamily.InterNetwork,SocketType.Stream,
ProtocolType.Tcp);
                    try
                        {
                    server.Connect(ipep);
                       }
                    catch (SocketException e)
                    {
                    Console.WriteLine("Unable to connect to server.");
                    Console.WriteLine(e.ToString());
                    return;
                    }
                    int recv = server.Receive(data);
                    stringData = Encoding.ASCII.GetString(data, 0, recv);
                    Console.WriteLine(stringData);
                    while(true)
                    {
                        Console.WriteLine("Client sent: ");
                            input = Console.ReadLine();
                            if (input == "exit")
                                break;
                          server.Send(Encoding.ASCII.GetBytes(input));

                            data = new byte[1024];
                            recv = server.Receive(data);
                            stringData = Encoding.ASCII.GetString(data, 0, recv);
                            Console.WriteLine("Client Received : ");
                        Console.Write(stringData);
                        }

                    Console.WriteLine("Disconnecting from server...");
                   server.Shutdown(SocketShutdown.Both);
                  server.Close();
 }
}

        }
```

Output:


chp 5 5.2 - Microsoft Visual Studio
C:\WINDOWS\system32\cmd.exe
Waiting for a client...
Connected with 192.168.254.1 at port 8982
Server received :
hello
Server sent :
hi
Server received :
thankyou
Server sent :
mypleasure
Server received :
bushra
Server sent :
63759
Server received :
ok
Server sent :
C:\WINDOWS\system32\cmd.exe
Welcome to my test server
Client sent:
hello
Client Received :
hiClient sent:
thankyou
Client Received :
mypleasureClient sent:
bushra
Client Received :
63759Client sent:
ok
Ready
Ln 48
Col 39
Ch 39
INS

Modified 5.1 5.2 Missing ?

Bad TCP Server and Client (5.3 AND 5.4)

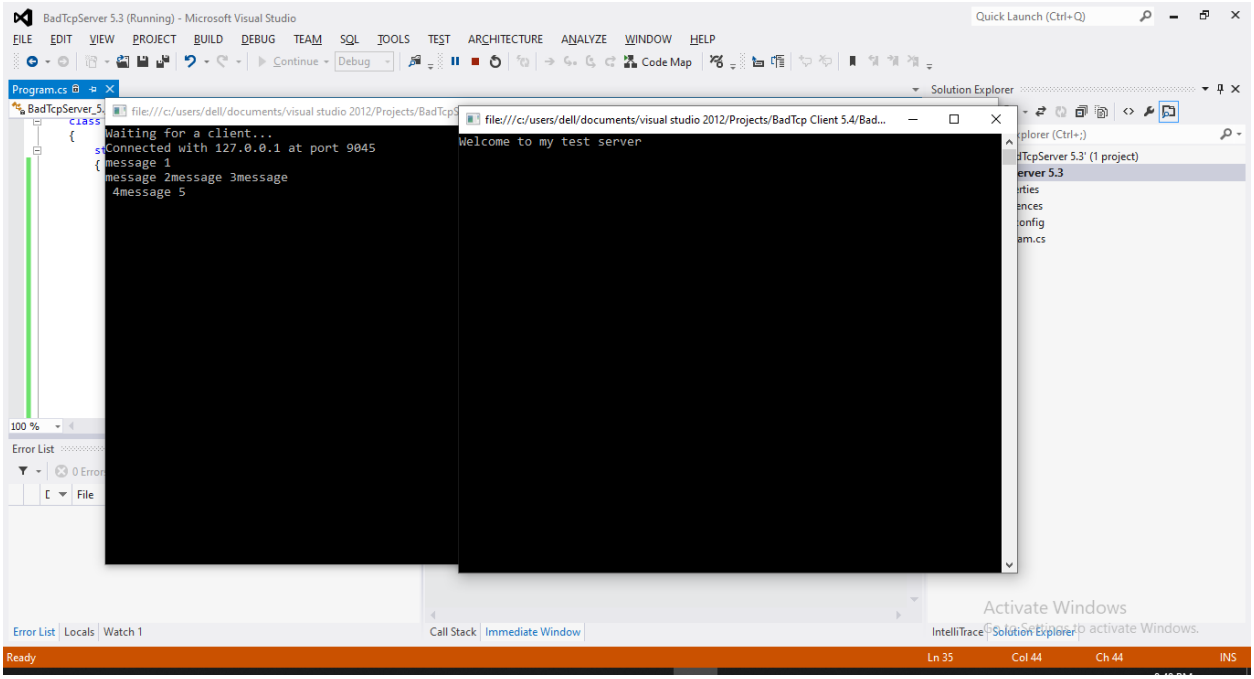
SERVER CODE:

```
using System.Net;
using System.Net.Sockets;

namespace BadTcpServer_5._3
{
    class Program
    {
        static void Main(string[] args)
        {
            int recv;
             byte[] data = new byte[1024];
             IPEndPoint ipep = new IPEndPoint(IPAddress.Any, 9050);
             Socket newsock = new Socket(AddressFamily.InterNetwork,
             SocketType.Stream, ProtocolType.Tcp);
             newsock.Bind(ipep);
             newsock.Listen(10);
             Console.WriteLine("Waiting for a client...");
             Socket client = newsock.Accept();
             string welcome = "Welcome to my test server";
             data = Encoding.ASCII.GetBytes(welcome);
             client.Send(data, data.Length,SocketFlags.None);
             IPEndPoint newclient = (IPEndPoint)client.RemoteEndPoint;
             Console.WriteLine("Connected with {0} at port {1}",
             newclient.Address, newclient.Port);
                        for (int i = 0; i < 5; i++)
                {
                            recv = client.Receive(data);
                Console.WriteLine(Encoding.ASCII.GetString(data, 0, recv));
                }
                        Console.ReadLine();
                 Console.WriteLine("Disconnecting from {0}", newclient.Address);
             client.Close();
             newsock.Close();


        }
    }
}
```

CLIENT CODE:

```
using System.Net;
using System.Net.Sockets;
namespace BadTcp_Client_5._4
{
    class Program
    {
        static void Main(string[] args)
        {
            byte[] data = new byte[1024];
            string stringData;
            IPEndPoint ipep = new IPEndPoint(IPAddress.Parse("127.0.0.1"), 9050);
            Socket server = new Socket(AddressFamily.InterNetwork,
            SocketType.Stream, ProtocolType.Tcp);
            try
            {
                server.Connect(ipep);
            }
            catch (SocketException e)
            {
                Console.WriteLine("Unable to connect to server.");
                Console.WriteLine(e.ToString());
                return;
            }
            int recv = server.Receive(data);
            stringData = Encoding.ASCII.GetString(data, 0, recv);
            Console.WriteLine(stringData);
            server.Send(Encoding.ASCII.GetBytes("message 1"));
            server.Send(Encoding.ASCII.GetBytes("message 2"));
            server.Send(Encoding.ASCII.GetBytes("message 3"));
            server.Send(Encoding.ASCII.GetBytes("message 4"));
            server.Send(Encoding.ASCII.GetBytes("message 5"));
            Console.ReadLine();
            Console.WriteLine("Disconnecting from server...");
            server.Shutdown(SocketShutdown.Both);
            server.Close();

        }
    }
}
```

OUTPUT:



- Then this will print disconnecting and will be closed.

Solving TCP Message Problem

1. Always send fixed-sized messages (5.5 And 5.6)

Server Code:

```
using System.Net;
using System.Net.Sockets;
namespace Chap_5_5._5
{
   class Program
    {
        private static int SendData(Socket s, byte[] data)
        {
            int total = 0;
            int size = data.Length;
            int dataleft = size;
            int sent;
            while (total < size)
            {
                sent = s.Send(data, total, dataleft, SocketFlags.None);
                total += sent;
                dataleft -= sent;
            }
            return total;
        }
        private static byte[] ReceiveData(Socket s, int size)
        {
            int total = 0;
            int dataleft = size;
            byte[] data = new byte[size];
            int recv;
            while (total < size)
            {
                recv = s.Receive(data, total, dataleft, 0);
                if (recv == 0)
                {
                    data = Encoding.ASCII.GetBytes("exit");
                    break;
                }
                total += recv;
                dataleft -= recv;
            }
            return data;
        }

        static void Main(string[] args)
        {
            byte[] data = new byte[1024];
            IPEndPoint ipep = new IPEndPoint(IPAddress.Any, 9050);
            Socket newsock = new Socket(AddressFamily.InterNetwork,
            SocketType.Stream, ProtocolType.Tcp);
            newsock.Bind(ipep);
            newsock.Listen(10);
            Console.WriteLine("Waiting for a client...");
            Socket client = newsock.Accept();
            IPEndPoint newclient = (IPEndPoint)client.RemoteEndPoint;
```

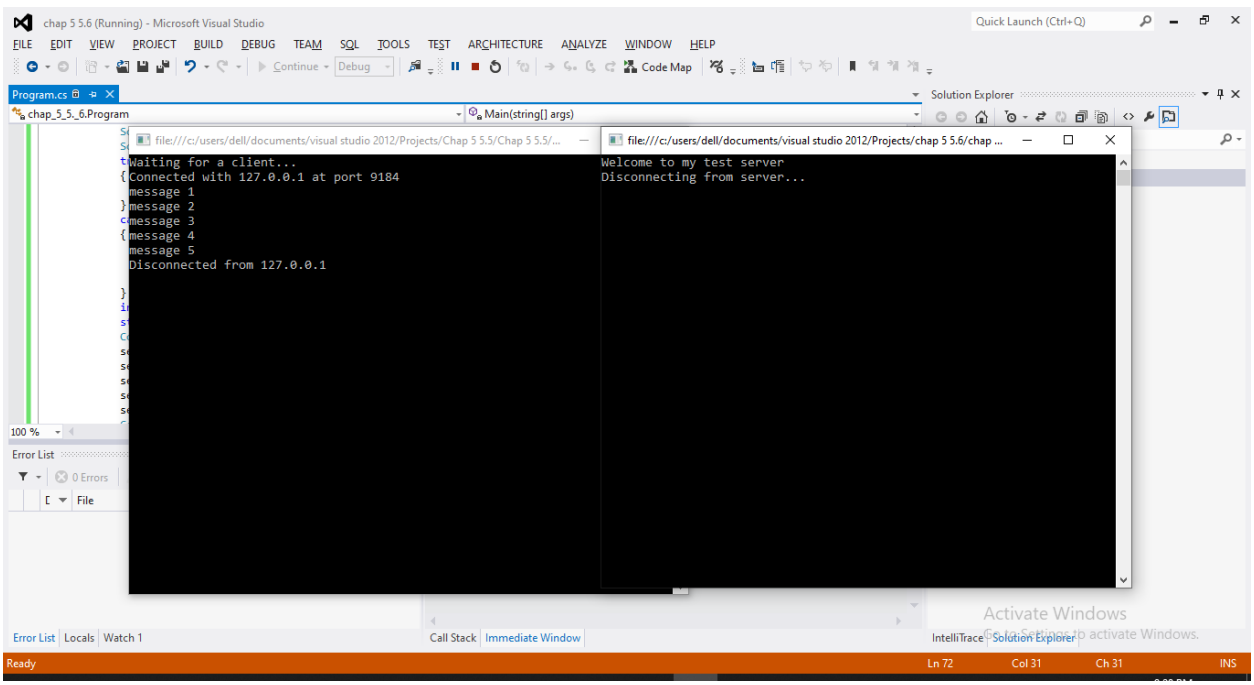
```
            Console.WriteLine("Connected with {0} at port {1}",
            newclient.Address, newclient.Port);
            string welcome = "Welcome to my test server";
            data = Encoding.ASCII.GetBytes(welcome);
            int sent = SendData(client, data);
            for (int i = 0; i < 5; i++)
            {
                data = ReceiveData(client, 9);
                Console.WriteLine(Encoding.ASCII.GetString(data));
            }
            Console.WriteLine("Disconnected from {0}", newclient.Address);
            Console.ReadKey();
            client.Close();
            newsock.Close();



        }
    }
}
```

Client Code:

```
using System.Net;
using System.Net.Sockets;
namespace chap_5_5._6
{
    class Program
    {
        private static int SendData(Socket s, byte[] data)
        {
            int total = 0;
            int size = data.Length;
            int dataleft = size;
            int sent;
            while (total < size)
            {
                sent = s.Send(data, total, dataleft, SocketFlags.None);
                total += sent;
                dataleft -= sent;
            }
            return total;
        }
        private static byte[] ReceiveData(Socket s, int size)
 {
 int total = 0;
 int dataleft = size;
 byte[] data = new byte[size];
 int recv;
 while(total < size)
 {
 recv = s.Receive(data, total, dataleft, 0);
 if (recv == 0)
 {
 data = Encoding.ASCII.GetBytes("exit ");
 break;
 }
 total += recv;
 dataleft -= recv;
 }
 return data;
 }

        static void Main(string[] args)
        {
            byte[] data = new byte[1024];
            int sent;
            IPEndPoint ipep = new IPEndPoint(IPAddress.Parse("127.0.0.1"), 9050);
            Socket server = new Socket(AddressFamily.InterNetwork,
            SocketType.Stream, ProtocolType.Tcp);
            try
            {
                server.Connect(ipep);
            }
            catch (SocketException e)
            {
                Console.WriteLine("Unable to connect to server.");
                Console.WriteLine(e.ToString());
                return;
```

```
            }
            int recv = server.Receive(data);
            string stringData = Encoding.ASCII.GetString(data, 0, recv);
            Console.WriteLine(stringData);
            sent = SendData(server, Encoding.ASCII.GetBytes("message 1"));
            sent = SendData(server, Encoding.ASCII.GetBytes("message 2"));
            sent = SendData(server, Encoding.ASCII.GetBytes("message 3"));
            sent = SendData(server, Encoding.ASCII.GetBytes("message 4"));
            sent = SendData(server, Encoding.ASCII.GetBytes("message 5"));
            Console.WriteLine("Disconnecting from server...");
            Console.ReadKey();
            server.Shutdown(SocketShutdown.Both);
            server.Close();

        }
    }
}
```

Output:



2. Send the message size with each message (5.7 And 5.8)

Server Code:

```
using System.Net;
using System.Net.Sockets;


namespace chap_5__5._7_
{
    class Program
    {
        private static int SendVarData(Socket s, byte[] data)
        {
            int total = 0;
            int size = data.Length;
            int dataleft = size;
            int sent;
            byte[] datasize = new byte[4];
            datasize = BitConverter.GetBytes(size);
            sent = s.Send(datasize);
            while (total < size)
            {
                sent = s.Send(data, total, dataleft, SocketFlags.None);
                total += sent;
                dataleft -= sent;
            }
            return total;
        }
        private static byte[] ReceiveVarData(Socket s)
        {
            int total = 0;
            int recv;
            byte[] datasize = new byte[4];
            recv = s.Receive(datasize, 0, 4, 0);
            int size = BitConverter.ToInt32(datasize, 0);
            int dataleft = size;
            byte[] data = new byte[size];
            while (total < size)
            {
                recv = s.Receive(data, total, dataleft, 0);
                if (recv == 0)
                {
                    data = Encoding.ASCII.GetBytes("exit ");
                    break;
                }
                total += recv;

                dataleft -= recv;
            }
            return data;
        }

        static void Main(string[] args)
        {

            byte[] data = new byte[1024];
            IPEndPoint ipep = new IPEndPoint(IPAddress.Any, 9050);
```

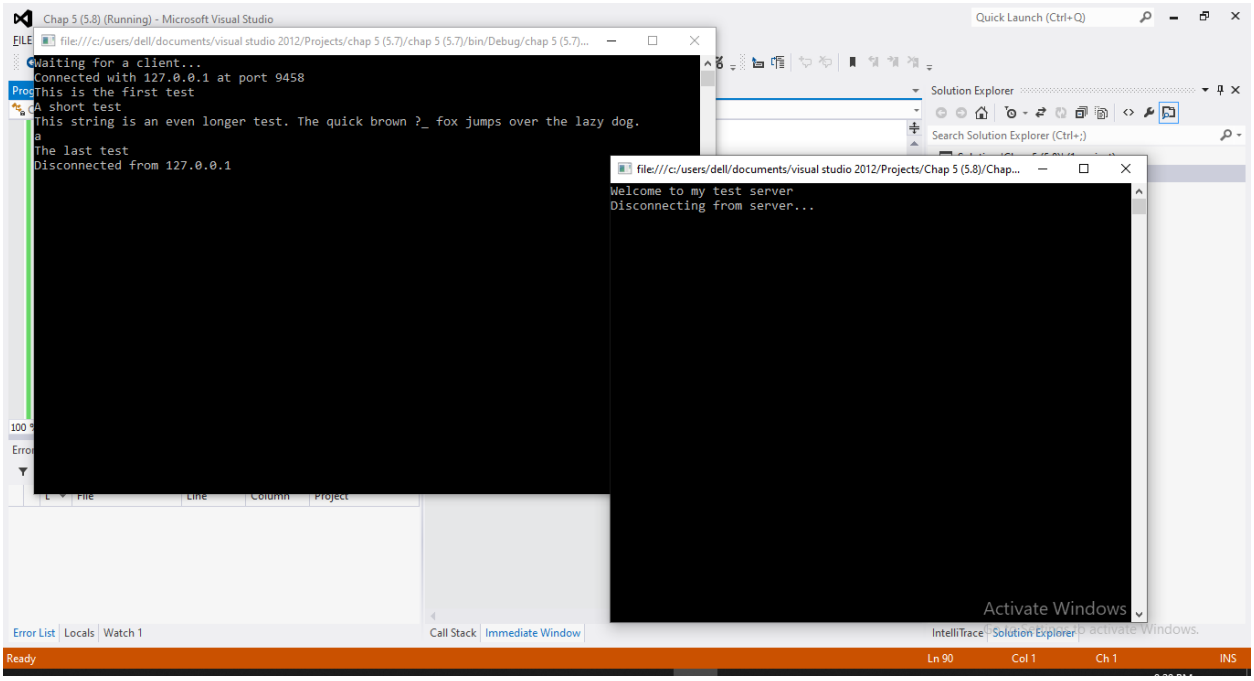
```
            Socket newsock = new Socket(AddressFamily.InterNetwork,
            SocketType.Stream, ProtocolType.Tcp);
            newsock.Bind(ipep);
            newsock.Listen(10);
            Console.WriteLine("Waiting for a client...");
            Socket client = newsock.Accept();
            IPEndPoint newclient = (IPEndPoint)client.RemoteEndPoint;
            Console.WriteLine("Connected with {0} at port {1}",
            newclient.Address, newclient.Port);
            string welcome = "Welcome to my test server";
            data = Encoding.ASCII.GetBytes(welcome);
            int sent = SendVarData(client, data);
            for (int i = 0; i < 5; i++)
            {
                data = ReceiveVarData(client);
                Console.WriteLine(Encoding.ASCII.GetString(data));
            }
            Console.WriteLine("Disconnected from {0}", newclient.Address);
            Console.ReadKey();
            client.Close();
            newsock.Close();

        }
    }
}
```

Client Code:

```
using System.Net;
using System.Net.Sockets;

namespace Chap_5__5._8_
{
    class Program
    {
        private static int SendVarData(Socket s, byte[] data)
 {
        int total = 0;
        int size = data.Length;
        int dataleft = size;
        int sent;
        byte[] datasize = new byte[4];
        datasize = BitConverter.GetBytes(size);
        sent = s.Send(datasize);
        while (total < size)
        {
            sent = s.Send(data, total, dataleft, SocketFlags.None);
            total += sent;
            dataleft -= sent;
        }
        return total;
        }
        private static byte[] ReceiveVarData(Socket s)
        {
            int total = 0;
            int recv;
            byte[] datasize = new byte[4];
            recv = s.Receive(datasize, 0, 4, 0);
            int size = BitConverter.ToInt32(datasize, 0);
            int dataleft = size;
            byte[] data = new byte[size];
            while (total < size)
            {
                recv = s.Receive(data, total, dataleft, 0);
                if (recv == 0)
                {
                    data = Encoding.ASCII.GetBytes("exit ");
                    break;
                }
                total += recv;
                dataleft -= recv;
            }
            return data;
        }
        static void Main(string[] args)
        {
            byte[] data = new byte[1024];
            int sent;
            IPEndPoint ipep = new IPEndPoint(IPAddress.Parse("127.0.0.1"), 9050);
            Socket server = new Socket(AddressFamily.InterNetwork,
            SocketType.Stream, ProtocolType.Tcp);
            try
            {
                server.Connect(ipep);
```

```
            }
            catch (SocketException e)
            {
                Console.WriteLine("Unable to connect to server.");
                Console.WriteLine(e.ToString());
                return;
            }
            data = ReceiveVarData(server);
            string stringData = Encoding.ASCII.GetString(data);
            Console.WriteLine(stringData);
            string message1 = "This is the first test";
            string message2 = "A short test";
            string message3 = "This string is an even longer test. The quick brown Â_ fox
jumps over the lazy dog.";
            string message4 = "a";
            string message5 = "The last test";
            sent = SendVarData(server, Encoding.ASCII.GetBytes(message1));
            sent = SendVarData(server, Encoding.ASCII.GetBytes(message2));
            sent = SendVarData(server, Encoding.ASCII.GetBytes(message3));
            sent = SendVarData(server, Encoding.ASCII.GetBytes(message4));
            sent = SendVarData(server, Encoding.ASCII.GetBytes(message5));
            Console.WriteLine("Disconnecting from server...");
            Console.ReadKey();
            server.Shutdown(SocketShutdown.Both);
            server.Close();
        }
    }
}
```

Output:



Network Stream (5.9)

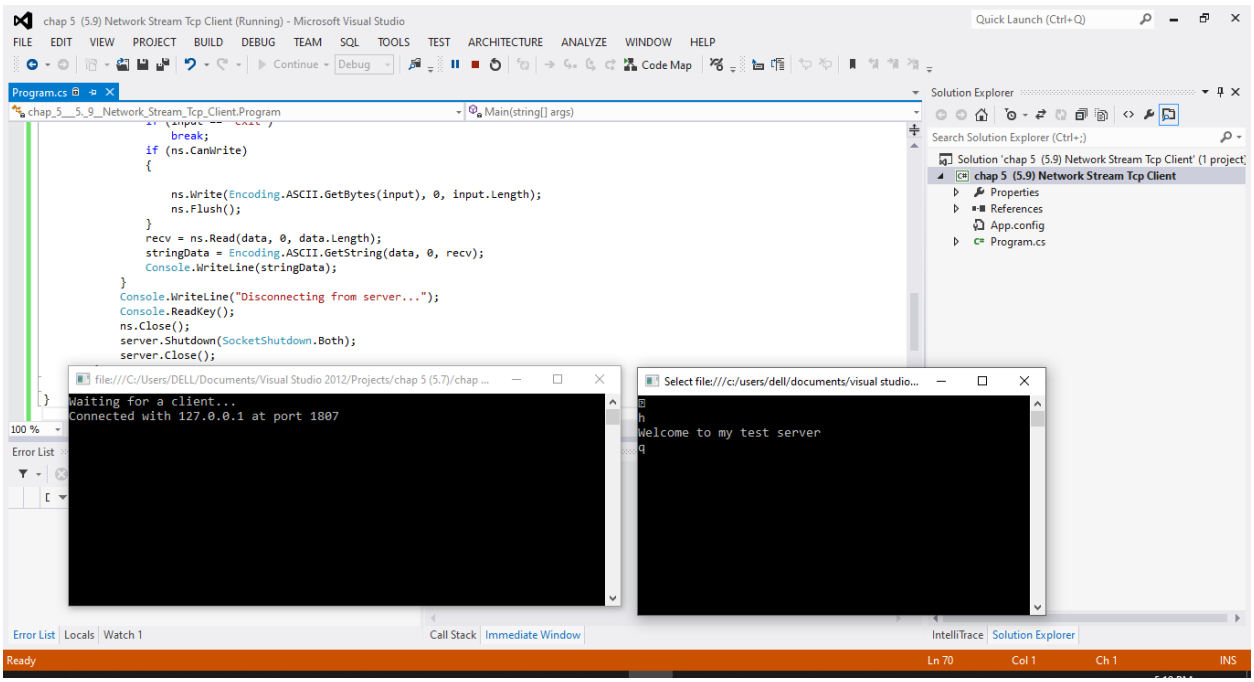
TCP

Client Code:

```
using System.Net;
using System.Net.Sockets;

namespace chap_5___5._9__Network_Stream_Tcp_Client
{
    class Program
    {
        static void Main(string[] args)
        {

            byte[] data = new byte[1024];
            string input, stringData;
            int recv;
            IPEndPoint ipep = new IPEndPoint(
            IPAddress.Parse("127.0.0.1"), 9050);
            Socket server = new Socket(AddressFamily.InterNetwork,
            SocketType.Stream, ProtocolType.Tcp);
            try
            {
                server.Connect(ipep);
            }
            catch (SocketException e)
            {
                Console.WriteLine("Unable to connect to server.");
                Console.WriteLine(e.ToString());
                return;
            }
            NetworkStream ns = new NetworkStream(server);
            if (ns.CanRead)
            {
                recv = ns.Read(data, 0, data.Length);
                stringData = Encoding.ASCII.GetString(data, 0, recv);
                Console.WriteLine(stringData);
            }
            else
            {
                Console.WriteLine("Error: Can't read from this socket");
                ns.Close();
                server.Close();
                return;
            }
            while (true)
            {
                input = Console.ReadLine();
                if (input == "exit")
                    break;
                if (ns.CanWrite)
                {

                    ns.Write(Encoding.ASCII.GetBytes(input), 0, input.Length);
                    ns.Flush();
                }
```

```
                recv = ns.Read(data, 0, data.Length);
                stringData = Encoding.ASCII.GetString(data, 0, recv);
                Console.WriteLine(stringData);
            }
            Console.WriteLine("Disconnecting from server...");
            Console.ReadKey();
            ns.Close();
            server.Shutdown(SocketShutdown.Both);
            server.Close();
        }
    }
}
```

Output:



Stream TCP Server and Client (5.10 And 5.11)

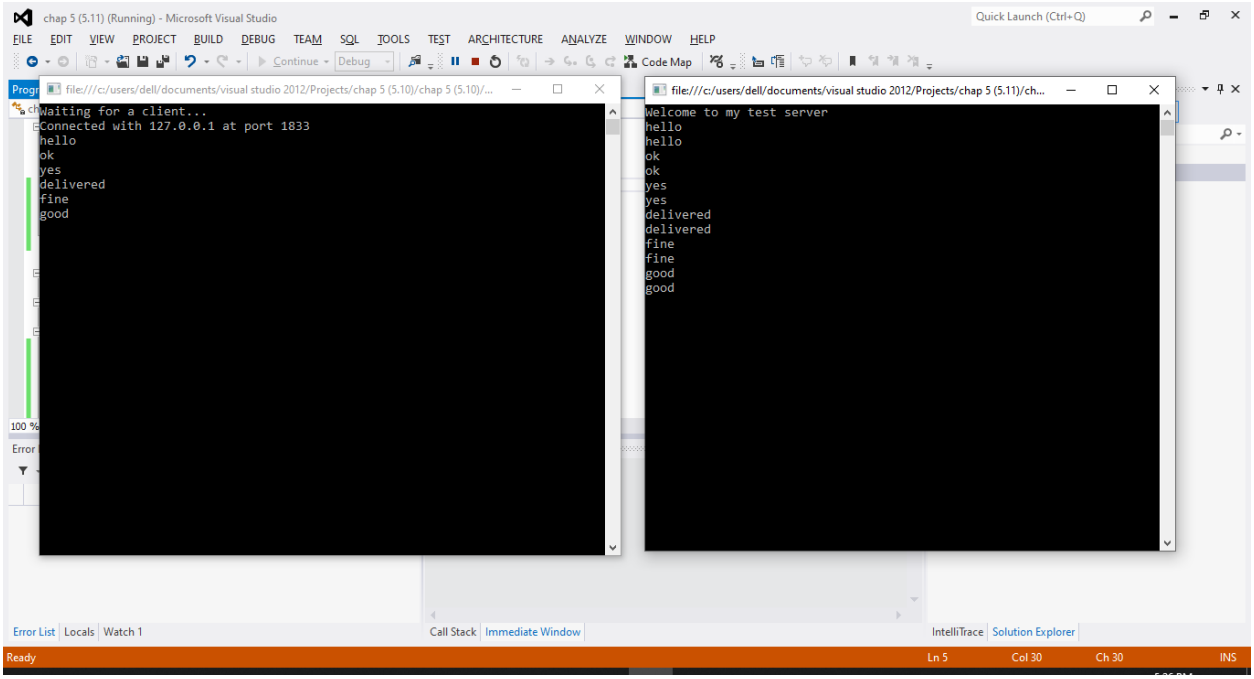
Server Code:

```
using System.Net;
using System.Net.Sockets;
using System.IO;

namespace chap_5__5._10_
{
    class Program
    {
        static void Main(string[] args)
        {
            string data;
            IPEndPoint ipep = new IPEndPoint(IPAddress.Any, 9050);
            Socket newsock = new Socket(AddressFamily.InterNetwork,
            SocketType.Stream, ProtocolType.Tcp);
            newsock.Bind(ipep);
            newsock.Listen(10);
            Console.WriteLine("Waiting for a client...");
            Socket client = newsock.Accept();
            IPEndPoint newclient = (IPEndPoint)client.RemoteEndPoint;
            Console.WriteLine("Connected with {0} at port {1}",
            newclient.Address, newclient.Port);
            NetworkStream ns = new NetworkStream(client);
            StreamReader sr = new StreamReader(ns);
            StreamWriter sw = new StreamWriter(ns);
            string welcome = "Welcome to my test server";
            sw.WriteLine(welcome);
            sw.Flush();
            while (true)
            {
                try
                {
                    data = sr.ReadLine();
                }
                catch (IOException)
                {
                    break;
                }

                Console.WriteLine(data);
                sw.WriteLine(data);
                sw.Flush();
            }
            Console.WriteLine("Disconnected from {0}", newclient.Address);
            sw.Close();
            sr.Close();
            ns.Close();

        }
    }
}
```

Client Code:

```
using System.Net;
using System.Net.Sockets;
using System.IO;

namespace chap_5__5._11_
{
    class Program
    {
        static void Main(string[] args)
        {
            string data;
            string input;
            IPEndPoint ipep = new IPEndPoint(
            IPAddress.Parse("127.0.0.1"), 9050);
            Socket server = new Socket(AddressFamily.InterNetwork,
            SocketType.Stream, ProtocolType.Tcp);
            try
            {
                server.Connect(ipep);
            }
            catch (SocketException e)
            {
                Console.WriteLine("Unable to connect to server.");
                Console.WriteLine(e.ToString());
                return;
            }
            NetworkStream ns = new NetworkStream(server);
            StreamReader sr = new StreamReader(ns);
            StreamWriter sw = new StreamWriter(ns);
            data = sr.ReadLine();
            Console.WriteLine(data);
            while (true)
            {
                input = Console.ReadLine();
                if (input == "exit")
                    break;
                sw.WriteLine(input);
                sw.Flush();
                data = sr.ReadLine();
                Console.WriteLine(data);
            }
            Console.WriteLine("Disconnecting from server...");
            sr.Close();
            sw.Close();
            ns.Close();
            server.Shutdown(SocketShutdown.Both);
            server.Close();

        }
    }
}
```

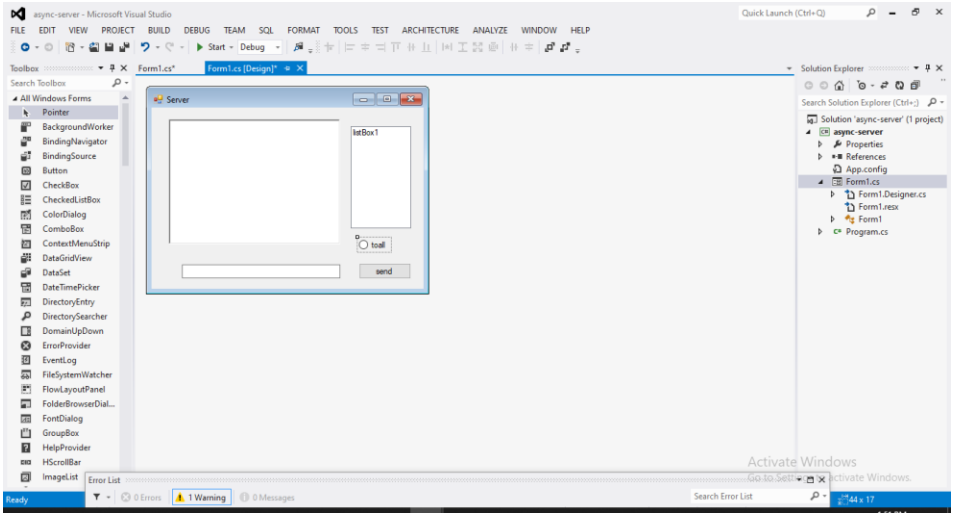
Output:


chap 5 (5.11) (Running) - Microsoft Visual Studio
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM SQL TOOLS TEST ARCHITECTURE ANALYZE WINDOW HELP
Continue
Debug
Code Map
Quick Launch (Ctrl+Q)
file:///c:/users/dell/documents/visual studio 2012/Projects/chap 5 (5.10)/chap 5 (5.10)/...
Waiting for a client...
Connected with 127.0.0.1 at port 1833
hello
ok
yes
delivered
fine
good
file:///c:/users/dell/documents/visual studio 2012/Projects/chap 5 (5.11)/ch...
Welcome to my test server
hello
hello
ok
ok
yes
yes
delivered
delivered
fine
fine
good
good
Error List
Locals
Watch 1
Call Stack
Immediate Window
IntelliTrace
Solution Explorer
Ready
Ln 5
Col 30
Ch 30
INS

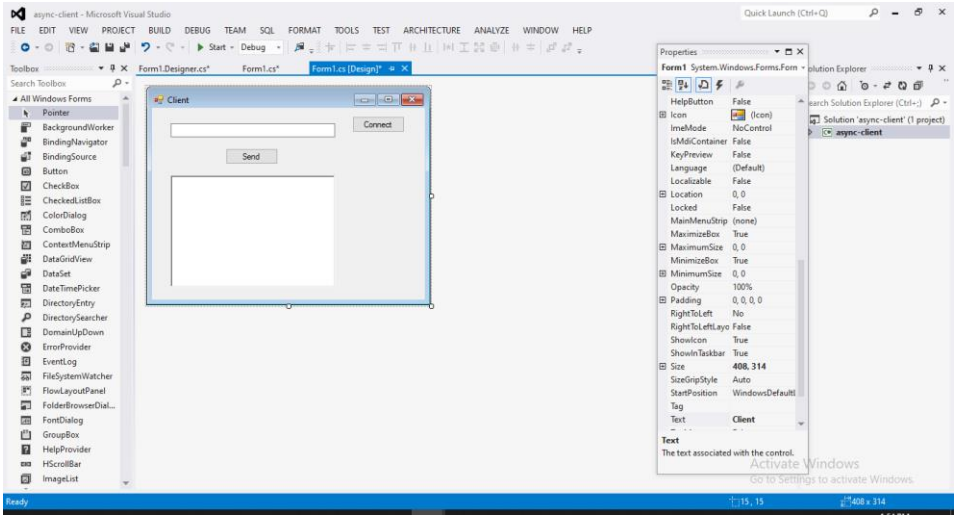
Asynchronous Client/Server

Date October 13, 2020

Individual Messaging



Client Form:



Server Code

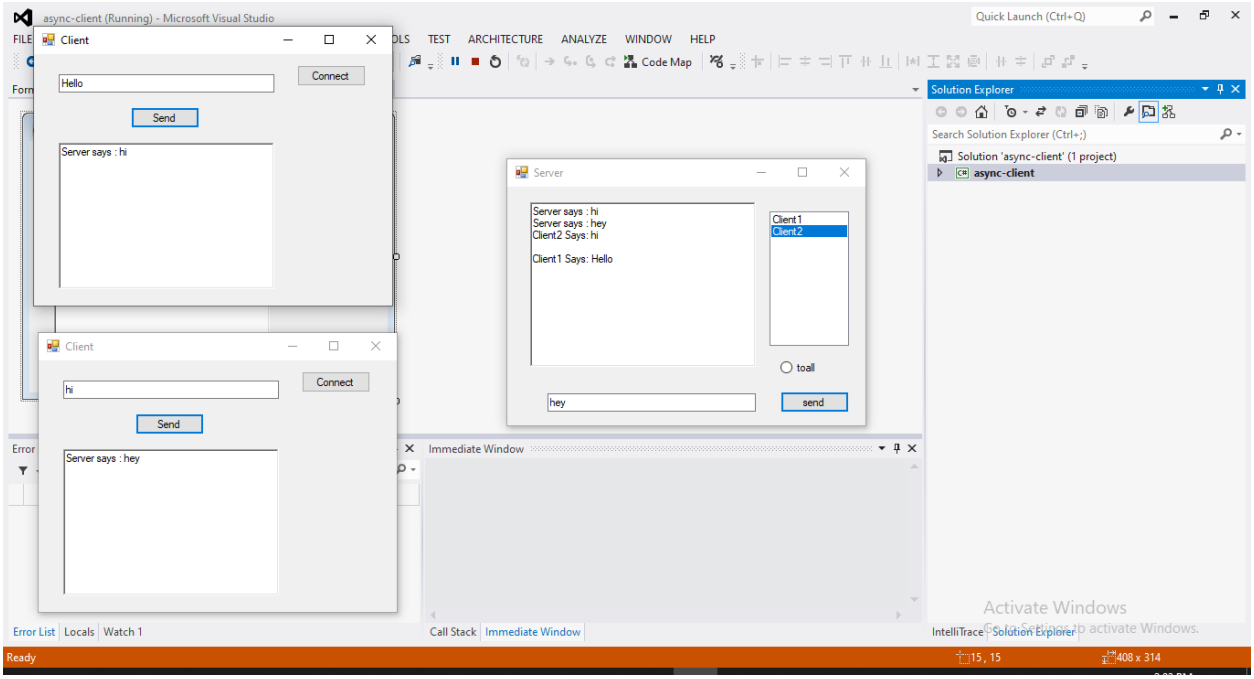
```
using System.Net;
using System.Net.Sockets;
using System.IO;

namespace async_server
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            CheckForIllegalCrossThreadCalls = false;
            TcpListener listener = new TcpListener(IPAddress.Loopback, 11000);
            listener.Start(10);
            listener.BeginAcceptTcpClient(new AsyncCallback(client_connect),listener);

        }
        Dictionary<string, TcpClient> lstClients = new Dictionary<string, TcpClient>();
        byte[] b = new byte[1024];
        private void client_connect(IAsyncResult ar){

            TcpListener listener = (TcpListener)ar.AsyncState;
            TcpClient client = listener.EndAcceptTcpClient(ar);
            NetworkStream ns = client.GetStream();
            object [] a  = new object[2];
            a[0]=ns;
            a[1]=client;
            ns.BeginRead(b,0,b.Length ,new AsyncCallback(ReadMsg),a);
            listener.BeginAcceptTcpClient(new AsyncCallback(client_connect),listener);

        }
        private void ReadMsg(IAsyncResult ar){

            object [] a = (object[])ar.AsyncState;
            NetworkStream ns = (NetworkStream)a[0];
            TcpClient client = (TcpClient)a[1];
            int count = ns.EndRead(ar);
            string msg=ASCIIEncoding.ASCII.GetString(b,0,count);
            if (msg.Contains("@name@")){

            string name= msg.Replace("@name@","");
            lstClients.Add(name, client);
             listBox1.Items.Add(name);

            }
            else{
            richTextBox1.Text+=msg+Environment.NewLine;

            }
            ns.BeginRead(b,0,b.Length, new AsyncCallback(ReadMsg),a);

        }
```

```
        private void button1_Click(object sender, EventArgs e)
        {


                TcpClient client=(TcpClient)lstClients[listBox1.SelectedItem.ToString()];

                NetworkStream ns = client.GetStream();
                StreamWriter sw = new StreamWriter(ns);

                string texttosend = "Server says : " + textBox1.Text;
                sw.WriteLine(texttosend);

                richTextBox1.Text += texttosend + Environment.NewLine;

                sw.Flush();


              }
    }
}
```

Client Code

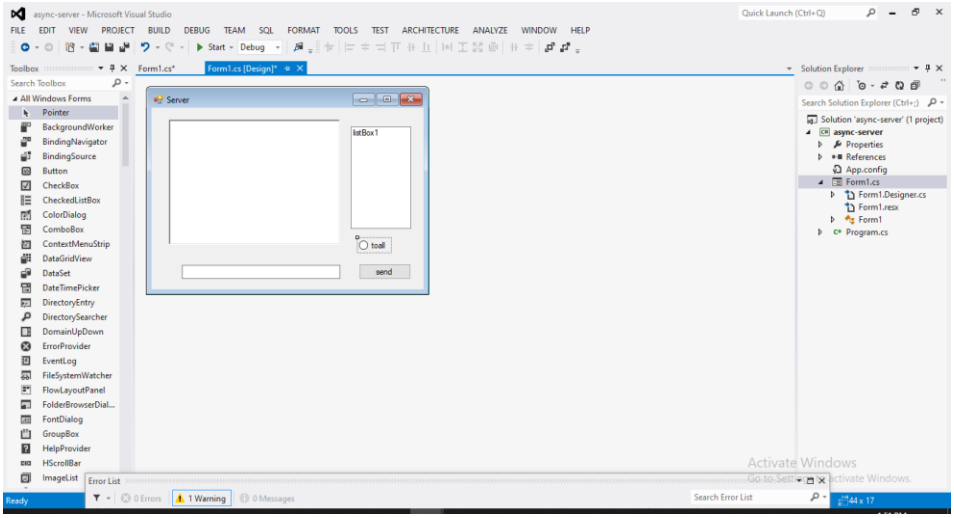
```
using System.Net;
using System.Net.Sockets;
using System.IO;

namespace async_client
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        public string name;
        byte[] b = new byte[1024];
        TcpClient client = new TcpClient();
        private void button1_Click(object sender, EventArgs e)
        {
            CheckForIllegalCrossThreadCalls = false;
            client.Connect(IPAddress.Loopback,11000);

            NetworkStream ns = client.GetStream();
            StreamWriter sw = new StreamWriter(ns);
           name = textBox1.Text;
            sw.WriteLine("@name@"+textBox1.Text);
            sw.Flush();
            ns.BeginRead(b,0,b.Length,readmsg,ns);
        }
        private void readmsg(IAsyncResult ar) {

            NetworkStream ns = (NetworkStream)ar.AsyncState;
            int count = ns.EndRead(ar);
            richTextBox1.Text += ASCIIEncoding.ASCII.GetString(b, 0, count);
            ns.BeginRead(b,0,b.Length,readmsg,ns);

        }
        private void Form1_Load(object sender, EventArgs e)
        {

        }

        private void button2_Click(object sender, EventArgs e)
        {
            NetworkStream ns = client.GetStream();
            StreamWriter sw = new StreamWriter(ns);
            sw.WriteLine(name+" Says: "+textBox1.Text);
            sw.Flush();
        }

    }
}
```

Output

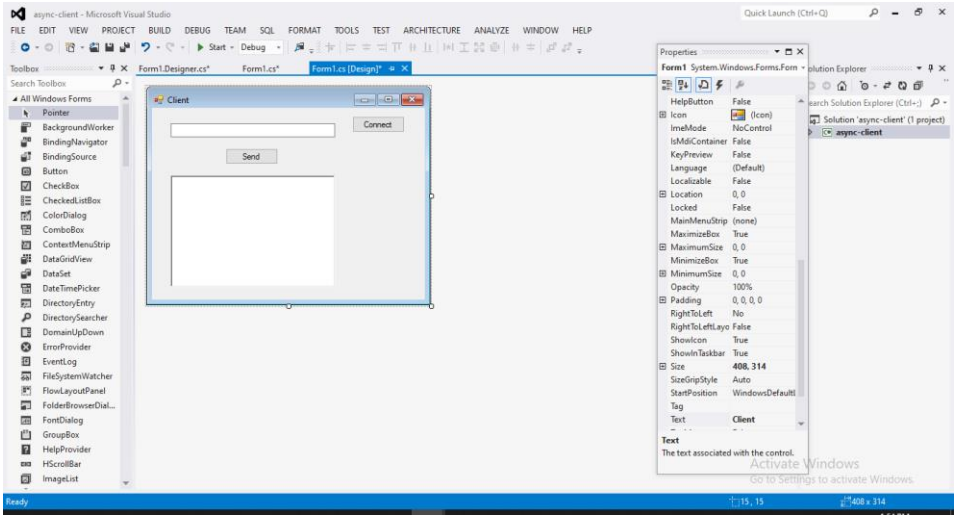

async-client (Running) - Microsoft Visual Studio
Quick Launch (Ctrl+Q)
FILE
TEST
ARCHITECTURE
ANALYZE
WINDOW
HELP
Code Map
Client
Hello
Connect
Send
Server says : hi
Client
hi
Connect
Send
Server says : hey
Server
Server says : hi
Server says : hey
Client2 Says: hi
Client1 Says: Hello
Client1
Client2
toall
hey
send
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'async-client' (1 project)
async-client
Immediate Window
Error List
Locals
Watch 1
Call Stack
Immediate Window
IntelliTrace
Solution Explorer
Activate Windows
Go to Settings to activate Windows.
Ready
15, 15
408 x 314

Group Messaging (Task)

Server Form:



Client Form:



Server Code

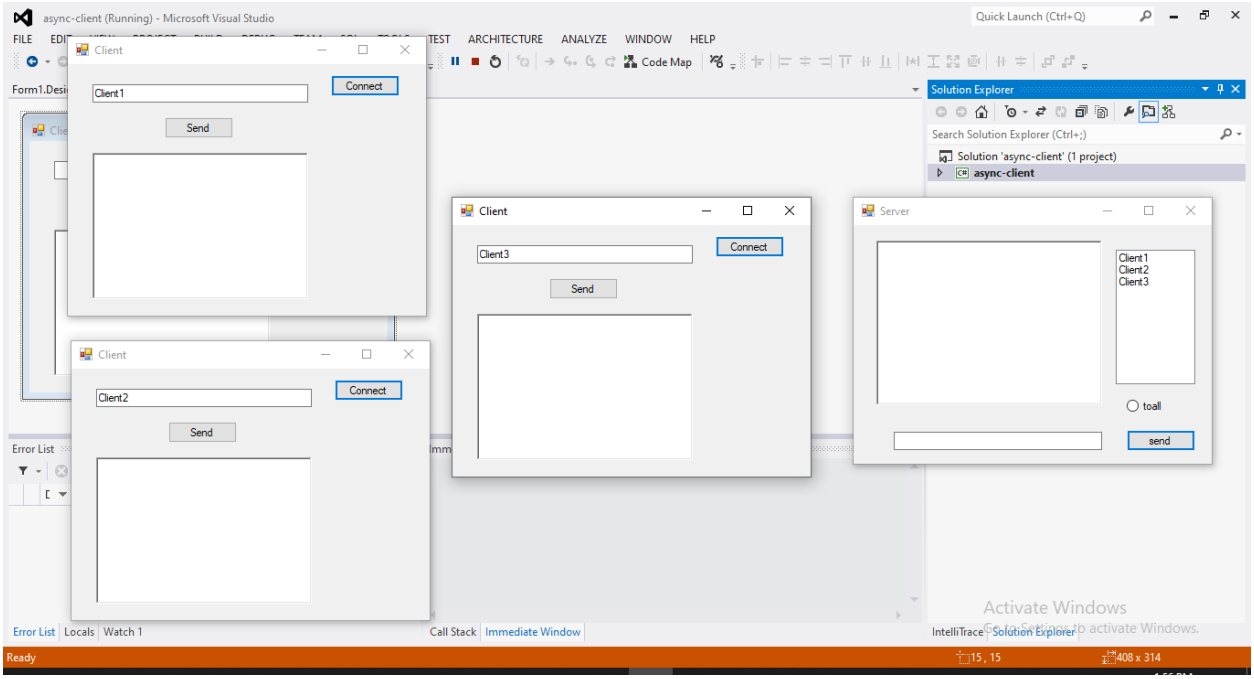
```
using System.Net;
using System.Net.Sockets;
using System.IO;

namespace async_server
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            CheckForIllegalCrossThreadCalls = false;
            TcpListener listener = new TcpListener(IPAddress.Loopback, 11000);
            listener.Start(10);
            listener.BeginAcceptTcpClient(new AsyncCallback(client_connect),listener);

        }
        Dictionary<string, TcpClient> lstClients = new Dictionary<string, TcpClient>();
        byte[] b = new byte[1024];
        private void client_connect(IAsyncResult ar){

            TcpListener listener = (TcpListener)ar.AsyncState;
            TcpClient client = listener.EndAcceptTcpClient(ar);
            NetworkStream ns = client.GetStream();
            object [] a  = new object[2];
            a[0]=ns;
            a[1]=client;
            ns.BeginRead(b,0,b.Length ,new AsyncCallback(ReadMsg),a);
            listener.BeginAcceptTcpClient(new AsyncCallback(client_connect),listener);

        }
        private void ReadMsg(IAsyncResult ar){

            object [] a = (object[])ar.AsyncState;
            NetworkStream ns = (NetworkStream)a[0];
            TcpClient client = (TcpClient)a[1];
            int count = ns.EndRead(ar);
            string msg=ASCIIEncoding.ASCII.GetString(b,0,count);
            if (msg.Contains("@name@")){

            string name= msg.Replace("@name@","");
            lstClients.Add(name, client);
             listBox1.Items.Add(name);

            }
            else{
            richTextBox1.Text+=msg+Environment.NewLine;

            }
            ns.BeginRead(b,0,b.Length, new AsyncCallback(ReadMsg),a);

        }
```

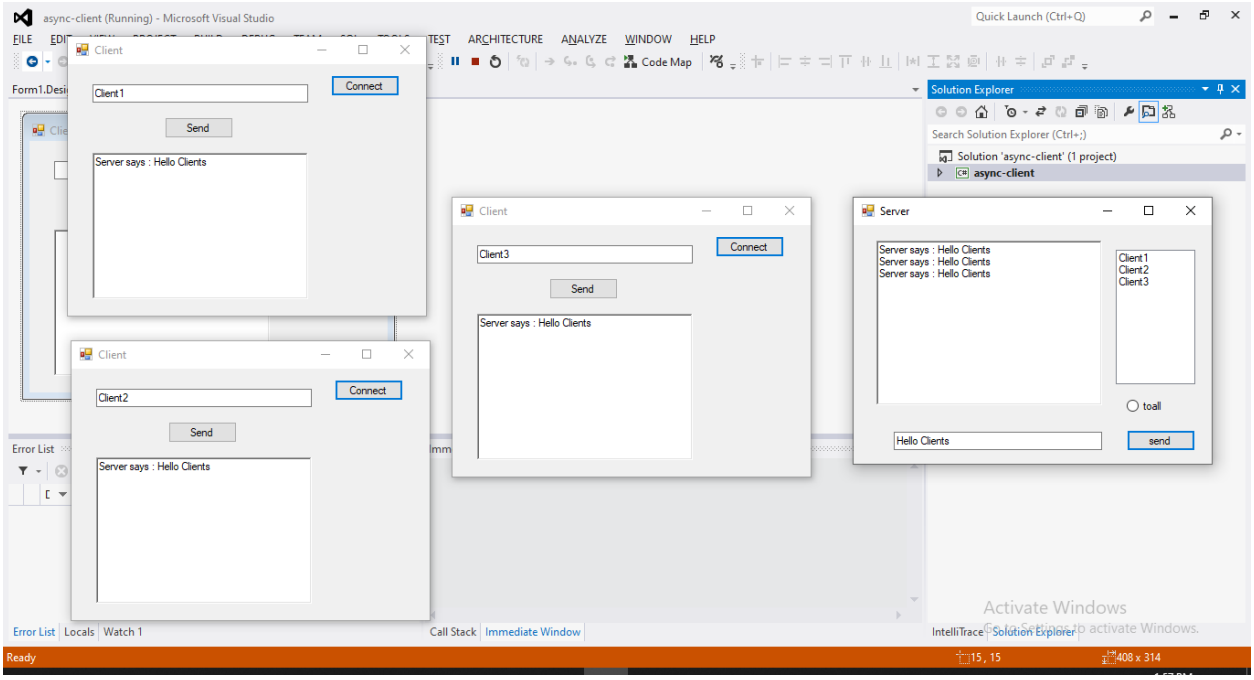
```
        private void button1_Click(object sender, EventArgs e)
        {

            if (radioButton1.Checked == true)
            {
                string items = "";
                foreach (var item in listBox1.Items)
                {
                    TcpClient client = (TcpClient)lstClients[item.ToString()];

                    NetworkStream ns = client.GetStream();
                    StreamWriter sw = new StreamWriter(ns);

                    string texttosend = "Server says : " + textBox1.Text;
                    sw.WriteLine(texttosend);

                    richTextBox1.Text += texttosend + Environment.NewLine;

                    sw.Flush();

                }

            }

            else if (radioButton1.Checked==false)
            {

                TcpClient client =
(TcpClient)lstClients[listBox1.SelectedItem.ToString()];

                NetworkStream ns = client.GetStream();
                StreamWriter sw = new StreamWriter(ns);

                string texttosend = "Server says : " + textBox1.Text;
                sw.WriteLine(texttosend);

                richTextBox1.Text += texttosend + Environment.NewLine;

                sw.Flush();

            }
            radioButton1.Checked = false;
        }
    }
}
```

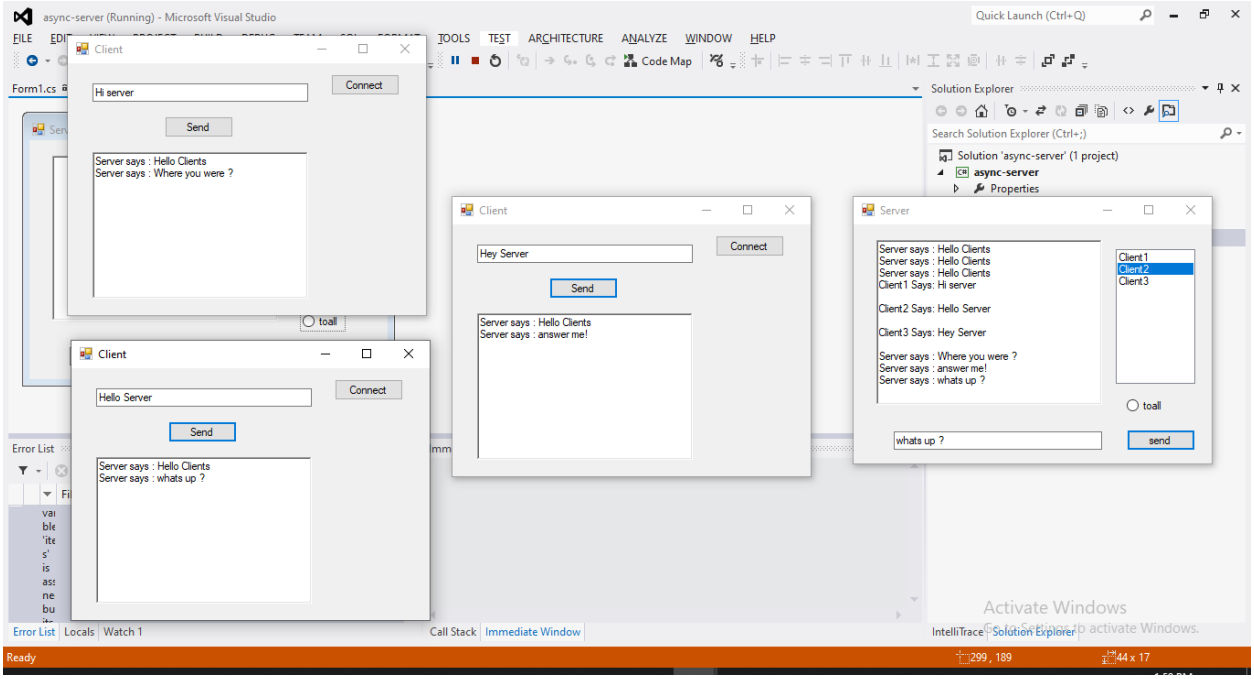
Client Code

```
using System.Net;
using System.Net.Sockets;
using System.IO;

namespace async_client
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        public string name;
        byte[] b = new byte[1024];
        TcpClient client = new TcpClient();
        private void button1_Click(object sender, EventArgs e)
        {
            CheckForIllegalCrossThreadCalls = false;
            client.Connect(IPAddress.Loopback,11000);

            NetworkStream ns = client.GetStream();
            StreamWriter sw = new StreamWriter(ns);
           name = textBox1.Text;
            sw.WriteLine("@name@"+textBox1.Text);
            sw.Flush();
            ns.BeginRead(b,0,b.Length,readmsg,ns);
        }
        private void readmsg(IAsyncResult ar) {

            NetworkStream ns = (NetworkStream)ar.AsyncState;
            int count = ns.EndRead(ar);
            richTextBox1.Text += ASCIIEncoding.ASCII.GetString(b, 0, count);
            ns.BeginRead(b,0,b.Length,readmsg,ns);

        }
        private void Form1_Load(object sender, EventArgs e)
        {

        }

        private void button2_Click(object sender, EventArgs e)
        {
            NetworkStream ns = client.GetStream();
            StreamWriter sw = new StreamWriter(ns);
            sw.WriteLine(name+" Says: "+textBox1.Text);
            sw.Flush();
        }

    }
}
```

Output


async-client (Running) - Microsoft Visual Studio
Client
Client1
Connect
Send
Client
Client2
Connect
Send
Client
Client3
Connect
Send
Server
Client1
Client2
Client3
toall
send
Solution Explorer
Solution 'async-client' (1 project)
async-client
Ready


async-client (Running) - Microsoft Visual Studio
Client
Client1
Connect
Send
Server says : Hello Clients
Client
Client2
Connect
Send
Server says : Hello Clients
Client
Client3
Connect
Send
Server says : Hello Clients
Server
Server says : Hello Clients
Server says : Hello Clients
Server says : Hello Clients
Client1
Client2
Client3
toall
Hello Clients
send
Solution Explorer
Solution 'async-client' (1 project)
async-client
Ready


async-server (Running) - Microsoft Visual Studio
Client
Hi server
Connect
Send
Server says : Hello Clients
Server says : Where you were ?
Client
Hey Server
Connect
Send
Server says : Hello Clients
Server says : answer me!
Server
Server says : Hello Clients
Server says : Hello Clients
Server says : Hello Clients
Client1 Says: Hi server
Client2 Says: Hello Server
Client3 Says: Hey Server
Server says : Where you were ?
Server says : answer me!
Server says : whats up ?
Client1
Client2
Client3
toall
whats up ?
send
Client
Hello Server
Connect
Send
Server says : Hello Clients
Server says : whats up ?
Solution Explorer
Solution 'async-server' (1 project)
async-server
Properties
Error List
Locals
Watch 1
Call Stack
Immediate Window
IntelliTrace
Solution Explorer
Ready