Mansoura University
Faculty of Computers and Information Sciences
Department of Computer Science
First Semester- 2020-2021

[CS412P] Distributed Systems

Grade: FORTH GRADE

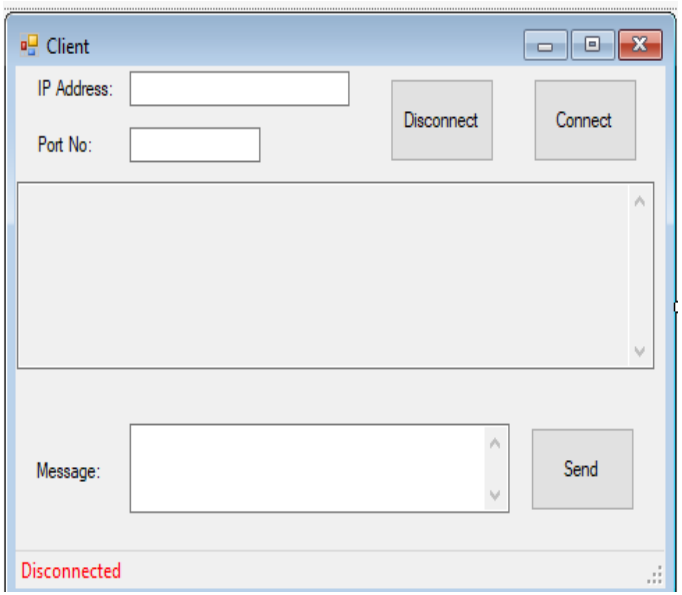
By : Zeinab Awad

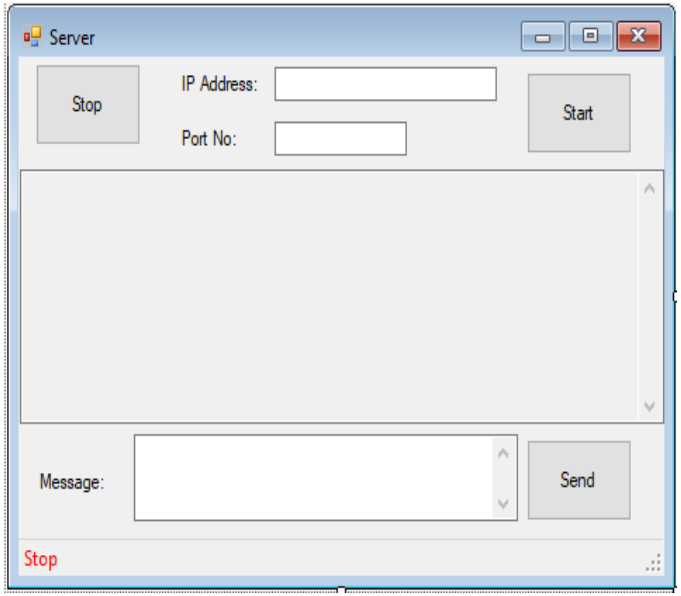
IMPLEMENTATION OF TCP AND UDP

- In the previous section we introduce the difference between the communication using TCP and UDP protocols the client server architecture. and introduce the socket programming.
- We also implement a simple socket application using the UDP protocol in the client server-based systems.
- In this section we implement a simple chat application between the client and server using the TCP protocol and ASYNC callback method.

IMPLEMENTATION OF TCP AND UDP

- Step 1:use the visual studio .NET to create a new windows form project.
- Step 2:Design the following forms for the client and server respectively.





IMPLEMENTATION OF TCP AND UDP

- Step 3:Add the following code to the client form:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Net.Sockets;
using System.Net;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Client
{
    3 references
    public partial class Form1 : Form
    {
        Socket client;
        byte[] buffer;
        1 reference
        public Form1()
        {
            InitializeComponent();

            buffer = new byte[1024];
            IPTxt.Text = GetLocalIP();
            PortTxt.Text = "8000";
        }
```

IMPLEMENTATION OF TCP AND UDP

- Step 3:Add the following code to the client form:

```
string GetLocalIP()
{
    string hostname = Dns.GetHostName();
    IPHostEntry ipHost = Dns.GetHostByName(hostname);
    return ipHost.AddressList[ipHost.AddressList.Length - 1].ToString();
}

2 references
void UpdateGUI(bool isRunning)
{
    StartBtn.Enabled = !isRunning;
    StopBtn.Enabled = isRunning;
    SendBtn.Enabled = isRunning;

    StatusLbl.Text = isRunning ? "Connected" : "Disconnected";
    StatusLbl.ForeColor = isRunning ? Color.Green : Color.Red;


    IPTxt.Enabled = !isRunning;
    PortTxt.Enabled = !isRunning;


}
```

IMPLEMENTATION OF TCP AND UDP

- Step 3:Add the following code to the client form:

```
void ConnectToServer()
{
    IPAddress serverIP;
    if (!IPAddress.TryParse(IPTxt.Text.Trim(), out serverIP))
    {
        MessageBox.Show("Please, insert IP Address in correct format");
        IPTxt.Focus();
        return;
    }
    int serverPortNo;
    if (!int.TryParse(PortTxt.Text.Trim(), out serverPortNo))
    {
        MessageBox.Show("Please, insert Port No in correct format");
        PortTxt.Focus();
        return;
    }
    //Create Socket
    client = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);
    //Server Address
    IPEndPoint serverAddress = new IPEndPoint(serverIP, serverPortNo);
```

IMPLEMENTATION OF TCP AND UDP

- Step 3:Add the following code to the client form:

```
    try
    {
        client.Connect(serverAddress);
        //Waiting for message
        client.BeginReceive(buffer, 0, buffer.Length, SocketFlags.None,
            new AsyncCallback(OnDataReceived), null);
        UpdateGUI(true);
    }
    catch(Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
}
2 references
private void OnDataReceived(IAsyncResult ar)
{
    try
    {
        if(client!=null&&client.Connected)
        {
            //End Receive
            int byteCount = client.EndReceive(a
            if(byteCount==0)
            {
                //Server is down or disconnect
                this.Invoke(new Action(delegate {
                    DisconnectFromServer();
                }));
            }
            else
```

IMPLEMENTATION OF TCP AND UDP

- Step 3:Add the following code to the client form:

```
            else
            {
                string msg = Encoding.ASCII.GetString(buffer, 0, byteCount);
                LogTxt.Invoke(new Action(delegate {
                    LogTxt.AppendText( msg);
                }));
                //Waiting for message
                client.BeginReceive(buffer, 0, buffer.Length, SocketFlags.None,
                    new AsyncCallback(OnDataReceived), null);
            }
        }
    }
    catch
    {
        //Server is down or disconnect
        this.Invoke(new Action(delegate
        {
            DisconnectFromServer();
        }));
    }
}
```

IMPLEMENTATION OF TCP AND UDP

- Step 3:Add the following code to the client form:

```
private void DisconnectFromServer()
{
    if(client!=null&&client.Connected)
    {
        client.Disconnect(false);
        client.Close();
        client = null;

        UpdateGUI(false);
    }
}

1 reference
private void StartBtn_Click(object sender, EventArgs e)
{
    ConnectToServer();
}

1 reference
private void StopBtn_Click(object sender, EventArgs e)
{
    DisconnectFromServer();
}
1 reference
private void Form1_FormClosing(object sender, FormClosingEventArgs e)
{
    DisconnectFromServer();
}
```

IMPLEMENTATION OF TCP AND UDP

- Step 3:Add the following code to the client form:

```
private void SendBtn_Click(object sender, EventArgs e)
{
    if(MsgTxt.Text.Trim()==="")
    {
        MessageBox.Show("Please, insert message first");
        MsgTxt.Focus();
        return;
    }

    byte[] bufferData = Encoding.ASCII.GetBytes("Client: "+MsgTxt.Text.Trim()+Environment.NewLine);

    if(client!=null&&client.Connected)
    {
        client.Send(bufferData);
        LogTxt.AppendText("Client: " + MsgTxt.Text.Trim() + Environment.NewLine);
        MsgTxt.Text = "";
    }
}
```

IMPLEMENTATION OF TCP AND UDP

- Step 3:Add the following code to the server form:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Net.Sockets;
using System.Net;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Server
{
    3 references
    public partial class Form1 : Form
    {

        Socket mainSocket;
        List<Socket> workerSockets = new List<Socket>();
        1 reference
        public Form1()
        {
            InitializeComponent();

            IPTxt.Text = GetLocalIP();
            PortTxt.Text = "8000";
        }
```

IMPLEMENTATION OF TCP AND UDP

- Step 3:Add the following code to the server form:

```
1 reference
string GetLocalIP()
{
    string hostname = Dns.GetHostName();
    IPHostEntry iphost = Dns.GetHostByName(hostname);
    return iphost.AddressList[iphost.AddressList.Length - 1].ToString();
}

2 references
void UpdateGui(bool isRunning)
{
    StartBtn.Enabled = !isRunning;
    StopBtn.Enabled = isRunning;
    SendBtn.Enabled = isRunning;

    StatusLbl.Text = isRunning ? "Start" : "Stop";
    StatusLbl.ForeColor = isRunning ? Color.Green : Color.Red;

    IPTxt.Enabled = !isRunning;
    PortTxt.Enabled = !isRunning;

}
```

IMPLEMENTATION OF TCP AND UDP

- Step 3:Add the following code to the server form:

```
void StartServer()
{
    IPAddress serverIP;
    if(!IPAddress.TryParse(IPTxt.Text.Trim(),out serverIP))
    {
        MessageBox.Show("Please, insert IP Address in correct format");
        IPTxt.Focus();
        return;
    }

    int serverPortNo;
    if (!int.TryParse(PortTxt.Text.Trim(), out serverPortNo))
    {
        MessageBox.Show("Please, insert Port No in correct format");
        PortTxt.Focus();
        return;
    }

    //Create Socket
    mainSocket = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);

    //Server Address
    IPEndPoint serverAddress = new IPEndPoint(serverIP, serverPortNo);

    //Bind
    mainSocket.Bind(serverAddress);

    //Listen & Max
    mainSocket.Listen(5);
```

IMPLEMENTATION OF TCP AND UDP

- Step 3:Add the following code to the server form:

```
        //Waiting for connection
        mainSocket.BeginAccept(new AsyncCallback(OnClientConnect), null);
        UpdateGui(true);
    }
    2 references
    private void OnClientConnect(IAsyncResult ar)
    {
        try
        {
            if(mainSocket!=null)
            {
                //End Accept
                Socket worker = mainSocket.EndAccept(ar);

                //Save worker socket --> database
                workerSockets.Add(worker);

                //Waiting For message
                WaitingForData(worker);

                //Waiting for connection
                mainSocket.BeginAccept(new AsyncCallback(OnClientConnect), null);
            }
        }
        catch //(Exception ex)
        {
              MessageBox.Show(ex.Message);
        }
    }
```

IMPLEMENTATION TO TCP AND UDP

- Step 4:Add the following code to the server form:

```
private void WaitingForData(Socket worker)
{
    if(worker!=null&&worker.Connected)
    {
        SocketPacket socketpacket = new SocketPacket(worker);
        //Waiting for message
        worker.BeginReceive(socketpacket.buffer, 0, socketpacket.buffer.Length,
            SocketFlags.None,
            new AsyncCallback(OnDataReceived), socketpacket);
    }
}
2 references
private void OnDataReceived(IAsyncResult ar)
{
    try
    {
        SocketPacket socketpacket = (SocketPacket)ar.AsyncState;

        if(socketpacket.Socket!=null&&socketpacket.Socket.Connected)
        {
            //Message received
            int bytecount = socketpacket.Socket.EndReceive(ar);

            if(bytecount==0)
            {
                //client is down or disconnected
                socketpacket.Socket.Close();
            }
```

IMPLEMENTATION OF TCP AND UDP

- Step 4:Add the following code to the server form:

```
                else
                {
                    string msg = Encoding.ASCII.GetString(socketpacket.buffer, 0, bytecount);
                    LogTxt.Invoke(new Action(delegate {
                        LogTxt.AppendText(msg);
                    }));

                    //Waiting for message
                    socketpacket.Socket.BeginReceive(socketpacket.buffer, 0, socketpacket.buffer.Length,
                        SocketFlags.None,
                        new AsyncCallback(OnDataReceived), socketpacket);
                }
            }
        }
        catch
        {

        }
    }

    1 reference
    private void StartBtn_Click(object sender, EventArgs e)
    {
        StartServer();
    }

    1 reference
    private void StopBtn_Click(object sender, EventArgs e)
    {
        StopServer();
    }
```

IMPLEMENTATION TO TCP AND UDP

- Step 4:Add the following code to the server form:

```
private void StopServer()
{
    if(mainSocket!=null)
    {
        mainSocket.Close();
        mainSocket = null;

        foreach (Socket worker in workerSockets)
        {
            worker.Close();
        }

        workerSockets.Clear();

        UpdateGui(false);
    }
}
1 reference
private void Form1_FormClosing(object sender, FormClosingEventArgs e)
{
    StopServer();
}
```

IMPLEMENTATION OF TCP AND UDP

```
        private void SendBtn_Click(object sender, EventArgs e)
        {
            SendMessage();
        }
        1 reference
        private void SendMessage()
        {
            if(MsgTxt.Text.Trim()==="")
            {
                MessageBox.Show("Please, Insert message first");
                MsgTxt.Focus();
                return;
            }
            byte[] buffer = Encoding.ASCII.GetBytes("Server: "+MsgTxt.Text.Trim()+Environment.NewLine);

            //Send message to all clients.
            foreach (Socket worker in workerSockets)
            {
                if(worker!=null&&worker.Connected)
                {
                    worker.Send(buffer);
                    LogTxt.AppendText("Server: " + MsgTxt.Text.Trim()+Environment.NewLine);
                    MsgTxt.Text = "";
                }
            }
        }

    }
}
```

IMPLEMENTATION OF TCP AND UDP

Thanks