2022년 IoT기반 스마트 솔루션 개발자 양성과정

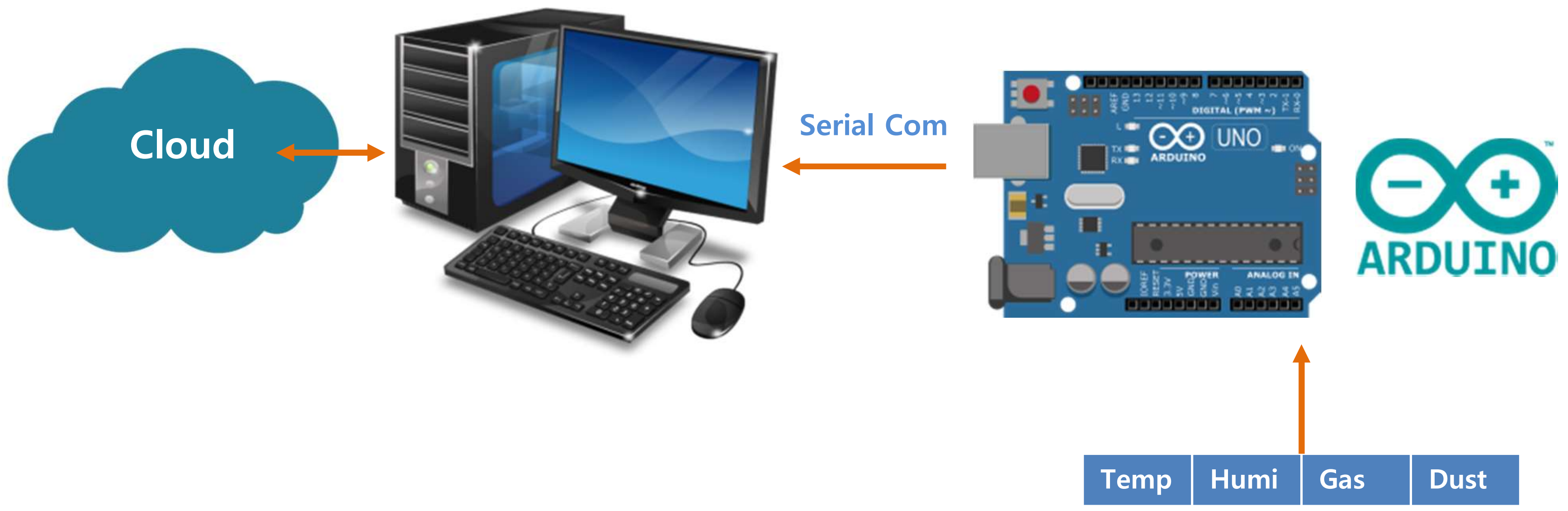
Embedded Application

15-TCP Client

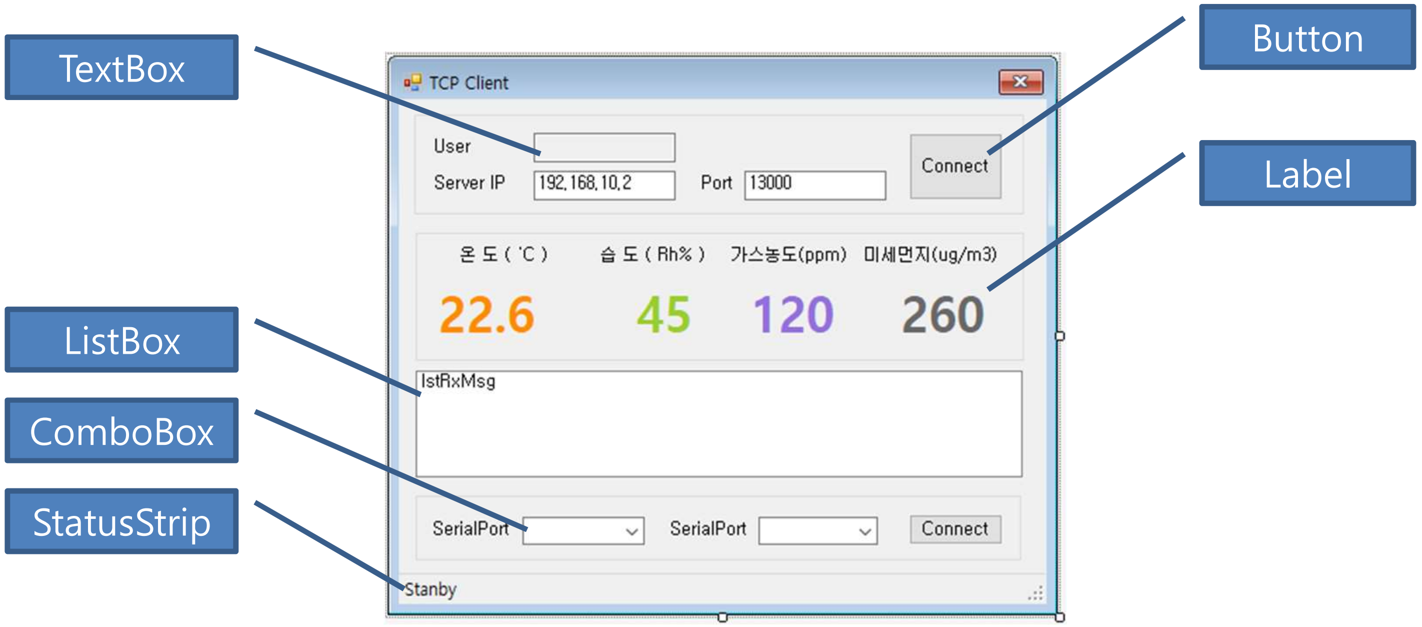
담당 교수 : 윤 종 이
010-9577-1696
ojo1696@naver.com
https://cafe.naver.com/yoons2022

충북대학교 공동훈련센터

Blockdiagram



Form Design


TextBox
ListBox
ComboBox
StatusStrip
Button
Label
TCP Client
User
Server IP
192.168.10.2
Port
13000
Connect
온 도 (˚C)
습 도 (Rh%)
가스농도(ppm)
미세먼지(ug/m3)
22.6
45
120
260
lstRxMsg
SerialPort
SerialPort
Connect
Stanby

Property

Object	Property
txtUser	ReadOnly
txtIPaddress	
txtLocalPort	
btnTcpConnect	

Object	Property
lblTemp	
lblHumi	
lblGas	
lblDust	

Object	Property
lstRxMsg	
cmbSerialPort	
cmbBoardRate	
btnSerial	
lblStatus	

Define

```
using System;
using System.Windows.Forms;
using System.Threading;
using System.Net;
using System.Net.Sockets;
using System.IO;
using System.IO.Ports;
```

```
private int LocalPort = 13000;
private IPAddress IPaddress = IPAddress.Parse("127.0.0.1");
private string HostName = Dns.GetHostName( );

private TcpClient Client;
private Thread ReceiveThread;
private bool Connected = false;

private NetworkStream stream;
private StreamReader Reader;
private StreamWriter Writer;

private delegate void SetTextDelegate(string getString);

SerialPort ComPort = new SerialPort( );

private string dataTemp;
private string dataHumi;
private string dataGas;
Private string dataDust;

public Random rnd = new Random( );
```

Serial Receive

```
public Form1( )    {
     InitializeComponent( );
     ComPort.DataReceived  +=  new SerialDataReceivedEventHandler(DataReceived);
}
private void DataReceived(object sender, System.IO.Ports.SerialDataReceivedEventArgs e)  {
     string rxd = ComPort.ReadTo("₩n");
     this.BeginInvoke(new SetTextDelegate(SerialReceived), new object[ ] { rxd });
}

private void SerialReceived(string inString)   {
     string Head = inString.Substring(0, 1);
     string Data = inString.Substring(1);

     if (Head == "@")  {
          string[ ] PasingData = Data.Split(',');
          dataTemp= PasingData[0];       dataHumi= PasingData[1];        dataGas= PasingData[2];       dataDust= PasingData[3];

          lblTemp.Text = dataTemp;        lblHumi.Text = dataHumi;         lblGas.Text= dataGas;         lblDust.Text= dataDust;

          if (Connected) SendToServer( );
     }
}
```

Send to Server

```
private void SendToServer( )   {
      string msg = "@" + HostName + "," + dataTemp + "," + dataHumi + "," + dataGas + "," + dataDust;
      Writer.WriteLine(msg);
      Writer.Flush( );
}
```

btnTcp

```
private void btnTcpConnect_Click(object sender, EventArgs e
    try  {
                 -------------------------------------------------→
                 }   else  {
                 -------------------------------------------------→
                 }
     }    catch (SocketException ex)   {
           ------→
     }
}
```

```
if (btnTcpConnect.Text == "Connect") {
   IPaddress = IPAddress.Parse(txtIPaddress.Text);

   Client = new TcpClient( );
   Client.Connect(IPaddress, LocalPort);
   Connected = true;
   stream = Client.GetStream( );
   Reader = new StreamReader(stream);
   Writer = new StreamWriter(stream);

   ReceiveThread = new Thread(new ThreadStart(TCP_Receive));
   ReceiveThread.Start( );

   lblStatus.Text = "Connected to Server!";
   btnTcpConnect.Text = "Close";
```

```
Connected = false;
if (ReceiveThread != null) ReceiveThread.Abort(  );
if (Reader != null) Reader.Close( );
if (Writer != null) Writer.Close( );
if (Client != null) Client.Close( );

lblStatus.Text = "Closed to Server!";
btnTcpConnect.Text = "Connect";
```

```
MessageBox.Show(ex.Message.ToString(), "TCP Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
```

TCP_Receive

```
private void TCP_Receive( )  {
   try  {
      while (Connected)  {
           Thread.Sleep(1);

           if (stream.CanRead)  {
                string strReceived = Reader.ReadLine( );

                if (strReceived.Length > 0) {
                     strReceived = "Receive : " + strReceived;
                     this.BeginInvoke(new SetTextDelegate(TCPmsgReceive), new object[ ] { strReceived });
                }
           }
      }
   }

   ----→
```

```
catch(Exception ex)  {
    lblStatus.Text = ex.Message.ToString( );
    Connected = false;
    if (ReceiveThread != null) ReceiveThread.Abort();
    if (Reader != null) Reader.Close();
    if (Writer != null) Writer.Close();
    if (Client != null) Client.Close();
    btnTcpConnect.Text = "Connect";
}
}
```

TCPmsgReceive

```
private void TCPmsgReceive(string msg)   {
    lstRxMsg.Items.Add( DateTime.Now.ToString("HH:mm:ss ") + msg);
    if (lstRxMsg.Items.Count > 10) lstRxMsg.Items.RemoveAt(0);
    lstRxMsg.SelectedIndex = lstRxMsg.Items.Count - 1;
}
```

Form1_Load

```
private void Form1_Load(object sender, EventArgs e)  {
            this.Text = "TCP Client - " + HostName;
            txtUser.Text = HostName;

            cmbSerialPort.Items.Clear( );
            var portName = System.IO.Ports.SerialPort.GetPortNames( );
            cmbSerialPort.Items.AddRange(portName);
            cmbSerialPort.SelectedIndex = cmbSerialPort.Items.Count - 1;

            cmbBoardRate.Items.Clear( );
            cmbBoardRate.Items.Add("9600");
            cmbBoardRate.Items.Add("19200");
            cmbBoardRate.Items.Add("57600");
            cmbBoardRate.Items.Add("115200");
            cmbBoardRate.SelectedIndex = 0;
}
```

FormClosing

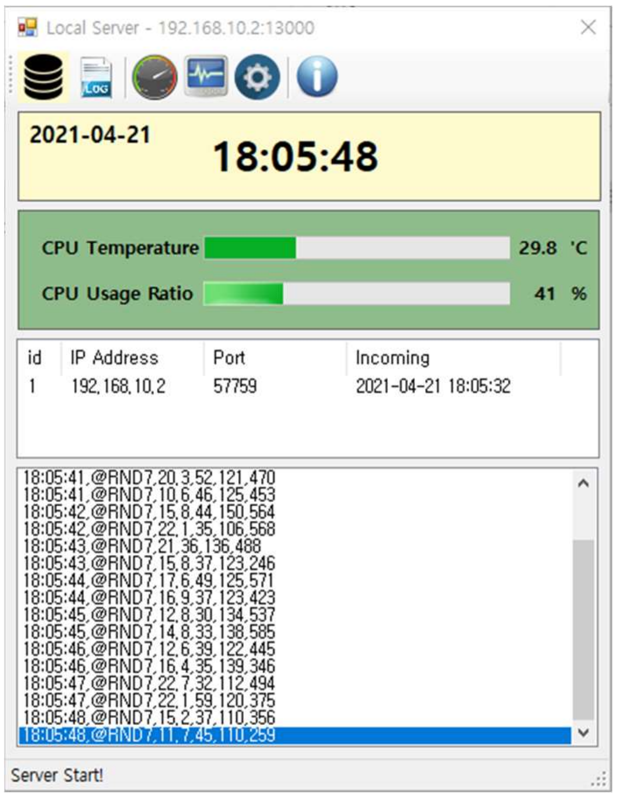
```
private void Form1_FormClosing(object sender, FormClosingEventArgs e)  {
            Connected = false;
            if (ReceiveThread != null) ReceiveThread.Abort( );
            if (Reader != null) Reader.Close( );
            if (Writer != null) Writer.Close( );
            if (Client != null) Client.Close( );
}
```

btnSerial –Random Demo

```
private void btnSerial_Click(object sender, EventArgs e)  {
    if (btnSerial.Text == "Connect")   {
          timer1.Start( );
          btnSerial.Text = "Close";
    }  else  {
          timer1.Stop( );
          btnSerial.Text = "Connect";
    }

}
```

```
private void timer1_Tick(object sender, EventArgs e)  {
      dataTemp = (rnd.Next(100, 250) / 10.0).ToString( );
      dataHumi = rnd.Next(30, 60).ToString( );
      dataGas = rnd.Next(100, 160).ToString( );
      dataDust = rnd.Next(200, 600).ToString( );

       lblTemp.Text = dataTemp;
       lblHumi.Text = dataHumi;
       lblGas.Text = dataGas;
       lblDust.Text = dataDust;

       if (Connected) SendToServer( );
}
```

Demo Run


Local Server - 192.168.10.2:13000

2021-04-21 18:05:48

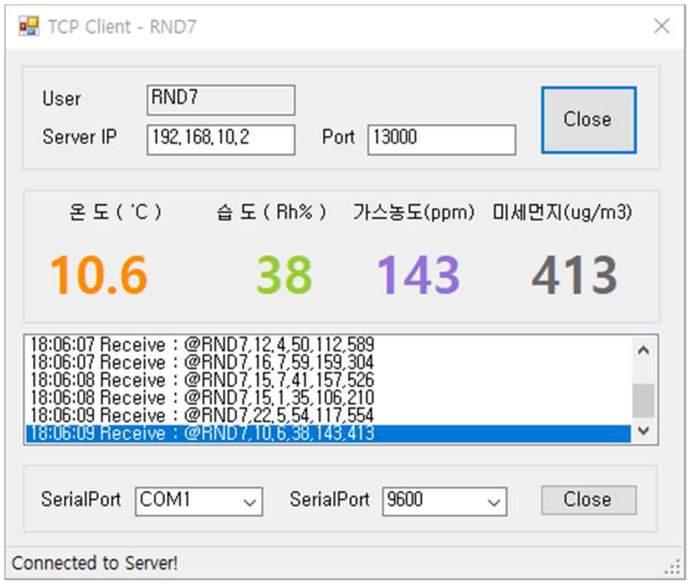
CPU Temperature 29.8 'C

CPU Usage Ratio 41 %

id	IP Address	Port	Incoming
1	192.168.10.2	57759	2021-04-21 18:05:32

```
18:05:41,@RND7,20,3,52,121,470
18:05:41,@RND7,10,6,46,125,453
18:05:42,@RND7,15,8,44,150,564
18:05:42,@RND7,22,1,35,106,568
18:05:43,@RND7,21,36,136,488
18:05:43,@RND7,15,8,37,123,246
18:05:44,@RND7,17,6,49,125,571
18:05:44,@RND7,16,9,37,123,423
18:05:45,@RND7,12,8,30,134,537
18:05:45,@RND7,14,8,33,138,585
18:05:46,@RND7,12,6,39,122,445
18:05:46,@RND7,16,4,35,139,346
18:05:47,@RND7,22,7,32,112,494
18:05:47,@RND7,22,1,59,120,375
18:05:48,@RND7,15,2,37,110,356
18:05:48,@RND7,11,7,45,110,259
```

Server Start!


TCP Client - RND7

User RND7

Server IP 192.168.10.2 Port 13000

온 도 ('C)	습 도 (Rh%)	가스농도(ppm)	미세먼지(ug/m3)
10.6	38	143	413

```
18:06:07 Receive : @RND7,12,4,50,112,589
18:06:07 Receive : @RND7,16,7,59,159,304
18:06:08 Receive : @RND7,15,7,41,157,526
18:06:08 Receive : @RND7,15,1,35,106,210
18:06:09 Receive : @RND7,22,5,54,117,554
18:06:09 Receive : @RND7,10,6,38,143,413
```

SerialPort COM1 SerialPort 9600

Connected to Server!

btnSerial Open / Close

```
private void btnSerial_Click(object sender, EventArgs e)   {
    if (btnSerial.Text == "Connect")   {
        ComPort.PortName = cmbSerialPort.Text;
        ComPort.BaudRate = Convert.ToInt32(cmbBoardRate.Text);
        ComPort.DataBits = 8;
        ComPort.Parity = Parity.None;
        ComPort.StopBits = StopBits.One;
        ComPort.Handshake = Handshake.None;
        ComPort.Open( );
        ComPort.DiscardInBuffer( );
        btnSerial.Text = "Close";
    }  else  {
        ComPort.Close( );
        btnSerial.Text = "Connect";
    }
}
```