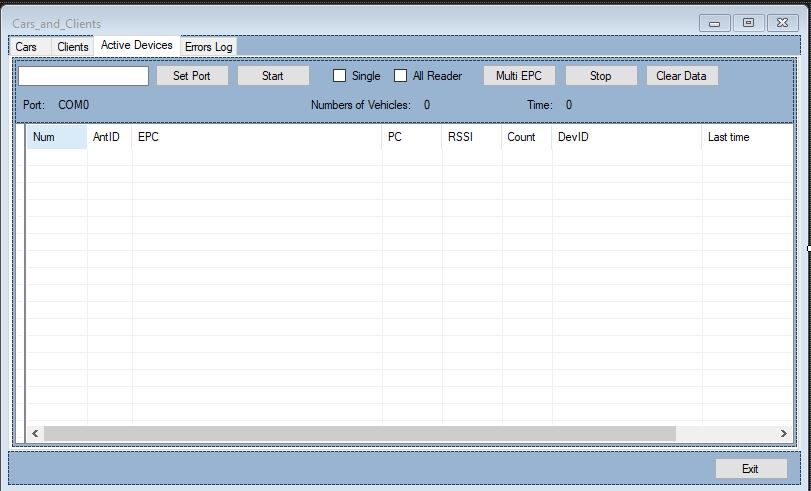
**Car Management Codes and Forms**

Display data from the device-Form:



Codes:

using System;

using System.Collections.Generic;

using System.Data;

using System.Windows.Forms;

using System.Data.SqlClient;

using NetFrame.Net.TCP.Sock.Asynchronous;

using System.Threading;

using System.IO.Ports;

using System.IO;

using System.Reflection;

using System.Text;

public partial class Cars\_and\_Clients : Form

{

private const int cash= 100;

private long totalnum1 = 0x00;

private long totalnum2 = 0x00;

private long totaltime = 0x00;

private const int listView\_md\_epc\_Num = 0;

private const int listView\_md\_epc\_AntID = 1;

private const int listView\_md\_epc\_EPC = 2;

private const int listView\_md\_epc\_PC = 3;

private const int listView\_md\_epc\_Rssi = 4;

private const int listView\_md\_epc\_Count = 5;

private const int listView\_md\_epc\_IP = 6;

private const int listView\_md\_epc\_Last\_Time = 7;

private const int listView\_md\_epc\_Direction = 8;

private const int listView\_md\_State = 3;

private volatile List<\_epc\_t> epcs\_list = new List<\_epc\_t>(1000);

private string portname = "";

private int baudRate = 230400;

private int dataBits = 8;

private Parity parity = Parity.None;

private StopBits stopbits = StopBits.One;

string error;

List<AsyncSocketState> clients;

bool serialisstart = false;

bool serverisstart = false;

public Cars\_and\_Clients()

{

InitializeComponent();

Control.CheckForIllegalCrossThreadCalls = false;

ReaderControllor.cmd.MultiEPC\_Event += ShowEPC;

this.listView\_md\_addr.Columns.Add("Num", 30, HorizontalAlignment.Left);

this.listView\_md\_addr.Columns.Add("IP", 100, HorizontalAlignment.Left);

this.listView\_md\_addr.Columns.Add("Port", 50, HorizontalAlignment.Left);

this.listView\_md\_addr.Columns.Add("ID", 50, HorizontalAlignment.Left);

this.listView\_md\_addr.Columns.Add("State", 50, HorizontalAlignment.Left);

this.listView\_md\_addr.GridLines = true;

this.listView\_md\_addr.FullRowSelect = true;

this.listView\_md\_addr.MultiSelect = false;

}

// Sets the port to be used

private void btnSet\_Click(object sender, EventArgs e)

{

try

{

PortConfig SerialPortForm = new PortConfig();

SerialPortForm.ShowDialog();

if (SerialPortForm.result == true)

{

textBox1.Text = SerialPortForm.PortName;

portname = textBox1.Text;

baudRate = SerialPortForm.BuadRate;

dataBits = SerialPortForm.dataBits;

parity = SerialPortForm.parity;

stopbits = SerialPortForm.stopbits;

}

}

catch(Exception ex)

{

new LogWriter(ex);

}

}

private Reader ReaderControllor = new Reader();

private AsyncSocketState currentclient;

// *Starts the port*

private void btnStartPort\_Click(object sender, EventArgs e)

{

if (btnStartPort.Text == "Start")

{

portname = textBox1.Text;

try

{

ReaderControllor.ComStart(portname, baudRate, dataBits, parity, stopbits);

if (timer\_md\_query\_Tick.Enabled == false)

{

timer\_md\_query\_Tick.Enabled = true;

}

serialisstart = true;

lblPort.Text = textBox1.Text;

btnStartPort.Text = "Stop";

}

catch (Exception ex)

{

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

new LogWriter(ex);

}

}

else

{

serialisstart = false;

ReaderControllor.SerialPortClose();

if (serverisstart == false && serialisstart == false && timer\_md\_query\_Tick.Enabled == true)

{

timer\_md\_query\_Tick.Enabled = false;

}

btnStartPort.Text = "Start";

}

}

private void btnMultiEPC\_Click(object sender, EventArgs e)

{

try

{

if (checkBoxMulti.Checked == true)

{

if (checkBoxSingle.Checked == true)

{

ReaderControllor.SingleEPC();

}

else

{

ReaderControllor.SatrtMultiEPC();

}

}

else

{

if (checkBoxSingle.Checked == true)

{

ReaderControllor.SingleEPC(currentclient);

}

else

{

ReaderControllor.SatrtMultiEPC(currentclient);

}

}

btnMultiEPC.Enabled = false;

}

catch (Exception ex)

{

new LogWriter(ex);

}

}

//Stops the multiEPC

private void btnStop\_Click(object sender, EventArgs e)

{

try

{

if (checkBoxMulti.Checked == true)

{

if (checkBoxSingle.Checked == true)

{

;

}

else

{

ReaderControllor.StopMultiEPC();

}

}

else

{

if (checkBoxSingle.Checked == true)

{

;

}

else

{

ReaderControllor.StopMultiEPC(currentclient);

}

}

btnMultiEPC.Enabled = true;

}

catch (Exception ex)

{

new LogWriter(ex);

}

}