How to Use ParkingSystem Library

1.Why library?

1) Ease you to communicate with the DC(Data Collect) device of parking system.
2) Ease you to parse the packet. You do not need to care about the format of packet.
3) Ease you to build your own parking system application on PC.
4) Support two communication modes: Serial port & TCP .

2.Information of the library

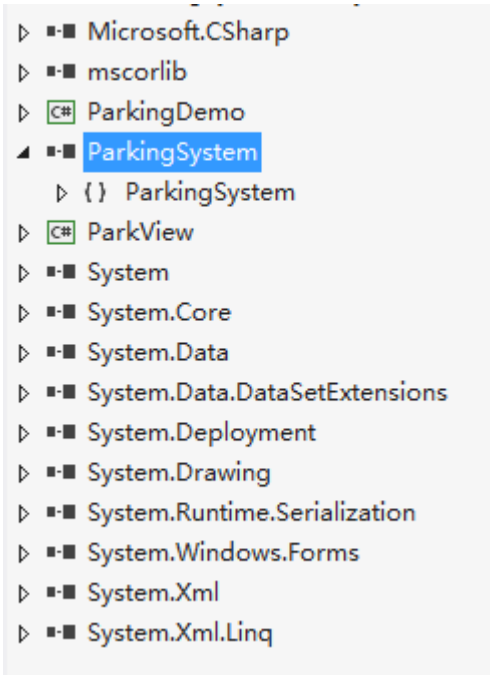
DLL file: ParkingSystem.dll

Built by c#.net

Runtime version: v4.5

ParkingSystem.dll version: v1.00

So far, only c# language is supported.



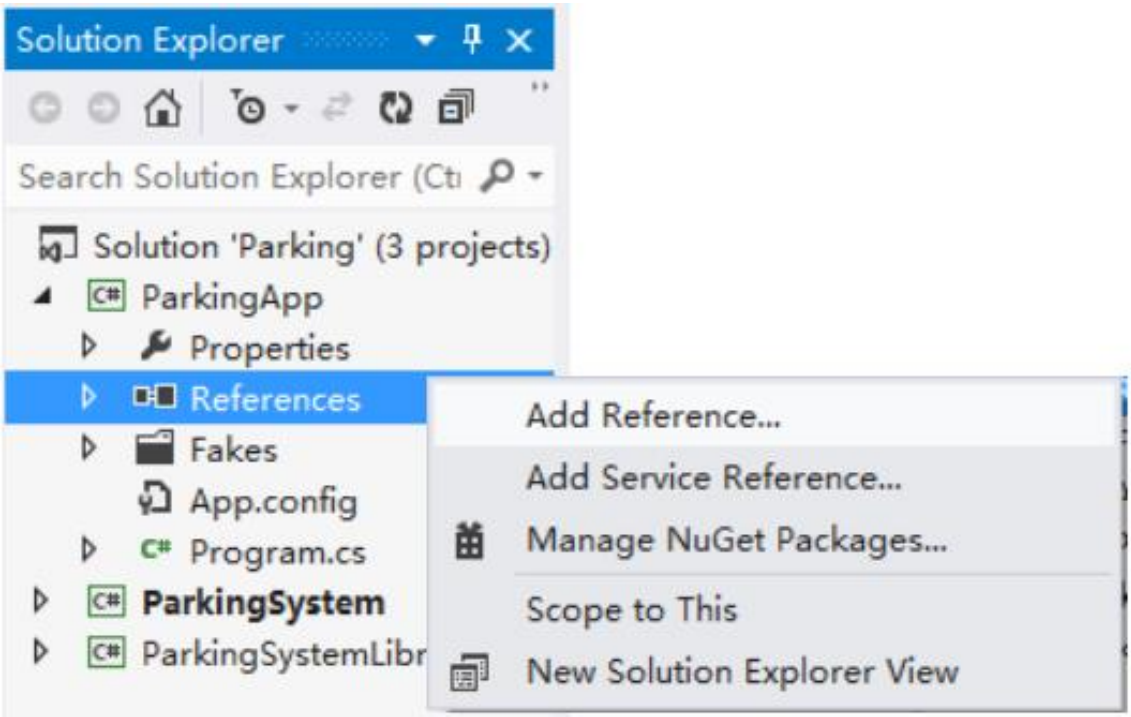
3.Create a new project

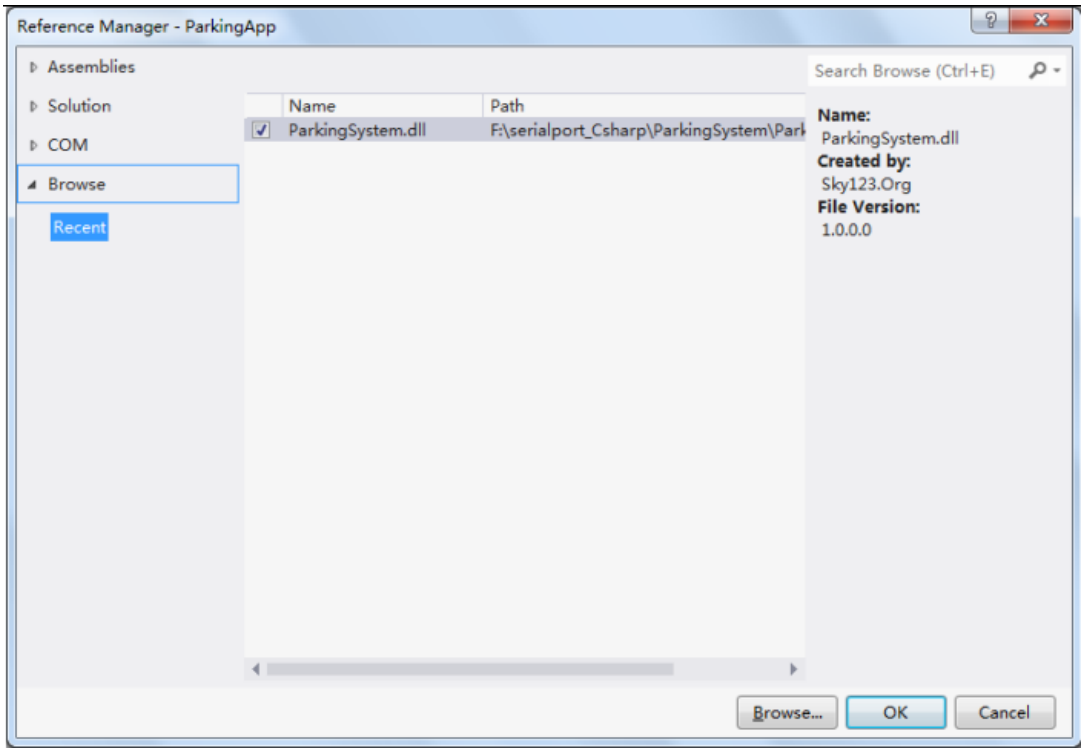
3.1.step 1#: Create a new C#.net project

Make sure select ".NET Framework 4.5" for new project.

3.2.step 2#: Add Reference

Right click on the "References", select "Add Reference…" , in the "Reference Manager - xxx" dialog click "Browse" to
select the "ParkingSystem.dll" file.





3.3.step 3#: Add source(Serial port example)

1) If you are using serial port to communicate with device. It is necessary to use ***ParkingSerialPort*** calss. In the source file, at the "using" area, type the following "using" words:

```
using ParkingSystem; // Must add this namespace
```

2) In the appropriate place of the program, you must add a ***ParkingSerialPort*** object as the following code:

```
public static ParkingSystem.ParkingSerialPort sp = new ParkingSerialPort();
```

3) Then, call the constructor to initialize the ***ParkingSerialPort*** object :

```
sp = new ParkingSerialPort ("COM4", 115200);
```

4) Add the event handler function for received packets as following:

```
ParkingOriginalPacket.EvProcessReceivedPacket += sp_ProcessReceivedPacket;
```

5) Use the ***Open***() method to open the serial prot:

sp.Open();

This is recommended to put these code in try-catch block. Here is a code example:

```
try
{
    sp = new ParkingSerialPort("COM4", 115200);
    sp.Open();
}
catch (IOException ipexp)
{
    Console.WriteLine(ipexp.Message);
```

```
    }
```

6) Implement event handler for received packet. These event handler function will be automatically called when the corresponding packet is received. Here is a simple example of handling SensorHBeat packet:

```
        /// <summary>
        /// <remarks>Process Received Packet</remarks>
        /// </summary>
        /// <param name="pk">Received Packet</param>
        private void sp_ProcessReceivedPacket(baseReceivedPacket pk)
        {
            try
            {
                byte revType = Convert.ToByte(pk.type_ver >> 8);
                string wpsdid = "";
                string WDCid = "";
                string RSSI = "";
                byte carState = 0;
                string voltage = "";
                string hardVer = "";
                string softVer = "";
                string deviceName = "";
                string madeDate = "";
                string hbPeriod = "";
                this.Invoke((EventHandler)delegate
                {
                    #region Senser Heart Beat
                    if (pk is SensorHBeat)
                       {
                           SensorHBeat hb = (SensorHBeat)pk;
                            reshow(hb.recRawData,true);
                            wpsdid = (hb.WPSD_ID).ToString("X2").PadLeft(8, '0');
                            WDCid = (hb.WDC_ID).ToString("X2").PadLeft(8, '0');
                            softVer = "v" +
int.Parse(hb.APP_VER.ToString("X2").Substring(0, 1)).ToString() + "." +
int.Parse(hb.APP_VER.ToString("X2").Substring(1, 1)).ToString().PadLeft(2, '0');
                            hardVer = ((int)(hb.HARD_VER) + 10).ToString();
                            hardVer = "v" + hardVer.Substring(0, 1) + "." +
hardVer.Substring(1, 1);
                            voltage = (Math.Round((decimal)hb.VOLT / 10,
2)).ToString()+"V";
                            madeDate = hb.DATE_YEAR.ToString() + "-" +
hb.DATE_MONTH.ToString().PadLeft(2, '0') + "-" + hb.DATE_DAY.ToString().PadLeft(2, '0')
+ " " + hb.DATE_HOUR.ToString().PadLeft(2, '0') + " : " +
hb.DATE_MINUTE.ToString().PadLeft(2, '0');
```

```
                            RSSI = ((Int16)hb.RSSI - 30).ToString();
                            hbPeriod = hb.HB_PERIOD.ToString();
                            deviceName = GetDevName(hb.DEV_TYPE);
                            carState = hb.CAR_STATE;
                            if (carState == 0x01)
                            {
                                richTextBox1.AppendText("wpsd id:" + wpsdid + "\nsoft
Ver:" + softVer + "\nhard Ver:" + hardVer + "\nvoltage:" + voltage + "\nMade Date:" +
madeDate + "\nRSSI:" + RSSI + "\ncar State:Have Car\n");
                            }
                            else
                            {
                                richTextBox1.AppendText("wpsd id:" + wpsdid + "\nsoft
Ver:" + softVer + "\nhard Ver:" + hardVer + "\nvoltage:" + voltage + "\nMade Date:" +
madeDate + "\nRSSI:" + RSSI + "\ncar State:No Car\n");
                            }

                        }
                          #endregion
                    });
                }
                catch (Exception ex)
                {
                    Console.WriteLine(ex);
                }
                  }
```

Of course, you can implement more complicated handler function than this one of course.

3.4.step 3#: Add source(TCP example)

1) If you are using TCP to communicate with device. In the source file, at the “using” area, type the following “using”words:

```
using ParkingSystem; // Must add this namespace
```

2) In the appropriate place of the program, you must create a server first like following code:

```
IPAddress ip = new IPAddress(new byte[] { 127, 0, 0, 1 });
TcpListener listener = new TcpListener(ip, 6000);
listener.Start(); // Start to listening
```

3) Create ***ParkingRemoteTCP*** object using constructor function, here is an example that theoretically can accepts infinite connection from remote:

```
while (true) // Here give an example which can theoretically accepts infinite
connection from remote.
    {
        TcpClient client = listener.AcceptTcpClient();// Stop here, wait to accept a
connection;
        ParkingRemoteTCP wapper = new ParkingRemoteTCP(client);
```

```
}
```

4) Add the event handler function for received packets as following:

```
ParkingOriginalPacket.EvProcessReceivedPacket += sp_ProcessReceivedPacket;
```

5) Implement event handler for received packet. These event handler function will be automatically called when the corresponding packet is received.this please refer to **3.3. Add source(Serial port example) step 6.**