```
1 /*----*\
2
   * Author : Salvi Cyril
 3
   * Date : 7th juny 2017
 4
   * Diploma : RaspiHome
   * Classroom : T.IS-E2B
 5
 6
    * Description:
 7
8
          RaspiHomeSenseHAT is a program who use a
9
        Sense HAT, it's an electronic card who can be
10
        mesured value with sensor. This program use
11
       the Sense HAT to mesure the temperature, the
12
        humidity and the pressure.
13 \*-----*/
14
15 using System;
16 using Emmellsoft.IoT.Rpi.SenseHat;
17 using Windows.UI;
18 using Windows.UI.Xaml;
19
20 namespace RaspiHomeSenseHAT
21 {
22
       public class ModelSenseHAT
23
24
          #region Fields
25
          #region Constants
          #endregion
26
27
28
          #region Variables
29
          private ViewSenseHAT _vSenseHAT;
          private CommunicationWithServer _comWithServer;
30
31
32
          // Sense HAT librairy
          private ISenseHat _senseHat;
33
34
          private ISenseHatDisplay senseHatDisplay;
          private SenseHatData _data;
35
36
          // Set default color matrix to OFF
37
38
          private Color _uiColor = Color.FromArgb(0, 0, 0, 0);
39
          #endregion
          #endregion
40
41
          #region Properties
42
          public ViewSenseHAT VSenseHAT
43
           {
44
              get
45
46
              {
47
                  return _vSenseHAT;
48
              }
49
50
              set
51
              {
52
                  vSenseHAT = value;
53
              }
54
          }
55
56
          public SenseHatData Data
```

```
\dots \texttt{de} \\ \texttt{RaspiHomeSenseHAT} \\ \texttt{RaspiHomeSenseHAT} \\ \texttt{ModelSenseHAT.cs}
```

```
57
 58
                 get
 59
                 {
 60
                     return _data;
 61
                 }
 62
 63
                 set
 64
                 {
                     _data = value;
 65
 66
                 }
             }
 67
 68
 69
             public CommunicationWithServer ComWithServer
 70
 71
                 get
 72
                 {
 73
                     return _comWithServer;
 74
                 }
 75
 76
                 set
 77
                 {
 78
                     _comWithServer = value;
 79
 80
             }
             #endregion
 81
 82
 83
             #region Constructors
 84
             /// <summary>
             /// Constructor: Initializer
 85
             /// </summary>
 86
             /// <param name="paramView"></param>
 87
 88
             public ModelSenseHAT(ViewSenseHAT paramView)
 89
             {
 90
                 // Communication like Model-View
 91
                 this.VSenseHAT = paramView;
 92
                 // Initilize the Sense HAT (don't need to be initialized before
 93
                   the communication start because it's only a sensor)
 94
                 InitializeSenseHat();
 95
                 // Initilize the communication with the server
 96
 97
                 this.ComWithServer = new CommunicationWithServer(this);
 98
             #endregion
 99
100
             #region Methods
101
102
             /// <summary>
103
             /// Initialize the Sense HAT
104
             /// </summary>
             public async void InitializeSenseHat()
105
106
107
                 this. senseHat = await SenseHatFactory.GetSenseHat();
108
                 this._senseHatDisplay = this._senseHat.Display;
109
                 this._senseHatDisplay.Fill(_uiColor);
110
```

```
...de\RaspiHomeSenseHAT\RaspiHomeSenseHAT\ModelSenseHAT.cs
```

146

```
111
                 SetValue();
112
             }
113
114
             /// <summary>
115
             /// Set the value get with sensor
116
             /// </summary>
             public void SetValue()
117
118
119
                 // Update values
120
                 this._senseHat.Sensors.HumiditySensor.Update();
                 this. senseHat.Sensors.PressureSensor.Update();
121
122
                 this. senseHatDisplay.Update();
123
124
                 // Set values
                 this.Data = new SenseHatData();
125
126
                 this.Data.Temperature = this._senseHat.Sensors.Temperature;
127
                 this.Data.Humidity = this._senseHat.Sensors.Humidity;
128
                 this.Data.Pressure = this._senseHat.Sensors.Pressure;
129
             }
130
131
             /// <summary>
             /// Send the values with a special format: "TEMP=x; HUMI=y; PRES=z" \,
132
133
             /// /// Values are rounded
134
             /// </summary>
135
             /// <returns></returns>
136
             public string SendValues()
137
             {
                 // Update values of sensors
138
139
                 SetValue();
140
                 return "TEMP=" + Math.Round((decimal)this.Data.Temperature) + ";" →
141
                   + "HUMI=" + Math.Round((decimal)this.Data.Humidity) + ";" +
                   "PRES=" + Math.Round((decimal)this.Data.Pressure);
142
             }
143
             #endregion
144
         }
145 }
```