```
1 /*----*\
 2
   * Author : Salvi Cyril
   * Date : 8th juny 2017
 3
 4
   * Diploma : RaspiHome
 5
   * Classroom : T.IS-E2B
 6
   * Description:
 7
8
           RaspiHomeTabletWindows is a program
9
        compatible with the Windows tablet. It's a
10
        program that can be use as tactil graphic
11
       interface to order the component linked with
12
        the other Raspberry Pi.
13 \*-----*/
14
15 using System;
16 using System.Collections.Generic;
17 using System.Linq;
18 using System.Threading.Tasks;
19 using Windows.Networking;
20 using Windows.Networking.Sockets;
21 using Windows.Storage.Streams;
22 using Windows.UI.Xaml;
23
24 namespace RaspiHomeTabletWindows
25 {
       public class CommunicationWithServer
26
27
       {
28
           #region Fields
29
           #region Constants
           // Default information to connect on the server
30
           private const int PORT = 54565;
31
32
           //// Need to be changed fo each configuration
           private const string IPSERVER = "10.134.97.117";// "192.168.2.8";
33
34
           private const string FORMATSTRING = "IPRasp={0};Location=
35
             {1};Component={2}";
           private const string COMMUNICATIONSEPARATOR = "@";
36
37
38
           // Important need to be changed if it's another room!
39
           private const string LOCATION = "Salon";
           private const string COMPONENT = "Tablet";
40
           private const string RPINAME = "Tablet " + LOCATION;
41
42
           private const int MESSAGE_FULL_LENGHT = 512;
43
44
           #endregion
45
46
           #region Variables
47
           private StreamSocket _socket = new StreamSocket();
48
           private StreamSocketListener listener = new StreamSocketListener();
49
           private List<StreamSocket> _connections = new List<StreamSocket>();
           private bool _isConnected = false;
50
           private bool _connecting = false;
51
52
53
           private Windows.Storage.ApplicationDataContainer localSettings =
54
       Windows.Storage.ApplicationData.Current.LocalSettings;
55
```

```
56
             private string messageCommand = "";
 57
 58
             private string _nameButtonClicked = "";
 59
             DispatcherTimer _dTimer = null;
 60
 61
             #endregion
 62
             #endregion
 63
 64
             #region Properties
 65
             public StreamSocket Socket
 66
 67
                  get
 68
 69
                  {
 70
                      return _socket;
 71
                  }
 72
 73
                  set
 74
                  {
 75
                      _socket = value;
 76
                  }
 77
             }
 78
 79
             public StreamSocketListener Listener
 80
             {
 81
                  get
 82
                  {
 83
                      return _listener;
 84
                  }
 85
 86
                  set
 87
                  {
 88
                      _listener = value;
 89
                  }
 90
             }
 91
             public List<StreamSocket> Connections
 92
 93
 94
                  get
 95
                  {
 96
                      return _connections;
 97
                  }
 98
 99
                  set
100
                  {
                      _connections = value;
101
102
                  }
             }
103
104
105
             public bool IsConnected
106
107
                  get
108
                  {
109
                      return _isConnected;
110
                  }
111
```

```
... ndows \verb|\RaspiHomeTabletWindows \verb|\CommunicationWithServer.cs| \\
```

```
3
```

```
112
                 set
113
                 {
114
                      _isConnected = value;
115
                 }
             }
116
117
118
             public bool Connecting
119
                 get
120
121
                 {
122
                      return _connecting;
123
                 }
124
125
                 set
126
                 {
                      _connecting = value;
127
128
                 }
129
             }
130
131
             public string MessageCommand
132
133
                 get
134
                 {
135
                      return _messageCommand;
136
                 }
137
138
                 set
139
140
                      _messageCommand = value;
141
             }
142
143
144
             public string NameButtonClicked
145
146
                 get
147
                 {
148
                      return _nameButtonClicked;
149
                 }
150
151
                 set
152
                 {
153
                      nameButtonClicked = value;
                 }
154
155
             #endregion
156
157
158
             #region Constructors
159
             /// <summary>
160
             /// Constructor: Initializer
161
             /// </summary>
             public CommunicationWithServer()
162
163
164
                 Connect();
165
                 this._dTimer = new DispatcherTimer();
166
167
                 this._dTimer.Interval = new TimeSpan(10);
```

```
...ndows\RaspiHomeTabletWindows\CommunicationWithServer.cs
```

```
Δ
```

```
168
                  this. dTimer.Tick += dTimer Tick;
169
170
                 this._dTimer.Start();
             }
171
172
             #endregion
173
174
             #region Events
175
             private void _dTimer_Tick(object sender, object e)
176
177
                 if (localSettings.Values["SendMessageToServer"] != null)
178
                 {
179
                      var messageToSend = localSettings.Values
                        ["SendMessageToServer"];
180
                      this.SendCommandToServer(messageToSend.ToString());
                      localSettings.Values.Remove("SendMessageToServer");
181
182
                 }
183
             }
184
             #endregion
185
             #region Methods
186
187
             /// <summary>
             \ensuremath{///} Connect the raspberry to the server
188
189
             /// </summary>
190
             private async void Connect()
191
             {
192
                 try
193
                 {
194
                      this.Connecting = true;
195
                      await this.Socket.ConnectAsync(new HostName(IPSERVER),
                        PORT.ToString());
196
                      SendForInitialize();
197
                      this.Connecting = false;
198
                      this.IsConnected = true;
199
                     WaitForData(this.Socket);
200
                 }
201
                 catch (Exception)
202
203
                 {
204
                      this.Connecting = false;
205
                      this.IsConnected = false;
206
                 }
             }
207
208
209
             /// <summary>
210
             /// Listen the traffic on the port
             /// </summary>
211
212
             private async void Listen()
213
214
                 this.Listener.ConnectionReceived += listenerConnectionReceived;
215
                 await this.Listener.BindServiceNameAsync(PORT.ToString());
             }
216
217
218
             void listenerConnectionReceived(StreamSocketListener sender,
               StreamSocketListenerConnectionReceivedEventArgs args)
219
             {
220
                 this.Connections.Add(args.Socket);
```

```
...ndows\RaspiHomeTabletWindows\CommunicationWithServer.cs
```

```
5
```

```
221
222
                 WaitForData(args.Socket);
223
            }
224
225
            /// <summary>
226
            /// Send the message in input to output
            /// </summary>
227
228
            /// <param name="socket"></param>
229
            /// <param name="message"></param>
230
            private async void SendMessage(StreamSocket socket, string message)
231
                 DataWriter dataWriter = new DataWriter(socket.OutputStream);
232
                 var len = dataWriter.MeasureString(message); // Gets the UTF-8
233
                   string length.
234
                 dataWriter.WriteInt32((int)len);
235
                 dataWriter.WriteString(message);
236
                 var ret = await dataWriter.StoreAsync();
237
                 dataWriter.DetachStream();
238
            }
239
            /// <summary>
240
            /// Send to initialize the raspberry to the server
241
242
            /// </summary>
            private void SendForInitialize()
243
244
            {
                 SendMessage(this.Socket, string.Format(COMMUNICATIONSEPARATOR +
245
                   RPINAME + COMMUNICATIONSEPARATOR + "Connection:" + FORMATSTRING, →
                    GetHostName(), LOCATION, COMPONENT));
246
            }
247
            /// <summary>
248
249
            /// Send the command to the server
250
            /// </summary>
251
            public void SendCommandToServer(string message)
252
             {
                 SendMessage(this.Socket, COMMUNICATIONSEPARATOR + "Send:" +
253
                   message);
254
                 this.MessageCommand = message;
255
             }
256
            /// <summary>
257
            /// Wait data readed if exist
258
259
            /// </summary>
260
            /// <param name="socket"></param>
261
            private async void WaitForData(StreamSocket socket)
262
             {
263
                 DataReader dataReader = new DataReader(socket.InputStream);
264
                 dataReader.InputStreamOptions = InputStreamOptions.Partial;
265
                 var messageLenght = dataReader.UnconsumedBufferLength;
266
                 uint stringBytes = messageLenght;
267
268
                 try
269
                 {
270
                     // Read modification in the stream
                     stringBytes = await dataReader.LoadAsync(MESSAGE_FULL_LENGHT);
271
272
```

```
... ndows \verb|\RaspiHomeTabletWindows \verb|\CommunicationWithServer.cs| \\
```

```
6
```

```
273
                     // read message
274
                     string messageRead = dataReader.ReadString(stringBytes);
275
276
                     await Task.Delay(TimeSpan.FromMilliseconds(200));
277
                     // Store value
278
                     localSettings.Values["ReceiveMessageFromServer"] =
                       messageRead;
                 }
279
280
                 catch (Exception e)
281
                 {
282
                     string output = e.Message;
283
284
                     if (messageLenght < 1)</pre>
285
                         return;
                 }
286
287
288
                 WaitForData(socket);
289
             }
290
291
             /// <summary>
292
             /// Get the ip of the raspberry
293
             /// </summary>
             /// <returns>return a string like 192.168.1.2</returns>
294
295
             public string GetHostName()
296
             {
297
                 List<string> IpAddress = new List<string>();
298
                 var Hosts =
                   Windows.Networking.Connectivity.NetworkInformation.GetHostNames →
                   ().ToList();
299
                 foreach (var Host in Hosts)
300
                 {
                     string IP = Host.DisplayName;
301
302
                     IpAddress.Add(IP);
303
304
                 return IpAddress.Last();
305
             }
306
             #endregion
307
         }
308 }
309
```