```
1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Linq;
7  using System.Text;
8  using System.Threading.Tasks;
9  using System.IO.Ports;
10 using System.IO;
11 using System.Windows.Forms;
12
13 namespace _1811C_Ticketing
14 {
15
16     public partial class Form1 : Form
17     {
18
19         public static int BaudRate, Data_bits;
20         public static string Parity,Handshake, Stop_Bits, CB_Port_Past;
21
22         public static bool New_Port = true, Unconnect = false, Start = true,
              Confirme_Change,Close_Windows;
23
24         public static Form1 frm1 = new Form1();
25
26         public Form1()
27         {
28             InitializeComponent();
29
30             //transfert des variables de la form2 à la form1
31             BaudRate = Form2.BaudRate_Change;
32             Parity = Form2.Parity_Change;
33             Data_bits = Form2.Data_Bits_Change;
34             Handshake = Form2.Handshake_Change;
35             Stop_Bits = Form2.Stop_Bits_Change;
36             Confirme_Change = Form2.Change;
37             Close_Windows = Form2.Windows_Close;
38         }
39
40         private void Form1_Load(object sender, EventArgs e)
41         {
42
43         }
44
45         private void Btn_Acc_Click(object sender, EventArgs e)
46         {
47             if (Global.Receive_Message == true)
48             {
49                 // preparation du message
50                 // indicateur de start = AA
51                 // derniere adresse recu
52                 // Message, ici c'est "ACCEPT"
53                 //indicateur de fin = BB
54                 string envoie = Constants.Start + Global.Actual_Adress +
                      Constants.Accept + Constants.End;
```

```
55                  if (AnyClass.port.IsOpen)//Verifier si le port com est ouvert
56                  {
57                      //envoie du message
58                      SerialPort.Write(envoie);
59                  }
60                  else
61                  {
62                      //Si le port n'est pas ouvert ouverture de fenetre pour
                        indiquer d'en selectionner un
63                      MessageBox.Show("Please select a port first and connect
                        it");
64                  }
65
66                  Global.Receive_Message = false;
67              }
68
69          }
70
71          private void Btn_Dec_Click(object sender, EventArgs e)
72          {
73              if (Global.Receive_Message == true)
74              {
75                  // preparation du message
76                  // indicateur de start = AA
77                  // derniere adresse recu
78                  // Message, ici c'est "DECLINE"
79                  //indicateur de fin = CC
80                  string envoie = Constants.Start + Global.List_Adress
                      [Global.Last_Adress] + Constants.Decline + Constants.End;
81                  if (AnyClass.port.IsOpen)//Verifier si le port com est ouvert
82                  {
83                      //envoie du message
84                      SerialPort.Write(envoie);
85                  }
86                  else
87                  {
88                      //Si le port n'est pas ouvert ouverture de fenetre pour
                        indiquer d'en selectionner un
89                      MessageBox.Show("Please select a port first and connect
                        it");
90                  }
91
92                  Global.Receive_Message = false;
93              }
94          }
95
96          private void Btn_Rst_Click(object sender, EventArgs e)
97          {
98              if (Global.Receive_Message == true)
99              {
100                 // preparation du message
101                 // indicateur de start = AA
102                 // Adresse de broadcast
103                 // Message, ici c'est "RESET"
104                 //indicateur de fin = CC
105                 string envoie = Constants.Start +
```

```
                    Constants.Adress_Broadcast_ES + Constants.Reset +
                    Constants.End;
106                 if (AnyClass.port.IsOpen)//Verifier si le port com est ouvert
107                 {
108                     //envoie du message
109                     SerialPort.Write(envoie);
110                 }
111                 else
112                 {
113                     //Si le port n'est pas ouvert ouverture de fenetre pour
                    indiquer d'en selectionner un
114                     MessageBox.Show("Please select a port first and connect
                    it");
115                 }
116             }
117
118         }
119
120         private void Btn_Opt_Click(object sender, EventArgs e)
121         {
122             //si on n'a aucun message à traiter
123             if (Global.Receive_Message == false)
124             {
125
126                 try
127                 {
128                     //on rend visible l'interface 2
129                     Form2.frm2.Visible = true;
130                     //on rend invisible la premiere interface
131                     Hide();
132                 }
133                 catch(Exception)
134                 {
135                     Form2 frm2 = new Form2();
136                     frm2.Show();
137                     Hide();
138                 }
139
140
141             }
142         }
143
144         private void CB_Port_Click(object sender, EventArgs e)
145         {
146
147             //on recois une list de nom de port
148             var ports = SerialPort.GetPortNames();
149             //on affiche les nom dans la combo box
150             CB_Port.DataSource = ports;
151         }
152
153         private void CB_Port_SelectedIndexChanged(object sender, EventArgs e)
154         {
155             //on convertie le nom selectionner en string et on l'affiche
156             CB_Port.Text = CB_Port.SelectedItem.ToString();
157             //on verifie si le Texte afficher à été modifier
```

```
158             if (CB_Port_Past != CB_Port.Text)
159             {
160                 //on met un flag pour deconnecter le port precedent
161                 Unconnect = true;
162 
163                 //vérifie si c'est la premiere fois que nous passons
164                 if(Start != true)
165                     //si ce n'est pas la premiere fois en entre dans la
                        méthode connect
166                 Connect();
167 
168                 //on copie la valeur de Texte
169                 CB_Port_Past = CB_Port.Text;
170 
171                 //on reset le flag de deconnection
172                 Unconnect = false;
173                 Start = false;
174             }
175             //on verifie si c'est on va utiliser un nouveau port
176             if (New_Port == true)
177             {
178                 //Si on entre dans cette déclaration cela veut dire que le
                      port est déjà fermer
179 
180                 //en cree une nouvelle instance
181                 AnyClass.port = new SerialPort(CB_Port.SelectedItem.ToString
                      ());
182                 //on reset le flag de nouveau port
183                 New_Port = false;
184             }
185 
186 
187 
188         }
189 
190         private void Btn_USB_Click(object sender, EventArgs e)
191         {
192             //on verifie si un port a été selectionner
193             if (CB_Port.SelectedIndex > -1)
194             {
195                 //on indique dans une nouvelle fenetre que le port à été
                      electionner
196                 MessageBox.Show(string.Format("You selected port '{0}'",
                      CB_Port.SelectedItem));
197                 Connect();
198             }
199             else
200             {
201                 MessageBox.Show("Please select a port first");
202             }
203         }
204 
205         //class global
206         abstract class AnyClass
207         {
208             //public static SerialPort port = new SerialPort();
```

```
209             public static SerialPort port;
210
211         }
212         private void Connect()
213         {
214
215             //SerialPort port = new SerialPort(CB_Port.SelectedItem.ToString
                ());
216             //AnyClass.port = new SerialPort(CB_Port.SelectedItem.ToString());
217
218             if (Unconnect == true)
219             {
220                 AnyClass.port.Close();
221                 New_Port = true;
222             }
223             else
224             {
225                 //on vérifie si le port est ouvert
226                 if(AnyClass.port.IsOpen)
227                     //s'il est ouvert on le ferme par précaution
228                     AnyClass.port.Close();
229
230                 if(Confirme_Change == true)
231                 {
232                     //modification des parametres de l'UART selon valeur
                        inscrit par l'utilisateur
233                     AnyClass.port.BaudRate = BaudRate;
234                     AnyClass.port.Parity = (Parity)Enum.Parse(typeof(Parity),
                        Parity);
235                     AnyClass.port.DataBits = Data_bits;
236                     AnyClass.port.Handshake = (Handshake)Enum.Parse(typeof
                        (Handshake), Handshake);
237                     AnyClass.port.StopBits = (StopBits)Enum.Parse(typeof
                        (StopBits), Stop_Bits);
238                 }
239                 else
240                 {
241                     //modification des parametres de l'UART selon valeur par
                        default
242                     AnyClass.port.BaudRate = Constants.Default_BaudRate;
243                     AnyClass.port.Parity = Constants.Default_Parity;
244                     AnyClass.port.DataBits = Constants.Default_DataBits; ;
245                     AnyClass.port.Handshake = Constants.Default_Handshake;
246                     AnyClass.port.StopBits = Constants.Default_Stop_Bits;
247                 }
248             }
249                 //on ouvre le port
250                 AnyClass.port.Open();
251         }
252
253         private void SerialPort_DataReceived(object sender,
              SerialDataReceivedEventArgs e)
254         {
255
256             try
257             {
```

```
258                 //on copie le message recu
259                 Global.All_Messages = SerialPort.ReadLine();
260             }
261             catch (TimeoutException) { }
262             //on met un flag de reception
263             bool Receive_Message = true;
264 
265         }
266     }
267 }
```