```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Threading.Tasks;
5  using System.IO.Ports;
6  using System.Threading;
7  using System.Windows.Forms;
8
9
10 namespace _1811C_Ticketing
11 {
12     public static class Global
13     {
14         public static string All_Messages;//reception du message complet
15         public static bool Receive_Message = false;//vérification si un
             message a ete recu
16         public static string[] List_Name_student = new string[25];//nom des
             eleves enregistrer
17         public static string[] List_Adress = new string[25];//adresse recu
18
19         public static char[] Name_student;//nom des eleves enregistrer
20         public static char[] Adress = new char[20];//adresse recu
21
22         public static int Nbr_Student_Save = 0;//Nombre de nom d eleve
             different
23         public static int Nbr_Adress_Save = 0;//Nombre d adresse different
24
25         public static string Actual_Adress;//derniere adresse recu
26         public static string Actual_Student;
27
28         public static bool Student_OK = false;//vérification si un message a
             ete recu
29
30         public static int Last_Adress;
31     }
32     static class Constants
33     {
34         public const string Start = "AA",  End = "BB";
35         public const int Size_Indicator = 2;
36         public const int Size_Adress = 6; // 4 pour le Xbee XB24CAPIT-001 ou
             16 selon mode utiliser
37         public const int Size_Adress_Broadcast = 8;// 4 pour le Xbee
             XB24CAPIT-001 et 8 pour le broadcast du xbee de l es
38         public const string Adress_Broadcast_ES = "00000000";//adresse de
             broadcast du xbee de l es
39         public const string Adress_Broadcast_Xbee4 = "FFFF";//adresse de
             broadcast du xbee XB24CAPIT-001
40         public const string Adress_Broadcast_Xbee16 = "000000000000FFFF";//
             adresse de broadcast du xbee XB24CAPIT-001
41         public const string Accept = "ACCEPT", Decline = "DECLINE", Reset =
             "RESET";// constant des reponses à envoyer
42
43
44         public const int Default_BaudRate = 57600;// valeur de default du
             BaudRate de la communication serial
45         public const System.IO.Ports.Parity Default_Parity =
```

```
            System.IO.Ports.Parity.None;// valeur de default de la Partiy de la communication serial
46          public const int Default_DataBits = 8;// valeur de default du nombre de bytes de la communication serial
47          public const System.IO.Ports.Handshake Default_Handshake = System.IO.Ports.Handshake.None;// valeur de default du handshake de la communication serial
48          public const System.IO.Ports.StopBits Default_Stop_Bits = System.IO.Ports.StopBits.One;// valeur de default de la taille du stop bits de la communication serial
49
50      }
51      static class Program
52      {
53          /// <summary>
54          /// Point d'entrée principal de l'application.
55          /// </summary>
56
57
58          [STAThread]
59          static void Main()
60          {
61              Application.EnableVisualStyles();
62              Application.SetCompatibleTextRenderingDefault(false);
63              Application.Run(new Form1());
64
65              if (Global.Receive_Message == true)// on verifie que l'on a bien receptionner un message
66
67              {
68                  char[] Message = new char[Global.All_Messages.Length];
69                  int i = 0, compt = 0;
70
71                  for (i = 0; i < Global.All_Messages.Length; i++)
72                  {
73                      Message[i] = Global.All_Messages[i]; //copier le string dans un tableau de char
74                  }
75                  if (Message[0] == 'A' && Message[1] == 'A')// on verifie si le debut de la trame recu correspond à l'indicateur Start_First = AA
76                  {
77                      do
78                      {
79                          Global.Adress[compt] = Message[compt + Constants.Size_Indicator]; //copier l'adresse dans un tableau de char (en ignorant les 2 premier caracteres qui sont l'indicateur d envoie)
80                          compt++; //indicateur de la position du curseur
81
82                      } while (compt != Constants.Size_Adress + Constants.Size_Indicator); // tant que l'on atteint pas 8 continuer la boucle (2 caracteres pour indicateur + 6 de l adresse)
83                      compt = 0;// on reinitialise la position du curseur
84                      string Adress = new string(Global.Adress);// on copie le
```

```
                    tableau de caratere dans une variable string
85
86
87                  for (i = 0; i < Global.List_Adress.Length; i++)// on fait
                    une boucle de la taille du tableau liste d adresse
88                  {
89                      if (Global.List_Adress[i] != Adress)//On verifie s'il
                    existe
90                          Global.List_Adress[Global.Nbr_Adress_Save] =
                    Adress; // on copie l adresse dans le tableau de string
                    liste d adresse
91                  }
92                  Global.Actual_Adress = Adress;
93
94
95                  for (i = Adress.Length + Constants.Size_Indicator; i <
                    Global.All_Messages.Length - Adress.Length - 2 *
                    Constants.Size_Indicator; i++)// faire la boucle (taille
                    max du message total moins la taille de l adress et des
                    indicateur de début et de fin)
96                  {
97                      Global.Name_student[i] = Message[i + Adress.Length +
                    Constants.Size_Indicator];//copie du nom de l eleve du
                    message totale dans un tableau de char
98                  }
99
100                 string Name_Student = new string(Global.Name_student);//on
                     copie le tableau de caratere dans une variable string
101                 Global.Nbr_Student_Save++;
102                 Global.List_Name_student[Global.Nbr_Student_Save] =
                    Name_Student;// on copie le nom de l eleve dans le tableau
                     de string liste de nom d eleve
103                 Global.Actual_Student = Name_Student;
104             }
105             /*else if (Message[0] == 'B' && Message[1] == 'B')// on
                  verifie si le debut de la trame recu correspond à
                  l'indicateur Start = BB
106             {
107                 do
108                 {
109                     Global.Adress[compt] = Message[compt +
                    Constants.Size_Indicator]; //copier l'adresse dans un
                    tableau de char (en ignorant les 2 premier caracteres qui
                    sont l'indicateur d envoie)
110                     compt++; //indicateur de la position du curseur
111
112                 } while (compt != Constants.Size_Adress +
                    Constants.Size_Indicator); // tant que l'on atteint pas 8
                    continuer la boucle (2 caracteres pour indicateur + 6 de l
                     adresse)
113                 compt = 0;// on reinitialise la position du curseur
114                 string Adress = new string(Global.Adress);// on copie le
                    tableau de caratere dans une variable string
115                 Global.Actual_Adress = Adress;
116                 while(Global.Student_OK == true)
117                 {
```

```
118                         if(string.Equals(Global.List_Adress[compt] , Adress))
119                         {
120                             Global.Student_OK = true;
121 
122                             Global.Actual_Student = Global.List_Name_student
                        [compt];
123                         }
124                         else
125                         {
126                             compt++;
127                         }
128                     }
129                     compt = 0;
130 
131                 }*/
132             }
133         }
134     }
135 }
136 
```