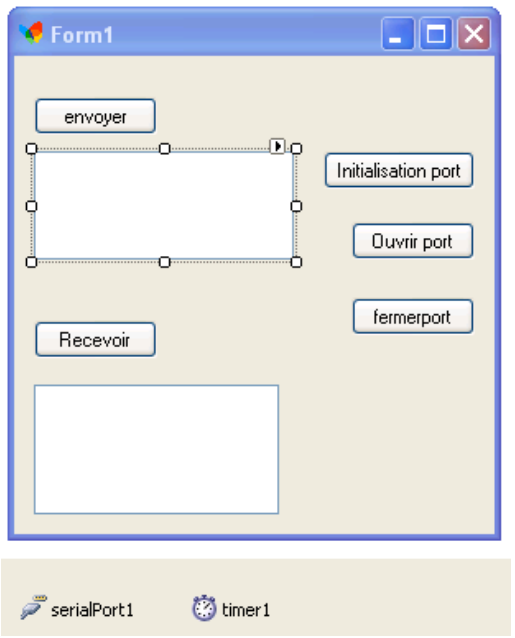
TP Gestion du port série en C# (par lecture périodique).

I) Création du projet

- Créer un nouveau projet C#, nommé PortSerieAvecTimer.



II) Interface graphique suivante:

- Réaliser l'interface graphique suivante:
- Ne pas oublier de renommer les différents contrôles

III) Contrôles

- Placer le contrôle serialPort.
- Placer le contrôle Timer.

IV) Code

ATTENTION LE CODE DONNEE CI-DESSOUS EST UN EXEMPLE, IL FAUT L'ADAPTER A VOS NOM DE VARIABLES

- Compléter la classe de formulaire avec le code donné ci-dessous (attributs et constucteur):

```
public partial class Form1 : Form
{

    string MaChaineAEmettre;
    string MaChaineCompleteRecue;

    public Form1()
    {
        InitializeComponent();
        timer1.Enabled = false;
        MaChaineCompleteRecue = "";
    }
```

- Créer le gestionnaire clic du bouton "Initialisation port", et compléter le code:

ATTENTION: dans notre cas utiliser "COM1"

```
private void BInit_Click(object sender, EventArgs e)
{
    serialPort1.BaudRate = 9600;
    serialPort1.PortName = "COM2";
    serialPort1.Parity = System.IO.Ports.Parity.None;
    serialPort1.StopBits = System.IO.Ports.StopBits.One;


}
```

- Créer le gestionnaire clic du bouton "Ouvrir port", et compléter le code:

```
private void button1_Click(object sender, EventArgs e)
{
    serialPort1.Open();
}
```

- Créer le gestionnaire clic du bouton "Fermer port", et compléter le code:

```
private void button3_Click(object sender, EventArgs e)
{
    timer1.Enabled = false;
    serialPort1.Close();
}
```

- Créer le gestionnaire clic du bouton "Envoyer", et compléter le code:

```
private void button2_Click(object sender, EventArgs e)
{
    serialPort1.WriteLine(MaChaineAEmettre);
}
```

- Créer le gestionnaire de l'événement **TextChanged** pour la TextBox d'envoi, et compléter le code:

```
private void textBox1_TextChanged(object sender, EventArgs e)
{

    MaChaineAEmettre = textBox1.Text;

}
```

- Créer le gestionnaire clic du bouton "Recevoir", et compléter le code:

```
private void buttonReception_Click(object sender, EventArgs e)
{
    timer1.Enabled = true;
}
```

- Créer le gestionnaire Tick du Timer (par double-clic) et compléter le code:

```
private void timer1_Tick(object sender, EventArgs e)
{
    string MaChaineRecue;
    //MaChaineRecue = serialPort1.ReadLine();
    //MaChaineRecue = serialPort1.ReadByte();
    MaChaineRecue = serialPort1.ReadExisting();
    MaChaineCompleteRecue = MaChaineCompleteRecue + MaChaineRecue;
    textBox2.Text = MaChaineCompleteRecue;
}
```

Paramétrage du timer:

Nous utilisons un timer afin de **lire périodiquement le buffer de réception** du port série.

- **Calcul de l'intervalle de temps entre deux lectures du buffer:**

Ex: 1 caractère = 10 bits (données: 8, parité: 1, stop: 1), à 9600 bits/s on a 960 caractères/s soit environ 1 ms par caractère. Le buffer de réception ayant une taille de 4096 octets soit 4096 caractères (Valeur par défaut), il est donc rempli au bout de 4096 ms. L'intervalle de temps du timer doit être inférieur à 4096 ms si on ne veut pas perdre de caractères.

Par contre le timer réalise aussi la mise à jour de l'affichage de la TextBox de réception. Pour une actualisation fréquente de l'affichage, une valeur de l'ordre de 100ms semble correct entre chaque Tick du timer.

- Fixer la propriété Interval du Timer à 100.

- Compiler le projet

V) Tests

- Faire des essais de communication entre votre pc et un autre pc avec hyperterminal pour la mise au point.
- Faire des essais de communication entre votre pc et un autre pc avec la même application..

TP Gestion du port série en C# (par événement).

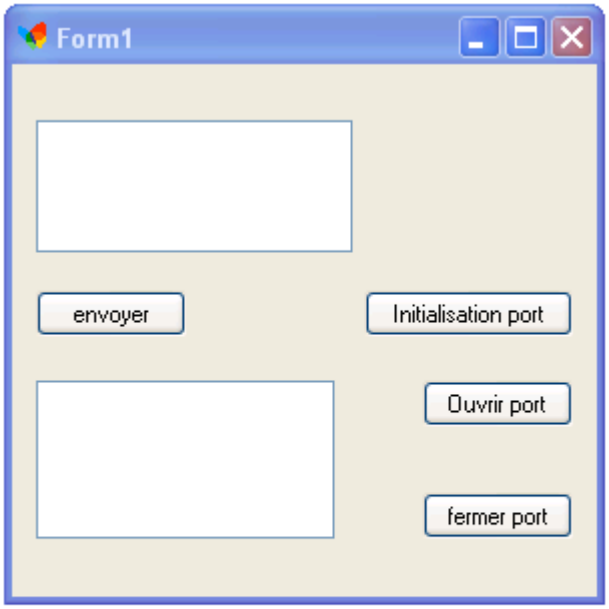
A la différence du Tp précédent, on ne va pas utiliser de timer, mais l' événement caractère reçu.
Ainsi la lecture du tampon de réception se fera uniquement lorsqu'un caractère sera reçu, ce qui monopolise moins de ressources système.

I) Création du projet

- Créer un nouveau projet C#, nommé PortSerieAvecE-venement.

II) Interface graphique suivante:

- Réaliser l'interface graphique suivante:
- Ne pas oublier de renommer les différents contrôles



III) Contrôles

- Placer le contrôle serialPort.
- Placer un contrôle timer (celui-ci sera uniquement utilisé pour l'actualisation de l'affichage).

IV) Code

Seul le code différent du tp précédent est donné.

ATTENTION LE CODE DONNEE CI-DESSOUS EST UN EXEMPLE, IL FAUT L'ADAPTER A VOS NOM DE VARIABLES

- **compléter le constructeur de la classe de formulaire:**

```
public Form1()
{
    InitializeComponent();
    MaChaineCompleteRecue = "";
}
```

- **Créer le gestionnaire d'événement DataReceived du port série, et compléter son code:**

```
private void serialPort1_DataReceived(object sender, System.IO.Ports.SerialDataReceivedEventArgs e)
{
    string MaChaineRecue;
    //MaChaineRecue = serialPort1.ReadLine();
    //MaChaineRecue = serialPort1.ReadByte();
    MaChaineRecue = serialPort1.ReadExisting();
    MaChaineCompleteRecue = MaChaineCompleteRecue + MaChaineRecue;

}
```

- Créer le gestionnaire Tick du timer et compléter son code:

```
private void timer1_Tick(object sender, EventArgs e)
{
    textBox2.Text = MaChaineCompleteRecue;
}
```

- Compiler et tester le projet dans les mêmes conditions que le tp précédent.