UNIVERSIDAD DEL BÍO-BÍO
La Universidad de la Región del Biobío

Laboratorio de MAC-IPP
Profesor: Luís Vera Quiroga
Alumno ayudante: José I. Veloso Inzunza

Practico Nº2 Comunicación Serial y Socket

Ejercicio 3: “Trabajando con SerialPort y placa Arduino Uno”

Descripción: En este práctico el objetivo es enviar comandos y recibir respuestas de una placa Arduino Uno conectada a través del puerto serial, donde la placa contendrá en su memoria un programa o “sketch” previamente cargado y en ejecución con sus respectivos sensores conectados.

Abrir el programa Visual Studio y seleccionar **Crear un proyecto**, luego se debe seleccionar la opción **Aplicación de Windows Forms (.Net Framework),** implementado en C#. Asigne el nombre que usted desee y luego diseñe el formulario agregando los siguientes componentes:

Nombre	**Propiedades**
Form1	Text: Conexión con Arduino Size: 672; 451
Button1	Text: Conectar
Button2	Text: Desconectar
Button3	Text: Enviar
Button4	Text: Guardar
Button5	Text: Finalizar comunicación
ComboBox1	Ítems: COM1, COM2, COM3, COM4, COM5 DropDownStyle: DropDownList
textBox1	Text: TextAlign: Center Enabled: False
RichTextBox1	Text:
RichTextBox2	Text:
Label1	Text: Estado de conexión:
Label2	Text: Puerto
Label3	Text: Entrada
Label4	Text: Salida
Timer1	
SaveFileDialog1	

El diseño del formulario debería quedar de la siguiente manera:

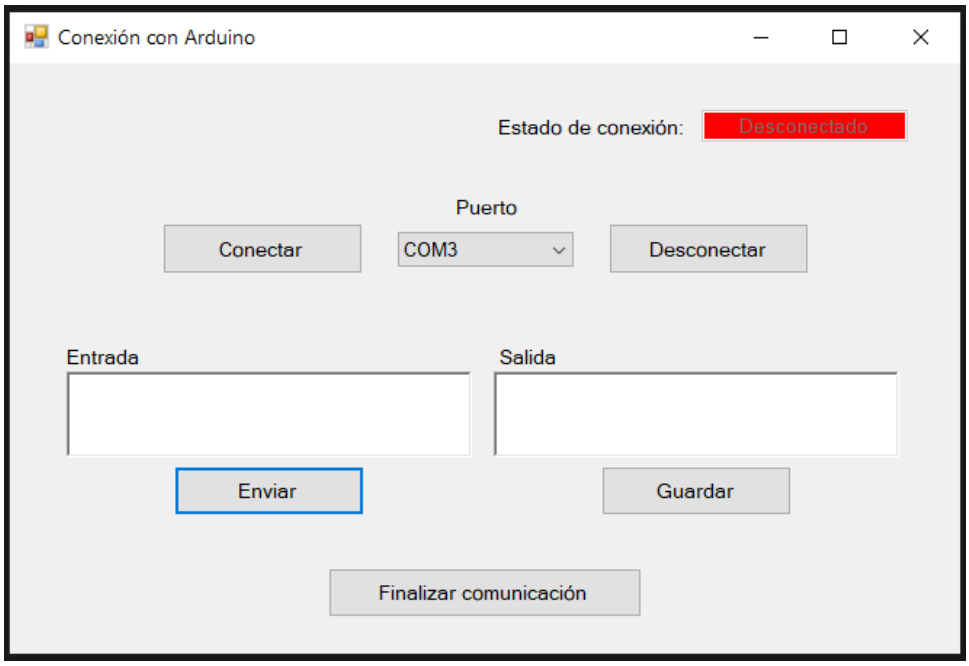

Figura 1. Diseño del Form

Importante: para llevar a cabo la actividad agregue una biblioteca adicional de entrada y salida para comunicación, para ello haga doble clic en el Form1 y en el encabezado añada lo siguiente:

```
using System.IO;
```

1. Agregue al *ComboBox1* los puertos COM a disposición, por medio de la propiedad Ítems. Para bloquear la edición del *ComboBox1* cambie la propiedad *DropDownStyle* a *DropDownList*.

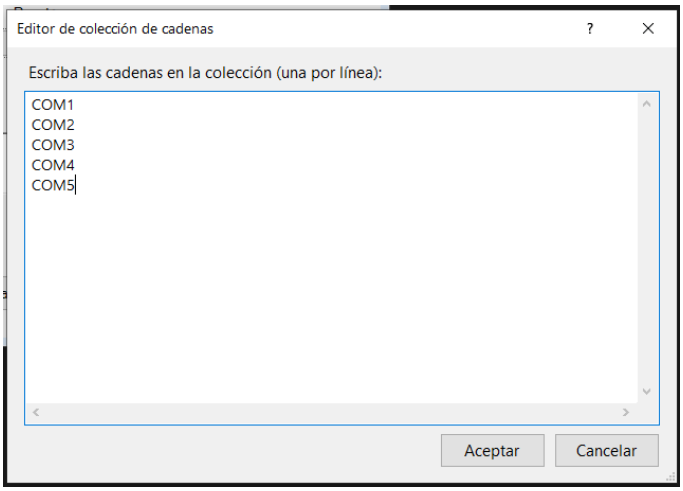

Figura 2. ComboBox

2. Agregue el siguiente código a la clase principal:

```
System.IO.Ports.SerialPort Arduino;
1 referencia
public Form1()
{
    InitializeComponent();
    Arduino = new System.IO.Ports.SerialPort(); //renombra al puerto serial como arduino
    comboBox1.SelectedIndex = 2; //selecciona "COM3" del listado comboBox1, como item por defecto

    textBox1.Text = "Desconectado";
    textBox1.BackColor = Color.Red; //indica que el puerto COM esta desconectado por defecto
}
```

Esto le asigna el nombre “Arduino” a la conexión serial para ser utilizada fácilmente, además selecciona por defecto el ítem 2 (“COM3”) para cada vez que se ejecute la aplicación.

3. Agregue el siguiente código a la función timer1_Tick, acceda a ella haciendo doble click en el botón “timer1” ubicado en la parte inferior del Form:

```
try
{
    if (Arduino.IsOpen)
    {
        string salida = Arduino.ReadExisting().ToString(); //guarda en string "salida" la salida entregada por el arduino
        if (salida.Length > 0)
        {
            richTextBox1.Text = salida; //guarda en richTextBox1 la ultima salida del arduino
            //richTextBox1.Text += salida + "\n"; //concatena todas las salidas del arduino y las guarda en richTextBox1
        }
    }
} catch (Exception ex)
{
    MessageBox.Show(ex.Message);
}
```

Esto captura las salidas de Arduino a intervalos de 1000 ms (configurable en propiedades de timer1) siempre y cuando esté conectado al puerto COM. El registro se muestra en el richTextBox1 y es posible concatenar o reemplazar cada registro con el ultimo que se obtenga.

4. Agregue el siguiente código para el botón “Conectar”, haciendo doble clic sobre el mismo botón:

```
try
{
    timer1.Enabled = true; //inicia el timer, el que monitorea la salida de arduino
    Arduino.PortName = comboBox1.SelectedItem.ToString(); //el puerto de conexion es el seleccionado en el comboBox1
    Arduino.BaudRate = 9600;

    if (!Arduino.IsOpen)
    {
        Arduino.Open(); //conecta el puerto COM
        textBox1.Text = "Conectado";
        textBox1.BackColor = Color.Lime; //cuando el puerto COM se conecta, un label coloreado lo indica
    }
}
catch (Exception ex)
{
    MessageBox.Show(ex.Message);
}
```

En cuanto se presione el botón se iniciará el timer, se obtendrá el nombre del puerto seleccionado y en la condición if se comprobará que el puerto esté desconectado previamente. La velocidad de comunicación serial es de 9600 bits por segundo.

5. Luego agregue el código del botón “Desconectar”:

```
timer1.Enabled = false;
if (Arduino.IsOpen)
{
    Arduino.Close(); //desconecta el puerto COM
    textBox1.Text = "Desconectado";
    textBox1.BackColor = Color.Red; //cuando el puerto COM se desconecta, un label coloreado lo indica
}
```

6. Agregue el siguiente código al botón “Enviar”:

```
try
{
    Arduino.WriteLine(richTextBox2.Text.ToString()); //envia lo que contiene richTextBox2 al arduino
    //timer1.Enabled = true; //inicia el timer, el que monitorea la salida de arduino
}
catch (Exception ex)
{
    MessageBox.Show(ex.Message);
}
```

Esto obtendrá el texto contenido en el richTextBox de entrada y lo enviará al Arduino conectado.

7. Agregue el siguiente código del botón “Guardar”:

```
if (saveFileDialog1.ShowDialog() == DialogResult.OK)
    File.WriteAllLines(saveFileDialog1.FileName + " .txt", richTextBox1.Lines);
```

Esto almacenará en un archivo el texto contenido en el richTextBox de salida.

8. Agregue el siguiente código para el botón “Finalizar Comunicación”.

```
timer1.Enabled = false;
if (Arduino.IsOpen)
{
    Arduino.Close(); //desconecta el puerto COM
    textBox1.Text = "Desconectado";
    textBox1.BackColor = Color.Red; //cuando el puerto COM se desconecta, un label coloreado lo indica
}
```

Este botón realiza una desconexión del puerto COM y detiene el timer, dejando el último mensaje de Arduino en el richTextBox de salida.