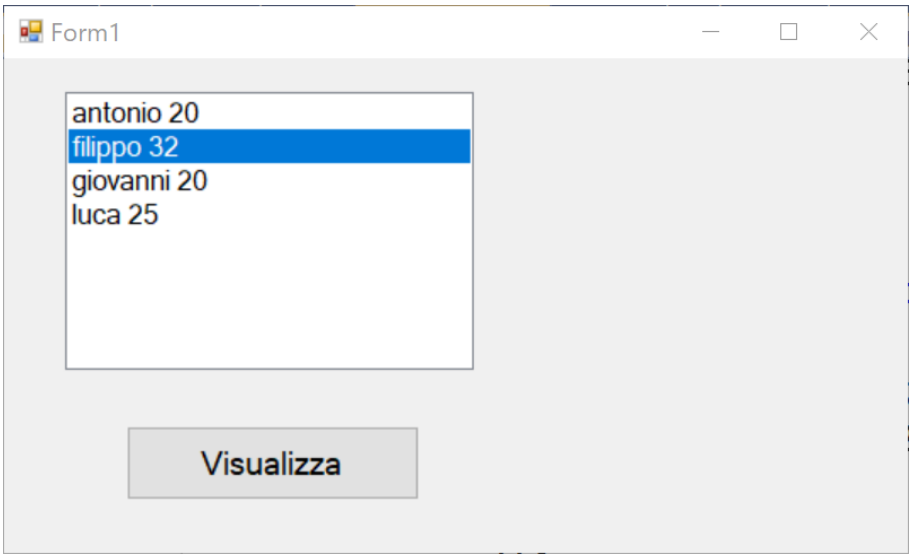
Uso di due Windows Form

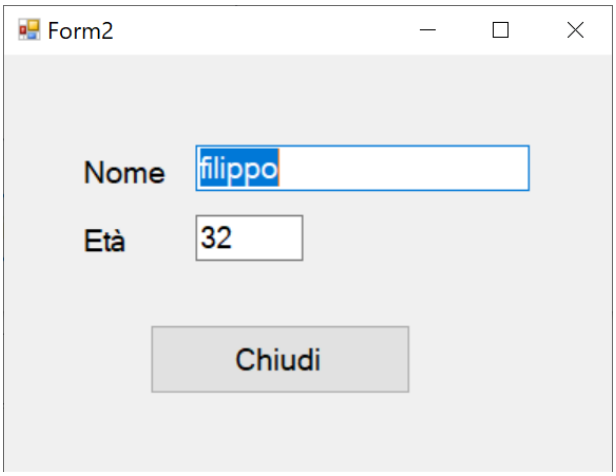
Quando dal form principale di un programma si vuole creare un secondo form per mostrare all'utente dei dati di dettaglio o per acquisire dei dati in input, si ha il problema del passaggio di informazioni dal form principale al form secondario.

Esempio con sola visualizzazione di dati

Il form principale mostra una listbox con un elenco di persone e il pulsante visualizza consente di vedere nel dettaglio i dati della persona selezionata.



Il Form principale



Il Form secondario

In pratica il form1 passa al costruttore del form2 il riferimento all'oggetto da visualizzare.

La classe Persona ridefinisce il metodo ToString() per consentire la visualizzazione dei dati nel listbox.

```
public class Persona
    {
        public string Nome { get; set; }
        public int Età { get; set; }



        public override string ToString()
        {
            return Nome + " " + Età;


        }
    }
```

```
    public partial class Form1 : Form
    {


        private List<Persona> elenco;


        public Form1()
        {
            elenco = new List<Persona>();
            elenco.Add(new Persona() { Nome = "antonio", Età = 20 });
            elenco.Add(new Persona() { Nome = "filippo", Età = 32 });
            elenco.Add(new Persona() { Nome = "giovanni", Età = 20 });
            elenco.Add(new Persona() { Nome = "luca", Età = 25 });


            InitializeComponent();
        }
```

```
        private void Form1_Load(object sender, EventArgs e)
        {
            listBox1.DataSource = elenco;
        }

        // pulsante Visualizza
        private void button1_Click(object sender, EventArgs e)
        {
            Persona corrente = (Persona) listBox1.SelectedItem;
            Form2 form2 = new Form2(corrente);
            form2.Show();
        }
    }
```

```
    public partial class Form2 : Form
    {
        private Persona persona;
        public Form2(Persona p)
        {
            persona = p;
            InitializeComponent();

        }



        private void Form2_Load(object sender, EventArgs e)
        {
                textBox1.Text = persona.Nome;
```

```
                textBox2.Text = persona.Età.ToString();
        }

        // pulsante Chiudi
        private void button1_Click(object sender, EventArgs e)
        {
            this.Close();
        }
    }
```

Si noti che la finestra secondaria è indipendente da quella principale: essa può restare aperta e l'utente può tornare alla finestra principale e continuare il suo lavoro. Ad esempio, risulta possibile aprire più di una finestra secondaria contemporaneamente.

Una modalità diversa di apertura della finestra secondaria consiste nell'aprirla come "finestra di dialogo modale". In questo modo l'utente, se vuole ritornare al form principale, deve prima chiudere la finestra di dialogo, ovvero il form secondario.

In questo caso l'istruzione di apertura diventa:

```
form2.ShowDialog(); // apertura come finestra di dialogo modale
```

Esempio con sola visualizzazione e anche modifica di dati

Se il form secondario effettua una modifica dei dati con un apposito pulsante

```
// pulsante Aggiorna
        private void button1_Click(object sender, EventArgs e)
        {
            persona.Nome = textBox1.Text;
            persona.Età = Convert.ToInt32(textBox2.Text);
            this.Close();
        }
```

si ottiene la modifica dei dati memorizzati nell'elenco delle persone ma non avviene l'aggiornamento della visualizzazione nel listbox.

Affinché nel form principale vengano visualizzati i dati aggiornati, occorre riassegnare il DataSource della listbox con le istruzioni del seguente metodo della classe Form1:

```
public void AggiornaListBox()
{
       listBox1.DataSource = null;
       listBox1.DataSource = elenco;
}
```

Per fare in modo di rendere automatico questo aggiornamento si deve fare in modo che il form secondario chiami il suddetto metodo al momento della chiusura dello stesso.

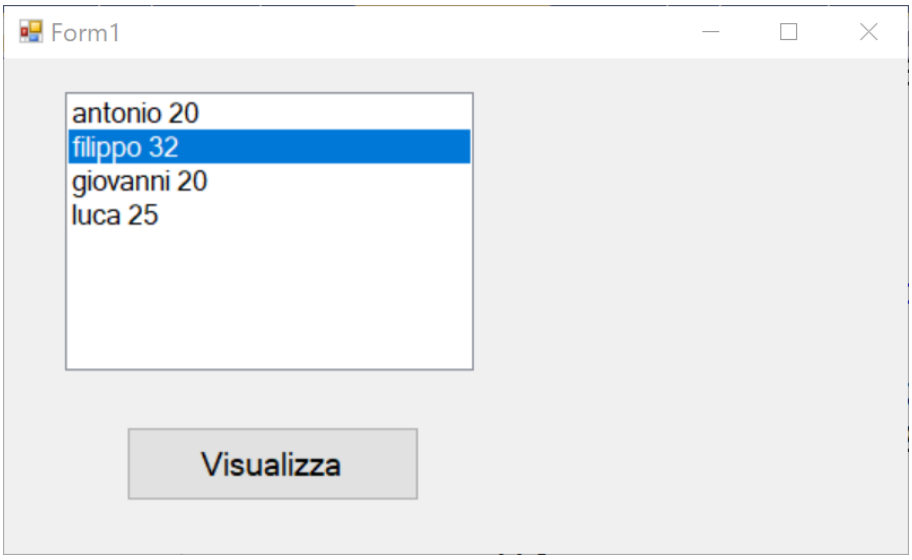
Pertanto il form principale quando crea il form secondario gli deve passare, come parametro fornito al suo costruttore, anche un riferimento a se stesso (this).

```
// modifiche apportate alla classe Form1


        // pulsante Visualizza
        private void button1_Click(object sender, EventArgs e)
        {
            Persona corrente = (Persona) listBox1.SelectedItem;
            Form2 form2 = new Form2(this, corrente);
            //form2.ShowDialog(); // finestra di dialogo modale
            form2.Show();
        }


// metodo per aggiornare la list box
        public void AggiornaListBox()
        {
            listBox1.DataSource = null;
            listBox1.DataSource = elenco;
        }
```

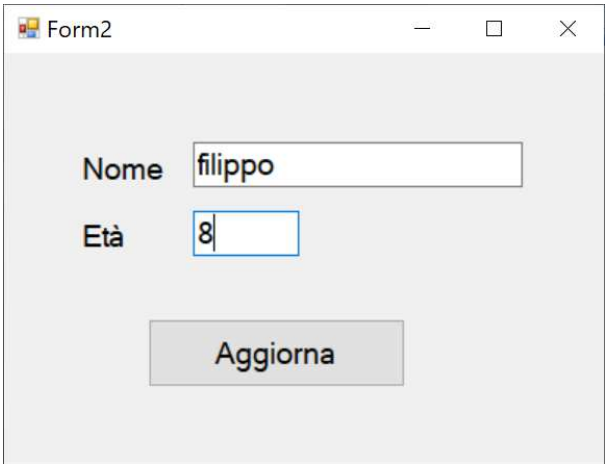
La nuova versione della classe Form2 prevede come attributo aggiuntivo un riferimento al form genitore:

```
// versione che consente di aggiornare la visualizzazione del form principale


    public partial class Form2 : Form
    {
        private Form1 genitore;
        private Persona persona;
        public Form2(Form1 f, Persona p)
        {
            genitore = f;
            persona = p;
            InitializeComponent();
        }


        private void Form2_Load(object sender, EventArgs e)
        {
                textBox1.Text = persona.Nome;
                textBox2.Text = persona.Età.ToString();
        }


        // pulsante Aggiorna
        private void button1_Click(object sender, EventArgs e)
        {
            persona.Nome = textBox1.Text;
            persona.Età = Convert.ToInt32(textBox2.Text);
            genitore.AggiornaListBox();
            this.Close();
        }
    }
```

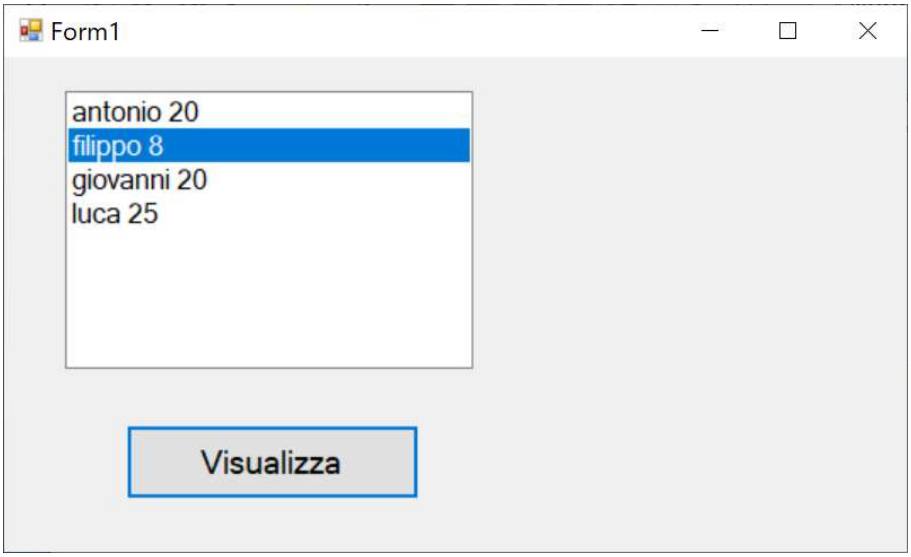
Le seguenti schermate mostrano il corretto funzionamento del codice:



Il Form principale



Modifica dell'età di filippo



Visualizzazione aggiornata