Label e caselle di Testo

Prof. Francesco Accarino
IIS Altiero Spinelli Sesto San Giovanni

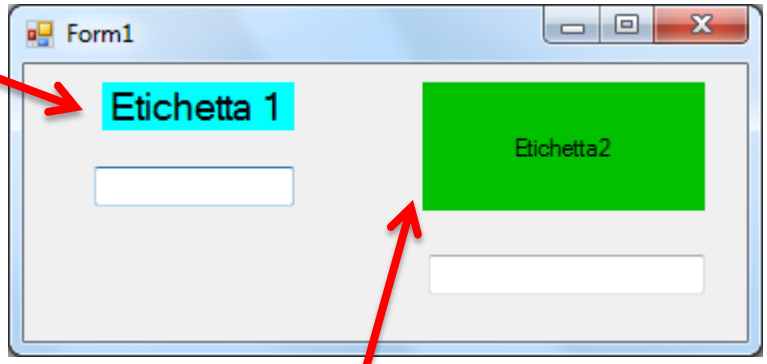
Classe «Label»

L'uso più comune delle Label è quello di svolgere il ruolo di testo informativo, con lo scopo di qualificare gli altri controlli, soprattutto TextBox, specificando la natura delle informazioni che essi visualizzano e/o consentono di inserire.

PROPRIETÀ - TIPO	DESCRIZIONE
AutoSize `bool`	Impostando a `true` questa proprietà le dimensioni del controllo variano in modo automatico in relazione allo spazio occupato dal testo da visualizzare. Tale spazio dipende oltre che dalla "quantità" di testo anche dalle caratteristiche del Font impostato.
Text `string`	Contiene il testo da visualizzare. Se all'interno del testo è presente il carattere & («e» commerciale), il carattere che lo segue viene interpretato come «tasto di accesso» o «acceleratore», e cioè un tasto che premuto in combinazione con ALT determina la selezione del controllo associato alla `Label`.
TextAlign `ContentAlignment`	Consente di impostare l'allineamento del testo rispetto all'area del controllo. (Funziona solo se `AutoSize` è `false`). I possibili valori sono: `BottomCenter`, `BottomLeft`, `BottomRight`, `MiddleCenter`, `MiddleLeft`, `MiddleRight`, `TopCenter`, `TopLeft`, `TopRight`.

Esempio Classe «Label»

```
//
// lblLabel1
//
this.lblLabel1.AutoSize = true;
this.lblLabel1.BackColor = System.Drawing.Color.Cyan;
this.lblLabel1.Font = new System.Drawing.Font("Microsoft Sans Serif", 14.25F, System.Drawing.FontStyle.Regular,
                                              System.Drawing.GraphicsUnit.Point, ((byte)(0)));
this.lblLabel1.Location = new System.Drawing.Point(39, 9);
this.lblLabel1.Name = "lblLabel1";
this.lblLabel1.Size = new System.Drawing.Size(96, 24);
this.lblLabel1.TabIndex = 0;
this.lblLabel1.Text = "Etichetta 1";
```



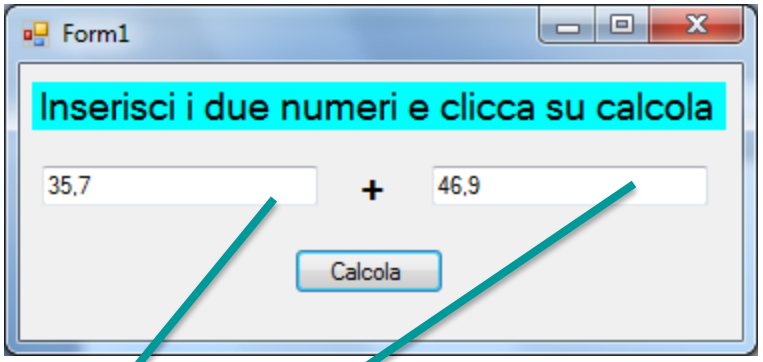

```
//
// lblLlabel2
//
this.lblLlabel2.BackColor = System.Drawing.Color.FromArgb(0,192,45);
this.lblLlabel2.Location = new System.Drawing.Point(199, 9);
this.lblLlabel2.Name = "lblLlabel2";
this.lblLlabel2.Size = new System.Drawing.Size(141, 64);
this.lblLlabel2.TabIndex = 2;
this.lblLlabel2.Text = "Etichetta2";
this.lblLlabel2.TextAlign = System.Drawing.ContentAlignment.MiddleCenter;
```

Classe «TextBox»

PROPRIETÀ - TIPO	DESCRIZIONE
CharacterCasing `CharacterCasing`	Impostando questa proprietà è possibile forzare il «case» dei caratteri digitati dall'utente, trasformandoli in maiuscolo, minuscolo oppure lasciandoli inalterati. I possibili valori sono: `Normal`, `Upper`, `Lower`.
MaxLength `int`	Mediante questa proprietà è possibile specificare il numero massimo di caratteri che l'utente può inserire.
MultiLine `bool`	Definisce la natura – «singola linea» o «multilinea» – del `TextBox`. Un `TextBox` multilinea consente la visualizzazione e l'inserimento di più righe di testo.
Lines `string[]`	Consente, nel caso di un TextBox multilinea (vedi prop. `MultiLine`) di accedere al testo contenuto nel controllo come ad un array di stringhe.
ReadOnly `bool`	Impostando a `true` questa proprietà si rende il `TextBox` di «sola lettura», impedendo all'utente di modificarne il contenuto.
TextAlign `HorizontalAlignment`	Consente di specificare l'allineamento orizzontale del testo. I possibili valori sono: `Center`, `Left`, `Right`.
WordWrap `bool`	Questa proprietà funziona solo per i `TextBox` multilinea (vedi proprietà `MultiLine`). Impostata a `true` fa sì che il controllo esegua un ritorno a capo automatico quando il testo digitato dall'utente raggiunge il limite destro.

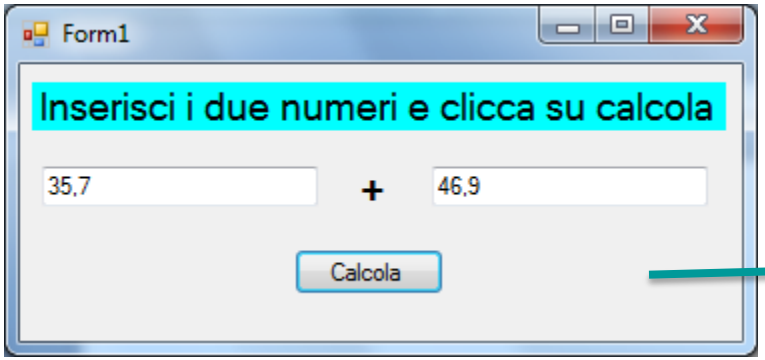
Classe «TextBox»

La proprietà più importante di una TextBox è la proprietà Text attraverso la quale si accede al testo contenuto in essa .

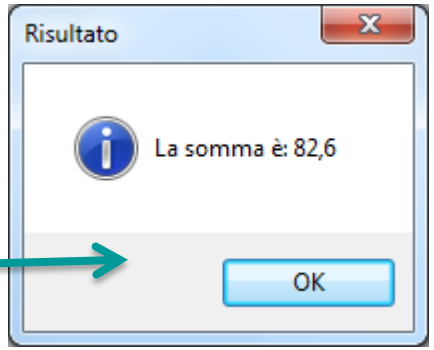



```
private void btnCalcola_Click(object sender, EventArgs e)
        {
            double nOp1=0, nOp2=0;
            try
            {
                nOp1 = Convert.ToDouble(txtN1.Text);
                nOp2 = Convert.ToDouble(txtN2.Text);
                MessageBox.Show("La somma è: " + Convert.ToString(nOp1 + nOp2), "Risultato",
                                 MessageBoxButtons.OK, MessageBoxIcon.Information);
            }

            catch (FormatException fe) {

                if(nOp1==0)
                MessageBox.Show(fe.Message.ToString() + "\n Devi inserire un numero valido come primo
                                addendo", "Errore", MessageBoxButtons.OK, MessageBoxIcon.Error);
                else
                MessageBox.Show(fe.Message.ToString() + "\n Devi inserire un numero valido come
                                secondo addendo", "Errore", MessageBoxButtons.OK,
                                MessageBoxIcon.Error);
            }
```

Esercizio 2: Sviluppare un'applicazione che si comporta come illustrato di seguito:



Valori idonei



Nessun valore o Valori idonei