

Primi Esempi

Visual Basic viene definito dalla Microsoft come un "Sistema di programmazione per Windows". Il termine "sistema" viene utilizzato ad indicare il fatto che si tratta di un insieme di componenti che vengono forniti insieme, ed insieme operano, nell'ambito della programmazione. Vediamo perciò prima di tutto che cosa si intende per "programmazione": con tale parola si identifica la branca dell'informatica che si occupa di creare programmi software. Ogni programma che si utilizza sul proprio computer – dal sistema operativo Windows, a Word o ad Excel – sono stati prodotti utilizzando la programmazione. Lo scopo della programmazione è quello di produrre file eseguibili (file di estensione EXE), che possono poi essere eseguiti dall'utente, per rispondere alle sue esigenze.

Visual Basic si occupa di programmazione mettendo a disposizione dell'utente (non l'utente finale del programma da creare, ma l'"utente-sviluppatore" che crea il programma) un insieme di strumenti, che sono poi quelli "classici" di tutti i sistemi di programmazione. Vediamoli:

- **un linguaggio di programmazione:** costituisce la parte fondamentale. Il linguaggio di programmazione è l'insieme di regole "grammaticali" attraverso cui si "disegna" il programma che si vuole creare e gli si spiega come fare ciò che deve fare. Visual Basic, in quanto applicazione Windows è facile da usare, infatti si serve degli strumenti classici di Windows, come i pulsanti o le finestre: la vera parte da imparare è proprio il linguaggio. A sua volta, il linguaggio di Visual Basic può essere scomposto in due grandi famiglie, come sempre distinte ma interagenti:

- la scrittura del codice,
- la costruzione dell'interfaccia.

Il codice costituisce il linguaggio in senso più stretto: il suo significato sarà più chiaro non appena ne vedremo i primi esempi. Nei linguaggi di vecchio tipo (ante-Windows) il codice costituiva praticamente l'unica cosa da creare (anche l'interfaccia veniva creata attraverso il codice, peraltro con notevole difficoltà, sugli stessi vostri banchi io ho studiato il vecchio Basic e il Cobol). Con l'avvento di Windows, l'interfaccia, che pure assume un rilievo ed una complessità assai maggiori, può essere costruita graficamente (con il mouse), in modo assai più semplice ... e più divertente.

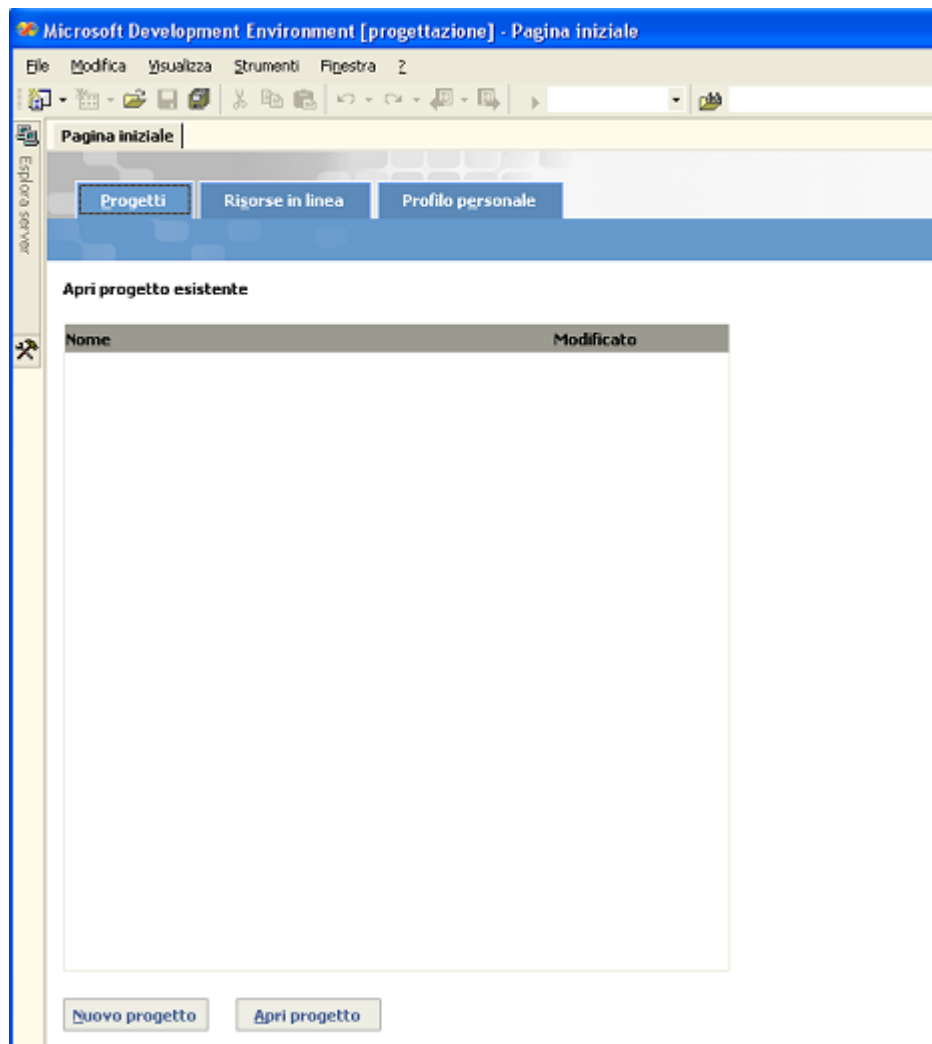
L'insieme di file che contengono il codice e descrivono l'interfaccia del programma da creare costituiscono i file "sorgenti" del programma stesso.

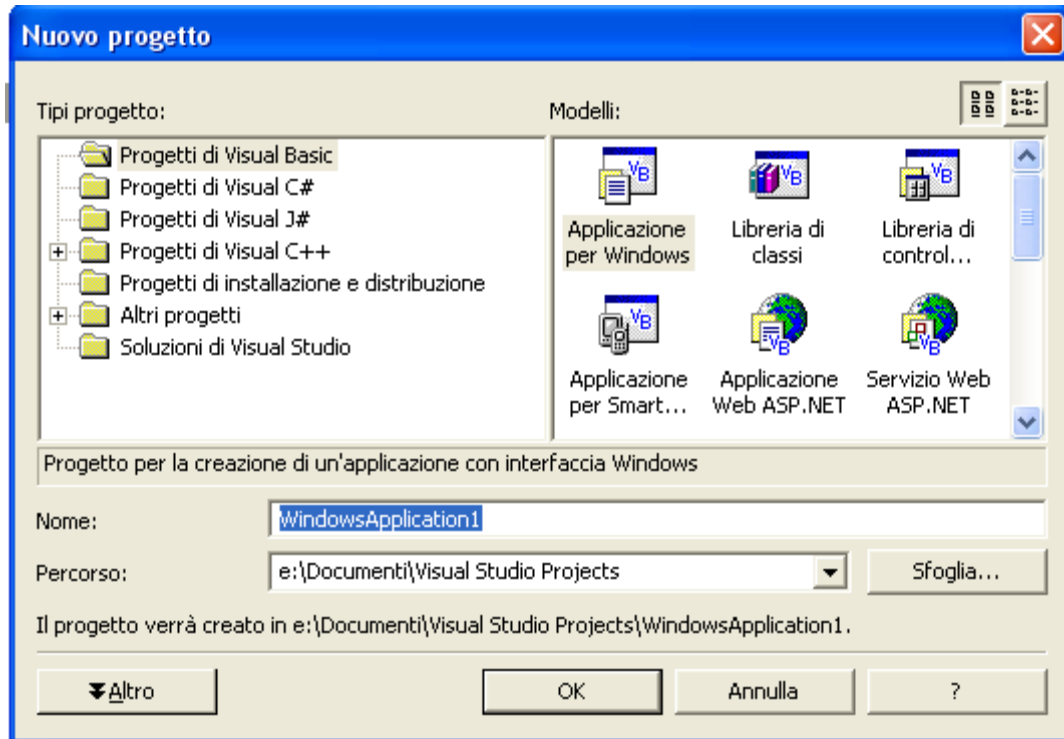
- **un sistema di esecuzione e debug:** Visual Basic permette di eseguire il programma man mano che lo si sta costruendo, in modo da metterlo a punto, ad esempio verificando passo passo il funzionamento (il valore delle variabili, gli effetti delle scelte dell'utente), potendolo fermare, correggere e far proseguire o ripartire. Tutte queste operazioni, assolutamente fondamentali visto che non esiste un programma che funzioni al primo colpo, sono note con il termine di "debug".

- **un compilatore:** una volta che il programma funziona all'interno di Visual Basic, esso deve essere trasformato in un file eseguibile autonomo (o quasi). "Compilare" il programma significa appunto creare tale file. Una volta creato, il file eseguibile non può essere più modificato. Se si trova un errore, si può tuttavia correggerlo nel codice sorgente e poi ricompilarlo.

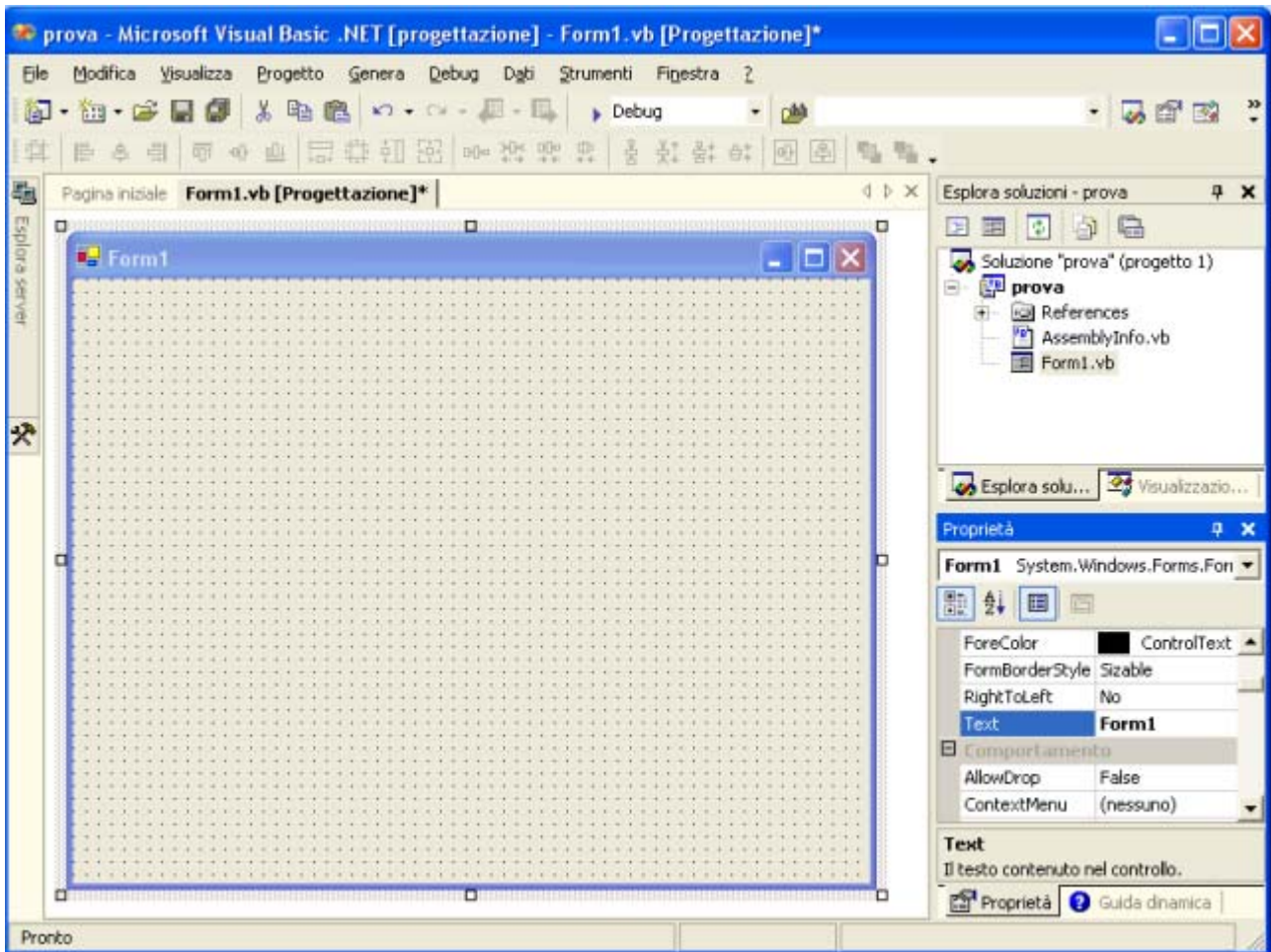
• **una guida in linea:** costituisce un aspetto fondamentale anche per lo sviluppatore più esperto, che non è tale perché "sa tutto a memoria" quanto piuttosto perché sa sempre dove andare a recuperare l'informazione che gli occorre, all'interno della guida che Visual Basic gli offre.

Penso sia ora il caso di provare con un primo esempio. Lanciamo Visual studio e scegliamo nuovo progetto da questa finestra.





Impostiamo un nome per il nostro progetto che, se non cambierete la cartella di destinazione, verrà salvato automaticamente nella cartella Visual Studio Project . Dato l'ok vi troverete davanti alla finestra che rappresenta l'ambiente di programmazione Visual Basic.



Al centro vedete il form. **I forms** costituiscono l'interfaccia dell'applicazione, cioè quello che dell'applicazione "si vede". Esse contengono tutto ciò in cui l'utente può scrivere, premere, selezionare o muovere con il mouse, ecc. È ben difficile che un'applicazione non contenga alcun form, non fosse altro per dare all'utente un modo per vedere cosa l'applicazione fa, ed un pulsante per chiuderla .

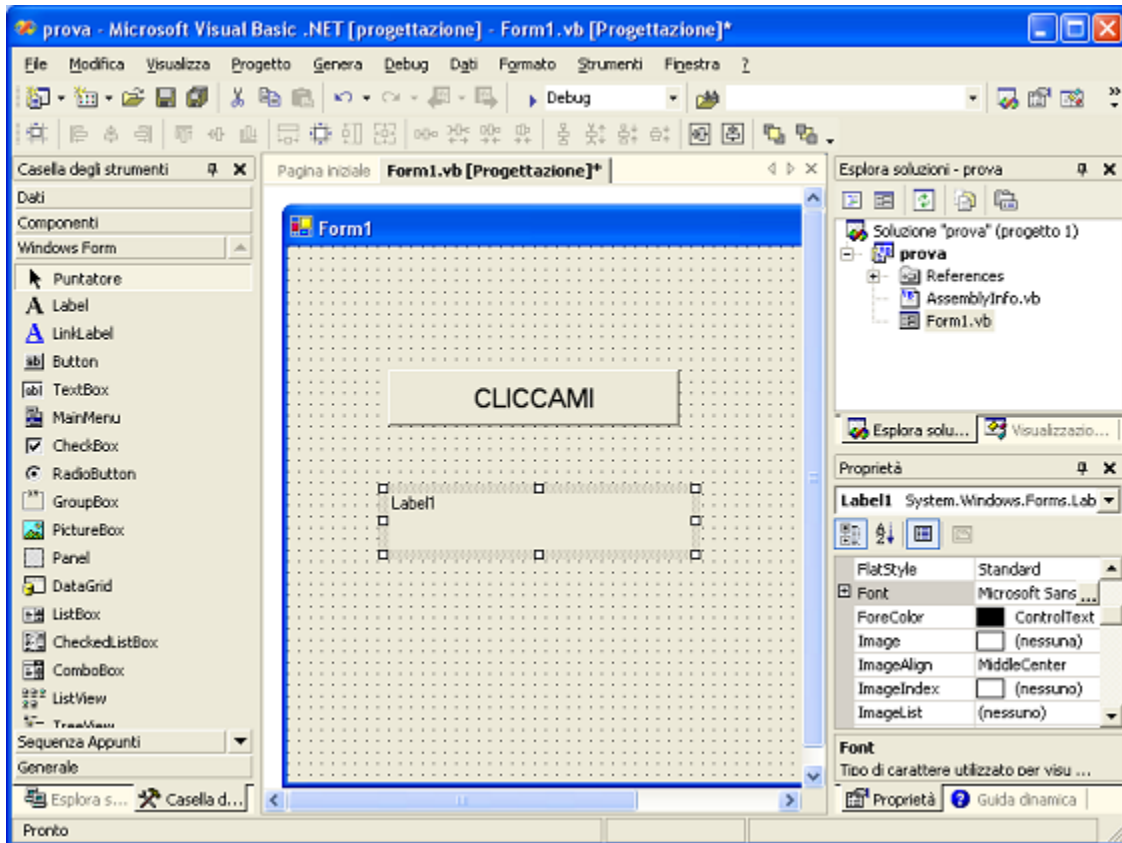
Si noti come esistano almeno due famiglie di form:

il form che sarà l'oggetto principale dell'interfaccia del vostro progetto perchè sarà contenitori di tutti gli oggetti che vorrete mostrare all'utente: **i controlli.**

Nel nostro primo esempio noi useremo 2 tipi di controlli:

- un'etichetta (label), dove si scrive del testo che in genere dà informazioni all'utente (e che l'utente deve solo leggere)
- un pulsante (command button), che è una delle cose più "classiche" di Windows, con su scritto "Cliccami"

I controlli possono essere di tanti altri tipi. In seguito vedremo quelli più consueti in Visual Basic



Per disegnare gli oggetti nel form, occorre selezionare gli appositi strumenti dalla casella degli strumenti a sinistra nell'immagine, nel caso questa non sia visualizzata sul vostro monitor, scegliete **visualizza casella degli strumenti** dal menu. Gli oggetti si disegnano sul form utilizzando il mouse come fosse la nostra matita, quindi tenendo premuto il tasto Sx e rilasciandolo quando il controllo ha raggiunto la dimensione desiderata. Sulla destra in alto la scheda **Esplora soluzioni** e sotto la **scheda delle proprietà**. Quest'ultima evidenzia le proprietà, quindi le caratteristiche degli oggetti del nostro progetto, in particolare di quello evidenziato in quel momento, quindi nell'esempio indica le proprietà della label (etichetta) ammanigliata (selezionata-evidenziata).

Procedendo con il nostro esempio, supponiamo che il nostro obiettivo sia quello di realizzare un programma che permetta all'utente di cliccare un pulsante e ricevere un messaggio di conferma, un saluto.

Le fasi di realizzazione dunque saranno tre: una, la prima l'abbiamo già vista e consiste nel disegno dell'interfaccia, poi si deve continuare con la **definizione delle proprietà**. Si può visualizzare la scheda delle proprietà cliccando con il tasto Dx del mouse l'oggetto e selezionando **Proprietà**. Ogni oggetto ha molte proprietà che ne fissano il colore, le dimensioni, la posizione sulla scheda, il carattere (se compare del testo), e un numero incredibile di altre cose che non sarà necessario sempre andare ad impostare, vediamo quali sono le proprietà che sicuramente dovremo andare a modificare:

- name: definisce il nome dell'oggetto

- text: definisce il valore dell'oggetto, il contenuto che apparirà nella finestra all'utente.

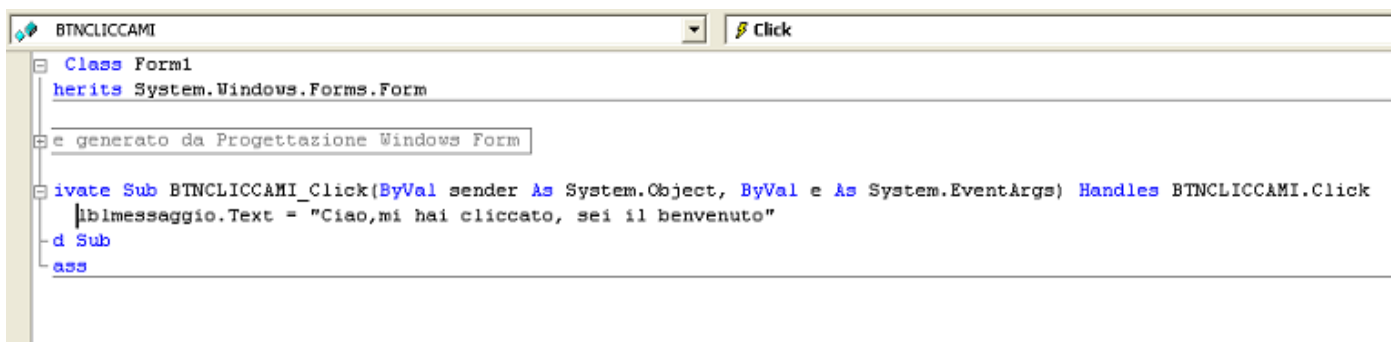
La sintassi per accedere alle proprietà degli oggetti dal codice è:

NomeControllo.NomeProprietà

Ma ora vediamo come arrivare al codice e come collegare questo all'interfaccia. L'evento di premere il pulsante non lo chiamo io, ma ha un nome dato da Visual Basic, che è "Click". Quello che il programma deve fare quando tale evento si verifica è, come abbiamo detto, una procedura evento, il cui nome ha la sintassi:

NomeOggetto_NomeEvento

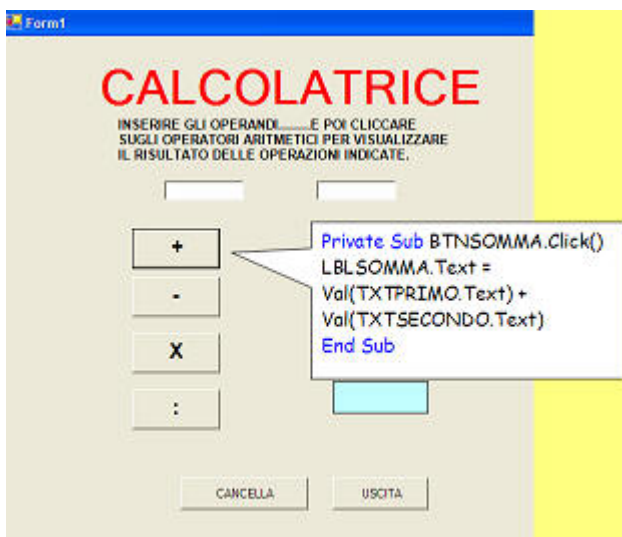
Quindi tutto dipenderà da cosa scriveremo in questa procedura. Intanto per scrivere il codice ci si deve spostare nella scheda codice lasciando quindi la scheda progettazione. Si può accedere alla finestra codice dalla linguetta apposita oppure direttamente facendo doppio click sull'oggetto al quale è collegata la procedura d'evento, nel nostro caso il bottone:



```
BTNCLICCAMI Click
Class Form1
  Inherits System.Windows.Forms.Form
  Generated from Progettazione Windows Form
  Private Sub BTNCLICCAMI_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles BTNCLICCAMI.Click
    lblmessaggio.Text = "Ciao,mi hai cliccato, sei il benvenuto"
  End Sub
End Class
```

Quella che vediamo in questo esempio è una banale istruzione di assegnazione, cioè assegniamo alla proprietà text della label il contenuto indicato tra apici.

Vediamo ora un secondo esempio in cui utilizzeremo un altro controllo: la casella di testo. **Nel text box** l'utente può scrivere direttamente in fase di esecuzione oltre che il programmatore come per la label, può scrivere "via codice". Il fatto che



l'utente possa scrivere nella casella di testo, fa di questo controllo un utile strumento da utilizzare per l'immissione degli input dall'utente. Se ad esempio volessimo simulare il funzionamento di una calcolatrice, come vedete nell'immagine,

avremmo bisogno di due caselle di testo che l'utente utilizzerà per immettere i numeri che saranno oggetto

dell'operazione scelta attraverso l'evento Click dei relativi pulsanti. A ciascuno di quei pulsanti sarà collegato il codice indicato nel fumetto, che si differenzierà solo per l'operatore aritmetico + - * /. Anche in questo caso si tratta di una istruzione di assegnazione che attribuisce alla proprietà text della label la somma dei valori numerici delle proprietà text delle due caselle di testo. Per questo motivo è stata usata la funzione **VAL()** questa *restituisce il valore numerico di una stringa*. All'inverso la funzione **Cstr** *converte in stringa un valore numerico*.

Durante la fase di progettazione dell'applicazione – quando cioè ci troviamo all'interno di Visual Basic – abbiamo a disposizione tre modalità di Visual Basic stesso, e cioè:

* **la modalità progettazione (design)** in cui l'applicazione viene costruita, si disegnano le schede e i controlli, si impostano le proprietà che si desidera fissare e si scrive il codice;

* **la modalità esecuzione (run)** in cui si esegue l'applicazione come se fosse finita, pur restando all'interno di Visual Basic;

* **la modalità interruzione (break)** in cui si sospende temporaneamente l'esecuzione dell'applicazione in modo da fare tipicamente una delle due seguenti cose:

– verificare il contenuto di alcune variabili o il valore di alcune proprietà, tramite l'apposita "finestra di verifica" (debug window) in modo da cercare di capire se variabili e proprietà contengono davvero quello che ci si aspetta;

– modificare parti del codice (in fase di interruzione è possibile modificare buona parte del codice; se si dovesse incorrere in qualcosa che non può essere modificato in questa fase – per esempio la dichiarazione di nuove variabili globali

– Visual Basic avverte che sarà necessario fermare l'esecuzione e ripartire dall'inizio).

Attraverso questi tre pulsanti possiamo passare da una modalità all'altra:



Il primo pulsante passa da design a run, cioè fa partire il programma, come se fosse un eseguibile (scelta rapida con F5).

Il secondo pulsante impone una pausa (da run a break).

Il terzo pulsante ferma il programma (sia esso in run o in break) e lo riporta in design, come se l'utente scegliesse un (ipotetico) pulsante Chiudi