

[Home Page](#) [Informazioni](#) [Aiuto](#)

## Creare una casella di testo che accetti soltanto numeri

[http://www.vbsimple.net/howto/ht\\_000.htm](http://www.vbsimple.net/howto/ht_000.htm)

- [Una semplice TextBox numerica.](#)
- [Più TextBox numeriche in una matrice di controlli.](#)
- [Implementazione della classe NumTextBox.](#)
- [Rendere una TextBox numerica mediante l'utilizzo dell'API.](#)

### Una semplice TextBox numerica

Difficoltà: 2 / 5

Quante volte vi siete trovati con l'esigenza di inserire delle caselle di testo all'interno di un form, per permettere all'utente l'immissione di un numero intero quale l'anno di nascita o altro?

Il problema è molto semplice da analizzare: ogni volta che l'utente preme un tasto invia un carattere alla casella di testo; è pertanto necessario controllare se esso è un tasto numerico oppure è una lettera o un simbolo. Se il tasto premuto non è un numero è necessario eliminare il carattere prima che venga visualizzato nella casella di testo.

Tutte le caselle di testo possiedono tre [eventi](#) relativi alle pressioni dei tasti da tastiera: questi eventi sono **KeyPress**, **KeyDown** e **KeyUp**. Il primo corrisponde all'invio del carattere alla casella di testo, il secondo indica la pressione di un tasto sulla casella, mentre il terzo indica il rilascio di un tasto premuto.

Ogni volta che viene premuto un carattere speciale, non stampabile come ad esempio le freccette, scattano gli eventi *KeyDown* e *KeyUp*. Il primo alla pressione del tasto mentre il secondo al rilascio di tale tasto.

Se il tasto premuto è un carattere stampabile ovvero una lettera o un numero, vengono generati i tre eventi nell'ordine *KeyDown*, *KeyPress* e *KeyUp*. *KeyDown* viene generato sempre e comunque per qualsiasi tasto. L'evento *KeyPress* scatta nel momento in cui deve essere inviato una lettera o un numero alla casella di testo. Il *KeyUp* scatta sempre al rilascio di qualunque tasto.

La soluzione più semplice al problema consiste nell'utilizzare l'evento *KeyPress*, la cui sintassi è la seguente:

```
Private Sub Oggetto_KeyPress(KeyAscii As Integer)
```

Oggetto indica il nome di una casella di testo. Questo evento passa un parametro *KeyAscii* [per riferimento](#) ovvero ogni cambiamento fatto a tale valore sarà comunicato alle funzioni che utilizzano il valore *KeyAscii*. Il parametro *KeyAscii* indica il codice ASCII del

carattere che verrà inviato alla casella di testo.

Per permettere l'immissione di soli numeri all'interno di una casella di testo ci basterà controllare il parametro `KeyAscii` dell'evento `KeyPress`. Se il valore di tale parametro non è compreso nell'insieme dei caratteri numerici il carattere verrà scartato. Per scartare un carattere è necessario impostare il valore di `KeyAscii` a 0. Vediamo il codice da utilizzare:

```
1. Private Sub Text1_KeyPress(KeyAscii As Integer)
2.     If KeyAscii < Asc("0") Or KeyAscii > Asc("9") Then
3.         KeyAscii = 0
4.         Beep
5.     End If
6. End Sub
```

La riga che effettua il controllo del tasto premuto è la seconda. Essa controlla se il tasto premuto `KeyAscii` è minore del codice ASCII del numero 0 oppure è maggiore del codice ASCII del numero 9. In tal caso scarta il tasto impostando il `KeyAscii` a 0 e manda un avvertimento sonoro attraverso l'istruzione `Beep`.

L'istruzione `Asc` ci restituisce il codice ASCII del carattere indicato, quindi `Asc("0")` ci riporta il codice ASCII del numero 0, ovvero il valore 48. `Asc("9")` ci riporta il codice ASCII del numero 9, quindi il valore 57.

Per questo motivo avremmo anche potuto scrivere:

```
If KeyAscii < 48 Or KeyAscii > 57 Then
```

Avremmo però peggiorato di molto la leggibilità del codice.

Una soluzione alternativa consiste nello sfruttamento dell'[enumerazione](#)  [KeyCodeConstants](#).

```
If KeyAscii < vbKey0 Or KeyAscii > vbKey9 Then
```

In quest'ultimo caso il codice è maggiormente leggibile di quello precedente.

## Più TextBox numeriche in una matrice di controlli

Difficoltà:  3 / 5

Lo svantaggio principale della soluzione appena vista è che per ogni singola casella di testo numerica sarà necessario riscrivere lo stesso codice.

La soluzione più semplice a questo secondo problema consiste nell'utilizzare le [matrici](#) di controlli viste nel corso Intermedio. L'unica differenza funzionale a livello di codice consiste nella sintassi dell'evento `KeyPress`:

```
Private Sub Oggetto_KeyPress(Index As Integer, KeyAscii As Integer)
```

Come si vede, oltre il parametro `KeyAscii` abbiamo anche un parametro `Index` che indica l'indice del controllo che ha eseguito l'evento `KeyPress` all'interno della matrice. Il valore di questo parametro corrisponde al valore della Proprietà `Index` della casella di testo che ha

ricevuto la pressione del tasto.

Se inserissimo all'interno di un form due o più TextBox, verrebbe comunque richiamata questa routine di evento. Per cui a livello di codice non avremo nessun cambiamento:

```
1. Private Sub Text2_KeyPress(Index As Integer, KeyAscii As Integer)
2.     If KeyAscii < Asc("0") Or KeyAscii > Asc("9") Then
3.         KeyAscii = 0
4.         Beep
5.     End If
6. End Sub
```

L'utilizzo delle matrici consente la riutilizzabilità di segmenti di codice e l'enorme vantaggio di poter creare tanti controlli dello stesso tipo rispondenti tutti a dei comportamenti comuni.

---

## Implementazione della classe NumTextBox

Difficoltà:  4 / 5

Ina soluzione più professionale e più flessibile delle due viste prima è quella detta [Multicasting](#). Genereremo una nuova classe di nome **NumTextBox** che amplia le funzionalità delle normali TextBox. La classe consiste di poche e semplici righe. Tuttavia non supporta l'utilizzo di elementi appartenenti a matrice.

Ecco tutta la classe:

```
1. Public WithEvents CasellaTesto As TextBox
2.
3. Private Sub CasellaTesto_KeyPress(KeyAscii As Integer)
4.     If KeyAscii < Asc("0") Or KeyAscii > Asc("9") Then
5.         KeyAscii = 0
6.         Beep
7.     End If
8. End Sub
9.
10. Private Sub Class_Terminate()
11.     Set CasellaTesto = Nothing
12. End Sub
```

La prima riga è il cuore di questa classe. Essa definisce una nuova variabile di tipo TextBox con eventi. L'istruzione *WithEvents* permette l'utilizzo di eventi all'interno di un codice dove non sono stati allocati gli oggetti.

Quello che fa la prima riga è preparare solo una variabile, non creare un nuovo oggetto in memoria e inserirlo all'interno del form. Non sarà creato nulla di visibile con questa classe.

**CasellaTesto** è la variabile pubblica che gestirà gli eventi delle caselle di testo che utilizzeranno tale classe. Abbiamo quindi la routine dell'evento *KeyPress* relativa alla variabile *CasellaTesto*.

Com'è intuibile dai parametri di questa routine, non è previsto l'utilizzo di elementi appartenenti a matrici di controlli; manca, infatti, il parametro *Index*.

All'interno di questa routine c'è il solito codice visto in precedenza.

La classe si chiude con la funzione *Class\_Terminate* eseguita nel momento in cui

un'istanza della classe NumTextBox viene [deallocata](#). Questa funzione rende nuovamente disponibile la memoria occupata dalla variabile CasellaTesto.

Passiamo ora all'interfacciamento della classe NumTextBox con il form contenente le caselle di testo normali. Abbiamo due caselle di testo di nome **Text3\_1** e **Text3\_2**, non in matrice per i motivi già accennati.

```
1. Private CasellaNumBox1 As New NumTextBox
2. Private CasellaNumBox2 As New NumTextBox
3.
4. Private Sub Form_Load()
5.     Set CasellaNumBox1.CasellaTesto = Me.Text3_1
6.     Set CasellaNumBox2.CasellaTesto = Me.Text3_2
7. End Sub
8.
9. Private Sub Form_Terminate()
10.    Set CasellaNumBox1 = Nothing
11.    Set CasellaNumBox2 = Nothing
12. End Sub
```

Le prime due righe definiscono all'interno del form due variabili di nome **CasellaNumBox1** e **CasellaNumBox2** ed allocano la memoria necessaria all'istanza della classe.

Nota l'uso della parola chiave *New* che permette l'istanza di una classe all'interno della stessa dichiarazione della variabile.

Abbiamo sfruttato anche la funzione Form\_Load per attivare prima possibile la funzionalità della classe NumTextBox. Invero fintanto che non colleghiamo le nostre caselle di testo normali con la variabile [membro](#) CasellaTesto della classe, non saranno generati gli eventi all'interno della classe.

All'interno di questa funzione abbiamo l'assegnazione del membro CasellaTesto delle singole istanze della classe ad una casella di testo. CasellaNumBox1 gestirà la casella di testo di nome Text3\_1, mentre CasellaNumBox2 gestirà la pressione dei tasti della casella Text3\_2.

L'implementazione si conclude con la deallocazione delle due istanze CasellaNumBox1 e CasellaNumBox2. La mancata deallocazione può provocare rallentamenti del sistema e l'occupazione di blocchi di memoria da variabili inaccessibili.

Con quest'ultima soluzione possiamo assegnare delle funzioni comuni a determinati oggetti, senza essere obbligati a formare una matrice di controlli.

Inoltre abbiamo ancora a disposizione le routine dell'evento KeyPress delle singole caselle di testo; infatti all'interno del codice del form non abbiamo nessuna Text3\_1\_KeyPress.

---

## Rendere una TextBox numerica mediante l'utilizzo dell'API

Difficoltà:  3 / 5

L'ultima soluzione è la più efficiente e semplice da attuare. Infatti è possibile definire una TextBox numerica mediante l'impostazione di uno stile mediante l'API.

L'esempio in questione utilizza due funzioni API: *GetWindowLong* e *SetWindowLong*. La prima servirà per leggere un dato valore dalla casella di testo e la seconda per cambiare tale valore.

```
1. Private Declare Function GetWindowLong Lib "USER32" Alias "GetWindowLongA" (ByVal  
   hwnd As Long, ByVal nIndex As Long) As Long  
2. Private Declare Function SetWindowLong Lib "USER32" Alias "SetWindowLongA" (ByVal  
   hwnd As Long, ByVal nIndex As Long, ByVal dwNewLong As Long) As Long  
3. Private Const GWL_STYLE = (-16)  
4. Private Const ES_NUMBER As Long = &H2000
```

La riga 3 definisce una costante di nome **GWL\_STYLE** che verrà utilizzata dalle due funzioni API per ottenere lo stato (il comportamento) di una finestra.

La riga 4 definisce una costante di nome **ES\_NUMBER** e verrà utilizzata per impostare lo stile di casella numerica.

```
5. Private Sub Form_Load()  
6.     Dim style As Long  
7.     style = GetWindowLong(Text3.hwnd, GWL_STYLE)  
8.     SetWindowLong Text3.hwnd, GWL_STYLE, style Or ES_NUMBER  
9. End Sub
```

All'interno dell'evento Load del form richiameremo le due funzioni per impostare lo stile della casella di testo Text3.

Alla riga 7 viene letto il valore iniziale dello stile della finestra il cui [handle](#) è Text3.hwnd e tale valore viene memorizzato nella variabile **style**.

Alla riga successiva viene impostato il nuovo stile della casella di testo impostando un particolare [bit](#) dello stile mediante l'[operazione di OR](#).

L'utilizzo delle classi sta alla base della [programmazione ad oggetti](#) vista nella [terza lezione del corso Base](#). Le classi permettono la riutilizzabilità di determinati segmenti di codice.

[Fibia FBI](#) e [Giuseppe Della Bianca](#)

Ultima soluzione di Lawrence Lebowsky

14 Novembre 2000



[Torna all'indice degli HowTo](#)

---