



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

Creare controlli in fase di esecuzione

http://www.vbsimple.net/howto/ht_015.htm

Difficoltà:  2 / 5

Per taluni progetti può essere molto utile creare una serie di controlli; in questi casi basta creare una matrice di controlli in [fase di progettazione](#).

Ma se in fase di progettazione fosse impossibile determinare il numero di controlli necessari all'utente in un determinato momento?

Ipotizziamo un semplice programmino: un'agenda legge dei dati da un database. Il programma apre la tabella apposita e, per ogni campo contenuto in essa, mostra il contenuto in una casella di testo. In fase di progettazione il programmatore potrebbe contare il numero di campi della tabella e creare le caselle di testo, ma il progetto finirebbe lì, non potendo essere migliorato senza modificare il programma in questione.

Invece si potrebbe lasciare all'utente una scelta in più: il programma, in fase di esecuzione, effettua il conteggio dei campi presenti all'interno della tabella, ed in base al numero, crea i controlli (le caselle di testo) necessari.

Per effettuare quest'operazione sarà necessario utilizzare le matrici di controlli: basterà inserire almeno un controllo e dargli indice (proprietà *Index*) 0.

Ci penserà un'altra funzione a fare la clonazione di tale controllo per tanti elementi richiesti.

Vediamo come effettuare queste operazioni.

Inseriamo all'interno di un form una *TextBox*  di nome **CasellaTesto** ed impostiamo per essa la proprietà *Index* a 0 e la proprietà *Visible* a False.

Un po' più a destra inseriamo una *Label*  di nome **LabelNumero** contenente il testo "*Numero controlli*".



Subito sotto essa aggiungiamo un'altra *TextBox*  di nome **NumeroControlli**.

Aggiungiamo, infine, un semplice *CommandButton*  di nome **PulsanteCrea**.

L'utilizzo del programma è molto semplice: l'utente deve inserire un numero nella casella **NumeroControlli** e premere il pulsante **PulsanteCrea** per vedere apparire la serie di controlli, la cui base è **CasellaTesto**.

Apriamo la finestra del codice ed inseriamo questa semplice funzione di nome **Clona**:

```
1. Private Sub Clona(NumContr As Integer)
2.   Dim FirstContr As Integer
3.   Dim LastContr As Integer
4.   Dim ContrNum As Integer
```

La funzione richiede un parametro di nome NumContr; esso determina il numero di elementi da creare.

All'interno della funzione utilizziamo tre variabili:

FirstContr che indica l'indice del primo elemento, ovvero 0

LastContr che indica l'indice dell'ultimo controllo

ContrNum utilizzata per effettuare i calcoli ed i cicli

```
5. With CasellaTesto
6.   FirstContr = .LBound
7.   LastContr = .UBound
8.
```

Poiché all'interno del codice è necessario effettuare calcoli prendendo come oggetto la matrice di controlli CasellaTesto, abbiamo inserito alla riga 5 un'istruzione *With* che assume come oggetto predefinito la matrice CasellaTesto. Per accedere ad un [membro](#) di essa sarà sufficiente scrivere il nome del membro, precedendolo con un punto.

Alla riga 6 troviamo l'indice del primo elemento della matrice e lo memorizziamo nella variabile **FirstContr**. Lo ricordiamo, il primo elemento è 0. Questa scrittura serve soltanto per mostrare come ricavare il primo elemento di una matrice.

Alla riga 7 effettuiamo il calcolo degli elementi. La variabile **LastContr** conterrà l'indice dell'ultimo elemento della matrice.

```
9.   For ContrNum = LastContr To NumContr + FirstContr Step -1
10.    If ContrNum = FirstContr Then
11.      .Item(ContrNum).Visible = False
12.    Else
13.      Unload .Item(ContrNum)
14.    End If
15.  Next ContrNum
```

Alla riga 9 inizia un ciclo per scaricare i controlli inutilizzati (riga 13), qualora ne esistessero, e per nascondere il primo elemento (riga 11).

Nota che questo ciclo non elimina tutti gli oggetti, per ricostruirli in seguito, ma scarica soltanto quelli superflui alla richiesta. Se esistessero 10 elementi ed il parametro **NumContr** contenesse il valore 4, sarebbero scaricati soltanto i 6 controlli superflui.

```
16.   If NumContr > 0 Then .Item(FirstContr).Visible = True
```

Se il parametro **NumContr** è maggiore di 0 (riga 16), ciò significa che vogliamo creare almeno un elemento. Poiché il ciclo precedente non scarica tutti gli elementi, ma lascia in memoria il primo elemento, sarà necessario renderlo visibile perché nascosto dalla riga 11.

```
17.   For ContrNum = LastContr To FirstContr + NumContr - 1
18.     If ContrNum > LastContr Then Load .Item(ContrNum)
19.   Next ContrNum
```

Questo piccolissimo ciclo carica nella matrice uno per uno i controlli mancanti, fino a quando il numero di controlli **ContrNum** non raggiunge il numero richiesto con **NumContr**.

Essendo gli elementi a base variabile, sarà necessario contare fino a *FirstContr* + *NumContr* - 1.

```

20.     If NumContr > 0 Then
21.         For ContrNum = FirstContr To .UBound
22.             .Item(ContrNum).Text = "Controllo N° " & ContrNum
23.             If ContrNum > FirstContr Then
24.                 .Item(ContrNum).Visible = True
25.                 .Item(ContrNum).Top = .Item(ContrNum - 1).Top + .Item(ContrNum -
1).Height
26.                 .Item(ContrNum).TabIndex = .Item(ContrNum - 1).TabIndex + 1
27.             End If
28.         Next ContrNum
29.     End If
30. End With
31. End Sub

```

Alla riga 20 viene controllato se esiste almeno un elemento; in caso negativo il ciclo salta alla riga 29 ed la funzione termina.

Quando esiste almeno un elemento della matrice, inizia un ciclo che imposta i singoli controlli (riga 21).

Il ciclo innanzitutto imposta la proprietà *Text* di ogni controllo con la scritta "**Controllo N°**", seguita dal numero dell'indice.

A questo punto (riga 23) per tutti gli elementi il cui indice sia successivo al primo, vengono impostate tre proprietà. Alla riga 24 viene reso il controllo visibile mediante la proprietà *Visible* impostata a True; alla riga 25 viene calcolata la posizione (*Top*) prendendo come riferimento l'elemento precedente ed incrementando quella coordinata dell'altezza (*Height*) dell'elemento precedente. Questo assicura che tutte le caselle di testo siano in colonna, una sotto l'altra.

Infine alla riga 26 viene calcolato il valore *TabIndex* della casella di testo, prendendo come riferimento il valore della casella precedente. Per cui, quando utilizzeremo il tasto TAB per spostarci da un controllo all'altro, le caselle di testo saranno consecutive.

La funzione finisce qui. Viene ovviamente chiusa l'istruzione *With* con un *End With*.

Prima di provare il programma dovremmo scrivere un piccola funzioncina che richiama la funzione *Clona*.

```

33. Private Sub PulsanteCrea_Click()
34.     If Val(NúmeroControlli.Text) >= 0 Then Clona NúmeroControlli.Text
35. End Sub

```

All'interno dell'evento *Click* del pulsante **PulsanteCrea** abbiamo inserito tale funzione.

Se il valore impostato nella casella **NúmeroControlli** è maggiore o uguale a 0, verrà richiamata la funzione *Clona* e passato come parametro tale numero.

Questo controllo è necessario poiché non ha senso tentare di creare un numero di elementi negativo e la funzione genererebbe un errore.

Possiamo ora provare il programma.

Basta inserire un numero nella casella apposita e premere il pulsante "**Crea controlli**" per vedere apparire i controlli richiesti.

Naturalmente è estremamente importante non esagerare nell'impostare il numero di controlli. Un numero grande (1000-5000) può occupare tutte le risorse del sistema, rendendo il computer instabile.



La funzione è molto comoda, ma la limitazione principale riguarda l'impossibilità di conoscere a priori le richieste minime per il programma, in quanto gli oggetti vengono allocati e deallocati nel momento in cui viene effettuata la richiesta.

L'altro grosso limite è legato alla struttura di Windows 95/98/ME. La memoria è strutturata in aree. Per cui, anche se abbiamo una grossissima quantità di memoria, ad esempio 128 o 256 MB, la funzione che crea i controlli sarà comunque limitata ad una piccola area di questa memoria.

In questo esempio abbiamo impostato il primo elemento con indice 0, ma la funzione è progettata per funzionare anche con indici superiori. Per esempio se il primo elemento ha indice 2 e viene richiesta creazione di 5 controlli, il primo elemento verrà nuovamente mostrato e verranno creati altri 4 elementi, con indice dal 3 al 6.

[Giuseppe Della Bianca](#)
6 Gennaio 2001



[Torna all'indice degli HowTo](#)
