



Creazione di pulsanti di sistema

http://www.vbsimple.net/howto/ht_001.htm

- [I pulsanti di sistema.](#)
- [Il pulsante di chiusura di un form.](#)
- [Il pulsante di riduzione ad icona.](#)
- [Il pulsante di ingrandimento.](#)
- [Il pulsante di ripristino.](#)
- [Il menu di sistema](#) (senza API e con API).

I pulsanti di sistema.

Difficoltà: 2 / 5

Vedremo in questo HowTo come è possibile collegare dei pulsanti normali ai pulsanti di sistema di un Form. Posizioneremo sopra un form 5 pulsanti al cui click verranno eseguite le operazioni di chiusura del form , la riduzione ad icona , la massimizzazione del form , il ripristino del suo stato normale e l'apertura del menu di sistema di un form.

Il pulsante di chiusura di un form.

Le operazioni da eseguire sono facilissime.

Nel momento in cui clicchiamo il pulsante di chiusura scatta l'evento *Unload* del form. Per cui al nostro scopo serve una funzione per far scattare tale evento. Il comando più adatto è l'istruzione **Unload** seguita dal nome di un form. `Unload Me` provoca la chiusura del form dentro il quale quest'istruzione viene scritta.

Inseriamo un pulsante e denominiamolo **PulsanteChiudi** e nella routine che gestisce l'evento Click scriviamo `Unload Me`. Il primo pulsante è completo.

Il pulsante di riduzione ad icona.

Passiamo al pulsante che riduce ad icona il form. Nel momento in cui viene cliccato, il valore della proprietà *WindowState* del form assume il valore 1, corrispondente al valore della [costante](#) `vbMinimized`.

Nel momento in cui vorremo ridurre ad icona un form ci basterà impostare il valore della proprietà *WindowState* a 1 oppure a `vbMinimized`.

Inseriamo nel nostro form un pulsante e diamogli il nome di **PulsanteRiduci**. All'interno dell'evento Click scriviamo `WindowState=vbMinimized`. Secondo pulsante completo.


Il pulsante di ingrandimento.

Il terzo pulsante è quello di ingrandimento, detto anche pulsante di massimizzazione del

form. Al click sopra di esso, la proprietà *WindowState* riceve il valore 2 ovvero la costante *vbMaximized*.

Inseriamo un pulsante di nome **PulsanteIngrandisci** e scriviamo all'interno della routine dell'evento Click `WindowState=vbMaximized`. Anche questo è finito.

Il pulsante di ripristino.

Il penultimo pulsante è il pulsante di ripristino  che appare ogni volta che si ingrandisce la finestra al massimo. Anche per questo stato si sfrutta la proprietà *WindowState*.

L'impostazione d'essa al valore 0 oppure alla costante *vbNormal* provoca il ritorno alla dimensione originale del form, prima che venisse effettuata la massimizzazione.

Inseriamo il pulsante **PulsanteRipristina** e scriviamo all'interno della routine dell'evento Click `WindowState=vbNormal`. Ora che questo è completo possiamo passare all'ultimo.

Il menu di sistema.

Questo pulsante è il più complesso. Nei progetti Visual Basic in genere ha l'icona generica dei form  ma essa può essere cambiata tramite la proprietà *Icon* del form.

All'interno di questo menu sono disponibili le funzioni le funzioni degli altri pulsanti di sistema e le possibilità di ingrandimento e spostamento del form.

Non esiste nessun evento e nessuna proprietà che regolano la pressione di tale menu.

Inoltre tutti gli altri pulsanti di sistema potranno funzionare anche se essi saranno nascosti tramite le proprietà *ControlBox*, *MinButton* e *MaxButton* del form, mentre se il menu di sistema è nascosto non potrà essere sfruttato con il metodo che vedremo fra poco.

Non esistendo elementi all'interno del linguaggio useremo un trucchetto per forzare la comparsa di tale menu. In tutti i programmi per Windows premendo la combinazione di tasti ALT SPAZIO si apre il menu di sistema.

Faremo in modo che al click del mouse sopra il pulsante **PulsanteMenuSys** venga inviata tale combinazione di tasti. Per il nostro scopo utilizzeremo la funzione *SendKeys*.

All'interno della routine dell'evento Click scriveremo `SendKeys "% "`.

Nota bene che dopo il simbolo di percentuale segue uno spazio.

L'istruzione si aspetta una stringa come parametro dei tasti da inviare. Il simbolo % indica la pressione del tasto ALT.

Il menu di sistema attraverso l'uso dell'API.

Difficoltà:    3 / 5

Esiste un'altra soluzione al problema di far apparire il menu di sistema, ma anch'essa è sottoposta alla limitazione imposta dalla proprietà *ControlBox* del form.

Per questa soluzione utilizzeremo due funzioni [API](#):

1. Private Declare Function GetSystemMenu Lib "USER32" (ByVal hwnd As Long, ByVal bRevert As Long) As Long
2. Private Declare Function TrackPopupMenu Lib "USER32" (ByVal hMenu As Long, ByVal wFlags As Long, ByVal X As Long, ByVal Y As Long, ByVal nReserved As Long, ByVal hwnd As Long, ByVal lprc As Any) As Long
- 3.

La prima dichiarazione è la funzione **GetSystemMenu** che riporta l'[handle](#) del menu di sistema della finestra il cui handle è il parametro `hwnd`.

La seconda dichiarazione è la funzione **TrackPopupMenu** che mostra un menu sullo schermo. È molto simile all'istruzione Visual Basic *PopupMenu*, ma invece di un oggetto Menu richiede il suo handle e le coordinate sono obbligatorie.

Inseriamo un nuovo pulsante di nome **PulsanteMenuSysAPI** e scriviamo questo codice all'interno della routine dell'evento Click:

```
4. Dim COORDX As Long
5. Dim COORDY As Long
6. Dim hMenu As Long
7.
8. COORDX = Me.Left / Screen.TwipsPerPixelX + 2
9. COORDY = Me.Top / Screen.TwipsPerPixelY + 21
10. hMenu = GetSystemMenu(Me.hwnd, False)
11. TrackPopupMenu hMenu, 0, COORDX, COORDY, 0, Me.hwnd, 0&
```

Le prime due righe dichiarano due variabili che conterranno le coordinate dove apparirà il menu. La terza riga invece è una variabile che conterrà l'handle del menu di sistema.

Poiché la funzione API *TrackPopupMenu* vuole le coordinate del menu in Pixel, mentre Visual Basic regolarmente utilizza come unità di misura il Twip, è necessario svolgere un semplice operazione di conversione da Twips a Pixel.

All'interno dell'oggetto Screen abbiamo le proprietà **TwipsPerPixelX** e **TwipsPerPixelY**, le quali indicano rispettivamente il numero di Pixel per ogni Twip sull'asse X e quelli sull'asse Y.

Per convertire un valore da Twips a Pixel è necessario dividere il valore in Twips per il numero di Pixel per Twip.

Alla riga 8 abbiamo il calcolo `Me.Left / Screen.TwipsPerPixelX`.

Tale operazione ci restituisce la coordinata `Left` del form in Pixel. A questa coordinata aggiungiamo 2 ovvero la dimensione in pixel del bordo sinistro, in modo da evitare che il menu venga disegnato sopra di esso.

La nona riga effettua lo stesso calcolo per la coordinata `Top` del form.

Al risultato aggiungiamo 21 che è la somma dei pixel occupati dalla barra del titolo più il bordo superiore.

Alla riga 10 utilizziamo la funzione *GetSystemMenu* per ricevere l'handle del menu di sistema del form `Me` e memorizzarlo nella variabile `hMenu`.

La riga 11 completa la routine con l'istruzione *TrackPopupMenu* che fa apparire un menu di cui abbiamo l'handle. Alla funzione API forniamo l'handle `hMenu`, le due coordinate `COORDX` e `COORDY`, l'handle del form cui si riferisce (`Me.hwnd`) più alcune costanti 0.

Il nostro problema finisce qui. Segue un riepilogo dell'intero codice del form:

```
1. Private Declare Function TrackPopupMenu Lib "USER32" (ByVal hMenu As Long, ByVal
  wFlags As Long, ByVal X As Long, ByVal Y As Long, ByVal nReserved As Long, ByVal
  hwnd As Long, ByVal lprc As Any) As Long
2. Private Declare Function GetSystemMenu Lib "USER32" (ByVal hwnd As Long, ByVal
```

```
        bRevert As Long) As Long
3.
4. Private Sub PulsanteChiudi_Click()
5.     Unload Me
6. End Sub
7.
8. Private Sub PulsanteIngrandisci_Click()
9.     WindowState = vbMaximized
10. End Sub
11.
12. Private Sub PulsanteMenuSys_Click()
13.     SendKeys "% "
14. End Sub
15.
16. Private Sub PulsanteMenuSysAPI_Click()
17.     Dim COORDX As Long
18.     Dim COORDY As Long
19.     Dim hMenu As Long
20.
21.     COORDX = Me.Left / Screen.TwipsPerPixelX + 2
22.     COORDY = Me.Top / Screen.TwipsPerPixelY + 21
23.     hMenu = GetSystemMenu(Me.hwnd, False)
24.     TrackPopupMenu hMenu, 0, COORDX, COORDY, 0, Me.hwnd, 0&
25. End Sub
26.
27. Private Sub PulsanteRiduci_Click()
28.     WindowState = vbMinimized
29. End Sub
30.
31. Private Sub PulsanteRipristina_Click()
32.     WindowState = vbNormal
33. End Sub
```

[Fibia FBI](#)

18 Novembre 2000

[Torna all'indice degli HowTo](#)
