Visual Basic Simple ★★★ Terzo anno

Home Page
Informazioni
Aiuto

Spostare un controllo a runtime

http://www.vbsimple.net/howto/ht_007.htm

Difficoltà: 3 / 5

Avete presente l'IDE di Visual Basic?
In fase di progettazione si possono *"prendere"* dei controlli e trascinarli per lo schermo. Avete mai pensato di fare questo in fase di esecuzione? Dare all'utente la possibilità di scegliersi la posizione dei controlli?

La soluzione più semplice ed anche migliore consiste nell'utilizzare l'API ed i messaggi delle finestre. Per un discorso più approfondito sui messaggi vedi la sezione informazioni aggiuntive.

Nel nostro esempio inseriremo tre controlli (tanto per dare l'idea e provare il nostro codice): una ***DirListBox*** di nome Dir1, un ***CommandButton*** di nome Command1 ed una ***TextBox*** di nome Text1. Per il momento lasciamoli lì, inutilizzati.

Per effettuare lo spostamento di questi controlli utilizzeremo due dichiarazioni API: la *ReleaseCapture* e la *SendMessage*. La prima serve per rilasciare il cursore del mouse; chiariamo questo concetto: nel momento in cui l'utente clicca con il tasto sinistro del mouse sopra un controllo, quell'oggetto diventa l'oggetto attivo. Questo fa sì che tutti i messaggi inviati (il movimento del mouse, il click, il rilascio) saranno diretti a tale oggetto.

La funzione *ReleaseCapture* rilascia l'oggetto selezionato in modo da redirigere i messaggi nella maniera più idonea. Non è necessario ripristinare l'oggetto attivo perché di questo si occuperanno tanti altri eventi.

La funzione *SendMessage* serve per inviare un messaggio specifico ad una determinata finestra. Nel nostro esempio invieremo il messaggio che permette lo spostamento del controllo.

Passiamo al nostro codice; innanzitutto dichiariamo le nostre funzioni API:

```
1. Private Declare Sub ReleaseCapture Lib "USER32" ()
2. Private Declare Function SendMessage Lib "USER32" Alias "SendMessageA" (ByVal hwnd
   As Long, ByVal wMsg As Long, ByVal wParam As Long, lParam As Any) As Long
```

Dichiariamo anche due costanti utilizzate per definire il messaggio che permette lo spostamento di un controllo:

```
3. Private Const WM_NCLBUTTONDOWN = &HA1
```



```
4. Private Const SC_DLG_NO_UI = &H2
```

Per evitare di dover riscrivere ogni volta il codice di spostamento del controllo abbiamo scritto le istruzioni di spostamento all'interno di una sub. Per distinguere un controllo dall'altro utilizzeremo il suo handle:

```
6. Public Sub MuoviControllo(ByVal Handle As Long)
7.     ReleaseCapture
8.     SendMessage Handle, WM_NCLBUTTONDOWN, SC_DLG_NO_UI, 0&
9. End Sub
```

In questa funzione c'è poco da commentare: viene semplicemente rilasciato l'oggetto attivo tramite la funzione API *ReleaseCapture* e poi viene inviato tramite la funzione API *SendMessage* il messaggio SC_DLG_NO_UI associato a WM_NCLBUTTONDOWN alla finestra il cui handle è quello passato alla funzione.

Per spostare un controllo utilizzeremo l'evento *MouseDown* associato ad ogni controllo. Se il tasto del mouse premuto sarà quello sinistro, verrà chiamata la funzione **MuoviControllo** vista poco prima.
Scriveremo per cui:

```
11. Private Sub Command1_MouseDown(Button As Integer, Shift As Integer, X As Single, Y
    As Single)
12.     If Button = 1 Then MuoviControllo Command1.hwnd
13. End Sub
14.
15. Private Sub Dir1_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As
    Single)
16.     If Button = 1 Then MuoviControllo Dir1.hwnd
17. End Sub
18.
19. Private Sub Text1_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As
    Single)
20.     If Button = 1 Then MuoviControllo Text1.hwnd
21. End Sub
```

Come possiamo vedere, la chiamata dei diversi controlli, differisce soltanto per l'handle passato alla funzione.

A questo punto possiamo provare il nostro progetto: una semplice operazione di trascinamento provoca lo spostamento del controllo trascinato. Ovviamente ad ogni riavvio del form tutti i controlli torneranno alla loro posizione originale perché non abbiamo una funzione di salvataggio delle posizioni dei controlli.

L'utilizzo dei messaggi delle finestre semplifica enormemente la scrittura di codici per il calcolo delle coordinate di movimento del controllo.

Esiste una seconda soluzione funzionante però soltanto su Windows 9x. Consiste nell'utilizzare i messaggi WM_SYSCOMMAND e SC_MOVE.
È stata abbandonata in favore della nuova soluzione, provata e funzionante anche su Windows 2000.

Fibia FBI
Nuova soluzione di: Rapta e KyeMan
25 Aprile 2001


Torna all'indice degli HowTo

