



Inserire delle immagini nelle voci di un menu

http://www.vbsimple.net/howto/ht_047.htm

Difficoltà:  2 / 5

Quasi tutti i moderni programmi presentano all'interno delle voci dei menu delle icone che rendono il programma un po' più accattivante e semplice da capire.

Esistono due diverse soluzioni per inserire le icone all'interno delle voci di un menu: la prima soluzione è molto semplice ma altrettanto limitata ed insoddisfacente; la seconda soluzione utilizza il [subclassing](#) della finestra e si presenta pertanto davvero molto complessa ma i risultati che produce sono davvero soddisfacenti. In questo articolo sarà presentata la prima delle due soluzioni, ovvero quella più semplice e che non richiede il subclassing della finestra.

Il progetto sviluppato si compone di un solo form, senza alcun controllo, eccetto un menu scritto con l'[Editor di menu](#)  incluso nell'[IDE](#) di Visual Basic, contenente al suo interno soltanto quattro voci. Il nome ed i contenuti del menu non importa; utilizzeremo infatti le funzioni [API](#) per effettuare le modifiche che non richiedono i nomi dei menu ma si basano sugli [handle](#) dei menu. Unico requisito per il progetto: il primo menu deve contenere almeno 4 voci al suo interno. Nel nostro esempio il menu "**Menu con bitmap**" contiene al suo interno 4 voci: VB, ASP, HTM E TXT ed inseriremo un'icona per ognuna delle quattro voci.



Vediamo il codice del progetto:

```

1. Option Explicit
2.
3. Private Const MF_BYPOSITION = &H400&
4.
5. Private Declare Function GetMenu Lib "user32" (ByVal hwnd As Long) As Long
6. Private Declare Function GetSubMenu Lib "user32" (ByVal hMenu As Long, ByVal nPos
  As Long) As Long
7. Private Declare Function SetMenuItemBitmaps Lib "user32" (ByVal hMenu As Long,
  ByVal nPosition As Long, ByVal wFlags As Long, ByVal hBitmapUnchecked As Long,
  ByVal hBitmapChecked As Long) As Long
8.
9. Private imgBitmaps(3) As IPictureDisp
10.

```

Alla riga 3 è dichiarata una costante di nome **MF_BYPOSITION** che indica che le operazioni che eseguiremo sulle voci di menu si baseranno sulla loro posizione relativa rispetto al menu che le contiene.

Alla riga 5 è stata dichiarata la funzione **GetMenu** che recupera l'[handle](#) della barra dei menu della finestra. La funzione **GetSubMenu** restituisce invece l'handle dei singoli menu della finestra, che sono contenuti all'interno della barra dei menu. La terza funzione API è **SetMenuItemBitmaps** che consente di associare un'immagine ad una voce di menu.

Alla riga 9 è dichiarato un [array](#) di immagini (**IPictureDisp**) che verrà utilizzato per contenere le immagini da associare ai menu. Purtroppo è stato necessario [allocare](#) questa matrice per bloccare le immagini in memoria. Senza di essa le immagini sarebbero state caricate ed automaticamente [deallocare](#) dalla memoria, rendendole inutilizzabili.

Gli oggetti in Visual Basic infatti vengono scaricati dalla memoria soltanto quando tutti i [puntatori](#) all'oggetto rilasciano l'oggetto puntato, ovvero vengono eliminati tutti i riferimenti all'oggetto in memoria. L'array **imgBitmaps** terrà le immagini in memoria con la minima occupazione di memoria possibile. Avremmo potuto utilizzare anche un controllo *Image* o *Picture* sul form ma avrebbe richiesto una quantità maggiore di risorse.

```
11. Private Function AppPath() As String
12.     AppPath = App.Path
13.     If Right$(AppPath, 1) <> "\" Then AppPath = AppPath & "\"
14. End Function
15.
```

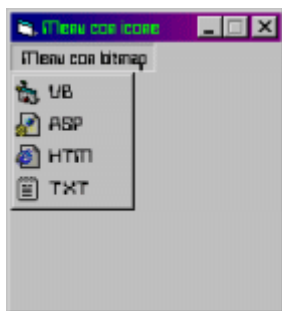
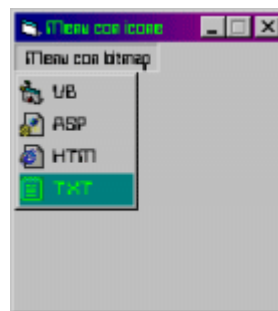
La funzione AppPath, peraltro già vista in un [altro articolo](#), restituisce il percorso dell'applicazione, assicurandosi che esso termini con una barra rovesciata.

```
16. Private Sub Form_Load()
17.     Dim hSubMenu As Long
18.     Set imgBitmaps(0) = LoadPicture(AppPath & "0.BMP")
19.     Set imgBitmaps(1) = LoadPicture(AppPath & "1.BMP")
20.     Set imgBitmaps(2) = LoadPicture(AppPath & "2.BMP")
21.     Set imgBitmaps(3) = LoadPicture(AppPath & "3.BMP")
22.
23.     hSubMenu = GetSubMenu(GetMenu(Me.hwnd), 0)
24.     Call SetMenuItemBitmaps(hSubMenu, 0, MF_BYPOSITION, imgBitmaps(0).Handle,
imgBitmaps(0).Handle)
25.     Call SetMenuItemBitmaps(hSubMenu, 1, MF_BYPOSITION, imgBitmaps(1).Handle,
imgBitmaps(1).Handle)
26.     Call SetMenuItemBitmaps(hSubMenu, 2, MF_BYPOSITION, imgBitmaps(2).Handle,
imgBitmaps(2).Handle)
27.     Call SetMenuItemBitmaps(hSubMenu, 3, MF_BYPOSITION, imgBitmaps(3).Handle,
imgBitmaps(3).Handle)
28. End Sub
29.
```

Alle righe 18-21 sono caricate quattro immagini presenti nella cartella del programma. La riga 23 recupera l'handle della prima (a base 0) voce del menu del form. Successivamente (righe 24-27) ad ogni voce del menu è assegnato l'handle di ognuna delle quattro immagini tramite la funzione **SetMenuItemBitmap**.

```
30. Private Sub Form_Unload(Cancel As Integer)
31.     Erase imgBitmaps
32. End Sub
```

L'applicazione che richiama la funzione **SetMenuItemBitmaps** è responsabile di deallocare le immagini al termine dell'utilizzo. Alla riga 31 la matrice di oggetti **imgBitmaps** è azzerata e tutti i suoi elementi deallocati utilizzando la funzione **Erase**.

**Figura 2****Figura 3**

Passando all'esecuzione del progetto possiamo vedere che all'interno del menu sono presenti le quattro icone. Purtroppo la loro qualità è davvero scarsa e nel momento in cui una voce è selezionata (Figura 3) la loro qualità è ulteriormente inferiore.

La soluzione proposta è davvero molto semplice ma altamente limitata. Le icone da inserire nel menu devono essere immagini bitmap di dimensioni 15x15 preferibilmente a 16 colori o in bianco e nero. In aggiunta le bitmap devono restare in memoria, deve ovvero essere presente un oggetto in memoria che tenga un riferimento all'immagine per prevenire lo scaricamento dell'oggetto.

In futuro potrebbe essere pubblicata anche l'altra soluzione, che utilizza subclassing e come tale si presenta molto più difficile della soluzione appena sviluppata.

[Fibia FBI](#)
25 Aprile 2002



[Torna all'indice degli HowTo](#)
