



Suonare con Visual Basic

http://www.vbsimple.net/activity/act_21.htm

Richiesta di: Anna - 10 Aprile 2002

Difficoltà: ▶▶ 2 / 5

È possibile fare programmi musicali, tipo la tastiera musicale?

Mi spiego meglio: quando studiavo Quick Basic, ho fatto un programmino molto simpatico, con l'immagine della tastiera di un pianoforte, e premendo i vari tasti della tastiera del PC, i corrispettivi "toni" venivano suonati. In VB non sono riuscita a trovare il modo di riprodurre gli stessi "toni", non c'e' modo di farlo?

In Quick Basic, così come in Turbo Pascal e tantissimi altri linguaggi, era presente la funzione *Play* che riproduceva dall'altoparlante interno al PC le note richieste. In Visual Basic tale istruzione non esiste più e l'[API](#) di sistema (utilizzando Windows 95/98/ME) provvede alcuna soluzione per simulare la generazione di toni dall'altoparlante.

Esistono alcuni componenti esterni ([OCX](#) e [DLL](#)) che consentono di integrare in Visual Basic questa mancanza. In questo articolo verrà invece utilizzata una tecnica alternativa per riprodurre le note musicali, molto semplice, ma che richiede la presenza di una scheda audio installata nel sistema. Questo articolo ha molti punti in comune con un'[altro articolo della sezione Richieste](#).

La soluzione consiste nel riprodurre le singole note musicali preregistrate utilizzando la funzione API *PlaySound* in grado di riprodurre files di tipo Wave, specificando il nome del file oppure passando un [buffer](#) contenente i dati del file da riprodurre.

In questo articolo includeremo due possibilità d'utilizzo: la prima consiste nello sfruttare i files esternamente, posti in una sottocartella del programma; la seconda utilizza un [file di risorse](#)  contenente i suddetti files e pertanto non richiede la loro presenza dopo la compilazione del progetto.

Il progetto si compone di un semplicissimo form contenente otto *CommandButton* posti in una [matrice](#) di controlli di nome **cmdSuonaNota** con proprietà *Index* da 0 a 7, uno per ogni nota musicale. A fianco di questi abbiamo un *CheckBox* di nome **chkRisorse** che consentirà di scegliere quale delle due modalità utilizzare (files esterni o file di risorse). Vediamo il codice relativo:



1. Option Explicit
- 2.
3. Private Const SND_ASYNC = &H1
4. Private Const SND_NODEFAULT = &H2

```

5. Private Const SND_MEMORY = &H4
6. Private Const lngRES_NOTE As Long = 1001
7.
8. Private Declare Function waveOutGetNumDevs Lib "winmm.dll" () As Long
9. Private Declare Function PlaySound Lib "winmm.dll" Alias "PlaySoundA" (ByVal
    lpszName As String, ByVal hModule As Long, ByVal dwFlags As Long) As Long
10.

```

Alle righe 3-5 sono dichiarate tre costanti  API utilizzate dalla funzione **PlaySound**. La prima di queste indica che la riproduzione del suono deve avvenire in maniera [asincrona](#) ovvero senza interrompere l'esecuzione del nostro programma fino al termine della riproduzione del suono. La seconda costante indica invece di non produrre il suono di default in caso che il file specificato non esista. L'ultima delle tre costanti API indica invece che il suono proverrà da un buffer in memoria e non leggendo un file WAVE. La terza costante sarà utilizzata soltanto nella riproduzione dei suoni dal file di risorse.

Alla riga 6 è definita una nuova costante di nome **lngRES_NOTE** ed indicherà il numero identificativo del primo suono nel file di risorse. Tutte le altre note avranno un identificativo successivo a quello indicato dalla costante **lngRES_NOTE**.

Vedremo il contenuto del file di risorse  soltanto in seguito. Intanto ci basti sapere che le singole note sono identificate dal tipo "**SOUND**", i loro numeri identificativi sono consecutivi ed il primo di essi è identificato dalla costante **lngRES_NOTE** (1001). Giusto per essere più chiari, la prima nota avrà numero 1001, la seconda 1002, etc...

Seguono le dichiarazioni di due funzioni API: la prima, **waveOutGetNumDevs**, restituisce il numero di dispositivi audio di [output](#) per la riproduzione dei suoni. Questa funzione verrà utilizzata per verificare la presenza di una scheda audio nel sistema. La seconda funzione **PlaySound** consente la riproduzione di un file audio WAVE, sia da file che da buffer in memoria.

```

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** protegge dal comune ed inatteso errore causato dall'utilizzo della proprietà *Path* dell'oggetto **App**. Nella maggior parte delle situazioni l'utilizzo di **App.Path** restituisce il percorso della cartella in cui si trova il programma, senza la barra separatrice alla fine. Se invece il programma è eseguito dalla radice del disco, la stessa **App.Path** restituirebbe la radice del disco, includendo quindi anche la barra separatrice alla fine della stringa. La funzione **AppPath** invece restituisce sempre la cartella in cui è eseguito il programma, assicurandosi che il percorso termini con il carattere "\ " alla fine della stringa.

```

16. Private Sub cmdSuonaNota_Click(Index As Integer)
17.     Dim strDati As String
18.     If waveOutGetNumDevs = 0 Then
19.         MsgBox "Nessuna scheda audio rilevata o accessibile!"
20.         Exit Sub
21.     End If

```

Nel momento in cui l'utente clicca sopra uno degli otto pulsanti verrà generato l'evento  **Click** dell'oggetto **cmdSuonaNota**, fornendo l'indice del pulsante premuto.

Alla riga 18 è verificata la presenza di una scheda audio e nel caso essa non fosse presente verrà mostrato un avviso e l'esecuzione della funzione verrà interrotta.

I files delle note da riprodurre sono scaricabili liberamente tramite [questo collegamento](#) e dovranno essere posti in una sottocartella di nome NOTE all'interno della cartella del nostro progetto.

```
22.   If chkRisorse.Value = vbChecked Then
23.       strDati = StrConv(LoadResData(lngRES_NOTE + Index, "SOUND"), vbUnicode)
24.       Call PlaySound(strDati, ByVal 0&, SND_MEMORY + SND_ASYNC Or SND_NODEFAULT)
```

In base allo stato della casella **chkRisorse** dovrà essere eseguita la riproduzione delle note dal file di risorse (righe 23 e 24) oppure dai files esterni (righe 26-28).

Alla riga 23 sono estratti i dati dal file di risorse tramite la funzione **LoadResData**, specificando il numero identificativo della risorsa ed suo tipo "**SOUND**". Poiché i numeri identificativi delle note sono successivi tra loro, per estrarre la nota richiesta richiederemo la risorsa con identificativo **lngRES_NOTE + Index**. Infatti alla pressione del primo pulsante Index avrà valore 0, per il secondo pulsante 1, etc.. che sommato al valore della costante **lngRES_NOTE** identificherà la risorsa richiesta.

I dati estratti dovranno però essere convertiti in formato **Unicode** per poter essere utilizzati dalla funzione *PlaySound*. La riproduzione dei dati estratti è effettuata alla riga 24, fornendo alla funzione il buffer contenente i dati ed il valori delle tre costanti API (suono da buffer, in maniera asincrona e senza suono di default).

```
25.   Else
26.       strDati = "" & Choose(Index + 1, "DO1", "RE", "MI", "FA", "SOL", "LA", "SI",
           "DO2")
27.       strDati = AppPath & "NOTE\" & strDati & ".WAV"
28.       Call PlaySound(strDati, ByVal 0&, SND_ASYNC Or SND_NODEFAULT)
29.   End If
30. End Sub
31.
```

Nel caso che **chkRisorse** non fosse selezionato, le note dovranno essere lette dalla sottocartella NOTE. Il nome del file da estrarre sarà recuperato mediante l'utilizzo della funzione **Choose**. Index conterrà l'indice del valore da estrarre, a base 0; ad esso sarà aggiunto il valore 1 poiché la funzione **Choose** indica il primo elemento con il valore 1.

La riga 26 da sola recupera il nome del file da estrarre, senza il percorso o la sua [estensione](#). Tali dati saranno quindi aggiunti alla riga 27.

Infine alla riga 28 è riprodotto il file indicato nella variabile **strDati**. Non dovrà essere inclusa la costante **SND_MEMORY** poiché il file non è contenuto in memoria.

```
32. Private Sub Form_KeyPress(KeyAscii As Integer)
33.   Dim Tasto As Integer
34.   Const OrdineTasti = "QWERTYUI"
35.   Tasto = InStr(1, OrdineTasti, Chr$(KeyAscii), vbTextCompare)
36.   If Tasto > 0 Then cmdSuonaNota_Click Tasto - 1
37. End Sub
```

Prima di passare al file di risorse abbiamo aggiunto una semplicissima routine che gestisce la pressione dei tasti della tastiera per simulare una tastiera musicale. Affinché questa

funzione operi nella maniera corretta è necessario che la proprietà *KeyPreview* sia impostata tu **True**. I tasti predisposti a suonare sono indicati dalla costante **OrdineTasti** ed iniziano con la lettera Q e terminano linearmente con la lettera I.

Se il tasto premuto fa parte dell'insieme di questi tasti sarà lanciato il metodo **cmdSuonaNota_Click** con l'indice della nota da suonare. Il succitato metodo non è altro che la routine legata all'evento *Click* dell'oggetto **cmdSuonaNota** vista in precedenza.

Per comprendere la creazione e la compilazione di un file di risorse si raccomanda la lettura dell'[HowTo dedicato](#) e della pagina delle [Informazioni aggiuntive](#).

Abbiamo già specificato che i files con le note si trovano nella sottocartella NOTE ed all'interno del file di risorse hanno numeri identificativi consecutivi, il cui primo è il valore 1001 e sono tutti identificati dal tipo SOUND.

Il nostro file di risorse si compone pertanto di:

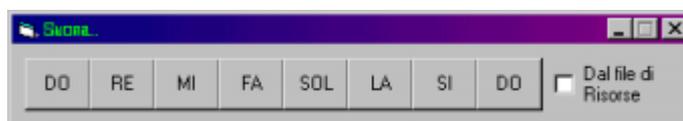
```

1. #define WAV_SOUND1 1001
2. #define WAV_SOUND2 1002
3. #define WAV_SOUND3 1003
4. #define WAV_SOUND4 1004
5. #define WAV_SOUND5 1005
6. #define WAV_SOUND6 1006
7. #define WAV_SOUND7 1007
8. #define WAV_SOUND8 1008
9.
10. WAV_SOUND1 SOUND PRELOAD MOVEABLE "NOTE\DO1.WAV"
11. WAV_SOUND2 SOUND PRELOAD MOVEABLE "NOTE\RE.WAV"
12. WAV_SOUND3 SOUND PRELOAD MOVEABLE "NOTE\MI.WAV"
13. WAV_SOUND4 SOUND PRELOAD MOVEABLE "NOTE\FA.WAV"
14. WAV_SOUND5 SOUND PRELOAD MOVEABLE "NOTE\SOL.WAV"
15. WAV_SOUND6 SOUND PRELOAD MOVEABLE "NOTE\LA.WAV"
16. WAV_SOUND7 SOUND PRELOAD MOVEABLE "NOTE\SI.WAV"
17. WAV_SOUND8 SOUND PRELOAD MOVEABLE "NOTE\DO2.WAV"

```

L'aver utilizzato **PRELOAD** invece di **LOADONCALL** determinerà il carimento della risorsa anticipatamente per velocizzare il suo utilizzo in seguito.

Possiamo quindi passare alla prova pratica di questo codice lanciando l'esecuzione del progetto e cliccando sui pulsanti con le note per sentire l'audio provenire dagli altoparlanti collegati.



Si potrà simulare una tastiera musicale premendo i tasti che vanno dalla lettera Q alla lettera I.

In questo articolo sono state presentate entrambe le soluzioni - files esterni o risorse - perché l'utilizzo di una invece di un'altra comporta leggere differenze. Naturalmente in un progetto reale **soltanto una delle due soluzioni dovrà essere applicata**, tenendo a mente che l'utilizzo delle risorse andrà ad incrementare la grandezza del file compilato perché

esse saranno incluse ([linked](#), collegate) nel file eseguibile compilato. L'utilizzo dei files esterni comporta invece la necessità di dover installare un numero di files maggiore.

[Fibia FBI](#)
25 Aprile 2002



[Torna all'introduzione delle Richieste dei lettori](#)
