



Gioco dell'impiccato

http://www.vbsimple.net/activity/act_10.htm

Richiesta di: [Rosario Manto](#) - 11 Gennaio 2001

Difficoltà: 3 / 5

Vorrei implementare in Visual Basic il gioco dell'impiccato.

Il problema in esame non è complesso, ma richiede un bel po' di codice.

Il gioco è il solito rompicapo dove il giocatore deve indovinare una determinata frase, della quale è nota soltanto la lunghezza e la presenza di eventuali spazi, lettere accentate e numeri. Il giocatore può scegliere una lettera dell'alfabeto a caso. Se tale lettera esiste all'interno della frase sarà mostrata, altrimenti, il numero di tentativi messi a disposizione diminuirà di 1.

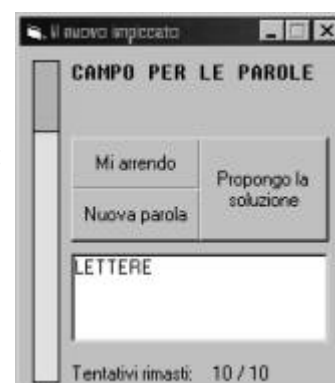
Nel nostro esempio daremo al giocatore fino a 10 possibilità e le parole saranno scelte in maniera casuale dal computer da un file di testo di nome **PAROLE.TXT** presente nella stessa cartella dove si trova il programma.






Prima di cominciare ci si assicuri di avere tale file o crearlo ed inserirlo nella cartella del programma.

Cominciamo disegnando l'interfaccia utente per il giocatore.

Essa si compone dei seguenti controlli:

- Un'area *Shape* di nome **AreaCompleta** con tali proprietà:
 - *BackColor*: Celeste (&H00FFFF00&)
 - *BackStyle*: 1 - Opaque
 - *Shape*: 0 - Rectangle
 - *Top*: 120
 - *Height*: 3015
- Un'area *Shape* di nome **AreaAttuale** con tali proprietà:
 - *BackColor*: Giallo (&H0000FFFF&)
 - *BackStyle*: 1 - Opaque
 - *Shape*: 0 - Rectangle
 - *Top*: 780
 - *Height*: 2355
- Una *Label* **A** di nome **Parola** con tali proprietà:
 - *Caption*: "CAMPO PER LE PAROLE"
 - *AutoSize*: True
 - *Font*: Fixedsys stile normale e grandezza 9
- Un *CommandButton* di nome **Arrendo** e *Caption* "Mi arrendo"



- Un *CommandButton*  di nome **NuovaParola** e *Caption* "Nuova parola"
- Un *CommandButton*  di nome **Soluzione** e *Caption* "Propongo la soluzione"
- Una *ListBox*  di nome **Lettere** con proprietà *Columns* a 7
- Una *Label*  di nome **TentativiLabel** e *Caption* "Tentativi rimasti"
- Una *Label*  di nome **TentativiRimasti** e *Caption* "10 / 10"

Definita l'interfaccia utente, spieghiamone rapidamente il funzionamento, per comprendere cosa dovrà fare il codice.

All'inizio della partita il giocatore premerà il pulsante "Nuova parola" e sarà mostrata la frase in maniera nascosta, ovvero ogni carattere della parola sarà sostituito con un trattino. Il giocatore di tale frase capirà soltanto la lunghezza ed il numero di parole di cui si compone; per il resto vedrà soltanto una serie di trattini.

Vediamo le singole routine di cui il codice si compone:

```
1. Option Explicit
2. Private Const MAXTENTATIVI = 10
3. Private TENTATIVI As Integer
4. Private LISTAPAROLE As Collection
5. Private PAROLACERCATA As String
6.
```

Innanzitutto definiamo alcuni dati globali all'interno del form.

Alla riga 2 abbiamo definito la [costante](#)  **MAXTENTATIVI** che indica il numero massimo di tentativi concessi per indovinare la frase.

Alla riga 3 abbiamo la variabile **TENTATIVI** che conterrà il numero di tentativi finora fatti per indovinare la frase. Il numero andrà da 0 a **MAXTENTATIVI**.

Abbiamo definito anche una [Collection](#) di nome **LISTAPAROLE** che conterrà tutte le frasi estratte dal file. Per semplificare il processo, inizialmente tutte le frasi del file saranno lette e memorizzate nella Collection. Man mano avanti con il gioco le frasi saranno estratte in maniera casuale dalla Collection.

Alla riga 5 abbiamo la variabile **PAROLACERCATA** che ospiterà la parola (frase) ricercata durante il gioco.

```
7. Private Sub Form_Load()
8.     Dim FILENR As Integer
9.     Dim BUFFER As String
10.    Set LISTAPAROLE = New Collection
11.    Lettere.Clear
12.    TentativiRimasti.Caption = "0 / " & MAXTENTATIVI
13.    Parola.Caption = ""
14.    AreaAttuale.Top = AreaCompleta.Top
15.    AreaAttuale.Height = AreaCompleta.Height
16.    FILENR = FreeFile
17.    Open App.Path & "\PAROLE.TXT" For Input As FILENR
18.    While Not EOF(FILENR)
19.        Line Input #FILENR, BUFFER
20.        LISTAPAROLE.Add UCase(BUFFER)
21.    Wend
22.    Close FILENR
```

```
23. End Sub
24.
```

All'esecuzione dell'[evento](#) ⚡ Load del form verranno eseguite alcune inizializzazioni dei controlli del form e delle variabili.

Alla riga 8 abbiamo dichiarato la variabile **FILENR** che conterrà il numero dell'[handle](#) del file da cui leggere i dati.

Alla riga 9 dichiariamo la variabile **BUFFER** che servirà come [buffer](#) di lettura dal file contenente le frasi.

Prima di poter utilizzare la Collection **LISTAPAROLE** è necessario [istanziare](#) la [classe](#) mediante l'utilizzo della parola chiave *New* (riga 10).

Nelle righe dal 10 al 15 vengono inizializzati i [controlli](#) sul form: viene azzerata la *ListBox* **Lettere**, viene preparata la *Label* **TentativiRimasti** scrivendo in essa il numero massimo di tentativi **MAXTENTATIVI**, viene azzerata la *Label* **PAROLA** che conterrà la frase da indovinare ed infine viene posta l'area *Shape* **AreaAttuale** alle stesse coordinate dell'altra area **AreaCompleta**.

Alle righe da 16 a 22 viene effettuata [l'apertura, la lettura e chiusura del file](#). Ogni riga del file viene scritta (in maiuscolo) all'interno della *Collection*.

```
25. Private Sub Form_Terminate()
26.     Set LISTAPAROLE = Nothing
27. End Sub
28.
```

Poiché abbiamo [istanziato](#) la *Collection* **LISTAPAROLE** è consigliabile effettuare la [deallocazione](#) della *Collection* alla chiusura del form. L'evento *Terminate* è l'ultima routine eseguita prima dell'effettivo scaricamento del form dalla memoria.

La riga 26 effettua lo scaricamento della *Collection* **LISTAPAROLE** dalla memoria.

```
29. Private Sub NuovaParola_Click()
30.     Dim PAROLACASUALE As Integer
31.     Dim CONTA As Integer
32.     Dim PAROLANASCOSTA As String
33.     Randomize
34.     PAROLACASUALE = Rnd * (LISTAPAROLE.Count - 1) + 1
35.     PAROLACERCATA = LISTAPAROLE.Item(PAROLACASUALE)
36.     PAROLANASCOSTA = ""
37.     For CONTA = 1 To Len(PAROLACERCATA)
38.         PAROLACASUALE = Asc(Mid(PAROLACERCATA, CONTA, 1))
39.         If (PAROLACASUALE >= Asc("A")) And (PAROLACASUALE <= Asc("Z")) Then
40.             PAROLANASCOSTA = PAROLANASCOSTA & "-"
41.         Else
42.             PAROLANASCOSTA = PAROLANASCOSTA & Chr(PAROLACASUALE)
43.         End If
44.     Next CONTA
45.     Parola.Caption = PAROLANASCOSTA
46.     Lettere.Clear
47.     For CONTA = Asc("A") To Asc("Z")
48.         Lettere.AddItem Chr(CONTA)
49.     Next CONTA
50.     Lettere.ListIndex = 0
51.     AreaAttuale.Top = AreaCompleta.Top
52.     AreaAttuale.Height = AreaCompleta.Height
```

```

53.     Lettere.Enabled = True
54.     Soluzione.Enabled = True
55.     Arrendo.Enabled = True
56.     TENTATIVI = 0
57.     TentativiRimasti.Caption = MAXTENTATIVI - TENTATIVI & " / " & MAXTENTATIVI
58. End Sub
59.

```

Il click sopra il pulsante **NuovaParola** effettuerà la scelta casuale di una frase dalla Collection, nasconderà la frase immettendo trattini al posto delle lettere alfabetiche e reinizializzerà i controlli.

All'interno di questa routine abbiamo dichiarato tre variabili: **PAROLACASUALE** indicherà il numero casuale della frase da estrarre dalla Collection, **CONTA** servirà per effettuare dei cicli ripetitivi e **PAROLANASCOSTA** sarà uno spazio temporaneo per preparare la frase da indovinare.

Alla riga 33 viene inizializzato il generatore di numeri casuali ed alla riga successiva viene scelta una frase casuale compresa tra 1 ed il numero complessivo di frasi contenute nella Collection. Il numero della frase viene memorizzato nella variabile **PAROLACASUALE**. Alla riga 35 viene memorizzata nella variabile globale **PAROLACERCATA** la frase da indovinare.

Nelle righe comprese tra 36 e 45 viene creata la frase nascosta. Prima viene azzerata la variabile **PAROLANASCOSTA**, e man mano viene estratto un carattere dalla frase **PAROLACERCATA** e memorizzato in **PAROLACASUALE**. Se il carattere estratto è compreso nell'insieme delle lettere alfabetiche esso viene sostituito con un trattino, altrimenti non viene toccato. Così, man mano viene generata la frase in maniera nascosta e memorizzata in **PAROLANASCOSTA**.

Al termine della generazione viene mostrata nella *Label* **Parola** la frase nascosta (riga 45).

Nelle righe tra 46 e 57 vengono reinizializzati i controlli.

Viene prima azzerata e poi riempita nuovamente la *ListBox* **Lettere** con le lettere alfabetiche; fatto questo viene selezionata la prima lettera, quella con indice 0.

In seguito viene ridisposta l'area *Shape* **AreaAttuale**, sono riattivati i controlli **Lettere**, **Soluzione** ed **Arrendo**.

Alla riga 56 viene azzerato il numero di tentativi effettuati e viene ridisegnata la *Label* indicante il numero di possibilità rimaste.

```

60. Public Function TrovaLettera(ByVal LETTERA As String) As Boolean
61.     Dim CONTA As Integer
62.     Dim PAROLANASCOSTA As String
63.     PAROLANASCOSTA = Parola.Caption
64.     TrovaLettera = False
65.     For CONTA = 1 To Len(PAROLACERCATA)
66.         If Mid(PAROLACERCATA, CONTA, 1) = LETTERA Then
67.             PAROLANASCOSTA = Left(PAROLANASCOSTA, CONTA - 1) & LETTERA & Mid
(PAROLANASCOSTA, CONTA + 1)
68.             TrovaLettera = True
69.         End If
70.     Next CONTA
71.     Parola.Caption = PAROLANASCOSTA
72. End Function
73.

```

Prepariamo una funzione booleana di nome **TrovaLettera**. Essa effettua la sostituzione

dei trattini della frase nascosta con la lettera scelta e passata come parametro. Il valore che la funzione restituisce indica se la frase da indovinare conteneva almeno un'occorrenza della lettera ricercata.

Alla riga 62 viene dichiarata la variabile **PAROLANASCOSTA** che conterrà la frase nascosta, contenuta nella *Label Parola*.

Alla riga 65 inizia un ciclo che controlla tutte le lettere della frase originale memorizzata nella variabile **PAROLACERCATA**. Se la lettera in analisi è uguale a quella ricercata (riga 66), viene sostituito il trattino della frase nascosta con la lettera ricercata (riga 67) e viene impostato a True il valore **TrovaLettera** che sarà restituito all'uscita della funzione (riga 68).

Alla riga 71, infine, viene aggiornata la *Label Parola* con la nuova frase.

```

74. Private Sub Lettere_DblClick()
75.     Dim LETTERAOK As Boolean
76.     Dim SPOSTAMENTO As Single
77.     LETTERAOK = TrovaLettera(Lettere.List(Lettere.ListIndex))
78.     Lettere.RemoveItem Lettere.ListIndex
79.     Lettere.ListIndex = 0
80.     If LETTERAOK = True Then
81.         If Parola.Caption = PAROLACERCATA Then
82.             MsgBox "Hai vinto!", vbInformation + vbOKOnly, "Il nuovo impiccato"
83.             Lettere.Enabled = False
84.             Soluzione.Enabled = False
85.             Arrendo.Enabled = False
86.         End If
87.     Else
88.         TENTATIVI = TENTATIVI + 1
89.         SPOSTAMENTO = (AreaCompleta.Height / MAXTENTATIVI)
90.         AreaAttuale.Top = AreaAttuale.Top + SPOSTAMENTO
91.         If AreaAttuale.Height < SPOSTAMENTO Then AreaAttuale.Height = SPOSTAMENTO
92.         AreaAttuale.Height = AreaAttuale.Height - SPOSTAMENTO
93.     End If
94.     TentativiRimasti.Caption = MAXTENTATIVI - TENTATIVI & " / " & MAXTENTATIVI
95.     If TENTATIVI = MAXTENTATIVI Then Arrendo_Click
96. End Sub
97.

```

Nel momento in cui l'utente clicca due volte rapidamente sopra una lettera della *ListBox* viene effettuato il controllo di tale lettera. Se essa esiste, saranno scoperte le occorrenze della lettera, altrimenti il numero di tentativi incrementerà di 1. La lettera sarà in ogni caso eliminata dalla *ListBox Lettere*.

Alla riga 77 viene chiamata la funzione TrovaLettera con la lettera al momento selezionata, indicata dalla proprietà **ListIndex** della *ListBox*. Il valore restituito dalla funzione TrovaLettera viene memorizzato nella variabile **LETTERAOK**.

Se è stata trovata almeno un'occorrenza della lettera cercata viene verificato se la nuova frase (riga 81), con le lettere scoperte, è uguale alla frase originale PAROLACERCATA. In tal caso il giocatore ha completato il gioco. Sarà mostrato un messaggio e verranno bloccati i controlli **Lettere**, **Soluzione** e **Arrendo**.

Se, invece, la ricerca della lettera specificata dà esito negativo (riga 87), il giocatore ha commesso un errore; viene aumentato il valore di TENTATIVI (riga 88) e viene aggiornata la posizione dell'area **AreaAttuale**, spostandola di una certa misura, data dalla

divisione dell'area complessiva per il numero massimo di tentativi.

Alla riga 94 viene aggiornata la *Label* che indica il numero di tentativi rimasti.

Se il numero di tentativi effettuati è uguale al numero massimo di tentativi concessi, la partita termina, come se il giocatore si arrendesse (riga 95).

```

98. Private Sub Arrendo_Click()
99.     TENTATIVI = MAXTENTATIVI
100.     TentativiRimasti.Caption = MAXTENTATIVI - TENTATIVI & " / " & MAXTENTATIVI
101.     MsgBox "Hai perso!", vbCritical + vbOKOnly, "Il nuovo impiccato"
102.     Lettere.Enabled = False
103.     Soluzione.Enabled = False
104.     Arrendo.Enabled = False
105.     Parola.Caption = PAROLACERCATA
106. End Sub
107.

```

Il click sopra il pulsante **Arrendo** provoca la perdita della partita, la visualizzazione di un messaggio (riga 101) ed il blocco dei soliti controlli **Lettere**, **Soluzione** e **Arrendo**.

Alla riga 105 viene mostrata la frase da indovinare (**PAROLACERCATA**).

```

108. Private Sub Soluzione_Click()
109.     If (UCase(InputBox("La soluzione secondo me è:", "Prova soluzione")) =
PAROLACERCATA) Then
110.         Parola.Caption = PAROLACERCATA
111.         MsgBox "Hai vinto!", vbInformation + vbOKOnly, "Il nuovo impiccato"
112.         Lettere.Enabled = False
113.         Soluzione.Enabled = False
114.         Arrendo.Enabled = False
115.     Else
116.         Arrendo_Click
117.     End If
118. End Sub

```

L'ultima routine da vedere è il click sopra il pulsante **Soluzione**. Esso provoca la visualizzazione di una finestra che chiede la soluzione secondo il giocatore. Se il messaggio immesso è uguale alla frase ricercata (riga 109), viene mostrata la frase, appare un messaggio che indica la vittoria e sono bloccati i controlli **Lettere**, **Soluzione** ed **Arrendo**.

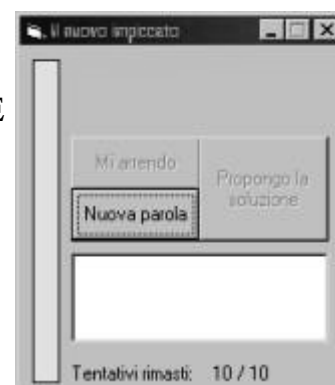
Se la soluzione proposta dal giocatore è sbagliata, la partita termina immediatamente, come se il giocatore si arrendesse (riga 116).

Il progetto termina qui. Rimane solo di provarlo.

All'avvio del programma appare soltanto un form tutto bloccato. È attivo soltanto il pulsante la cui Caption indica "**Nuova parola**".

Provando a cliccarlo inizia una nuova partita, con una frase scelta a caso dal file PAROLE.TXT.

Si attiva la *ListBox* **Lettere** ed i pulsanti "**Mi arrendo**" e "**Propongo la soluzione**".



Man mano che il gioco procede, le lettere dalla *ListBox* spariscono.

Se la lettera scelta compare nella frase da indovinare essa sarà mostrata. In caso contrario il numero di tentativi rimasti diminuisce di 1 e la barra gialla alla sinistra scende.

Se, entro il numero massimo di tentativi concessi, il giocatore indovina la frase apparirà il messaggio mostrato nella figura 4.



Se il giocatore si arrende, non riesce ad indovinare la frase nascosta entro il numero di tentativi messi a disposizione, o sbaglia la soluzione, apparirà il messaggio mostrato nella figura 5.

Nel momento in cui l'utente clicca sul pulsante **"Propongo la soluzione"** gli viene presentata la finestra mostrata nella figura 6. Viene richiesta la soluzione.

Se la frase inserita è quella corretta apparirà un messaggio che indica la vittoria (figura 4), altrimenti la partita si chiuderà con una sconfitta (figura 5).



Figura 4



Figura 5



Figura 6

Il gioco non offre alcuna tecnica particolare. È dedicata soltanto una particolare cura sugli elementi che compongono l'interfaccia utente.

Non è un progetto ambizioso, ma può essere uno spunto per qualche altro progetto. Male che vada potrete godervi qualche piacevole ora con questo gioco che prende sempre molto piede tra i giovanissimi tra i banchi di scuola.

[Fibia FBI](#)
15 Gennaio 2001



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