

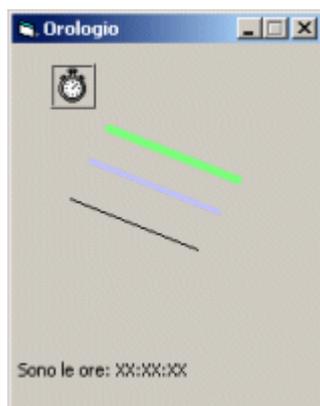

[Home Page](#)
[Novità](#)
[Aiuto](#)

Realizzare un orologio

http://www.vbsimple.net/howto/ht_063.htm

Difficoltà: 3 / 5

La domanda mi è stata posta tantissime volte che date le numerose richieste ho deciso di realizzarne un articolo accessibile a tutti; per una corretta comprensione si raccomanda una conoscenza sommaria della geometria del cerchio.



Verrà realizzato un orologio analogico, quello con le lancette, per capirci, il tutto con pochissime righe di codice, sfruttando gli oggetti *Line* ed assegnando loro le coordinate dovute. Il form si comporrà di un controllo **Timer** di nome **tmrTimer** che scandirà i secondi, quindi con la sua proprietà *Interval* impostata su 1000 millisecondi (1 secondo); ad esso aggiungeremo 3 controlli *Line* dai nomi **linSeconds**, **linMinutes** e **linHours**, senza preoccuparci delle loro coordinate. Per distinguere le 3 lancette abbiamo preferito assegnar loro 3 differenti colori (*BorderColor*) e spessori (*BorderWidth*). In fondo al form abbiamo anche aggiunto una **Label** di nome **lblOre** che riproduca l'orario in maniera digitale affiancato dalla

data corrente.

Per rendere l'orologio più realistico abbiamo fatto in modo che lo spostamento della lancetta delle ore avvenisse gradualmente, in maniera legata ai minuti, anziché scattare da un'ora all'altra con uno spostamento di 30° (360/12) per volta. Lo stesso procedimento è stato applicato alla lancetta dei minuti, sincronizzata con l'avanzare dei secondi.

Quasi tutto il codice si trova all'interno di una routine richiamata dall'evento *Timer* del controllo **tmrTimer** ma prima di vederla in dettaglio affrontiamo quei piccoli accorgimenti necessari alla preparazione della superficie su cui si muoveranno le lancette:

```

1. Option Explicit
2.
3. Private Const PI As Single = 3.1415
4. Private Const intCenterX As Integer = 1400
5. Private Const intCenterY As Integer = 1400
6.
7. Private Sub Form_Load()
8.     linHours.ZOrder vbSendToBack
9.     linMinutes.ZOrder vbBringToFront
10.    linSeconds.ZOrder vbBringToFront
11.    Me.DrawStyle = vbDot
12.    Me.Circle (intCenterX, intCenterY), 1300, vbRed
13.    tmrTimer_Timer
14. End Sub
15.

```

La costante **PI** rappresenta semplicemente il PI greco, fondamentale per qualsiasi operazione basata sulle ellissi. Incredibile a dirsi in Visual Basic questa costante non esiste da nessuna parte. Le altre due costanti **intCenterX** e **intCenterY** indicano la posizione del centro dell'orologio. Saranno utilizzate per il posizionamento delle tre lancette.

All'avvio del form sarà opportuno assicurare la posizione delle lancette sull'asse Z, ponendo quella delle ore sotto tutte e quella dei secondi sopra tutte le altre, per evitar in tal modo che la lancetta delle ore copra parte di quella dei minuti o dei secondi.

Alle righe 11 e 12 sarà disegnato un semplice cerchio all'interno del quale si muovono le nostre lancette; il cerchio ha un raggio poco superiore al raggio della lancetta dei secondi, la più lunga delle tre. Sarà immediatamente richiamato l'evento Timer per posizionare subito le nostre lancette e non dover attendere il passaggio del primo secondo per aggiornare lo schermo.

```

16. Private Sub tmrTimer_Timer()
17.     Dim Orario As Date
18.     Orario = Now
19.     RefreshLine linHours, (Hour(Orario) Mod 12) + Minute(Orario) / 60, 12, 800
20.     RefreshLine linMinutes, (Minute(Orario) + Second(Orario) / 60), 60, 1000
21.     RefreshLine linSeconds, Second(Orario), 60, 1200
22.     lblOre.Caption = "Sono le ore: " & Format$(Orario, "hh:nn:ss") & vbNewLine & _
23.         "di " & Format$(Orario, "dddd dd mmmm yyyy")
24. End Sub
25.

```

Ad ogni secondo scatterà questa routine che si occuperà di richiamare la funzione **RefreshLine** che ridisegnerà la lancetta specificata nel primo argomento; ogni lancetta sarà disegnata autonomamente e successivamente saranno indicati nella Label **lblOre** data e ora correnti.

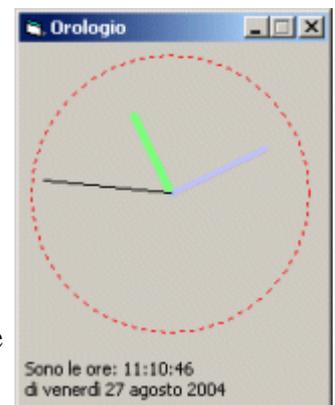
```

26. Private Sub RefreshLine(ByRef Line As Line, ByVal Value As Single, ByVal Divisor As
    Integer, ByVal Radius As Integer)
27.     Dim SngAngle As Single
28.     SngAngle = Value * (2 * PI / Divisor) - (PI / 2)
29.     With Line
30.         .X1 = intCenterX
31.         .Y1 = intCenterY
32.         If (SngAngle < PI) And (SngAngle >= 0) Then
33.             .Y2 = intCenterY + Abs(Radius * Sin(SngAngle))
34.         Else
35.             .Y2 = intCenterY - Abs(Radius * Sin(SngAngle))
36.         End If
37.         .X2 = intCenterX + Radius * Cos(SngAngle)
38.     End With
39. End Sub

```

La routine **RefreshLine** è il centro di questo problema: essa riceve 4 argomenti di cui il primo indica l'oggetto *Line* da muovere, il secondo indica il valore del divisore (terzo parametro) e l'ultimo parametro indica il raggio della linea da tracciare. Per chiarire il problema il cerchio di 360° viene diviso in tante parti quante indicate in **Divisor** e la lancetta verrà posizionata in funzione dell'argomento **Value**. La variabile **sngAngle** verrà utilizzata per determinare l'angolo in radianti alla cui fine sarà posizionata la lancetta.

Il calcolo da effettuare è molto semplice: determinato l'angolo semplicemente dividendo l'angolo giro (doppio PI greco) in tante parti quante indicate in Divisor e moltiplicando per il valore, sottraiamo 90° (PI / 2) per spostare l'origine alle ore 12, anziché alle ore 3, origine del cerchio.



Ottenuto **sngAngle**, con due semplici operazioni saranno determinate le coordinate X e Y finali della lancetta; alle righe 30 e 31 sono posizionate le origini, quindi, in funzione che l'angolo sia positivo o negativo, sarà calcolata la coordinata finale, aggiungendo, in caso di segno positivo, al centro la distanza data da raggio per il seno dell'angolo (coordinata Y) o sottraendola in caso di angolo

negativo; per quanto riguarda la coordinata X basterà moltiplicare il raggio per il coseno dello stesso angolo.

Questa procedura verrà ripetuta per tutte e tre le lancette, ed ognuna delle quali verrà posizionata al suo corrispondente punto. Il risultato è quello mostrato nella figura a fianco.

Si tratta di un problema a prima vista difficile ma una semplice conoscenza della geometria di base delle ellissi è in grado di risolvere il problema con estrema semplicità.

[Fibia FBI](#)

27 Agosto 2004



[Torna all'indice degli HowTo](#)
