Lavorare con le Date in VBA

Microsoft Excel tramite Il linguaggio Visual Basic, offre una vasta gamma di funzioni per poter creare e manipolare le date ed è possibile inserire una data in qualsiasi formato leggibile, ad esempio usando lo stile americano (mese-giorno-anno) o il formato europeo (giorno-mese-anno), dipende dalle impostazioni internazionali del vostro sistema

Per memorizzare una data o un orario, e necessario definire una variabile di tipo **Date** e assegnare un valore alla variabile, con una sintassi particolare che prevede di inserire la data tra due caratteri "cancelletto" (**#**): per esempio, il 10 agosto 2015, può essere scritto:

```
Sub Prova()
    Dim Mydata As Date
   Mydata = #10/8/2015#
   MsgBox ("La data corrente è : " & Mydata)
End Sub
```

Ciò produrrebbe:

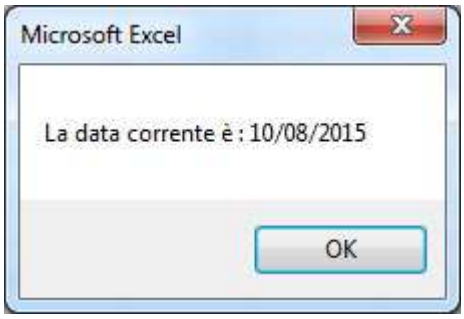

Fig. 1

Come si può notare nel codice la data è inserita nel formato americano, cioè viene espressa nella forma Mese/Giorno/Anno, perchè, le istruzioni in VBA sono in inglese, ma verrà convertita nel formato inserito nel sistema per cui nel box appare la data scritta all'italiana :10/08/2015.

La visualizzazione del formato della data dipende dalle impostazioni di settaggio contenute nel Pannello di Controllo seguendo il percorso: Pannello di controllo/Impostazioni Internazionali/Data oppure per Windows 7 o superiori dal percorso: Pannello di controllo/Paese e lingua

Per ottenere la data corrente del computer, è possibile richiamare la funzione **Date** in questo modo:

```
Sub Prova()
   MsgBox Date
End Sub
```

Quando si compone un valore data, è necessario seguire alcune regole che dipendono dalla lingua che si sta utilizzando. Vediamo quelle Italiane, dove un mese è riconosciuto da un indice in un intervallo da 1 a 12 e ha anche un nome che può essere rappresentato in due formati: *completo* o *breve*.

Indice Mese	Nome Completo	Nome Breve
1	Gennaio	Gen
2	Febbraio	Feb
3	Marzo	Mar
4	Aprile	Apr
5	Maggio	Mag
6	Giugno	Giu
7	Luglio	Lug
8	Agosto	Ago
9	Settembre	Set
10	Ottobre	Ott
11	Novembre	Nov
12	Dicembre	Dic

Fig. 2

Inoltre una settimana è una combinazione di 7 giorni consecutivi di un mese, per cui ogni giorno può essere riconosciuto da un indice da 1 a 7 e il giorno di ciascun indice è riconosciuto da un nome, dove il primo giorno ha un indice di 1 e si chiama Domenica, mentre l'ultimo giorno ha un indice di 7 si chiama Sabato. Come i mesi di un anno, i giorni di una settimana hanno nomi completi e brevi.

Indice Giorni	Nome Completo	Nome Breve
1	Domenica	Dom
2	Lunedì	Lun
3	Martedì	Mar
4	Mercoledì	Mer
5	Giovedì	Gio
6	Venerdì	Ven
7	Sabato	Sab

Fig. 3

Abbiamo visto che, durante la creazione di una data, è possibile includere il suo valore tra i segni **#**. Un'alternativa è quella di fornire una data come una stringa. A sostegno di questa, il linguaggio Visual Basic fornisce una funzione chiamata **DateValue** la cui sintassi è:

Function DateValue(ByVal StringDate As String) As Variant

Quando si chiama questa funzione, si deve fornire una data valida come argomento. La validità dipende dalla lingua del sistema operativo. Ecco un esempio:

```
Sub Prova()
   Dim data1 As Date
   data1 = DateValue("22-Ago-2010")
   MsgBox ("La Data impostata è : " & data1)
End Sub
```

Ciò produrrebbe:

Fig. 4

Un'alternativa a inizializzare una variabile data è quella di utilizzare una funzione denominata **DateSerial**, la sua sintassi è:

Function DateSerial(ByVal [Year] As Integer, ByVal [Month] As Integer, ByVal [Day] As Integer) As Variant

Come potete vedere, questa funzione consente di specificare l'anno, il mese, il giorno di un valore di data, ovviamente senza i segni **#**. Quando è stata chiamata, questa funzione restituisce un valore di tipo Variant, che può essere convertita in una data . Ecco un esempio:

```
Sub Prova()
   Dim data1 As Date
   data1 = DateSerial(2010, 2, 8)
   MsgBox ("La Data impostata è : " & data1)
End Sub
```

Ciò produrrebbe:

Fig. 5

Quando si passa il valore di questa funzione, è necessario limitare ogni componente per l'intervallo consentito di valori. È possibile passare l'anno con due cifre da 0 a 99. Ecco un esempio:

```
Sub Prova()
    Dim data1 As Date
    data1 = DateSerial(3, 2, 8)
    MsgBox ("La Data impostata è : " & data1)
End Sub
```

Se si passa l'anno come un valore compreso tra 0 e 99, l'interprete rimanda all'orologio del computer per ottenere il secolo. Al momento in cui scriviamo, il secolo sarebbe il 20 e l'anno specificato sarebbe aggiunto, che producono 2003. Per essere più precisi e ridurre la confusione, si dovrebbe sempre passare l'anno con 4 cifre.

Il mese deve essere un valore compreso tra 1 e 12, e se si passa un valore superiore a 12, l'interprete dovrebbe calcolare il resto di quel numero da 12 (il numero MOD 12 =?). Il risultato della divisione intera sarebbe stato usato come il numero di anni e aggiunto al primo argomento. Il resto sarebbe stato usato come il mese del valore data. Ad esempio, se si passa il mese come 18, la divisione intera produrrebbe 1, quindi 1 anno sarebbe stato aggiunto al primo argomento. Il resto è 6 (18 MOD 12 = 6); così il mese sarebbe stato usato come 6 (Giugno). Ecco un esempio:

```
Sub Prova()
    Dim data1 As Date
    data1 = DateSerial(2003, 18, 8)
    MsgBox ("La Data impostata è : " & data1)
End Sub
```

Ciò produrrebbe:

Fig. 6

Per fare un altro esempio, se si passa il mese come 226, la divisione intera (226 / 12) produce 18 e il numero sarebbe stato aggiunto al primo argomento (2003 + 18 = 2021). Il resto di 226 a 12 è 10 (226 MOD 12 = 10) e sarebbe stato utilizzato come il mese. Ecco un esempio:

```
Sub Prova()
    Dim data1 As Date
    data1 = DateSerial(2003, 226, 8)
    MsgBox ("La Data impostata è : " & data1)
End Sub
```

Ciò produrrebbe:

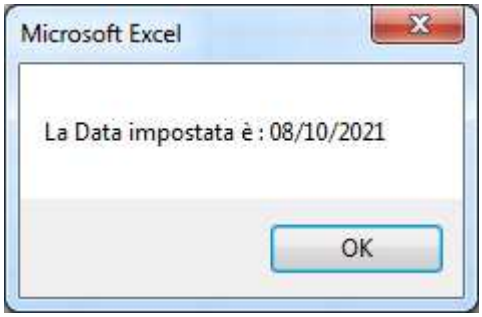

Fig. 7

Se il mese è passato come 0, si considera 12 (dicembre), mentre se è passato come -1, è considerato 11 (novembre) dell'anno precedente e così via. Se il mese è passato come un numero inferiore a -11, l'interprete avrebbe calcolare la divisione intera a 12, aggiungendo 1 a questo risultato e utilizzando tale numero come l'anno, calcolando il resto a 12, e usandolo come mese.

A seconda del mese, il valore dell'argomento giorno può essere passato come un numero compreso tra 1 e 28, tra 1 e 29, tra 1 e 30, o tra 1 e 31. Se l'argomento giorno viene passato come un numero inferiore a 1 o superiore a 31, l'interprete utilizza il primo giorno del mese passato come secondo argomento. Se il giorno è passato come -1, il giorno è considerato l'ultimo giorno del mese precedente del mese in discussione. Ad esempio, se l'argomento mese è passato come 4 (aprile) e l'argomento giorno è passato come -1, l'interprete avrebbe usato 31 come giorno perché l'ultimo giorno del mese di marzo è 31.

Se l'argomento mese è passato come 3 (marzo) e l'argomento giorno è passato come -1, l'interprete si riferirebbe all'argomento Anno per determinare se l'anno è bisestile o meno. Ciò consentirebbe all'interprete di utilizzare sia 28 o 29 per il valore del giorno. L'interprete utilizza questo algoritmo per qualsiasi valore giorno passato come terzo parametro quando il numero è inferiore a 1.

Se il Giorno argomento è passato con un valore superiore a 28, 29, 30, o 31, l'interprete usa questo stesso algoritmo in senso inverso per determinare il mese e il giorno.

Conversione di un valore in Data

Se si dispone di un valore come quello fornito come una stringa e si desidera convertirlo in una data, è possibile richiamare la funzione **CDate** la cui sintassi è:

Function CDate(Value As Object) As Date

Questa funzione accetta qualsiasi tipo di valore, ma il valore deve essere convertibile in una data valida. Se la funzione riesce nella conversione, produce un valore Date. Se la conversione non riesce, si verifica un errore.
Come abbiamo visto finora, una data è un valore composto da tre parti: l'anno, il mese e il giorno. L'ordine di questi componenti e il modo in cui sono messi insieme per costituire una data riconoscibile dipendono dalla lingua e vengono definiti nel linguaggio e nelle Impostazioni internazionali nel Pannello di controllo.

L'Anno di una data

Il linguaggio Visual Basic supporta l'anno di una data che va da 1 a 9999. Ciò significa che questa è il Range che si può considerare quando si tratta di date nei fogli di lavoro. Nella maggior parte delle operazioni, durante la creazione di una data, se si specifica un valore compreso tra 1 e 99, l'interprete avrebbe usato il secolo in corso per la sinistra due cifre, ciò significa che, un anno come 4 o 04 indicherebbe l'anno 2004.

Nella maggior parte dei casi, per essere più precisi, di solito dovrete specificare sempre l'anno con 4 cifre. Se si dispone di un valore di data e si vuole scoprire l'anno, è possibile richiamare la funzione **Year**, la cui sintassi è:

Public Function Year(ByVal DateValue As Variant) As Integer

Come potete vedere, questa funzione assume un valore di data come argomento e dovrebbe contenere una data valida. In caso affermativo, la funzione restituisce l'anno in forma numerica di una data. Ecco un esempio:

```
Sub Prova()
  Dim data1 As Date
  data1 = #2/8/2004#
  MsgBox ("Sono nata nell'anno " & Year(data1))
End Sub
```

Ciò produrrebbe:

Fig. 8

Il mese di un anno

La parte del mese di una data è un valore numerico che va da 1 a 12 e quando si crea una data, è possibile specificarlo con 1 o 2 cifre. Se il mese è compreso tra 1 e 9, è possibile precederlo con uno 0. Se si dispone di un valore data e vuole ottenere il suo mese, è possibile chiamare la funzione **Month**, la cui sintassi è:

Function Month(ByVal DateValue As Variant) As Integer

Questa funzione richiede un oggetto Data come argomento e se la data è valida, la funzione restituisce un numero compreso tra 1 e 12 per il mese. Ecco un esempio:

```
Sub Prova()
   Dim data1 As Date
   data1 = #2/8/2004#
   MsgBox ("Il mese scelto è  " & Month(data1))
End Sub
```

Ciò produrrebbe:

Fig. 9

Come già menzionato, la funzione **Month** produce un valore numerico che rappresenta il mese di una data, se invece si vuole ottenere il nome del mese, è possibile chiamare la funzione **MonthName**, la cui sintassi è:

Function MonthName(ByVal Month As Integer, Optional ByVal Abbreviate As Boolean = False) As String

Questa funzione richiede due argomenti di cui uno opzionale. L'argomento richiesto deve rappresentare il valore di un mese e se è un valore valido, questa funzione restituisce il nome corrispondente. Ecco un esempio:

```
Sub Prova()
   Dim data1 As Date
   data1 = #2/8/2004#
   MsgBox ("Il mese scelto è  " & MonthName(Month(data1)))
End Sub
```

Ciò produrrebbe:

Fig. 10

Il secondo argomento consente di specificare se si desidera ottenere il nome breve o completo del mese. Il valore predefinito è il nome completo e in questo caso il valore predefinito dell'argomento è False, mentre se si desidera ottenere il nome breve, si deve passare il secondo argomento come True . Ecco un esempio:

```
Sub Prova()
Dim data1 As Date
   data1 = #2/8/2004#
   MsgBox ("Il mese scelto è  " & MonthName(Month(data1), True))
End Sub
```

Ciò produrrebbe:

Fig. 11

Il giorno di un mese

Il giorno è un valore numerico in un mese e a seconda del mese (e dell'anno), il suo valore può variare da 1 a 29 (febbraio bisestile), da 1 a 28 (Febbraio in un anno non bisestile), da 1 a 31 (gennaio, marzo, maggio, luglio, agosto, ottobre e dicembre), o da 1 a 30 (aprile, giugno, settembre e novembre).
Se si dispone di un valore di data e volete sapere il giorno in un anno, è possibile richiamare la funzione **Day**, la cui sintassi è:

Function Day(ByVal DateValue As Variant) As Integer

Questa funzione richiede una data come argomento e se la data è valida, la funzione restituisce il giorno numerico del mese dell'argomento data. Ecco un esempio:

```
Sub Prova()
   Dim data1 As Date
   data1 = #2/8/2004#
   MsgBox ("Il mese scelto è  " & Day(data1))
End Sub
```

Ciò produrrebbe:

Fig. 12

Il giorno della settimana

Per ottenere il nome del giorno della settimana, è possibile usare una funzione denominata **WeekdayName**, la cui sintassi è:

Function WeekdayName(ByVal Weekday As Integer, Optional ByVal Abbreviate As Boolean = False, Optional ByVal FirstDayOfWeekValue As Integer = 0) As String

Questa funzione richiede un argomento obbligatorio e due opzionali. L'argomento obbligatorio deve essere, o rappresentare un valore compreso tra 0 e 7. Se si passa come 0, l'interprete si riferisce alla lingua del sistema operativo per determinare il primo giorno della settimana, che in inglese americano è la Domenica. In caso contrario, la funzione riporterebbe il nome corrispondente del giorno. Ecco un esempio:

```
Sub Prova()
    MsgBox ("Il Giorno scelto è : " & WeekdayName(4))
End Sub
```

Ciò produrrebbe:

Fig. 13

Se si passa un valore negativo o un valore superiore a 7, si riceverà un errore. Il secondo argomento consente di specificare se si desidera ottenere il nome breve o completo. Il valore di default di questo argomento è **False**, che produce un nome completo, se si desidera il nome breve, si deve passare il secondo argomento come **True**. Ecco un esempio:

```
Sub Prova()
    MsgBox ("Il Giorno scelto è : " & WeekdayName(4, True))
End Sub
```

Per supportare più opzioni, il linguaggio Visual Basic fornisce la funzione **Format ()** la cui sintassi è:

Function Format(ByVal Expression As Object, Optional ByVal Style As String = "") As String

Il primo argomento è la data in cui deve essere formattata e il secondo argomento è una stringa che contiene la formattazione da applicare. Per crearla, si utilizza una combinazione del mese, giorno e anno. Ecco un esempio:

```
Sub Prova1()
   Dim data1 As Date
   data1 = #12/28/2014#
   MsgBox ("La data è  " & Format(data1, "dd MMMM, yyyy"))
End Sub
```

Ciò produrrebbe:

Fig. 14