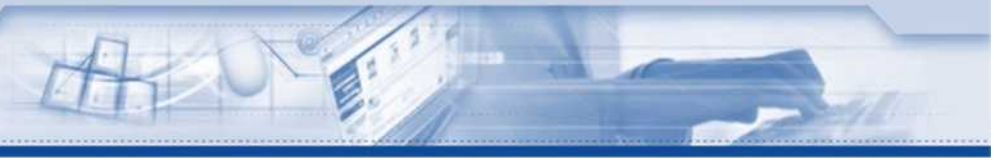
db2000 functions library

© 2008 Massimo Mascalchi
massimo.mascalchi@db2000web.net

INDICE



DATE FUNCTIONS

Age

restituisce gli anni trascorsi (età) tra le due date indicate

```
sintassi:  r = fDATE.Age([data 1], [data 2])
esempio:   r = fDATE.Age("12/09/1956", "12/09/1976") ' 20
```

BaseStartWeekday

legge/imposta il primo giorno di inizio della settimana

```
sintassi:  r = fDATE.BaseStartWeekday   ' ritorna il primo giorno di
                                        ' inizio della settimana....

           fDATE.BaseStartWeekday = [ID giorno settimana]   ' imposta il primo giorno di
                                                            ' inizio della settimana....
```

P.S. si consideri la tabella qui di lato anche per tutte quelle funzioni dove uno o più parametri fanno riferimento all'identificativo del giorno della settimana

```
[ID giorno settimana] =
1 = lunedì
2 = martedì
3 = mercoledì
4 = giovedì
5 = venerdì
6 = sabato
7 = domenica
0 = primo giorno della settimana come specificato nelle impostazioni di sistema
```

```
esempi:    r = fDATE.BaseStartWeekDay   ' ritorna il primo giorno di inizio della settimana....

           fDATE.BaseStartWeekDay = 1   ' imposta il lunedì come primo giorno
                                        ' di inizio della settimana..........
```

CountHolidays

restituisce quanti giorni festivi sono presenti tra le due date indicate

```
sintassi:  r = fDATE.CountHolidays([data 1], [data 2])
esempio:   r = fDATE.CountHolidays("01/01/2008", "31/12/2008") ' 113
```

CountWeekdayInMonth

restituisce quante volte il giorno di una settimana (domenica, lunedì, martedì, ecc.) è presente nel mese della data indicata

```
sintassi:  r = fDATE.CountWeekdayInMonth([ID giorno settimana], [data])
esempio:   r = fDATE.CountWeekdayInMonth(1, "12/09/2008") ' 4
```

CountWorkdays

restituisce quanti giorni lavorativi sono presenti tra le due date indicate

```
sintassi:  r = fDATE.CountWorkdays([data 1], [data 2])
esempio:   r = fDATE.CountWorkdays("01/01/2008", "31/12/2008") ' 253
```

DateTest

effettua il test sulla data indicata restituendola formattata correttamente con gli attributi di sistema

```
sintassi:  r = fDATE.DateTest([data])
esempio:   r = fDATE.DateTest("12/09/2008") ' "12/09/2008"
```

DayFromDate
restituisce il giorno relativo alla data indicata

```
sintassi:  r = fDATE.DayFromDate([data])
esempio:   r = fDATE.DayFromDate("12/09/2008")  ' 12
```

EasterDate
restituisce la data della pasqua relativa all'anno indicato

```
sintassi:  r = fDATE.EasterDate([anno])
esempio:   r = fDATE.EasterDate(2008)  ' "23/03/2008"
```

FindDateNextWeekday
restituisce la data corrispondente al prossimo identificativo del giorno della settimana indicato trovato dopo la data

```
sintassi:  r = fDATE.FindDateNextWeekday([ID giorno settimana], [data])
esempio:   r = fDATE.FindDateNextWeekday(1, "12/09/2008")  ' "15/09/2008"
```

FindDatePreviousWeekday
restituisce la data corrispondente al precedente identificativo del giorno della settimana indicato trovato prima della data

```
sintassi:  r = fDATE.FindDatePreviousWeekday([ID giorno settimana], [data])
esempio:   r = fDATE.FindDatePreviousWeekday(1, "12/09/2008")  ' "08/09/2008"
```

FirstDayInMonth
restituisce la data corrispondente al primo giorno del mese riferito alla data indicata

```
sintassi:  r = fDATE.FirstDayInMonth([data])
esempio:   r = fDATE.FirstDayInMonth("12/09/2008")  ' "01/09/2008"
```

FirstDayInQuarter
restituisce la data corrispondente al primo giorno di un trimestre riferito alla data indicata

```
sintassi:  r = fDATE.FirstDayInQuarter([data])
esempio:   r = fDATE.FirstDayInQuarter("12/09/2008")  ' "01/07/2008"
```

FirstDayInWeekFromDate
restituisce la data corrispondente al primo giorno della settimana riferita alla data indicata

```
sintassi:  r = fDATE.FirstDayInWeekFromDate([data])
esempio:   r = fDATE.FirstDayInWeekFromDate("12/09/2008")  ' "08/09/2008"
```

FirstDayInWeekFromYear
restituisce la data corrispondente al primo giorno della settimana riferita all'identificativo della settimana e all'anno indicati

```
sintassi:  r = fDATE.FirstDayInWeekFromYear([ID settimana], [anno])
                                          ' [ID settimana] = 0...51
esempio:   r = fDATE.FirstDayInWeekFromYear(3, 2008)  ' "28/01/2008"
```

FirstDayInWeekNow

restituisce la data corrispondente al primo giorno della settimana attuale

```
sintassi:  r = fDATE.FirstDayInWeekNow
esempio:   r = fDATE.FirstDayInWeekNow ' "18/08/2008" (la data attuale era il 22/08/2008)
```

FirstWorkdayInMonth

restituisce la data corrispondente al primo giorno lavorativo del mese relativo alla data indicata

```
sintassi:  r = fDATE.FirstWorkdayInMonth([data])
esempio:   r = fDATE.FirstWorkdayInMonth("12/09/2008")  ' "01/09/2008"
```

IsItalianHoliday

restituisce ***True*** se la data indicata corrisponde ad una festività italiana altrimenti ***False***

```
sintassi:  r = fDATE.IsItalianHoliday([data])
esempio:   r = fDATE.IsItalianHoliday("25/04/2008")  ' True
```

IsWeekend

restituisce ***True*** se la data indicata corrisponde ad un fine settimana altrimenti ***False***

```
sintassi:  r = fDATE.IsWeekend([data])
esempio:   r = fDATE.IsWeekend("10/08/2008")  ' True
```

LastDayInMonth

restituisce la data corrispondente all'ultimo giorno del mese riferito alla data indicata

```
sintassi:  r = fDATE.LastDayInMonth([data])
esempio:   r = fDATE.LastDayInMonth("12/09/2008")  ' "30/09/2008"
```

LastDayInQuarter

restituisce la data corrispondente all'ultimo giorno di un trimestre riferito alla data indicata

```
sintassi:  r = fDATE.LastDayInQuarter([data])
esempio:   r = fDATE.LastDayInQuarter("12/09/2008")  ' "30/09/2008"
```

LastDayInWeek

restituisce la data corrispondente al primo giorno della settimana riferita alla data indicata

```
sintassi:  r = fDATE.LastDayInWeek([data])
esempio:   r = fDATE.LastDayInWeek("12/09/2008")  ' "14/09/2008"
```

LastWorkdayInMonth

restituisce la data corrispondente all'ultimo giorno lavorativo del mese relativo alla data indicata

```
sintassi:  r = fDATE.LastWorkdayInMonth([data])
esempio:   r = fDATE.LastWorkdayInMonth("12/09/2008")  ' "30/09/2008"
```

MonthFromDate
restituisce il mese relativo alla data indicata

```
sintassi:   r = fDATE.MonthFromDate([data])
esempio:    r = fDATE.MonthFromDate("12/09/2008") ' 9
```

MonthNameFromDate
restituisce il nome corrispondente al mese relativo alla data indicata

```
sintassi:   r = fDATE.MonthNameFromDate([data], [abbreviazione])
```

```
[abbreviazione] =  False  =  non effettua alcuna abbreviazione
                   True   =  effettua l'abbreviazione sul testo restituito
```

P.S. si consideri la tabella precedente anche per tutte quelle funzioni dove uno o più parametri fanno riferimento al flag di abbreviazione

```
esempio:    r = fDATE.MonthNameFromDate("12/09/2008", False) ' "settembre"
```

New1
inizializza gli oggetti della libreria

```
sintassi:   fDATE.New1
esempio:    fDATE.New1
```

NextAnniversary
restituisce la data del prossimo anniversario relativo alla data indicata purché antecedente all'attuale

```
sintassi:   r = fDATE.NextAnniversary([data])
esempio:    r = fDATE.NextAnniversary("12/09/2007") ' "12/09/2008"
```

NextDate
restituisce la data successiva alla data indicata

```
sintassi:   r = fDATE.NextDate([data])
esempio:    r = fDATE.NextDate("12/09/2008") ' "13/09/2008"
```

NextDay
restituisce il giorno successivo alla data indicata

```
sintassi:   r = fDATE.NextDay([data])
esempio:    r = fDATE.NextDay("12/09/2008") ' 13
```

NextMonth
restituisce il mese successivo alla data indicata

```
sintassi:   r = fDATE.NextMonth([data])
esempio:    r = fDATE.NextMonth("12/09/2008") ' 10
```

NextMonthName
restituisce il nome del mese successivo alla data indicata

```
sintassi:  r = fDATE.NextMonthName([data], [abbreviazione])
esempio:   r = fDATE.NextMonthName("12/09/2008", False) ' "ottobre"
```

NextWeekday
restituisce l'identificativo del giorno della settimana successivo alla data indicata

```
sintassi:  r = fDATE.NextWeekday([data])
esempio:   r = fDATE.NextWeekday("12/09/2008") ' 7
```

NextWeekdayName
restituisce il nome del giorno della settimana successivo alla data indicata

```
sintassi:  r = fDATE.NextWeekdayName([data], [abbreviazione])
esempio:   r = fDATE.NextWeekdayName("12/09/2008", False) ' "sabato"
```

NextWorkday
restituisce la data del prossimo giorno lavorativo successivo alla data indicata

```
sintassi:  r = fDATE.NextWorkday([data])
esempio:   r = fDATE.NextWorkday("12/09/2008") ' "15/09/2008"
```

NextYear
restituisce l'anno successivo alla data indicata

```
sintassi:  r = fDATE.NextYear([data])
esempio:   r = fDATE.NextYear("12/09/2008") ' 2009
```

NthWeekday
restituisce la data successiva all'identificativo del giorno della settimana indicato per il numero delle ricorrenze dello stesso giorno partendo dalla indicata

```
sintassi:  r = fDATE.NthWeekday([data], [nr. ricorrenze], [ID giorno settimana], [modo])

                    [modo] =  False  =  inizia dal giorno indicato nella data
                              True   =  inizia dal primo giorno del mese indicato nella data

esempio:   r = fDATE.NthWeekday("12/09/2008", 3, 1, True) ' "22/09/2008"
```

PreviousDate
restituisce la data precedente alla data indicata

```
sintassi:  r = fDATE.PreviousDate([data])
esempio:   r = fDATE.PreviousDate("12/09/2008") ' "11/09/2008"
```

PreviousDay
restituisce il giorno precedente alla data indicata

```
sintassi:  r = fDATE.PreviousDay([data])
esempio:   r = fDATE.PreviousDay("12/09/2008") ' 11
```

PreviousMonth
restituisce il mese precedente alla data indicata

```
sintassi:  r = fDATE.PreviousMonth([data])
esempio:   r = fDATE.PreviousMonth("12/09/2008")  ' 8
```

PreviousMonthName
restituisce il nome del mese precedente alla data indicata

```
sintassi:  r = fDATE.PreviousMonthName([data], [abbreviazione])
esempio:   r = fDATE.PreviousMonthName("12/09/2008", False)  ' "agosto"
```

PreviousWeekday
restituisce l'identificativo del giorno della settimana precedente alla data indicata

```
sintassi:  r = fDATE.PreviousWeekday([data])
esempio:   r = fDATE.PreviousWeekday("12/09/2008")  ' 5
```

PreviousWeekdayName
restituisce il nome del giorno della settimana precedente alla data indicata

```
sintassi:  r = fDATE.PreviousWeekdayName([data])
esempio:   r = fDATE.PreviousWeekdayName("12/09/2008")  ' "giovedì"
```

PreviousWorkday
restituisce la data del primo giorno lavorativo precedente alla data indicata

```
sintassi:  r = fDATE.PreviousWorkday([data])
esempio:   r = fDATE.PreviousWorkday("12/09/2008")  ' "11/09/2008"
```

PreviousYear
restituisce l'anno precedente alla data indicata

```
sintassi:  r = fDATE.PreviousYear([data])
esempio:   r = fDATE.PreviousYear("12/09/2008")  ' 2007
```

SkipHolidays
restituisce la prima data valida saltando le eventuali festività

```
sintassi:  r = fDATE.SkipHolidays([data], [direzione])

                [direzione] =  False  =  percorre i giorni all'indietro
                               True   =  percorre i giorni in avanti

esempio:   r = fDATE.SkipHolidays("25/04/2008", True)  ' "28/09/2008"
```

WeekdayFromDate
restituisce l'identificativo del giorno della settimana relativo alla data indicata

```
sintassi:  r = fDATE.WeekdayFromDate([data])
esempio:   r = fDATE.WeekdayFromDate("12/09/2008")  ' 6
```

WeekdayNameFromDate

restituisce il nome del giorno della settimana relativo alla data indicata

```
sintassi:   r = fDATE.WeekdayNameFromDate([data], [abbreviazione])
esempio:    r = fDATE.WeekdayNameFromDate("12/09/2008", False) ' "venerdì"
```

YearFromDate

restituisce l'anno relativo alla data indicata

```
sintassi:   r = fDATE.YearFromDate([data])
esempio:    r = fDATE.YearFromDate("12/09/2008") ' 2008
```

INITIALIZATION FILES MANAGEMENT

(.INI FILES – MODE 1)

DeleteSection1

elimina un'intera sezione dal file .INI

```
sintassi:  fINI.DeleteSection1([nome sezione])
esempio:   fINI.DeleteSection1("TEST")
```

---

read1

restituisce il valore di una chiave letto in una sezione del file .INI

```
sintassi:  r = fINI.read1([nome sezione], [nome chiave], [valore di default])
esempio:   r = fINI.read1("TEST", "Autore", "")
```

New1

inizializza gli oggetti della libreria

```
sintassi:  fINI.New1([nome del file .INI])
esempio:   fINI.New1(AppPath & "test.ini")
```

---

write1

scrive il valore di una chiave in una sezione del file .INI

```
sintassi:  fINI.write1([nome sezione], [nome chiave], [valore da assegnare])
esempio:   fINI.write1("TEST", "Autore", "Massimo Mascalchi")
```

INITIALIZATION FILES MANAGEMENT

(.INI FILES – MODE 2)

DeleteSection2

elimina un'intera sezione dal file .INI indicato

```
sintassi:  fINI.DeleteSection1([nome file .INI], [nome sezione])
esempio:   fINI.DeleteSection1(AppPath & "test.ini", "TEST")
```

---

read2

legge il valore di una chiave di una sezione del file .INI indicato

```
sintassi:  r = fINI.read1([nome file .INI], [nome sezione], [nome chiave], [valore di default])
esempio:   r = fINI.read1(AppPath & "test.ini", "TEST", "Autore", "")
```

New2

inizializza gli oggetti della libreria

```
sintassi:  fINI.New2
esempio:   fINI.New2
```

---

write2

scrive il valore di una chiave in una sezione del file .INI indicato

```
sintassi:  fINI.write2([nome file .INI], [nome sezione], [nome chiave], [valore da assegnare])
esempio:   fINI.write2(AppPath & "test.ini", "TEST", "Autore", "Massimo Mascalchi")
```

ITALIAN FISCAL TOOLS

CheckFiscalCode

restituisce ***True*** se il codice fiscale indicato è corretto altrimenti ***False***

```
sintassi:  r = fIFT.CheckFiscalCode([codice fiscale])
esempio:   r = fIFT.CheckFiscalCode("RSSPLA56P12H791O") ' True
```

CheckIVA1

restituisce ***True*** se la partita IVA indicata è corretta altrimenti ***False*** (modo **1**)

```
sintassi:  r = fIFT.CheckIVA1([partita IVA])
esempio:   r = fIFT.CheckIVA1("05158550482") ' True
```

CheckIVA2

verifica la partita IVA indicata (modo **2**), se corretta restituisce il nome dell'ufficio di emissione altrimenti una stringa vuota

```
sintassi:  r = fIFT.CheckIVA2([partita IVA])
esempio:   r = fIFT.CheckIVA2("05158550482") ' "Firenze"
```

GenerateFiscalCode

restituisce il codice fiscale elaborato con i parametri indicati

```
sintassi:  r = fIFT.GenerateFiscalCode([cognome], _
                                       [nome], _
                                       [sesso], _
                                       [data di nascita], _
                                       [cod. località])
esempio:   r = fIFT.GenerateFiscalCode("ROSSI", "PAOLO", "M", "12/09/1956", "H791")
                                                                 ' "RSSPLA56P12H791"
```

New1

inizializza gli oggetti della libreria

```
sintassi:  fIFT.New1
esempio:   fIFT.New1
```

MATH FUNCTIONS

AreaCircle

restituisce l'area del cerchio calcolata in base al valore della misura raggio indicato

```
sintassi:  r = fMATH.AreaCircle([raggio])
esempio:   r = fMATH.AreaCircle(15.8) ' 784.267190042156
```

AreaCone

restituisce l'area del cono calcolata in base ai valori delle misure del raggio e dell'altezza indicati

```
sintassi:  r = fMATH.AreaCone([raggio], [altezza])
esempio:   r = fMATH.AreaCone(15.8, 27.3) ' 2349.94797906383
```

AreaCube

restituisce l'area del cubo calcolata in base al valore della misura del lato indicato

```
sintassi:  r = fMATH.AreaCube([misura del lato])
esempio:   r = fMATH.AreaCube(15.8) ' 1497.84
```

AreaCylinder

restituisce l'area del cilindro calcolata in base ai valori delle misure del raggio e dell'altezza indicate

```
sintassi:  r = fMATH.AreaCylinder([raggio], [altezza])
esempio:   r = fMATH.AreaCylinder(15.8, 27.3) ' 4278.72353048316
```

AreaRectangle

restituisce l'area del rettangolo calcolata in base ai valori delle misure della base e dell'altezza indicate

```
sintassi:  r = fMATH.AreaRectangle([base], [altezza])
esempio:   r = fMATH.AreaRectangle(27.3, 15.8) ' 431.34
```

AreaSphere

restituisce l'area della sfera calcolata in base al valore della misura raggio indicato

```
sintassi:  r = fMATH.AreaSphere([raggio])
esempio:   r = fMATH.AreaSphere(15.8) ' 3137.06876016862
```

AreaTrapezoid

restituisce l'area del trapezio calcolata in base ai valori delle misure delle basi e dell'altezza indicate

```
sintassi:  r = fMATH.AreaTrapezoid([base 1], [base 2], [altezza])
esempio:   r = fMATH.AreaTrapezoid(27.3, 11.5, 15.8) ' 306.52
```

BinToDec

restituisce il valore decimale ottenuto dalla conversione della stringa binaria indicata

```
sintassi:  r = fMATH.BinToDec([stringa binaria])
esempio:   r = fMATH. BinToDec("01111111") ' 127
```

BinToHex
restituisce il valore esadecimale ottenuto dalla conversione della stringa binaria indicata

```
sintassi:  r = fMATH.BinToHex([stringa binaria])
esempio:   r = fMATH.BinToHex("01111111") ' "7F"
```

ConvertDegreesToRadians
restituisce il valore in radianti ottenuto dalla conversione dei gradi indicati

```
sintassi:  r = fMATH.ConvertDegreesToRadians([gradi])
esempio:   r = fMATH.ConvertDegreesToRadians(180) ' 3.14159265358979
```

ConvertRadiansToDegrees
restituisce il valore in gradi ottenuto dalla conversione dei radianti indicati

```
sintassi:  r = fMATH.ConvertRadiansToDegrees([radianti])
esempio:   r = fMATH.ConvertRadiansToDegrees(3.14159265358979) ' 180
```

Decrement
restituisce il valore indicato decrementato per quanto è stato ulteriormente indicato nella quantità di decremento

```
sintassi:  r = fMATH.Decrement([valore], [quantità di decremento])
esempio:   r = fMATH.Decrement(25, 1) ' 24
```

DecToBin
restituisce la stringa binaria ottenuta dalla conversione del numero decimale indicato

```
sintassi:  r = fMATH.DecToBin([numero decimale])
esempio:   r = fMATH.DecToBin(127) ' "01111111"
```

DecToHex
restituisce il valore esadecimale ottenuto dalla conversione del numero decimale indicato

```
sintassi:  r = fMATH.DecToHex([numero decimale])
esempio:   r = fMATH.DecToHex(127) ' "7F"
```

Factorial
restituisce il fattoriale relativo al valore numerico e al modo indicati

```
sintassi:  r = fMATH.Factorial([valore numerico], [modo])
esempi:    r = fMATH.Factorial(7, 0) ' 5040
           r = fMATH.Factorial(7, 1) ' 48
```

HexToBin
restituisce la stringa binaria ottenuta dalla conversione del valore esadecimale indicato

```
sintassi:  r = fMATH.HexToBin([valore esadecimale])
esempio:   r = fMATH.HexToBin("7F") ' "01111111"
```

HexToDec
restituisce il valore decimale ottenuto dalla conversione del valore esadecimale indicato

```
sintassi:  r = fMATH.HexToDec([valore esadecimale])
esempio:   r = fMATH.HexToDec("7F")  ' 127
```

Increment
restituisce il valore indicato incrementato per quanto è stato ulteriormente indicato nella quantità di incremento

```
sintassi:  r = fMATH.Increment([valore], [quantità di incremento])
esempio:   r = fMATH.Increment(24, 1)  ' 25
```

IsPrime
restituisce ***True*** se se il valore indicato rappresenta un numero primo altrimenti ***False***

```
sintassi:  r = fMATH.IsPrime([valore])
esempio:   r = fMATH.IsPrime(3) ' True
```

MCD
restituisce il massimo comune divisore relativo ai due valori numerici indicati

```
sintassi:  r = fMATH.MCD([numero 1], [numero 2])
esempio:   r = fMATH.MCD(300, 5)  ' 5
```

MCM
restituisce il minimo comune multiplo relativo ai due valori numerici indicati

```
sintassi:  r = fMATH.MCM([numero 1], [numero 2])
esempio:   r = fMATH.MCM(300, 5)  ' 300
```

New1
inizializza gli oggetti della libreria

```
sintassi:  fMATH.New1
esempio:   fMATH.New1
```

VolumeCone
restituisce il volume del cono calcolato in base ai valori delle misure del raggio e dell'altezza indicate

```
sintassi:  r = fMATH.VolumeCone([raggio], [altezza])
esempio:   r = fMATH.VolumeCone(15.8, 27.3)  ' 7136.83142938362
```

VolumeCube
restituisce il volume del cubo calcolato in base al valore della misura del lato indicato

```
sintassi:  r = fMATH.VolumeCube([misura del lato])
esempio:   r = fMATH.VolumeCube(15.8)  ' 3944.312
```

VolumeCylinder

restituisce il volume del cilindro calcolato in base ai valori delle misure del raggio e dell'altezza indicate

```
sintassi:  r = fMATH.VolumeCylinder([raggio], [altezza])
esempio:   r = fMATH.VolumeCylinder(15.8, 27.3) ' 21410.4942881509
```

VolumePrismRect

restituisce il volume del prisma rettangolare calcolato in base ai valori delle misure della lunghezza, della larghezza e dell'altezza indicate

```
sintassi:  r = fMATH.VolumePrismRect([lunghezza], [larghezza], [altezza])
esempio:   r = fMATH.VolumePrismRect(27.3, 11.5, 15.8) ' 4960.41
```

VolumePyramid

restituisce il volume della piramide calcolato in base ai valori delle misure della lunghezza, della larghezza e dell'altezza indicate

```
sintassi:  r = fMATH.VolumePrismRect([lunghezza], [larghezza], [altezza])
esempio:   r = fMATH.VolumePrismRect(27.3, 11.5, 15.8) ' 1653.47
```

VolumeSphere

restituisce il volume della sfera calcolato in base al valore della misura raggio indicato

```
sintassi:  r = fMATH.VolumeSphere([raggio])
esempio:   r = fMATH.VolumeSphere(15.8) ' 16521.8954702214
```

MEASURE UNITS CONVERSION & TOOLS

ATTENZIONE: per l'utilizzo di queste funzioni è necessario disporre del file "*db2000 - measure units.dat*" (archivio delle unità di misura) oppure di un'altro ma avente la stessa struttura dei dati.

ConvertUnit

restituisce la conversione del valore/quantità indicato dall'unità di misura origine nell'unità di misura di conversione per il tipo di misura anch'esse indicate purché le due unità siano presenti nell'archivio delle unità di misura

sintassi:
```
r = fMC.ConvertUnit([tipo misura], _
                    [unità di misura origine], _
                    [valore/quantità unità di misura origine], _
                    [unità di misura di conversione])
```

esempi:
```
r = fMC.ConvertUnit("5", "8", 10, "23") ' 0.1
r = fMC.ConvertUnit("length", "centimeter", 10, "meter") ' 0.1
```

CountConversionUnits

restituisce, per il tipo di misura indicato, il numero delle unità di conversione attualmente presenti nell'archivio

sintassi:
```
r = fMC.CountConversionUnits([tipo misura])
```

esempi:
```
r = fMC.CountConversionUnits("5")
r = fMC.CountConversionUnits("length")
```

CountMeasureCategories

restituisce il numero dei tipi di misura (categorie) attualmente presenti nell'archivio

sintassi:
```
r = fMC.CountMeasureCategories
```

esempio:
```
r = fMC.CountMeasureCategories
```

DataMeasureUnitsFileName

restituisce/imposta il nome del file contenente i dati necessari per effettuare le conversioni tra le vaire unità di misura (archivio delle unità di misura)

sintassi:
```
r = fMC.DataMeasureUnitsFileName ' restituisce il nome del file
fMC.DataMeasureUnitsFileName = [nome del file]  ' imposta il nome del file
```

esempi:
```
r = fMC.DataMeasureUnitsFileName  ' restituisce il nome del file
fMC.DataMeasureUnitsFileName = AppPath & "\db2000 - measure units.dat"
                               ' imposta il nome del file
```

ExistConversionUnit

verifica nell'archivio l'esistenza dell'unità di misura indicata, restituisce ***True*** se trovata altrimenti ***False***

sintassi:
```
r = fMC.ExistConversionUnit([tipo misura], [unità di misura])
```

esempi:
```
r = fMC.ExistConversionUnit("5", "8")
r = fMC.ExistConversionUnit("length", "centimeter")
```

ExistMeasureCategory

verifica nell'archivio l'esistenza del tipo di misura indicata, restituisce ***True*** se trovata altrimenti ***False***

sintassi:
```
r = fMC.ExistMeasureCategory([tipo misura])
```

esempi:
```
r = fMC.ExistMeasureCategory("5")
r = fMC.ExistMeasureCategory("length")
```

New1

inizializza gli oggetti della libreria

sintassi:
```
fMC.New1
```

esempio:
```
fMC.New1
```

ReadConversionUnitParameters

restituisce in un array di stringhe i parametri di conversione dell'unità di misura indicata purché presente nell'archivio

sintassi:
```
ra() = fMC.ReadConversionUnitParameters([tipo misura], [unità di misura])
```

```
l'array restituito è una matrice di 5 elementi:   (0)  =  ID dell'unità di misura
                                                  (1)  =  nome dell'unità di misura
                                                  (2)  =  simbolo o abbreviazione
                                                  (3)  =  rapporto del valore di conversione
                                                  (4)  =  rettifica del valore di conversione
```

esempi:
```
ra() = fMC.ReadConversionUnitParameters("5", "8")
       ' se è stato dichiarato di usare come archivo dei dati necessari alle conversioni
       ' il file "db2000 - measure units.dat" verrano restituiti i seguenti parametri:
       '                                                    ra(0) = "8"
       '                                                    ra(1) = "centimeter"
       '                                                    ra(2) = "cm"
       '                                                    ra(3) = "0.01"
       '                                                    ra(4) = "0"

ra() = fMC.ReadConversionUnitParameters("length", "centimeter")
       ' se è stato dichiarato di usare come archivo dei dati necessari alle conversioni
       ' il file  "db2000 - measure units.dat" si otterrà lo stesso risultato dell'esem-
       ' pio precedente...
```

ReadConversionUnitsIdentifiers

restituisce in un'array di stringhe le identificazioni di tutte le unità di conversione presenti nell'archivio per il tipo di misura e i dati richiesti indicati

```
sintassi:  ra() = fMC.ReadConversionUnitsIdentifiers([tipo di misura], [dati richiesti])

                                [dati richiesti] =  (0)  =  ID delle unità di misura
                                                    (1)  =  nomi dell'unità di misura

esempi:    ra() = fMC.ReadConversionUnitsIdentifiers("5", 1)
           ra() = fMC.ReadConversionUnitsIdentifiers("length", 1)
```

ReadMeasureCategories

restituisce in un array di stringhe tutti gli identificativi del tipo di misura presenti nell'archivio per il tipo di campo richiesto

```
sintassi:  ra() = fMC.ReadMeasureCategories([dati richiesti])

                                [dati richiesti] =  (0)  =  ID delle unità di misura
                                                    (1)  =  nomi dell'unità di misura

esempi:    ra() = fMC.ReadMeasureCategories(0)
           ra() = fMC.ReadMeasureCategories(1)
```

ReadMeasureCategory

se presente nell'archivio restituisce una stringa contenente l'identificativo della misura (categoria), se il tipo della misura indicato è l'identificativo numerico ritorna il nome della misura stessa; invece, se il tipo della misura indicato è il nome, ritorna l'identificativo numerico

```
sintassi:  r = fMC.ReadMeasureCategory([tipo di misura])

esempi:    r = fMC.ReadMeasureCategory("5")       ' "length"
           r = fMC.ReadMeasureCategory("length")  ' 5
```

RemoveConversionUnit

rimuove dall'archivio l'unità di misura indicata associata al tipo di misura indicato (categoria), ritorna ***True*** se l'operazione termina correttamente altrimenti ***False***

```
sintassi:  r = fMC.RemoveConversionUnit([tipo di misura], [unità di misura])

esempi:    r = fMC.RemoveConversionUnit("5", "8")
           r = fMC.RemoveConversionUnit("length", "centimeter")
```

RemoveMeasureCategory

rimuove dall'archivio il tipo di misura indicato (categoria) e tutte le unità di misura associate, ritorna ***True*** se l'operazione termina correttamente altrimenti ***False***

```
sintassi:  r = fMC.RemoveMeasureCategory([tipo di misura])

esempi:    r = fMC.RemoveMeasureCategory("5")
           r = fMC.RemoveMeasureCategory("length")
```

WriteConversionUnitParameters

aggiorna i parametri contenuti in un array dell'unità di misura indicata o ne inserisce una nuova, ritorna ***True*** se l'operazione termina correttamente altrimenti ***False***

sintassi:

```
r = fMC.WriteConversionUnitParameters([tipo di misura], _
                                      [unità di misura], _
                                      [array parametri unità di misura], _
                                      [nuovo inserimento])
```

l'array dei parametri è una matrice di 5 elementi:

```
(0)  =  ID dell'unità di misura
(1)  =  nome dell'unità di misura
(2)  =  simbolo o abbreviazione
(3)  =  rapporto del valore di conversione
(4)  =  rettifica del valore di conversione
```

```
[nuovo inserimento] =  False  =  aggiorna i dati dell'unità di misura
                       True   =  inserisce una nuova unità di misura
```

ATTENZIONE: quando si tratta di un nuovo inserimento è necessario indicare in ***[unità di misura]*** il valore corrispondente al primo elemento dell'array dei parametri (**0**) o al contenuto del secondo elemento (**1**) indifferentemente

esempi:

```
r = fMC.WriteConversionUnitParameters("5", "23", ra(), False)
r = fMC.WriteConversionUnitParameters("length", "meter", ra(), False)
r = fMC.WriteConversionUnitParameters("5", "37", ra(), True)
```

WriteMeasureCategory

aggiorna i parametri del tipo di misura indicata (categoria) o ne inserisce una nuova, ritorna ***True*** se l'operazione termina correttamente altrimenti ***False***

sintassi:

```
r = fMC.WriteMeasureCategory([tipo di misura], _
                             [ID misura], _
                             [nome misura], _
                             [nuovo inserimento])
```

```
[nuovo inserimento] =  False  =  aggiorna i dati dell'unità di misura
                       True   =  inserisce una nuova unità di misura
```

ATTENZIONE:

1. quando si tratta di un nuovo inserimento è necessario indicare in ***[tipo di misura]*** il valore corrispondente a ***[ID misura]*** o a ***[nome misura]*** indifferentemente

2. si consiglia di evitare di cambiare il nome ad un tipo di misura (categoria) alla quale sono state già assegnate delle unità di misura, quest'ultime, anche se ancora presenti nell'archivio, non saranno più rintracciabili

esempi:

```
r = fMC.WriteMeasureCategory("5", "5", "length", False)
r = fMC.WriteMeasureCategory("length", "5", "length", True)
```

MISCELLANY FUNCTIONS

AddCharsToString

restituisce il testo indicato con l'aggiunta a destra o a sinistra di un carattere di riempimento replicato per 'n' volte fino a completare la lunghezza del testo desiderata come specificato nei parametri indicati

sintassi:
```
r = fMSC.AddCharsToString([testo], _
                          [lunghezza testo desiderata], _
                          [carattere di riempimento], _
                          [posizione - 0 = sinistra, 1 = destra])
```

esempi:
```
r = fMSC.AddCharsToString("123", 5, "0", 0) ' "00123"
r = fMSC.AddCharsToString("AAA", 6, "B", 1) ' "AAABBB"
```

AdjASCII

restituisce il testo indicato filtrato nei limiti dei codici ASCII indicati

sintassi:
```
r = fMSC.AdjASCII([testo], _
                  [codice ASCII (decimale) carattere limite inferiore], _
                  [codice ASCII (decimale) carattere limite superiore], _
                  [preserva il carattere CR - True/False], _
                  [preserva il carattere LF - True/False])
```

esempio:
```
r = fMSC.AdjASCII("!!!test§§§", 33, 127, False, False) ' "test"
```

CheckSum

restituisce il checksum del testo indicato, il risultato è condizionato dalla base indicata per il calcolo

sintassi:
```
r = fMSC.CheckSum([testo], [base])
```

esempi:
```
r = fMSC.CheckSum("db2000 functions - (c) by Massimo Mascalchi", 256) ' 59
r = fMSC.CheckSum("db2000 functions - (c) by Massimo Mascalchi", 65536) ' 3643
```

GeoDistance

restituisce l'indicazione della distanza di due punti terrestri relativa ai parametri indicati

sintassi:
```
r = fMSC.GeoDistance([latitudine 1], _
                     [longitudine 1], _
                     [latitudine 2], _
                     [longitudine 2], _
                     [unità di misura: "K"/"N"])
```

esempio:
```
r = fMSC.GeoDistance(43.46, 11.15, 45.28, 9.12, "K") ' 258.796334601266
```

New1

inizializza gli oggetti della libreria

sintassi:
```
fMSC.New1
```

esempio:
```
fMSC.New1
```

StrFileName

restituisce parti del nome del file indicato secondo il parametro di estrazione indicato

```
sintassi:   r = fMSC.StrFileName([nome file], [parametro di estrazione])

    [parametro di estrazione] =  0  =  nessuna operazione (restituisce lo stesso nome del file)
                                 1  =  esclude l'estensione del file
                                 2  =  restituisce l'estensione del file
                                 3  =  restituisce il nome del file escludendo il percorso
                                 4  =  restituisce il percorso del file
```

```
esempi:  r = fMSC.StrFileName("functions.doc", 0) ' "functions.doc"
         r = fMSC.StrFileName("functions.doc", 1) ' "functions"
         r = fMSC.StrFileName("functions.doc", 2) ' "doc"
         r = fMSC.StrFileName("c:\db2000\functions.doc", 3) '  "functions.doc"
         r = fMSC.StrFileName("c:\db2000\functions.doc", 4) ' "c:\db2000"
```

TIME FUNCTIONS

AddTimestring

restituisce la somma delle stringhe delle due ore indicate formattando i digit dell'ora per il numero dei caratteri indicati

```
sintassi:  r = fTIME.AddTimestring([stringa ora 1], [stringa ora 2], [numero digit ora])
esempio:   r = fTIME.AddTimestring("10:35:05", "01:35:15", 3) ' "012:10:20"
```

CheckTimestring

effettua il controllo sulla stringa dell'ora indicata restituendola formattata con i digit dell'ora indicati

```
sintassi:  r = fTIME.CheckTimestring([stringa ora], [numero digit ora])
esempio:   r = fTIME.CheckTimestring("10:35:05", 3) ' "010:35:05"
```

GetSeparatorInTimestring

restituisce il carattere separatore presente nella stringa dell'ora indicata

```
sintassi:  r = fTIME.GetSeparatorInTimestring([stringa ora])
esempio:   r = fTIME.GetSeparatorInTimestring("10:35:05") ' ":"
```

GetTimestringValues

restituisce un array di valori (ore, minuti e secondi) presenti nella stringa dell'ora indicata divisi dal separatore

```
sintassi:  ra() = fTIME.GetTimestringValues([stringa ora])
esempio:   ra() = fTIME.GetTimestringValues("10:35:05")  ' ra(0) = 10 (ore)
                                                         ' ra(1) = 35 (minuti)
                                                         ' ra(2) = 5  (secondi)
```

HoursToTimestring

trasforma le ore indicate in una stringa formattata con i parametri indicati

```
sintassi:  r = fTIME.HoursToTimestring([ore], _
                                       [numero digit ora], _
                                       [digit minuti - True/False], _
                                       [digit secondi - True/False])
esempi:    r = fTIME.HoursToTimestring(10, 3, False, False) ' "010"
           r = fTIME.HoursToTimestring(10, 3, True, False)  ' "010:00"
           r = fTIME.HoursToTimestring(10, 3, True, True)   ' "010:00:00"
```

MinutesToTimestring

trasforma i minuti indicati in una stringa formattata con i parametri indicati

```
sintassi:  r = fTIME.MinutesToTimestring([ore], _
                                         [numero digit ora], _
                                         [digit secondi - True/False])
esempi:    r = fTIME.MinutesToTimestring(635, 3, False) ' "010:35"
           r = fTIME.MinutesToTimestring(635, 3, True)  ' "010:35:00"
```

New1

inizializza gli oggetti della libreria

```
sintassi:  fTIME.New1
esempio:   fTIME.New1
```

SecondsToMinutes

restituisce in array di valori (minuti e resto dei secondi) contenente la trasformazione dei secondi indicati in minuti

```
sintassi:  r = fTIME.SecondsToMinutes([secondi])
esempio:   ra() = fTIME.SecondsToMinutes(38105)  ' ra(0) = 635  (minuti)
                                                 ' ra(1) = 5    (resto dei secondi)
```

SecondssToTimestring

trasforma i minuti indicati in una stringa formattata con i parametri indicati

```
sintassi:  r = fTIME.SecondsToTimestring([ore], _
                                         [numero digit ora])
esempio:   r = fTIME.SecondsToTimestring(38105, 3) ' "010:35:05"
```

SubTimestring

restituisce la differenza (sottrazione) delle stringhe delle due ore indicate formattando i digit dell'ora per il numero dei caratteri indicati

```
sintassi:  r = fTIME.SubTimestring([stringa ora 1], [stringa ora 2], [numero digit ora])
esempio:   r = fTIME.SubTimestring("10:35:05", "01:35:15", 3) ' "010:35:05"
```

SystemTimestringSeparator

restituisce il separatore della stringa dell'ora di sistema

```
sintassi:  r = fTIME.SystemTimestringSeparator
esempio:   r = fTIME.SystemTimestringSeparator ' ":"
```

TimestringSeparator

restituisce o imposta il separatore indicato da utilizzare nella formattazione della stringa dell'ora

```
sintassi:  fTIME.TimestringSeparator = [separatore]
           r = fTIME.TimestringSeparator
esempio:   fTIME.TimestringSeparator = "."
           r = fTIME.TimestringSeparator  ' "."
```

TimestringToHours

trasforma la stringa dell'ora indicata in ore secondo i parametri indicati

```
sintassi:  r = fTIME.TimestringToHours([stringa ora], _
                                       [base arrotondamento minuti], _
                                       [base arrotondamento secondi])
esempio:   r = fTIME.TimestringToHours("10.35.05", 30, 30) ' 11
```

TimestringToMinutes

trasforma la stringa dell'ora indicata in ore secondo i parametri indicati

```
sintassi:  r = fTIME.TimestringToMinutes([stringa ora], _
                                         [base arrotondamento secondi])
esempio:   r = fTIME.TimestringToMinutes("10.35.05", 30) ' 635
```

TimestringToSeconds

trasforma la stringa dell'ora indicata in ore secondo i parametri indicati

```
sintassi:  r = fTIME.TimestringToSeconds([stringa ora])
esempio:   r = fTIME.TimestringToSeconds("10.35.05") ' 38105
```

UTF8 FUNCTIONS

decode

restituisce il testo indicato decodificato dal formato UTF8

sintassi:
```
r = fUTF8.decode([testo])
```

esempio:
```
r = fUTF8.decode("Â§Â§Â§ db2000 functions Â§Â§Â§") ' "§§§ db2000 functions §§§"
```

---

encode

restituisce codificato nel formato UTF8 (Unicode Transformation Format, 8 bit) il testo indicato

sintassi:
```
r = fUTF8.encode([testo])
```

esempio:
```
r = fUTF8.encode("§§§ db2000 functions §§§") ' "Â§Â§Â§ db2000 functions Â§Â§Â§"
```

New1

inizializza gli oggetti della libreria

sintassi:
```
fUTF8.New1
```

esempio:
```
fUTF8.New1
```

VB LIKE FUNCTIONS

Exp

restituisce un valore double che specifica 'e' (la base dei logaritmi naturali) elevato alla potenza indicata

```
sintassi:  r = fVB.Exp([potenza])
esempio:   r = fVB.Exp(10) ' 22026.465794857
```

Fix

restituisce la parte intera del numero indicato

```
sintassi:  r = fVB.Fix([numero])
esempio:   r = fVB.Fix(2.75) ' 2
```

Hex

restituisce una stringa che rappresenta il valore esadecimale del numero indicato

```
sintassi:  r = fVB.Hex([numero])
esempio:   r = fVB.Hex(255) ' "FF"
```

InStr

restituisce la posizione della prima occorrenza della stringa indicata all'interno del testo indicato

```
sintassi:  r = fVB.InStr([testo], [stringa da trovare])
esempio:   r = fVB.InStr("Ciao Mondo!", "Mondo") ' 6
```

InStrRev

restituisce la posizione della prima occorrenza della stringa indicata all'interno del testo indicato iniziando la ricerca dalla fine di questo ultimo

```
sintassi:  r = fVB.InStrRev([testo], [stringa da trovare])
esempio:   r = fVB.InStrRev("Mondo Ciao Mondo!", "Mondo") ' 12
```

LCase

converte tutti i caratteri del testo indicato in minuscolo

```
sintassi:  r = fVB.LCase([testo])
esempio:   r = fVB.LCase("CIAO MONDO!") ' "ciao mondo!"
```

Left

restituisce una stringa a partire da sinistra del testo indicato che contiene il numero dei caratteri indicati

```
sintassi:  r = fVB.Left([testo], [numero caratteri])
esempio:   r = fVB.Left("Ciao Mondo!", 4) ' "Ciao"
```

Len
restituisce la lunghezza in caratteri del testo indicato

```
sintassi:  r = fVB.Len([testo])
esempio:   r = fVB.Len("Ciao Mondo!") ' 11
```

LSet
restituisce una stringa allineata a sinistra contenente il testo indicato adattato alla lunghezza in caratteri indicata

```
sintassi:  r = fVB.LSet([testo], [lunghezza in caratteri])
esempi:    r = fVB.LSet("Ciao Mondo!", 4)  ' "Ciao"
           r = fVB.LSet("Ciao Mondo!", 20) ' "Ciao Mondo!        "
```

LTrim
restituisce una stringa contenente una copia del testo indicato ma privo degli eventuali spazi a sinistra

```
sintassi:  r = fVB.LTrim([testo])
esempio:   r = fVB.LTrim("      Ciao Mondo!      ") ' "Ciao Mondo!     "
```

Mid
restituisce una stringa estratta dal testo indicato dalla posizione e per il numeri di caratteri indicati

```
sintassi:  r = fVB.Mid([testo], [posizione di inizio estrazione], [numero caratteri da estrarre])
esempio:   r = fVB.Mid("Ciao Mondo!", 6, 5) ' "Mondo"
```

MkDir
crea la cartella indicata, restituisce ***True*** se la cartella è stata creata correttamente altrimenti ***False***

```
sintassi:  r = fVB.MkDir([nome cartella])
esempio:   r = fVB.MkDir("c:\db2000\tmp")
```

New1
inizializza gli oggetti della libreria

```
sintassi:  fVB.New1
esempio:   fVB.New1
```

QBColor

restituisce un valore che rappresenta il codice in decimale del colore **RGB** corrispondente al numero di colore indicato

```
sintassi:   r = fVB.QBColor([colore])

   [colore] =   0  =  nero
                1  =  blu
                2  =  verde
                3  =  ciano
                4  =  rosso
                5  =  magenta
                6  =  giallo
                7  =  bianco
                8  =  grigio
                9  =  blu chiaro
               10  =  verde chiaro
               11  =  ciano chiaro
               12  =  rosso chiaro
               13  =  magenta chiaro
               14  =  giallo chiaro
               15  =  bianco brillante

 esempio:   r = fVB.QBColor(12) ' 255
```

Replace

restituisce il testo indicato aver operato la sostituzione della stringa indicata con la stringa di ricerca indicata

```
sintassi:   r = fVB.Replace([testo], [stringa da ricercare], [stringa da sostituire])

 esempio:   r = fVB.Replace("Ciao Terra!", "Terra", "Mondo") ' "Ciao Mondo!"
```

Right

restituisce una stringa a partire da destra del testo indicato che contiene il numero dei caratteri indicati

```
sintassi:   r = fVB.Right([testo], [numero caratteri])

 esempio:   r = fVB.Right("Ciao Mondo!", 6) ' "Mondo!"
```

RmDir

rimuove la cartella indicata, restituisce ***True*** se la cartella viene eliminata correttamente altrimenti ***False***

```
sintassi:   r = fVB.RmDir([nome cartella])

 esempio:   r = fVB.RmDir("c:\db2000\tmp")
```

RSet

restituisce una stringa adattata alla lunghezza in caratteri indicata a partire dalla destra del testo indicato

```
sintassi:   r = fVB.RSet([testo], [lunghezza in caratteri])

 esempio:   r = fVB.RSet("Ciao Mondo!", 20)   ' "         Ciao Mondo!"
```

RTrim

restituisce una stringa contenente una copia del testo indicato ma privo degli eventuali spazi a destra

```
sintassi:   r = fVB.LTrim([testo])

 esempio:   r = fVB.LTrim("      Ciao Mondo!      ") ' "      Ciao Mondo!"
```

Space
restituisce una stringa contenente il numero di spazi indicati

```
sintassi:  r = fVB.Space([numero di spazi])
esempio:   r = fVB.Space(10) ' "          "
```

Split
restituisce una matrice a una dimensione con base zero contenente un numero di sottostringhe come specificato nei parametri estratte dall'espressione stringa indicata

```
sintassi:  ra() = fVB.Slipt([espressione, delimitatore, nr. elementi da estrarre, tipo di confronto])
           ' se [nr. elementi da estrarre] = -1 nessun limite viene posto
           ' se [tipo di confronto] = True il confronto è di tipo testo altrimenti (False) è binario
esempio:   ra() = fVB.Split("1,2,3,4,5", ",", -1, True)
```

SplitRegex
suddivide l'espressione stringa indicata in una matrice di sottostringhe con base zero in corrispondenza delle posizioni definite dalla corrispondenza dell'espressione regolare indicata

```
sintassi:  ra() = fVB.SliptRegex([espressione stringa, espressione regolare])
esempio:   ra() = fVB.SplitRegex("1*-*2*-*3", "\*-\*")
```

Str
restituisce una stringa contenente il valore dell'oggetto indicato

```
sintassi:  r = fVB.Str([valore oggetto])
esempio:   r = fVB.Str(23.45) ' "23.45"
```

StrComp
restituisce un numero che indica il risultato del confronto tra le due stringhe indicate

```
sintassi:  r = fVB.StrComp([stringa 1], [stringa 2])  ' -1  ([stringa 1] > [stringa 2])
                                                      '  0  ([stringa 1] = [stringa 2])
                                                      '  1  ([stringa 1] < [stringa 2])
esempi:    r = fVB.StrComp("Ciao Mondo!", "Ciao!") ' -1
           r = fVB.StrComp("Ciao Mondo!", "Ciao Mondo!") ' 0
           r = fVB.StrComp("Ciao!", "Ciao Mondo!") ' 1
```

StrDup
restituisce una stringa che contiene la ripetizione del carattere indicato per il numero delle volte indicato
(come in *VB.NET*, equivale alla funzione ***String*** del *VB6*)

```
sintassi:  r = fVB.StrDup([numero ripetizioni], [carattere])
esempio:   r = fVB.StrDup(5, "A") ' "AAAAA"
```

StringEx
restituisce una stringa che contiene la ripetizione del carattere indicato per il numero delle volte indicato
(equivale alla funzione ***String*** del *VB6*)

```
sintassi:  r = fVB.StringEx([numero ripetizioni], [carattere])
esempio:   r = fVB.StringEx(5, "A") ' "AAAAA"
```

StrReverse

restituisce una stringa in cui l'ordine dei caratteri del testo indicato è stato invertito

```
sintassi:  r = fVB.StrReverse([testo])
esempio:   r = fVB.StrReverse("Ciao Mondo!") ' "!odnoM oaiC"
```

Trim

restituisce una stringa contenente una copia del testo indicato ma privo degli eventuali spazi sia destra che a sinistra

```
sintassi:  r = fVB.Trim([testo])
esempio:   r = fVB.Trim("      Ciao Mondo!      ") ' "Ciao Mondo!"
```

UCase

converte tutti i caratteri del testo indicato in maiuscolo

```
sintassi:  r = fVB.UCase([testo])
esempio:   r = fVB.UCase("ciao mondo!") ' "CIAO MONDO!"
```

Val

restituisce il valore numerico rappresentato nella stringa indicata

```
sintassi:  r = fVB.Val([stringa])
esempi:    r = fVB.Val("34.50") ' 34.5
           r = fVB.Val("&H" & "0FF") ' 255
```

XBS (maXim BASIC SCRIPT)

AddNewLine

inserisce una nuova riga alla fine dello script, restituisce ***True*** se la riga è stata inserita correttamente altrimenti ***False***

```
sintassi:  r = fXBS.AddNewLine([nuova riga dello script])
esempio:   r = fXBS.AddNewLine("A = B + 1")
```

ClearDisplayResults

svuota il buffer dei risultati

```
sintassi:  fXBS.ClearDisplayResults
esempio:   fXBS.ClearDisplayResults
```

CountLines

restituisce il numero delle righe dello script

```
sintassi:  r = fXBS.CountLines
esempio:   r = fXBS.CountLines
```

DisplayResults

restituisce il buffer dei risultati

```
sintassi:  r = fXBS.DisplayResults
esempio:   r = fXBS.DisplayResults
```

ExecuteCommand

esegue un comando ***XBS***, restituisce ***True*** se il comando è stato eseguito correttamente altrimenti ***False***

```
sintassi:  r = fXBS.ExecuteCommand([comando XBS])
esempio:   r = fXBS.ExecuteCommand("PRINT A")
```

ExecuteExpression

esegue una espressione ***XBS***, restituisce il valore (risultato) dell'espressione

```
sintassi:  r = fXBS.ExecuteExpression([espressione XBS])
esempio:   r = fXBS.ExecuteExpression("(PI * r * (r + SQR((r ^ 2) + (h ^ 2))))")
```

GetLastErrorCode

restituisce il codice dell'ultimo errore avvenuto durante l'esecuzione dello script
(vedere la **TABELLA DEI CODICI DI ERRORE**)

```
sintassi:  r = fXBS.GetLastErrorCode
esempio:   r = fXBS.GetLastErrorCode
```

GetLastErrorLine

restituisce il numero della riga dove è avvenuto l'ultimo errore durante l'esecuzione dello script

sintassi:
```
r = fXBS.GetLastErrorLine
```

esempio:
```
r = fXBS.GetLastErrorLine
```

GetLine

restituisce il contenuto della riga dello script indicata

sintassi:
```
r = fXBS.GetLine([numero della riga dello script])
```

esempio:
```
r = fXBS.GetLine(5)
```

GetVAR

recupera il valore della variabile indicata

sintassi:
```
r = fXBS.GetVAR([id variabile]) ' 0...25 ("A"..."Z")
```

esempio:
```
r = fXBS.GetVAR(5) ' recupera il valore di "F"
```

InsertLine

inserisce una nuova riga nello script nella posizione indicata, restituisce ***True*** se la riga è stata inserita correttamente altrimenti ***False***

sintassi:
```
r = fXBS.InsertLine([posizione della riga da inserire],[nuova riga da inserire])
```

esempio:
```
r = fXBS.InsertLine(5, "PRINT A + B")
```

List

restituisce il listato completo dello script

sintassi:
```
r = fXBS.List
```

esempio:
```
r = fXBS.List
```

LoadScript

carica uno script da un file esterno, restituisce ***True*** se lo script viene caricato correttamente altrimenti ***False***

sintassi:
```
r = fXBS.LoadScript([nome del file dello script da caricare], [mode])
' se mode = True inserisce lo script in coda all'eventuale script già presente
```

esempio:
```
r = fXBS.LoadScript(AppPath & "\XBS samples\test.xbs", False)
```

MaxLines

imposta o restituisce il numero massimo delle righe consentite in uno script (per default è 100)

sintassi:
```
fXBS.MaxLines = [numero delle righe] ' imposta il numero delle righe consentite
r = fXBS.MaxLines ' recupera il numero delle righe consentite
```

esempi:
```
fXBS.MaxLines = 500
r = fXBS.MaxLines
```

New1

inizializza gli oggetti della libreria

sintassi:
```
fXBS.New1
```

esempio:
```
fXBS.New1
```

NewScript
reset dell'area di lavoro dello script (predispone l'area di lavoro per un nuovo script)

sintassi:
```
fXBS.NewScript
```

esempio:
```
fXBS.NewScript
```

RemoveLine
rimuove (elimina) la riga specificata dallo script, restituisce ***True*** se la riga viene rimossa correttamente altrimenti ***False***

sintassi:
```
r = fXBS.RemoveLine([numero della riga da eliminare dallo script])
```

esempio:
```
r = fXBS.RemoveLine(5)
```

ResultsCRLF
imposta o restituisce la stringa contenente i caratteri di controllo di fine riga per il buffer dei risultati

sintassi:
```
fXBS.ResultsCRLF = [stringa contenente i caratteri di controllo di fine riga]
r = fXBS.ResultsCRLF
```

esempi:
```
fXBS.ResultsCRLF = Chr(13) & Chr(10) ' imposta i caratteri di controllo di fine riga
r = fXBS.ResultsCRLF ' recupera i caratteri di controllo di fine riga
```

Run
esegue lo script

sintassi:
```
r = fXBS.Run([True/False])
                ' se True svuota il buffer dei risultati prima dell'esecuzione
```

esempio:
```
r = fXBS.Run(True)
```

SaveScript
salva lo script in un file esterno

sintassi:
```
r = fXBS.SaveScript([nome del file  dello script da salvare])
```

esempio:
```
r = fXBS.SaveScript(AppPath & "\XBS samples\test.xbs")
```

SetVAR
assegna il valore indicato alla variabile indicata

sintassi:
```
r = fXBS.SetVAR([id variabile],[valore])
```

esempio:
```
r = fXBS.SetVAR(5, "3.01") ' assegna a "F" il numero "3.01"
```

UpdateLine
aggiorna/modifica il contenuto della riga dello script indicata

sintassi:
```
r = fXBS.UpdateLine([posizione della riga da aggiornare],[contenuto riga modificata])
```

esempio:
```
r = fXBS.UpdateLine(5, "PRINT A - B")
```

VARS

imposta o restituisce i valori di tutte le variabili

sintassi:

```
fXBS.VARS = [array valori variabili] ' assegna i valori di tutte le variabili
ra() = fXBS.VARS ' recupera i valori di tutte le variabili
```

esempi:

```
fXBS.VARS = ra()
ra() = fXBS.VARS
```

XBS – LE ISTRUZIONI E LE FUNZIONI DELLO SCRIPT

OPERATORI E FUNZIONI MATEMATICHE				
{[(	)]}	+	-	*
/	^	\	|	&
<	>	=	==	<=
>=	<>	AND	NOT	OR
XOR	EQV	IMP	MOD	ABS()
ACOS()	ACOSH()	ACOT()	ACOTH()	ACSC()
ACSCH()	ASEC()	ASECH()	ASIN()	ASINH()
ATAN()	ATANH()	COS()	COSH()	COT()
COTH()	CSC()	CSCH()	DEG()	EXP()
FAC()	FIX()	INT()	ISQR()	LENNUM()
LN()	LOG()	RAD()	SEC()	SECH()
SGN()	SIN()	SINH()	SQR()	TAN()
TANH()	VAL()			

FUNZIONI PER IL TRATTAMENTO DELLE STRINGHE ALFANUMERICHE				
ASC()	CHR()	FORMAT()	HEX()	INSTR()
INSTRREV()	LCASE()	LEFT()	LEN()	LSET()
LTRIM()	MID()	REPLACE()	RIGHT()	RSET()
RTRIM()	SPACE()	STR()	STRCOMP()	STRING()
STRREVERSE()	TRIM()	UCASE()		

FUNZIONI PER IL CONTROLLO DEI FILE				
CLOSE()	EOF()	FILEEXIST()	LOF()	MKDIR()
RMDIR()				

FUNZIONI DI USO GENERALE				
GETSYSDATE()	GETSYSTIME()	MSGBOX()	TIMER()	VAR()

ISTRUZIONI
' commento
<label>: (etichetta di riga)
BEEP
CLEAR
CLOSEALL / **RESET**
CLS
DIGITS [<exp>]
DO [{**WHILE** | **UNTIL**} <exp>] ... [**EXIT DO**] ... **LOOP** [{**WHILE** | **UNTIL**} <exp>]
END
FOR <var> **=** <exp> **TO** <exp> [**STEP** <exp>] ... [**EXIT FOR**] ... **NEXT** [<var>]
GOSUB <label> ... **RETURN**
GOTO <label>
IF <exp> **THEN** <statement> [**ELSE** <statement>]
IF <exp> **THEN** ... [**ELSEIF** <exp> **THEN**] ... [**ELSE**] ... **ENDIF**
INCLUDE <filename>
INPUT [;] [<str>{,|;}] <var> [,<var>[,var]...]
INPUT #<filenum>, <var> [,<var> [,var]...]
OPEN <filename> **FOR** {**INPUT** | **OUTPUT** | **APPEND**} **AS** #<filenum>
PRINT [{<exp>|<str>|**TAB**(<exp>)|**SPC**(<exp>)}[<,|;><stuff>]] [,|;]
PRINT #<filenum>, [<stuff>[<,|;><stuff>]] [,|;]
PRINTMSG [{<exp>|<str>|**TAB**(<exp>)|**SPC**(<exp>)}[<,|;><stuff>]] [,|;]
RANDOMIZE [<exp>|**TIMER**]
WHILE <exp> ... [**EXIT WHILE**] ... **WEND**

TABELLA DEI CODICI DI ERRORE (vedere la funzione GetLastErrorCode)	
CODICE	**DESCRIZIONE**
< 10000	errori intercettati e codificati dal compilatore
10000	script non valido
10001	assegnazione errata
10002	parametri non validi o struttura della riga non valida
10003	superato il limite dello stack o variabile autoreferenziata
10004	indice dell'array fuori limite
10005	errore di apertura file
10006	canale del file non valido (è possibile utilizzare solo fino a 9 file contemporaneamente, 1...9)
10007	canale del file già in uso
10008	modo di apertura del file non valido
10009	impossibile valutare la variabile
10010	divisione per zero
10011	operazione MOD con zero o non definita
10012	funzione errata o non definita
10013	logaritmo errato o non definito
10014	funzione errata o non definita
10015	funzione trigonometrica errata o non definita
10016	operazione errata, non valida o non riconosciuta
10017	(riservato)
10018	(riservato)
10019	(riservato)
10020	nessuno script risulta caricato in memoria
10021	errore di sintassi
10022	ELSE senza IF
10023	errore nella condizione IF...ENDIF
10024	errore nell'istruzione di EXIT
10025	errore nel ciclo FOR...NEXT
10026	errore nella condizione WHILE...WEND
10027	errore nel ciclo DO...LOOP
10028	etichetta (label) errata o non valida
10029	doppi apici mancanti (")
10030	delimitatore mancante (“,” o “;”)
10031	impossibile valutare l'espressione
10032	le righe dello script eccedono del massimo consentito (vedere la funzione MaxLines)
10033	(riservato)
10034	(riservato)
10035	(riservato)
10036	(riservato)
10037	(riservato)
10038	(riservato)
10039	(riservato)
10040	errore di runtime
10041	(riservato)
10042	errore generico nello stack durante l'esecuzione dello script
10043	trovato stack vuoto durante l'esecuzione dello script
10044	superato il limite dello stack durante l'esecuzione dello script
10045	(riservato)
10046	variabile utente errata o non valida, sono consentite solo 26 variabili (A...Z), VAR(0) ... VAR(25)
10047	numero errato o non valido

the **db2000** team