db2000 maXim Basic Script

© 2008 Massimo Mascalchi
massimo.mascalchi@db2000web.net

DESCRIZIONE DELLE FUNZIONI

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 – COMMANDS & FUNCTIONS 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


db2000