Monte Carlo Software

C. H. A. R. L. I. E.™
for
Microsoft® Word for Windows™

The CHARLIE (**C**ontemporary **H**elp **A**t **R**eal-**L**ife **I**nformation **E**nvironments) for Word for Windows is a comprehensive set of utilities that will enhance your interface with the word for windows software, and will aid in the programming of macros for this environment.

This Software consists of a library with different groups of routines, and of a set of complete programs to aid you in using word for windows.

The Library - MCS.

Group I Environment Functions for use within Word for Windows

Clean the window as much as possible for use with macro - user dialogs.

WordClean(n, p$, k$, i$)
n : If 1 then clean the screen else (0) reset the screen.
If 0 (reset) the following parameter will not be used, the situation from before the last call with -1- will be reinstated.
p$ is text which will replace the window title.
k$ is text which will replace the menu's under the title.
i$ is text which will be in the status bar.

Stop updating the screen, and restart this.

Echo(n)
n: If 0 then stop updating, else (1) restart updating.
Warning : Always use this in pairs, otherwise the connection to the window will be lost, and you'll be in big trouble.

Store and retrieve global variables (global to word, and not one macro)

Txt$ = GetGlob$(Name$, Erase)
Txt$ is the receiving variable for the string
Name$ is the name of the string you want to get
If Erase is -1 then the global variable will be erased, otherwise it will be retained.
&
PutGlob(Name$, Value$)
Name$ is the name of the variable you want to create or update
Value$ is the string you want to store in the variable

Note that these vars stay in existance, even after word and/or windows is closed, if they are not deleted.

Set & Reset the title bar of the word for windows window

Txt$ = WinTit$(Text$)
Txt$ will be filled with the old window-title text.
Text$ is the new title text for the window. If You have a clock like f.i. “capclock” running in your title bar, you have to disable it, otherwise you might run into problems.

Start and Stop a clock running in your menu between Windows & Help
SetClock(n)
If n is Yes (-1) then the clock will be started, else it will be stopped.
Put a startup command for this like
Sub Main
SetClock(-1)
End Sub
Into Your AutoExec macro in your normal.dot file if you want the clock to run allways when you are in word.

Code and Decode a selected piece of text in your document
HideTxt(n)
n is -1 means code the text, else decode him. Watch it, this routine uses your Word for Windows data (personal data), i.e. if you change anything there after code, you wont be able to decode that piece of text, and it will be lost for ever.

Group II String Handling Functions

Break up a string in its parts
Txt$ = Part$(Source$, Ptr$)
Txt$ is the string which receives the part extracted
Source$ is the string to search, the first extracted part will be elevated from it (including the pointer)
Ptr$ is the string to use as delimiter of the components.

Search strings for a pattern and optionally replace with a pattern
Result = Medit(Source$, Pat1$, Pat2$)
Result is the return code (-1 = O.K., 0 = False) or the position of the pattern in the string.
Source$ is the string to be searched
Pat1$ is the pattern to search with and is defined as follows :

abc	*Matches the string ‘abc’*
[abc]	*Matches a OR b OR c*
*	*Matches any number of the previous f.i [ab]* matches 1 to n times the ab string (a or b).*
[a-z]	*Matches all letters from and inclusive a to z*
?	*Matches any one character position*
\	*Is used to make the following (^\ etc.) stand for its self*
$	*Marks the begin of the string source$*
^	*Marks the end of the string source$*
&	*Marks the start of a word within source$*
@	*Marks the end of a word within source$*

Example : &[A-Za-z][0-9,] Selects all words within the source$ which start with an upper or lower-case letter and after that consists of any number of numerics or an ‘,’.*
Pat2$ is the pattern to replace with, or “” in which case only the position

is returned in ‘Result’. This pattern is defined as with pattern one, under matching circumstances the replacement pattern will be adjusted to the search pattern.
f.i If you have a search pattern of ‘&[a-z]’ all words, beginning with a small letter will be selected. If in This case your replacement pattern is :
‘&[A-Z]’ The first letter of each word will be capitalized.
‘&Monkey’ The first letter of each word will be replaced by the sequence ‘Monkey’
‘Monkey’ The selected words will be replaced by ‘Monkey’
The working of Pat1$ and Pat2$ can be described as follows:
If 1 & 2 Contain a set of [ ], then the items between them must be the same in number (f.i. [ABC] and [abc] or [1-5] and [A-E]) Furthermore , when you want this replacement done they must be equal (both with our without the [] set). Also some extra directives (like &, ^ etc.) can only be used under certain - matching - circumstances. Only if you want to replace completely, you can have different lengths. The combinations can be summarized as follows :

Rep/S	**$**	**^**	**&**	**@**	**Nothing**
$	Repl. match	Insert Before source			Replace match
^	Add to source	Repl. Match			Replace match
&			Repl. Match	Insert before word	Replace match
@			Insert after word	Repl. Match	Replace match
Not.	Repl. match	Repl. match	Repl. match	Repl. Match	Repl. match

Where there is nothing noted in the above table, that possibility does not exist, or has no meaning.

Strip (groups of) characters from a string or part of a string
Result = Strip(Source$, Chars$, Place$)
Result = 0 - no characters found, or the number of characters stripped from the string.
Source$ is the string to be searched.
Chars$ is the string to search with. This may contain any number of characters or special characters but must be smaller as the Source string. There are some special cases to search for, they can be commanded with the following entry's in Chars$:

Chars$	**Means, strip all :**
N_A	Not Alphabetic characters
N_U	Not Numeric characters
A_A	Alphabetic characters

U_U	Numeric characters
A_U	Alpha-Numeric characters
U_A	Not Alpha-Numeric characters
N_S	Not ASCII characters
S_S	ASCII characters

Where Alphabetic means a-z and A-Z, numeric is 0-9, alphanumeric is they both and ASCII is all plus special symbols like +, -, [etc. below 127 Decimal.
Place$ is the area of the source string that has to be stripped, this can be defined as follows , where by you should consider that certain combinations are only meaningful if your source string contains it's delimiters. f.i the option "N" works only correct if Your searched item exist minimal 3 times (separated) in the source string to be searched.
The Table is as follows :

Place$	**Means :**
B	Only begin of source
E	Only end of source
O	Both, begin and end of source
Y	Everywhere in source
N	Not in Begin or End of source

Sort An array of strings

Since Word basic does not allow for transfer of arrays between routines and functions, You have to xfer this routine into your code. It's also available under the name "MCSSort" in the list of macro's. This fast sort routine is based on Don Knuth, ("The Art of Programming" Vol. 3, Addison-Wesley, 1975). It was slightly modified to include element 0 in the sort.

Identify a strings contents, select it's group

Ret = Ident(String$)
Ret Is the Returncode, indicating the components in the string "String" as follows :

Return Cde	**Means :**

0	No Classification
1	Only UpperCase Alpha ASCII Characters
2	Only LowerCase Alpha ASCII Characters
3	Only Numeric characters
4	Only Uppercase Alpha
5	Only Lowercase Alpha
6	Only Alpha ASCII
7	Only Alpha
8	Only Alpha-Numeric Ascii Characters
9	Only Alpha-Numeric Characters
10	Only ASCII
11	Only Printable
9999	Empty String

We tried to include all foreign letters in the Alpha checks that does not include the word ASCII, however due to inconsistency of fonts this might not be 100% correct. The blank is considered a valid ASCII alpha character.

Find the last occurrence of a string in a string

Ret = LastInstr(C$, M$)
Ret is the start position of the last occurrence of the string M$ in the string C$, or Null if there is none.

Group III Numeric Functions

See if a number is even or odd

Res = Even(Number)
Res is the resultcode : -1 is even, 0 is odd
Number is the number to be tested

Round a number on N decimal places exact

Rest = Round(Num, Plcs)
Rest = stripped amount to be able to round off
Num is the Number to be rounded (in place!) and

Plcs is the number of decimal places requested
When places is negative rounding will be before the decimal point. f.i. Round(1234.56, -3) will generate 1000. Plcs in itself is rounded to the nearest place (i.e. 1.3 will be 1). If the rounding is on 0 or on to large numbers, nothing will be done.
A last example : Round (12.123, 2) will result in 12.12 with a rest returned of 0.003.

Round off to the nearest multiple of n, either above or below multiple.
Rest = RoundM(Num, Factor)
Rest = the remaining difference between the multiple of Factor and Num.
Num is the number to be changed to the nearest multiple.
Factor is the factor to multiply. If the factor is positive, the next higher multiple of Factor will be chosen, else (factor is negative) the next lower multiple of factor will be chosen.

Format a number as a Dollar amount following windows rules.
Res$ = Dollar$(Amount)
Res$ is the formatted string with the result of the conversion.
Amount is the amount you want to have converted.
This function follows the rules for formatting (Currency, decimal point, layout negative numbers, etc. etc.) as fixed by the International section of the Control panel from windows.
i.e. If you fixed there the format for French currency the following conversion could occur :
-123456.78 ----> -123.456,78 Ff. Or 13 ---> 13,00 Ff.

Give the ordinal of a certain number
Res$ = Ordi$(Number)
Res$ is the result (3rd, 5th etc.) based on the number
Number is the number to analyze. This function will give you the ordinates for the language in which you are currently working. So it will adjust to French, English, Dutch, German. Other languages are under way. Watch it for French , there the ordinals must be written as Superscript, and with the number one, are dependent on what follows (Masc. of Fem.) which produces : er or ère. which cannot be done by this function, that's why both are given with a blank in-between. Use either one of them.

Translate a number in words in the current language.
Res$ = Trans$(n)
n is the number to be written in text. It might be from 0 to 999 999 999 (1 Billion - 1), no negatives, no decimal places.
Res$ is the resulting string. The number will be in words (i.e. onethousend threehundert eighteen). It will have no capitals, and milions, thousends and below will be separated by blanks. The text will be in the language where the selection of your text is written in. At the moment are available : English, Dutch. German and French.

Perform diverse calculations with two numbers
Res = Math(N1, N2, Fu$)
Res is the field containing the result of the calculation

N1 and N2 are the numbers to calculate with, and
Fu$ is the 3 letter command code. The available commands are:
“scm” - Gives the smallest common multiple of N1 and N2
“max” - Returns 1 if N1 larger as N2, 2 if it is the other way,
and 0 if both are equal.
“min” - Returns 1 if N1 is smaller as N2, 2 if it’s the other way
around, and 0 if both are equal
“scd” - Returns the smallest common devisor
“lcd” - Returns the largest common divisor
“roo” - Returns The $N2^{th}$ root of N1.
For these last three functions the precision will be as high as needed, but maximal Word-basic’s limit of 12 positions after the decimal point. If Your numbers are to big, an overflow exemption will occur.

Format a number according to windows standard or a mask.
Res$ = FPrint$(Num, Mask$)
Res$ is the formatted string that is returned.
Num is the number that is to be formatted, it might be positive, negative and have any number of decimal places. The highest- / lowest-number is that of windows basic. The mask$ is giving the rules for the formatting. If it’s an empty string (““) then the rules as defined by windows-control panel for the format of numbers will be used to format the string. Else the following characters may be applied :

Character	**Meaning / Effect**
6	In this Position either a 0 if no digit in the source string, or the digit of the source string will printed
9	In this position a digit will only be printed if there are digits not equal to zero to the left.
*	All “empty” positions in the string to the left of the first number will be filled with *, rather than “ “
-	If the number is negative, a minus will be written just before it.
( )	If the number is negative it will be written inside brackets
, (if thousand separator in Windows)	Inserts a thousand separator if there are numbers to the left of it.
. (if decimal point in Win.)	Inserts a decimal point in your string
0	Inserts a digit, or 0 if no digit exists in this position

Some examples of usage of the previously defined parameters :
12,345,678.90 = Fprint$(12345678.9, “-999,999,999.00”)
-12,6478 = Fprint$(-12.64782, “-999.999,6666”)
12.6479 = Fprint$ (12.64786, “-99,6666”)
0.12 = Fprint$(0.123, “-999,999,999.66”)

12,345.66 = FPrint$(12345.66, “”)
This last one presuming that your windows setup International has defined the format like this.

Group IV Functions relating to External (visavis. Word) components

Get the Dos environment into a string
Res$ = DosSet$
Res$ is the receiving string which will contain a list of all variables from the DOS environment (Type SET, under DOS) delimited by a “|” character.

Group V Date functions. These are implemented in seperate Macro’s (MCSDate) which makes it easyer to use them to insert dates etc. in your text. You can just attach them to a button or a menu item.

Get a formatted (system) date in a string or your document.
MCSDate Is the form which will insert the date direct into your text.
MCSDate.FDate$(Date$, Mask$)
Gives You back the formatted string of the date. The format will (if words are used) be in the language of the current document, or selection. The macro knows dates from 1/1/1950 till 31/12/2049. If the date you supply (the year part of it) is under 50, it will presume a date in the 2th century, else it will be 19xx.
If you fill in a date, then do this in the short date format your windows system knows. (Control-panel International settings) Normally this is mm/dd/yy. Give the date wanted exact like this. If you don't give any date (“”) the system date (today) will be used. The same, with the mask (Mask$), if you don't give any the windows system mask for long dates will be used. That is also set in Control panel - International section. The Mask variable you can give are the following :

Entry given	**Causes following output**
d	Day of the month like ‘3’ or ‘12’
dd	Day of the month like ‘03’ or ‘10’
ddd	Day of the week in short form like ‘Sun’ or ‘Fry’
dddd	Day of the week like ‘Sunday’ or ‘Wednesday’
m	Month of the year like ‘2’ or ‘12’
mm	Month of the year like ‘02’ or ‘10’
mmm	Month name in short form like ‘Jan’ or ‘ Feb’
mmmm	Name of the Month like ‘April’ or ‘September’

yy	The year number like ‘50’ or ‘93’
yyy	The year in letters like ‘Two thousand twelve’
yyyy	The year like ‘1993’ or ‘2015’
‘	Marks the beginning and ending of a literal string
Any other	Within literal string will be copied -> output.