T I M E   T O   W I N   L i g h t (version 3.52)

Contents

@Blank

Purpose :

Declare Syntax :

Call Syntax :

Where :

Comments :

Examples :

See also :

AddD

Purpose :

AddD adds a constant value to all of the elements of a Double array.

Declare Syntax :

Declare Function cAddD Lib "t2wlight.dll" (array() As Double, ByVal nValue As Double) As Integer

Call Syntax :

status = cAddD(array(), nValue)

Where :

array()	is the Double array.
nValue	is the value to add (if positive) or to substract (if negative) to all of the elements of the Double array.

Comments :

See Also : cAddD, cAddI, cAddL, cAddS, Array routines

AddI

Purpose :

AddI adds a constant value to all of the elements of an Integer array.

Declare Syntax :

Declare Function cAddI Lib "t2wlight.dll" (array() As Integer, ByVal nValue As Integer) As Integer

Call Syntax :

status = cAddI(array(), value)

Where :

array()	is the Integer array.
nValue	is the value to add (if positive) or to substract (if negative) to all of the elements of the Integer array.

Comments :

See Also : cAddD, cAddI, cAddL, cAddS, Array routines

AddL

Purpose :

AddL adds a constant value to all of the elements of a Long array.

Declare Syntax :

Declare Function cAddL Lib "t2wlight.dll" (array() As Long, ByVal nValue As Long) As Integer

Call Syntax :

status = cAddL(array(), value)

Where :

array()	is the Long array.
nValue	is the value to add (if positive) or to substract (if negative) to all of the elements of the Long array.

Comments :

See Also : cAddD, cAddI, cAddL, cAddS, Array routines

AddS

Purpose :

AddS adds a constant value to all of the elements of a Single array.

Declare Syntax :

Declare Function cAddS Lib "t2wlight.dll" (array() As Single, ByVal nValue As Single) As Integer

Call Syntax :

status = cAddS(array(), value)

Where :

array()	is the Single array.
nValue	is the value to add (if positive) or to substract (if negative) to all of the elements of the Single array.

Comments :

See Also : cAddD, cAddI, cAddL, cAddS, Array routines

AddTime

Purpose :

AddTime retrieves only the part for hours on one day.

Declare Syntax :

```
Declare Function cAddTime Lib "t2wlight.dll" (ByVal Hr As Integer) As Integer
```

Call Syntax :

```
test = cAddTime(Hr)
```

Where :

Hr	is the total minutes
test	is the result value.

Comments :

Examples :

```
test = cAddTime(1439+2)
        -> test = 1

test = cAddTime(2-4)
        -> test = 1438
```

See also : Date, Hour and Time routines

ArabicToRoman

Purpose :

ArabicToRoman converts an integer or a long integer into Roman representation

Declare Syntax :

```
Declare Function cArabicToRoman Lib "t2wlight.dll" (Var As Variant) As String
```

Call Syntax :

```
test = cArabicToRoman(var)
```

Where :

var	is the integer or long integer value
test	returns the Roman representation of var

Comments :

The string returned by this function is always in lowercase

Examples :

```
test = cArabicToRoman(1994)
        test -> MCMXCIV

test = cArabicToRoman(1995)
        test -> MCMXCV

test = cArabicToRoman(1993)
        test -> MCMXCIII
```

ArrayPrm

Purpose :

ArrayPrm retrieves the definition of a gived array (only one dimension and for numeric array)

Declare Syntax :

Declare Function cArrayPrm Lib "t2wlight.dll" (array() As Any, nArray As Any) As Integer

Call Syntax :

status% = cArrayPrm(array(), nArray)

Where :

array()	the array to proceed
nArray	a type variable 'ArrayType' for receiving the definition
status%	always TRUE

Comments :

The definition of an array is gived by the following parameters :

Bounds	is the far address of the array in memory.
LBound	is the smallest available subscript for the first dimension of the array.
UBound	is the highest available subscript for the first dimension of the array.
ElemSize	is the size of the element of the array
IndexCount	is the number of dimension of the array.
TotalElem	is the number of element in the array (UBound - LBound + 1) in the first dimension.

Examples :

```
Dim array(1 To 16)          As Integer
Dim arrayDef                as ArrayType
status% = cArrayPrm(array(), arrayDef)
        array1.Bounds               is 1048577
        array1.LBound               is 1
        array1.UBound               is 16
        array1.ElemSize             is 2 (INTEGER)
        array1.IndexCount           is 1
        array1.TotalElem            is 16

Dim array(-7 To 25)         As Double
Dim arrayDef                as ArrayType
status% = cArrayPrm(array(), arrayDef)
        array1.Bounds               is 1703929
        array1.LBound               is -7
        array1.UBound               is 25
        array1.ElemSize             is 8 (DOUBLE)
        array1.IndexCount           is 1
        array1.TotalElem            is 33

Dim array(-10 To 10, 1 TO 7)        As Long
Dim arrayDef                as ArrayType
status% = cArrayPrm(array(), arrayDef)
        array1.Bounds               is 458753
        array1.LBound               is 1
        array1.UBound               is 7
        array1.ElemSize             is 4 (SINGLE)
        array1.IndexCount           is 2
        array1.TotalElem            is 7
```

See also : Constants and Types declaration

Between

Purpose :

Between checks to see if a value is between two other values.

Declare Syntax :

Declare Function cBetween Lib "t2wlight.dll" (Var As Variant, Var1 As Variant, Var2 As Variant) As Integer

Call Syntax :

test = cBetween(var, var1, var2)

Where :

var	value to test
var1	first value
var2	second value
test	TRUE if var is between var1 and var2
	FALSE if var is not between var1 and var2

Comments :

var, var1, var2 are Variant value. In this routine, only Integer, Long, Single, Double are supported.

Examples :

```
var = 5
var1 = 1
var2 = 10
test = cBetween(var, var1, var2)
        -> test = TRUE

var = 10
test = cBetween(var, var1, var2)
        -> test = TRUE
```

See Also : cTrueBetween

BlockCharFromLeft

Purpose :

BlockCharFromLeft reads n chars from the left of a string.

Declare Syntax :

Declare Function cBlockCharFromLeft Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer) As String

Call Syntax :

Test = cBlockCharFromLeft(Txt, Position)

Where :

Txt	the string to extract some left chars
Position	the number of chars to read
Test	the result

Comments :

This fonction is the same that Left$(Txt, Position) but doesn't generate an Error if a problem occurs.

Examples :

```
Txt = "ABCDEF"
Position = 3
Test = cBlockCharFromLeft(Txt, Position)
        Test = "ABC"
```

See also : cBlockCharFromLeft, cBlockCharFromRight, cOneCharFromLeft, cOneCharFromRight

BlockCharFromRight

Purpose :

BlockCharFromRight reads n chars from the right of a string.

Declare Syntax :

Declare Function cBlockCharFromRight Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer) As String

Call Syntax :

Test = cBlockCharFromRight(Txt, Position)

Where :

Txt	the string to extract some right chars
Position	the number of chars to read
Test	the result

Comments :

This fonction is the same that Right$(Txt, Position) but doesn't generate an Error if a problem occurs.

Examples :

```
Txt = "ABCDEF"
Position = 3
Test = cBlockCharFromRight(Txt, Position)
        Test = "DEF"
```

See also : cBlockCharFromLeft, cBlockCharFromRight, cOneCharFromLeft, cOneCharFromRight

ChDir

Purpose :

ChDir changes the directory.

Declare Syntax :

Declare Function cChDir Lib "t2wlight.dll" (ByVal lpDir As String) As Integer

Call Syntax :

status = cChDir(lpDir)

Where :

lpDir	is the new directory
status	TRUE is all is OK
	<> TRUE is an error occurs

Comments :

This fonction is the same that ChDir but doesn't generate an VB Error if a problem occurs.

See also : cChDrive

ChDrive

Purpose :

ChDir changes the drive.

Declare Syntax :

Declare Function cChDrive Lib "t2wlight.dll" (ByVal lpDrive As String) As Integer

Call Syntax :

status = cChDrive(lpDrive)

Where :

lpDrive	is the new drive
status	TRUE is all is OK
	<> TRUE is an error occurs

Comments :

This fonction is the same that ChDrive but doesn't generate an Error if a problem occurs.

See also : cChDir

CheckChars

Purpose :

CheckChars verifies that all chars specifien are present in a string.

Declare Syntax :

```
Declare Function cCheckChars Lib "t2wlight.dll" (Txt As String, charSet As String) As Integer
```

Call Syntax :

```
status = cCheckChars(Txt, charSet)
```

Where :

Txt	the string to proceed
charSet	the chars to be verified
status	TRUE if all chars specifien in charSet are present in Txt
	FALSE if all chars specifien in charSet are not present in Txt

Comments :

Examples :

```
Txt = "ABCDEFG"
charSet = "CAD"
status = cCheckChars(Txt, charSet)
        status = TRUE

Txt = "ABCDEFG"
charSet = "CADZ"
status = cCheckChars(Txt, charSet)
        status = FALSE
```

FilterX

Purpose :

FilterBlocks removes one or more sub-string separated by two delimitors in a gived string.
FilterChars removes some chars specifien in a gived string.
FilterFirstChars removes some chars beginning at first position of a gived string.
FilterNotChars removes all chars except speficien chars in a gived string.

Declare Syntax :

```
Declare Function cFilterBlocks Lib "t2wlight.dll" (Txt As String, Delimitor As String) As String
Declare Function cFilterChars Lib "t2wlight.dll" (Txt As String, charSet As String) As String
Declare Function cFilterFirstChars Lib "t2wlight.dll" (Txt As String, charSet As String) As String
Declare Function cFilterNotChars Lib "t2wlight.dll" (Txt As String, charSet As String) As String
```

Call Syntax :

```
test = cFilterBlocks(Txt, Delimitor)
test = cFilterChars(Txt, charSet)
test = cFilterFirstChars(Txt, charSet)
test = cFilterNotChars(Txt, charSet)
```

Where :

Txt	the string to proceed
Delimitor	two chars for filter the string
charSet	the chars for filter the string
test	the result

Comments :

Examples :

```
Txt = "A/BC/DEF/GHIJ"
Delimitor = "//"
test = cFilterBlocks(Txt, Delimitor)
        test = "ADEF"
```

```
Txt = "A/BC/DEF/GHIJ"
Delimitor = "BI"
test = cFilterBlocks(Txt, Delimitor)
        test = "A/J"
```

```
Txt = "A/BC/DEF/GHIJ"
charSet = "B/"
test = cFilterChars(Txt, charSet)
        test = "ACDEFGHIJ"
```

```
Txt = "A/BC/DEF/GHIJ"
charSet = "AF/"
test = cFilterChars(Txt, charSet)
        test = "BCDEGHIJ"
```

```
Txt = "A/BC/DEF/GHIJ"
charSet = A/"
test = cFilterFirstChars(Txt, charSet)
        test = "BC/DEF/GHIJ"
```

```
Txt = "A/BC/DEF/GHIJ"
charSet = "A/BC/"
test = cFilterFirstChars(Txt, charSet)
        test = "DEF/GHIJ"
```

```
Txt = "A/BC/DEF/GHIJ"
charSet = "B/"
test = cFilterNotChars(Txt, charSet)
        test = "/B//"
```

```
Txt = "A/BC/DEF/GHIJ"
charSet = "AF/"
test = cFilterNotChars(Txt, charSet)
        test = "A//F/"
```

CheckNumericity

See cIsDigit

CheckTime

Purpose :

CheckTime verifies if an hour (in minutes) is between two others hours (in minutes)

Declare Syntax :

Declare Function cCheckTime Lib "t2wlight.dll" (ByVal Hr As Integer, ByVal Hr1 As Integer, ByVal Hr2 As Integer) As Integer

Call Syntax :

test = cCheckTime(Hr, Hr1, Hr2)

Where :

Hr	the hour (in minutes) to test
Hr1	the first hour
Hr2	the second value
test	TRUE if Hr is between Hr1 and Hr2

Comments :

Examples :

```
Hr = 1439       (23:59)
Hr1 = 1400      (23:20)
Hr2 = 10 (00:10)
test = cCheckTime(Hr, Hr1, Hr2)
        -> test = TRUE

Hr = 120 (02:00)
test = cCheckTime(Hr, Hr1, Hr2)
        -> test = FALSE
```

See also : cBetween, cTrueBetween, Date, Hour and Time routines

FileLastX

Purpose :

These routines read the date/time for a specified file.

Declare Syntax :

```
Declare Function cFileDateCreated Lib "t2wlight.dll" (ByVal lpFilename As String) As String
Declare Function cFileLastDateAccess Lib "t2wlight.dll" (ByVal lpFilename As String) As String
Declare Function cFileLastDateModified Lib "t2wlight.dll" (ByVal lpFilename As String) As String
Declare Function cFileTimeCreated Lib "t2wlight.dll" (ByVal lpFilename As String) As String
Declare Function cFileLastTimeAccess Lib "t2wlight.dll" (ByVal lpFilename As String) As String
Declare Function cFileLastTimeModified Lib "t2wlight.dll" (ByVal lpFilename As String) As String
```

Call Syntax :

```
test = cFileDateCreated(lpFilename)
test = cFileLastDateAccess(lpFilename)
test = cFileLastDateModified(lpFilename)
test = cFileTimeCreated(lpFilename)
test = cFileLastTimeAccess(lpFilename)
test = cFileLastTimeModifed(lpFilename)
```

Where :

lpFileName	the file to read date and/or time	
test	HH:MM	for time
	DD/MM/YYYY	for date

Comments :

The created, access, modified time/date are the same. The different routines are present for future version of Windows.

Compact

Purpose :

Compact compacts a string composed of numeric chars.

Declare Syntax :

Declare Function cCompact Lib "t2wlight.dll" (Txt As String) As String

Call Syntax :

test = cCompact(Txt)

Where :

Txt	is the string (only numeric chars) to compact
test	returns the string compacted

Comments :

If the size of the string is not a multiple of 2, the size used is the nearest below multiple of 2.

Examples :

```
Txt = "39383736353433323130"
test = cCompact(Txt)
        test = "9876543210"
```

See also : cUncompact

Compress

Purpose :

Compress removes all chr$(0):ASCII NULL, chr$(9):TAB, chr$(32):SPACE from a string

Declare Syntax :

Declare Function cCompress Lib "t2wlight.dll" (Txt As String) As String

Call Syntax :

test = cCompress(Txt)

Where :

Txt	the string to proceed
test	the string returned without any chr$(0), chr$(9), chr$(32)

Comments :

See also : cCompressTab, cExpandTab

CompressTab

Purpose :

CompressTab packs all n space chars into a tab char.

Declare Syntax :

Declare Function cCompressTab Lib "t2wlight.dll" (Txt As String, ByVal nTab As Integer) As String

Call Syntax :

test = cCompressTab(Txt, nTab)

Where :

Txt	the string to proceed
nTab	the number of space chars to replace by a tab char
test	the result

Comments :

Examples :

```
Txt = "A" + space$(2) + "B" + space$(3) + "C" + space$(4) + "D"
nTab = 2
test = cCompressTab(Txt, nTab)
        test = "A" + chr$(9) + "B" + chr$(9) + space$(1) + "C" + char$(9) + chr$(9) + "D"
```

See also : cCompress, cExpandTab

Count

Purpose :

Count counts the number of a specified char in a string.

Declare Syntax :

```
Declare Function cCount Lib "t2wlight.dll" (Txt As String, Separator As String) As Integer
```

Call Syntax :

```
test = cCount(Txt, Separator)
```

Where :

Txt	the string to proceed
Separator	the char to be counted
test	the total number of Separator in the string

Comments :

Examples :

```
Txt = "A/BC/DEF/G"
Separator = "/"
test = cCount(Txt, Separator)
        test = 3
```

CountDirectories

Purpose :

CountDirectories counts the total directory in a specified directory.

Declare Syntax :

```
Declare Function cCountDirectories Lib "t2wlight.dll" (ByVal lpFilename As String) As Integer
```

Call Syntax :

```
test = cCountDirectories(lpFilename)
```

Where :

lpFilename	the directory (root or sub-dir)
test	the number of sub-dir founded in the specified directory

Comments :

See also : cCountFiles

CountFiles

Purpose :

CountFiles counts the total files founded in a specified directory.

Declare Syntax :

Declare Function cCountFiles Lib "t2wlight.dll" (ByVal lpFilename As String) As Integer

Call Syntax :

test = cCountFiles(lpFilename)

Where :

lpFilename	the directory (root or sub-dir)
test	the number of files in the specified directory

Comments :

See also : cCountDirectories

CreateAndFill

Purpose :

CreateAndFill creates a string with the specified size and fill it with some chars.

Declare Syntax :

```
Declare Function cCreateAndFill Lib "t2wlight.dll" (ByVal Length As Integer, Txt As String) As String
```

Call Syntax :

```
test = cCreateAndFill(Length, Txt)
```

Where :

Length	the length of the result string
Txt	the chars to fill in the result string
test	the result

Comments :

Examples :

```
Length = 14
Txt = "aBc"
test = cCreateAndFill(Length, Txt)
        test = "aBcaBcaBcaBcaB"
```

See also : cFill

CreateBits

Purpose :

CreateBits creates a string which containes how many bits specified by a number.

Declare Syntax :

Declare Function cCreateBits Lib "t2wlight.dll" (ByVal nBits As Integer) As String

Call Syntax :

test = cCreateBits(nBits)

Where :

nBits	number of bits wished
test	the result

Comments :

Examples :

```
nBits = 10
test = cCreateBits(nBits)
        test will be a size of 2 chars
```

See also : Bit String Manipulation routines

CurrentTime

Purpose :

CurrentTime returns the minutes elapsed since midnight.

Declare Syntax :

```
Declare Function cCurrentTime Lib "t2wlight.dll" () As Integer
```

Call Syntax :

```
test% = cCurrentTime()
```

Where :

test%	the minutes

Comments :

Examples :

```
test% = cCurrentTime()                    -> 1234
```

MKx

Purpose :

MKB, MKC, MKD, MKI, MKL, and MKS return a string containing the IEEE representation of a number. Six separate functions are provided, with one each intended for BYTE, CURRENCY, DOUBLE, INTEGER, LONG, SINGLE.

Declare Syntax :

```
Declare Function cMKB Lib "t2wlight.dll" (ByVal Value As Integer) As String
Declare Function cMKC Lib "t2wlight.dll" (ByVal Value As Currency) As String
Declare Function cMKD Lib "t2wlight.dll" (ByVal Value As Double) As String
Declare Function cMKI Lib "t2wlight.dll" (ByVal Value As Integer) As String
Declare Function cMKL Lib "t2wlight.dll" (ByVal Value As Long) As String
Declare Function cMKS Lib "t2wlight.dll" (ByVal Value As Single) As String

Declare Function cMKN Lib "t2wlight.dll" (ByVal Value As String) As String
```

Call Syntax :

```
Nm$ = cMKB(Value%)
Nm$ = cMKC(Value@)
Nm$ = cMKD(Value#)
Nm$ = cMKI(ValueM)
Nm$ = cMKL(Value&)
Nm$ = cMKS(Value!)
```

Where :

Nm$ receives the IEEE representation of Value?.

Comments :

See also : cCVB, cCVC, cCVD, cCVI, cCVL, cCVS

DaysInMonth

Purpose :

DaysInMonth returns the total days in a month.

Declare Syntax :

```
Declare Function cDaysInMonth Lib "t2wlight.dll" (ByVal nYear As Integer, ByVal nMonth As Integer) As Integer
```

Call Syntax :

```
test = cDaysInMonth(nYear, nMonth)
```

Where :

nYear	is the year with the century
nMonth	is the month

Comments :

Examples :

```
nYear = 1994
nMonth = 12
test = cDaysInMonth(nYear, nMonth)
            test = 31

nYear = 1995
nMonth = 2
test = cDaysInMonth(nYear, nMonth)
            test = 28
```

Decrypt

Purpose :

Decrypt decodes a string encoded with Encrypt function.

Declare Syntax :

Declare Function cDecrypt Lib "t2wlight.dll" (Txt As String, password As String, ByVal level As Integer) As String

Call Syntax :

test = cDecrypt(Txt, password, level)

Where :

Txt	is the string to decrypt
password	is the key to use for decryption
level	level of the encryption
test	is the string decrypted

Comments :

The password/key is case sensitive.
The level is a number between **0** and **4** (Constants and Types declaration).
You must use the same level for encrypt/decrypt a gived string.

Examples :

```
Txt = "Under the blue sky, the sun is yellow"
password = "a new encryption"
level = ENCRYPT_LEVEL_4
test = cEncrypt(Txt, password, level)
        txt = cDecrypt(test, password, level)
```

See also : cEncrypt

DeviationD

Purpose :

DeviationD will calculate the standard deviation from all elements in a Double array.

Declare Syntax :

Declare Function cDeviationD Lib "t2wlight.dll" (array() As Double) As Double

Call Syntax :

deviation = cDeviationD(array())

Where :

array()	is the Double array.
deviation	is the standard deviation calculated. This value is always a Double value.

Comments :

See Also : cDeviationD, cDeviationI, cDeviationL, cDeviationS, Array routines

DeviationI

Purpose :

DeviationI will calculate the standard deviation from all elements in an Integer array.

Declare Syntax :

```
Declare Function cDeviationI Lib "t2wlight.dll" (array() As Integer) As Double
```

Call Syntax :

```
deviation = cDeviationI(array())
```

Where :

array()	is the Integer array.
deviation	is the standard deviation calculated. This value is always a Double value.

Comments :

See Also : cDeviationD, cDeviationI, cDeviationL, cDeviationS, Array routines

DeviationL

Purpose :

DeviationL will calculare the standard deviation from all elements in a Long array.

Declare Syntax :

Declare Function cDeviationL Lib "t2wlight.dll" (array() As Long) As Double

Call Syntax :

deviation = cDeviationL(array())

Where :

array()	is the Long array.
deviation	is the standard deviation calculated. This value is always a Double value.

Comments :

See Also : cDeviationD, cDeviationI, cDeviationL, cDeviationS, Array routines

DeviationS

Purpose :

DeviationS will calculare the standard deviation from all elements in a Single array.

Declare Syntax :

Declare Function cDeviationS Lib "t2wlight.dll" (array() As Single) As Double

Call Syntax :

deviation = cDeviationS(array())

Where :

array()	is the Single array.
deviation	is the standard deviation calculated. This value is always a Double value.

Comments :

See Also : cDeviationD, cDeviationI, cDeviationL, cDeviationS, Array routines

Encrypt

Purpose :

Encrypt encodes a string with a password/key.

Declare Syntax :

Declare Function cEncrypt Lib "t2wlight.dll" (Txt As String, password As String, ByVal level As Integer) As String

Call Syntax :

test = cEncrypt(Txt, password, level)

Where :

Txt	is the string to encrypt
password	is the key to use for encryption
level	level of the encryption
test	is the string decrypted

Comments :

The password/key is case sensitive.
The level is a number between **0** and **4** (Constants and Types declaration).
Higher is the level, better is the encryption
You must use the same level for encrypt/decrypt a gived string.

Examples :

```
Txt = "Under the blue sky, the sun is yellow"
password = "a new encryption"
level = ENCRYPT_LEVEL_4
test = cEncrypt(Txt, password, level)
        txt = cDecrypt(test, password, level)
```

See also : cDecrypt

FileCRC32

Purpose :

FileCRC32 calculates a 32 bits CRC for a gived file.

Declare Syntax :

Declare Function cFileCRC32 Lib "t2wlight.dll" (ByVal lpFilename As String, ByVal mode As Integer) As Long

Call Syntax :

test = cFileCRC32(lpFilename, mode)

Where :

lpFilename	the file to proceed
mode	OPEN_MODE_BINARY (calculates the CRC on the full length of the file). This is the default mode.
	OPEN_MODE_TEXT (calculates the CRC until a EOF is encountered)
test	the calculated CRC 32 bits in a LONG.

Comments :

The returned value can be negative and have only a value :

-1 If the filename is not a good filename or if the filename not exist or if an error occurs when accessing the filename.

Examples :

test = cFileCRC32("C:\COMMAND.COM") &h1131ADD3 (MS-DOS 6.22)

See also : cStringCRC32, Constants and Types declaration

FileDrive

Purpose :

FileDrive extracts the drive on which the file is present.

Declare Syntax :

```
Declare Function cFileDrive Lib "t2wlight.dll" (ByVal lpFilename As String) As String
```

Call Syntax :

```
test$ = cFileDrive(lpFilename)
```

Where :

lpFilename	the file to proceed
test$	EMPTY is the file not exist or an error occurs when accessing the file DRIVE LETTER for the file

Comments :

FileLineCount

Purpose :

FileLineCount counts the total number of lines in an ASCII file.

Declare Syntax :

```
Declare Function cFileLineCount Lib "t2wlight.dll" (ByVal lpFilename As String) As Long
```

Call Syntax :

```
test& = cFileLineCount(lpFilename$)
```

Where :

lpFilename$	is the name of the file.
test&	is the total number of lines.

Comments :

Each line is determined only if a CR is ending the line.

The returned value can be negative and have the following value :

-1	error opening file (not exist, not a valid filename).
-2	error reading file.
-3	error when allocating memory buffer.

Examples :

```
test& = cFileLineCount("c:\autoexec.bat")
```

On my system :

```
test& =
```

See also :

FilePathExists

Purpose :

FilePathExists verifies if the specified file is present.

Declare Syntax :

Declare Function cFilePathExists Lib "t2wlight.dll" (ByVal lpFilename As String) As Integer

Call Syntax :

test% = cFilePathExists(lpFilename)

Where :

lpFilename	the file to proceed
test%	TRUE is the file exists
	<> TRUE if the file not exists or if an error occurs when accessing the file.

Comments :

CVx

Purpose :

CVB, CVC, CVD, CVI, CVL and CVS returns number in a certain precision given a string containing the IEEE representation of the number. Six separate functions are provided, with one each intended for BYTE, CURRENCY, DOUBLE, INTEGER, LONG and SINGLE.

Declare Syntax :

```
Declare Function cCVB Lib "t2wlight.dll" (Value As String) As Integer
Declare Function cCVC Lib "t2wlight.dll" (Value As String) As Currency
Declare Function cCVD Lib "t2wlight.dll" (Value As String) As Double
Declare Function cCVI Lib "t2wlight.dll" (Value As String) As Integer
Declare Function cCVL Lib "t2wlight.dll" (Value As String) As Long
Declare Function cCVS Lib "t2wlight.dll" (Value As String) As Single
```

Call Syntax :

```
test% = cCVB(Value$)
test@ = cCVC(Value$)
test# = cCVD(Value$)
test% = cCVI(Value$)
test& = cCVL(Value$)
test! = cCVS(Value$)
```

Where :

test? receives the value represented by the IEEE string held in Value$

Comments :

See also : cMKB, cMKC, cMKD, cMKI, cMKL, cMKS

GetDiskFree, GetDiskSpace, GetDiskUsed, GetDiskClusterSize

Purpose :

GetDiskFree, GetDiskSpace, GetDiskUsed and GetDiskClusterSize retrieves respectively the free disk space, the size of the disk, the part of the disk used and the size of a cluster on a specified disk (hard disk or floppy disk).

Declare Syntax :

```
Declare Function cGetDiskFree Lib "t2wlight.dll" (ByVal lpDrive As String) As Long
Declare Function cGetDiskSpace Lib "t2wlight.dll" (ByVal lpDrive As String) As Long
Declare Function cGetDiskUsed Lib "t2wlight.dll" (ByVal lpDrive As String) As Long
Declare Function cGetDiskClusterSize Lib "t2wlight.dll" (ByVal lpDrive As String) As Long
```

Call Syntax :

```
test& = cGetDiskFree(lpDrive)
test& = cGetDiskSpace(lpDrive)
test& = cGetDiskUsed(lpDrive)
test& = cGetDiskClusterSize(lpDrive)
```

Where :

lpDrive	is the letter for the disk
test&	is the result.

Comments :

If the disk is not present or if the disk is not available or if an error occurs when accessing the disk, the returned value is always -1.
This function works with local disk (hard, floppy or cd-rom) als well on remote disk (network).

Examples :

```
test& = cGetDiskFree("C")          -> 268197888
test& = cGetDiskSpace("C")         -> 527654912
test& = cGetDiskUsed("C")-> 259457024
test& = cGetDiskClusterSize("C")   -> 8192
```

See also : cFileSize, cFilesSize, cFilesSizeOnDisk, cFilesSlack

FilesInDirectory

Purpose :

FilesInDirectory retrieves each file in the specified directory.

Declare Syntax :

```
Declare Function cFilesInDirectory Lib "t2wlight.dll" (ByVal nFilename As String, ByVal firstnext As Integer) As String
```

Call Syntax :

```
test$ = cFilesInDirectory(nFilename, firstnext )
```

Where :

nFilename	the directoty to proceed with the file mask (*.* for all)
firstnext	TRUE for the first file FALSE for each next file
test$	the returned file

Comments :

Examples :

```
    Dim i           As Integer
    Dim Tmp         As String

    i = 0
    Tmp = cFilesInDirectory("c:\*.*", True)

    Debug.Print "The first 7 files in C:\ are : "

    Do While (Len(Tmp) > 0)
        Debug.Print Tmp
        Tmp = cFilesInDirectory("c:\*.*", False)
        i = i + 1
        If (i >= 7) Then Exit Do
    Loop
```

On my system:

```
The first 7 files in C:\ are :

863DATA
WINA20.386
AUTOEXEC.BAT
COMMAND.COM
IMAGE.DAT
BOOTSECT.DOS
ACD.IDX
```

See also : cSubDirectory

FileSize

Purpose :

FileSize returns the size of the specified file.

Declare Syntax :

```
Declare Function cFileSize Lib "t2wlight.dll" (ByVal lpFilename As String) As Long
```

Call Syntax :

```
test& = cFileSize(lpFilename)
```

Where :

lpFilename	the file to proceed
test&	the size of the file

Comments :

If the file is not present or if an error occurs when accessing the file, the return value is 0

See also : cFilesSize, cFilesSizeOnDisk, cFilesSlack

FilesSize

Purpose :

FilesSize returns the logical size of all files specified by file mask.
FilesSizeOnDisk returns the physical size of all files specified by file mask.
FilesSlack returns in one call, the slack from all files specified by file mask, the logical size and the physical size..

Declare Syntax :

```
Declare Function cFilesSize Lib "t2wlight.dll" (ByVal lpFilename As String) As Long
Declare Function cFilesSizeOnDisk Lib "t2wlight.dll" (ByVal nFileName As String) As Long
Declare Function cFilesSlack Lib "t2wlight.dll" (ByVal nFileName As String, Size1 As Long, Size2 As Long) As Integer
```

Call Syntax :

```
test& = cFilesSize(nFilename)
test& = cFilesSizeOnDisk(nFilename)
test% = cFilesSlack(nFilename, Size1, Size2)
```

Where :

nFilename	is the mask file to proceed.
test&	is the size of all files founden with the file mask.
test%	is the slack for all files fouden with the file mask.
Size1	is the logical size of all files fouden with the file mask.
Size2	is the physical size of all files fouden with the file mask.

Comments :

If the mask is invalid or if the file not exists or if an error occurs when accessing the file, the return value is 0
The slack of a file or files by file mask is the % of all spaces not used on all last clusters.

Examples :

```
test& = cFilesSize("*.*")              on my system, 5607689 bytes
test& = cFilesSizeOnDisk("*.*")        on my system, 5890048 bytes
test% = cFilesSlack("*.*", 0, 0)       on my system, 4 %
```

See also : cFileSize, cGetDiskClusterSize

IsFileX

Purpose :

The routines checks if a specified file has or not the specified attribute.
IsFilenameValid checks if the filename follows the DOS syntax for a file.
FileGetAttrib retrieves in a Call, all attributes of a gived file.

Declare Syntax :

```
Declare Function cIsFileArchive Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cIsFileHidden Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cIsFilenameValid Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cIsFileNormal Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cIsFileReadOnly Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cIsFileSubDir Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cIsFileSystem Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cIsFileVolId Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cIsFileFlag Lib "t2wlight.dll" (ByVal nFilename As String, ByVal nStatus As Integer) As Integer

Declare Function cFileGetAttrib Lib "t2wlight.dll" (ByVal nFilename As String, nFileAttribute As Any) As Integer
```

Call Syntax :

```
test% = cIsFileArchive(nFilename)
test% = cIsFileHidden(nFilename)
test% = cIsFilenameValid(nFilename)
test% = cIsFileNormal(nFilename)
test% = cIsFileReadOnly(nFilename)
test% =cIsFileSubDir(nFilename)
test% = cIsFileSystem(nFilename)
test% = cIsFileVolId(nFilename)
test% = cIsFileFlag(nFilename, nStatus)

test% = cFileGetAttrib(nFilename, nFileAttribute)
```

Where :

nFilename	the filename to check
nStatus	the status to check (only for cIsFileFlag) combine A_NORMAL, A_RDONLY, A_HIDDEN, A_SYSTEM, A_VOLID, A_SUBDIR, A_ARCH with logical OR.
nFileAttribute	the type variable 'FileAttributeType' (only for cFileGetAttrib)
test	TRUE if the specified flag is present FALSE if the specified flag is not present

Comments :

IsFilenameValid checks only the validity of a file (normal file or network file) not the presence on a disk, the returned code can be :

IFV_ERROR	bad char in the filename
IFV_NAME_TOO_LONG	the length of the file part is too long (> 8)
IFV_EXT_TOO_LONG	the length of the extension part is too long (> 3)
IFV_TOO_MANY_BACKSLASH	too many successing backslash (> 2)
IFV_BAD_DRIVE_LETTER	bad drive letter before the colon ':'
IFV_BAD_COLON_POS	bad colon ':' position (<>2)
IFV_EXT_WITHOUT_NAME	extension without a name

If the filename is not a good filename or if the filename not exist or if an error occurs when accessing the filename, the return value is always FALSE.

See also : IsX Family Test routines, Constants and Types declaration

FillD

Purpose :

FillD fills, with an automatic incremented value, all of the elements of a Double array.

Declare Syntax :

Declare Function cFillD Lib "t2wlight.dll" (array() As Double, ByVal nValue As Double) As Integer

Call Syntax :

status = cFillD(array(), nValue)

Where :

array()	is the Double array.
nValue	is the Double value automatiCally incremented by one.
status	is always TRUE.

Comments :

See Also : cFillD, cFillI, cFillL, cFillS, Array routines

FillI

Purpose :

FillI fills, with an automatic incremented value, all of the elements of an Integer array.

Declare Syntax :

Declare Function cFillI Lib "t2wlight.dll" (array() As Integer, ByVal nValue As Integer) As Integer

Call Syntax :

status = cFillI(array(), nValue)

Where :

array()	is the Integer array.
nValue	is the Integer value automatiCally incremented by one.
status	is always TRUE.

Comments :

See Also : cFillD, cFillI, cFillL, cFillS, Array routines

FillL

Purpose :

FillL fills, with an automatic incremented value, all of the elements of a Long array.

Declare Syntax :

Declare Function cFillL Lib "t2wlight.dll" (array() As Long, ByVal nValue As Long) As Integer

Call Syntax :

status = cFillL(array(), nValue)

Where :

array()	is the Long array.
nValue	is the Long value automatiCally incremented by one.
status	is always TRUE.

Comments :

See Also : cFillD, cFillI, cFillL, cFillS, Array routines

FillS

Purpose :

FillS fills, with an automatic incremented value, all of the elements of a Single array.

Declare Syntax :

Declare Function cFillS Lib "t2wlight.dll" (array() As Single, ByVal nValue As Single) As Integer

Call Syntax :

status = cFillS(array(), nValue)

Where :

array()	is the Single array.
nValue	is the Single value automatiCally incremented by one.
status	is always TRUE.

Comments :

See Also : cFillD, cFillI, cFillL, cFillS, Array routines

Conversion table for Hundreds

The table below show the international table conversion between minutes and hundreds.
Don't forget that some hundreds are rounded.

Minutes	Hundreds	true value		Minutes	Hundreds	true value
0	**00**	0	|	30	**50**	50
1	**02**	1,666667	|	31	**52**	51,666667
2	**03**	3,333333	|	32	**53**	53,333333
3	**05**	5	|	33	**55**	55
4	**07**	6,666667	|	34	**57**	56,666667
5	**08**	8,333333	|	35	**58**	58,333333
6	**10**	10	|	36	**60**	60
7	**12**	11,66667	|	37	**62**	61,66667
8	**13**	13,33333	|	38	**63**	63,33333
9	**15**	15	|	39	**65**	65
10	**17**	16,66667	|	40	**67**	66,66667
11	**18**	18,33333	|	41	**68**	68,33333
12	**20**	20	|	42	**70**	70
13	**22**	21,66667	|	43	**72**	71,66667
14	**23**	23,33333	|	44	**73**	73,33333
15	**25**	25	|	45	**75**	75
16	**27**	26,66667	|	46	**77**	76,66667
17	**28**	28,33333	|	47	**78**	78,33333
18	**30**	30	|	48	**80**	80
19	**32**	31,66667	|	49	**82**	81,66667
20	**33**	33,33333	|	50	**83**	83,33333
21	**35**	35	|	51	**85**	85
22	**37**	36,66667	|	52	**87**	86,66667
23	**38**	38,33333	|	53	**88**	88,33333
24	**40**	40	|	54	**90**	90
25	**42**	41,66667	|	55	**92**	91,66667
26	**43**	43,33333	|	56	**93**	93,33333
27	**45**	45	|	57	**95**	95
28	**47**	46,66667	|	58	**97**	96,66667
29	**48**	48,33333	|	59	**98**	98,33333

Note : you can see if you've a good look in this table that some difference between two minutes are "better" than others if converted in hundreds. This is due to the rounding value.

if I works from 12 to 16 minutes (4 minutes), I've worked (27 - 20) = 7 hundreds
if I works from 16 to 20 minutes (4 minutes), I've worked (33 - 27) = 6 hundreds

In the two cases, I've worked 4 minutes but in the first case, I receive 7 hundreds and in the second case, I receive only 6 hundreds.

TypeX

Purpose :

TypesCompare compares two Types variable.
CompareTypeString compares a Type to a String.
CompareStringType compares a String to a Type.

TypeClear clears a Type variable.
TypeMid extracts information from a Type variable.

TypesCopy copies a Type variable into a variable.
TypeTransfert transfers a Type variable into a String.

StringToType copies a String to a Type variable.
TypeToString copies a Type variable to a String.

Declare Syntax :

```
Declare Function cTypesCompare Lib "t2wlight.dll" (Type1 As Any, Type2 As Any, ByVal lenType1 As Integer) As Integer
Declare Function cCompareTypeString Lib "t2wlight.dll" Alias "cTypesCompare" (TypeSrc As Any, ByVal Dst As String, ByVal lenTypeSrc As Integer) As Integer
Declare Function cCompareStringType Lib "t2wlight.dll" Alias "cTypesCompare" (ByVal Src As String, TypeDst As Any, ByVal lenTypeSrc As Integer) As Integer

Declare Sub cTypeClear Lib "t2wlight.dll" (TypeSrc As Any, ByVal lenTypeSrc As Integer)
Declare Function cTypeMid Lib "t2wlight.dll" (TypeSrc As Any, ByVal Offset As Integer, ByVal Length As Integer) As String

Declare Sub cTypesCopy Lib "t2wlight.dll" (TypeSrc As Any, TypeDst As Any, ByVal lenTypeSrc As Integer)
Declare Function cTypeTransfert Lib "t2wlight.dll" (TypeSrc As Any, ByVal lenTypeSrc As Integer) As String

Declare Sub cStringToType Lib "t2wlight.dll" Alias "cTypesCopy" (ByVal Src As String, TypeDst As Any, ByVal lenTypeSrc As Integer)
Declare Sub cTypeToString Lib "t2wlight.dll" Alias "cTypesCopy" (TypeSrc As Any, ByVal Dst As String, ByVal lenTypeSrc As Integer)
```

Call Syntax :

```
test% = cTypesCompare(Type1, Type2, len(Type1))
test% = cCompareTypeString(TypeSrc, Dst, len(TypeSrc))
test% = cCompareStringType(Src, TypeDst, len(TypeDst))

Call cTypeClear(TypeSrc, len(TypeSrc)
test$ = cTypeMid(TypeSrc, Offset, Length)

Call cTypesCopy(TypeSrc, TypeDst, len(TypeSrc))
test$ = cTypeTransfert(TypeSrc, len(TypeSrc)

Call cStringToType(Src, TypeDst, len(TypeDst))
Call cTypeToString(TypeSrc, Dst, len(TypeSrc))
```

Where :

Type1, Type2, TypeSrc, TypeDst	the Type variable
Src, Dst,	the String variable
Offset	the offset in the Type variable
Length	the length in the Type variable
test%	TRUE if the variables to compare are the same

	FALSE if the variables to compare are not the same
test$	the result

Comments :

Only Type variable mixed with INTEGER, LONG, SINGLE, DOUBLE, CURRENCY and FIXED STRING can be used.

When   you compare 2 types variables or 1 type variable and 1 string, the size of each variable must be same.
When you copy 1 Type variable into a string or a string into Type variable, the size of each variable must be same.

Examples :

See also :

FindBitReset

Purpose :

FindBitReset finds the first bit Reset starting at the position gived for a a gived string.

Declare Syntax :

Declare Function cFindBitReset Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer) As Integer

Call Syntax :

test = cFindBitReset(Txt, Position)

Where :

Txt	the string to proceed
Position	the starting position
test	TRUE if no bit founded
	<> TRUE if a bit founded

Comments :

This function is useful to find or scan a string for the bit Reset. The first bit in the string to start the test is -1.

See also : Bit String Manipulation routines

FindBitSet

Purpose :

FindBitSet finds the first bit Set starting at the position gived for a a gived string.

Declare Syntax :

Declare Function cFindBitSet Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer) As Integer

Call Syntax :

test = cFindBitSet(Txt, Position)

Where :

Txt	the string to proceed
Position	the starting position
test	TRUE if no bit founded
	<> TRUE if a bit founded

Comments :

This function is useful to find or scan a string for the bit Set. The first bit in the string to start the test is -1.

See also : Bit String Manipulation routines

FindFileInEnv

Purpose :

FindFileInEnv searches if a specified file is present is the specified environment variable.

Declare Syntax :

Declare Function cFindFileInEnv Lib "t2wlight.dll" (ByVal lpFilename As String, ByVal lpEnv As String) As Integer

Call Syntax :

test% = cFindFileInEnv(lpFilename, lpEnv)

Where :

lpFilename	name of file to search for
lpEnv	environment to search
test%	TRUE if founded
	FALSE if not founded

Comments :

This function searches for the target file in the specified domain. The lpEnv variable can be any environment variable that specifies a list of directory paths, such as PATH, LIB, INCLUDE, or other user-defined variables. This function function is case-sensitive, so the lpEnv variable should match the case of the environment variable.
The routine first searches for the file in the current working directory. If it doesn't find the file, it next looks through the directories specified by the environment variable.

Examples :

test% = cFileFileInEnv("win.com", "windir")   -> TRUE

See also : cFindFileInPath

FindFileInPath

Purpose :

FindFileInPath searches if a specified file is present is the path.

Declare Syntax :

Declare Function cFindFileInPath Lib "t2wlight.dll" (ByVal lpFilename As String) As Integer

Call Syntax :

test% = cFindFileInPath(lpFilename)

Where :

lpFilename	name of file to search for
test%	TRUE if founded
	FALSE if not founded

Comments :

This function searches for the target file in the PATH environment variable that specifies a list of directory paths.
The routine first searches for the file in the current working directory. If it doesn't find the file, it next looks through the all directories specified in the PATH environment variable.
This function is a subset of cFindFileInEnv : cFileFileInEnv(lpFilename, "PATH")

Examples :

test% = cFileFileInPath("xcopy.exe"")   -> TRUE

See also : cFindFileInEnv

FromBinary, FromBinary2, ToBinary, ToBinary2

Purpose :

FromBinary converts a binary string (0, 1) to a string
FromBinary2 converts a binary string (custom letters) to a string

ToBinary converts a string to a binary representation with 0, 1
ToBinary2 converts a string to a binary representation with two custom letters for 0, 1representation

Declare Syntax :

```
Declare Function cFromBinary Lib "t2wlight.dll" (Text As String) As String
Declare Function cFromBinary2 Lib "t2wlight.dll" (Text As String, Bin As String) As String

Declare Function cToBinary Lib "t2wlight.dll" (Text As String) As String
Declare Function cToBinary2 Lib "t2wlight.dll" (Text As String, Bin As String) As String
```

Call Syntax :

```
test$ = cFromBinary(Text)
test$ = cFromBinary2(Text, Bin)

test$ = cToBinary(Text)
test$ = cToBinary2(Text, Bin)
```

Where :

Text	the string to proceed
Bin	the two custom letters for 0, 1 representation
test$	the result

Comments :

Examples :

```
test$ = cToBinary("MC")                          -> "0100110101000011"
test$ = cToBinary2("MC","mc")                    -> "cmccmmcmcmccccmm"

test$ = cFromBinary("0100110101000011")          -> "MC"
test$ = cFromBinary2("cmccmmcmcmccccmm","mc")  -> "MC"
```

See also : cFromHexa, cToHexa

FromHexa, ToHexa

Purpose :

ToHexa converts a ascii string to hexa string.
FromHexa converts a hexa string to an ascii string.

Declare Syntax :

```
Declare Function cFromHexa Lib "t2wlight.dll" (Text As String) As String
Declare Function cToHexa Lib "t2wlight.dll" (Text As String) As String
```

Call Syntax :

```
test$ = cFromHexa(Text)
test$ = cToHexa(Text)
```

Where :

Text	the string to proceed
test$	the result

Comments :

The returned string from ToHexa is always a multiple of 2
If the size of the string passed to FromHexa is not a multiple of 2, only n-1 chars are used

Examples :

```
test$ = cToHexa("ABCDEFG")                  -> "41424344454647"
test$ = cFromHexa("47464544434241")         -> "GFEDCBA"
```

See also : cFromBinary, cToBinary

Get, GetBlock, GetIn, GetInPart, GetInPartR, GetInR, TokenIn

Purpose :

Get extratcs a sub-string delimited by **'|'** in a gived string.
GetBlock reads a block of n chars starting at a gived block in a gived string.
GetIn extracts a left sub-string delimited by a separator in a gived string.
GetInPart extracts the first left sub-string or the rest after the first sub-string delimited by a separator in a gived string.
GetInPartR extracts the first right sub-string or the rest after the first sub-string delimited by a separator in a gived string.
GetInR extracts a right sub-string delimited by a separator in a gived string.
TokenIn extracts a sub-string delimited by a separator's list in a gived string.

Declare Syntax :

```
Declare Function cGet Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer) As String
Declare Function cGetBlock Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer, ByVal Length As Integer) As String
Declare Function cGetIn Lib "t2wlight.dll" (Txt As String, Separator As String, ByVal Position As Integer) As String
Declare Function cGetInPart Lib "t2wlight.dll" (Txt As String, Separator As String, ByVal Position As Integer) As String
Declare Function cGetInPartR Lib "t2wlight.dll" (Txt As String, Separator As String, ByVal Position As Integer) As String
Declare Function cGetInR Lib "t2wlight.dll" (Txt As String, Separator As String, ByVal Position As Integer) As String
Declare Function cTokenIn Lib "t2wlight.dll" (Txt As String, Separator As String, ByVal Position As Integer) As String
```

Call Syntax :

```
test$ = cGet(Txt, Position)
test$ = cGetBlock(Txt, Position, Length)
test$ = cGetIn(Txt, Separator, Position)
test$ = cGetInPart(Txt, Separator, Position)
test$ = cGetInPartR(Txt, Separator, Position)
test$ = cGetInR(Txt, Separator, Position)
test$ = cTokenIn(Txt, SeparatorList, Position)
```

Where :

Txt	the string to proceed.
Position	the position of the sub-string or the block.
Length	the length of each block.
Separator	the delimitor for each sub-string.
SeparatorList	the separator's list for each sub-string.
test$	the result.

Comments :

•If the size of the string is 0 or if the position is < 1 or greater than the maximum block is the string or if the length is 0. The returned string is an empty string.
•The function cGet is a subset of the cGetIn function.
•The function cGetBlock is similar to MID$(Txt, 1+ ((n-1) * m), m)
•The function cTokenIn is a superset of the cGetIn function, in the fact that you can pass a separator's list.
•For the function cGetInPart, cGetInPartR, you must set Position to TRUE for first part (left or right) and to FALSE for second part (left or right).
•The function cGetInPartR is very usefull when you must isolate a file extension or the full directory and the filename function.

Examples :

```
test$ = cGet("A|BC|DEF|G", 1)                    -> "A"
test$ = cGet("A|BC|DEF|G", 3)                    -> "DEF"
```

```
test$ = cGetIn("A/BC/DEF/G", "/", 4)                        -> "G"
test$ = cGetIn("A/BC/DEF/G","D", 2)                                 -> "EF/G"

test$ = cGetInR("A/BC/DEF/G", "/", 4)                       -> "A"
test$ = cGetInR("A/BC/DEF/G","D", 2)                        -> "A/BC/"

test$ = cGetInPart("A/BC/DEF/G", "/", True)                 -> "A"
test$ = cGetInPart("A/BC/DEF/G","/", False)                 -> "BC/DEF/G"

test$ = cGetInPartR("c:\vberr.hnd\test.mak", ".", True)     -> "mak"
test$ = cGetInPartR("c:\vberr.hnd\test.mak", ".", False)    -> "c:\vberr.hnd\test"

test$ = cGetBlock("A/BC/DEF/G",1,2)                         -> "A/"
test$ = cGetBlock("A/BC/DEF/G",4,2)                         -> "EF"

test$ = cTokenIn("A/BC:DEF\G", "/:\", 4)                    -> "G"
test$ = cTokenIn("A/BC:DEF\G", "/:\", 3)                    -> "DEF"
```

See also : cSetDefaultSeparator, cInsertBlocks, cInsertBlockBy, cInsertByMask, cInsertChars

GetBit

Purpose :

GetBit returns if a gived bit in a gived string if Set or Reset.

Declare Syntax :

Declare Function cGetBit Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer) As Integer

Call Syntax :

test = cGetBit(Txt, Position)

Where :

Txt	the string to proceed
Position	the bit position
test	TRUE if the bit is Set
	FALSE if the bit is Reset

Comments :

The first bit in the string is the bit 0.

See also : Bit String Manipulation routines

IsFormEnabled

Purpose :

IsFormEnabled checks if the specified form is enabled or not.

Declare Syntax :

Declare Function cIsFormEnabled Lib "t2wlight.dll" (ByVal hWnd As Integer) As Integer

Call Syntax :

test% = cIsFormEnabled(hWnd)

Where :

hWnd	is the .hWnd of the specified form.
test%	TRUE if the form is enabled. FALSE is the form is disabled.

Comments :

If you disable a form with the cDisableForm or cDisableFI and if you display a MODAL form, you must take care that Windows reenables the disabled form.

Examples :

test% = cIsFormEnabled(Me.hWnd)

See also : cDisableForm, cEnableForm, cDisableFI, cEnableFI

GetChangeTaskName

Purpose :

GetChangeTaskName gets and changes the name of the task. You see change in the Task Manager by pressing the CTRL + ESC keys.

Declare Syntax :

Declare Function cGetChangeTaskName Lib "t2wlight.dll" (ByVal hWnd As Integer, ByVal Text As String) As String

Call Syntax :

test$ = cGetChangeTaskName(Form.hWnd, Text)

Where :

Form.hWnd	is the hWnd of your application
Text	is the new task name to given at your application
test$	is the old task name of the application

Comments :

This is useful to set a particular task name at your application and backups the old task name.
This function is a mix of cGetTaskName and cChangeTaskName.

Examples :

```
    Dim OldTaskName         As String

    OldTaskName = cGetChangeTaskName(Me.hWnd, "Hello world")
    MsgBox OldTaskName
            -> press the CTRL + ESC keys to see the change in the Task Manager
            OldTaskName is "Microsoft Visual Basic"

if you repeat the test
            OldTaskName is "Hello world"
```

See also : cChangeTaskName, cGetTaskName

FullPath

Purpose :

FullPath converts a partial path stored in path to a fully qualified path.

Declare Syntax :

```
Declare Function cFullPath Lib "t2wlight.dll" (ByVal nFilename As String) As String
```

Call Syntax :

```
test$ = cFullPath(nFilename)
```

Where :

nFilename	is the partial path.
test$	is the returned full qualified path.

Comments :

If the file is not available or if an error occurs when accessing the file, the returned path is always an EMPTY string.

Examples :

```
tmp$ = cFilesInDirectory(cGetDefaultCurrentDir() + "\*.*", True) 'retrieves the first file in the default current directory
test$ = cFullPath(tmp$)
```

On my system :

```
        tmp$ = "AWARE.BAS"
        test$ = "M:\VB\AWARE.BAS"
```

See also : cSplitPath, cMakePath

SetCtlX

Purpose :

The functions below applies to a custom control.

SetCtlCaption sets the .Caption property of the control.
SetCtlDataField sets the .DataField property of the control.
SetCtlFocus gives the Focus to a control.
SetCtlPropString sets the specified property (founded with cGetCtlPropString function) of the control.
SetCtlTag sets the .Tag property of the control.
SetCtlText sets the .Text property of the control.

Declare Syntax :

```
Declare Sub cSetCtlCaption Lib "t2wlight.dll" (Ctl As Control, ByVal Text As String)
Declare Sub cSetCtlDataField Lib "t2wlight.dll" (Ctl As Control, ByVal Text As String)
Declare Sub cSetCtlFocus Lib "t2wlight.dll" (Ctl As Control)
Declare Sub cSetCtlPropString Lib "t2wlight.dll" (Ctl As Control, ByVal PropIndex As Integer, ByVal Text As String)
Declare Sub cSetCtlTag Lib "t2wlight.dll" (Ctl As Control, ByVal Text As String)
Declare Sub cSetCtlText Lib "t2wlight.dll" (Ctl As Control, ByVal Text As String)
```

Call Syntax :

The purpose and the declare syntax are very explicite.

Where :

Ctl	the name of the control to proceed

Comments :

•The advantage to use these routines is that these routines doesn't generates an error if the property not exists.

Examples :

See also : cSetX, cGetX, cGetCtlX

GetCurrentDrive

Purpose :

GetCurrentDrive returns the current default drive.

Declare Syntax :

Declare Function cGetCurrentDrive Lib "t2wlight.dll" () As String

Call Syntax :

test$ = cGetCurrentDrive()

Where :

test$	the drive in a letter

Comments :

Examples :

See also : cGetDefaultCurrentDir

GetDefaultCurrentDir

Purpose :

GetDefaultCurrentDir retrieves the current dir on the current drive.

Declare Syntax :

Declare Function cGetDefaultCurrentDir Lib "t2wlight.dll" () As String

Call Syntax :

test$ = cGetDefaultCurrentDir()

Where :

test$	the dir

Comments :

The GetDefaultCurrentDir function gets the full path of the current working directory for the default drive . The integer The GetDefaultCurrentDir function returns a string that represents the path of the current working directory. If the current working directory is set to the root, the string will end with a backslash (\ ). If the current working directory is set to a directory other than the root, the string will end with the name of the directory and not with a backslash.

Examples :

See also : cGetDriveCurrentDir, cGetCurrentDrive

GetDefaultPrinter

Purpose :

GetDefaultPrinter returns the default printer in the [windows] section of Win.INI

Declare Syntax :

Declare Function cGetDefaultPrinter Lib "t2wlight.dll" () As String

Call Syntax :

test$ = cGetDefaultPrinter()

Where :

test$	is the default printer

Comments :

Examples :

```
test$ = cGetDefaultPrinter()          -> "HP LASERJET III,HPPCL5MS,LPT1:"
```

See also : cGetPrinterPorts

GetDevices

Purpose :

GetDevices returns all devices founden in the [devices] section in the Win.INI

Declare Syntax :

Declare Function cGetDevices Lib "t2wlight.dll" () As String

Call Syntax :

test$ = cGetDevices()

Where :

test$	all devices separated by a chr$(13).

Comments :

Use the cGetIn function to extract each device.

Examples :

```
test$ = cGetDevices()                    -> "HP LaserJet III=HPPCL5MS,LPT1:"
```

See also : cGetDefaultPrinter

GetDriveCurrentDir

Purpose :

GetDriveCurrentDir retrieves the current dir on the specified drive.

Declare Syntax :

Declare Function cGetDriveCurrentDir Lib "t2wlight.dll" (ByVal lpDrive As String) As String

Call Syntax :

test$ = cGetDefaultCurrentDir(lpDrive)

Where :

lpDrive	the letter for the drive
test$	the dir

Comments :

The GetDriveCurrentDir function gets the full path of the current working directory on the specified drive
The GetDriveCurrentDir function returns a string that represents the path of the current working directory on the specified drive. If the current working directory is set to the root, the string will end with a backslash (\). If the current working directory is set to a directory other than the root, the string will end with the name of the directory and not with a backslash.
If the disk is not present or if the disk is not available or if an error occurs when accessing the disk, the returned value is always an EMPTY string.
This function works with local disk (hard, floppy or cd-rom) als well on remote disk (network).

Examples :

See also : cGetDefaultCurrentDir, cGetCurrentDrive

GetDriveType

Purpose :

GetDriveType determines whether a disk drive is removable, fixed, or remote.

Declare Syntax :

```
Declare Function cGetDriveType Lib "t2wlight.dll" (ByVal lpDrive As String) As Integer
```

Call Syntax :

```
test% = cGetDriveType(lpDrive$)
```

Where :

lpDrive$	is the letter disk to proceed
test%	is the returned drive type

Comments :

The returned value can be :

DRIVE_UNKNOW (drive type can't be founded, drive not present or unknow)
DRIVE_REMOVABLE (disk can be removed from the drive)
DRIVE_FIXED (disk cannot be removed from the drive)
DRIVE_REMOTE (drive is a remote, or network, drive)
DRIVE_CDROM (drive is a cd-rom)

Examples :

On my system :

```
test% = cGetDriveType("A")          -> DRIVE_REMOVABLE
test% = cGetDriveType("C")          -> DRIVE_FIXED
test% = cGetDriveType("X")          -> DRIVE_CDROM
test% = cGetDriveType("Z")          -> DRIVE_REMOTE
```

See also : Constants and Types declaration

GetFullNameInEnv

Purpose :

Declare Syntax :

Call Syntax :

Where :

Comments :

GetFullNameInPath

Purpose :

Declare Syntax :

Call Syntax :

Where :

Comments :

SetX

Purpose :

The functions below applies to the .hWnd of a custom control.

SetCaption sets the .Caption property of the control.
SetDataField sets the .DataField property of the control.
SetFocus gives the Focus to a control.
SetTag sets the .Tag property of the control.
SetText sets the .Text property of the control.

Declare Syntax :

```
Declare Sub cSetCaption Lib "t2wlight.dll" (ByVal hWnd As Integer, ByVal Text As String)
Declare Sub cSetDataField Lib "t2wlight.dll" (ByVal hWnd As Integer, ByVal Text As String)
Declare Sub cSetFocus Lib "t2wlight.dll" (ByVal hWnd As Integer)
Declare Sub cSetTag Lib "t2wlight.dll" (ByVal hWnd As Integer, ByVal Text As String)
Declare Sub cSetText Lib "t2wlight.dll" (ByVal hWnd As Integer, ByVal Text As String)
```

Call Syntax :

The purpose and the declare syntax are very explicite.

Where :

hWnd	the hWnd of the custom control.

Comments :

•The advantage to use these routines is that these routines doesn't generates an error if the property not exists.
•If the custom control doesn't have a .hWnd (Label control b.e.), you must use the cSetCtlX function.

Examples :

See also : cSetCtlX, cGetX, cGetCtlX

GetIni

Purpose :

see Comments

Declare Syntax :

Declare Function cGetIni Lib "t2wlight.dll" (ByVal AppName As String, ByVal szItem As String, ByVal szDefault As String, ByVal InitFile As String) As String

Call Syntax :

test$ = cGetIni(AppName, szItem, szDefault, InitFile)

Where :

AppName	a string that specifies the section containing the entry.
szItem	a string containing the entry whose associated string is to be retrieved.
szDefault	a string that specifies the default value for the given entry if the entry cannot be found in the initialization file.
InitFile	a filename. If this parameter does not contain a full path, Windows searches for the file in the Windows directory.

Comments :

The function searches the file for an entry that matches the name specified by the szItem parameter under the section heading specified by the AppName parameter. If the entry is found, its corresponding string is returned. If the entry does not exist, the default character string specified by the szDefault parameter is copied. A string entry in the initialization file must have the following form:

```
[section]
entry=string
```

Examples :

test$ = cGetIni("Desktop","IconTitleFaceName","MS Sans Serif","WIN.INI")

See also : cPutIni

GetNetConnection

Purpose :

The GetNetConnection function returns the name of the network resource associated with the specified redirected local device.

Declare Syntax :

Declare Function cGetNetConnection Lib "t2wlight.dll" (ByVal lpDrive As String, ErrCode As Integer) As String

Call Syntax :

test$ = cGetNetConnection(lpDrive, ErrCode)

Where :

lpDrive	a string specifying the name of the redirected local device.
ErrCode	TRUE is all is ok
	<> TRUE if an error has occured
test$	the returned name of the remote network resource.

Comments :

FileReset

Purpose :

FileResetAllAttrib, FileResetArchive, FileResetHidden, FileResetReadOnly, FileResetSystem, FileResetFlag resets respectively all attributes, archive attribute, hidden attribute, read-only attribute, system attribute, specified attribute for the gived file.

Declare Syntax :

```
Declare Function cFileResetAllAttrib Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cFileResetArchive Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cFileResetHidden Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cFileResetReadOnly Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cFileResetSystem Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cFileResetFlag Lib "t2wlight.dll" (ByVal nFilename As String, ByVal nStatus As Integer) As Integer
```

Call Syntax :

```
status = cFileResetAllAttrib(nFilename)
status = cFileResetArchive(nFilename)
status = cFileResetHidden(nFilename)
status = cFileResetReadOnly(nFilename)
status = cFileResetSystem(nFilename)
status = cFileResetFlag(nFilename, nStatus)
```

Where :

nFilename	is the filename to change the attributes
nStatus	is a combination of A_NORMAL, A_RDONLY, A_HIDDEN, A_SYSTEM, A_ARCH
status	TRUE if all is OK. FALSE if an error has been detected.

Comments :

Examples :

```
nFilename = "tmp.tmp"
nStatus = A_RDONLY or A_SYSTEM or A_HIDDEN

status = cFileResetAllAttrib(nFilename)
status = cFileResetFlag(nFilename, nStatus)
```

See also : FileSet

GetPid

Purpose :

cGetPid returns the process ID, an integer that uniquely identifies the Calling process.

Declare Syntax :

Declare Function cGetPid Lib "t2wlight.dll" () As Integer

Call Syntax :

test% = cGetPid()

Where :

test%	the return process ID

Comments :

In the MS-DOS environment, the process ID is usually considered to be the address of the program segment prefix, or PSP. However, in environments with multiple MS-DOS sessions, such as Windows, this value is often not unique. Therefore, the value returned by cGetPid in the MS-DOS libraries is a value based on a combination of the program segment prefix and the system time at the moment when cGetPid is Called for the first time.

GetPrinterPorts

Purpose :

GetPrinterPorts returns all printers set in the [printerports] section in the Win.INI

Declare Syntax :

```
Declare Function cGetPrinterPorts Lib "t2wlight.dll" () As String
```

Call Syntax :

```
test$ = cGetPrinterPorts()
```

Where :

test$	all printer founded separated by a chr$(13).

Comments :

Use the cGetIn function to extract each printer

See also : cGetDefaultPrinter

GetSectionItems

Purpose :

GetSectionItems retrieves all items founden in a section of a specified INI file.

Declare Syntax :

```
Declare Function cGetSectionItems Lib "t2wlight.dll" (ByVal Section As String, ByVal InitFile As String, nItems As Integer) As String
```

Call Syntax :

```
test$ = cGetSectionItems(Section, InitFile, nItems)
```

Where :

Section	the section to proceed
InitFile	the INI file to proceed.
nItems	the total items founden in the section
test$	the items in the specified section

Comments :

If the section don't exists, the returned file is an EMPTY string and nItems is 0.
The InitFile is any file which have a INI structure.
Each item is the section is separated by a chr$(13).

Examples :

```
    Dim n              As Integer

    Debug.Print cGetSectionItems("desktop", "win.ini", n)

    Debug.Print "Total Items founded in this section is " & n
```

On my system :

```
            Pattern=(None)
            GridGranularity=0
            IconSpacing=77
            TileWallPaper=1
            IconTitleFaceName=MS Sans Serif
            IconTitleSize=-11
            IconTitleStyle=0
            IconVerticalSpacing=72
            wallpaper=(None)

            Total Items founded in this section is = 9
```

```
    Debug.Print cGetSectionItems("intl", "win.ini", n)

    Debug.Print "Total Items founded in this section is " & n

            sLanguage=fra
            sCountry=Belgium (French)
            iCountry=32
            iDate=1
            iTime=1
            iTLZero=0
            iCurrency=3
            iCurrDigits=2
```

```
iNegCurr=8
iLzero=0
iDigits=2
iMeasure=0
s1159=
s2359=
sCurrency=FB
sThousand=.
sDecimal=,
sDate=/
sTime=:
sList=;
sShortDate=d/MM/yy
sLongDate=dddd d MMMM yyyy
sFrameNum=#mmjk`sdnm

Total Items founded in this section is = 23
```

GetSystemDirectory

Purpose :

GetSystemDirectory retrieves the full path of the System directory for Windows.

Declare Syntax :

Declare Function cGetSystemDirectory Lib "t2wlight.dll" () As String

Call Syntax :

test$ = cGetSystemDirectory()

Where :

test$	the full path of the System directory

Comments :

Examples :

```
test$ = cGetSystemDirectory()                    -> "K:\WINDOWS\SYSTEM"
```

See also : cGetWindowsDirectory

GetTaskName

Purpose :

GetTaskName reads the name of the task. You see the name in the Task Manager by pressing the CTRL + ESC keys.

Declare Syntax :

```
Declare Function cGetTaskName Lib "t2wlight.dll" (ByVal hWnd As Integer) As String
```

Call Syntax :

```
test$ = cGetTaskName(Form.hWnd)
```

Where :

Form.hWnd	is the hWnd of your application
test$	is the old task name of the application

Comments :

This is useful to retrieve the task name.

Examples :

```
    Dim TaskName        As String

    TaskName = cGetTaskName(Me.hWnd)
    MsgBox TaskName
            TaskName is "Microsoft Visual Basic"
```

See also : cChangeTaskName, cGetChangeTaskName

SetCapture, ResetCapture

Purpose :

SetCapture and ResetCapture captures or liberates the mouse and keyboard inputs to a hWnd of a control. Only this control can receive the inputs.

Declare Syntax :

```
Declare Sub cSetCapture Lib "t2wlight.dll" (ByVal hWnd As Integer)
Declare Sub cResetCapture Lib "t2wlight.dll" ()
```

Call Syntax :

```
Call cSetCapture(hWnd)
Call cResetCapture
```

Where :

hWnd	the hWnd of a control

Comments :

Use this with caution.
If your program crashes, the inputs are limited to the window specified by the control.
Only a control at a gived time can be use these functions.

GetWindowsDirectory

Purpose :

GetWindowsDirectory retrieves the full path for the Windows directory

Declare Syntax :

Declare Function cGetWindowsDirectory Lib "t2wlight.dll" () As String

Call Syntax :

test$ = cGetWindowsDirectory()

Where :

test$	is the full path

Comments :

Examples :

```
test$ = cGetWindowsDirectory()                    -> "K:\WINDOWS"
```

See also : cGetSystemDirectory

Distribution Note

When you create and distribute applications that use 'TIME TO WIN Light', you should install the file T2WLIGHT.DLL in the customer's Microsoft Windows \SYSTEM subdirectory. The setup kit included with Visual Basic provides tools that help you write setup programs that install your applications correctly.

You are not allowed to distribute '**T2WLIGHT.LIC**' *file with any application that you distribute.*

GetWinSection

Purpose :

GetWinSection retrieves all items founden in a section of the Win.INI.

Declare Syntax :

Declare Function cGetWinSection Lib "t2wlight.dll" (ByVal Section As String) As String

Call Syntax :

test$ = cGetWinSection(Section)

Where :

Section	is the section to proceed
test$	is the contents of the specified section

Comments :

Each item in the section is separated by a chr$(13).

Examples :

```
    Dim n               As Integer

    Debug.Print cGetWinSection("desktop")
```

On my system :

```
            Pattern=(None)
            GridGranularity=0
            IconSpacing=77
            TileWallPaper=1
            IconTitleFaceName=MS Sans Serif
            IconTitleSize=-11
            IconTitleStyle=0
            IconVerticalSpacing=72
            wallpaper=(None)
```

See also : cGetSectionItems

GiveBitPalindrome

Purpose :

GiveBitPalindrome returns all chars on which bit 0 is bit 7, bit 1 is bit 6, bit 2 is bit 5, bit 3 is bit 4.

Declare Syntax :

Declare Function cGiveBitPalindrome Lib "t2wlight.dll" () As String

Call Syntax :

test = cGiveBitPalindrome

Where :

test	the result

Comments :

See also : Bit String Manipulation routines

HourTo

Purpose :

HourTo converts a time string to a VARIANT value in minutes (INTEGER or LONG)

Declare Syntax :

Declare Function cHourTo Lib "t2wlight.dll" (Txt As String) As Variant

Call Syntax :

test = cHourTo(Txt)

Where :

Txt	the time to convert
test	the time in minutes

Comments :

The maximum format is for positive time "HHHHHHH:MM" and for negative time "-HHHHHH:MM"
The returned value is a VARIANT (INTEGER or LONG).

Examples :

The time "123:45"	is 7425 minutes
The time "23:58"	is 1438 minutes
The time "7:36"	is 456 minutes
The time ":24"	is 24 minutes
The time ":4"	is 4 minutes
The time ":"	is 0 minutes

The time "-123:45"	is -7425 minutes
The time "-23:58" is -1438 minutes	
The time "-7:36"	is -456 minutes
The time "-:24"	is -24 minutes
The time "-:4"	is -4 minutes
The time "-:"	is 0 minutes

See also : Date, Hour and Time routines

MixChars

Purpose :

MixChars will mix all chars in a gived string in a random position.

Declare Syntax :

Declare Function cMixChars Lib "t2wlight.dll" (Txt As String) As String

Call Syntax :

test$ = cMixChars(Txt)

Where :

Txt	is the string to mix all chars.
test$	is the returned mixed string.

Comments :

MixChars use a random number generator to perform the mix of the chars. The starting random number is depending of the actual date and time.

If the passed string is an EMPTY string, the returned string is an EMPTY string.

Examples :

```
test1$ = cMixChars("TIME TO WIN")
test2$ = cMixChars("Nothing can beat the fox")
```

On my system :

```
        test1$ = "ON EI WMTIT"
        test2$ = "Nt honn ia ttechx baefog"
```

See also :

IntoBalance, IntoBalanceFill

Purpose :

IntoBalance converts a VARIANT value (INTEGER or LONG) in a time string.
IntoBalance converts a VARIANT value (INTEGER or LONG) in a time string with leading zero.

Declare Syntax :

Declare Function cIntoBalance Lib "t2wlight.dll" (Var As Variant) As String
Declare Function cIntoBalanceFill Lib "t2wlight.dll" (Var As Variant) As String

Call Syntax :

test$ = cIntoBalance(Var)
test$ = cIntoBalanceFill(Var)

Where :

Var	the value to convert
test$	the time string

Comments :

For a positive value :
The format returned for the time string is "HHHHHH:MM"

For a negative value :
The maximum format and the minimum formart returned for the time string is "-HHHHH:MM"

Examples :

IntoBalanceFill	IntoBalance
1234 is "00020:34"	" 20:34"
1235 is "00020:35"	" 20:35"
1236 is "00020:36"	" 20:36"
1237 is "00020:37"	" 20:37"
1238 is "00020:38"	" 20:38"
1239 is "00020:39"	" 20:39"
1240 is "00020:40"	" 20:40"
1241 is "00020:41"	" 20:41"
1242 is "00020:42"	" 20:42"
1243 is "00020:43"	" 20:43"
1244 is "00020:44"	" 20:44"
1245 is "00020:45"	" 20:45"

See also : Date, Hour and Time routines

IntoDate, IntoDateFill, IntoDateNull

Purpose :

IntoDate converts a date value into a date string specified the short date format order in the Control Panel.
IntoDateFill converts a date value into a date string specified the short date format order in the Control Panel. But if the date is 0, the returned string is 10 spaces according to the maximum chars in the short date format ("dd/mm/yyyy" or "mm/dd/yyyy" or "yyyy/mm/dd").
IntoDateNull converts a date value into a date string specified the short date format order in the Control Panel. But if the date is 0, the returned string is an EMPTY string.

Declare Syntax :

```
Declare Function cIntoDate Lib "t2wlight.dll" (ByVal nDate As Long) As String
Declare Function cIntoDateFill Lib "t2wlight.dll" (ByVal nDate As Long) As String
Declare Function cIntoDateNull Lib "t2wlight.dll" (ByVal nDate As Long) As String
```

Call Syntax :

```
test$ = cIntoDate(nDate)
test$ = cIntoDateFill(nDate)
test$ = cIntoDateNull(nDate)
```

Where :

nDate	the date to proceed
test$	the date string returned

Comments :

The date to be proceed is always a LONG.
This fonction take care of the date separator specified in the Control Panel.

Examples :

```
test$ = cIntoDate(Int(Now))          -> "09/12/1994"
test$ = cIntoDateFill(Int(Now))      -> "09/12/1994"
test$ = cIntoDateNull(Int(Now))      -> "09/12/1994"

test$ = cIntoDate(-1)                -> "29/12/1899"
test$ = cIntoDateFill(-1)            -> "29/12/1899"
test$ = cIntoDateNull(-1)            -> "29/12/1899"

test$ = cIntoDate(0)                 -> "30/12/1899"
test$ = cIntoDateFill(0)             -> "          "
test$ = cIntoDateNull(0)             -> ""

test$ = cIntoDate(1)                 -> "31/12/1899"
test$ = cIntoDateFill(1)             -> "31/12/1899"
test$ = cIntoDateNul(1)              -> "31/12/1899"
```

See also : Date, Hour and Time routines

AndToken, AndTokenIn, OrToken, OrTokenIn

Purpose :

AndToken checks if all items of a list of token separated by '|' is present in a specified string.
AndTokenIn checks if all items of a list of token separated by a separator is present in a specified string.

OrToken checks if one item of a list of token separated by '|' is present in a specified string.
OrTokenIn checks if one item of a list of token separated by a separator is present in a specified string.

Declare Syntax :

```
Declare Function cAndToken Lib "t2wlight.dll" (ByVal Txt As String, ByVal Token As String) As Integer
Declare Function cAndTokenIn Lib "t2wlight.dll" (ByVal Txt As String, ByVal Token As String, ByVal Separator As
String) As Integer

Declare Function cOrToken Lib "t2wlight.dll" (ByVal Txt As String, ByVal Token As String) As Integer
Declare Function cOrTokenIn Lib "t2wlight.dll" (ByVal Txt As String, ByVal Token As String, ByVal Separator As String)
As Integer
```

Call Syntax :

```
Test% = cAndToken(Txt$, Token$)
Test% = cAndTokenIn(Txt$, Token$, Separator$)

Test% = cOrToken(Txt$, Token$)
Test% = cOrTokenIn(Txt$, Token$, Separator$)
```

Where :

Txt$	is the specified string.
Token$	is the list of token.
Separator$	is the specified separator (default is '|').
Test%	TRUE if one of the list of token is present, FALSE if not

Comments :

AndToken, AndTokenIn, OrToken, OrTokenIn works only with string without embedded chr$(0).
AndToken, AndTokenIn, OrToken, OrTokenIn are case-sensitive. Use UCase$ or LCase$ to perform no case-sensitivity.

Examples :

```
Dim Txt                 As String
Dim Token               As String
Dim Separator           As String
Dim Test        As Integer

Txt = "THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG"

Token = "THE|DOG|QUICK"
Test = cOrToken(Txt, Token)                                   -> True

Token = "the|dog|quick"
Test = cOrToken(Txt, Token)                                   -> False

Token = "the\dog\quick"
Separator = "\"
Test = cOrTokenIn(lcase$(Txt), lcase$(Token), Separator)      -> True
```

```
Token = "THE|DOG|QUICK"
Test = cAndToken(Txt, Token)                                        -> True

Token = "the|dog|quick"
Test = cAndToken(Txt, Token)                                        -> False

Token = "the\dog\quick"
Separator = "\"
Test = cAndTokenIn(lcase$(Txt), lcase$(Token), Separator)           -> True
```

See also :

IntoFixHour, IntoHour, IntoVarHour

Purpose :

IntoFixHour is super-set for converting a VARIANT (INTEGER or LONG) into a fixed time string.
IntoHour concerts a VARIANT (INTEGER or LONG) into a hour string.
IntoVarHour concerts a VARIANT (INTEGER or LONG) into a hour string (variable length following the value).

Declare Syntax :

Declare Function cIntoFixHour Lib "t2wlight.dll" (Var As Variant, ByVal Length As Integer, ByVal fillZero As Integer, ByVal Hundreds As Integer) As String
Declare Function cIntoHour Lib "t2wlight.dll" (Var As Variant) As String
Declare Function cIntoVarHour Lib "t2wlight.dll" (Var As Variant) As String

Call Syntax :

test$ = cIntoFixHour(Var, Length, fillZero, Hundreds)
test$ = cIntoHour(Var)
test$ = cIntoVarHour(Var)

Where :

Var	the VARIANT value (LONG or INTEGER) to proceed
Length	the length of the returned time string
fillZero	TRUE if the time string must be filled with zero 0, FALSE if it not
Hundreds	TRUE if the minutes must be converted in Hundreds, FALSE if it not. (This is useful for making calculation)
test$	the returned time string

Comments :

For the cIntoFixHour function, if the value can be fitted in the length specified, the return string is filled with '?'
The maximum format for the returned time string is HHHHHHHH:MM

Examples :

Convert 12345 minutes into fixed hour :

```
Length  fillZero = TRUE          fillZero = FALSE

0       ""                       ""
1       "?"                      "?"
2       "??"                     "??"
3       "???"                    "???"
4       "????"                   "????"
5       "?????"                  "?????"
6       "205:45"                 "205:45"
7       "0205:45"                " 205:45"
8       "00205:45"               "  205:45"
9       "000205:45"              "   205:45"
10      "0000205:45"             "    205:45"
11      "00000205:45"            "     205:45"
```

See also : Date, Hour and Time routines, Conversion table for Hundreds

IsBitPalindrome

Purpose :

IsBitPalindrome checks if a string is Bit palindrome

Declare Syntax :

```
Declare Function cIsBitPalindrome Lib "t2wlight.dll" (Txt As String) As Integer
```

Call Syntax :

```
test = cIsBitPalindrome(Txt)
```

Where :

Txt	the string to proceed
test	TRUE if the string is Bit palindrome
	FALSE if the string is not Bit Palindrome

Comments :

See also : Bit String Manipulation routines

ExpandTab

Purpose :

ExpandTab unpacks all tab chars into n space chars.

Declare Syntax :

Declare Function cExpandTab Lib "t2wlight.dll" (Txt As String, ByVal nTab As Integer) As String

Call Syntax :

test = cExpandTab(Txt, nTab)

Where :

Txt	the string to proceed
nTab	the number of space chars which replace a tab char
test	the result

Comments :

Examples :

```
Txt = test = "A" + chr$(9) + "B" + chr$(9) + space$(1) + "C" + char$(9) + chr$(9) + "D"
nTab = 2
test = cExpandTab(Txt, nTab)
        test =   "A" + space$(2) + "B" + space$(3) + "C" + space$(4) + "D"
```

See also : cCompress, cCompressTab

FileToLower, FileToUpper

Purpose :

FileToLower converts a file to a file with lower case.
FileToLower converts a file to a file with upper case.

Declare Syntax :

```
Declare Function cFileToLower Lib "t2wlight.dll" (ByVal file1 As String, ByVal file2 As String) As Long
Declare Function cFileToUpper Lib "t2wlight.dll" (ByVal file1 As String, ByVal file2 As String) As Long
```

Call Syntax :

```
test& = cFileToLower(file1, file2)
test& = cFileToUpper(file1, file2)
```

Where :

file1$	is the source file.
file2$	is the destination file.
test&	> 0 if all is OK (the returned value is the total bytes copied), < 0 if an error has occured.

Comments :

The returned value can be negative and have the following value :

-32720	the number of chars in a block for writing differs from the number of chars for reading.
-32730	reading error for file 1.
-32740	writing error for file 2.
-32750	opening error for file 1.
-32751	opening error for file 2.
-32760	allocation error for memory buffer 1.
-32761	allocation error for memory buffer 2.

Examples :

```
test& = cFileToLower("c:\autoexec.bat","c:\autoexec.lwr")
test& = cFileToUpper("c:\autoexec.bat","c:\autoexec.upr")
```

See also :

IsX

Purpose :

These routines checks if the specified string is :

IsAlnum	Alphanumeric ('A'-'Z', 'a'-'z', or '0'-'9')
IsAlpha	Letter ('A'-'Z' or 'a'-'z')
IsAscii	ASCII character (0x00 - 0x7F)
IsCsym	Letter, underscore, or digit
IsCsymf	Letter or underscore
IsDigit	Digit ('0'-'9')
IsISBN	International Standard Book Numbers (ISBNs)
IsLower	Lowercase letter ('a'-'z')
IsPalindrome	the string and the reverse string are the same
IsPunct	Punctuation character
IsSpace	White-space character (0x09 - 0x0D or 0x20)
IsUpper	Uppercase letter ('A'-'Z')
IsXdigit	Hexadecimal digit ('A'-'F','a'-'f', or '0'-'9')

IsBalance	test if the specified balance is a valid balance
IsDate	test if the specified date is a valid date
IsHour	test if the specified hour is a valid hour
IsLeapYear	test if the specified year is a leap year

Declare Syntax :

```
Declare Function cIsAlnum Lib "t2wlight.dll" (Txt As String) As Integer
Declare Function cIsAlpha Lib "t2wlight.dll" (Txt As String) As Integer
Declare Function cIsAscii Lib "t2wlight.dll" (Txt As String) As Integer
Declare Function cIsCsym Lib "t2wlight.dll" (Txt As String) As Integer
Declare Function cIsCsymf Lib "t2wlight.dll" (Txt As String) As Integer
Declare Function cIsDigit Lib "t2wlight.dll" (Txt As String) As Integer
Declare Function cIsISBN Lib "t2wlight.dll" (Txt As String) As Integer
Declare Function cIsLower Lib "t2wlight.dll" (Txt As String) As Integer
Declare Function cIsPalindrome Lib "t2wlight.dll" (Txt As String) As Integer
Declare Function cIsPunct Lib "t2wlight.dll" (Txt As String) As Integer
Declare Function cIsSpace Lib "t2wlight.dll" (Txt As String) As Integer
Declare Function cIsUpper Lib "t2wlight.dll" (Txt As String) As Integer
Declare Function cIsXDigit Lib "t2wlight.dll" (Txt As String) As Integer

Declare Function cIsBalance Lib "t2wlight.dll" (ByVal nHour As Long, ByVal nMinute As Integer, ByVal nSecond As
Integer) As Integer
Declare Function cIsDate Lib "t2wlight.dll" (ByVal nYear As Integer, ByVal nMonth As Integer, ByVal nDay As Integer)
As Integer
Declare Function cIsHour Lib "t2wlight.dll" (ByVal nHour As Integer, ByVal nMinute As Integer, ByVal nSecond As
Integer) As Integer
Declare Function cIsLeapYear Lib "t2wlight.dll" (ByVal nYear As Integer) As Integer
```

Call Syntax :

```
test = cIsAlnum(Txt)
test = cIsAlpha(Txt)
test = cIsAscii(Txt)
test = cIsCsym(Txt)
test = cIsCsymf(Txt)
test = cIsDigit(Txt)
test = cIsLower(Txt)
test = cIsPalindrome(Txt)
test = cIsPunct(Txt)
test = cIsSpace(Txt)
```

test = cIsUpper(Txt)
test = cIsXdigit(Txt)

test = cIsBalance(nHour, nMinute, nSecond)
test = cIsDate(nYear, nMonth, nDay)
test = cIsHour(nHour, nMinute, nSecond)
test = cIsLeapYear(nYear)

Where :

Txt	the string to proceed
nHour	the hour to test (can be negative and/or greater than 1439 for cIsBalance)
nMinute	the minute to test
nSecond	the second to test
nYear	the year to test
nMonth	the month to test
nDay	the dat to test
test	TRUE if test is OK
	FALSE if the test fails

Comments :

Examples :

Txt = "ABCDEFG"

test = cIsAlnum(Txt)	TRUE
test = cIsAlpha(Txt)	TRUE
test = cIsAscii(Txt)	TRUE
test = cIsCsym(Txt)	TRUE
test = cIsCsymf(Txt)	TRUE
test = cIsDigit(Txt)	FALSE
test = cIsLower(Txt)	FALSE
test = cIsPalindrome(Txt)	FALSE
test = cIsPunct(Txt)	FALSE
test = cIsSpace(Txt)	FALSE
test = cIsUpper(Txt)	TRUE
test = cIsXdigit(Txt)	FALSE

test = cIsBalance(-1200, 58, 34)	TRUE
test = cIsDate(1995, 2, 29)	FALSE
test = cIsHour(23, 60, 10)	FALSE
test = cIsLeapYear(1996)	TRUE

See also : IsX Family Test routines

FileMerge

Purpose :

FileMerge merges two files in one.

Declare Syntax :

Declare Function cFileMerge Lib "t2wlight.dll" (ByVal file1 As String, ByVal file2 As String, ByVal fileTo As String) As Long

Call Syntax :

test& = cFileMerge(file1, file2, fileTo)

Where :

file1$	is the first file.
file2$	is the second file.
fileTo$	is the destination file.
test&	> 0 if all is OK (the returned value is the total bytes copied), < 0 if an error has occured.

Comments :

The returned value can be negative and have the following value :

-32720	the number of chars in a block for writing differs from the number of chars for reading file 1.
-32721	the number of chars in a block for writing differs from the number of chars for reading file 2.
-32730	reading error for file 1.
-32731	reading error for file 2.
-32740	writing error for file To.
-32750	opening error for file 1.
-32751	opening error for file 2.
-32752	opening error for file To.
-32760	allocation error for memory buffer.

Examples :

```
test& = cFileMerge("c:\autoexec.bat", "c:\config.sys", "c:\merge.byt")
```

See also : cFileCopy

GetClassName

Purpose :

GetClassName retrieves the full class name of a control.

Declare Syntax :

Declare Function cGetClassName Lib "t2wlight.dll" (ByVal hWnd As Integer) As String

Call Syntax :

test$ = cGetClassName(hWnd)

Where :

hWnd	is the .hWnd of a control.
test$	is the returned class name.

Comments :

if the .hWnd is not exist, the returned string is an EMPTY string.

Examples :

```
test$ = cGetClassName(Me.hWnd)              -> "ThunderForm"
test$ = cGetClassName(Command1.hWnd)        -> "ThunderCommandButton"
test$ = cGetClassName(List1.hWnd)                   -> "ThunderListBox"
test$ = cGetClassName(Text1.hWnd)                   -> "ThunderTextBox"
```

See also : cGetClass, cGetCtlClass

Returned Errors

-32720
: The number of chars in a block for writing differs from the number of chars for reading.

-32730
: An error has occured when reading the file (bad CRC, bad cluster, ...).

-32740
: An error has occured when writing a file (bad CRC, bad cluster, not a valid drive, not enough space on drive).

-32759 to -32750
: An error has occured when opening a file.

-32767 to -32761
: An error has occured when allocating memory buffer

KillDir

Purpose :

KillDir deletes the specified empty directory.
KillDirs deletes the specified direcory and its associated directories.

Declare Syntax :

Declare Function cKillDir Lib "t2wlight.dll" (ByVal lpDir As String) As Integer
Declare Function cKillDirs Lib "t2wlight.dll" (ByVal lpDir As String, ByVal HeaderDirectory As Integer) As Integer

Call Syntax :

test% = cKillDir(lpDir$)
test% = cKillDirs(lpDir$)

Where :

lpDir$	is the directory to proceed
HeaderDirectory%	specify if lpDir$ must be delete also
test%	see below

Comments :

For cKillDir :

The directory must be empty, and it must not be the current working directory or the root directory.
The returned value is TRUE if all is OK, <> TRUE if an error has occured.

For cKillDirs :

Don't forget that this function can handle a maximum of 700 directories of 70 chars long each.

The returned value can be negative :
-32760 allocation error for memory buffer.

This function doesn't generates an VB Error if the speficied dir not exists.

See also : cKillFile, cKillFiles, cKillDirFilesAll

KillFile, KillFileAll

Purpose :

KillFile deletes the specified filename.
KillFileAll deletes the specified filename with any attribute.

Declare Syntax :

```
Declare Function cKillFile Lib "t2wlight.dll" (ByVal lpFilename As String) As Integer
Declare Function cKillFileAll Lib "t2wlight.dll" (ByVal lpFilename As String) As Integer
```

Call Syntax :

```
test% = cKillFile(lpFilename)
test% = cKillFileAll(lpFilename)
```

Where :

lpFileName	the filename to proceed
test%	TRUE if all is OK
	<> TRUE if an error has occured

Comments :

If the file is a combination of READ-ONLY or SYSTEM or HIDDEN attribute, you must use cKillFileAll to remove it.
If the file is an opened file, the returned value is always <> TRUE.
If the file not exist, the returned value is always = TRUE.
This function doesn't generates an VB Error if the speficied file not exists.

See also : cKillFiles, cKillFilesAll, cKillDir, cKillDirs, cKillDirFilesAll

KillFilesAll

Purpose :

KillFiles deletes all files specified by a file mask.
KillFilesAll deletes all files specified by a file mask even if some files are READ-ONLY files.

Declare Syntax :

Declare Function cKillFiles Lib "t2wlight.dll" (ByVal lpFilename As String) As Integer
Declare Function cKillFilesAll Lib "t2wlight.dll" (ByVal lpFilename As String) As Integer

Call Syntax :

test% = cKillFiles(lpFilename)
test% = cKillFilesAll(lpFilename)

Where :

lpFilename	the mask file to proceed
test%	> 0 if all is OK. The returned value specified the total files deleted.
	= 0 if an error has occured

Comments :

If some files are a combination of READ-ONLY or SYSTEM or HIDDEN attributes, you must use cKillFilesAll to remove it.
If the mask is invalid or if the file not exists or if an error occurs when accessing the files, the return value is 0.
This function doesn't generates an VB Error if the speficied files not exists.

See also : cKillFile, cKillFileAll, cKillDir, cKillDirs

Lrc

Purpose :

Lrc calculates the LRC of a gived string.

Declare Syntax :

Declare Function cLrc Lib "t2wlight.dll" (Txt As String) As String

Call Syntax :

test$ = cLrc(Txt)

Where :

Txt	the string to proceed
test$	the LRC calculated

Comments :

The LRC is always an Hexa string of two chars.
This function is used for communication between a program and a clocking terminal

Examples :

```
test$ = cLrc(chr$(2) & "0a12721536")                    -> "54"
```

See also : cStringCRC32, cFileCRC32

MakeDir, MakeMultipleDir

Purpose :

MakeDir creates the specified directory.
MakeMultipleDir creates a multiple directory in one call.

Declare Syntax :

```
Declare Function cMakeDir Lib "t2wlight.dll" (ByVal lpFilename As String) As Integer
Declare Function cMakeMultipleDir Lib "t2wlight.dll" (ByVal lpFilename As String) As Integer
```

Call Syntax :

```
test% = cMakeDir(lpFilename)
test% = cMakeMultipleDir(lpFilename)
```

Where :

lpFilename	the path for the new directory
test%	TRUE if all is OK
	<> TRUE if an error has occured

Comments :

The MakeDir function creates a new directory with the specified dirname. Only one directory can be created at a time, so only the last
component of dirname can name a new directory.
The MakeDir function does not do any translation of path delimiters. All operating systems accept either " or "/ " internally as valid delimiters within paths.
This fonction is the same that MkDir but doesn't generate an VB Error if a problem occurs.

The MakeMultipleDir function creates a new multiple directory with the specified dirname. MakeMultipleDir doesn't return an error if a sub-directory in the multiple directory is already present. The only final test is the existence of the full multiple directory when it was been created.

Examples :

```
test% = cMakeDir("C:\")                -> 13 (<> TRUE => an error has occured)
test% = cMakeDir("C:\~~TEST~~")        -> TRUE (no error, the directory has been created)

test% = cMakeMultipleDir("C:\~~TEST~~\TEST\TMP")        -> TRUE (no error, the directory has been created)
```

See also : cChDir, cKillDir

Max

Purpose :

Max returns the highest value of the two VARIANT value (INTEGER or LONG)

Declare Syntax :

Declare Function cMax Lib "t2wlight.dll" (Var1 As Variant, Var2 As Variant) As Variant

Call Syntax :

test = cMax(Var1, Var2)

Where :

Var1	the first value
Var2	the second value
test	the highest value of the two

Comments :

Examples :

```
test = cMax(1234, 4321)            -> 4321
```

See also : cMin

MaxD

Purpose :

MaxD will return the largest value in a Double array.

Declare Syntax :

Declare Function cMaxD Lib "t2wlight.dll" (array() As Double) As Double

Call Syntax :

largest = cMaxD(array())

Where :

array()	is the Double array.
largest	is the largest value from all of the elements of the Double array.

Comments :

See Also : cMaxI, cMaxL, cMaxS, Array routines

MaxI

Purpose :

MaxI will return the largest value in an Integer array.

Declare Syntax :

Declare Function cMaxI Lib "t2wlight.dll" (array() As Integer) As Integer

Call Syntax :

largest = cMaxI(array())

Where :

array()	is the Integer array.
largest	is the largest value from all of the elements of the Integer array.

Comments :

See Also : cMaxD, cMaxL, cMaxS, Array routines

MaxL

Purpose :

MaxL will return the largest value in a Long array.

Declare Syntax :

Declare Function cMaxL Lib "t2wlight.dll" (array() As Long) As Long

Call Syntax :

largest = cMaxL(array())

Where :

array()	is the Long array.
largest	is the largest value from all of the elements of the Long array.

Comments :

See Also : cMaxD, cMaxI, cMaxS, Array routines

MaxS

Purpose :

MaxS will return the largest value in a Single array.

Declare Syntax :

Declare Function cMaxS Lib "t2wlight.dll" (array() As Single) As Single

Call Syntax :

largest = cMaxS(array())

Where :

array()	is the Single array.
largest	is the largest value from all of the elements of the Single array.

Comments :

See Also : cMaxD, cMaxI, cMaxL, Array routines

MeanD

Purpose :

MeanD will calculate the mean from all elements in a Double array.

Declare Syntax :

Declare Function cMeanD Lib "t2wlight.dll" (array() As Double) As Double

Call Syntax :

mean = cMeanD(array())

Where :

array()	is the Double array.
mean	is the mean calculated. This value is always a Double value.

Comments :

See Also : cMeanD, cMeanI, cMeanL, cMeanS, Array routines

MeanI

Purpose :

MeanI will calculate the mean from all elements in an Integer array.

Declare Syntax :

Declare Function cMeanI Lib "t2wlight.dll" (array() As Integer) As Double

Call Syntax :

mean = cMeanI(array())

Where :

array()	is the Integer array.
mean	is the mean calculated. This value is always a Double value.

Comments :

See Also : cMeanD, cMeanI, cMeanL, cMeanS, Array routines

MeanL

Purpose :

MeanL will calculate the mean from all elements in a Long array.

Declare Syntax :

Declare Function cMeanL Lib "t2wlight.dll" (array() As Long) As Double

Call Syntax :

mean = cMeanL(array())

Where :

array()	is the Long array.
mean	is the mean calculated. This value is always a Double value.

Comments :

See Also : cMeanD, cMeanI, cMeanL, cMeanS, Array routines

MeanS

Purpose :

MeanS will calculate the mean from all elements in a Single array.

Declare Syntax :

Declare Function cMeanS Lib "t2wlight.dll" (array() As Single) As Double

Call Syntax :

mean = cMeanS(array())

Where :

array()	is the Single array.
mean	is the mean calculated. This value is always a Double value.

Comments :

See Also : cMeanD, cMeanI, cMeanL, cMeanS, Array routines

Min

Purpose :

Max returns the smallest value of the two VARIANT value (INTEGER or LONG)

Declare Syntax :

Declare Function cMin Lib "t2wlight.dll" (Var1 As Variant, Var2 As Variant) As Variant

Call Syntax :

test = cMin(Var1, Var2)

Where :

Var1	the first value
Var2	the second value
test	the smallest value of the two

Comments :

Examples :

```
test = cMin(1234, 4321)              -> 1234
```

See also : cMax

MinD

Purpose :

MinD will return the smallest value in a Double array.

Declare Syntax :

Declare Function cMinD Lib "t2wlight.dll" (array() As Double) As Double

Call Syntax :

smallest = cMinD(array())

Where :

array() is the Double array.
smallest is the smallest value from all of the elements of the Double array.

Comments :

See Also : cMinI, cMinL, cMinS, Array routines

MinI

Purpose :

MinI will return the smallest value in an Integer array.

Declare Syntax :

Declare Function cMinI Lib "t2wlight.dll" (array() As Integer) As Integer

Call Syntax :

smallest = cMinI(array())

Where :

array() is the Integer array.
smallest is the smallest value from all of the elements of the Integer array.

Comments :

See Also : cMinD, cMinL, cMinS, Array routines

MinL

Purpose :

MinL will return the smallest value in a Long array.

Declare Syntax :

Declare Function cMinL Lib "t2wlight.dll" (array() As Long) As Long

Call Syntax :

smallest = cMinL(array())

Where :

array()	is the Long array.
smallest	is the smallest value from all of the elements of the Long array.

Comments :

See Also : cMinD, cMinI, cMinS, Array routines

MinS

Purpose :

MinS will return the smallest value in a Single array.

Declare Syntax :

Declare Function cMinS Lib "t2wlight.dll" (array() As Single) As Single

Call Syntax :

smallest = cMinS(array())

Where :

array()	is the Single array.
smallest	is the smallest value from all of the elements of the Single array.

Comments :

See Also : cMinD, cMinI, cMinL, Array routines

NextHwnd

Purpose :

Declare Syntax :

Call Syntax :

Where :

Comments :

OneCharFromLeft

Purpose :

OneCharFromLeft reads 1 char at a position starting from the left of a string.

Declare Syntax :

Declare Function cOneCharFromLeft Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer) As String

Call Syntax :

test = cOneCharFromLeft(txt, position)

Where :

Txt	the string to extract one char
Position	the position of the char
Test	the result

Comments :

This function is the same that MID$(Txt, Position, 1)

Examples :

```
Txt = "ABCDEF"
Position = 3
Test = cOneCharFromLeft(Txt, Position)
        Test = "C"
```

See also : cBlockCharFromLeft, cBlockCharFromRight, cOneCharFromLeft, cOneCharFromRight

OneCharFromRight

Purpose :

OneCharFromRight reads 1 char at a position starting from the right of a string.

Declare Syntax :

Declare Function cOneCharFromRight Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer) As String

Call Syntax :

Test = cOneCharFromRight(Txt, Position)

Where :

Txt	the string to extract one char
Position	the position of the char
Test	the result

Comments :

This function is the same that MID$(Txt, Len(Txt) - Position + 1, 1)

Examples :

```
Txt = "ABCDEF"
Position = 3
Test = cOneCharFromRight(Txt, Position)
        Test = "D"
```

See also : cBlockCharFromLeft, cBlockCharFromRight, cOneCharFromLeft, cOneCharFromRight

RemoveBlockChar

Purpose :

Declare Syntax :

Call Syntax :

Where :

Comments :

RemoveOneChar

Purpose :

Declare Syntax :

Call Syntax :

Where :

Comments :

RenameFile

Purpose :

RenameFile renames a file or moves a file from one path to an other path.

Declare Syntax :

Declare Function cRenameFile Lib "t2wlight.dll" (ByVal lpFilename1 As String, ByVal lpFilename2 As String) As Integer

Call Syntax :

test% = cRenameFile(lpFilename1, lpFilename2)

Where :

lpFileName1	the old filename to rename
lpFileName2	the new filename to be used
test%	TRUE if all is OK
	<> TRUE if an error has occured

Comments :

The rename function renames the file or directory specified by lpFilename1 to the name given by lpFilename2. The lpFilename1 must be the
path of an existing file or directory. The lpFilename1 must not be the name of an existing file or directory.
The rename function can be used to move a file from one directory to another by giving a different path in the lpFilename2 argument.
However, files cannot be moved from one device to another (for example, from drive A to drive B). Directories can only be renamed, not
moved.
This function doesn't generates an VB Error if the speficied old filename not exists.

ResizeString

Purpose :

ResizeString resizes the size of a string to a new length.

Declare Syntax :

Declare Function cResizeString Lib "t2wlight.dll" (Txt As String, ByVal newLength As Integer) As String

Call Syntax :

Test$ = cResizeString(Txt$, Length%)

Where :

Txt$	is the specified string.
Length%	is the new length (can be shorter than the current length).
Test$	is the new string.

Comments :

The new length can be greater than the current length. In this case, chr$(0) is used to fill the rest of the string.

Examples :

```
Test$ = cResizeString("TIME TO WIN", 7)
        -> "TIME TO"
```

See also : cResizeStringAndFill

ResizeStringAndFill

Purpose :

ResizeStringAndFill the size of a string to a new length and fill it with chars if the new length is greater than the current length.

Declare Syntax :

Declare Function cResizeStringAndFill Lib "t2wlight.dll" (Txt As String, ByVal newLength As Integer, Fill As String) As String

Call Syntax :

Test$ = cResizeStringAndFill(Txt$, Length%, Fill$)

Where :

Txt$	is the specified string.
Length%	is the new length (can be shorter than the current length).
Fill$	is a char or a string to use to fill the new string.
Test$	is the new string.

Comments :

The new length can be greater than the current length. In this case, the fill string is used to fill the rest of the string.

Examples :

```
Test$ = cResizeStringAndFill("TIME TO WIN", 21, "@")
        -> "TIME TO WIN@@@@@@@@@@"

Test$ = cResizeStringAndFill("TIME TO WIN", 21, "time")
        -> "TIME TO WINtimetimeti"
```

See also : cResizeString

Reverse

Purpose :

Reverse reverses all chars in a gived string.

Declare Syntax :

```
Declare Function cReverse Lib "t2wlight.dll" (Txt As String) As String
```

Call Syntax :

```
Test$ = cReverse(Txt$)
```

Where :

Txt$	is the specified string
Test$	is the string reversed

Comments :

Examples :

```
Test$ = cReverse("TIME TO WIN")
        -> "NIW OT EMIT"
```

See also :

ReverseSortD

Purpose :

ReverseSortD will sort, in descending order, all elements in a Double array.

Declare Syntax :

Declare Function cReverseSortD Lib "t2wlight.dll" (array() As Double) As Integer

Call Syntax :

status = cReverseSortD(array())

Where :

array()	is the Double array.
status	is always TRUE.

Comments :

See Also : cReverseSortD, cReverseSortI, cReverseSortL, cReverseSortS, cReverseSortStr, Array routines

ReverseSortI

Purpose :

ReverseSortD will sort, in descending order, all elements in an Integer array.

Declare Syntax :

Declare Function cReverseSortI Lib "t2wlight.dll" (array() As Integer) As Integer

Call Syntax :

status = cReverseSortI(array())

Where :

array()	is the Integer array.
status	is always TRUE.

Comments :

See Also : cReverseSortD, cReverseSortI, cReverseSortL, cReverseSortS, cReverseSortStr, Array routines

ReverseSortL

Purpose :

ReverseSortL will sort in descending order all elements in a Long array.

Declare Syntax :

Declare Function cReverseSortL Lib "t2wlight.dll" (array() As Long) As Integer

Call Syntax :

status = cReverseSortL(array())

Where :

array()	is the Long array.
status	is always TRUE.

Comments :

See Also : cReverseSortD, cReverseSortI, cReverseSortL, cReverseSortS, cReverseSortStr, Array routines

ReverseSortS

Purpose :

ReverseSortS will sort in descending order all elements in a Single array.

Declare Syntax :

Declare Function cReverseSortS Lib "t2wlight.dll" (array() As Single) As Integer

Call Syntax :

status = cReverseSortS(array())

Where :

array()	is the Single array.
status	is always TRUE.

Comments :

See Also : cReverseSortD, cReverseSortI, cReverseSortL, cReverseSortS, cReverseSortStr, Array routines

ReverseSortStr

Purpose :

ReverseSortD will sort, in descending order, a string divided in basis elements of a fixed length.

Declare Syntax :

Declare Function cReverseSortStr Lib "t2wlight.dll" (Txt As String, ByVal nItem As Integer, ByVal ItemLength As Integer) As Integer

Call Syntax :

status = cReverseSortStr(txt, nItem, ItemLength)

Where :

txt	is the string to sort.
nItem	is the total element is the string.
ItemLength	is the length for one element.
status	is FALSE if the length of the string is not the 'nItem * ItemLength', or if length of the string is 0. is TRUE if all is OK.

Comments :

See Also : cReverseSortD, cReverseSortI, cReverseSortL, cReverseSortS, cReverseSortStr, Array routines

RomanToArabic

Purpose :

RomanToArabic converts a Roman string into an integer or a long integer.

Declare Syntax :

Declare Function cRomanToArabic Lib "t2wlight.dll" (Txt As String) As Variant

Call Syntax :

test = cRomanToArabic(txt)

Where :

txt	is a Roman string.
test	returns the Arabic representation of txt.

Comments :

The value returned by this function is an integer or a long integer.

Examples :

```
test = cArabicToRoman(1994)
        test -> MCMXCIV

test = cArabicToRoman(1995)
        test -> MCMXCV

test = cArabicToRoman(1993)
        test -> MCMXCIII
```

See Also : cArabicToRoman

SetD

Purpose :

SetD fills, with the same value, all of the elements of a Double array.

Declare Syntax :

Declare Function cSetD Lib "t2wlight.dll" (array() As Double, ByVal nValue As Double) As Integer

Call Syntax :

status = cSetD(array(), nValue)

Where :

array()	is the Double array.
nValue	is the Double value to initialize the array.
status	is always TRUE.

Comments :

See Also : cSetD, cSetI, cSetL, cSetS, Array routines

SetI

Purpose :

SetI fills, with the same value, all of the elements of an Integer array.

Declare Syntax :

Declare Function cSetI Lib "t2wlight.dll" (array() As Integer, ByVal nValue As Integer) As Integer

Call Syntax :

status = cSetI(array(), nValue)

Where :

array()	is the Integer array.
nValue	is the Integer value to initialize the array.
status	is always TRUE.

Comments :

See Also : cSetD, cSetI, cSetL, cSetS, Array routines

SetL

Purpose :

SetL fills, with the same value, all of the elements of a Long array.

Declare Syntax :

Declare Function cSetL Lib "t2wlight.dll" (array() As Long, ByVal nValue As Long) As Integer

Call Syntax :

status = cSetL(array(), nValue)

Where :

array()	is the Long array.
nValue	is the Long value to initialize the array.
status	is always TRUE.

Comments :

See Also : cSetD, cSetI, cSetL, cSetS, Array routines

SetS

Purpose :

SetS fills, with the same value, all of the elements of a Single array.

Declare Syntax :

Declare Function cSetS Lib "t2wlight.dll" (array() As Single, ByVal nValue As Single) As Integer

Call Syntax :

status = cSetS(array(), nValue)

Where :

array()	is the Single array.
nValue	is the Single value to initialize the array.
status	is always TRUE.

Comments :

See Also : cSetD, cSetI, cSetL, cSetS, Array routines

Sleep

Purpose :

Sleep suspends the current execution of a routine for a gived delay.

Declare Syntax :

```
Declare Function cSleep Lib "t2wlight.dll" (ByVal Delay As Long) As Integer
```

Call Syntax :

```
status% = cSleep(Delay)
```

Where :

Delay	is the time to sleep the current execution of a routine in milliseconds.
status%	TRUE if all is OK FALSE if the delay is below 0.

Comments :

Use this function with care.
Don't set a delay to bigger.
Don't forget that the delay is in milliseconds.

Examples :

```
status% = cSleep(-10)          -> Don't sleep, the delay is negative value.
status% = cSleep(0)            -> A very short sleeping.
status% = cSleep(7000)         -> Sleep for 7 seconds

   Dim status      As Integer

   Call cStartBasisTimer
   status = cSleep(7000)
   MsgBox "Time elapsed for the current sleeping is " & cReadBasisTimer() & " milliseconds"
```

On my system : "Time elapsed for the current sleeping is 7031 milliseconds"

SortD

Purpose :

SortD will sort, in ascending order, all elements in a Double array.

Declare Syntax :

Declare Function cSortD Lib "t2wlight.dll" (array() As Double) As Integer

Call Syntax :

status = cSortD(array())

Where :

array()	is the Double array.
status	is always TRUE.

Comments :

See Also : cSortD, cSortI, cSortL, cSortS, cSortStr, Array routines

SortI

Purpose :

SortI will sort, in ascending order, all elements in an Integer array.

Declare Syntax :

Declare Function cSortD Lib "t2wlight.dll" (array() As Integer) As Integer

Call Syntax :

status = cSortI(array())

Where :

array()	is the Integer array.
status	is always TRUE.

Comments :

See Also : cSortD, cSortI, cSortL, cSortS, cSortStr, Array routines

SortL

Purpose :

SortL will sort, in ascending order, all elements in a Long array.

Declare Syntax :

Declare Function cSortL Lib "t2wlight.dll" (array() As Long) As Integer

Call Syntax :

status = cSortL(array())

Where :

array()	is the Long array.
status	is always TRUE.

Comments :

See Also : cSortD, cSortI, cSortL, cSortS, cSortStr, Array routines

SortS

Purpose :

SortS will sort, in ascending order, all elements in a Single array.

Declare Syntax :

Declare Function cSortS Lib "t2wlight.dll" (array() As Single) As Integer

Call Syntax :

status = cSortS(array())

Where :

array()	is the Single array.
status	is always TRUE.

Comments :

See Also : cSortD, cSortI, cSortL, cSortS, cSortStr, Array routines

SortStr

Purpose :

SortD will sort, in ascending order, a string divided in basis elements of a fixed length.

Declare Syntax :

Declare Function cSortStr Lib "t2wlight.dll" (Txt As String, ByVal nItem As Integer, ByVal ItemLength As Integer) As Integer

Call Syntax :

status = cSortStr(txt, nItem, ItemLength)

Where :

txt	is the string to sort.
nItem	is the total element is the string.
ItemLength	is the length for one element.
status	is FALSE if the length of the string is not the 'nItem * ItemLength', or if length of the string is 0.
	is TRUE if all is OK.

Comments :

See Also : cSortD, cSortI, cSortL, cSortS, cSortStr, Array routines

StringCRC32

Purpose :

StringCRC32 calculates a 32 bits CRC for a gived string.

Declare Syntax :

Declare Function cStringCRC32 Lib "t2wlight.dll" (Txt As String) As Long

Call Syntax :

test = cStringCRC32(Txt)

Where :

Txt	the string to proceed
test	the calculated CRC 32 bits in a LONG.

Comments :

if the string if empty, the return value is always -1 (&hFFFFFFFF).

Examples :

test = cStringCRC32("ABCDEFG")	&hE6F94BC
test = cStringCRC32("GFEDCBA")	&hF0EC0AB3

See also : cFileCRC32, Constants and Types declaration

SubDirectory

Purpose :

SubDirectory retrieves all sub-directories from the specified mask.

Declare Syntax :

Declare Function cSubDirectory Lib "t2wlight.dll" (ByVal nFilename As String, ByVal firstnext As Integer) As String

Call Syntax :

test$ = cSubDirectory(nFilename, firstnext)

Where :

nFilename	the specified mask
firstnext	TRUE to retrieve the first directory FALSE to retrieve the next directory
test$	the retrieved directory

Comments :

To retrieve all sub-directory is a directory, you must Call first this function with the firstnext argument on TRUE and set it to FALSE for all next directory

Examples :

```
Dim Test        As String

Test = cSubDirectory("c:\*.*", True)
Do Until (Len(Test) = 0)
    Debug.Print Test
    Test = cSubDirectory("c:\*.*", False)
Loop
```

Directories with "c:\*.*" argument are :

DOS
TEMP
TMP
BAD.DIR

See also : cFilesInDirectory

SumD

Purpose :

SumD will calculate the sum from all elements in a Double array.

Declare Syntax :

Declare Function cSumD Lib "t2wlight.dll" (array() As Double) As Double

Call Syntax :

sum = cSumD(array())

Where :

array()	is the Double array.
sum	is the sum calculated. This value is always a Double value.

Comments :

See Also : cSumD, cSumI, cSumL, cSumS, Array routines

SumI

Purpose :

SumI will calculate the sum from all elements in an Integer array.

Declare Syntax :

Declare Function cSumI Lib "t2wlight.dll" (array() As Integer) As Double

Call Syntax :

sum = cSumI(array())

Where :

array()	is the Integer array.
sum	is the sum calculated. This value is always a Double value.

Comments :

See Also : cSumD, cSumI, cSumL, cSumS, Array routines

SumL

Purpose :

SumL will calculate the sum from all elements in a Long array.

Declare Syntax :

Declare Function cSumL Lib "t2wlight.dll" (array() As Long) As Double

Call Syntax :

sum = cSumL(array())

Where :

array()	is the Long array.
sum	is the sum calculated. This value is always a Double value.

Comments :

See Also : cSumD, cSumI, cSumL, cSumS, Array routines

SumS

Purpose :

SumS will calculate the sum from all elements in a Single array.

Declare Syntax :

Declare Function cSumS Lib "t2wlight.dll" (array() As Single) As Double

Call Syntax :

sum = cSumS(array())

Where :

array()	is the Single array.
sum	is the sum calculated. This value is always a Double value.

Comments :

See Also : cSumD, cSumI, cSumL, cSumS, Array routines

TimeBetween

Purpose :

TimeBetween calculates the time (in minutes) between two hours (in minutes).

Declare Syntax :

Declare Function cTimeBetween Lib "t2wlight.dll" (ByVal Hr1 As Integer, ByVal Hr2 As Integer) As Integer

Call Syntax :

test% = cTimeBetween(Hr1, Hr2)

Where :

Hr1	the first time (0 to 1439)
Hr2	the second time (0 to 1439)

Comments :

Examples :

```
test% = cTimeBetween(600, 721)          -> 121
test% = cTimeBetween(1438, 62)          -> 64
```

See also : Date, Hour and Time routines

InsertBlocks, InsertBlocksBy, InsertByMask, InsertChars

Purpose :

InsertBlocks inserts different block of char in a gived string separated by '~'.
InsertBlocks inserts different block of char in a gived string separated by a gived separator.
InsertByMask replaces the specified char by a string in a gived string.
InsertChars insert a string starting at a gived position in a gived string.

Declare Syntax :

```
Declare Function cInsertBlocks Lib "t2wlight.dll" (Txt As String, Insert As String) As String
Declare Function cInsertBlocksBy Lib "t2wlight.dll" (Txt As String, Insert As String, Delimitor As String) As String
Declare Function cInsertByMask Lib "t2wlight.dll" (Txt As String, Mask As String, Insert As String) As String
Declare Function cInsertChars Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer, Insert As String) As String
```

Call Syntax :

```
test$ = cInsertBlocks(Txt, Insert)
test$ = cInsertBlocksBy(Txt, Insert, Delimitor)
test$ = cInsertByMask(Txt, Mask, Insert)
test$ = cInsertChars(Txt, Position, Insert)
```

Where :

Txt	the string to proceed
Insert	the string to insert
Delimitor	the delimitor to use for the insert string
Mask	the mask to use for the insert string
Position	the position to use for the insert string

Comments :

•If the size of the string is 0 The returned string is an empty string.
•The function cInsertBlocks is a subset of the cInsertBlocksBy function.
•The number of blocks for cInsertBlocks, cInsertBlocksBy functions in the string to proceed must be greater than one from the number of block in the insert string.
•The function cInsertChars is similar to LEFT$(Txt, n) + Insert + RIGHT$(Txt, LEN(Txt) - n)

Examples :

```
test$ = cInsertBlocks("A~BC~DEF", "x~yz")                    -> "AxBCyzDEF"

test$ = cInsertBlocksBy("U/VW/XYZ", "a/bc", "/")             -> "UaVWbcXYZ"

test$ = cInsertByMask("Nr ## Price $###.##", "#", "0705200")  -> "Nr 07 Price $052.00"

test$ = cInsertChars("ABCDEFG", 3, "wxyz")                    -> "ABCwxyzDEFG"
test$ = cInsertChars("ABCDEFG", 90, "wxyz")                   -> "ABCDEFGwxyz"
test$ = cInsertChars("ABCDEFG", 0, "wxyz")                    -> "wxyzABCDEFG"
```

See also : cGet, cGetIn, cGetBlock

AddDigit, CplDigit, NumDigit, CplAlpha

Purpose :

AddDigit sums all numerics chars in a gived string.
CplDigit returns the complementary string from a gived string composed with numerics chars.
NumDigit sums and sums all numerics chars in a gived string to have a maximum value of 9.
CplDigit returns the complementary string from a gived string composed with ascii chars.

Declare Syntax :

```
Declare Function cAddDigit Lib "t2wlight.dll" (Txt as string) As Integer
Declare Function cCplDigit Lib "t2wlight.dll" (Txt as string) As String
Declare Function cNumDigit Lib "t2wlight.dll" (Txt as string) As Integer
Declare Function cCplAlpha Lib "t2wlight.dll" (Txt As String) As String
```

Call Syntax :

```
test% = cAddDigit(Txt)
test$ = cCplDigit(Txt)
test% = cNumDigit(Txt)
test$ = cCplAlpha(Txt)
```

Where :

Txt$	the string to proceed
test%	the result
test$	the result for CplAlpha

Comments :

For AddDigit, CplDigit, NumDigit if one or more chars are different from digit, the value for each one is 0

Examples :

```
test% = cAddDigit("123456789098765432171234567890987654321 7")  -> 194
```

```
test% = cAddDigit("1234567890987654321712345678909876543217")  -> 194
test% = cNumDigit("1234567890987654321712345678909876543217")-> 5

test$ = cCplDigit("1234567890987654321712345678909876543217")    ->
"8765432109012345678287654321090123456782"

test% = cAddDigit("8765432109012345678287654321090123456782")  -> 166
test% = cNumDigit("8765432109012345678287654321090123456782")-> 4

test$ =    cCplAlpha("ÀÁÂÃÄÅÆ")                                    -> "?>=<;:9"
```

GetCtlX

Purpose :

The functions below applies to a custom control.

GetCtlCaption returns the .Caption property.
GetCtlClass returns the class name defined in the properties windows in the design-mode of VB.
GetCtlContainer returns the name of the container did contains the control. The container can be the form or an another control.
GetCtlDataField returns the .DataField property.
GetCtlForm returns the name of the form did contains the control.
GetCtlIndex returns the .Index property. If the control has no index, -1 is returned.
GetCtlName returns the .Name of the control.
GetCtlNameIndex returns the name and the of the control. The format is Name(x), if no index => Name is used.
GetCtlPropCaption returns the position of the .Caption property in the definition table of the control.
GetCtlPropDataField returns the position of the .DataField property in the definition table of the control.
GetCtlPropText returns the position of the .Text property in the definition table of the control.
GetCtlTag returns the .Tag property of the control. The returned string is limited to the first chr$(0) founded.
GetCtlTagSized returns the full .Tag property of the control.
GetCtlText returns the .Text property of the control.
GetHwnd returns the .hWnd of the control. If the control has no .hWnd, the returned value is 0.

Declare Syntax :

```
Declare Function cGetCtlCaption Lib "t2wlight.dll" (Ctl As Control) As String
Declare Function cGetCtlClass Lib "t2wlight.dll" (Ctl As Control) As String
Declare Function cGetCtlContainer Lib "t2wlight.dll" (Ctl As Control) As String
Declare Function cGetCtlDataField Lib "t2wlight.dll" (Ctl As Control) As String
Declare Function cGetCtlForm Lib "t2wlight.dll" (Ctl As Control) As String
Declare Function cGetCtlIndex Lib "t2wlight.dll" (Ctl As Control) As Integer
Declare Function cGetCtlName Lib "t2wlight.dll" (Ctl As Control) As String
Declare Function cGetCtlNameIndex Lib "t2wlight.dll" (Ctl As Control) As String
Declare Function cGetCtlPropCaption Lib "t2wlight.dll" (Ctl As Control) As Integer
Declare Function cGetCtlPropDataField Lib "t2wlight.dll" (Ctl As Control) As Integer
Declare Function cGetCtlPropText Lib "t2wlight.dll" (Ctl As Control) As Integer
Declare Function cGetCtlTag Lib "t2wlight.dll" (Ctl As Control) As String
Declare Function cGetCtlTagSized Lib "t2wlight.dll" (Ctl As Control) As String
Declare Function cGetCtlText Lib "t2wlight.dll" (Ctl As Control) As String
Declare Function cGetHwnd Lib "t2wlight.dll" (Ctl As Control) As Integer
```

Call Syntax :

The purpose and the declare syntax are very explicite.

Where :

Ctl	the name of the control to proceed

Comments :

•The advantage to use these routines is that these routines doesn't generates an error if the property not exists.

Examples :

See also : cGetX, cSetX, cSetCtlX

TrueBetween

Purpose :

TrueBetween checks to see if a value is fully between two other values.

Declare Syntax :

Declare Function cTrueBetween Lib "t2wlight.dll" (Var As Variant, Var1 As Variant, Var2 As Variant) As Integer

Call Syntax :

test = cTrueBetween(var, var1, var2)

Where :

var	value to test
var1	first value
var2	second value
test	TRUE if var is fully between var1 and var2
	FALSE if var is not fully between var1 and var2

Comments :

var, var1, var2 are Variant value. In this routine, only Integer, Long, Single, Double are supported.

Examples :

```
var = 5
var1 = 1
var2 = 10
test = cTrueBetween(var, var1, var2)
        -> test = TRUE

var = 10
test = cTrueBetween(var, var1, var2)
        -> test = FALSE
```

See Also : cBetween

GetX

Purpose :

The functions below applies to the .hWnd of a custom control.

GetCaption returns the .Caption property.
GetClass returns the class name defined in the properties windows in the design-mode of VB.
GetContainer returns the name of the container did contains the control. The container can be the form or an another control.
GetDataField returns the .DataField property.
GetForm returns the name of the form did contains the control.
GetIndex returns the .Index property. If the control has no index, -1 is returned.
GetNameIndex returns the name and the of the control. The format is Name(x), if no index => Name is used.
GetText returns the .Text property of the control.

Declare Syntax :

```
Declare Function cGetCaption Lib "t2wlight.dll" (ByVal hWnd As Integer) As String
Declare Function cGetClass Lib "t2wlight.dll" (ByVal hWnd As Integer) As String
Declare Function cGetContainer Lib "t2wlight.dll" (ByVal hWnd As Integer) As String
Declare Function cGetDataField Lib "t2wlight.dll" (ByVal hWnd As Integer) As String
Declare Function cGetForm Lib "t2wlight.dll" (ByVal hWnd As Integer) As String
Declare Function cGetIndex Lib "t2wlight.dll" (ByVal hWnd As Integer) As Integer
Declare Function cGetNameIndex Lib "t2wlight.dll" (ByVal hWnd As Integer) As String
Declare Function cGetText Lib "t2wlight.dll" (ByVal hWnd As Integer) As String
```

Call Syntax :

The purpose and the declare syntax are very explicite.

Where :

hWnd	the hWnd of the custom control.

Comments :

•The advantage to use these routines is that these routines doesn't generates an error if the property not exists.
•If the custom control doesn't have a .hWnd (Label control b.e.), you must use the cGetCtlX function.

Examples :

See also : cGetCtlX ,cSetX, cSetCtlX

MakePath

Purpose :

MakePath creates a single path, composed of a drive letter, directory path, filename, and filename extension.

Declare Syntax :

Declare Function cMakePath Lib "t2wlight.dll" (ByVal nDrive As String, ByVal nDir As String, ByVal nFilename As String, ByVal Ext As String) As String

Call Syntax :

test$ = cMakePath(nDrive, nDir, nFilename, Ext)

Where :

nDrive

The nDrive argument contains a letter (A, B, etc.) corresponding to the desired drive and an optional trailing colon. MakePath routine will insert the colon automatically in the composite path if it is missing. If drive is a null character or an empty string, no drive letter and colon will appear in the composite path string.

nDir

The nDir argument contains the path of directories, not including the drive designator or the actual filename. The trailing slash is optional, and either forward slashes (\) or backslashes (/) or both may be used in a single dir argument. If a trailing slash (/ or \ ) is not specified, it will be inserted automatically. If dir is a null character or an empty string, no slash is inserted in the composite path string.

nFilename

The nFilename argument contains the base filename without any extensions. If nFilename is an EMPTY string, no filename is inserted in the composite path string.

Ext

The Ext argument contains the actual filename extension, with or without a leading period (.). MakePath routine will insert the period automatically if it does not appear in ext. If ext is a null character or an empty string, no period is inserted in the composite path string.

Comments :

Examples :

```
test1$ = cMakePath("c","tmp","test","dat")
test2$ = cMakePath("c","\tmp","test","dat")
test3$ = cMakePath("c","tmp","test","")
test4$ = cMakePath("c","","test","dat")
```

On my system :

```
        test1$ = "c:tmp\test.dat"
        test2$ = "c:\tmp\test.dat"
        test3$ = "c:tmp\test"
        test4$ = "c:test.dat"
```

See also : cSplitPath, cFullPath

Uncompact

Purpose :

Uncompact uncompacts a string composed of numeric chars.

Declare Syntax :

Declare Function cUncompact Lib "t2wlight.dll" (Txt As String) As String

Call Syntax :

test = cUncompact(Txt)

Where :

Txt	is the string (only numeric chars) to uncompact
test	returns the string uncompacted

Comments :

The size of the returned string is always a multiple of 2.

Examples :

```
Txt = "0123456789"
test = cUncompact(Txt)
        test = "30313233343536373839"
```

See also : cCompact

UniqueFileName

Purpose :

UniqueFileName creates a unique filename by modifying the given template argument. The template argument must be a string with two chars maximum.

Declare Syntax :

Declare Function cUniqueFileName Lib "t2wlight.dll" (Txt As String) As String

Call Syntax :

test$ = cUniqueFileName(Txt)

Where :

Txt	the filename pattern. If the size is greater than 2, the default pattern is used.
test$	the unique filename in the form of the chars specifien in Txt plus one char and five digits.

Comments :

The alphanumeric character is 0 ('0') the first time cUniqueFileName is Called with a given template.
In subsequent Calls from the same process with copies of the same template, cUniqueFileName checks to see if previously returned names have been used to create files. If no file exists for a given name, cUniqueFileName returns that name. If files exist for all previously returned names, cUniqueFileName creates a new name by replacing the alphanumeric character in the name with the next available lowercase letter. For example, if the first name returned is t012345 and this name is used to create a file, the next name returned will be ta12345. When creating new names, cUniqueFileName uses, in order, '0' and then the lowercase letters 'a' through 'z'.

Note that the original template is modified by the first Call to cUniqueFileName. If you then Call the cUniqueFileName function again with the same template (i.e., the original one), you will get an error.

The cUniqueFileName function generates unique filenames but does not create or open files. If the filename returned is not created, each subsequent Calls returns the same filename.

If the filename pattern is not specified (by passing an EMPTY string), the default pattern '~~' is used.

Examples :

```
Dim Tmp       As String

Tmp = cUniqueFileName("MC")                    -> "MC040201"
debug.print Tmp
Close #1
Open "c:\" + Tmp For Output Shared As #1
Close #1

Tmp = cUniqueFileName("MC")                    -> "MCa40201"
debug.print Tmp
Close #1
Open "c:\" + Tmp For Output Shared As #1
Close #1

Tmp = cUniqueFileName("MC")                    -> "MCb40201"
debug.print Tmp
Close #1
Open "c:\" + Tmp For Output Shared As #1
Close #1
```

If you don't create the file, the same filename is returned, see below :

```
Tmp = cUniqueFileName("MC")                    -> "MCc40201"
Tmp = cUniqueFileName("MC")                    -> "MCc40201"
Tmp = cUniqueFileName("MC")                    -> "MCc40201"
```

ChangeChars

Purpose :

ChangeChars changes all chars specifien by others chars in a string.

Declare Syntax :

Declare Sub cChangeChars Lib "t2wlight.dll" (Txt As String, charSet As String, newCharSet As String)

Call Syntax :

Call cChangeChars(Txt, charSet, newCharSet)

Where :

Txt	the string to process
charSet	the chars in the string to be changed
newCharSet	the new chars

Comments :

Normally, the size of the newCharSet and charSet must be the same. If the size are not the same, the smallest size is used.

Examples :

```
Txt = "ABCDEF"
charSet = "ACE"
newCharSet = "ace"
Call cChangeChars(Txt, charSet, newCharSet)
        Txt = "aBcDeF"
```

See also : cChangeCharsUntil

ChangeCharsUntil

Purpose :

ChangeCharsUntil changes all chars specifien by others chars in a string until a char is encountered.

Declare Syntax :

Declare Sub cChangeCharsUntil Lib "t2wlight.dll" (Txt As String, charSet As String, newCharSet As String, nUntil As String)

Call Syntax :

Call cChangeChars(Txt, charSet, newCharSet, nUntil)

Where :

Txt	the string to process
charSet	the chars in the string to be changed
newCharSet	the new chars
nUntil	the char to stop the change

Comments :

Normally, the size of the newCharSet and charSet must be the same.   If the size are not the same, the smallest size is used.
If the size of nUntil is 0 then all chars of the string is proceeded.
If the size of nUntil is >1 only the first char is used.

Examples :

```
Txt = "ABCDEF"
charSet = "ACE"
newCharSet = "ace"
nUntil = "D"
Call cChangeCharsUntil(Txt, charSet, newCharSet, nUntil)
            Txt = "aBcDEF"
```

See also : cChangeChars

ChangeTaskName

Purpose :

ChangeTaskName changes the name of the task. You see change in the Task Manager by pressing the CTRL + ESC keys.

Declare Syntax :

```
Declare Sub cChangeTaskName Lib "t2wlight.dll" (ByVal hWnd As Integer, ByVal Text As String)
```

Call Syntax :

```
Call cChangeTaskName(Form.hWnd, Text)
```

Where :

Form.hWnd	is the hWnd of your application
Text	is the new task name to given at your application

Comments :

This is useful to set a particular task name at your application.

Examples :

```
Call cChangeTaskName(Me.hWnd, "Hello world")
        -> press the CTRL + ESC keys to see the change in the Task Manager
```

See also : cGetTaskName, cGetChangeTaskName

ArrayStringOnDisk

Purpose :

Put/Get full variable string array (one dimension) on/from disk ascii file.

Declare Syntax :

Declare Function cArrayStringOnDisk Lib "t2wlight.dll" (ByVal File As String, Array() As Any, ByVal GetPut As Integer, rRecords As Long) As Long

Call Syntax :

test& = cArrayOnDisk(File$, Array(), GetPut%, rRecords&)

Where :

File$	is the file to use.
Array()	is the variable array string with one dimension.
GetPut%	PUT_ARRAY_ON_DISK to put the array on disk, GET_ARRAY_ON_DISK to get the array from disk.
rRecords&	the returned number of records.
test&	>=0 is the returned length of the file, < 0 is an error occurs (error n° is the negative value of all DA_x values, see Constants and Types declaration).

Comments :

This function can handle only a variable type'd string derived from tagVARSTRING (see below).

Don't forget that if you use the 'ReDim' statement at the procedure level without have declared the array als Global, you must initialize the array before using this function (see below). You must initialize the array with enough space to handle the size of the file This is due to a VB limitation.

When reading, if the number of lines in the file is below the size of the array, the remain items in the array are set to EMPTY string. The CR + LF are not included in the array.

When writing, all lines are appended with CR + LF.

This function can handle huge array (greater than 65535 bytes) (see the example below).

```
Type tagVARSTRING
        Contents                As String
End Type
```

Examples :

```
ReDim AD(-999 To 1000)          As tagVARSTRING
Dim i                           As Long
Dim r                           As Long

For i = -999 To 1000
        AD(i).Contents = Space$(256)
Next i

Debug.Print cArrayOnDisk("c:\autoexec.bat", AD(), GET_ARRAY_ON_DISK, r)

Debug.Print cArrayOnDisk("c:\autoexec.tab", AD(), PUT_ARRAY_ON_DISK, r)

For i = -999 To 1000
        AD(i).Contents = Space$(256)
Next i

Debug.Print cArrayOnDisk("c:\autoexec.tab", AD(), GET_ARRAY_ON_DISK, r)

Debug.Print AD(-999).Contents
Debug.Print AD(-998).Contents
```

See also : cArrayOnDisk

EnableFI, DisableFI

Purpose :

EnableFI and DisableFI enables or disables mouse and keyboard input to the given form by sending a WM_ENABLE message and displaying an invisible control such a picture or an image. When input is disabled, the form ignores input such as mouse clicks and key presses. When input is enabled, the form processes all input.

Declare Syntax :

```
Declare Sub cEnableFI Lib "t2wlight.dll" (Ctl As Control)
Declare Sub cDisableFI Lib "t2wlight.dll" (Ctl As Control)
```

Call Syntax :

```
Call cEnableFI(Ctl)
Call cDisableFI(Ctl)
```

Where :

Ctl	the invisible control that you want become visible (cDisableFI) or invisible (cEnableFI).

Comments :

I use this function with a picture control which containes a timer BMP.

If the enabled state of the form is changing, a WM_ENABLE message is sent before this function returns. If a form is already disabled, all its child forms are implicitly disabled, although they are not sent a WM_ENABLE message.

After some experience, I've noted that some custom controls doesn't answers correctly to this function. In fact, all controls can't receive the input when you Call cDisableFI.

Use this with caution.

See also : cEnableForm, cDisableForm

EnableForm, DisableForm

Purpose :

EnableForm and DisableForm enables or disables mouse and keyboard input to the given form by sending a WM_ENABLE message. When input is disabled, the form ignores input such as mouse clicks and key presses. When input is enabled, the form processes all input.

Declare Syntax :

```
Declare Sub cEnableForm Lib "t2wlight.dll" (ByVal hWnd As Integer)
Declare Sub cDisableForm Lib "t2wlight.dll" (ByVal hWnd As Integer)
```

Call Syntax :

```
Call cEnableForm(Form.hWnd)
Call cDisableForm(Form.hWnd)
```

Where :

Form.hWnd	the .hWnd of the specified form

Comments :

If the enabled state of the form is changing, a WM_ENABLE message is sent before this function returns. If a form is already disabled, all its child forms are implicitly disabled, although they are not sent a WM_ENABLE message.

Use this with caution.

See also : cEnableFI, cDisableFI

EnableRedraw, DisableRedraw, EnableCtlRedraw, DisableCtlRedraw

Purpose :

EnableRedraw and DisableRedraw sends a WM_SETREDRAW message from a hWnd of a control to allow changes in that window to be redrawn or to prevent changes in that window from being redrawn.

EnableCtlRedraw and DisableCtlRedraw sends a WM_SETREDRAW message to a control to allow changes in that window to be redrawn or to prevent changes in that window from being redrawn.

Declare Syntax :

```
Declare Sub cEnableRedraw Lib "t2wlight.dll" (ByVal hWnd As Integer)
Declare Sub cDisableRedraw Lib "t2wlight.dll" (ByVal hWnd As Integer)

Declare Sub cEnableCtlRedraw Lib "t2wlight.dll" (Ctl As Control)
Declare Sub cDisableCtlRedraw Lib "t2wlight.dll" (Ctl As Control)
```

Call Syntax :

```
Call cEnableRedraw(Ctl.hWnd)
Call cDisableRedraw(Ctl.hWnd)

Call cEnableCtlRedraw(Ctl)
Call cDisableCtlRedraw(Ctl)
```

Where :

Comments :

The WM_SETREDRAW message can be used to set and clear the redraw flag for a window. This message is very useful for
preventing a list box from being updated when many items are being added to it, and then allowing the list box to be redrawn when all
of the changes have been made to its contents. Using this technique prevents a list box that is currently visible from flashing
constantly as its contents are being updated.

This message sets or clears the redraw flag. If the redraw flag is cleared, the contents of the specified window will not be updated
after each change, and the window will not be repainted until the redraw flag is set. For example, an application that needs to add
several items to a list box can clear the redraw flag, add the items, and then set the redraw flag. Finally, the application can Call the
InvalidateRect function to cause the list box to be repainted.

If the custom control doesn't have a .hWnd (Label control b.e.), you must use the XCtlRedraw routine.

Fill

Purpose :

Fill fills a string with some chars.

Declare Syntax :

Declare Sub cFill Lib "t2wlight.dll" (Txt As String, Fill As String)

Call Syntax :

Call cCreateAndFill(Txt, Fill)

Where :

Txt	the string to proceed
Fill	the chars to fill in the string

Comments :

This routine is a superset of String$. In fact, STRING$ can only use a char to fill a string.

Examples :

```
Txt = space$(14)
Fill = "AbC"
Call cFill(Txt, Fill)
        test = "AbCAbCAbCAbCAb"
```

See also : cCreateAndFill

KillFocus

Purpose :

KillFocus kills and recreates the focus of a gived hWnd

Declare Syntax :

Declare Sub cKillFocus Lib "t2wlight.dll" (ByVal hWnd As Integer)

Call Syntax :

Call cKillFocus(hWnd)

Where :

hWnd	the hWnd of the control

Comments :

PutIni

Purpose :

see Comments

Declare Syntax :

Declare Sub cPutIni Lib "t2wlight.dll" (ByVal AppName As String, ByVal szItem As String, ByVal szDefault As String, ByVal InitFile As String)

Call Syntax :

Call cPutIni(AppName, szItem, szDefault, InitFile)

Where :

AppName	a string that specifies the section to which the string will be copied. If the section does not exist, it is created.
szItem	a string containing the entry to be associated with the string. If the entry does not exist in the specified section, it is created. If this parameter is NULL, the entire section, including all entries within the section, is deleted.
szDefault	a string to be written to the file. If this parameter is NULL, the entry specified by the szItem parameter is deleted.
InitFile	a filename that names the initialization file.

Comments :

To improve performance, Windows keeps a cached version of the most-recently accessed initialization file. If that filename is specified and the other three parameters are NULL, Windows flushes the cache.

Sections in the initialization file have the following form:

```
[section]
entry=string
```

Examples :

Call cPutIni("Desktop","IconTitleFaceName","MS Sans Serif","WIN.INI")

See also : cGetIni

ResetFocus

Purpose :

ResetFocus kills the focus of a gived hWnd and set the focus to an another hWnd.

Declare Syntax :

```
Declare Sub cResetFocus Lib "t2wlight.dll" (ByVal hWnd1 As Integer, ByVal hWnd2 As Integer)
```

Call Syntax :

```
Call cResetFocus(hWnd1, hWnd2)
```

Where :

hWnd1	the hWnd of the control that you want kill the focus.
hWnd2	the hWnd of the control that you want set the focus.

Comments :

ReverseAllBits

Purpose :

ReverseAllBits reverses all bits in a gived string

Declare Syntax :

Declare Sub cReverseAllBits Lib "t2wlight.dll" (Txt As String)

Call Syntax :

Call cReverseAllBits(Txt)

Where :

Txt	the string to proceed

Comments :

See also : Bit String Manipulation routines

ReverseAllBitsByChar

Purpose :

ReverseAllBitsByChar reverses all bits by each char in a gived string

Declare Syntax :

Declare Sub cReverseAllBitsByChar Lib "t2wlight.dll" (Txt As String)

Call Syntax :

Call cReverseAllBitsByChar(Txt)

Where :

Txt	the string to proceed

Comments :

See also : Bit String Manipulation routines

SetAllBits

Purpose :

SetAllBits sets all bits of a gived string to Set state or Reset state.

Declare Syntax :

Declare Sub cSetAllBits Lib "t2wlight.dll" (Txt As String, ByVal Value As Integer)

Call Syntax :

Call cSetAllBits(Txt, Value)

Where :

Txt	the string to proceed
Value	TRUE to Set all bits
	FALSE to Reset all bits

Comments :

See also : Bit String Manipulation routines

SetBit

Purpose :

SetBit sets a gived bit in a gived string to Set state or Reset state.

Declare Syntax :

```
Declare Sub cSetBit Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer, ByVal Value As Integer)
```

Call Syntax :

```
Call cSetBit(Txt, Position, Value)
```

Where :

Txt	the string to proceed
Position	the bit position
Value	TRUE to Set the bit
	FALSE to Reset the bit

Comments :

The first bit in the string is the bit 0.

See also : Bit String Manipulation routines

SetBitToFalse

Purpose :

SetBitToFalse sets a gived bit in a gived string to Reset state.

Declare Syntax :

```
Declare Sub cSetBitToFalse Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer)
```

Call Syntax :

```
Call cSetBitToFalse(Txt, Position)
```

Where :

Txt	the string to proceed
Position	the bit position to Reset

Comments :

The first bit in the string is the bit 0. This routine is a short-cut routine from cSetBit(Txt, Position, FALSE)

See also : Bit String Manipulation routines

SetBitToTrue

Purpose :

SetBitToTrue sets a gived bit in a gived string to Set state.

Declare Syntax :

Declare Sub cSetBitToTrue Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer)

Call Syntax :

Call cSetBitToTrue(Txt, Position)

Where :

Txt	the string to proceed
Position	the bit position to Set

Comments :

The first bit in the string is the bit 0. This routine is a short-cut routine from cSetBit(Txt, Position, TRUE)

See also : Bit String Manipulation routines

FileFilter, FileFilterNot

Purpose :

FileFilter copies one file to an another file but filters some chars.
FileFilterNot copies one file to an another file but filters chars not present in the filter..

Declare Syntax :

Declare Function cFileFilter Lib "t2wlight.dll" (ByVal file1 As String, ByVal file2 As String, Filter As String) As Long
Declare Function cFileFilterNot Lib "t2wlight.dll" (ByVal file1 As String, ByVal file2 As String, Filter As String) As Long

Call Syntax :

test& = cFileFilter(file1, file2, filter)
test& = cFileFilterNot(file1, file2, filternot)

Where :

file1$	is the source file.
file2$	is the destination file.
filter$	is the filter to use to remove chars from the source file.
filternot$	is the filter to use to remove chars not present in the filter from the source file.
test&	> 0 if all is OK (the returned value is the total bytes copied), < 0 if an error has occured.

Comments :

The returned value can be negative and have the following value :

-1	the filter is an EMPTY string.
-32730	reading error for file 1.
-32740	writing error for file 2.
-32750	opening error for file 1.
-32751	opening error for file 2.
-32760	allocation error for memory buffer 1.
-32761	allocation error for memory buffer 2.

Examples :

test& = cFileFilter("c:\autoexec.bat", "c:\autoexec.tab",
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz")
test& = cFileFilterNot("c:\autoexec.bat", "c:\autoexec.tab",
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz")

See also : cFileCopy

SplitPath

Purpose :

SplitPath breaks a full path into its four components.

Declare Syntax :

Declare Function cSplitPath Lib "t2wlight.dll" (ByVal nFilename As String, SPLITPATH As Any) As Integer

Call Syntax :

test% = cSplitPath(nFilename, SPLITPATH)

Where :

nFilename	is the name of a file containing the full path to access it.
SPLITPATH	is the type'd variable to receive the four components.
test%	TRUE if all is OK, FALSE if an error occurs.

Comments :

If the file is not available or if an error occurs when accessing the file, the returned value is always 0.

The four components are :

nDrive	Contains the drive letter followed by a colon (:) if a drive is specified in path.
nDir	Contains the path of subdirectories, if any, including the trailing slash.
nName	Contains the base filename without any extensions.
nExt	Contains the filename extension, if any, including the leading period (.).

The return parameters in SPLITPATH will contain empty strings for any path components not found in path.

Examples :

```
Dim SPLITPATH              As tagSPLITPATH

Call cSplitPath("C:\AUTOEXEC.BAT", SPLITPATH)
```

On my system :

SPLITPATH.nDrive	is "C"
SPLITPATH.nDir	is "\"
SPLITPATH.nName	is "AUTOEXEC"
SPLITPATH.nExt	is ".BAT"

See also : cFullPath, cMakePath, Constants and Types declaration

Revision History

See also : New Features

Version	**Comments**
-	
4.00	Correct a GPF problem with cGetCurrentDrive.
3.52	Initial release of the 'TIME TO WIN Light' dynamic link library for VB 3.0.

New Features

See also : Revision History

Version	**Comments**
4.00	*no new features*
3.52	Initial release of the 'TIME TO WIN Light' dynamic link library.

FileCopy

Purpose :

FileCopy copies one file to an another file.

Declare Syntax :

Declare Function cFileCopy Lib "t2wlight.dll" (ByVal file1 As String, ByVal file2 As String) As Long

Call Syntax :

test& = cFileCopy(file1, file2)

Where :

file1$	is the source file.
file2$	is the destination file.
test&	> 0 if all is OK (the returned value is the total bytes copied), < 0 if an error has occured.

Comments :

The returned value can be negative and have the following value :

-32720	the number of chars in a block for writing differs from the number of chars for reading.
-32730	reading error for file 1.
-32740	writing error for file 2.
-32750	opening error for file 1.
-32751	opening error for file 2.
-32760	allocation error for memory buffer.

Examples :

test& = cFileCopy("c:\autoexec.bat", "c:\autoexec.tab")

See also : cFileFilter, cFileFilterNot, cFileMerge

SetDefaultSeparator

Purpose :

SetDefaultSeparator sets the default char for use the cGet function.

Declare Syntax :

```
Declare Sub cSetDefaultSeparator Lib "t2wlight.dll" (Separator As String)
```

Call Syntax :

```
Call cSetDefaultSeparator(Separator)
```

Where :

Separator	the new separator

Comments :

The default char is '|'.
This char is changed for all applications did use the T2WLIGHT.DLL.
If you must initialize the default, change it only at the starting of your program.

GetSeparatorX

Purpose :

All values returned are readed from the Win.INI file.

GetCountry returns the country name.
GetCountryCode returns the country code.
GetCurrency returns the currency.
GetDateFormat returns the format for the date.
GetDateSeparator returns the separator for the date.
GetHourFormat returns the format for the hour.
GetLanguage returns the letters for the language.
GetListSeparator returns the separator for list.
GetTimeSeparator returns the separator for the date.
GetWinINI returns the information for a gived item (see Constants and Types declaration)

Declare Syntax :

```
Declare Function cGetCountry Lib "t2wlight.dll" () As String
Declare Function cGetCountryCode Lib "t2wlight.dll" () As String
Declare Function cGetCurrency Lib "t2wlight.dll" () As String
Declare Function cGetDateFormat Lib "t2wlight.dll" () As String
Declare Function cGetDateSeparator Lib "t2wlight.dll" () As String
Declare Function cGetHourFormat Lib "t2wlight.dll" () As String
Declare Function cGetLanguage Lib "t2wlight.dll" () As String
Declare Function cGetListSeparator Lib "t2wlight.dll" () As String
Declare Function cGetTimeSeparator Lib "t2wlight.dll" () As String
Declare Function cGetWinINI Lib "t2wlight.dll" (ByVal Info As Integer) As String
```

Call Syntax :

The purpose and the declare syntax are very explicite.

Where :

Info	the number of the desired item GET_TIME_SEPARATOR GET_DATE_SEPARATOR GET_TIME_FORMAT GET_DATE_FORMAT GET_CURRENCY GET_LANGUAGE GET_COUNTRY GET_COUNTRY_CODE GET_LIST_SEPARATOR GET_DEFAULT_PRINTER

Comments :

•The advantage to use these routines is that these routines is very fast and doesn't use the WINDOWS API in VB.

Examples :

GetDateSeparator	is '/'
GetTimeSeparator	is ':'
GetListSeparator	is ';'
GetDateFormat	is 'dd/mm/yyyy'
GetHourFormat	is 'hh:nn'
GetCurrency	is 'FB'
GetLanguage	is 'fra'
GetCountry	is 'Belgium (French)'
GetCountryCode	is '32'

See also : cGetIni

Installation

Demonstration version :

The files T2WLIGHT.DLL and T2WLIGHT.HLP should be copied in your   WINDOWS\SYSTEM directory.

Registered version :

The files T2WLIGHT.DLL, T2WLIGHT.HLP should be copied in your WINDOWS\SYSTEM directory.
The file T2WLIGHT.LIC should be copied in your WINDOWS directory.

Distribution note:

When you create and distribute applications that use 'TIME TO WIN Light' dynamic link library, you should install the file 'T2WLIGHT.DLL' in the customer's Microsoft Windows \SYSTEM subdirectory. The Visual Basic Setup Kit included with the Professional VB product provides tools to help you write setup programs that install you applications correctly.

You are not allowed to distribute '**T2WLIGHT.LIC**' *file with any application that you distribute.*

FileEncrypt, FileDecrypt

Purpose :

FileEncrypt copies one file to an another file but with encryption.
FileDecrypt copies one file to an another file but with decryption.

Declare Syntax :

```
Declare Function cFileEncrypt Lib "t2wlight.dll" (ByVal file1 As String, ByVal file2 As String, Password As String, ByVal Level As Integer) As Long
Declare Function cFileDecrypt Lib "t2wlight.dll" (ByVal file1 As String, ByVal file2 As String, Password As String, ByVal Level As Integer) As Long
```

Call Syntax :

```
test& = cFileEncrypt(file1, file2, password, level)
test& = cFileDecrypt(file1, file2, password, level)
```

Where :

file1$	is the source file.
file2$	is the destination file.
password	is the key to use for encryption/decryption.
level	level of the encryption/decryption.
test&	> 0 if all is OK (the returned value is the total bytes copied), < 0 if an error has occured.

Comments :

The password/key is case sensitive.
The level is a number between **0** and **4** (Constants and Types declaration).
Higher is the level, better is the encryption.
You must use the same level for encrypt/decrypt a gived string.

The returned value can be negative and have the following value :

-1	the password is an EMPTY string.
-32720	the number of chars in a block for writing differs from the number of chars for reading.
-32730	reading error for file 1.
-32740	writing error for file 2.
-32750	opening error for file 1.
-32751	opening error for file 2.
-32760	allocation error for memory buffer 1.
-32761	allocation error for memory buffer 2.

Examples :

```
test& = cFileEncrypt("c:\autoexec.bat", "c:\autoexec.tb1", "Time To Win", ENCRYPT_LEVEL_4)
test& = cFileDecrypt("c:\autoexec.tb1", "c:\autoexec.tb2", "Time To Win", ENCRYPT_LEVEL_4)
```

See also :

ToggleAllBits

Purpose :

ToggleAllBits toggles all bits in a gived string. If a bit is in Set state, it comes in Reset state. If a bit is in Reset state, it comes is Set state.

Declare Syntax :

```
Declare Sub cToggleAllBits Lib "t2wlight.dll" (Txt As String)
```

Call Syntax :

```
Call cToggleAllBits(Txt)
```

Where :

Txt	the string to proceed

Comments :

See also : Bit String Manipulation routines

ToggleBit

Purpose :

ToggleBit toggles a gived bit in a gived string. If a bit is in Set state, it comes in Reset state. If a bit is in Reset state, it comes is Set state.

Declare Syntax :

Declare Sub cToggleBit Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer)

Call Syntax :

Call cToggleBit(Txt, Position)

Where :

Txt	the string to proceed
Position	the bit position

Comments :

The first bit in the string is the bit 0.

See also : Bit String Manipulation routines

CmpFileAttribute, CmpFileContents, CmpFileSize, CmpFileTime

Purpose :

CmpFileAttribute compares the attribute of two files.
CmpFileContents compares the contents of two files.
CmpFileSize compares the size of two files.
CmpFileTime compares the date and time of two files.

Declare Syntax :

Declare Function cCmpFileAttribute Lib "t2wlight.dll" (ByVal file1 As String, ByVal file2 As String) As Integer
Declare Function cCmpFileContents Lib "t2wlight.dll" (ByVal file1 As String, ByVal file2 As String, ByVal sensitivity As Integer) As Integer
Declare Function cCmpFileSize Lib "t2wlight.dll" (ByVal file1 As String, ByVal file2 As String) As Integer
Declare Function cCmpFileTime Lib "t2wlight.dll" (ByVal file1 As String, ByVal file2 As String) As Integer

Call Syntax :

test% = cCmpFileAttribute(file1, file2)
test% = cCmpFileContents(file1, file2, sensitivity)
test% = cCmpFileSize(file1, file2)
test% = cCmpFileTime(file1, file2)

Where :

file1$	is the first file.	
file2$	is the second file.	
sensitivity%	TRUE for case sensitive, FALSE for no case sensitive.	
test%	-1	if file1 < file2 for the specified function,
	0	if file1 = file2 for the specified function,
	1	if file1 > file2 for the specified function.

Comments :

When using cCmpFileAttribute, only -1 (attribute are the same) or 0 (attribute are different) or -2 (error) is returned.
When using cCmpFileContents

-1	files are the same
0	files are not the same, or file size differs
-32740	reading error for files.
-32750	opening error for file 1.
-32751	opening error for file 2.
-32760	allocation error for memory buffer 1.
-32761	allocation error for memory buffer 2.

Examples :

test% = cCmpFileAttribute("c:\command.com", "c:\dos\command.com")
test% = cCmpFileContents("c:\command.com", "c:\dos\command.com", True)
test% = cCmpFileContents("c:\command.com", "c:\dos\command.com", False)
test% = cCmpFileSize("c:\command.com", "c:\dos\command.com")
test% = cCmpFileTime("c:\command.com", "c:\dos\command.com")

See also :

All Functions and Subs

Declare Function cAddD Lib "t2wlight.dll" (array() As Double, ByVal nValue As Double) As Integer
Declare Function cAddDigit Lib "t2wlight.dll" (Txt as string) As integer
Declare Function cAddI Lib "t2wlight.dll" (array() As Integer, ByVal nValue As Integer) As Integer
Declare Function cAddL Lib "t2wlight.dll" (array() As Long, ByVal nValue As Long) As Integer
Declare Function cAddS Lib "t2wlight.dll" (array() As Single, ByVal nValue As Single) As Integer
Declare Function cAddTime Lib "t2wlight.dll" (ByVal Hr As Integer) As Integer
Declare Function cAddTwoTimes Lib "t2wlight.dll" (ByVal Time1 As String, ByVal Time2 As String) As String
Declare Function cAlign Lib "t2wlight.dll" (Txt As String, ByVal TypeAlign As Integer, ByVal NewLength As Integer) As String
Declare Function cAndToken Lib "t2wlight.dll" (ByVal Txt As String, ByVal Token As String) As Integer
Declare Function cAndTokenIn Lib "t2wlight.dll" (ByVal Txt As String, ByVal Token As String, ByVal Separator As String) As Integer
Declare Function cArabicToRoman Lib "t2wlight.dll" (Var As Variant) As String
Declare Sub cArrangeDesktopIcons Lib "t2wlight.dll" ()
Declare Function cArrayOnDisk Lib "t2wlight.dll" (ByVal File As String, Array() As Any, ByVal GetPut As Integer) As Long
Declare Function cArrayPrm Lib "t2wlight.dll" (array() As Any, nArray As Any) As Integer
Declare Function cArrayStringOnDisk Lib "t2wlight.dll" (ByVal File As String, Array() As Any, ByVal GetPut As Integer, rRecords As Long) As Long
Declare Function cBaseConversion Lib "t2wlight.dll" (ByVal Num As String, ByVal RadixIn As Integer, ByVal RadixOut As Integer) As String
Declare Function cBetween Lib "t2wlight.dll" (Var As Variant, Var1 As Variant, Var2 As Variant) As Integer
Declare Function cBlockCharFromLeft Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer) As String
Declare Function cBlockCharFromRight Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer) As String
Declare Sub cChangeChars Lib "t2wlight.dll" (Txt As String, charSet As String, newCharSet As String)
Declare Sub cChangeCharsUntil Lib "t2wlight.dll" (Txt As String, charSet As String, newCharSet As String, nUntil As String)
Declare Sub cChangeTaskName Lib "t2wlight.dll" (ByVal hWnd As Integer, ByVal Text As String)
Declare Function cChDir Lib "t2wlight.dll" (ByVal lpDir As String) As Integer
Declare Function cChDrive Lib "t2wlight.dll" (ByVal lpDrive As String) As Integer
Declare Function cCheckChars Lib "t2wlight.dll" (Txt As String, charSet As String) As Integer
Declare Function cCheckNumericity Lib "t2wlight.dll" (Txt As String) As Integer
Declare Function cCheckTime Lib "t2wlight.dll" (ByVal Hr As Integer, ByVal Hr1 As Integer, ByVal Hr2 As Integer) As Integer
Declare Function cCloseAllEditForm Lib "t2wlight.dll" () As Integer
Declare Function cCmpFileAttribute Lib "t2wlight.dll" (ByVal file1 As String, ByVal file2 As String) As Integer
Declare Function cCmpFileContents Lib "t2wlight.dll" (ByVal file1 As String, ByVal file2 As String, ByVal sensitivity As Integer) As Integer
Declare Function cCmpFileSize Lib "t2wlight.dll" (ByVal file1 As String, ByVal file2 As String) As Integer
Declare Function cCmpFileTime Lib "t2wlight.dll" (ByVal file1 As String, ByVal file2 As String) As Integer
Declare Sub cCnvASCIItoEBCDIC Lib "t2wlight.dll" (Txt As String)
Declare Sub cCnvEBCDICtoASCII Lib "t2wlight.dll" (Txt As String)
Declare Function cCompact Lib "t2wlight.dll" (Txt As String) As String
Declare Function cCompareTypeString Lib "t2wlight.dll" Alias "cTypesCompare" (TypeSrc As Any, ByVal Dst As String, ByVal lenTypeSrc As Integer) As Integer
Declare Function cCompareStringType Lib "t2wlight.dll" Alias "cTypesCompare" (ByVal Src As String, TypeDst As Any, ByVal lenTypeSrc As Integer) As Integer
Declare Function cCompress Lib "t2wlight.dll" (Txt As String) As String
Declare Function cCompressTab Lib "t2wlight.dll" (Txt As String, ByVal nTab As Integer) As String
Declare Function cCount Lib "t2wlight.dll" (Txt As String, Separator As String) As Integer
Declare Function cCountDirectories Lib "t2wlight.dll" (ByVal lpFilename As String) As Integer
Declare Function cCountFiles Lib "t2wlight.dll" (ByVal lpFilename As String) As Integer
Declare Function cCplAlpha Lib "t2wlight.dll" (Txt As String) As String
Declare Function cCplDigit Lib "t2wlight.dll" (Txt As String) As String
Declare Function cCreateAndFill Lib "t2wlight.dll" (ByVal Length As Integer, Txt As String) As String
Declare Function cCreateBits Lib "t2wlight.dll" (ByVal nBits As Integer) As String
Declare Function cCurrentTime Lib "t2wlight.dll" () As Integer
Declare Function cCVB Lib "t2wlight.dll" (Value As String) As Integer
Declare Function cCVC Lib "t2wlight.dll" (Value As String) As Currency
Declare Function cCVD Lib "t2wlight.dll" (Value As String) As Double

Declare Function cCVI Lib "t2wlight.dll" (Value As String) As Integer
Declare Function cCVL Lib "t2wlight.dll" (Value As String) As Long
Declare Function cCVS Lib "t2wlight.dll" (Value As String) As Single
Declare Function cDateToScalar Lib "t2wlight.dll" (ByVal nYear As Integer, ByVal nMonth As Integer, ByVal nDay As Integer) As Long
Declare Function cDayOfWeek Lib "t2wlight.dll" (ByVal nYear As Integer, ByVal nMonth As Integer, ByVal nDay As Integer, ByVal nISO As Integer) As Integer
Declare Function cDayOfYear Lib "t2wlight.dll" (ByVal nYear As Integer, ByVal nMonth As Integer, ByVal nDay As Integer) As Integer
Declare Function cDaysInMonth Lib "t2wlight.dll" (ByVal nYear As Integer, ByVal nMonth As Integer) As Integer
Declare Sub cDecrI Lib "t2wlight.dll" (Value As Integer)
Declare Sub cDecrL Lib "t2wlight.dll" (Value As Long)
Declare Function cDecrypt Lib "t2wlight.dll" (Txt As String, password As String, ByVal level As Integer) As String
Declare Function cDeviationD Lib "t2wlight.dll" (array() As Double) As Double
Declare Function cDeviationI Lib "t2wlight.dll" (array() As Integer) As Double
Declare Function cDeviationL Lib "t2wlight.dll" (array() As Long) As Double
Declare Function cDeviationS Lib "t2wlight.dll" (array() As Single) As Double
Declare Sub cDisableCtlRedraw Lib "t2wlight.dll" (Ctl As Control)
Declare Sub cDisableFI Lib "t2wlight.dll" (Ctl As Control)
Declare Sub cDisableForm Lib "t2wlight.dll" (ByVal hWnd As Integer)
Declare Sub cDisableRedraw Lib "t2wlight.dll" (ByVal hWnd As Integer)
Declare Sub cEnableCtlRedraw Lib "t2wlight.dll" (Ctl As Control)
Declare Sub cEnableFI Lib "t2wlight.dll" (Ctl As Control)
Declare Sub cEnableForm Lib "t2wlight.dll" (ByVal hWnd As Integer)
Declare Sub cEnableRedraw Lib "t2wlight.dll" (ByVal hWnd As Integer)
Declare Function cEncrypt Lib "t2wlight.dll" (Txt As String, password As String, ByVal level As Integer) As String
Declare Function cEXEnameActiveWindow Lib "t2wlight.dll" () As String
Declare Function cEXEnameTask Lib "t2wlight.dll" (ByVal nFileName As String) As String
Declare Function cEXEnameWindow Lib "t2wlight.dll" (ByVal hModule As Integer) As String
Declare Function cExpandTab Lib "t2wlight.dll" (Txt As String, ByVal nTab As Integer) As String
Declare Function cFileCompress Lib "t2wlight.dll" (ByVal file1 As String, ByVal file2 As String) As Long
Declare Function cFileCopy Lib "t2wlight.dll" (ByVal file1 As String, ByVal file2 As String) As Long
Declare Function cFileCRC32 Lib "t2wlight.dll" (ByVal lpFilename As String, ByVal mode As Integer) As Long
Declare Function cFileDateCreated Lib "t2wlight.dll" (ByVal lpFilename As String) As String
Declare Function cFileDecrypt Lib "t2wlight.dll" (ByVal file1 As String, ByVal file2 As String, ByVal password As String, ByVal level As Integer) As Long
Declare Function cFileDrive Lib "t2wlight.dll" (ByVal lpFilename As String) As String
Declare Function cFileEncrypt Lib "t2wlight.dll" (ByVal file1 As String, ByVal file2 As String, ByVal password As String, ByVal level As Integer) As Long
Declare Function cFileExpand Lib "t2wlight.dll" (ByVal file1 As String, ByVal file2 As String) As Long
Declare Function cFileFilter Lib "t2wlight.dll" (ByVal file1 As String, ByVal file2 As String, ByVal Filter As String) As Long
Declare Function cFileFilterNot Lib "t2wlight.dll" (ByVal file1 As String, ByVal file2 As String, ByVal Filter As String) As Long
Declare Function cFileGetAttrib Lib "t2wlight.dll" (ByVal nFilename As String, nFileAttribute As Any) As Integer
Declare Function cFileLastDateAccess Lib "t2wlight.dll" (ByVal lpFilename As String) As String
Declare Function cFileLastDateModified Lib "t2wlight.dll" (ByVal lpFilename As String) As String
Declare Function cFileLastTimeAccess Lib "t2wlight.dll" (ByVal lpFilename As String) As String
Declare Function cFileLastTimeModified Lib "t2wlight.dll" (ByVal lpFilename As String) As String
Declare Function cFileLineCount Lib "t2wlight.dll" (ByVal lpFilename As String) As Integer
Declare Function cFileMerge Lib "t2wlight.dll" (ByVal file1 As String, ByVal file2 As String, ByVal fileTo As String) As Long
Declare Function cFilePathExists Lib "t2wlight.dll" (ByVal lpFilename As String) As Integer
Declare Function cFileResetAllAttrib Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cFileResetArchive Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cFileResetFlag Lib "t2wlight.dll" (ByVal nFilename As String, ByVal nStatus As Integer) As Integer
Declare Function cFileResetHidden Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cFileResetReadOnly Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cFileResetSystem Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cFileSetAllAttrib Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cFileSetArchive Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cFileSetAttrib Lib "t2wlight.dll" (ByVal nFilename As String, nFileAttribute As Any) As Integer

```
Declare Function cFileSetFlag Lib "t2wlight.dll" (ByVal nFilename As String, ByVal nStatus As Integer) As Integer
Declare Function cFileSetHidden Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cFileSetReadOnly Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cFileSetSystem Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cFilesInDirectory Lib "t2wlight.dll" (ByVal nFilename As String, ByVal firstnext As Integer) As String
Declare Function cFileSize Lib "t2wlight.dll" (ByVal lpFilename As String) As Long
Declare Function cFilesSize Lib "t2wlight.dll" (ByVal nFilename As String) As Long
Declare Function cFilesSizeOnDisk Lib "t2wlight.dll" (ByVal nDrive As String, ByVal nFileName As String) As Long
Declare Function cFilesSlack Lib "t2wlight.dll" (ByVal nDrive As String, ByVal nFileName As String, Size1 As Long,
Size2 As Long) As Integer
Declare Function cFileStatistics Lib "t2wlight.dll" (ByVal nFilename As String, nLines As Long, nWords As Long,
nChars As Long) As Long
Declare Function cFileTimeCreated Lib "t2wlight.dll" (ByVal lpFilename As String) As String
Declare Function cFileToComboBox Lib "t2wlight.dll" (ByVal hWnd As Integer, ByVal nFile As String) As Integer
Declare Function cFileToListBox Lib "t2wlight.dll" (ByVal hWnd As Integer, ByVal nFile As String) As Integer
Declare Sub cFill Lib "t2wlight.dll" (Txt As String, Fill As String)
Declare Function cFillD Lib "t2wlight.dll" (array() As Double, ByVal nValue As Double) As Integer
Declare Function cFillI Lib "t2wlight.dll" (array() As Integer, ByVal nValue As Integer) As Integer
Declare Function cFillL Lib "t2wlight.dll" (array() As Long, ByVal nValue As Long) As Integer
Declare Function cFillS Lib "t2wlight.dll" (array() As Single, ByVal nValue As Single) As Integer
Declare Function cFillIncrD Lib "t2wlight.dll" (Array() As Double, ByVal nValue As Double, ByVal Increment As
Double) As Integer
Declare Function cFillIncrI Lib "t2wlight.dll" (Array() As Integer, ByVal nValue As Integer, ByVal Increment As Integer)
As Integer
Declare Function cFillIncrL Lib "t2wlight.dll" (Array() As Long, ByVal nValue As Long, ByVal Increment As Long) As
Integer
Declare Function cFillIncrS Lib "t2wlight.dll" (Array() As Single, ByVal nValue As Single, ByVal Increment As Single)
As Integer
Declare Function cFilterBlocks Lib "t2wlight.dll" (Txt As String, Delimitor As String) As String
Declare Function cFilterChars Lib "t2wlight.dll" (Txt As String, charSet As String) As String
Declare Function cFilterFirstChars Lib "t2wlight.dll" (Txt As String, charSet As String) As String
Declare Function cFilterNotChars Lib "t2wlight.dll" (Txt As String, charSet As String) As String
Declare Function cFindBitReset Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer) As Integer
Declare Function cFindBitSet Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer) As Integer
Declare Function cFloppyInfo Lib "t2wlight.dll" (ByVal nDrive As String, nHeads As Integer, nCylinders As Integer,
nSectors As Integer) As Integer
Declare Function cFromBinary Lib "t2wlight.dll" (Text As String) As String
Declare Function cFromBinary2 Lib "t2wlight.dll" (Text As String, Bin As String) As String
Declare Function cFromHexa Lib "t2wlight.dll" (Text As String) As String
Declare Function cFullPath Lib "t2wlight.dll" (ByVal nFilename As String) As String
Declare Function cGet Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer) As String
Declare Function cGetBit Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer) As Integer
Declare Function cGetBlock Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer, ByVal Length As Integer) As
String
Declare Function cGetCaption Lib "t2wlight.dll" (ByVal hWnd As Integer) As String
Declare Function cGetChangeTaskName Lib "t2wlight.dll" (ByVal hWnd As Integer, ByVal Text As String) As String
Declare Function cGetClass Lib "t2wlight.dll" (ByVal hWnd As Integer) As String
Declare Function cGetClassName Lib "t2wlight.dll" (ByVal hWnd As Integer) As String
Declare Function cGetContainer Lib "t2wlight.dll" (ByVal hWnd As Integer) As String
Declare Function cGetCountry Lib "t2wlight.dll" () As String
Declare Function cGetCountryCode Lib "t2wlight.dll" () As String
Declare Function cGetCtlCaption Lib "t2wlight.dll" (Ctl As Control) As String
Declare Function cGetCtlClass Lib "t2wlight.dll" (Ctl As Control) As String
Declare Function cGetCtlContainer Lib "t2wlight.dll" (Ctl As Control) As String
Declare Function cGetCtlDataField Lib "t2wlight.dll" (Ctl As Control) As String
Declare Function cGetCtlForm Lib "t2wlight.dll" (Ctl As Control) As String
Declare Function cGetCtlIndex Lib "t2wlight.dll" (Ctl As Control) As Integer
Declare Function cGetCtlName Lib "t2wlight.dll" (Ctl As Control) As String
Declare Function cGetCtlNameIndex Lib "t2wlight.dll" (Ctl As Control) As String
Declare Function cGetCtlPropCaption Lib "t2wlight.dll" (Ctl As Control) As Integer
Declare Function cGetCtlPropDataField Lib "t2wlight.dll" (Ctl As Control) As Integer
Declare Function cGetCtlPropText Lib "t2wlight.dll" (Ctl As Control) As Integer
Declare Function cGetCtlTag Lib "t2wlight.dll" (Ctl As Control) As String
Declare Function cGetCtlTagSized Lib "t2wlight.dll" (Ctl As Control) As String
Declare Function cGetCtlText Lib "t2wlight.dll" (Ctl As Control) As String
Declare Function cGetCurrency Lib "t2wlight.dll" () As String
Declare Function cGetCurrentDrive Lib "t2wlight.dll" () As String
Declare Function cGetDataField Lib "t2wlight.dll" (ByVal hWnd As Integer) As String
Declare Function cGetDateFormat Lib "t2wlight.dll" () As String
Declare Function cGetDateSeparator Lib "t2wlight.dll" () As String
Declare Function cGetDefaultCurrentDir Lib "t2wlight.dll" () As String
Declare Function cGetDefaultPrinter Lib "t2wlight.dll" () As String
Declare Function cGetDevices Lib "t2wlight.dll" () As String
Declare Function cGetDiskClusterSize Lib "t2wlight.dll" (ByVal lpDrive As String) As Long
Declare Function cGetDiskFree Lib "t2wlight.dll" (ByVal lpDrive As String) As Long
Declare Function cGetDiskSpace Lib "t2wlight.dll" (ByVal lpDrive As String) As Long
Declare Function cGetDiskUsed Lib "t2wlight.dll" (ByVal lpDrive As String) As Long
Declare Function cGetDriveCurrentDir Lib "t2wlight.dll" (ByVal lpDrive As String) As String
Declare Function cGetDriveType Lib "t2wlight.dll" (ByVal lpDrive As String) As Integer
Declare Function cGetForm Lib "t2wlight.dll" (ByVal hWnd As Integer) As String
Declare Function cGetHourFormat Lib "t2wlight.dll" () As String
Declare Function cGetHwnd Lib "t2wlight.dll" (Ctl As Control) As Integer
Declare Function cGetIn Lib "t2wlight.dll" (Txt As String, Separator As String, ByVal Position As Integer) As String
Declare Function cGetIndex Lib "t2wlight.dll" (ByVal hWnd As Integer) As Integer
Declare Function cGetIni Lib "t2wlight.dll" (ByVal AppName As String, ByVal szItem As String, ByVal szDefault As
String, ByVal InitFile As String) As String
Declare Function cGetInPart Lib "t2wlight.dll" (Txt As String, Separator As String, ByVal Position As Integer) As String
Declare Function cGetInPartR Lib "t2wlight.dll" (Txt As String, Separator As String, ByVal Position As Integer) As
String
Declare Function cGetInR Lib "t2wlight.dll" (Txt As String, Separator As String, ByVal Position As Integer) As String
Declare Function cGetLanguage Lib "t2wlight.dll" () As String
Declare Function cGetListSeparator Lib "t2wlight.dll" () As String
Declare Function cGetName Lib "t2wlight.dll" (ByVal hWnd As Integer) As String
Declare Function cGetNameIndex Lib "t2wlight.dll" (ByVal hWnd As Integer) As String
Declare Function cGetNetConnection Lib "t2wlight.dll" (ByVal lpDrive As String, ErrCode As Integer) As String
Declare Function cGetPrinterPorts Lib "t2wlight.dll" () As String
Declare Function cGetSectionItems Lib "t2wlight.dll" (ByVal Section As String, ByVal InitFile As String, nItems As
Integer) As String
Declare Function cGetSystemDirectory Lib "t2wlight.dll" () As String
Declare Function cGetTaskName Lib "t2wlight.dll" (ByVal hWnd As Integer) As String
Declare Function cGetText Lib "t2wlight.dll" (ByVal hWnd As Integer) As String
Declare Function cGetTimeSeparator Lib "t2wlight.dll" () As String
Declare Function cGetVersion Lib "t2wlight.dll" () As Single
Declare Function cGetWindowsDirectory Lib "t2wlight.dll" () As String
Declare Function cGetWinINI Lib "t2wlight.dll" (ByVal Info As Integer) As String
Declare Function cGetWinSection Lib "t2wlight.dll" (ByVal Section As String) As String
Declare Function cGiveBitPalindrome Lib "t2wlight.dll" () As String
Declare Function cHideAllEditForm Lib "t2wlight.dll" () As Integer
Declare Function cHideDebugForm Lib "t2wlight.dll" () As Integer
Declare Function cHourTo Lib "t2wlight.dll" (Txt As String) As Variant
Declare Sub cIncrI Lib "t2wlight.dll" (Value As Integer)
Declare Sub cIncrL Lib "t2wlight.dll" (Value As Long)
Declare Function cInsertBlocks Lib "t2wlight.dll" (Txt As String, Insert As String) As String
Declare Function cInsertBlocksBy Lib "t2wlight.dll" (Txt As String, Insert As String, Delimitor As String) As String
Declare Function cInsertByMask Lib "t2wlight.dll" (Txt As String, Mask As String, Insert As String) As String
Declare Function cInsertChars Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer, Insert As String) As String
Declare Function cIntoBalance Lib "t2wlight.dll" (Var As Variant) As String
Declare Function cIntoBalanceFill Lib "t2wlight.dll" (Var As Variant) As String
Declare Function cIntoDate Lib "t2wlight.dll" (ByVal nDate As Long) As String
Declare Function cIntoDateFill Lib "t2wlight.dll" (ByVal nDate As Long) As String
Declare Function cIntoDateNull Lib "t2wlight.dll" (ByVal nDate As Long) As String
Declare Function cIntoFixHour Lib "t2wlight.dll" (Var As Variant, ByVal Length As Integer, ByVal fillZero As Integer,
ByVal Hundreds As Integer) As String
Declare Function cIntoHour Lib "t2wlight.dll" (Var As Variant) As String
Declare Function cIntoVarHour Lib "t2wlight.dll" (Var As Variant) As String
Declare Function cIsAlnum Lib "t2wlight.dll" (Txt As String) As Integer
Declare Function cIsAlpha Lib "t2wlight.dll" (Txt As String) As Integer
Declare Function cIsAscii Lib "t2wlight.dll" (Txt As String) As Integer
Declare Function cIsBalance Lib "t2wlight.dll" (ByVal nHour As Long, ByVal nMinute As Integer, ByVal nSecond As
Integer) As Integer
Declare Function cIsBitPalindrome Lib "t2wlight.dll" (Txt As String) As Integer
Declare Function cIsCsym Lib "t2wlight.dll" (Txt As String) As Integer
Declare Function cIsCsymf Lib "t2wlight.dll" (Txt As String) As Integer
Declare Function cIsDate Lib "t2wlight.dll" (ByVal nYear As Integer, ByVal nMonth As Integer, ByVal nDay As Integer)
As Integer
Declare Function cIsDigit Lib "t2wlight.dll" (Txt As String) As Integer
Declare Function cIsFileArchive Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cIsFileFlag Lib "t2wlight.dll" (ByVal nFilename As String, ByVal nStatus As Integer) As Integer
Declare Function cIsFileHidden Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cIsFileNormal Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cIsFilenameValid Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cIsFileReadOnly Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cIsFileSubDir Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cIsFileSystem Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cIsFileVolId Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cIsFormEnabled Lib "t2wlight.dll" (ByVal hWnd As Integer) As Integer
Declare Function cIsHour Lib "t2wlight.dll" (ByVal nHour As Integer, ByVal nMinute As Integer, ByVal nSecond As
Integer) As Integer
Declare Function cIsISBN Lib "t2wlight.dll" (Txt As String) As Integer
Declare Function cIsLeapYear Lib "t2wlight.dll" (ByVal nYear As Integer) As Integer
Declare Function cIsLower Lib "t2wlight.dll" (Txt As String) As Integer
Declare Function cIsPalindrome Lib "t2wlight.dll" (Txt As String) As Integer
Declare Function cIsPunct Lib "t2wlight.dll" (Txt As String) As Integer
Declare Function cIsSpace Lib "t2wlight.dll" (Txt As String) As Integer
Declare Function cIsUpper Lib "t2wlight.dll" (Txt As String) As Integer
Declare Function cIsXdigit Lib "t2wlight.dll" (Txt As String) As Integer
Declare Function cKillDir Lib "t2wlight.dll" (ByVal lpFilename As String) As Integer
Declare Function cKillDirFilesAll Lib "t2wlight.dll" (ByVal lpDir As String, ByVal lpMask As String) As Integer
Declare Function cKillDirs Lib "t2wlight.dll" (ByVal lpDir As String, ByVal HeaderDirectory As Integer) As Integer
Declare Function cKillFile Lib "t2wlight.dll" (ByVal lpFilename As String) As Integer
Declare Function cKillFileAll Lib "t2wlight.dll" (ByVal lpFilename As String) As Integer
Declare Function cKillFiles Lib "t2wlight.dll" (ByVal lpFilename As String) As Integer
Declare Function cKillFilesAll Lib "t2wlight.dll" (ByVal lpFilename As String) As Integer
Declare Function cLrc Lib "t2wlight.dll" (Txt As String) As String
Declare Function cMakeDir Lib "t2wlight.dll" (ByVal lpFilename As String) As Integer
Declare Function cMakeMultipleDir Lib "t2wlight.dll" (ByVal lpFilename As String) As Integer
Declare Function cMakePath Lib "t2wlight.dll" (ByVal nDrive As String, ByVal nDir As String, ByVal nFilename As
String, ByVal Ext As String) As String
Declare Function cMax Lib "t2wlight.dll" (Var1 As Variant, Var2 As Variant) As Variant
Declare Function cMaxD Lib "t2wlight.dll" (array() As Double) As Double
Declare Function cMaxI Lib "t2wlight.dll" (array() As Integer) As Integer
Declare Function cMaxL Lib "t2wlight.dll" (array() As Long) As Long
Declare Function cMaxS Lib "t2wlight.dll" (array() As Single) As Single
Declare Function cMeanD Lib "t2wlight.dll" (array() As Double) As Double
Declare Function cMeanI Lib "t2wlight.dll" (array() As Integer) As Double
Declare Function cMeanL Lib "t2wlight.dll" (array() As Long) As Double
Declare Function cMeanS Lib "t2wlight.dll" (array() As Single) As Double
Declare Function cMin Lib "t2wlight.dll" (Var1 As Variant, Var2 As Variant) As Variant
Declare Function cMinD Lib "t2wlight.dll" (array() As Double) As Double
Declare Function cMinI Lib "t2wlight.dll" (array() As Integer) As Integer
Declare Function cMinL Lib "t2wlight.dll" (array() As Long) As Long
Declare Function cMinS Lib "t2wlight.dll" (array() As Single) As Single
Declare Function cMixChars Lib "t2wlight.dll" (Txt As String) As String
Declare Function cMKB Lib "t2wlight.dll" (ByVal Value As Integer) As String
Declare Function cMKC Lib "t2wlight.dll" (ByVal Value As Currency) As String
Declare Function cMKD Lib "t2wlight.dll" (ByVal Value As Double) As String
Declare Function cMKI Lib "t2wlight.dll" (ByVal Value As Integer) As String
Declare Function cMKL Lib "t2wlight.dll" (ByVal Value As Long) As String
Declare Function cMKN Lib "t2wlight.dll" (ByVal Value As Double) As String
Declare Function cMKS Lib "t2wlight.dll" (ByVal Value As Single) As String
Declare Function cNextHwnd Lib "t2wlight.dll" (ByVal hWnd As Integer) As Integer
Declare Function cNumDigit Lib "t2wlight.dll" (Txt as string) As integer
Declare Function cOneCharFromLeft Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer) As String
Declare Function cOneCharFromRight Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer) As String
Declare Function cOrToken Lib "t2wlight.dll" (ByVal Txt As String, ByVal Token As String) As Integer
Declare Function cOrTokenIn Lib "t2wlight.dll" (ByVal Txt As String, ByVal Token As String, ByVal Separator As String) As Integer
Declare Function cProperName Lib "t2wlight.dll" (Txt As String) As String
Declare Function cProperName2 Lib "t2wlight.dll" (Txt As String, ByVal TokenToUse As String, ByVal Options As Integer) As String
Declare Sub cPutIni Lib "t2wlight.dll" (ByVal AppName As String, ByVal szItem As String, ByVal szDefault As String, ByVal InitFile As String)
Declare Function cRemoveBlockChar Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer, ByVal Length As Integer) As String
Declare Function cRemoveOneChar Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer) As String
Declare Function cRenameFile Lib "t2wlight.dll" (ByVal lpFilename1 As String, ByVal lpFilename2 As String) As Integer
Declare Function cResizeString Lib "t2wlight.dll" (Txt As String, ByVal newLength As Integer) As String
Declare Function cResizeStringAndFill Lib "t2wlight.dll" (Txt As String, ByVal newLength As Integer, Fill As String) As String
Declare Function cReverse Lib "t2wlight.dll" (Txt As String) As String
Declare Sub cReverseAllBits Lib "t2wlight.dll" (Txt As String)
Declare Sub cReverseAllBitsByChar Lib "t2wlight.dll" (Txt As String)
Declare Function cReverseSortD Lib "t2wlight.dll" (array() As Double) As Integer
Declare Function cReverseSortI Lib "t2wlight.dll" (array() As Integer) As Integer
Declare Function cReverseSortL Lib "t2wlight.dll" (array() As Long) As Integer
Declare Function cReverseSortS Lib "t2wlight.dll" (array() As Single) As Integer
Declare Function cReverseSortStr Lib "t2wlight.dll" (Txt As String, ByVal nItem As Integer, ByVal ItemLength As Integer) As Integer
Declare Function cRomanToArabic Lib "t2wlight.dll" (Txt As String) As Variant
Declare Sub cScalarToDate Lib "t2wlight.dll" (ByVal Scalar As Long, nYear As Integer, nMonth As Integer, nDay As Integer)
Declare Function cScrollL Lib "t2wlight.dll" (Txt As String) As String
Declare Function cScrollR Lib "t2wlight.dll" (Txt As String) As String
Declare Sub cSetAllBits Lib "t2wlight.dll" (Txt As String, ByVal Value As Integer)
Declare Sub cSetBit Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer, ByVal Value As Integer)
Declare Sub cSetBitToFalse Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer)
Declare Sub cSetBitToTrue Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer)
Declare Sub cSetCaption Lib "t2wlight.dll" (ByVal hWnd As Integer, ByVal Text As String)
Declare Sub cSetCtlCaption Lib "t2wlight.dll" (Ctl As Control, ByVal Text As String)
Declare Sub cSetCtlDataField Lib "t2wlight.dll" (Ctl As Control, ByVal Text As String)
Declare Sub cSetCtlPropString Lib "t2wlight.dll" (Ctl As Control, ByVal PropIndex As Integer, ByVal Text As String)
Declare Sub cSetCtlTag Lib "t2wlight.dll" (Ctl As Control, ByVal Text As String)
Declare Sub cSetCtlText Lib "t2wlight.dll" (Ctl As Control, ByVal Text As String)
Declare Function cSetD Lib "t2wlight.dll" (array() As Double, ByVal nValue As Double) As Integer
Declare Sub cSetDataField Lib "t2wlight.dll" (ByVal hWnd As Integer, ByVal Text As String)
Declare Sub cSetDefaultSeparator Lib "t2wlight.dll" (Separator As String)
Declare Function cSetI Lib "t2wlight.dll" (array() As Integer, ByVal nValue As Integer) As Integer
Declare Function cSetL Lib "t2wlight.dll" (array() As Long, ByVal nValue As Long) As Integer
Declare Function cSetS Lib "t2wlight.dll" (array() As Single, ByVal nValue As Single) As Integer
Declare Sub cSetTag Lib "t2wlight.dll" (ByVal hWnd As Integer, ByVal Text As String)
Declare Sub cSetText Lib "t2wlight.dll" (ByVal hWnd As Integer, ByVal Text As String)
Declare Function cSleep Lib "t2wlight.dll" (ByVal Delay As Long) As Integer
Declare Function cSortD Lib "t2wlight.dll" (array() As Double) As Integer
Declare Function cSortI Lib "t2wlight.dll" (array() As Integer) As Integer
Declare Function cSortL Lib "t2wlight.dll" (array() As Long) As Integer
Declare Function cSortS Lib "t2wlight.dll" (array() As Single) As Integer
Declare Function cSortStr Lib "t2wlight.dll" (Txt As String, ByVal nItem As Integer, ByVal ItemLength As Integer) As Integer
Declare Sub cSplitPath Lib "t2wlight.dll" (ByVal nFilename As String, SPLITPATH As Any)
Declare Function cStringCompress Lib "t2wlight.dll" (Txt As String) As String
Declare Function cStringCRC32 Lib "t2wlight.dll" (Txt As String) As Long
Declare Function cStringExpand Lib "t2wlight.dll" (Txt As String) As String
Declare Sub cStringToType Lib "t2wlight.dll" Alias "cTypesCopy" (ByVal Src As String, TypeDst As Any, ByVal lenTypeSrc As Integer)
Declare Function cSubDirectory Lib "t2wlight.dll" (ByVal nFilename As String, ByVal firstnext As Integer) As String
Declare Function cSumD Lib "t2wlight.dll" (array() As Double) As Double
Declare Function cSumI Lib "t2wlight.dll" (array() As Integer) As Double
Declare Function cSumL Lib "t2wlight.dll" (array() As Long) As Double
Declare Function cSumS Lib "t2wlight.dll" (array() As Single) As Double
Declare Sub cSwapD Lib "t2wlight.dll" (swap1 As Double, swap2 As Double)
Declare Sub cSwapI Lib "t2wlight.dll" (swap1 As Integer, swap2 As Integer)
Declare Sub cSwapL Lib "t2wlight.dll" (swap1 As Long, swap2 As Long)
Declare Sub cSwapS Lib "t2wlight.dll" (swap1 As Single, swap2 As Single)
Declare Sub cSwapStr Lib "t2wlight.dll" (swap1 As String, swap2 As String)
Declare Function cTimeBetween Lib "t2wlight.dll" (ByVal Hr1 As Integer, ByVal Hr2 As Integer) As Integer
Declare Function cToBinary Lib "t2wlight.dll" (Text As String) As String
Declare Function cToBinary2 Lib "t2wlight.dll" (Text As String, Bin As String) As String
Declare Sub cToggleAllBits Lib "t2wlight.dll" (Txt As String)
Declare Sub cToggleBit Lib "t2wlight.dll" (Txt As String, ByVal Position As Integer)
Declare Function cToHexa Lib "t2wlight.dll" (Text As String) As String
Declare Function cTokenIn Lib "t2wlight.dll" (Txt As String, Separator As String, ByVal Position As Integer) As String
Declare Function cTrueBetween Lib "t2wlight.dll" (Var As Variant, Var1 As Variant, Var2 As Variant) As Integer
Declare Sub cTypeClear Lib "t2wlight.dll" (TypeSrc As Any, ByVal lenTypeSrc As Integer)
Declare Function cTypeMid Lib "t2wlight.dll" (TypeSrc As Any, ByVal Offset As Integer, ByVal Length As Integer) As String
Declare Function cTypesCompare Lib "t2wlight.dll" (Type1 As Any, Type2 As Any, ByVal lenType1 As Integer) As Integer
Declare Sub cTypesCopy Lib "t2wlight.dll" (TypeSrc As Any, TypeDst As Any, ByVal lenTypeSrc As Integer)
Declare Function cTypeTransfert Lib "t2wlight.dll" (TypeSrc As Any, ByVal lenTypeSrc As Integer) As String
Declare Sub cTypeToString Lib "t2wlight.dll" Alias "cTypesCopy" (TypeSrc As Any, ByVal Dst As String, ByVal lenTypeSrc As Integer)
Declare Function cUncompact Lib "t2wlight.dll" (Txt As String) As String
Declare Function cUniqueFileName Lib "t2wlight.dll" (Txt As String) As String
Declare Function cUnHideAllEditForm Lib "t2wlight.dll" () As Integer
Declare Function cUnHideDebugForm Lib "t2wlight.dll" () As Integer
Declare Function cWeekOfYear Lib "t2wlight.dll" (ByVal nYear As Integer, ByVal nMonth As Integer, ByVal nDay As Integer, ByVal nISO As Integer) As Integer
```

Array routines

Put/Get full array on/from disk

cArrayOnDisk cArrayStringOnDisk

Adding a value to all elements in a single array

cAddD cAddI cAddL cAddS

Read the configuration of a single array

cArrayPrm

Calculating the standard deviation from all elements in a single array

cDeviationD cDeviationI cDeviationL cDeviationS

Filling on all elements on a single array with a value incremented by one for any element

cFillD cFillI cFillL cFillS

Filling on all elements on a single array with a value incremented by an increment for any element

cFillIncrD cFillIncrI cFillIncrL cFillIncrS

Finding the maximum value in a single array

cMaxD cMaxI cMaxL cMaxS

Calculating the mean from all elements in a single array

cMeanD cMeanI cMeanL cMeanS

Finding the minimum value in a single array

cMinD cMinI cMinL cMinS

Sort a single array in descending order

cReverseSortD cReverseSortI cReverseSortL cReverseSortS cReverseSortStr

Setting all elements in a single array with the same value

cSetD cSetI cSetL cSetS

Sort a single array in ascending order

cSortD cSortI cSortL cSortS cSortStr

Add all elements from a single array

cSumD cSumI cSumL cSumS

Bit String Manipulation routines

All strings used in these functions can be have embedded chr$(0) (if needed). These functions use the full description of a VB string.

DOS routines

cIsFileArchive
cIsFileFlag
cIsFileHidden
cIsFileNormal
cIsFileReadOnly
cIsFileSubDir
cIsFileSystem
cIsFileVolId
cKillDir
cKillDirFilesAll
cKillDirs
cKillFile
cKillFileAll
cKillFiles
cKillFilesAll
cMakeDir
cMakeMultipleDir
cMakePath
cRenameFile
cSplitPath
cSubDirectory
cUniqueFileName

IsX Family Test routines

String Manipulation routines

All strings used in these functions can be have embedded chr$(0) (if needed). These functions use the full description of a VB string.

cToBinary2
cToHexa
cTokenIn
cUncompact

Type functions

cCompareStringType
cCompareTypeString
cStringToType
cTypeClear
cTypeMid
cTypesCompare
cTypesCopy
cTypeToString
cTypeTransfert

VB Control Specific routines

cCloseAllEditForm
cDisableCtlRedraw
cDisableFI
cDisableForm
cDisableRedraw
cEnableCtlRedraw
cEnableFI
cEnableForm
cEnableRedraw
cGetCaption
cGetClass
cGetContainer
cGetCtlCaption
cGetCtlClass
cGetCtlContainer
cGetCtlDataField
cGetCtlForm
cGetCtlIndex
cGetCtlName
cGetCtlNameIndex
cGetCtlPropCaption
cGetCtlPropDataField
cGetCtlPropText
cGetCtlTag
cGetCtlTagSized
cGetCtlText
cGetDataField
cGetForm
cGetHwnd
cGetIndex
cGetName
cGetNameIndex
cGetText
cHideAllEditForm
cHideDebugForm
cKillFocus
cResetCapture
cResetFocus
cSetCaption
cSetCapture
cSetCtlCaption
cSetCtlDataField
cSetCtlFocus
cSetCtlPropString
cSetCtlTag
cSetCtlText
cSetDataField
cSetFocus
cSetTag
cSetText
cHideAllEditForm
cUnHideDebugForm

Windows Specific routines

Constants and Types declaration

```
Option Explicit

' definition for win.ini section
Global Const GET_TIME_SEPARATOR = 1
Global Const GET_DATE_SEPARATOR = 2
Global Const GET_TIME_FORMAT = 3
Global Const GET_DATE_FORMAT = 4
Global Const GET_CURRENCY = 5
Global Const GET_LANGUAGE = 6
Global Const GET_COUNTRY = 7
Global Const GET_COUNTRY_CODE = 8
Global Const GET_LIST_SEPARATOR = 9
Global Const GET_DEFAULT_PRINTER = 10

' definition for drive type
Global Const DRIVE_UNKNOW = 0
Global Const DRIVE_REMOVABLE = 2
Global Const DRIVE_FIXED = 3
Global Const DRIVE_REMOTE = 4
Global Const DRIVE_CDROM = 20

' definition for file attributes
Global Const A_NORMAL = &H0                'Normal file - No read/write restrictions
Global Const A_RDONLY = &H1                'Read only file
Global Const A_HIDDEN = &H2                'Hidden file
Global Const A_SYSTEM = &H4                'System file
Global Const A_VOLID = &H8                 'Volume ID file
Global Const A_SUBDIR = &H10               'Subdirectory
Global Const A_ARCH = &H20                 'Archive file

' definition for encrypt/decrypt
Global Const ENCRYPT_LEVEL_0 = 0
Global Const ENCRYPT_LEVEL_1 = 1
Global Const ENCRYPT_LEVEL_2 = 2
Global Const ENCRYPT_LEVEL_3 = 3
Global Const ENCRYPT_LEVEL_4 = 4

' definition for FILECRC32
Global Const OPEN_MODE_BINARY = 0
Global Const OPEN_MODE_TEXT = 1

' definition for ARRAYONDISK
Global Const PUT_ARRAY_ON_DISK = 0
Global Const GET_ARRAY_ON_DISK = 1

' definition for properties for language management
Global Const RS_CAPTION = 1
Global Const RS_TEXT = 2
Global Const RS_DATAFIELD = 4
Global Const RS_DATASOURCE = 8
Global Const RS_TAG = 16

' definition for error type for ISFILENAMEVALID
Global Const IFV_ERROR = 0
Global Const IFV_NAME_TOO_LONG = 1
Global Const IFV_EXT_TOO_LONG = 2
Global Const IFV_TOO_MANY_BACKSLASH = 3
Global Const IFV_BAD_DRIVE_LETTER = 4
Global Const IFV_BAD_COLON_POS = 5
Global Const IFV_EXT_WITHOUT_NAME = 6
```

```
' definition for variable type in DISK ARRAY
Global Const DA_BYTE = 1
Global Const DA_TYPE = 0
Global Const DA_INTEGER = -2
Global Const DA_LONG = -3
Global Const DA_SINGLE = -4
Global Const DA_DOUBLE = -5
Global Const DA_CURRENCY = -6

' definition for compress/expand
Global Const LZH_ENCODE = True
Global Const LZH_DECODE = False

' definition for PROPERNAME2
Global Const PN_UPPERCASE = 1
Global Const PN_PUNCTUATION = 2
Global Const PN_KEEP_ORIGINAL = 4
Global Const PN_ONLY_LEADING_SPACE = 8

' structure for splittin path
Type tagSPLITPATH
        nDrive              As String
        nDir                As String
        nName               As String
        nExt                As String
End Type

' structure for file attributes
Type FileAttributeType
        ErrNo               As Integer
        Archive             As Integer
        Hidden              As Integer
        Normal              As Integer
        ReadOnly            As Integer
        SubDir              As Integer
        System              As Integer
        VolId               As Integer
End Type

' structure for VB array
Type ArrayType
        Bounds              As Long
        LBound              As Integer
        UBound              As Integer
        ElemSize            As Integer
        IndexCount          As Integer
        TotalElem           As Integer
End Type

' structure for ARRAYSTRINGONDISK
Type tagVARSTRING
        Contents            As String
End Type
```

EXEnameActiveWindow

Purpose :

EXEnameActiveWindow retrieves the full filename (path and file) of the active window.

Declare Syntax :

Declare Function cEXEnameActiveWindow Lib "t2wlight.dll" () As String

Call Syntax :

test$ = cEXEnameActiveWindow()

Where :

test$	is the name of the active window

Comments :

Examples :

test$ = cEXEnameActiveWindow()

On my system : test$ = "K:\WINDOWS\VB\VB.EXE"

See also : cEXEnameTask, cEXEnameWindow

EXEnameWindow

Purpose :

EXEnameActiveWindow retrieves the full filename (path and file) of the specified window.

Declare Syntax :

Declare Function cEXEnameWindow Lib "t2wlight.dll" (ByVal hModule As Integer) As String

Call Syntax :

test$ = cEXEnameWindow(Form.Hwnd)

Where :

hModule	is the hWnd of the window
test$	is the name of the specified window

Comments :

Examples :

test$ = cEXEnameWindow(Me.hWnd)

On my system : test$ = "K:\WINDOWS\VB\VB.EXE"

See also : cEXEnameTask, cEXEnameActiveWindow

EXEnameTask

Purpose :

The EXEnameTask function retrieves the full path and filename of the executable file from which the specified module was loaded.

Declare Syntax :

Declare Function cEXEnameTask Lib "t2wlight.dll" (ByVal nFileName As String) As String

Call Syntax :

test$ = cEXEnameTask(nFileName)

Where :

nFileName	is the task name as you fin when pressing CTRL + ESC keys
test$	is the returned full path and filename

Comments :

Examples :

test$ = cEXEnameTask("PROGMAN")

On my system : test$ = "K:\WINDOWS\PROGMAN.EXE"

See also : cEXEnameWindow, cEXEnameActiveWindow

Align

Purpose :

Align aligns a give string (left, center, right) into an another new string.

Declare Syntax :

```
Declare Function cAlign Lib "t2wlight.dll" (Txt As String, ByVal TypeAlign As Integer, ByVal NewLength As Integer) As String
```

Call Syntax :

```
Test$ = cAlign(Txt$, TypeAlign%, NewLength%)
```

Where :

Txt$	is the specified string
TypeAlign%	< 0 : left align, = 0 : center align, > 0 : right align.
NewLength%	the length of the new string
Test$	is the string aligned

Comments :

If NewLength is below that the length of the string, the left part of the string is returned.
The new string is padded with spaces.

Examples :

```
Test$ = cAlign("TIME TO WIN", -1, 20)
        -> "TIME TO WIN         "

Test$ = cAlign("TIME TO WIN", 0, 20)
        -> "    TIME TO WIN     "

Test$ = cAlign("TIME TO WIN", 1, 20)
        -> "         TIME TO WIN"
```

See also :

Date, Hour and Time routines

IEEE Conversion routines

cCVB
cCVC
cCVD
cCVI
cCVL
cCVS

cMKB
cMKC
cMKD
cMKI
cMKL
cMKN
cMKS

Miscellaneous routines

cAddDigit
cBaseConversion
cBetween
cCplAlpha
cCplDigit
cCurrentTime
cFileCRC32
cGetVersion
cLrc
cMax
cMin
cNumDigit
cStringCRC32
cSwapD
cSwapI
cSwapL
cSwapS
cSwapStr
cTrueBetween

Technical Support

Only registered users can receive support and update.

To receive support, you must specify your registration ID.

However, any report on any problem are the welcome.

The following information may be of help to you in streamlining your efforts to resolve any technical problems you may have with 'TIME TO WIN Light' dynamic link library for Visual Basic® 3.0 for Windows®.

GPF?

If you are getting a GPF (General Protection Fault), write down the information that is displayed when the error occurs. Also, make a note of what your code was doing (in general terms.)

ISOLATE IT

Try to isolate the cause of the error. If at all possible, step through your code with F8 and F9. Try to find the one line of code that is causing the error.

SCALE IT DOWN

If at all possible, try to reproduce the problem in a small test program that you can send in. Send your test on CompuServe.

CompuServe Mail:

Name : Michaël RENARD
CIS : 100042,3646

I'm on CompuServe one time a day.

License Agreement

Acknowledgement

Thanks to Andreas Thoele for some translations in German language.
Thanks to Silvio Sorrentino for some translations in Italian language.
Thanks to Manuel Tobarra Narro for some translations in Spanish language.

Special thanks to J. Kercheval, Michael M. Dodd, Ray Gardner, Bob Stout.
Special thanks to Brian Pirie for REGISTRATION KEY SYSTEM FOR C PROGRAMMERS.
Special thanks to Andy Brown for MD5 HASH ALGORITHM. (derived from the RSA ** ** Data Security, Inc. MD5 Message-Digest Algorithm).

This help has been writed by using ForeHelp v1.04 from ForeFront, Inc.

Overview

'TIME TO WIN Light' is a DLL (**D**ynamic **L**ink **L**ibrary) only for use with Visual Basic® 3.0 for Windows®.

I've writed 'TIME TO WIN Light' to help any users of VB to find a solution at some missing functions in VB.
VB is a powerfull product but by some aspects it is very limited.

I hope that 'TIME TO WIN Light' will be a great advantage for you and for your application.

'TIME TO WIN Light' contains more over **360** functions or subroutines. You can find functions or routines over the following sections :

• Array routines
• Bit String Manipulation routines
• Date, Hour and Time routines
• DOS, Disk and Files routines
• IEEE Conversion routines
• IsX Family Test routines
• Miscellaneous routines
• String Manipulation routines
• Type functions
• VB Control Specific routines
• Windows Specific routines

Registering 'TIME TO WIN Light'

The easiest way to Register 'TIME TO WIN Light' is through CompuServe's SWREG forum.

1) GO SWREG
2) Choose Register Shareware.
3) 'TIME TO WIN Light' SWREG ID is : #**5808**.

As soon as I receive notification of your registration (usually 1 - 3 days) I will send you out via e-Mail the latest version and a license file for one site (only if lastest version is available (not currently in test)) if not you receive the license file for one site.

You also qualify to receive new versions of 'TIME TO WIN Light' during one year.

The price for 'TIME TO WIN Light' is fixed at $25.00

This price is much a contribution to my works that a payment. When you register 'TIME TO WIN Light', you help me to develop better products and others products.

'TIME TO WIN Light' is written in C and has been compiled using Visual C++ 1.51.
The code has been optimized for 80386 use with the 'maximize speed' option.

'TIME TO WIN Light' can only be used with Visual Basic 3.0.

Others products :

1) VB/Error Handler : Add/Remove error's management into a VB application by treatment of all files (.FRM, .BAS, .INC) in the .MAK project.

1.1) VB/Error Handler for ONLY REGISTERED USER of 'TIME TO WIN Light' is $20.00. SWREG ID is : #**4379**.
1.2) VB/Error Handler for UN-REGISTERED USER of 'TIME TO WIN Light' is $30.00. SWREG ID is : #**4380**.

2) VB/Tracer-Profiler : Add/Remove trace/profile information into VB application by treatment of all files (.FRM, .BAS, .INC) in the .MAK project.

2.1) VB/Tracer-Profiler for ONLY REGISTERED USER of 'TIME TO WIN Light' is $25.00. SWREG ID is : #**5295**.
2.2) VB/Tracer-Profiler UN-REGISTERED USER of 'TIME TO WIN Light' is $34.00. SWREG ID is : #**5294**.

3) Bundle of TIME TO WIN (full version), VB/Error Handler, VB/Tracer-Profiler

All the three products for the INCREDIBLE price of $99.00. SWREG ID is : #**5499**.

SwapD

Purpose :

SwapD swaps two Double values.

Declare Syntax :

Declare Sub cSwapD Lib "t2wlight.dll" (swap1 As Double, swap2 As Double)

Call Syntax :

Call cSwapD(swap1, swap2)

Where :

swap1	first Double value
swap2	second Double value

Comments :

Examples :

```
swap1 = 2345.12
swap2 = 5432.21
Call cSwapD(swap1, swap2
        -> swap1 = 5432.21
        -> swap2 = 2345.12
```

See Also : cSwapD, cSwapI, cSwapL, cSwapS, cSwapStr

SwapL

Purpose :

SwapL swaps two Long values.

Declare Syntax :

Declare Sub cSwapL Lib "t2wlight.dll" (swap1 As Long, swap2 As Long)

Call Syntax :

Call cSwapL(swap1, swap2)

Where :

swap1	first Long value
swap2	second Long value

Comments :

Examples :

```
swap1 = 234512
swap2 = 543221
Call cSwapL(swap1, swap2
        -> swap1 = 543221
        -> swap2 = 234512
```

See Also : cSwapD, cSwapI, cSwapL, cSwapS, cSwapStr

SwapI

Purpose :

SwapI swaps two Integer values.

Declare Syntax :

Declare Sub cSwapI Lib "t2wlight.dll" (swap1 As Integer, swap2 As Integer)

Call Syntax :

Call cSwapI(swap1, swap2)

Where :

swap1	first Integer value
swap2	second Integer value

Comments :

Examples :

```
swap1 = 2345
swap2 = 5432
Call cSwapI(swap1, swap2
        -> swap1 = 5432
        -> swap2 = 2345
```

See Also : cSwapD, cSwapI, cSwapL, cSwapS, cSwapStr

SwapS

Purpose :

SwapS swaps two Single values.

Declare Syntax :

Declare Sub cSwapS Lib "t2wlight.dll" (swap1 As Single, swap2 As Single)

Call Syntax :

Call cSwapS(swap1, swap2)

Where :

swap1	first Single value
swap2	second Single value

Comments :

Examples :

```
swap1 = 2345.1
swap2 = 5432.2
Call cSwapS(swap1, swap2
            -> swap1 = 5432.2
            -> swap2 = 2345.1
```

See Also : cSwapD, cSwapI, cSwapL, cSwapS, cSwapStr

SwapStr

Purpose :

SwapStr swaps two Strings.

Declare Syntax :

Declare Sub cSwapStr Lib "t2wlight.dll" (swap1 As String, swap2 As String)

Call Syntax :

Call cSwapStr(swap1, swap2)

Where :

swap1	first String
swap2	second String

Comments :

Examples :

```
swap1 = "Hello"
swap2 = "World"
Call cSwapStr(swap1, swap2
        -> swap1 = "World"
        -> swap2 = "Hello"
```

See Also : cSwapD, cSwapI, cSwapL, cSwapS, cSwapStr

FileSet

Purpose :

FileSetAllAttrib, FileSetArchive, FileSetHidden, FileSetReadOnly, FileSetSystem, FileSetFlag sets respectively all attributes, archive attribute, hidden attribute, read-only attribute, system attribute, specified attribute for the gived file.
FileSetAttrib sets in a Call, all attributes of a gived file.

Declare Syntax :

```
Declare Function cFileSetAllAttrib Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cFileSetArchive Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cFileSetHidden Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cFileSetReadOnly Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cFileSetSystem Lib "t2wlight.dll" (ByVal nFilename As String) As Integer
Declare Function cFileSetFlag Lib "t2wlight.dll" (ByVal nFilename As String, ByVal nStatus As Integer) As Integer

Declare Function cFileSetAttrib Lib "t2wlight.dll" (ByVal nFilename As String, nFileAttribute As Any) As Integer
```

Call Syntax :

```
status = cFileSetAllAttrib(nFilename)
status = cFileSetArchive(nFilename)
status = cFileSetHidden(nFilename)
status = cFileSetReadOnly(nFilename)
status = cFileSetSystem(nFilename)
status = cFileSetFlag(nFilename, nStatus)

test% = cFileSetAttrib(nFilename, nFileAttribute)
```

Where :

nFilename	is the filename to change the attributes
nStatus	is a combination of A_NORMAL, A_RDONLY, A_HIDDEN, A_SYSTEM, A_ARCH
nFileAttribute	the type variable 'FileAttributeType' (only for cFileSetAttrib)
status	TRUE if all is OK.
	FALSE if an error has been detected.

Comments :

Examples :

```
nFilename = "tmp.tmp"
nStatus = A_RDONLY or A_SYSTEM or A_HIDDEN

status = cFileSetAllAttrib(nFilename)
status = cFileSetFlag(nFilename, nStatus)
```

See also : FileReset, Constants and Types declaration

KillDirFilesAll

Purpose :

KillDirFilesAll deletes all files specified by a mask in the specified directory and its associated sub-dir.

Declare Syntax :

Declare Function cKillDirFilesAll Lib "t2wlight.dll" (ByVal lpDir As String, ByVal lpMask As String) As Integer

Call Syntax :

test% = cKillDirFilesAll(lpDir$, lpMask$)

Where :

lpDi$r	is the starting directory
lpMask$	is the file mask to use
test%	>= 0 if all is OK. The returned value specified the total files deleted, < 0 if an error has occured

Comments :

Don't forget that this function can handle a maximum of 700 directories of 70 chars long each.

This function doesn't generates an VB Error if the speficied dir not exists.

The returned value can be negative :

-32760	allocation error for memory buffer.

See also : cKillFile, cKillFiles, cKillDir, cKillDirs

BaseConversion

Purpose :

BaseConversion converts a number string (long integer) from a radix to another radix.

Declare Syntax :

Declare Function cBaseConversion Lib "t2wlight.dll" (ByVal Num As String, ByVal RadixIn As Integer, ByVal RadixOut As Integer) As String

Call Syntax :

test$ = cBaseConversion(Num$, RadixIn%, RadixOut%)

Where :

Num$	is the number string to convert
RadixIn%	is the base of the radix
RadixOut%	is the new base of the radix
test$	is the result

Comments :

If the number string can be converted, the returned string is an EMPTY string.

Examples :

Convert '1234567' base 10 to base 2 is 100101101011010000111
Convert '1234567' base 10 to base 3 is 2022201111201
Convert '1234567' base 10 to base 4 is 10231122013
Convert '1234567' base 10 to base 5 is 304001232
Convert '1234567' base 10 to base 6 is 42243331
Convert '1234567' base 10 to base 7 is 13331215
Convert '1234567' base 10 to base 8 is 4553207
Convert '1234567' base 10 to base 9 is 2281451
Convert '1234567' base 10 to base 10 is 1234567
Convert '1234567' base 10 to base 11 is 773604
Convert '1234567' base 10 to base 12 is 4b6547
Convert '1234567' base 10 to base 13 is 342c19
Convert '1234567' base 10 to base 14 is 241cb5
Convert '1234567' base 10 to base 15 is 195be7
Convert '1234567' base 10 to base 16 is 12d687
Convert '1234567' base 10 to base 17 is ed4ea
Convert '1234567' base 10 to base 18 is bdc71
Convert '1234567' base 10 to base 19 is 98ig4
Convert '1234567' base 10 to base 20 is 7e687

See also :

FileStatistics

Purpose :

FileStatictics counts the lines, words and chars in a specified file.

Declare Syntax :

Declare Function cFileStatistics Lib "t2wlight.dll" (ByVal nFilename As String, nLines As Long, nWords As Long, nChars As Long) As Long

Call Syntax :

test& = cFileStatictics(nFilename$, nLines, nWords, nChars)

Where :

nFilename$	is the file to proceed
nLines&	is the returned number of lines
nWords&	is the returned number of words
nChars&	is the returned number of chars
test&	> 0 if all is OK (the returned value is the total bytes in the file), < 0 if an error has occured.

Comments :

If all is ok, the returned value must be equal to nChars.

The returned value can be negative and have the following value :

-32730	reading error for file.
-32750	opening error for file.
-32760	allocation error for memory buffer.

Examples :

test& = cFileStatistics("c:\autoexec.bat", nLines&, nWords&, nChars&)

On my system :

nLines&	is 90
nWords&	is 282
nChars&	is 2212
test&	is 2212

test& = cFileStatistics("c:\config.sys", nLines&, nWords&, nChars&)

On my system :

nLines&	is 15
nWords&	is 44
nChars&	is 506
test&	is 506

See also :

Need assistance for some translations in different languages

Actually, 'TIME TO WIN Light' supports 6 languages :

French	
Dutch	
English	
German	translated by Andreas Thoele.
Italian	translated by Silvio Sorrentino.
Spanish	translated by Manuel Tobarra Narro.

If you're fluent in an another language, can you translate the following texts that I can include in my product :

long month	: "January","February","March","April","May","June","July","August","September","October","November","December"
short month	: "Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec"
tiny month	: "J","F","M","A","M","J","J","A","S","O","N","D"
long day	: "Sunday","Monday","Tuesday","Wednesday","Thursday","Friday","Saturday"
short day	: "Sun","Mon","Tue","Wed","Thu","Fri","Sat"
small day	: "Su","Mo","Tu","We","Th","Fr","Sa"
tiny day	: "S","M","T","W","T","F","S"
system menu	: "&Restore","&Move","&Size","Mi&nimize","Ma&ximize","&Close\tAlt+F4","S&witch To...\tCtrl+Esc"
message box	: "&Move","&Close\tAlt+F4","OK","Cancel","&Abort","&Retry","&Ignore","&Yes","&No"

Thanks you for any translation.

You can post any translations on CompuServe :

Name : **Michaël RENARD**
CIS : **100042,3646**

ScrollL, ScrollR

Purpose :

ScrollL scrolls one char to the left of a specified string.
ScrollR scrolls one char to the right of a specified string.

Declare Syntax :

```
Declare Function cScrollL Lib "t2wlight.dll" (Txt As String) As String
Declare Function cScrollR Lib "t2wlight.dll" (Txt As String) As String
```

Call Syntax :

```
test$ = cScrollL(Txt$)
test$ = cScrollR(Txt$)
```

Where :

Txt$	is the string to scroll.
test$	is the string scrolled to the left or to the right.

Comments :

The size of the string must be greater than 1.

Examples :

```
Txt$ = "TIME TO WIN "

test$ = cScrollL(Txt$)          "IME TO WIN T"
test$ = cScrollR(Txt$)          " TIME TO WIN"
```

See also :

CloseAllEditForm

Purpose :

CloseAllEditForm closes all VB edit form in the design environnement (windows with code only, the others are already closed by VB himself).

Declare Syntax :

```
Declare Function cCloseAllEditForm Lib "t2wlight.dll" () As Integer
```

Call Syntax :

```
test% = cCloseAllEditForm()
```

Where :

test%	TRUE if all is correct, FALSE if an error has occured.

Comments :

CloseAllEditForm use the Windows Enumeration to find which window class is an VB edit form.

Examples :

```
Dim Test            As Integer

Test = cCloseAllEditForm()
```

See also : cHideAllEditForm, cUnHideAllEditForm, cHideDebugForm, cUnHideDebugForm

Thanks you to register 'TIME TO WIN Light'.
SWREG #5808, price $25.00

DecrI, DecrL

Purpose :

DecrI auto-decrement an integer value by 1.
DecrL auto-decrement a long value by 1.

Declare Syntax :

Declare Sub cDecrI Lib "t2wlight.dll" (Value As Integer)
Declare Sub cDecrL Lib "t2wlight.dll" (Value As Long)

Call Syntax :

cDecrI Value%
cDecrL Value&

Where :

Value%	is the integer value to auto-decrement.
Valeu&	is the long value to auto-decrement.

Comments :

These routines are slower than the VB equivalent : Value = Value - 1 but are shorter to type.

Examples :

```
Dim Value As Integer

Value = 5

cDecrI Value              -> 4
cDecrI Value              -> 3
```

See also : cIncrI, cIncrL

HideDebugForm, UnHideDebugForm

Purpose :

HideDebugForm hides the debug window in the design environnement.
UnHideDebugForm unhides the debug window in the design environnement.

Declare Syntax :

```
Declare Function cHideDebugForm Lib "t2wlight.dll" () As Integer
Declare Function cUnHideDebugForm Lib "t2wlight.dll" () As Integer
```

Call Syntax :

```
test% = cHideDebugForm()
test% = cUnHideDebugForm()
```

Where :

test%	TRUE if all is correct, FALSE if an error has occured.

Comments :

HideDebugForm use the Windows Enumeration to find which window class is an VB debug form.
UnHideDebugForm use the Windows Enumeration to find which window class is an VB debug form.

Examples :

```
Dim Test            As Integer

Test = cHideDebugForm()
../.. some pieces of code
Test = cUnHideDebugForm()
```

See also : cCloseAllEditForm, cHideAllEditForm, cUnHideAllEditForm

HideAllEditForm, UnHideAllEditForm

Purpose :

HideAllEditForm hides all VB edit form in the IDE (windows with code only, the others are already closed by VB himself).
UnHideAllEditForm unhides all VB edit form in the IDE (windows with code only, the others are already closed by VB himself).

Declare Syntax :

```
Declare Function cHideAllEditForm Lib "t2wlight.dll" () As Integer
Declare Function cUnHideAllEditForm Lib "t2wlight.dll" () As Integer
```

Call Syntax :

```
test% = cHideAllEditForm()
test% = cUnHideAllEditForm()
```

Where :

test%	TRUE if all is correct, FALSE if an error has occured.

Comments :

HideAllEditForm use the Windows Enumeration to find which window class is an VB edit form.
UnHideAllEditForm use the Windows Enumeration to find which window class is an VB edit form.

Examples :

```
Dim Test            As Integer

../.. in a Form_Load event
Test = cHideAllEditForm()
../.. in a Form_UnLoad or Form_QueryUnLoad event
Test = cUnHideAllEditForm()
```

See also : cCloseAllEditForm, cHideDebugForm, cUnHideDebugForm

ArrayOnDisk

Purpose :

Put/Get full array on/from disk

Declare Syntax :

Declare Function cArrayOnDisk Lib "t2wlight.dll" (ByVal File As String, Array() As Any, ByVal GetPut As Integer) As Long

Call Syntax :

test& = cArrayOnDisk(File$, Array(), GetPut%)

Where :

File$	is the file to use.
Array()	is the array with any dimension.
GetPut%	PUT_ARRAY_ON_DISK to put the array on disk, GET_ARRAY_ON_DISK to get the array from disk.
test&	>=0 is the returned length of the file, < 0 is an error occurs (error n° is the negative value of all DA_x values, see Constants and Types declaration).

Comments :

This function can handle any type'd variable (if strings are used, you must use only fixed string).

Don't forget that if you use the 'ReDim' statement at the procedure level without have declared the array als Global, you must initialize the array before using this function (see below). You must initialize the array with enough space to handle the size of the file This is due to a VB limitation.

This function can handle huge array (greater than 65535 bytes) (see the example below).

Beware, the ANY parameter in the defintion of this function doesn't support string array (why ? ask to VB creator). To handle string (only fixed string), create a type'd variable with only an item, see below :

```
Type tagStringType
    newString           As String * 80
End Type

'This type replaces

Dim newString           As String * 80
```

Examples :

```
ReDim AD(-999 To 9000, 0 To 1)    As Long          'size is ((1+(9000 - -999)) * (1+(1 - 0)) * 4) = 80.000 bytes
Dim i                             As Long

For i = -999 To 9000
    AD(i, 0) = 1
    AD(i, 1) = 2
Next i

Debug.Print cArrayOnDisk("c:\tmp\test.dat", AD(), PUT_ARRAY_ON_DISK)          -> 80.000

For i = -999 To 9000
    AD(i, 0) = 0
    AD(i, 1) = 0
```

```
Next i

Debug.Print cArrayOnDisk("c:\tmp\test.dat", AD(), GET_ARRAY_ON_DISK)                    -> 80.000

Debug.Print AD(-999, 0), AD(9000, 0)
Debug.Print AD(-999, 1), AD(9000, 1)
```

See also : cArrayStringOnDisk

ArrangeDesktopIcons

Purpose :

This function arranges all desktop icons.

Declare Syntax :

Declare Sub cArrangeDesktopIcons Lib "t2wlight.dll" ()

Call Syntax :

Call cArrangeDesktopIcons()

Where :

Comments :

Examples :

See also :

CnvASCIItoEBCDIC, CnvEBCDICtoASCII

Purpose :

CnvASCIItoEBCDIC converts an ASCII string into EBCDIC equivalent.
CnvEBCDICtoASCII converts an EBCDIC string into ASCII equivalent.

Declare Syntax :

```
Declare Sub cCnvASCIItoEBCDIC Lib "t2wlight.dll" (Txt As String)
Declare Sub cCnvEBCDICtoASCII Lib "t2wlight.dll" (Txt As String)
```

Call Syntax :

```
Call cCnvASCIItoEBCDIC(Txt$)
Call cCnvEBCDICtoASCII(Txt$)
```

Where :

Txt$	the string to convert

Comments :

Examples :

```
Dim Tmp          As String

Tmp = "A/BC/DEF/GHIJ"

Call cCnvASCIItoEBCDIC(Tmp)
Debug.Print Tmp
        -> ÁaÂÃaÄÅÆaÇÈÉÑ

Call cCnvEBCDICtoASCII(Tmp)
Debug.Print Tmp
        -> A/BC/DEF/GHIJ
```

See also :

ProperName

Purpose :

ProperName converts the first letter of each word separated by a space in a string to upper case.

Declare Syntax :

Declare Function cProperName Lib "t2wlight.dll" (Txt As String) As String

Call Syntax :

Test$ = cProperName(Txt$)

Where :

Txt$	is the specified string.
Test$	is the returned string.

Comments :

Examples :

macdonald	becomes	Macdonald
mac donald	becomes	Mac Donald
John fitz,jr	becomes	John Fitz,jr
john Fitz, jr	becomes	John Fitz, Jr

See also :

FileCompress, FileExpand

Purpose :

FileCompress compress a file into a compressed format.
FileExpand expand a compressed file into a normal format.

Declare Syntax :

```
Declare Function cFileCompress Lib "t2wlight.dll" (ByVal file1 As String, ByVal file2 As String) As Long
Declare Function cFileExpand Lib "t2wlight.dll" (ByVal file1 As String, ByVal file2 As String) As Long
```

Call Syntax :

```
Test& = cFileCompress(File1$, File2$)
Test& = cFileExpand(File2$, File1$)
```

Where :

File1$	is the original file.
File2$	is the compressed file.
Test&	<0, an error has occured.
	>=0, the length of the created file.

Comments :

The compression gives the better result on TEXT file.

Examples :

See also :

ProperName2

Purpose :

ProperName2 convert the first letter of some words separated by a space or punctuation in upper letter case

Declare Syntax :

Declare Function cProperName2 Lib "t2wlight.dll" (Txt As String, ByVal TokenToUse As String, ByVal Options As Integer) As String

Call Syntax :

Test$ = cProperName2(Txt$, TokenToUse$, Options%)

Where :

Txt$	is the text to convert.
TokenToUse$	is the token list that can't be converted.
Options%	PN_UPPERCASE, works with upper case text.
	PN_PUNCTUATION, separator can be a space or a punctuation.
	PN_KEEP_ORIGINAL, keep case letter in the token list.
	PN_ONLY_LEADER_SPACE, don't use the leader trailer space for search in the token list.

Comments :

TokenToUse can be empty.
TokenToUse is a list of all words (separated by '/') which can't be converted (b.e. : "the/and/a/an/or/of")

Examples :

ProperName2 of 'JOHN FITZ,JR' is 'John Fitz,Jr'
ProperName2 of 'john Fitz,jr' is 'John Fitz,Jr'
ProperName2 of 'macdonald' is 'Macdonald'
ProperName2 of 'mac donald' is 'Mac Donald'
ProperName2 of 'a.l. greene jr.' is 'A.L. Greene Jr.'
ProperName2 of 'shale and sandstone and till' is 'Shale and Sandstone and Till'
ProperName2 of 'a sandstone or a shale' is 'a Sandstone or a Shale'

See also :

StringCompress, StringExpand

Purpose :

StringCompress compress a string into a compressed format.
StringExpand expand a compressed string into a normal format.

Declare Syntax :

```
Declare Function cStringCompress Lib "t2wlight.dll" (Txt As String) As String
Declare Function cStringExpand Lib "t2wlight.dll" (Txt As String) As String
```

Call Syntax :

```
Test$ = cFileCompress(Txt$)
Test$ = cFileExpand(Txt$)
```

Where :

Txt$	is the original string.
Test$	is the compressed string.

Comments :

The compression gives the better result on TEXT string.

Examples :

See also :

FillIncrD, FillIncrI, FillIncrL, FillIncrS

Purpose :

FillIncr fills, with an automatic incremented value, all of the elements of an array (double, integer, long, single).

Declare Syntax :

Declare Function cFillIncrD Lib "t2wlight.dll" (Array() As Double, ByVal nValue As Double, ByVal Increment As Double) As Integer
Declare Function cFillIncrI Lib "t2wlight.dll" (Array() As Integer, ByVal nValue As Integer, ByVal Increment As Integer) As Integer
Declare Function cFillIncrL Lib "t2wlight.dll" (Array() As Long, ByVal nValue As Long, ByVal Increment As Long) As Integer
Declare Function cFillIncrS Lib "t2wlight.dll" (Array() As Single, ByVal nValue As Single, ByVal Increment As Single) As Integer

Call Syntax :

status = cFillIncrD(array(), nValue, Increment)

Where :

array()	is the array.
nValue	is the starting value.
Increment	is the increment.
status	is always TRUE.

Comments :

See Also :

AddTwoTimes

Purpose :

AddTwoTimes adds two time string to form a third time string.

Declare Syntax :

Declare Function cAddTwoTimes Lib "t2wlight.dll" (ByVal Time1 As String, ByVal Time2 As String) As String

Call Syntax :

Test$ = cAddTwoTimes(Time1$, Time2$)

Where :

Time1$	is the first time string (format is HH:MM:SS).
Time2$	is the second time string (format is HH:MM:SS).
Test$	is the result (format is HH:MM:SS).

Comments :

The length of each time string must be absolutely 8 characters.
The format of each time string must be absolutely HH:MM:SS.
If the sum of the two time string exceed 24:00:00, the returned string is calculated from 00:00:00.

Examples :

```
Dim Time1 As String
Dim Time2 As String
Dim Time3 As String

Time1 = "23:58:58"
Time2 = "01:02:01"

Time3 = cAddTwoTimes(Time1$, Time2$)                -> "01:00:59"
```

See also :

IncrI, IncrL

Purpose :

IncrI auto-increment an integer value by 1.
IncrL auto-increment a long value by 1.

Declare Syntax :

Declare Sub cIncrI Lib "t2wlight.dll" (Value As Integer)
Declare Sub cIncrL Lib "t2wlight.dll" (Value As Long)

Call Syntax :

cIncrI Value%
cIncrL Value&

Where :

Value%	is the integer value to auto-increment.
Valeu&	is the long value to auto-increment.

Comments :

These routines are slower than the VB equivalent : Value = Value + 1 but are shorter to type.

Examples :

```
Dim Value As Integer

Value = 5

cIncrI Value            -> 6
cIncrI Value            -> 7
```

See also : cDecrI, cDecrL

FileToComboBox, FileToListBox

Purpose :

FileToComboBox read a file and append it to a Combo Box.
FileToListBox read a file and append it to a List Box.

Declare Syntax :

```
Declare Function cFileToComboBox Lib "t2wlight.dll" (ByVal hWnd As Integer, ByVal nFile As String) As Integer
Declare Function cFileToListBox Lib "t2wlight.dll" (ByVal hWnd As Integer, ByVal nFile As String) As Integer
```

Call Syntax :

```
Test% = cFileToComboBox(Combo1.hWnd, nFile$)
Test% = cFileToListBox(List1.hWnd, nFile$)
```

Where :

Combo1.hWnd	the .hWnd of a Combo Box.
List1.hWnd	the .hWnd of a List Box.
nFile$	the filename to read.
Test%	= True, if all is ok,
	<> True, if an error has occured.

Comments :

Examples :

```
Debug.Print cFileToComboBox(Combo1.hWnd, "c:\tmp\cmb_001.txt")
Debug.Print cFileToListBox(List1.hWnd, "c:\tmp\lst_001.txt")
```

See also :

FXpicture

Purpose :

FXpicture performs some specials effects on two Picture Box.

Declare Syntax :

Declare Function cFXpicture Lib "t2wlight.dll" (ByVal method As Integer, ByVal hdc1 As Integer, ByVal hbitmap As Integer, ByVal parameter As Integer, ByVal delay As Integer) As Integer

Call Syntax :

Test% = cFXpicture(method%, Picture1.hDC, Picture2.Picture, parameter%, delay%)

Where :

method%	FX_HORIZONTAL FX_VERTICAL FX_DIAGONAL_SQUARE FX_RECTANGLE
Picture1.hDC	is the .hDC of the first Picture Box.
Picture2.Picture	is the .Picture of the second Picture Box.
parameter%	= 0, default value will be 1, >0, the size of a line for special effect.
delay%	= 0, default value will be 10, >0, the delay between two lines for special effect.

Comments :

Normally, the .Visible property of the Picture2 must be set to False
Don't forget that the special effect works directly on the form not into the picture.

Examples :

```
Debug.Print cFXpicture(FX_HORIZONTAL, Picture1.hDC, Picture2.Picture, 0, 0)
Picture1.Picture = Picture2.Picture
```

See also :

FloppyInfo

Purpose :

FloppyInfo gives some informations on the selected floppy drive.

Declare Syntax :

Declare Function cFloppyInfo Lib "t2wlight.dll" (ByVal nDrive As String, nHeads As Integer, nCylinders As Integer, nSectors As Integer) As Integer

Call Syntax :

Size% = cFloppyInfo(nDrive$, nHeads%, nCylinders%, nSectors%)

Where :

nDrive$	is the drive letter ('A' or 'B')
nHeads%	is the returned number of Heads.
nCylinders%	is the returned number of Cylinders/Tracks.
nSectors%	is the returned number of Sectors by Cylinders/Tracks.
Size%	is the floppy size (360, 720, 1200, 1440, 2880).

Comments :

Examples :

```
Dim nSize               As Integer
Dim nHeads              As Integer
Dim nCylinders          As Integer
Dim nSectors            As Integer

nSize = cFloppyInfo("A", nHeads, nCylinders, nSectors)

nSize          -> 1440
nHeads         -> 2
nCylinders     -> 80
nSectors-> 18
```

See also :

DayOfWeek

Purpose :

DayofWeek calculate the day of the week.

Declare Syntax :

Declare Function cDayOfWeek Lib "t2wlight.dll" (ByVal nYear As Integer, ByVal nMonth As Integer, ByVal nDay As Integer, ByVal nISO As Integer) As Integer

Call Syntax :

Test% = cDayOfWeek(nYear%, nMonth%, nDay%, nISO%)

Where :

nYear%	is the year.
nMonth%	is the month.
nDay%	is the day.
nISO%	= True, for ISO specification,
	= False, for non-ISO specification.
Test%	is the returned day of the week.

Comments :

Following the ISO specification, the returned day of the week will be 0 (Monday) to 6 (Sunday).
Following the non-ISO specification, the returned day of the week will be 0 (Sunday) to 6 (Saturday).

If the parameters are incorrect, the returned value is -1.

Examples :

```
Dim Test            As Integer

'For ISO spefication

Test = cDayOfWeek(1995, 3, 25, True)              -> 5 (Saturday)
Test = cDayOfWeek(1995, 3, 26, True)              -> 6 (Sunday)
Test = cDayOfWeek(1995, 3, 27, True)              -> 0 (Monday)

'For non-ISO specification

Test = cDayOfWeek(1995, 3, 25, False)             -> 6 (Saturday)
Test = cDayOfWeek(1995, 3, 26, False)             -> 0 (Sunday)
Test = cDayOfWeek(1995, 3, 27, False)             -> 1 (Monday)
```

See also :

DateToScalar

Purpose :

DateToScalar compute a scalar from all date parts.

Declare Syntax :

Declare Function cDateToScalar Lib "t2wlight.dll" (ByVal nYear As Integer, ByVal nMonth As Integer, ByVal nDay As Integer) As Long

Call Syntax :

Test& = cDateToScalar(nYear%, nMonth%, nDay%)

Where :

nYear%	is the year.
nMonth%	is the month.
nDay%	is the day.
Test&	is the returned computed scalar.

Comments :

If the parameters are not correct, the returned value is -1.

Examples :

```
Dim Test        As Long

Test = cDateToScalar(1995, 3, 25)          -> 728377
```

See also : cScalarToDate

DayOfYear

Purpose :

DayOfYear calculates the day of the year.

Declare Syntax :

```
Declare Function cDayOfYear Lib "t2wlight.dll" (ByVal nYear As Integer, ByVal nMonth As Integer, ByVal nDay As Integer) As Integer
```

Call Syntax :

```
Test% = cDayOfYear(nYear%, nMonth%, nDay%)
```

Where :

nYear%	is the year.
nMonth%	is the month.
nDay%	is the day.
Test%	is the returned day of the year.

Comments :

The returned value is 365 or 366 (for a leap year).

If the parameters are incorrect, the returned value is -1.

Examples :

```
Dim Test As Integer

Test = cDayOfYear(1995, 1, 1)          -> 1
Test = cDayOfYear(1995, 3, 25)         -> 84
Test = cDayOfYear(1995, 12, 31)        -> 365
Test = cDayOfYear(1996, 12, 31)        -> 366
```

See also :

ScalarToDate

Purpose :

ScalarToDate decompose a scalar date into these components.

Declare Syntax :

Declare Sub cScalarToDate Lib "t2wlight.dll" (ByVal Scalar As Long, nYear As Integer, nMonth As Integer, nDay As Integer)

Call Syntax :

Call cScalarToDate(Scalar&, nYear%, nMonth%, nDay%)

Where :

Scalar&	is a scalar date.
nYear%	is the returned year.
nMonth%	is the returned month.
nDay%	is the returned day.

Comments :

Examples :

```
Dim nYear       As Integer
Dim nMonth      As Integer
Dim nDay        As Integer

Call cScalarToDate(728377, nYear%, nMonth%, nDay%)

        nYear%          -> 1995
        nMonth%         -> 3
        nDay%           -> 25
```

See also : cDateToScalar

WeekOfYear

Purpose :

WeekOfYear calculates the week of the year.

Declare Syntax :

Declare Function cWeekOfYear Lib "t2wlight.dll" (ByVal nYear As Integer, ByVal nMonth As Integer, ByVal nDay As Integer, ByVal nISO As Integer) As Integer

Call Syntax :

Test% = cWeekOfYear(nYear%, nMonth%, nDay%)

Where :

nYear%	is the year.
nMonth%	is the month.
nDay%	is the day.
nISO%	= True, for ISO specification,
	= False, for non-ISO specification.
Test%	is the returned week of the year.

Comments :

ISO defines the first week with 4 or more days in it to be week #1

Following the ISO specification, the returned week of the year will be 0 to 52.
Following the non-ISO specification, the returned week of the year will be 1 to 53.

If the parameters are incorrect, the returned value is -1.

Examples :

```
Dim Test            As Integer

'Following the ISO specification

Test = cWeekOfYear(1995, 12, 31, True)          -> 52
Test = cWeekOfYear(1995, 1, 1, True)            -> 0
Test = cWeekOfYear(1995, 1, 2, True)            -> 1
Test = cWeekOfYear(1995, 3, 25, True)           -> 12
Test = cWeekOfYear(1995, 3, 26, True)           -> 12
Test = cWeekOfYear(1995, 12, 31, True)          -> 52
Test = cWeekOfYear(1996, 1, 1, True)            -> 1

'Following the non-ISO specification

Test = cWeekOfYear(1995, 12, 31, False)         -> 53
Test = cWeekOfYear(1995, 1, 1, False)           -> 1
Test = cWeekOfYear(1995, 1, 2, False)           -> 1
Test = cWeekOfYear(1995, 3, 25, False)          -> 12
Test = cWeekOfYear(1995, 3, 26, True)           -> 13
Test = cWeekOfYear(1995, 12, 31, False)         -> 53
Test = cWeekOfYear(1996, 1, 1, False)           -> 1
```

See also :

GetVersion

Purpose :

GetVersion returns the version number of 'TIME TO WIN Light'

Declare Syntax :

Declare Function cGetVersion Lib "t2wlight.dll" () As Single

Call Syntax :

version% = cGetVersion()

Where :

Comments :

This is usefull to avoid version conflict with old version.

Examples :

```
version% = cGetVersion()          3.50
```

See also :