LibreOffice BASIC

LibreOffice Runtime Library

v. 1.12 – 12/02/2018

Written with LibreOffice v. 5.3.3 – Platform: All

Runtime Options

Must be specified for each module, before any executable code.

Option Explicit
Imposes explicit variable declaration. LibreO BASIC behaves like VBA.

Option VBA support 1
Activates VBA support.

Option Base 1
Arrays are 1-indexed instead of 0-indexed.

Option Class Module
To use for classes creation (+ Option Compatible).

BASIC Constants

True
False
Null
Empty
Pi

3.14159265358979
(Double)

Trim() Returns the ASCII value of the 1st character of a string.
Asc("Azerty") → 65
See Chr(), Asc() table.

Chr()
Returns the character which ASCII code is passed.
Chr(65) → "A"
See Asc(), Asc() table.

ConvertFromURL() Converts a file name in URL form to OS form.
URL form: protocol:///host/path/to/file.ext
Ex. Linux: file://home/user/somedir/file.xls

ConvertToURL() Converts a file name in OS form to URL form.
See ConvertFromURL().

Format()
Converts a number into a string, with mask formatting.
On the 7/14/2017, Format(Now(), "yyyy") → "2017" See Format function – Formatting Masks.

InStr()
Returns a string position within another.
InStr("LibreOffice", "Office") → 6
See InStr(), Asc() table.

Join()
Returns a string from an array of strings.
Join(MyArray, ";") → "C:\;Dir;SubDir;MyFile.xls"
See Join().

LCase()
Returns a string in lower case.
LCase("LibreOffice") → "libreoffice"
See UCase().

Left()
Extracts n characters from the left of a string.
Left("LibreOffice", 5) → "Libre"
See LCase(), Mid().

Len()
Returns the number of characters in a string.
Len("LibreOffice") → 11
See LCase(), Mid().

LTrim() Suppresses the leftmost spaces from a string.
See RTrim(), Trim().

Mid()
Extracts n characters in a string, starting at position P.
Mid("14/7/2017", 4, 1) → "7"
See Left(), Right(), Mid().

Right()
Extracts n characters from the right of a string.
See Left(), Mid().

RTrim() Suppresses the rightmost spaces from a string.
See LTrim(), Trim().

Space()
Returns a string made of a series of spaces.
Space(3) → " 
See String().

Split() Returns an array of strings from a single string, separating at a given character.
MyString = "C:\Dir\SubDir\MyFile.xls"
Split(MyString, ";") → 4 items array: "C:\", "Dir\", "SubDir\", "MyFile.xls"
See Join().

Str() Converts a numeric expression into a string.
Str(65) → "65"
See Asc(), Val().

StrComp() Compares two strings and returns an integer value that represents the comparison result.
Compares a string made of n times a character.
String(4, "+") → "YYYY"
See Space().

Trim() Suppresses the leftmost and rightmost spaces from a string.
See LTrim(), RTrim().

UCase()
Returns a string in upper case.
UCase("LibreOffice") → "LIBREOFFICE"
See LCase().

Val() Converts a string into a numerical value (e when not possible).
Val("12,34") → 12.34
See Str(), Val().

Functions

Numerical Functions

Abs() Returns a number absolute value. 
Exp() Exponential. Returns e to a given power.
Fix() Returns the integer part of a number (no rounding).
Hex() Returns the hexadecimal value of a decimal number.
Int() Returns a number integer part (rounded to the lower value).
Log() Returns the number logarithm.
Oct() Returns the octal value of a decimal number.
Randomize() Initializes the random number generator (before using Rand()).
Rand() Returns a random number, between 0 and 1. See Randomize().
Sign() Returns the sign of a number. 
Sgn() Returns a number square root.

Trigonometrical Functions

Angles in radians. radian = (degrees + π)/180
Arc tangent returns the angle in radians.
Cos() Cosine
Sin() Sine
Tan() Tangent.

Datetime Functions

LibreOffice API often uses "UNO" dates, that is of type com.sun.star.util.DateTime (or .Date or .Time), structured as follows:

ISO True if timezone is UTC.
Year Year number (if empty).
Month Month number (if empty).
Day Day number (if empty).
Hours Hours (0-23).
Minutes Minutes (0-59).
Seconds Seconds (0-59).
NanoSeconds Nanoseconds.

Date() Returns a date as a UNO com.sun.star.util.Date type.
DateAdd() Returns a new date from a starting date and an addition value.
See Now(), Time().

Day() Returns the current day in the month.
See Day(), Date().

Date() Returns the current date (Date type).
See Date(), DateAdd().

DateAdd() Adds a new date from a starting date and an addition value.
See Now(), Time().

DateDiff() Calculates a dates difference, expressed in the desired unit.
See DateValue().

DateValue() Returns a date as a UNO com.sun.star.util.Date type.

Day() Returns the current day in the month.

Hour() Returns the current time.

Minute() Returns the minutes of a Date type value.

Month() Returns the month number.

Second() Returns the seconds of a Date type value.

Time() Returns the current time as a Date type value.

Timer() Returns a Double value with the number of elapsed seconds from midnight.

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!

Set Timer() to a variable before use!
### Colors Functions

Colors are stored as longs. Red(), Green(), Blue() returns a color from its 3 components red, green, and blue.

RGB() = RGB(255,0,0) → #FF0000 (red)

### Array Functions

**Array()**

- Creates an array from discrete values.
- MyArray = Array("One", 2, Now())

**DimArray()**

- LikeArray() = DimArray("One", 2, Now())
- Use only if implicit variable declaration, otherwise use Array().

**Erase**

- (Instruction) Erases an array contents. In case of a dynamic array, frees the memory. Erase MyArray

**LBound()**

- Lower bound. LBound() Upper bound.

### Type Information Functions

These functions give information about the variables.

- **Any Variable**
  - **TypeName()** Returns a string that details a given variable.
  - **VarType()** Returns a value that reflects the OS, among:
    - Variant
    - String
    - Long
    - Integer
    - Date
    - Byte
    - Null
    - Boolean
    - Array
    - OLE object.

- **Error Information Functions**
  - Returns a value that reflects the OS, among:
    - Variant
    - String
    - Long
    - Integer
    - Date
    - Byte
    - Null
    - Boolean
    - Array
    - OLE object.

### UNO Structures And Objects

**CreateUnoService()**

- Creates a UNO service.

**GetUnoObject()**

- True if UNO structure.

**IsUnoStruct()**

- True if UNO structure.

**IsVoid()**

- Returns the UNO structure name (String).

**HasUnoInterfaces()**

- True if UNO object supports interfaces.

**IsObject()**

- True if the argument is an OLE object.

**IsEmpty()**

- Checks whether an optional parameter is omitted.

**IsArray()**

- Array.

**IsDate()**

- True if a value representing an OS date is available.

**CBool()**

- True if a boolean structured data item.

**CByte()**

- True if a byte structured data item.

**CChar()**

- True if a character structured data item.

**CLong()**

- True if a long structured data item.

**CInt()**

- True if an integer structured data item.

**CObj()**

- True if an object structured data item.

**CStr()**

- True if a string structured data item.

**CVar()**

- True if a general variable.

**CVarInt()**

- True if an integer structured data item.

**CVarLong()**

- True if a long structured data item.

**CVarString()**

- True if a string structured data item.

**CVarDate()**

- True if a date structured data item.

**CVarOle()**

- True if a variant structured data item.

**CVarArray()**

- True if an array structured data item.

**CVarOleArray()**

- True if an array of OLE objects.

**CVarStruct()**

- True if a structure structured data item.

**CVarNull()**

- True if a null structured data item.

**CreateUnoService()**

- Creates a UNO service.

**GetUnoObject()**

- True if UNO structure.

**IsUnoStruct()**

- True if UNO structure.

**IsVoid()**

- Checks whether an optional parameter is omitted.

### Typecast Functions

These functions convert a value from a compatible type into another. The function name reflects the target type name.

- **Code readability: always prefer an explicit typecast to an implicit one!**

### Error Information Functions

**Error**

- Error line number.

**Err**

- Error code.

**Misc. Functions**

**GetGUIType()**

- Returns a value that reflects the OS, among:
  - Windows
  - MacOS

**GetSolarVersion()**

- Returns LibreOffice version.

**IsMissing()**

- Checks whether an optional parameter is omitted.

### Calling System Commands

**Command syntax: Shell(CommandName, Style, Param, Synchro)** with:

- **Command**
  - The command to execute (String).

- **Style**
  - The window in which the process takes place, among (Integer):
    - 0: The program has focus, its window is hidden.
    - 1: The program has focus and runs in a standard window.
    - 2: The program has focus and runs as minimized.
    - 3: The program has focus and runs as maximized.
    - 4: The program starts in a standard non-focused window.
    - 5: The program starts in a minimized window, focus is on the current window.
    - 10: The program starts in full-screen mode.

**Param**

- Execution parameters to hand to the command (String).

**Synchro**

- True: Wait for the command execution to finish.
  - False: Do not wait for the command execution to finish.

### Format Function – Formatting Masks

The Format() function converts a number into a string by formatting it according to a mask.

A mask is a string that can be split in 3 sections separated with semicolons:

- `val;val2;val3`: One section only = all numbers.

### Numbers

<table>
<thead>
<tr>
<th>Number</th>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>(0)</td>
<td>(0) is mandatory at that position (0 if missing)</td>
</tr>
<tr>
<td>#</td>
<td>(#)</td>
<td># is the optional number separator</td>
</tr>
<tr>
<td>+</td>
<td>(+)</td>
<td>+ is a space-like character, appears as-is in the result.</td>
</tr>
<tr>
<td>-</td>
<td>(-)</td>
<td>- is the optional number separator</td>
</tr>
<tr>
<td>e</td>
<td>(\text{e})</td>
<td>(\text{e}) is the exponential separator</td>
</tr>
<tr>
<td>E</td>
<td>(\text{E})</td>
<td>(\text{E}) is the exponential separator</td>
</tr>
</tbody>
</table>

### Credits

**Author:** Jean-François Nifenecker – jean-francois.nifenecker@laposte.net

We are like dwarves perched on the shoulders of giants, and thus we are able to see more and further than the latter. And this is not at all because of the acuteness of our sight or the stature of our body, but because we are carried aloft and elevated by the magnitude of the giants. (Bernard of Clairvaux’s letters)

**History**

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.10</td>
<td>04/16/2018</td>
<td>First EN version</td>
</tr>
<tr>
<td>1.12</td>
<td>02/10/2018</td>
<td>Updates and fixes.</td>
</tr>
</tbody>
</table>

**License**

This ReCard is placed under the CreativeCommons BY-SA v3 (fl) license.

Information: https://creativecommons.org/licenses/by-sa/3.0/fr/