LibreOffice RefCard

LibreOffice BASIC Files

v. 3.0 - 09/02/2025

Written using LibreOffice v.25.2 - Platform: All



File paths

LibreOffice is multi-platform, thus file names are often expressed in URL form:

file:///device/path/to/somefile.ext

From OS to URL UrlFileName = ConvertToURL(OSFileName) = ConvertFromURL(URLFileName) From URL to OS OSFileName

BASIC Instructions

Returns a file or dir name, or the names of all files and directories Dir() found on a drive or dir which correspond to the specified path.

FileCopy() Copies a file.

FileDateTime() Returns a date and time of a file creation or last mod, as a string.

FileExists() Returns True if a file or directory exists. FileLen() Returns a file length (in bytes). GetAttr() Returns a file, volume or directory type.

GetPathSeparator() Returns the path separation character for the operating system.

Deletes a file from a drive. Kill() Creates a new directory. MkDir() Name()

Renames an existing file or directory. RmDir() Deletes an existing directory. Defines a given file attributes SetAttr()

Managing file contents using Instructions in BASIC

Process

- 1. Get an internal identifier on the file (FreeFile),
- Open the file (Open),
- Write to the file (Print, Put or Write) or read it (Get, Line Input# or Input#), 3.
- 4. Close the file (Close)

Accessing a file contents using its handle (identifier).

Close Closes a file that was opened using Open. Eof() Checks whether the file pointer is at the end.

FileAttr() Returns the handle of a file that was opened using Open. FreeFile Returns an available handle number before opening a file. Reads a record from a file and inserts it into a variable. Get

Sequentially reads an opened file data. Input

Line Input Reads a line from a file.

Returns the current position in an opened file. Loc()

Returns an opened file size in bytes. Lof() Opens a data channel.

0pen

Writes data into a sequential file (undelimited line). Print Writes a record into a file. Put

Close all open files and flushes the contents to the device of all file buff. Reset Seek() Returns the next read/write pos in a file opened using Open For Random.

Writes data into a sequential file (delimited line). Write

Print or Write?

Write adds delimiters according to the information type (text: ", date or boolean: #). These delimiters are ignored when read using Input.

Opening modes (Text files)

Sequential write: Open For Output Sequential read: Open For Input Finit: For other access modes (direct or binary access), use the stream APIs.

Sequential read example (Text file)

Read a file known from its FileURL address Read a file known from its FileURL address.

Dim Handle As Integer, TheLine As String
Handle = FreeFile

Open FileURL For Input As #Handle

'line reads

Do While Not Eof(Handle)

'we read the current line

Line Input #Handle, TheLine

Loop Close #Handle

Sequential write example (Text file)

Write to a file known from its FileURL address. Dim Handle As Integer: Handle = FreeFile
Open FileURL For Output As #Handle
'line writes
Print #Handle, "Some text."
Print #Handle, "More..."
Print #Handle, "The end."

Close #Handle

Using a SimpleFileAccess object

oSFA = createUNOService("com.sun.star.ucb.SimpleFileAccess")

Copies a file. сору

createFolder Creates a new directory

Checks whether a file or directory exists. exists Returns a file contents type. getContentType getDateTimeModified Returns a file last modification date.

Returns a directory contents getFolderContents getSize Returns a file size.

isFolder Checks whether an URL is a directory. isReadOnly Checks whether a file is set as read-only.

Del a file or directory. A dir is always deleted empty or not.

Moves a file. move

setInteractionHandlerDeclares an interaction handler for ulterior operations.

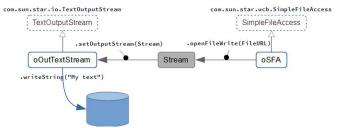
setReadOnly Sets a file read-only flag (rights necessary).

Managing file contents using the stream APIs

We call the LibreOffice API SimpleFileAccess and Stream services.

Overview

Example: writing to a text file (the actual code is shown in the example below).



Process

Create the object for accessing files,

oSFA = createUNOService ("com.sun.star.ucb.SimpleFileAccess")

Connect the stream that corresponds to the process (access type),

Write to the file or read it (according to the file type),

In write mode, flush the stream (.flush),

Close the file (.closeXxx).

Accessing file contents

SimpleFileAccess service (SFA)

openFileRead Opens a file in read mode. openFileWrite Opens a file in write mode. openFileReadWrite Opens a file in write and read mode.

Stream services (InputStream, OutputStream and Stream)

These are the "active" parts.

Correspondance between the SFA service and streams:

SFA methods Stream services

com.sun.star.io.InputStream
com.sun.star.io.TextInputStream
com.sun.star.io.OutputStream
com.sun.star.io.TextOutputStream openFileRead openFileWrite

com.sun.star.io.Stream openFileReadWrite

Stream service

getInputStream Returns the InputStream part of a read/write stream. getOutputStream Returns the OutputStream part of a read/write stream. Closing these streams closes the OutputStream or InputStream respectively.

InputStream service

Reads the specified number of bytes. readBytes

readSomeBytes Reads the available number of bytes, with the maximum specified.

skipBytes Skips the specified number of bytes (positive value).

available Returns the number of bytes that can safely be read or skipped.

closeInput Closes the stream.

TextInputStream service Inherits from InputStream.

readLine

Reads the text until a line break (CR, LF, or CRLF) or E0F is met. Returns the string read (without CR or LF).

The chars read are converted according to setEncoding. If EOF was

reached before running the method, a 0-length string is returned.

Reads the text until a delimiter or EOF is met. Returns the string read. readString

CRLF is **not** the default delimiter! This means that, when no delimiter is specified or found, the stream is read to E0F.
The characters read are converted according to the encoding set using setEncoding. If E0F was reached before running the method, a zero-

length string is returned. Returns the EOF status.

This status cannot be detected by a read attempt for an empty string because such a response may be valid when using readLine() (the line is empty) or readString() (two delimiters inline). Sets the char encoding (defaults to UTF-8). The available names are listed

setEncoding here: http://www.iana.org/assignments/character-sets (Name column). The current supported char sets depend upon the implementation.

OutputStream service

isE0F

writeBytes Writes a sequence of bytes in the stream (blocking call).

flush Flushes the stream buffers. closeOutput Used to signal that all data have been written.

TextOutputStream service Inherits from OutputStream,

writeString Writes a string to the stream (encoding specified using setEncoding).

Line breaks and delimiters that might be necessary, have to be manually

added to the string

See above ${\tt TextInputStream.setEncoding}$. setEncoding

Example: Reading from a text file

Dim osFA as Object, oInText As Object, FileURL As String, TheLine As String oSFA as Object, oInText As Object, FileURL as String, TheLine As String oSFA = createUNOService("com.sun.star.ucb.SimpleFileAccess")
FileURL = ConvertToURL("C:\path\to\MyFile.txt")
oInText = createUNOService("com.sun.star.io.TextInputStream")
oInText.setInputStream(oSFA.openFileRead(FileURL))
TheLine = oInText.readLine()
oInText.closeInput()

oInText.closeInput()

Example: Writing to a text file

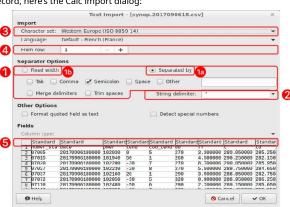
Dim oSFA As Object, oOutText As Object, FileURL As String
oSFA = createUNOService("com.sun.star.ucb.SimpleFileAccess")
FileURL = ConvertToURL("C:\path\to\MyFile.txt")
oOutText = createUNOService("com.sun.star.io.TextOutputStream")
oOutText.setOutputStream(oSFA.openFileWrite(FileURL))
'write (line delimiters must be specified)
oOutText.WriteString("HellO World" & Chr(13) & Chr(10)) 'CRLF (Windows)
oOutText.WriteString("line #2" & Chr(13) & Chr(10))
'flush buffers and close
oOutText.flush
oOutText.closeOutput()

LibOBasic 07 Files Flat A4 EN v300.odt

Importing text files (CSV)

Using the GUI

For the record, here's the Calc import dialog:



CSV filters items

CSV filters require a set of parameters (see the above figure for reference, #1 to #5). The final filter string is a wrap-up of all points below.

1. Field separator

a. Variable format

Separator ASCII code (see ASCII table) "9" Alternatives are separated with / "9/36" Merged characters: append /MRG "36/36/MRG" b. Fixed Format "FIX"

2. Text delimiter

"34" ("" if none) Delimiter ASCII code (see ASCII table)

3. Character set

Frequently found character sets:

"ANSI" ISO-8859-15/EURO "22" Windows-1252/WinLatin1 "12" "76" or "" ISO-8859-1 UTF-8

Up-to-date list of supported sets: service "com.sun.star.document.FilterFactory"

4. First line to process

"2" ("1" or "" if 1st line) That line number (1-based)

5. Column formats

a. Variable For each column, a 4-character sequence: "1/2/ 2/2/ 3/1/" ("" when only default #col (1-based) / format / values) (see the Column Formatting Codes table)

b. Fixed For each column, a 4-character sequence: "0/1/ 8/2/ 13/1/" 1st char pos. (0-based) / format /

(see the Column Formatting Codes table)

- You may insert spaces to improve the sequence readability.
- You may only specify used columns.

6. Language

0 (or omitted): interface language, otherwise language **decimal** code (see Windows Lan $guage\ Code\ Identifier\ (LCID)\ Reference).\ Ex: fr-FR = 1036\ (0x040C)$; en-US = 1033\ (0x0409).

7. Quoted as text field

True or False (default/omitted = False)

8. Detect special numbers

True or False (default/omitted = False)

9. (empty: export only)

10. (empty: export only)

11. Suppress spaces

True or False (default/omitted = False)

12. (empty: export only)

13. Evaluate formulae

An item starting with = (égale) is treated as a formula and can be evaluated at import time. Values True or False (default/omitted = False).

More about filters: https://wiki.documentfoundation.org/Documentation/DevGuide/ Spreadsheet_Documents#Filter_Options_for_the_CSV_Filter

CSV import filter

The filter to use is created by

concatenating the 5 parameters, using





Filter = "9,34,76,1,1/2/2/2/3/1" commas as separators: Depending on the context, some values may be omitted (see parameters details): Filter = "59,,76,,"

More Information

DD/MM/YY

Frequent ASCII Codes in CSV files (Decimal values)

Tabulation 34 " 36 \$ 44 59; 35 # 32 Space 39 58 :

Column formatting codes

Standard (automatic selection)

YY/MM/DD 9 (ignore column)

MM/DD/YY

10 US Format (decimal and thousands sep.)

Importing a CSV file into a spreadsheet

Say we've got a CSV file named MyFile.csv. We want to copy its contents to Sheet1 in the current document.

Any preexisting contents is **replaced** without warning.

Creating a Calc spreadsheet from a CSV

```
Creating a Calc spreadsheet from a CSV

You want to create a new Calc spreadsheet from a CSV file named MyFile.csv.

Dim props1(1) As New com.sun.star.beans.PropertyValue
Dim props2()

Dim CsvURL As String 'the .csv source address
Dim DocURL As String 'the .odt target address
Dim DocURL As String 'the target document

CsvURL = ConvertToURL("C:\path\to\MyFile.csv")

'csv file read options
props1(0).Name = "FilterName"
props1(0).Value = "Text - txt - csv (StarCalc)"
props1(1).Name = "FilterOptions"
props1(1).Value = "9,34,ANSI,1,1/2/2/2/3/1/4/1"

'load csv source file into the first sheet
oDoc = StarDesktop.loadComponentFromURL(CsvURL, "_blank", 0, props1())

'save to the ods target file
DocURL = ConvertToURL("C:\path\to\Spreadsheet.ods")
oDoc.storeAsURL(DocURL, props2())

There's only one sheet in the newly created document, named from the source file.
      There's only one sheet in the newly created document, named from the source file.
```

In the example, we display the created document. For hiding it:

 add a Hidden option with value True in props1()
 add oDoc.close(True) in the end.

Any preexisting document with the same URL is **overwritten** without warning.

Exporting LibreOffice documents

(saving documents principles: see RefCard #3)

Exportation principles

Define property arrays depending upon your needs. Ex. for Calc → PDF:

Dim Data(1) As New com.sun.star.beans.PropertyValue

Dim Filter(1) As New com.sun.star.beans.PropertyValue

Data(0).Name = <property1 name> 'see below

Data(0).Value = <property1 value>

Data(1).Name = <property2 name>

Data(1).Value = <property2 name>

Data(1).Value = <property2 value>

'[etc.]

Filter(0).Name = "FilterName"

Filter(0).Value = "calc odf Export" 'filter name

ThisComponent.storeToURL(ConvertToURL(FileName), Filter())

Export properties

Every filter may offer specific properties. Check on the Internet.

Frequent export filters

	From	10	Fliter name
	Calc	PDF	"calc_pdf_Export"
	Calc	Excel 97-2003 (.xls)	"MS Excel 97"
	Calc	Excel 2007-365(.xlsx)	"Calc MS Excel 2007 XML"
	Writer	PDF	"writer_pdf_Export"
	Writer	Word (.doc)	"MS Word 97"
	Writer	Word 2010-365 (.docx)	"Office Open XML Text"
Filters list: see help page File Conversion Filter Names.			

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 farther 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 Chartres (attr.1)

History

Version	Date	Comments
3.0	09/02/2025	Updates to LibO 25.2. Formatting.

License

This RefCard is distributed under the CreativeCommons BY-SA v4 (intl) license. More information:



