



Written using LibreOffice v. 7.0.5 - Platform : All

Manipulating files and directories

Using Basic Instructions

Dir()	Returns a file or directory name or the names of all files and directories found on a drive or directory which correspond to the specified path.
FileCopy()	Copies a file.
FileDateTime()	Returns a file date and time of a file creation or last modification, 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.
Kill()	Deletes a file from a drive.
MkDir()	Creates a new directory.
Name()	Renames an existing file or directory.
Rmdir()	Deletes an existing directory.
SetAttr()	Defines a given file attributes.

Using a SimpleFileAccess object

Methods of the com.sun.star.ucb.SimpleFileAccess service:

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

copy	Copies a file.
createFolder	Creates a new directory
exists	Checks whether a file or directory exists.
getContentTypes	Returns a file contents type.
getLastModified	Returns a file last modification date.
getFolderContents	Returns a directory contents
getSize	Returns a file size.
isFolder	Checks whether a URL is a directory.
isReadOnly	Checks whether a file is set as read-only.
kill	Deletes a file or directory.
move	A directory is always deleted whether empty or not. Moves a file.
setInteractionHandler	Declares an interaction handler for ulterior operations.
setReadOnly	Sets a file read-only flag (rights necessary).

File paths

LibreOffice is multi-platform, thus file names are often expressed in URL form:
file:///device/path/to/somefile.ext

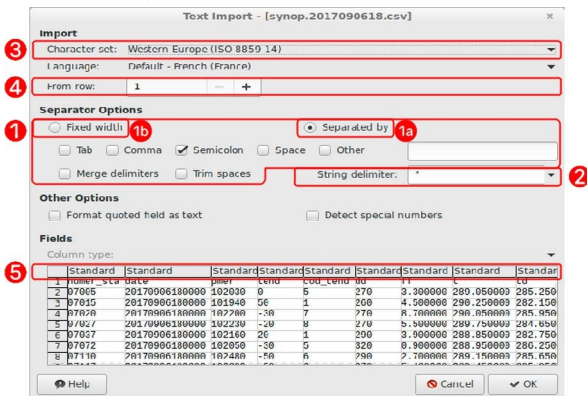
Conversion functions

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

Importing text files (CSV)

Using the GUI

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 5.

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

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

3. Character set

Frequently found character sets:

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

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

4. First line to process

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

5. Column formats

a. Variable	For each column, a 4-character sequence: #col (1-based) / format / (see the Column Formatting Codes table)	"1/2/2/2/3/1" (" " when only default values)
b. Fixed	For each column, a 4-character sequence: 1 st char pos. (0-based) / format / (see the Column Formatting Codes table)	"0/1/8/2/13/1"

Vous pouvez intercaler des espaces pour rendre la séquence plus lisible.

Vous pouvez ne spécifier que les seules colonnes utiles.

6. Language

θ (or omitted) : interface language, otherwise language **decimal** code (see *Windows Language 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).

Filters details: 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 commas as separators:

Filter = "9,34,76,1,1/2/2/2/3/1"

Depending on the context, some values may be omitted (see parameters details) : Filter = "59,,76,, "

More Information

Frequent ASCII Codes in CSV files (Decimal values)

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

Column formatting codes

1 Standard (automatic selection)	5 YY/MM/DD
2 Text	9 (ignore column)
3 MM/DD/YY	10 US Format (decimal and thousands sep.)
4 DD/MM/YY	

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.

The copy is made by creating a link then deleting it:

```
Dim CsvURL As String 'the .csv source address
Dim Filter As String
Dim oSheet As Object 'the target sheet in the document

oSheet = ThisComponent.Sheets.getByIndex("Sheet1")
CsvURL = ConvertToURL("C:\path\to\MyFile.csv")
'csv file read options
Filter = "9,34,ANSI,1,1/2/2/2/3/1/4/1"
'import creating a link between the sheet and the .csv source
oSheet.Link(CsvURL, "", "Text - txt - csv (StarCalc)",
Filter, com.sun.star.sheet.SheetLinkMode.VALUE)
'release link so that the document is independent
oSheet.SetLinkMode(com.sun.star.sheet.SheetLinkMode.NONE)
```

Any preexisting contents is replaced without warning.

Creating a Calc spreadsheet from a CSV

We've got a CSV file named MyFile.csv. We want to create a new Calc document from this file contents.

```
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 .ods target address
Dim oDoc As Object '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. The sheet is named from the source file name.

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 replaced without warning.

Managing file contents using Instructions in BASIC

Process

1. **Get** an internal identifier on the file (FreeFile),
2. **Open** the file (Open),
3. **Write** to the file (Print, Put or Write) or **read** it (Get, Line Input# or Input#),
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.
Get	Reads a record from a file and inserts it into a variable.
Input	Sequentially reads an opened file data.
Line Input	Reads a line from a file.
Loc()	Returns the current position in an opened file.
Lof()	Returns an opened file size in bytes.
Open	Opens a data channel.
Print	Writes data into a sequential file (undelimited line).
Put	Writes a record into a file.
Reset	Close all opened files and flushes the contents write to the device of all file buffers.
Seek()	Returns the next read or write position in a file opened using Open For Random instruction.
Write	Writes data into a sequential file (delimited line).

Print or Write?
Print saves the text as-is.
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

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.

```
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.

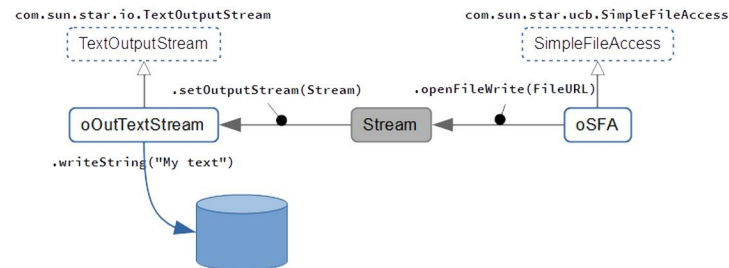
```
Dim Handle As Integer
Handle = FreeFile
Open FileURL For Output As #Handle
'line writes
Print #Handle, "Some text."
Print #Handle, "More..."
Print #Handle, "The end."
Close #Handle
```

Managing file contents using the stream APIs

We call the LibreOffice API SimpleFileAccess and Stream services.

Principle

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



Process

1. **Crte** the object for accessing files,
oSFA = createUNOService ("com.sun.star.ucb.SimpleFileAccess")
2. **Connect** the stream that corresponds to the process (access type),
3. **Write** to the file or **read** it (according to the file type),
4. In write mode, **flush** the stream (.flush),
5. **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
openFileRead	com.sun.star.io.InputStream
openFileWrite	com.sun.star.io.OutputStream
openFileReadWrite	com.sun.star.io.Stream

Stream service

getInputStream	Returns the InputStream part of a read/write stream. Closing this stream closes the OutputStream as well.
getOutputStream	Returns the OutputStream part of a read/write stream. Closing this stream closes the InputStream as well.

InputStream service

readBytes	Reads the specified number of bytes.
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 EOF is met. Returns the string read (without CR or LF). The characters read are converted according to the encoding set using setEncoding. If EOF was reached before running the method, a zero-length string is returned.
readString	Reads the text until a delimiter or EOF is met. Returns the string read. CRLF is not the default delimiter! This means that, when no delimiter is specified or found, the stream is read to EOF. The characters read are converted according to the encoding set using setEncoding. If EOF was reached before running the method, a zero-length string is returned.
isEOF	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).
setEncoding	Sets the character encoding (defaults to UTF-8). The available names are listed in this document: http://www.iana.org/assignments/character-sets (Name column). The current supported character sets depend upon the implementation.

OutputStream service

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 using the encoding specified using setEncoding. Line breaks and delimiters that might be necessary, have to be manually added to the string.
setEncoding	See above TextInputStream.setEncoding.

Example : Reading from a text file

```
Dim oSFA As Object, oInText As Object
Dim 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()
```

Example : Writing to a text file

```
Dim oSFA As Object, oOutText As Object
Dim 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)
'[here we use CRLF (Windows)]
oOutText.WriteString("Hello World" & Chr(13) & Chr(10))
oOutText.WriteString("Line #2" & Chr(13) & Chr(10))
'flush buffers and close
oOutText.flush
oOutText.closeOutput()
```

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.])

History

Version	Date	Comments
2.0	04/06/2021	Revised for LibO 7; formatting updated.
2.1.0	01/03/2024	Added: details about CSV filters.

License

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

<https://creativecommons.org/licenses/by-sa/4.0/>

