**LibreOffice RefCard**

**LibreOffice Basic**

**Calc**

v. 1.14 – 120208

Written using LibreOffice v. 5.3.3 Platform: All

---

### LibreOffice Documents

#### Current Document

Dim Doc As Object
Doc = ThisComponent

#### Opening An Existing Document

In Visible Mode

Dim Doc As Object, DocPath As String
Doc Props() 'here, an unitialized table
Doc = StarDesktop.loadComponentFromURL(DocPath, "_blank", 0, _
Props())

In Hidden Mode

Dim Doc As Object, DocPath As String
Doc Props() 'the document is opened as hidden
Props(0).Name = "hidden" 'the document opens
Props(0).Value = True
Doc = StarDesktop.loadComponentFromURL(DocPath, "_blank", 0, _
Props())

#### Make Visible An Hidden Document

Doc.CurrentController.Frame.ContainerWindowToFront()

#### Creating A New Calc Document

From (1) the default template or (2) a specified template.

Dim Doc As Object
Doc Props() 'here, an unitialized table
Template = "C:\Path\To\SomeTemplate.ots" '(2)
Template = "private:factory/scalc" '(1)
Props()   'here, an unitialized table
Dim Doc As Object
Props(0).Value = True
DocPath = ConvertToURL("C:\Path\To\SomeSpreadsheet.ods")
Dim Props(0) As New com.sun.star.beans.PropertyValue
Props(0).Name = "Version"      'string
Props(0).Value = "5.3.3"
Dim Doc As Object

#### Saving A Document

The Document Already Exists

(same as File > Save)
Use the document object store method. Ex: ThisComponent.store

The Document Wasn't Saved Yet

(same as A File > Save As)

#### Closing A Document

Use the document object close method: ThisComponent.close(True)

#### Getting Document Information

The Document Object Exposes Properties

Location

The document storage directory

TheType = ACell.Type

FileProperties (Object)

TheType = ACell.Type

Author

ModifyDate Last modification date.

CreationDate Creation time.

ModifiedBy Name of the user who modified the document.

Description Comments.

Subject (String).

ModifiedBy Name of the user who modified the document.

#### Is This A Calc Document?


The Calc object points to the document (ex: Doc = ThisComponent).

#### Automatic Calculation

Active? (Boolean)

Auto = Doc.isAutomaticCalculationEnabled
Doc.enableAutomaticCalculation(False)

Deactivate

Auto = Doc.isAutomaticCalculationEnabled
Doc.enableAutomaticCalculation(False)

Activate

Auto = Doc.isAutomaticCalculationEnabled
Doc.enableAutomaticCalculation(True)

Force calculation

Doc.calculateAll(only non updated formulas)
Doc.calculateAll(everything)

#### Protecting The Spreadsheet

Is the document protected?

Test = Doc.IsProtected
Doc.protect(Password) 'may be empty
Doc.unprotect(Password)

### Sheets

The Calc object points to the document (ex: Doc = ThisComponent).

#### Getting Sheets

The active sheet

ASheet = Doc.CurrentController.ActiveSheet

The sheets list

TheSheets = Doc.Sheets

Sheet count

SheetsCount = Doc.Sheets.Count

Sheet object (using its index)

ASheet = Doc.Sheets(index)

Sheet object (using its name)

ASheet = Doc.Sheets.getByName(SheetName)

Check existence (name)

Exists = Doc.Sheets.hasByName(SheetName)

Sheet index

SheetIndex = ASheet.RangeAddressSheet

#### Modifying Sheets

Adding a named sheet name at position p (0-based)

Doc.Sheets.insertByName(Name, p)

Deleting a sheet

Doc.Sheets.removeByName(SheetName)

Duplicating a sheet

Doc.Sheets.copyByName(SourceName, TargetName, p)

Moving a sheet to position p (0-based)

Doc.Sheets.moveByName(SheetName, p)

Protecting a sheet

ASheet.protect(Pwd)

Unprotecting a sheet

ASheet.unprotect(Pwd)

#### Tab coloring

ASheet.tabColor = RGB(255, 255, 0)

#### Linking A Sheet

Linking to a file

ASheet.link("", "Text - txt - csv (StarCalc)", _
(ex: CSV)"
Doc.Sheets.InsetByName(Name, p)

Describing a link

ASheet.getLinkMode(com.sun.star.sheet.SheetLinkMode.VALUE)

#### Finding The Last Used Row/Col

ASheet object.
Row and Col are the values to retrieve.

Dim Cur As Object
Cur RangeAddress.EndRow
Dim Range as Object
Range EndOfUsedArea(True)
Cur.gotoEndOfUsedArea(True)

#### Cells

Below, ACell is a cell object.

#### Getting Cells

ASheet is a sheet object. Row and Col are the values to retrieve.

Dim DimCur As Object
Dim DimCol As Long
Cur RangeAddress.EndRow
Row = Range.RangeAddress.EndRow

#### Knowing The Active Cell

Doc is a document object and ACell is the active cell we’re looking for.

If Doc.currentSelection.supportsService _
("com.sun.star.sheet.SheetCell") Then

#### Selecting A Cell

ThisComponent.CurrentController.select(ACell)

#### Cell Coordinates

Coordinates (Object)

Coord = ACell.Address

Column index (Long)

NumCol = ACell.Address.Column

Row index (Long)

NumRow = ACell.Address.Row

Container sheet object

ASheet = ACell.Address.Sheet

Absolute coordinates (String)

Coord = ACell.AbsoluteName

UnProtecting Cells

The ACell.CellProtection property can get the boolean values:

No modifications

CellProtection.IsLocked = True
Hide formula

CellProtection.IsFormulaHidden = True
Hide cell

CellProtection.IsHidden = True
Don’t print cell

CellProtection.IsPrintHidden = True

#### Properties

Getting text contents

MyText = ACell.String

Getting numerical contents

MyValue = ACell.Value

Getting a formula (US)

AFormula = ACell.Formula

Getting a formula (local lang.)

AFormula = ACell.FormulaLocal

Knowing content type (below)

TheType = ACell.Type

Emptying a cell

ACell.String = ""

#### Content Types (type property)

The Calc object points to the document (ex: Doc = ThisComponent).

The calc object points to the document (ex: Doc = ThisComponent).

#### Writing In A Cell

Replacing empty text

ACell.String = "He"
Ranges

Range = a set of cells (one cell is a range): Dim MyRange As Object

Getting Ranges

ASheet is a sheet object. You get the ARange range object:

By its name
ARange = ASheet.getCellRangeByName("C2:G4")

By its coordinates
ARange = ASheet.getRangeByPosition(2, 1, 6, 13)

Arbitration
ARange = ThisComponent.Sheets.getRangeByNamePosition(2, 1, 6, 13)

Getting The Active Range

ThisComponent.CurrentController.select(ARange) where ARange is an object.

Ranges Coordinates

Coordinates (Object)

Coord = ARange.RangeAddress

Sheet index (Integer)

Rang = ARange.RangeAddress.Sheet

Column index (Long)

NumCTL = ARange.RangeAddress.StartColumn
topleft

Row index (Long)

NumTL = ARange.RangeAddress.StartRow
topleft

Column index (Long)

NumCER = ARange.RangeAddress.EndColumn

topleft

Row index (Long)

NumBER = ARange.RangeAddress.EndRow

topleft

bottom/right

NumLBR = ARange.RangeAddress.StartColumn

bottom/right

NumLER = ARange.RangeAddress.EndRow

bottom/right

Containing sheet object

ASheet = ARange.Spreadsheet

Absolute coordinates (String)

Coord = ARange.AbsoluteName

Named ranges

The doc object points to the document. Also Ranges As Object

Ranges = Doc.NamedRanges

Ranges count (Long)

Count = Ranges.Count

Getting a range (by index)

ARange = Ranges(index)

Checking existence (by name)

Exist = Ranges(by name).Name

Getting a Range (by name)

ARange = Ranges(by name).Name

Adding

Ranges.addByName(ARange, CellRef, Address, b, )

Removing (by name)

Ranges.removeByName(ARange, Name)

Clearing A Range

Clearing ARange contents

ARange.clearContents(DeMode)

DeMode is a value that specifies the deletion type.

Ranges As Object

com.sun.star.sheet.CellFlags.XXX constants give that choice (combine with +): 0

Anonymous Comments STRING Text

DATE/TIME Number formatted as date-time

VALUE Numbers (except date-time)

FORMULA Formulas

Getting A Range Cells Contents

ARange.DataArray is a tableau des valeurs des cellules for ARange.

Copying A Range Contents To Another Range

Two ranges Source and Target, of the same dimensions.

Copying the contents (values) of Source Target.DataArray = Source.DataArray into Target.

Writing Values Into A Range

ARange is a range object. Table is a table of the same dimensions, which values have to be copied into the range.

Dim Table As Variant

Table = ARange.DataArray.

'Stable has the same dimensions as the range ('liste the array items values)

ARange.DataArray = Table

Browsing A Range Cells

From a collection (Ranges.Cells), we create an enumeration. It is then browsed by calling its hasMoreElements et NextElement properties:

Dim Ranges As Object

Ranges = TheEnum.createInstance("com.sun.star.sheet.CellCellSheetsRanges")

Ranges.insertByName("some name ", ARange)

TheEnum = Ranges.Cells.createEnumeration

Do While TheEnum.hasMoreElements

MyObject = TheCell.NextElement

'get the cell object

Loop

Empty cells are not browsed!

Ranges Misc.

Merging ARange object cells

ARange.Merge

Ranges Types

The range access mode defines which service it implements:

com.sun.star.sheet.CellCell

com.sun.star.sheet.CellCellRange

com.sun.star.table.CellCell

com.sun.star.table.CellCellRange

com.sun.star.sheet.NamedRange

The implemented service implies how the ranges should be used.

Check using their supportsService() method (ex. below)

Range Or Cell?

To know an object type, test supportsService() on a Range or Cell:

If MyObj.supportsService(Service_Name) Then ...

Replace Service_Name with:

Cell

"com.sun.star.sheet.CellCell"

Simple Range?

"com.sun.star.sheet.CellCellRange"

Multiple Range?

"com.sun.star.sheet.CellCellRange"

Always check for a Cell type before a Simple Range as a cell is also a Simple Range!

Rows/Columns

Lines and columns are properties of Sheet and Range objects.

Overview

Lines (ofows object)

oRows = ARange.Rows

Columns (ocols object)

oCols = ARange.Columns

Number

NumL = ARange.Columns.Count

NumC = ARange.Columns.Count

A-Row (ofow object) (base 0)

oRow = ARange.Rows(index)

A-Column (ocolumn object) (base 0)

oCol = ARange.Columns(index)

Rows/Columns Properties

Apply to Row or Rows objects (resp. Column or Columns).

Visible/Hidden (Boolean)

IsVisible = True

Optional width or not (Boolean)

OptimalWidth = True

Inserting/Deleting Rows/Columns

Given the objects RorC, FirstPos and LastPos the starting and ending positions for the rows set (resp. columns) to insert or delete (Long).

Inserting

RorC.insertByIndex(FirstPos, LastPos)

Deleting

RorC.removeByIndex(FirstPos, LastPos)

Fixing Rows Or Columns

Use the controller object oCtrlr = ThisComponent.CurrentController

Is there any?

Fixed = oCtrlr.hasFrozenPanes

Fix

oCtrlr.freezeAtPosition(1, 2)

Delete

oCtrlr.freezeAtPosition(0, 6)

Calling A Calc Function

Use the "com.sun.star.sheet.functionAccess" service.

DECLARE

Dim FCalc As Object

Dim Result As (according to context)

Dim Params As (according to context)

Dim FuncName As String

FCalc = CreateUnoService("com.sun.star.sheet.FunctionAccess")

Result = FCalc.callFunction(FuncName, Params)

Formula

FCalc = CreateUnoService("com.sun.star.sheet.FunctionAccess")

Result = FCalc.callFunction("SUM()", Array(1, 55, 321, 8))

Example 1 (sum() )

Example 2 (round() )

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 de Chartres [attr.])

Version Date Comments
1.14 02/12/2018 Minor updates

License

This work is licensed under the Creative Commons BY-SA v4 (int) license. More information: https://creativecommons.org/licenses/by-sa/4.0/