Guía de referencia LibreOffice

LibreOffice Basic n.º 3
Calc

Principiante

v. 1.15 – 14/12/2019

Redactado con LibreOffice v. 5.3.3 – Plataforma : Todas

Documentos de LibreOffice

Document activo

```
Dim Doc As Object
Doc = ThisComponent
```

Abrir un documento existente

Modo visible

```
Dim Doc As Object, RutaDoc As String
Dim Props()  'este array no se inicializa pero es necesario
    definirlo
RutaDoc = ConvertToURL("C:\Ruta\a\UnaHoja.ods")
Doc = StarDesktop.loadComponentFromURL(RutaDoc,"_blank",0,Props())
```

Modo invisible

```
Dim Doc As Object, RutaDoc As String
Dim Props(0) As New com.sun.star.beans.PropertyValue
RutaDoc = ConvertToURL("C:\Ruta\a\UnaHojaDeCalculo.ods")
Props(0).Name  = "Hidden"  'el documento se abre como "oculto"
Props(0).Value = True
Doc = StarDesktop.loadComponentFromURL(RutaDoc,"_blank",0,Props())
```

Hacerlo visible (*a posteriori , en la misma macro)*

```
Doc.CurrentController.Frame.ContainerWindow.Visible = True
Doc.CurrentController.Frame.ContainerWindow.toFront()
```

Crear una nueva hoja de cálculo

A partir (1) de la plantilla por defecto o (2) de una plantilla especifica.

```
Dim Doc As Object
Dim Props()   'este array no se inicializa  pero es necesario
Modelo = "private:factory/scalc"      '(1)
'o bien
Modelo = ConvertToURL("C:\Ruta\a\UnaPlantillaDeHoja.ots")  '(2)
Doc = StarDesktop.loadComponentFromURL(Modelo,"_blank",0,Props())
```

Guardar un documento

El documento ya existe

(equivale a **Archivo > Guardar**)

Utilice el método `store` del objeto documento. Ej. : `ThisComponent.store`

El documento todavía no se ha guardado

(equivale a **Archivo > Guardar como**)

```
Dim Doc As Object, RutaDoc As String
Dim Props()     'las propiedades de guardado (vacías)
RutaDoc = convertToURL("C:\Ruta\a\UnaHojaDeCálculo.ods")
Doc.storeAsURL(RutaDoc, Props())
```

② Si se guarda como un documento existente, este pasa a ser el documento activo.

Guardar una copia

Como arriba pero con `Doc.storeToURL(RutaDoc, Props())`

② Al guardarlo como copia, la copia **NO** pasa a estar activa, se sigue usando el original

Guardar con contraseña (cualquiera de los casos anteriores)

Añada una propiedad `Password` y asígnele el valor adecuado.

```
Props(n).Name  = "Password" '(n es la enumeración de propiedades...
Props(n).Value = "mypwd"      ' se empieza contando desde 0)
```

Cerrar un documento

Utilice el método close del objeto documento :

`ThisComponent.close(True)`

Información del documento

El objeto documento contiene las propiedades :

`Location`	El directorio de almacenamiento del documento. ☞ Cadena vacía si no se ha guardado.
`DocumentProperties (Object)`	Propiedades complementarias (abajo).

DocumentProperties

`Author`	Nombre del autor.	`ModifyDate`	Fecha última modificación
`CreationDate`	Fecha de creacción.	`Subject`	Asunto (String).
`Description`	Comentarios.	`Title`	Título (String).
`ModifiedBy`	Nombre del usuario que ha modificado el documento.	`UserDefine-dProperties`	Propriedades personalizadas (Objecto).

¿Es una hoja de cálculo?

El objeto `Doc` se refiere al documento (ej. : `Doc = ThisComponent`).

```
CalcOK = Doc.SupportsService("com.sun.star.sheet.SpreadsheetDocument")
```

Calc – Funcionalidades generales

El objeto `Doc` se refiere al documento (ej. : `Doc = ThisComponent`).

Cálculo automático

¿está activado ? (Booleano)	`Auto = Doc.isAutomaticCalculationEnabled`
Désactivar	`Doc.enableAutomaticCalculation(False)`
Activar	`Doc.enableAutomaticCalculation(True)`
Forzar el recálculo de fórmulas	`Doc.calculate` (sólo fórmulas no actualizadas) `Doc.calculateAll` (recalcula todo)

Proteger el libro

¿Libro protegido ? (Booleano)	`Test = Doc.isProtected`
Proteger el documento	`Doc.protect(Contraseña)` [puede estar vacía]
Desproteger el documento	`Doc.unprotect(Contraseña)`

Hojas (*sheets*)

El objeto `Doc` se refiere al documento (ej. : `Doc = ThisComponent`).

Acceder a las hojas

Se puede trabajar con los objetos hojas (se empieza contando por 0) :

Hoja activa	`Hoja = Doc.CurrentController.ActiveSheet`
Lista de hojas	`Hojas = Doc.Sheets`
Número de hojas	`NumHojas = Doc.Sheets.Count`
Objeto hoja (por su índice)	`Hoja = Doc.Sheets(número_índice)`
Objeto hoja (por su nombre)	`Hoja = Doc.Sheets.getByName(Nombre_Hoja)`
Verificar la existencia (nombre)	`Exist = Doc.Sheets.hasByName(Nombre_Hoja)`
Índice de una hoja	`Indice = Hoja.RangeAddress.Sheet`

Modificar las hojas

p es la posición en el documento (índice, empezando por 0).

Añadir una hoja con Nombre	`Doc.Sheets.insertNewByName(Nombre, p)`
Eliminar una hoja	`Doc.Sheets.removeByName(Nombre_Hoja)`
Duplicar una hoja	`Doc.Sheets.copyByName(NomOrigen, NomDestino, p)`
Mover una hoja	`Doc.Sheets.moveByName(Nombre_Hoja, p)`

Gestionar las hojas

Hoja se refiere a un objeto hoja.

Activar una hoja	`Hoja = Doc.CurrentController.ActiveSheet`
Ocultar/Mostrar una Hoja	`Hoja.IsVisible = False 'True`
Comprobar la protección	`Protegida = Hoja.IsProtected`
Proteger una hoja	`Hoja.protect(contraseña)` (la contraseña puede estar vacía)
Desproteter una hoja	`Hoja.unprotect(contraseña)`
Colorear la pestaña	`Hoja.tabColor = RGB(255, 255, 0)`

Enlazar una hoja

Enlazar a un archivo (ej : CSV)	`Hoja.link(URL, "", "Text - txt - csv (StarCalc)", _` `Filtre, com.sun.star.sheet.SheetLinkMode.VALUE)`
Eliminar enlace	`Hoja.setLinkMode(com.sun.star.sheet.SheetLinkMode.NONE)`

Encontrar la última fila/columna utilizada

`Hoja` es el objeto Hoja donde buscar. Fila y `Col` son informaciones buscadas

```
Dim oCur As Object          'cursor sobre la celda
Dim oRango As Object        'el rango usado
Dim Fila As Long, Col As Long
oCur = Hoja.createCursorByRange(Hoja.getCellRangeByName("A1"))
oCur.gotoEndOfUsedArea(True)
oRango = Hoja.getCellRangeByName(oCur.AbsoluteName)
Fila = oRango.RangeAddress.EndRow
Col  = oRango.RangeAddress.EndColumn
```

Celdas (*cells*)

`Celda` se refiere un objeto celda.

Acceder a las celdas

Hoja se refiere a un objeto hoja. Se puede acceder a un objeto celda :

Por sus coordenadas	`Celda = Hoja.getCellRangeByName("A4")`
Por su nombre	`Celda = Hoja.getCellRangeByName("TVA")`
Por sus coordenadas numéricas X, Y (índice 0)	`Celda = Hoja.getCellByPosition(0,3)` `** explicación X=0 (columna A) ; Y=3 (Fila 4)`

Acceder a la celda activa

Doc un objeto documento y `ActiveCel` el objeto celda activa (seleccionada) buscado

```
If Doc.currentSelection.supportsService _
                    ("com.sun.star.sheet.SheetCell") Then
  'comprobamos que sea una celda lo seleccionado
  ActiveCel = Doc.currentSelection
End If
```

Seleccionar una celda

`ThisComponent.CurrentController.select(Cel)`

Coordenadas de una celda

Coordenadas (Object)	`Coord = Cel.CellAddress`
Rango de la hoja (Integer)	`NumH  = Cel.CellAddress.Sheet`
Rango de la columna (Long)	`NumC  = Cel.CellAddress.Column`
Rango de la fila (Long)	`NumF  = Cel.CellAddress.Row`
hoja a que pertenece (Objeto)	`Hoja = Cel.Spreadsheet`
Coordenadas absolutas (String)	`Coord = Cel.AbsoluteName`

(Des)Proteger las celdas

La propiedad `Cel.CellProtection` utiliza valores booleanos :

Proteger contra modificaciones	`CellProtection.IsLocked = True`
Ocultar la fórmula	`CellProtection.IsFormulaHidden = True`
Ocultar la celda	`CellProtection.IsHidden = True`
Evitar impresión de la celda	`CellProtection.IsPrintHidden = True`

Acceder al contenido de una celda

Propiedades

obtener un contenido texto	`MiTexto = Cel.String`
Obtener un contenido numérico	`UnNumero = Cel.Value`
Obtener una fórmula (US)	`LaFormula = Cel.Formula`
Obtener una fórmula (regionalización)	`LaFormula = Cel.FormulaLocal`
Obtener el tipo de contenido	`ElTipo = Cel.Type`
Vaciar una celda	`Cel.String = ""`

Tipo de contenido (propiedad `Type`)

Las constantes `com.sun.star.table.CellContentType.XXX` permiten conocer el tipo de información que contienen las celdas (`Cel.Type`, se muestra abajo) :

`EMPTY`	Celda vacía	`VALUE`	Valor numérico
`TEXT`	Texto	`FORMULA`	Formula

Escribir en una celda

Reemplazar el **texto** existente	`Cel.String = "Cucú !"`
Reemplazar el **valor** existente	`Cel.Value = 1,234`
Reemplazar la fórmula existente	`Cel.Formula = "=AND(A1="SI";A2="NO")"`
Reemplazar la fórmula existente	`Cel.FormulaLocal = "=Y(A1="SI";A2="NO")"`

Rangos (*ranges*)

Rango = conjunto de celdas (puede referirse a una única celda): `Dim Rango As Object`

Acceder a los rangos

Hoja se refiere a un objeto hoja. Accedemos a un objeto rango : Rango

Por sus coordenadas	`Rango = Hoja.getCellRangeByName("C2:G14")`
Por su nombre	`Rango = Hoja.getCellRangeByName("NombreDeRango")`
Por coordenadas (X1, Y1, X2, Y2)	`Rango = Hoja.getCellRangeByPosition(2,1,6,13)`
Arbitrariamente (ej 3.a hoja documento)	`Rango = ThisComponent.Sheets.getCellRangeByPosition (2, 1, 6, 13, 2)`

Acceder al rango activo

De la misma manera que se accede a la celda activa (pero verifiqueantes) `"com.sun.star.sheet.SheetCellRange"` o `"[...].SheetCellRanges"`.

Seleccionar un rango

`ThisComponent.CurrentController.select(MiRango)` `MiRango` es un objeto.

Coordenadas de un rango

Coordenadas (Object)	`Coord = MiRango.RangeAddress`
Rango de la hoja (Integer)	`Rang = MiRango.RangeAddress.Sheet`
Columna de inicio (Long) (ángulo superior / izquierdo)	`NumCHG = MiRango.RangeAddress.StartColumn`
Fila de inicio (Long) (ángulo superior / izquierdo)	`NumLHG = MiRango.RangeAddress.StartRow`
Columna final (Long) (ángulo inferior/derecho)	`NumCBD = MiRango.RangeAddress.EndColumn`
Fila final (Long) (ángulo inferior/derecho)	`NumLBD = MiRango.RangeAddress.EndRow`
Hoja contenedora (Objeto)	`Hoja = MiRango.Spreadsheet`
Coordenadas absolutas (String)	`Coord = MiRango.AbsoluteName`

Rangos con nombre

El objeto `Doc` se refiere al documento. `Dim MisRangos As Object`

Los rangos con nombre	`MisRangos = Doc.NamedRanges`
Número (Long)	`Num = LosRangos.Count`
Acceder a un rango (por índice)	`MiRango = LosRangos(index)`
Verificar existencia de un rango (nombre)	`Existe = LosRangos.hasByName(Nombre)`
Acceder a un rango (por nombre)	`MiRango = LosRangos.getByName(Nombre)`
Añadir un rango	`LosRangos.addNewByName(Nombre,Coord,`
`Coord` : coordenadas del rango	`CellRef.CellAddress, 0)`
`CellRef` : objeto celda de réferencia	
Eliminar un rango (por nombre)	`LosRangos.removeByName(Nombre)`

Borrar el contenido de un rango

Borrar el contenido de MiRango `MiRango.clearContents(ModoBorrar)`

`ModoBorrar` es un valor que determina el tipo de borrado. Las constantes `com.sun.star.sheet.CellFlags.XXX` permiten la elección (combinar con +) :

`ANNOTATION`	Comentarios	`STRING`	Texto
`DATETIME`	Números en formato fecha-hora	`VALUE`	Números (excepto fecha-hora)
`FORMULA`	Fórmulas		

Acceder al contenido de las celdas de un rango

`MiRango.DataArray` est un array de los **valores** de las celdas para `MiRango`.

Copiar el contenido de un rango en otro

dos arrays Origen y destino, **de las mismas dimensiones.**

Copiar el contenido (**valores**) Orig. a Dest. `Destino.DataArray = Origen.DataArray`

Escribir valores en un rango (desde un array a un rango)

`MiRango` es un objeto rango y `Tabla` un array, **de las mismas dimensiones**,

```
Dim Tabla As Variant
Tabla = MiRango.DataArray 'Tabla obtiene las dimensiones del rango
'(transferir los valores a los elementos del array)
MiRango.DataArray = Tabla
```

☞ `.DataArray` puede ser un array anidado : utilice `.DataArray(i)(j)` si es el caso

Recorrer las celdas de un rango

Cree una enumeración desde una colección (`Rango.Celdas`). La cual se recorre llamando a sus propiedades `hasMoreElements` y `NextElement` :

```
Dim Rangos As Object
Rangos =
    ThisComponent.createInstance("com.sun.star.sheet.SheetCellRanges"
    )
Rangos.insertByName("", MiRango)
LEnum = Rangos.Cells.CreateEnumeration
Do While LEnum.hasMoreElements
  Celda = LEnum.NextElement
  'hacer algo en cada objeto Celda
Loop
```

Les celdas vacías **No** se recorren !

Rangos : varios

Fusionar las celdas de MiRango `MiRango.Merge`

Tipos de rangos

Según el modo de acceso a un rango, este implementa uno de los servicios :

① `com.sun.star.sheet.SheetCell` ④ `com.sun.star.sheet.SheetCellRange`
② `com.sun.star.table.CellRange` ⑤ `com.sun.star.sheet.SheetCellRanges`
③ `com.sun.star.sheet.NamedRange`

dependiendo del servicio implementado, los rangos deben ser utilizados de manera **diferente.** Compruebe por medio de su método `supportsService()` (ej. abajo)

¿Rango o celda?

Para conocer el tipo de objeto, compruebe el objeto rango/celda con `supportsService()`:

`If MiObj.supportsService(nomb_servicio)Then…(Reemplace nomb_servicio)`

¿Celda ?	`"com.sun.star.sheet.SheetCell"` ①
¿Rango simple ?	`"com.sun.star.sheet.SheetCellRange"` ④
¿Rango múltiple ?	`"com.sun.star.sheet.SheetCellRanges"` ⑤

compruebe siempre celda **antes** que rango (una celda es también un rango simple)

Filas / Columnas (*rows/columns*)

Filas y columnas son propiedades de los objetos `Sheet` et `Range`.

Generalidades

Filas (Objeto `LasFilas`)	`LasFilas = MiRango.Rows`
Columnas (Objeto `LasColumnas`)	`LasColumnas = MiRango.Columns`
Número de elementos	`NumF = MiRango.Rows.Count`
	`NumC = MiRango.Columns.Count`
Una fila (Objeto `Fila`) (base 0)	`Fila = MiRango.Rows(index)`
Una columna (Objeto `Columna`) (base 0)	`Columna = MiRango.Columns(index)`

Propiedades de filas / columnas

Se aplican a los objetos Fila o Filas / Columna o Columna.

Es Visible u oculta (Boolean)	`IsVisible = True`
Longitud óptima o fija (Boolean)	`OptimalWidth = True`

Insertar / eliminar filas / columnas

Use el objeto Sitio. `PosInicio` y `Num` son la posición inicial y el número de filas / columnas que quiera insertar / eliminar (Long).

Insertar	`Sitio.insertByIndex(PosInicio, Num)`
Borrar	`Sitio.clearContents(Modoborrado)` [vea **Borrar el contenido de un rango**]
Eliminar	`Sitio.removeByIndex(PosInicio, Num)`

Fijar filas / columnas

☞ Sólo en un documento visible.

Use el objeto Controlador : `MiContrlr = ThisComponent.CurrentController`

¿hay alguna fija ?	`Fijas = MonContrlr.hasFrozenPanes`
Fijar (X, Y)	`MiContrlr.freezeAtPosition(1,2)`
Liberar	`MiContrlr.freezeAtPosition(0,0)`

Llamar a una función Calc

Utilice el servicio `"com.sun.star.sheet.FunctionAccess"`

Principio

```
Dim FCalc As Object , Resultado As (según contexto)
Dim Params As (según contexto), NombreFunc As String
FCalc = CreateUnoService("com.sun.star.sheet.FunctionAccess")
Resultado = FCalc.callFunction(Nombre_Función, Parámetros)
```

☞ El nombre, los parámetros y el tipo de resultado dependen de la función elegida.

El nombre de la función debe ser su nombre en **inglés**.
Para averigurarlo, cambie temporalmente la presentación de nombres de funciones mediante **Herramientas ►Opciones ► LibreOffice Calc ► Formula, Usar nombres [...] inglés.**

Ejemplo 1 (función `SUMA()`)

```
Dim FCalc As Object, Resultado As Long
FCalc = CreateUnoService("com.sun.star.sheet.FunctionAccess")
Resultado = FCalc.callFunction("SUM", Array(1, 55, 321, 8))
```

Ejemplo 2 (función `REDONDEAR()`)

```
Dim FCalc As Object, Resultado As Double
Dim Params(1) As Variant
Params(0) = 1,2345 'número que se quiere redondear
Params(1) = 3      '3 decimales
FCalc = CreateUnoService("com.sun.star.sheet.FunctionAccess")
Resultado = FCalc.callFunction("ROUND", Params())
```

Crear una función Calc

Creación

Ejemplo : calculo de la superficie de un trapecio (`S = ((B + b) / 2) × H`)

```
Function SurperficieTrapecio(BMay As Double, BMen As Double, H As
    Double) As Double
  SurperficieTrapecio = ((BMay + BMen) / 2) * H
End Function
```

Uso en Calc

Si `A2` est la base mayor, `A3` la base menor y `A4` la altura, la superficie del trapecio se obtiene insertando la fórmula siguiente en una celda : `=SUPERFICIETRAPECIO(A2;A3;A4)`

☞ La macro recibe los **valores** de los argumentos y no el objeto celda.
☞ La macro **reenvía un valor, No** puede actuar sobre una celda
☞ La función debe colococarse en una biblioteca accesible en el momento de su uso (ej. : `Standard` del documento o del usuario) (sino devolverá el error `#VALOR!`)

Creditos

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

Somos como enanos sentados sobre los hombros de gigantes. Si vemos más cosas y más lejanas que ellos, no es por la perspicacia de nuestra visión, ni por nuestra grandeza, sino porque son ellos los que nos elevan. (Bernard de Chartres [atribuido])

Historial

Versión	Fecha	Comentarios
1.01	01/10/2017	Primera versión.
1.15	14/12/2019	Ajuste de protección. Pequeñas modificaciones.
	07/04/2021	Traducción al español : B. Antonio Fernández

El documento original se puede obtener en la Wiki francesa de publicaciones de L.O.