Guía de iniciación 7.2

Capítulo 13
Inicio con macros

Usar la grabadora de macros… y más

Derechos de autor

Colaboradores

De esta edición

Rafael Lima
Jean Hollis Weber

De ediciones previas

Jean Hollis Weber
Rafael Lima
Steve Fanning
Valerii Goncharuk
Ron Faile Jr.
Felipe Viggiano
Peter Schofield
Martin Fox
Hazel Russman
Kees Kriek
Andrew Pitonyak
Andrew Jensen
Olivier Hallot

De la edición en español

B. Antonio Fernández
Juan C. Sanz Cabrero
José Jesús Palacios
David Mamani Castro

Comentarios y sugerencias

Puede dirigir cualquier clase de comentario o sugerencia acerca de este documento a la lista de correo del equipo de documentación: documentation@es.libreoffice.org.

Nota

Todo lo que envíe a la lista de correo, incluyendo su dirección de correo y cualquier otra información personal que escriba en el mensaje se archiva públicamente y no puede ser borrada

Fecha de publicación y versión del programa

Versión en español publicada en septiembre de 2022. Basada en la versión de LibreOffice.

Uso de LibreOffice en macOS

Algunas pulsaciones de teclado y opciones de menú son diferentes en macOS de las usadas en Windows y Linux. La siguiente tabla muestra algunas sustituciones comunes para las instrucciones dadas en este capítulo. Para una lista detallada vea la ayuda de la aplicación.

Contenido

Introducción

Una macro es un conjunto de instrucciones que realizan una serie de acciones en un documento. Estas instrucciones se pueden guardar para realizar las mismas acciones en otro documento y así evitarnos tener que hacer tareas repetitivas en documentos que requieran estas mismas acciones. Una macro sencilla puede constar de unas instrucciones que ingresan su dirección en un documento abierto.

Puede usar macros para automatizar tareas sencillas o complejas. Las macros son muy útiles cuando hay que repetir una misma tarea de igual forma.

La forma más sencilla de crear una macro es mediante la *Grabadora de macros* a través de la interfaz de usuario de LibreOffice. La grabadora de macros utiliza el lenguaje de secuencias de comandos de código abierto *LibreOffice Basic*, que es una implementación del conocido lenguaje de programación BASIC. Estas macros se pueden editar y mejorar después de su grabación en el entorno de desarrollo integrado (IDE, por sus siglas en ingles) de LibreOffice.

Las macros para tareas más complejas se crean escribiendo directamente el código en uno de los cuatro lenguajes de secuencias de comandos admitidos por LibreOffice (Basic, Python, JavaScript y BeanShell).

Este capítulo proporciona una descripción general de las funciones de macros de LibreOffice, se centra principalmente en su lenguaje de secuencias de comandos predeterminado: *LibreOffice Basic*. Se incluyen algunos ejemplos introductorios para los lenguajes Python, BeanShell y JavaScript. Sin embargo, una descripción más avanzada sobre el uso de estos lenguajes para la creación de macros está más allá de la intención de este capítulo.

Importante

Antes de continuar con la lectura de este capítulo, tiene que conocer unas normas básicas sobre la nomenclatura y la sintaxis que se debe emplear en las *bibliotecas*, *módulos*, *subrutinas* y *variables* al escribir código Basic. Sin esta consideración sus macros fallarán porque Basic no interpretará correctamente las instrucciones.

– **Ningún nombre puede empezar por un número**: Al ver un número en primer lugar, Basic lo interpreta como el inicio de una operación matemática.

– **Los nombres no pueden contener espacios**: Basic interpretará cada palabra como un elemento aislado.

– **Los nombres no pueden tener tildes**: Basic no está preparado.

La primera macro

Crear una macro

Un primer paso para aprender a programar macros es encontrar y utilizar macros existentes. Esta sección asume que tiene una macro que desea usar, que puede haber encontrado en un libro o en Internet. El listado 1 puede servir de ejemplo.

Para albergar su primera macro primero debe crear una biblioteca y un módulo para contener la macro; en los siguientes pasos verá como se hace esto. Para una información más detallada sobre bibliotecas y módulos, consulte «Organización de las macros» más adelante.

Listado 1: Macro sencilla que muestra un mensaje de saludo

```
Sub MacroSaludo
  Print "Hola"
End Sub
```

Siga los siguientes pasos para crear una librería y un módulo que contenga la macro:

1. Abra cualquier aplicación de LibreOffice.
2. Vaya a **Herramientas > Macros > Organizar macros > Basic** para abrir el diálogo *Macros de Basic* (figura 1).

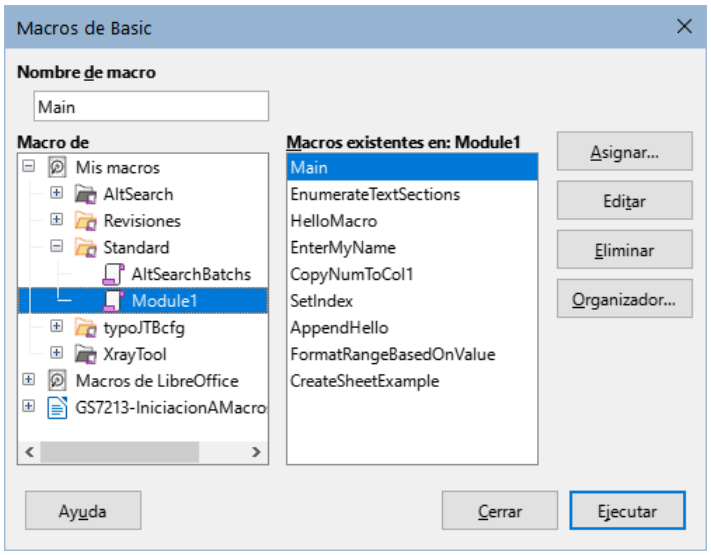


Figura 1: Diálogo Macros de Basic

3. Haga clic en el botón *Organizador* para abrir el diálogo *Organizador de macros de BASIC* (figura 2) y seleccione la pestaña *Bibliotecas*.

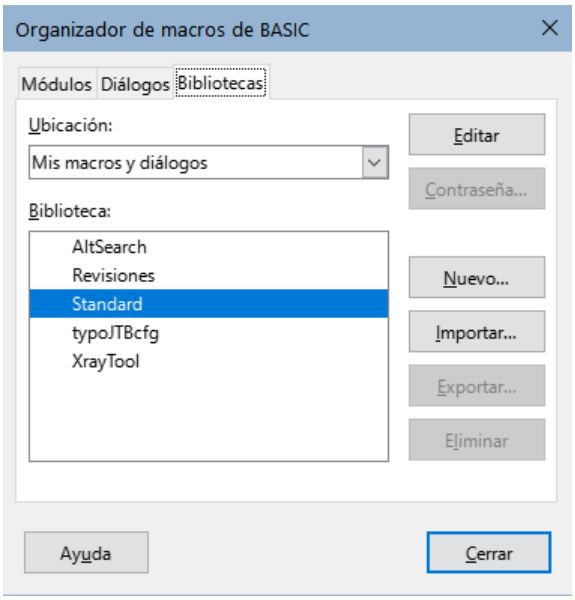


Figura 2: Pestaña Bibliotecas del diálogo Organizador de macros de BASIC

4. En el desplegable *Ubicación*, seleccione *Mis macros y diálogos*.
5. Haga clic en *Nuevo* para abrir el diálogo *Biblioteca nueva* (figura 3).

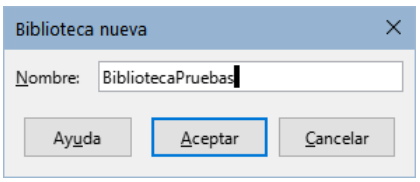


Figura 3: Biblioteca nueva

6. Escriba el nombre de biblioteca, *BibliotecaPruebas*, y haga clic en *Aceptar*.
7. Seleccione la pestaña *Módulos* del diálogo *Organizador de macros de BASIC*, (figura 4).

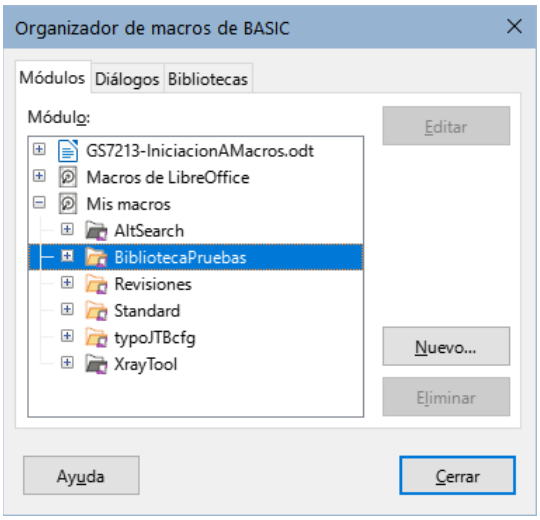


Figura 4: Pestaña Módulos del diálogo Organizador de macros de BASIC

8. En la lista de módulos, expanda *Mis macros* y seleccione la biblioteca *BibliotecaPruebas*. Verá que ya se ha creado un módulo llamado *Módule1* para contener la macro. Si lo desea, puede crear ahora otro módulo en la biblioteca pulsando *Nuevo*.
9. Seleccione *Module1* y haga clic en *Editar* para abrir el entorno de desarrollo integrado (IDE) (figura 5). El IDE es un editor de texto con funciones asociadas que está integrado en LibreOffice y le permite crear, editar, ejecutar y depurar macros.

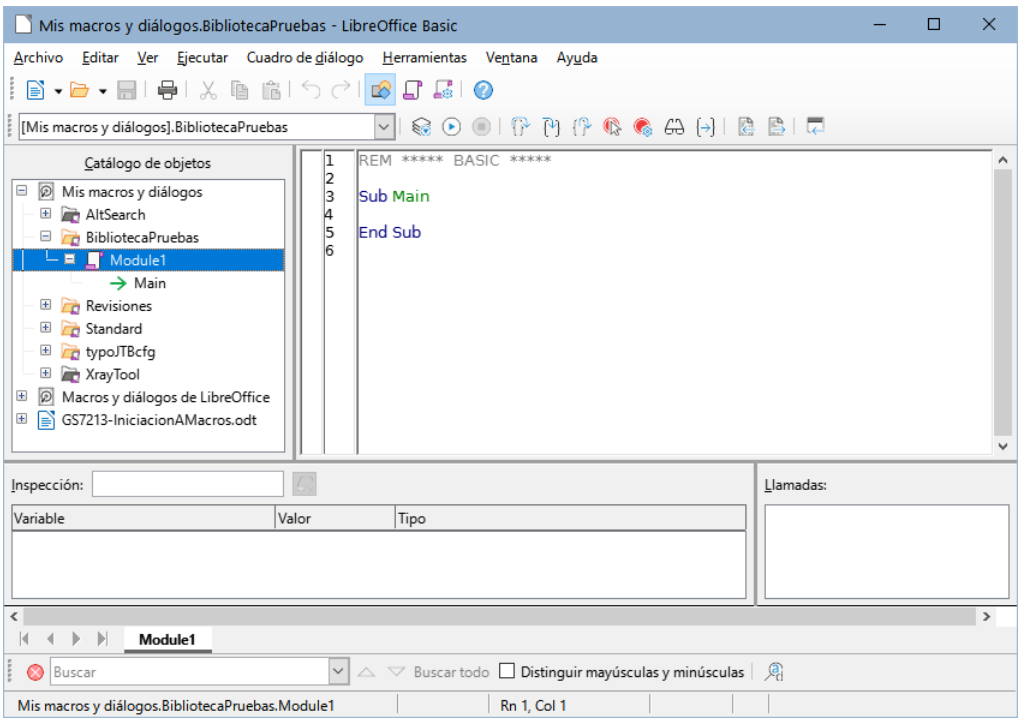


Figura 5: Ventana del IDE de BASIC (Integrated Development Environment)

10. Al crear un módulo nuevo, se incluye un comentario y una macro vacía llamada *Main*, que realmente no hace nada.
11. Agregue la macro del listado 1 antes de `Sub Main` o después de `End Sub`. El listado 2 muestra el contenido del módulo con la macro agregada.

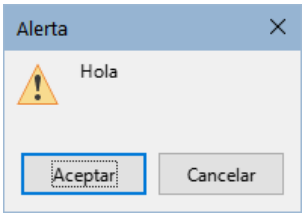
Listado 2: Module1 después de añadir la macro

```
REM  *****  BASIC  *****

Sub MacroHola
  Print "Hola"
End Sub

Sub Main

End Sub
```

Sugerencia

Si lo desea, puede eliminar el intervalo de líneas de código `Sub Main` … `End Sub` del módulo.

12. (Opcional) Para comprobar si la macro está escrita como espera el lenguaje Basic, haga clic en el icono *Compilar* en la barra de herramientas *Macro*.
13. En el panel *Catálogo de objetos* de la **BibliotecaPruebas > Module1** Haga doble clic en la subrutina `MacroHola` y haga clic en el ícono *Ejecutar* en la barra de herramientas *Macro* o presione la tecla *F5*, para ejecutar la subrutina. Se abrirá un pequeño diálogo con la palabra «Hola» como muestra la ilustración.



14. Haga clic en *Aceptar* para cerrar este diálogo.

Si no se selecciona ninguna subrutina o función, se abrirá un diálogo como el de la figura 6, seleccione la macro y haga clic en *Ejecutar*.

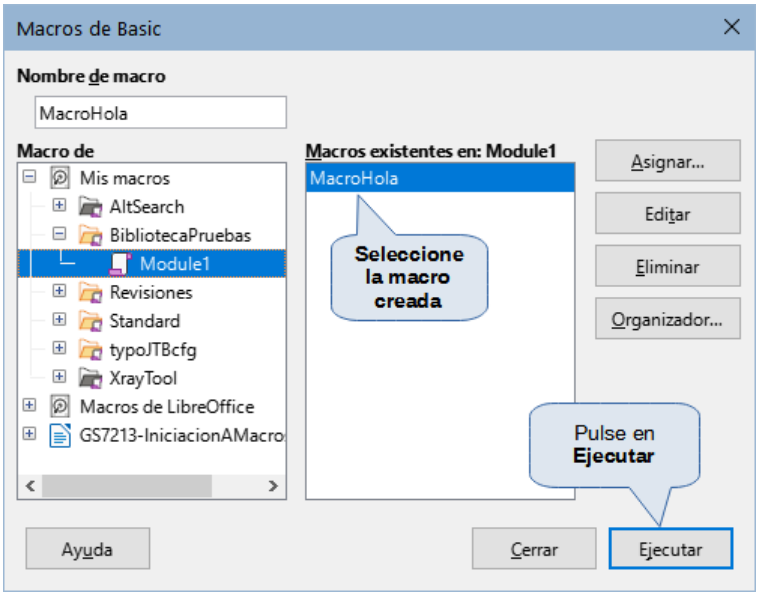


Figura 6: Diálogo para seleccionar y ejecutar macros

Para ejecutar esta o cualquier otra macro del módulo, haga clic en el icono *Seleccionar macro* en la barra de herramientas *Estándar* del IDE o vaya a **Herramientas > Macros > Organizar macros > BASIC**, seleccione una macro y luego haga clic en *Ejecutar*.

Grabar una macro

Cuando graba una macro en LibreOffice, en realidad está grabando los pasos para realizar una determinada tarea utilizando el lenguaje de programación Basic.

Por ejemplo, considere que tiene que ingresar repetidamente la misma información en varias partes de un documento. Puede copiar esta información después de haberla escrito por primera vez, y luego pegarla en el documento cada vez que quiera usarla. Si durante su tarea copia algo más, el contenido del portapapeles se cambia y no pegará la primera información copiada. Tendrá que volver a copiar la primera información para luego pegarla en el documento.

Para superar este problema, puede crear una macro que ingrese esta información sin tener que copiarla cada vez que la necesite.

Nota

En ciertas ocasiones, cuando desee introducir repetidamente una información en un documento, puede ser más conveniente crear un *Texto automático*. Consulte el «Capítulo 2, Trabajar con texto I» de la *Guía de Writer* para más información.

Asegúrese de que la grabación de macros esté habilitada: Vaya a **Herramientas > Opciones > LibreOffice > Avanzado** y marque la opción *Habilitar grabación de macros* en la sección *Funcionalidades opcionales*. Esta opción está desactivada de manera predeterminada.

1) Vaya a **Herramientas > Macros > Grabar macro** para comenzar a grabar una macro.
2) Se muestra un pequeño diálogo con el botón *Finalizar grabación* que indica que LibreOffice está grabando una macro.
3) Al hacer clic en *Finalizar grabación* se abrirá el diálogo *Macros de Basic* (similar al de la figura 1 anterior, pero con diferentes botones de acción).
4) Abra el contenedor de bibliotecas *Mis macros*.
5) Busque la biblioteca denominada *Standard* en *Mis macros*. Tenga en cuenta que todos los contenedores de biblioteca pueden tener una biblioteca con ese mismo nombre.
6) Seleccione la biblioteca *Standard* y luego elija un módulo existente en el que guardar la macro. Alternativamente, puede hacer clic en *Nuevo módulo* para crear un nuevo módulo que contenga la macro recién grabada.
7) Escriba un nombre para la macro que acaba de grabar en el cuadro de texto *Nombre de macro* en la sección superior izquierda del diálogo, en este caso, `EscribirMiNombre`.
8) Haga clic en *Guardar* para guardar la macro y cerrar diálogo *Macros de Basic*.

Si siguió los pasos anteriores, se habrá creado una macro llamada `EscribirMiNombre` dentro del módulo seleccionado.

Nota

Siempre que se crea un nuevo módulo se añade automáticamente una subrutina vacía con nombre `Main`.

Ejecutar una macro

1) Vaya a **Herramientas > Macros > Ejecutar macro** para abrir el diálogo *Selector de macros* (figura 7).
2) Seleccione la macro `EscribirMiNombre` recién creada y haga clic en *Ejecutar*.
3) Como alternativa, vaya a **Herramientas > Macros > Organizar macros > Basic** para abrir el diálogo *Macros de Basic* (figura 1), seleccione la macro y haga clic en *Ejecutar*.

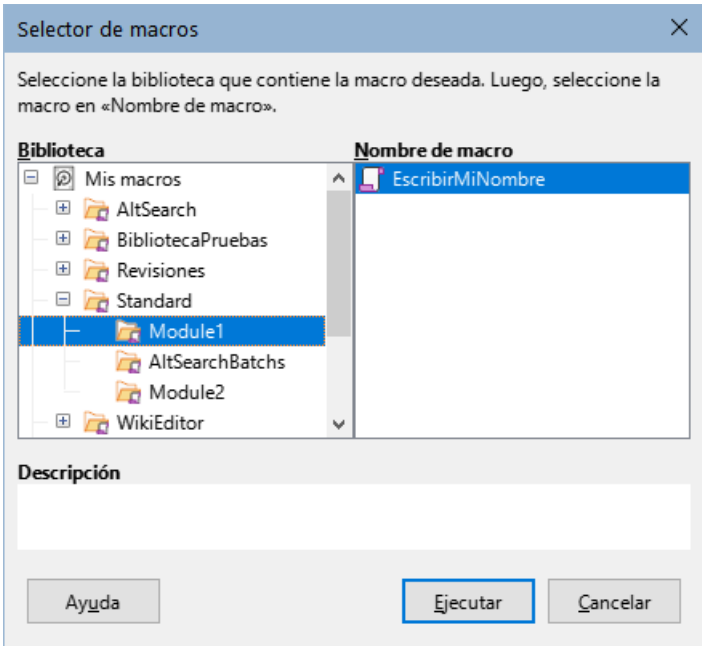


Figura 7: Diálogo Selector de macros

Ver y editar macros

Para ver o editar la macro que creó:

1. Vaya a **Herramientas > Macros > Organizar macros > Basic** para abrir el diálogo *Macros de Basic*.
2. Seleccione la macro `EscribirMiNombre` y haga clic en *Editar*. Se abrirá el IDE de Basic, mostrando el código de la macro (listado 3).

Esta primera macro no es complicada. Una pequeña explicación le ayudará significativamente a comprender las macros. La explicación comienza con la segunda línea de la macro y describe las funciones a lo largo del listado.

Listado 3: Código resultante de la grabación de la macro

```
Sub EscribirMiNombre
Rem ----------------------------------------------------------------
Rem Definir variables
 dim document as object
 dim dispatcher as object
Rem ----------------------------------------------------------------
Rem Obtener acceso al documento
 document = ThisComponent.CurrentController.Frame
 dispatcher = createUnoService("com.sun.star.frame.DispatchHelper")
Rem ----------------------------------------------------------------
 dim args1(0) as new com.sun.star.beans.PropertyValue
 args1(0).Name = "Text"
 args1(0).Value = "Escriba aquí su Nombre"
 dispatcher.executeDispatch(document,".uno:InsertText","",0,args1())
End Sub
```

Comentarios en el código (REM)

Todos los comentarios en el código de macros Basic empiezan con `REM`, (del inglés *remark*). Al ejecutar la macro, el intérprete Basic omite las líneas que empiezan por `REM`.

También puede usar la comilla sencilla o apóstrofo (') para iniciar un comentario.

LibreOffice Basic no distingue entre mayúsculas y minúsculas para las palabras clave o nombres constantes o variables, por lo tanto `REM`, `Rem` o `rem` son válidas para comenzar un comentario. Si utiliza constantes simbólicas pertenecientes a la API (interfaz de programación de aplicaciones), es más seguro el uso correcto de mayúsculas y minúsculas en su escritura.

El manejo de constantes simbólicas es un tema avanzado que no se trata en esta guía y no son necesarias cuando se emplea la *Grabadora de macros*.

Definición de subrutinas con SUB

Las secuencias de comandos se almacenan en subrutinas y estas subrutinas comienzan con la palabra clave `SUB`. El final de una subrutina se indica con las palabras `END SUB`. El código comienza definiendo la subrutina denominada `Main`, que está vacía y, por tanto, no hace nada. Observe como el código del listado 3 para la macro `EscribirMiNombre` comienza con la palabra clave `SUB` y termina con `END SUB`.

Los temas avanzados enumerados a continuación, están más allá del alcance de esta guía del usuario, pero conocerlos puede ser de interés:

- Puede escribir subrutinas que acepten valores de entrada para su uso dentro de la macro. Estos valores se denominan argumentos. Esto solo se puede hacer cuando crea subrutinas desde cero.

- Otro tipo de subrutina se denomina función, que es una subrutina que puede devolver un valor como resultado de su trabajo. Las funciones empiezan con la palabra clave `FUNCTION` y acaban con `END FUNCTION`
  Las macros creadas con la *Grabadora de macros* solo generan subrutinas y estas no aceptan argumentos.

Declaración de variables usando DIM

Piense en una variable como si escribiese una nota en un papel para poder verla más tarde. Una variable contiene información que se puede cambiar y leer. La palabra clave `Dim` originalmente significaba «Dimensión» y se usaba para definir las dimensiones de una matriz. La declaración de variables `Dim` utilizada en la macro es similar a reservar una nota.

En la macro `EscribirMiNombre` se emplean tres variables: las variables `document` y `dispatcher` se definen como de tipo objeto. (Otros tipos de variables comunes son: cadenas de texto, números y fechas). La tercera variable, llamada `args1`, es una matriz de propiedades y sus valores. Una variable de tipo matriz permite que una sola variable contenga múltiples valores, similar a tener varias páginas en un cuaderno.

Al definir una matriz, el número entre paréntesis indica el número máximo de elementos que va a contener la matriz. Los elementos de una matriz generalmente se numeran a partir de cero. En este ejemplo, nada más hay un valor y por eso está numerado como cero.

Explicación del código de la macro EscribirMiNombre

A continuación se explica cada línea de código de la subrutina `EscribirMiNombre`. Es posible que no comprenda todos los detalles, pero puede darle una idea de cómo funciona una macro.

```
Sub EscribirMiNombre
```

Define el inicio de la subrutina.

```
Dim document as object
```

Crea o declara la variable de nombre `document` y de tipo objeto. Los objetos son un tipo de variable específico con múltiples campos (o propiedades) y acciones (o métodos). Los campos se pueden entender como variables (que pueden contener otros objetos) y las acciones como subrutinas que nos permiten operar con el objeto.

Nota

En algunas ocasiones, puede encontrar la palabra `service`. Un servicio es un tipo especial de objeto que contiene instrucciones de cómo se utilizará.

```
Dim dispatcher as object
```

Declara `dispatcher` como una variable de tipo objeto.

```
document = ThisComponent.CurrentController.Frame
```

Asigna un valor a la variable `document`, en este caso es una propiedad del documento.

`ThisComponent` es un objeto creado por LibreOffice que se refiere al documento en uso.

`CurrentController` es una propiedad que se refiere al servicio que controla el documento. Por ejemplo, cuando escribe, toma nota de lo que escribe y envía los cambios al `Frame` (marco) del documento.

`Frame` es una propiedad de `CurrentController` que devuelve el marco principal de un documento.

Por tanto, la variable `document` se refiere al marco del documento que recibe los comandos enviados por el servicio `dispatcher`.

```
dispatcher = createUnoService("com.sun.star.frame.DispatchHelper")
```

La mayoría de las tareas en LibreOffice se llevan a cabo enviando un comando. LibreOffice incluye un servicio de ayuda para el envío de comandos (`DispatchHelper`), que proporciona una manera fácil de enviar datos en una sola llamada, en lugar de tener que hacerlo con múltiples llamadas. Es bastante utilizado porque realiza la mayor parte del trabajo en las macros. El método `CreateUnoService` acepta el nombre de un servicio e intenta crear una instancia de ese servicio. La variable `dispatcher` contiene una referencia al servicio `DispatchHelper`.

```
Dim args1(0) as new com.sun.star.beans.PropertyValue
```

Declara la variable `args1` como una matriz de propiedades. Cada propiedad tiene un nombre y un valor emparejado. La matriz creada en este caso tiene un único elemento (índice 0).

La expresión `com.sun.star.beans.PropertyValue` es una estructura UNO (siglas en inglés para *Universal Network Objects*). Las estructuras son un tipo especial de variable que contienen otras variables unidas lógicamente; son apropiadas para operar con conjuntos heterogéneos de información que deben ser considerados como una unidad. Para más información sobre la creación y uso de estructuras, vea la Ayuda del programa y otras guías de BASIC.

```
args1(0).Name = "Text"
args1(0).Value = "Escriba aquí su Nombre"
```

Asigna la palabra `"Text"` a la propiedad `Name` del elemento 0 de la matriz y le proporciona el valor `"Este es mi Nombre"` a la propiedad `Value`, que es el texto que se insertará al ejecutar la macro.

```
dispatcher.executeDispatch(document,".uno:InsertText","",0,args1())
```

El servicio de ayuda (`DispatchHelper`) envía al marco del documento (propiedad `Frame` del documento establecida en la variable `document`) mediante el comando `.uno:InsertText`. Los siguientes argumentos: nombre del marco (que aparece en este ejemplo vacío) y las banderas de búsqueda (0), están fuera del alcance de esta guía. El último argumento son los valores de la matriz que se utilizarán al ejecutar el comando `InsertText`.

En otras palabras, esta línea de código ejecutará el comando `.uno:InsertText` que escribirá la cadena de texto `"Escriba aquí su Nombre"` en el documento.

```
End Sub
```

Es la última línea de código con la que se termina la subrutina y detiene la ejecución.

Crear una macro

Al crear una macro mediante la *Grabadora de macro*, es importante que se plantee dos cuestiones importantes antes de grabarla:

1) ¿Se puede escribir esta tarea como un simple conjunto de comandos?
2) ¿Se pueden ordenar los pasos de manera que el último comando sitúe el cursor en el documento para el siguiente comando, o para la entrada de texto o datos?

Un ejemplo más complejo de una macro

Una tarea común es copiar una tabla con datos desde un sitio web, y modificarlos a nuestro antojo para presentarlos en un documento de texto con un orden y formato distintos. Se puede crear una macro para automatizar esta tarea de la siguiente manera:

Preparación de los datos

1) Copie la tabla de datos de la web al portapapeles.
2) Elimine el formato de tipos de letra y obtenga solo el texto (eliminando también la tabla), pegando el texto en un documento de Writer como texto sin formato y si es necesario, haga un formateo básico para conseguir homogeneidad en las líneas.
3) Active las marcas de formato en la barra de herramienas *Estándar* para ver claramente los espacios y tabulaciones que contienen las líneas.

Planteamiento de la macro

Con las dos preguntas planteadas al inicio en mente, examine el resultado obtenido para ver si puede grabar una macro dando formato al texto. En la figura 4 se muestra un ejemplo de información pegada en la que se muestran las constantes para el grosor del tipo de letra que utiliza la API de LibreOffice.

Ejemplo de datos copiados y pegados como texto sin formato

DONTKNOW	The font weight is not specified/known.
THIN	specifies a 50% font weight.
ULTRALIGHT	specifies a 60% font weight.
LIGHT	specifies a 75% font weight.
SEMILIGHT	specifies a 90% font weight.
NORMAL	specifies a normal font weight.
SEMIBOLD	specifies a 110% font weight.
BOLD	specifies a 150% font weight.
ULTRABOLD	specifies a 175% font weight.
BLACK	specifies a 200% font weight.

En cada línea de este ejemplo, tenemos un nombre de constante, seguido por un espacio y una tabulación, la descripción de la constante y al final de cada línea dos espacios.

Crearemos una macro para modificar el texto de manera que la primera columna de la tabla final contenga el valor numérico del porcentaje del grosor del tipo de letra, la segunda columna el nombre de la constante y la tercera columna su descripción. Esta conversión se logra fácilmente para cada fila excepto para las constantes `DONTKNOW` y `NORMAL`, que no contienen un valor numérico.

Grabación de la macro

A continuación se muestran los pasos para grabar esta macro usando pulsaciones de teclado.

1) Asegúrese de que la grabación de macros está habilitada: vaya a **Herramientas > Opciones > LibreOffice > Avanzado** en el menú y seleccione la opción **Activar grabación de macros (limitada)**.
2) Vaya al menú **Herramientas > Macros > Grabar macro** para iniciar la grabación.
3) Sitúese al inicio de la línea que contiene el nombre de constante THIN.
4) Pulse *Ctrl+Flecha derecha* para mover el cursor al inicio de la descripción (*specifies*).
5) Pulse *Retroceso* dos veces para eliminar la tabulación y el espacio.
6) Pulse *Tab* para agregar la tabulación (sin espacio) tras el nombre de la constante.
7) Pulse *Supr* para borrar la ese minúscula y luego Pulse *Mayús+S* para escribir una ese mayúscula.
8) Pulse *Ctrl+Flecha derecha* dos veces para mover el cursor al inicio del número.

9) Pulse *Ctrl+Mayús+Flecha derecha* para mover el cursor a la derecha, lo que seleccionará solo el valor numérico.
10) Pulse *Ctrl+C* para copiar el número sin el porcentaje al portapapeles.
11) Pulse *Fin* para mover el cursor al final de la línea.
12) Pulse *Retroceso* dos veces para eliminar los dos espacios del final.
13) Pulse *Inicio* para mover el cursor al inicio de la línea.
14) Pulse *Ctrl+V* para pegar el número seleccionado al inicio de la línea.
15) Al pegar el valor, también se pega un espacio extra, así que Pulse *Retroceso* para eliminar este espacio sobrante.
16) Pulse *Tab* para insertar una tabulación entre el número y el nombre.
17) Pulse *Inicio* para mover el cursor al inicio de la línea.
18) Pulse *Flecha abajo* para mover el cursor a la línea siguiente.
19) Detenga la grabación de la macro y guárdela. Vea «Grabar una macro» más atrás.

Lleva más tiempo leer y seguir estos pasos que grabar la macro. Trabaje lentamente y ponga atención en los pasos mientras los ejecuta. Con la práctica aprenderá cómo organizar los pasos para grabar macros de manera eficaz.

En el listado 4 se observa el código generado por la grabación al que se han agregado comentarios que muestran los pasos seguidos en el ejemplo.

Listado 4: Macro para modificar los datos copiando los valores numéricos al inicio de la línea

```
sub CopiarNumeroalInicio
Rem ----------------------------------------------------------------
Rem definir las variables
dim document as object
dim dispatcher as object
Rem ----------------------------------------------------------------
Rem obtener acceso al documento y crear el servicio de ayuda
document = ThisComponent.CurrentController.Frame
dispatcher = createUnoService("com.sun.star.frame.DispatchHelper")

Rem (3) mover el cursor al inicio de la siguiente palabra (specifies).
dispatcher.executeDispatch(document,".uno:GoToNextWord","",0,Array())

Rem (4) Eliminar la tabulación.(tecla retroceso).
dispatcher.executeDispatch(document,".uno:SwBackspace","",0,Array())

Rem ...Eliminar el espacio (con otro retroceso).
dispatcher.executeDispatch(document,".uno:SwBackspace","",0,Array())

Rem (5) Insertar una tabulación (sin espacio) después del nombre de constante.
dim args4(0) as new com.sun.star.beans.PropertyValue
args4(0).Name = "Text"
args4(0).Value = CHR$(9)

dispatcher.executeDispatch(document,".uno:InsertText","",0,args4())

Rem (6) Borrar la ese minúscula
dispatcher.executeDispatch(document,".uno:Delete","",0,Array())
```

```
Rem (6) Añadir una ese mayúscula.
dim args6(0) as new com.sun.star.beans.PropertyValue
args6(0).Name = "Text"
args6(0).Value = "S"

dispatcher.executeDispatch(document,".uno:InsertText","",0,args6())

Rem (7) Mover el cursor (dos palabras) a la derecha (inicio del número).
dispatcher.executeDispatch(document,".uno:GoToNextWord","",0,Array())

Rem ----------------------------------------------------------------
dispatcher.executeDispatch(document,".uno:GoToNextWord","",0,Array())

Rem (8)  Seleccionar el número.
dispatcher.executeDispatch(document,".uno:WordRightSel","",0,Array())

Rem (9) Copiar el número al portapapeles.
dispatcher.executeDispatch(document,".uno:Copy","",0,Array())

Rem (10) Mover el cursor al final de la linea.

dispatcher.executeDispatch(document,".uno:GoToEndOfLine","",0,Array())

Rem (11) Eliminar los dos espacios sobrantes (tecla retroceso).
dispatcher.executeDispatch(document,".uno:SwBackspace","",0,Array())

Rem ----------------------------------------------------------------
dispatcher.executeDispatch(document,".uno:SwBackspace","",0,Array())

Rem (12)  Mover el cursor al inicio de la línea.
dispatcher.executeDispatch(document,".uno:GoToStartOfLine","",0,Array())

Rem (13) Pegar el número del portapapeles al inicio de la línea.
dispatcher.executeDispatch(document,".uno:Paste","",0,Array())

Rem (14) Eliminar el espacio sobrante (tecla retroceso).
dispatcher.executeDispatch(document,".uno:SwBackspace","",0,Array())

Rem (15) Insertar un tabulador entre el número y el nombre.
dim args17(0) as new com.sun.star.beans.PropertyValue
args17(0).Name = "Text"
args17(0).Value = CHR$(9)

dispatcher.executeDispatch(document,".uno:InsertText","",0,args17())
Rem (16) Volver al inicio de la línea.
dispatcher.executeDispatch(document,".uno:GoToStartOfLine","",0,Array())

Rem (17) Bajar a la linea siguiente (sitúa el cursor para una nueva ejecución)
dim args19(1) as new com.sun.star.beans.PropertyValue
args19(0).Name = "Count"
args19(0).Value = 1
args19(1).Name = "Select"
```

```
args19(1).Value = false

dispatcher.executeDispatch(document,".uno:GoDown","",0,args19())
end sub
```

Ejecución de la macro

Para ejecutar esta macro, primero coloque el cursor al inicio de la línea a la que la quiere aplicar, vaya a **Herramientas > Macros > Ejecutar**, seleccione la macro `CopiarNumeroalInicio` y pulse *Ejecutar*, como se describe en la figura 6.

La figura 8 muestra la línea original y el resultado después de ejecutar la macro.

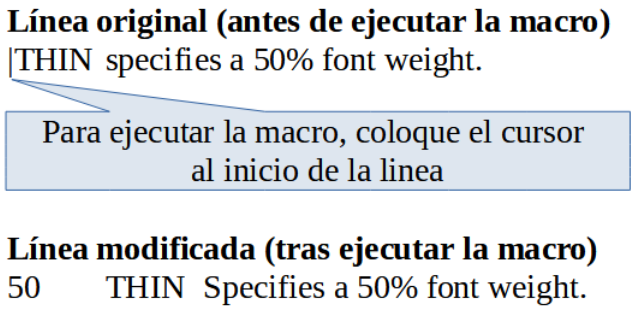


Figura 8: Resultado al aplicar la macro grabada

Tenga en cuenta que las instrucciones descritas solo funcionarán correctamente si la línea tiene la estructura que habíamos asumido al crear la macro. Si ejecuta esta macro en las líneas `DONTKNOW` y `NORMAL`, el resultado no será el esperado porque estas líneas tienen una estructura diferente. La figura 9 muestra el resultado de ejecutar la macro en la línea DONTKNOW.

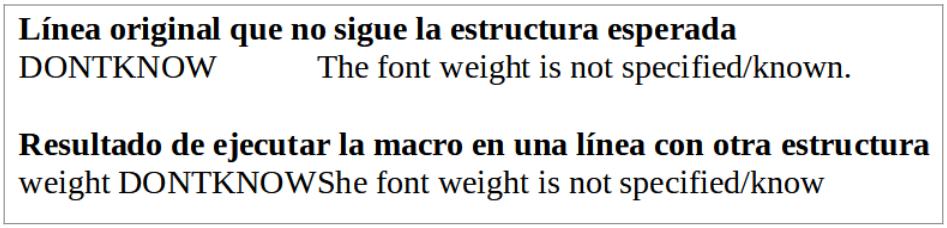


Figura 9: Resultado en una linea con una estructura diferente

Esta es una macro sencilla que se tiene que ejecutar para cada línea del texto. Para formatear la tabla completa de una pasada habría que editar la macro y modificarla.

Asignar un atajo de teclado a una macro

Ejecutar una macro que necesita utilizar frecuentemente usando **Herramientas > Macros > Ejecutar macro** es bastante tedioso. Puede asignar un atajo de teclado para ejecutarla sin tener que seguir esta secuencia de menú. Los siguientes pasos le permiten asignar el acceso directo *Ctrl+K* a la macro creada.

1) Vaya a **Herramientas > Personalizar** y seleccione la pestaña *Teclado* en el diálogo *Personalizar*.
2) Seleccione *Writer* en las opciones de la parte superior derecha.
3) Seleccione *Ctrl+K* u otra combinación de teclas no asignada en la sección *Atajos de teclado*
4) Busque *Macros de LibreOffice* en la sección *Categoría* y pulse en el signo **+** de la izquierda para abrir el listado de bibliotecas y macros disponibles.
5) Seleccione la biblioteca *BibliotecaPruebas* y en la sección *Función* seleccione la macro creada.

6) Pulse en el botón *Modificar* para guardar los cambios.
7) Pulse *Aceptar* para cerrar el diálogo *Personalizar*.

Ahora podrá ejecutar la macro `CopiarNumeroalInicio` usando el atajo de teclado *Ctrl+K*. Esta es una manera mucho más rápida de ejecutar una macro. La figura 10 ilustra los pasos descritos anteriormente.

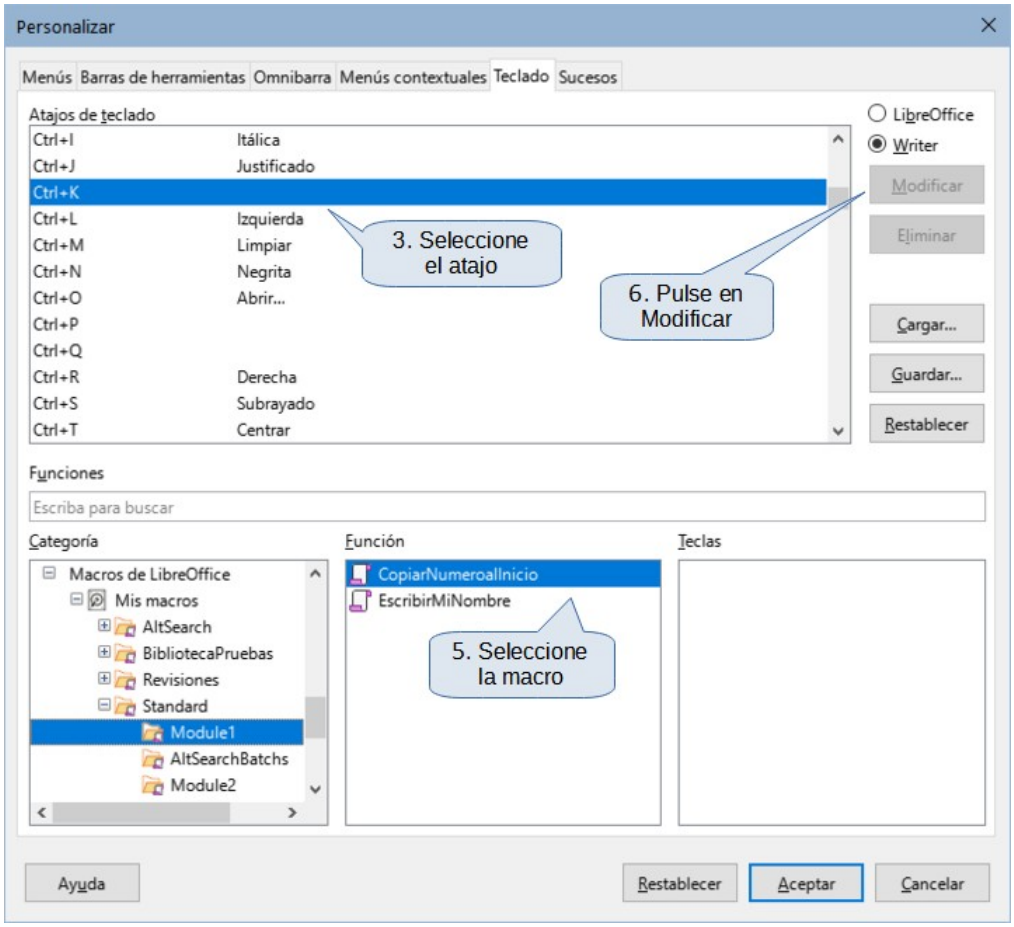


Figura 10: Asignar un atajo de teclado a una macro

Limitaciones de la grabadora de macros

La grabadora de macros tiene algunas limitaciones, por lo que es posible que algunas acciones no se graben. Un conocimiento más profundo del funcionamiento interno de LibreOffice le ayudará a comprender cómo funciona y cuando dejará de funcionar la grabadora de macros.

La causa principal de estas limitaciones es el *Dispatch framework* (sistema de envío de comandos) y su relación con la grabadora de macros.

Dispatch framework

El propósito de este sistema de envío es permitir el acceso uniforme de los componentes (documentos) a los comandos que normalmente se corresponden con elementos de menú. Tanto el comando **Archivo > Guardar** del menú como el atajo de teclado *Ctrl+G* o la pulsación en el icono *Guardar* de la barra de herramientas *Estándar* se traducen en el mismo comando del *Dispatch framework*.

Mediante este sistema también se pueden enviar comandos de regreso a la interfaz de usuario (IU). Por ejemplo, después de guardar un documento nuevo, actualizar la lista de documentos recientes.

Los comandos del *Dispatch framework*, son cadenas de texto similares a `.uno:InsertObject` o `.uno:GoToStartOfLine`. El comando se envía al marco del documento y este pasa el comando hasta que encuentra un objeto que puede manejar dicho comando.

Cómo usa la grabadora de macros el sistema *Dispatch framework*

La grabadora de macros graba los comandos generados por nuestras acciones en el programa. La grabadora es una herramienta relativamente sencilla de usar y los comandos que generamos se graban para un uso posterior. El problema es que no todos los comandos capturados están completos. Por ejemplo, insertar un objeto en un documento genera el siguiente código:

```
dispatcher.executeDispatch(document,".uno:InsertObject","",0,Array())
```

La grabadora no puede especificar que clase de objeto se va a crear o insertar. Si se inserta un objeto como un archivo, no puede especificar qué archivo se insertará.

Si durante la grabación de una macro usa **Herramientas > Opciones** desde el menú para abrir y modificar elementos de configuración, la macro generada no grabará ningún cambio en la configuración y el código generado aparecerá como comentario con lo que no se ejecutará.

```
Rem dispatcher.executeDispatch(document,".uno:OptionsTreeDialog","",0,Array())
```

Si se abre un diálogo, es probable que se genere el comando para abrir el diálogo. Pero normalmente, las opciones modificadas en el diálogo no se graban. Algunos ejemplos incluyen los diálogos de organización de macros, el de inserción de caracteres especiales, etc.

Pueden aparecer otros problemas al usar la grabadora de macros al insertar una fórmula, establecer los datos de usuario, establecer filtros en Calc, acciones en formularios de bases de datos y exportación de un documento a un archivo PDF cifrado. Nunca podrá saber con certeza qué funcionará, a menos que lo intente.

Como dato curioso, las acciones del diálogo **Buscar** se capturan correctamente.

Otras opciones para crear macros

Cuando la grabadora de macros no puede resolver un problema específico, la solución habitual es escribir el código usando los objetos de LibreOffice. Desafortunadamente, la curva de aprendizaje es muy pronunciada para estos objetos. Por lo general es mejor empezar con objetos simples e ir aumentando la complejidad de sus macros a medida que vaya aprendiendo. Intentar comprender las macros generadas es un principio para aprender.

Organización de las macros

En LibreOffice las macros se agrupan en *módulos*, los módulos se agrupan en *bibliotecas* y las bibliotecas se agrupan en *contenedores de bibliotecas*. Una biblioteca se usa generalmente como una agrupación mayor, ya sea para una categoría de macros o para una aplicación desarrollada con macros. Los módulos, generalmente, dividen la funcionalidad, agrupando procedimientos comunes como la interacción con el usuario o los cálculos. Las macros individuales son subrutinas o funciones. La figura 11 muestra un ejemplo de la estructura jerárquica de las bibliotecas de macros en LibreOffice.

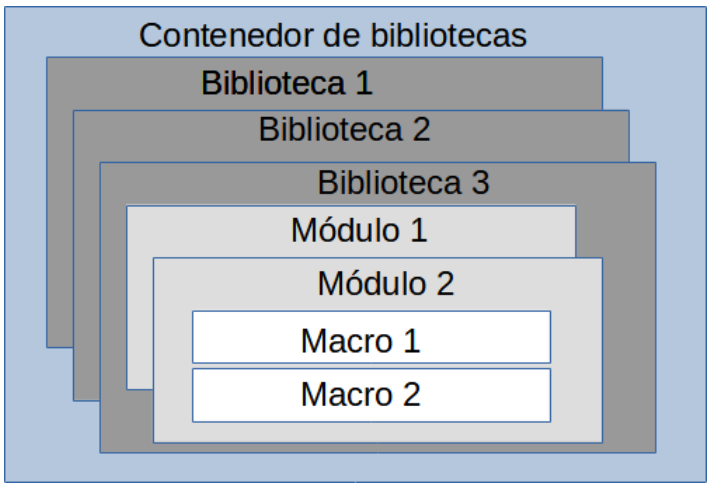


Figura 11: Jerarquía de las bibliotecas de macros

Vaya a **Herramientas > Macros > Organizar macros > LibreOffice Basic** en el menú para abrir el diálogo *Macros de Basic* (figura 1). En la lista *Macro* se muestran todos los contenedores de bibliotecas disponibles. Cada documento posee un contenedor de bibliotecas que, aunque puede estar vacío, es capaz de contener múltiples bibliotecas. La aplicación por sí misma tiene dos contenedores de bibliotecas, un contenedor para las macros distribuidas con LibreOffice llamado *Macros de LibreOffice* y el otro contenedor para las macros personales de los usuarios llamado *Mis macros*.

Las macros contenidas en *Macros de LibreOffice* se almacenan con los archivos propios de la aplicación y no podrá editarlas a menos que usted sea administrador. Esta medida ayuda a proteger las macros de LibreOffice, pues no deberían modificarse y no debería poder almacenar sus propias macros en el contenedor específico *Macros de LibreOffice*.

A no ser que sus macros se apliquen a un único documento y solo a ese documento, sus macros deberían almacenarse en el contenedor *Mis macros*. Este contenedor se almacena en su área de usuario o directorio personal.

Si una macro está incluida en un documento, la macro solo podrá ejecutarse desde ese documento porque usa principalmente `ThisComponent` para sus acciones.

Cada contenedor de bibliotecas de usuario contiene una biblioteca llamada *Standard*. Es mejor crear sus propias bibliotecas con nombres significativos que usar la biblioteca *Standard*. No solamente porque los nombres significativos son más fáciles de manejar, sino porque se pueden importar o exportar bibliotecas y esto no está permitido para la biblioteca *Standard*.

Precaución

LibreOffice permite importar bibliotecas de macros a su contenedor de bibliotecas *Mis macros*, pero no permite sobrescribir la biblioteca llamada *Standard*. Por lo que si almacena sus macros en la biblioteca *Standard* no la podrá importar desde otro contenedor de bibliotecas.

De manera predeterminada, al crear sus módulos LibreOffice los nombra como `Module1`, `Module2` y así sucesivamente. La misma recomendación de utilizar nombres significativos para sus bibliotecas debe servirle a la hora de nombrar sus módulos.

Cuando cree sus macros, debe decidir dónde almacenarlas. Almacenar una macro en un documento es útil si va a compartir el documento y solo se ejecutará en ese documento. En cambio, las macros almacenadas en el contenedor de bibliotecas de la aplicación llamado *Mis macros*, están disponibles de forma global para todos los documentos.

Para utilizar las macros contenidas en una biblioteca, primero debe cargar la biblioteca que las contiene. Las bibliotecas *Standard* y *Template* se cargan automáticamente.

Una biblioteca cargada se visualiza en el *Catálogo de objetos* del IDE de modo diferente a una que no está cargada. Para cargar una biblioteca y los módulos que contiene, haga doble clic en la biblioteca.

¿Dónde se almacenan las macros?

LibreOffice almacena los datos de configuración personalizados en un directorio dentro del directorio propio de cada usuario, específico para cada sistema operativo. En Windows suele ser `C:\Users\<Nombre_Usuario>\AppData\Roaming\LibreOffice\4\user\` y en Linux `/home/<Nombre_Usuario>/.config/libreoffice/4/user/`. Las macros se almacenan dentro de ese directorio de usuario, en un subdirectorio llamado `basic`, y cada biblioteca se almacena en su propio directorio dentro de este último.

Puede ver la ubicación de otros datos de su configuración mediante el menú **Herramientas > Opciones > LibreOffice > Rutas**.

Para un uso ocasional, no es preciso entender dónde se almacenan las macros, pero si conocer su ubicación, puede crear una copia de respaldo, compartirlas o inspeccionarlas si se le presenta un error.

Exportar macros

El *Organizador de macros de BASIC* le permite exportar bibliotecas de macros para utilizar en otro equipo o compartir con otros usuarios. Para exportar una biblioteca de macros:

1) Vaya a **Herramientas > Macros > Organizar macros > Basic** y haga clic en el botón *Organizador*.
2) Pulse en la pestaña *Bibliotecas* y elija la biblioteca que desee exportar.
3) Pulse el botón *Exportar* y seleccione *Exportar como biblioteca Basic* (observe que la biblioteca *Standard* no se puede exportar).
4) Elija la ubicación para guardar la biblioteca y pulse en *Guardar*.

Cuando se exporta una biblioteca, LibreOffice crea una carpeta que contiene todos los archivos relativos a la biblioteca. La figura 12 muestra el contenido de la biblioteca *BibliotecaPruebas* exportada.

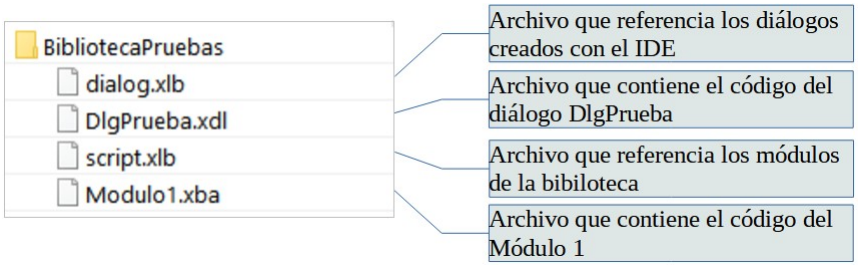


Figura 12: Contenido de la carpeta de la biblioteca exportada

Importar macros

El *Organizador de macros de BASIC* le permite importar bibliotecas, así como crear, borrar y renombrar bibliotecas, módulos y diálogos.

1) En la pestaña bibliotecas seleccione la *Ubicación* donde alojar la biblioteca (contenedor de bibliotecas) que puede ser un documento o el contenedor *Mis Macros y diálogos* y pulse el botón *Importar* para importar la biblioteca.

2) Navegue al directorio que contenga la biblioteca a importar (figura 13). Encontrará que puede elegir dos archivos: `dialog.xlb` y `script.xlb`. No importa cuál de estos archivos seleccione, cualquiera de ellos importará la biblioteca.

3) Las macros se pueden almacenar en bibliotecas dentro de los documentos de LibreOffice. Para importar las bibliotecas contenidas en documentos, seleccione el documento en lugar de un directorio en disco.

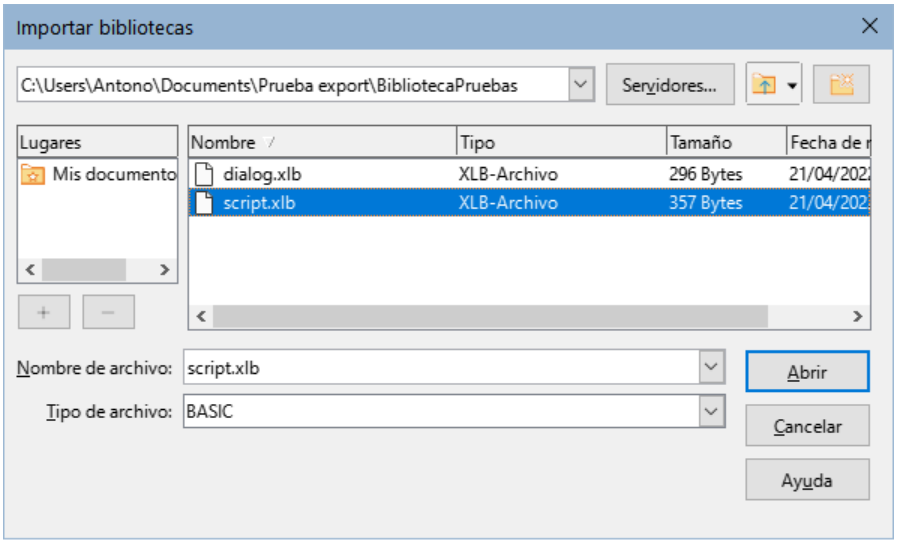


Figura 13: Diálogo Importar bibliotecas

4) Seleccione un archivo y pulse en *Abrir*, se abrirá el diálogo *Importar bibliotecas* (figura 14) con las opciones de importación.

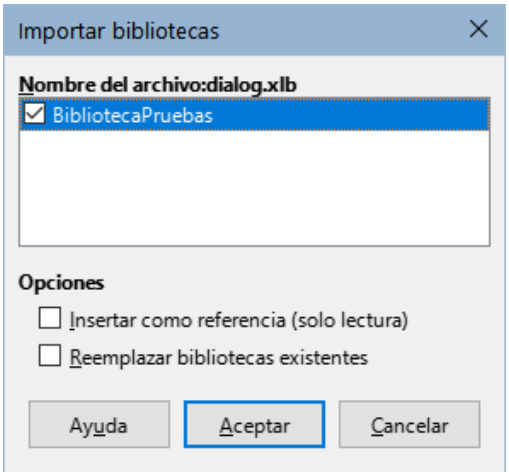


Figura 14: Opciones de importación

- Si no se indican opciones la biblioteca se copiará a su directorio personal de macros. Sin embargo, si la librería que quiere importar tiene el mismo nombre que una existente en su contenedor de bibliotecas, no se copiará.
- Si selecciona *Insertar como referencia (solo lectura)* la biblioteca, permanecerá en su propio directorio y aunque será totalmente operativa, no podrá ser editada o modificada por el IDE
- Si selecciona *Reemplazar bibliotecas existentes* se copiará a su directorio personal de macros reemplazando una biblioteca con el mismo nombre si existiera.

5) Pulse en *Aceptar* para importar la biblioteca de macros seleccionada.

Advertencia

No se puede exportar ni importar la biblioteca *Standard*.

Sugerencia

En Linux, los archivos de configuración de usuario de LibreOffice se almacenan en el directorio personal del usuario, dentro del subdirectrorio `.config`. Los archivos o directorios que empiezan por un punto están ocultos de manera predeterminada. Cuando utilice los diálogos de LibreOffice en lugar de los propios del sistema operativo, escriba el nombre del directorio en el cuadro correspondiente.

Descarga de macros para importación

Puede encontrar multitud de macros para su descarga en internet. Algunas macros están dentro de documentos, otras en archivos para su importación y otras se publican como texto en las páginas web. Si quiere utilizar estas últimas deberá copiar y pegar el texto en el editor de Basic. Consulte «Crear una macro» y «Ver y editar macros», para incluirlas en su sistema.

La tabla 1 contiene algunas direcciones de internet donde encontrará material relacionado con las macros para su descarga gratuita.

Tabla 1. Sitios recomendados con ejemplos de macros.

Enlace	***Contenido***
https://www.pitonyak.org/oo.php	Material de referencia sobre macros (inglés).
https://www.pitonyak.org/database/	Macros para bases de datos (inglés).
https://wiki.documentfoundation.org/Macros/es	Es la wiki de The Document Foundation para LibreOffice con bastantes enlaces a páginas relacionadas y páginas con macros.
https://ask.libreoffice.org/	Foro de LibreOffice seleccionando las etiquetas «Macros» encontrará respuestas a preguntas sobre macros.

Otras formas de ejecutar una macro

Como se indicó anteriormente, el uso de **Herramientas > Macros > Ejecutar macro** no es una manera eficaz para ejecutar macros que se utilizan con frecuencia. LibreOffice ofrece varios métodos para que pueda ejecutar macros de manera más inmediata.

Además de asignar un atajo de teclado explicado en «Ejecutar una macro» más atrás. También puede asignar una macro a un botón en una barra de herramientas, a un elemento de un menú, a un botón en un documento o a un suceso. La elección del método más adecuado depende de la utilidad de la macro, para ello, plantéese las siguientes cuestiones:

- ¿Debe estar disponible para un solo documento o para todos los documentos?
- ¿Es específica para cierto tipo de documentos, por ejemplo una hoja de cálculo?
- ¿Con qué frecuencia va a emplearla?

Las respuestas determinarán dónde colocar el acceso a la macro y su empleo. Por ejemplo: no es recomendable añadir una macro que usa esporádicamente a un botón de una barra de herramientas. La tabla 2 le ayudará a decidir que opción elegir.

Tabla 2. Posibles accesos a una macro

Situación del enlace a la macro	*Para todas las aplicaciones de LibreOffice*	*Para una aplicación específica*	*Para un único documento*
Barra de herramientas	No	Sí	Sí
Elemento de Menú	No	Sí	Sí
Atajo de teclado	Sí	Sí	No
Suceso	Sí	No	Sí

Barras de herramientas, elementos de menú y atajos de teclado

Para asignar una macro a un elemento de menú, un atajo de teclado, un icono de una barra de herramientas o un suceso, utilice **Herramientas > Personalizar**, que contiene páginas para este propósito. Use las pestañas disponibles para asignar el método adecuado. Para más información consulte el «Capítulo 14, Personalizar LibreOffice» en esta guía.

Suceso

Cuando realiza alguna acción en LibreOffice, se considera un suceso. Abrir un documento, modificarlo o cerrarlo, pulsar un botón o mover el ratón, son sucesos. LibreOffice permite asignar la ejecución de una macro a un suceso; En este ámbito, la macro se denomina como «Event handler» (manipulador de sucesos). La descripción del proceso para manejar los sucesos está fuera del alcance de este documento, pero algunas nociones pueden serle útiles.

Precaución

Tenga cuidado cuando configure un suceso para que ejecute una macro. Puede obtener resultados desastrosos. Por ejemplo, si utiliza una macro (manipulador de sucesos) que se ejecute cada vez que se modifica un documento, pero comete un error y las instrucciones no son adecuadas, puede que LibreOffice se bloquee y tenga que forzar su cierre.

1) Vaya a **Herramientas > Personalizar** en el menú para abrir el diálogo *Personalizar* y seleccione la pestaña **Sucesos** (figura 15). Los sucesos que aparecen en el diálogo están relacionados con la aplicación completa o con documentos específicos. Seleccione un suceso adecuado a su propósito.
2) En el desplegable *Guardar en*: Seleccione *LibreOffice* si quiere que la macro se ejecute globalmente para cualquier documento o en un documento abierto para que solamente se ejecute en ese documento.
3) Seleccione el suceso deseado y haga clic en *Macro* para abrir el diálogo *Selector de macros*.
4) Seleccione la macro que desee asignar y pulse en *Aceptar.* La página de sucesos mostrará la macro asignada a la derecha del suceso.

Sugerencia

Un uso frecuente de esta modalidad es asignar una macro al suceso *Abrir documento*, de esta manera, la macro efectúa unas tareas concretas con el documento, por ejemplo presentar un mensaje con instrucciones o información.

Muchos de los objetos de un documento se pueden configurar para ejecutar macros cuando se desencadena un suceso. El empleo más común es para agregar en un documento. Incluso al hacer doble clic sobre un gráfico abre un diálogo con una pestaña de macros que permite asignar una macro a un suceso.

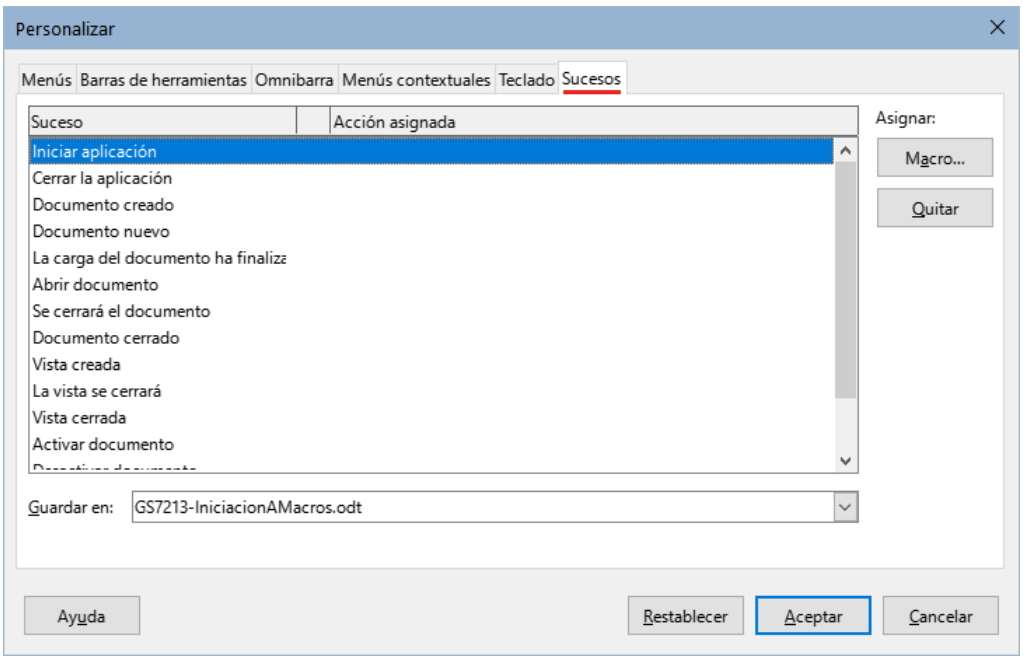


Figura 15: Pestaña Sucesos en el diálogo Personalizar

Extensiones

Una extensión es un paquete con macros integradas que se puede instalar en LibreOffice para añadir una nueva funcionalidad. Las extensiones se pueden escribir en casi cualquier lenguaje de programación y pueden ser simples o sofisticadas. Se pueden agrupar varios tipos:

- Complementos de Calc, los cuales proveen nuevas funcionalidades para Calc, incluyendo nuevas funciones que actúan como funciones integradas.
- Nuevos componentes y funcionalidad, que normalmente incluye cierto nivel de integración con la interfaz de usuario (IU), como nuevos menús o barras de herramientas.
- Complementos de «Chart» con nuevos tipos de gráficos.
- Componentes lingüísticos, como correctores ortográficos.
- Plantillas de documentos o imágenes para la galería.

Aunque las extensiones se pueden encontrar en muchos lugares, LibreOffice mantiene un repositorio específico en: http://extensions.libreoffice.org/ (en inglés) y también puede encontrar algunas en la página del Grupo de extensiones de Libre Planet (en inglés).

Consulte el «Capítulo 14, Personalizar LibreOffice» Para más información sobre extensiones.

Escribir macros (sin usar la grabadora)

Los ejemplos anteriores se han creado usando la grabadora de macros y el servicio de ayuda (*dispatcher*). Sin embargo, se pueden escribir macros que accedan directamente a los objetos que incluye LibreOffice si se siente cómodo escribiendo código de programación. En otras palabras, puede generar una macro para manipular directamente un documento usando una lógica de programación más avanzada.

Manipular directamente los objetos internos de LibreOffice es un tema avanzado que va más allá del alcance de este capítulo. Aquí solo se mostrarán unos ejemplos sencillos.

Un ejemplo de una macro para Writer

El código del listado 5 es un ejemplo sencillo creado sin la grabadora que inserta la cadena de texto «Hola» al final de un documento de Writer. Para agregar esta macro a una biblioteca siga los siguientes pasos:

1) Vaya a **Herramientas > Macros > Organizar macros > Basic**.
2) En el *Catálogo de objetos*, en *Mis macros y diálogos*, busque la biblioteca donde quiere escribir el código de la macro (para este ejemplo *BibliotecaPruebas)*.
3) Seleccione uno de los módulos disponibles, por ejemplo *Module1* o cree otro módulo usando el *Organizador de macros*.
4) Una vez seleccionado el módulo, pulse en *Editar*, lo que abrirá la ventana del IDE de Basic con las macros escritas en el módulo seleccionado.
5) Escriba o pegue el código del listado 5 en el módulo.

Listado 5: Macro para escribir «Hola» al final de un documento de Writer

```
Sub AgregarHola
  Dim oDoc
  Dim sTextService
  Dim oCurs

  oDoc = ThisComponent
Rem ThisComponent se refiere al documento activo.

Rem - Verificar que el documento activo es un documento de texto.
  sTextService = "com.sun.star.text.TextDocument"
Rem - Si no es un documento de texto presenta un aviso y sale de la macro
  If NOT oDoc.supportsService(sTextService) Then
    MsgBox "Esta macro solo funciona en documentos de texto"
    Exit Sub
  End If
Rem Obtener el control del cursor visible.
  oCurs = oDoc.currentController.getViewCursor()

Rem Mover el cursor al final del documento.
  oCurs.gotoEnd(False)

Rem Insertar la cadena de texto en la posición del cursor.
  oCurs.Text.insertString(oCurs,"Hola",False)
End Sub
```

Un ejemplo de una macro para Calc

Una forma de ampliar las funcionalidades de LibreOffice Calc es escribir macros para automatizar tareas. Puede usar el lenguaje Basic para escribir macros que realicen tareas que van desde el manejo y formato de simples celdas hasta la manipulación avanzada de datos.

Como un ejemplo simple, considere que desea analizar un rango de celdas para determinar si todos los valores están entre 0 y 100. Los valores que van desde 50 a 100 quiere resaltarlos en verde claro, mientras que los valores mayores o iguales a 0 y menores que 50 en rojo claro. Si se encuentran valores fuera del rango permitido (de 0 a 100), se debe mostrar un mensaje de advertencia y resaltar las celdas en gris. El listado 6 muestra el código para esta macro.

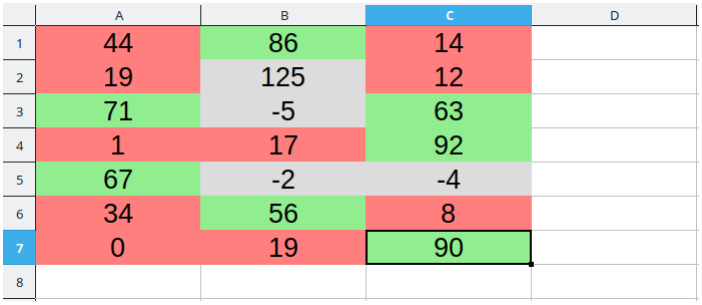
Listado 6: Macro para Calc que formatea celdas basándose en sus valores

```
Sub FormatoBasadoenValores '(el apóstrofo indica comentario)
   ' Declaración de variables
   Dim oRange as Object, oCell as Object
   ' Obtener el intervalo seleccionado
   Set oRange = Thiscomponent.getCurrentSelection()
   'Comprobar si el intervalo es un intervalo o rango sencillo
   If Not oRange.supportsService("com.sun.star.sheet.SheetCellRange") Then
      MsgBox "Esta macro solo funciona en rangos sencillos (celdas contiguas)"
      Exit Sub
   End If
   ' Obtener el número de columnas y filas en la selección
   Dim nCols as Long : nCols = oRange.Columns.getCount()
   Dim nRows as Long : nRows = oRange.Rows.getCount()
   Dim col as Long, row as Long
   Dim cellValue as Long
   Dim isError as Boolean : isError = False
   ' Colorear las celdas en función de sus valores, esto se ejecuta
   ' para todas las celdas del intervalo mediante un bucle anidado
   For col = 0 To nCols - 1 ' inicio del bucle para las columnas
      For row = 0 To nRows - 1 ' inicio del bucle para las filas
         Set oCell = oRange.getCellByPosition(col, row)
         cellValue = oCell.getValue()
         If cellValue >= 50 and cellValue <= 100 Then
            ' Establecer el color de fondo en verde claro
            oCell.CellBackcolor = RGB(144, 238, 144)
         ElseIf cellValue >= 0 and cellValue < 50 Then
            ' Establecer el color de fondo en rojo claro
            oCell.CellBackcolor = RGB(255, 127, 127)
         Else
            ' Establecer el color de fondo en gris claro
            oCell.CellBackcolor = RGB(220, 220, 220)
            isError = True
         End If
      Next row ' va a la siguiente fila
   Next col ' va a la siguiente columna
   ' Muestra un mensaje si hay errores (valor fuera del rango permitido)
   If isError Then
      MsgBox "Las celdas con valores fuera del rango de 0 a 100 se han
resaltado con gris claro"
   End If
End Sub
```

Para agregar esta macro a una biblioteca siga los pasos descritos en «Un ejemplo de una macro para Writer».

Para ejecutarla, primero cree una hoja de Calc y agregue algunos valores numéricos en un intervalo. Seleccione el intervalo y utilice uno de los métodos descritos en «Otras formas de ejecutar una macro» para ejecutarla.

La figura 16 muestra el resultado de haber ejecutado la macro en un intervalo. Cuando algunos valores están fuera del rango permitido (de 0 a 100), se muestra el mensaje de la figura 17.



	A	B	C	D
1	44	86	14	
2	19	125	12	
3	71	-5	63	
4	1	17	92	
5	67	-2	-4	
6	34	56	8	
7	0	19	90	
8				

Figura 16: Celdas formateadas por la macro

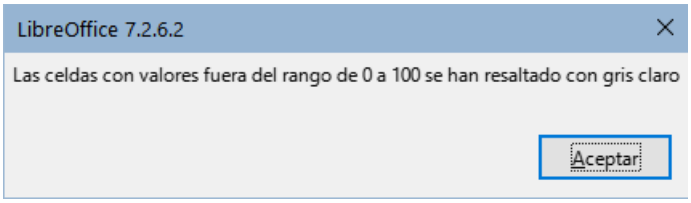


Figura 17: Mensaje al encontrar el error definido

La biblioteca *ScriptForge*

Los programadores de macros necesitan realizar ciertas tareas con frecuencia, como crear y abrir archivos, acceder a controles de formulario, leer datos de bases de datos incrustadas en documentos base, etc. El objetivo de la biblioteca *ScriptForge* es facilitar la ejecución de dichos comandos sin tener que aprender la API (interfaz de programación de aplicaciones) ni algunos comandos que pueden resultar complicados para programadores ocasionales.

La biblioteca de *ScriptForge* está organizada en un conjunto de servicios, cada uno proporciona métodos y propiedades relacionadas con un tema específico. Por ejemplo, el servicio *Dialog* proporciona acceso a los diálogos disponibles en los módulos de *script* y el servicio *Database* permite ejecutar comandos SQL en documentos de Base.

El ejemplo del listado 7 muestra una macro escrita en Basic usando la biblioteca *ScriptForge* que abre un documento Calc (miarchivo.ods), genera una nueva hoja (HojaNueva) e inserta la cadena «Hola» en la celda A1. La macro también guarda y cierra el documento.

Listado 7: Macro para Calc usando la biblioteca ScriptForge

```
Sub CreateSheetExample
   ' Carga la biblioteca ScriptForge
   GlobalScope.BasicLibraries.LoadLibrary("ScriptForge")
   ' Inicia el servicio UI
   Dim ui as Object, myDoc as Object
   ui = CreateScriptService("UI")
   ' En Linux, abre el archivo "miarchivo.ods" **vea la nota siguiente
   Set myDoc = ui.OpenDocument("/home/USER/Documents/miarchivo.ods")
   ' Inserta una hoja nueva con nombre «HojaNueva»
   myDoc.InsertSheet("HojaNueva")
   ' Inserta la cadena «Hola» en la celda A1 de la hoja «HojaNueva»
   myDoc.SetValue("HojaNueva.A1", "Hola")
   ' Muestra la hoja «HojaNueva»
   myDoc.Activate("HojaNueva")
   ' Guarda el documento
   myDoc.Save()
   ' Cierra el documento
   myDoc.CloseDocument()
```

```
End Sub
```

Como se ve en el ejemplo, con la biblioteca *ScriptForge* se escriben instrucciones muy sencillas para ejecutar comandos, lo que simplifica mucho la creación de macros.

Nota

Para que la macro funcione deberá crear un documento de Calc con el nombre `miarchivo.ods` en su directorio predeterminado de documentos. Para entornos Windows debe modificar la octava línea para que sea:

```
Set myDoc = ("file://C:/Users/USER/Documents/miarchivo.ods")
```

Tanto para Linux como para Windows debe cambiar la palabra `USER` para que coincida con su nombre de usuario

Información

Para más información sobre la biblioteca *ScriptForge*, visite la Ayuda en línea: https://help.libreoffice.org/latest/es/text/sbasic/shared/03/lib_ScriptForge.html. Cada uno de los servicios admitidos ha sido ampliamente documentado y se proporcionan ejemplos para los lenguajes de programación Basic y Python.

La biblioteca *ScriptForge* está disponible desde LibreOffice 7.1 con soporte para Basic. En LibreOffice 7.2, se agregó compatibilidad con Python. Actualmente, la mayoría de los servicios, métodos y propiedades se pueden utilizar indistintamente en Basic o Python.

Inspector de objetos UNO

LibreOffice tiene una API extensa que los programadores de macros pueden usar para automatizar casi cualquier aspecto de sus aplicaciones. Sin embargo, uno de los principales desafíos para los programadores es descubrir los tipos de objetos, así como sus servicios, métodos y propiedades compatibles.

A partir de LibreOffice 7.2, se puede utilizar el *Inspector de objetos UNO* como ayuda para los desarrolladores de macros a inspeccionar objetos, descubrir cómo acceder a ellos y utilizarlos en las macros. Esta función está disponible en Writer, Calc, Impress y Draw. Para habilitarlo, vaya a **Herramientas > Herramientas de desarrollo**. El *Inspector de objetos* se abrirá en la parte inferior de la interfaz de usuario, como se muestra en la figura 18.

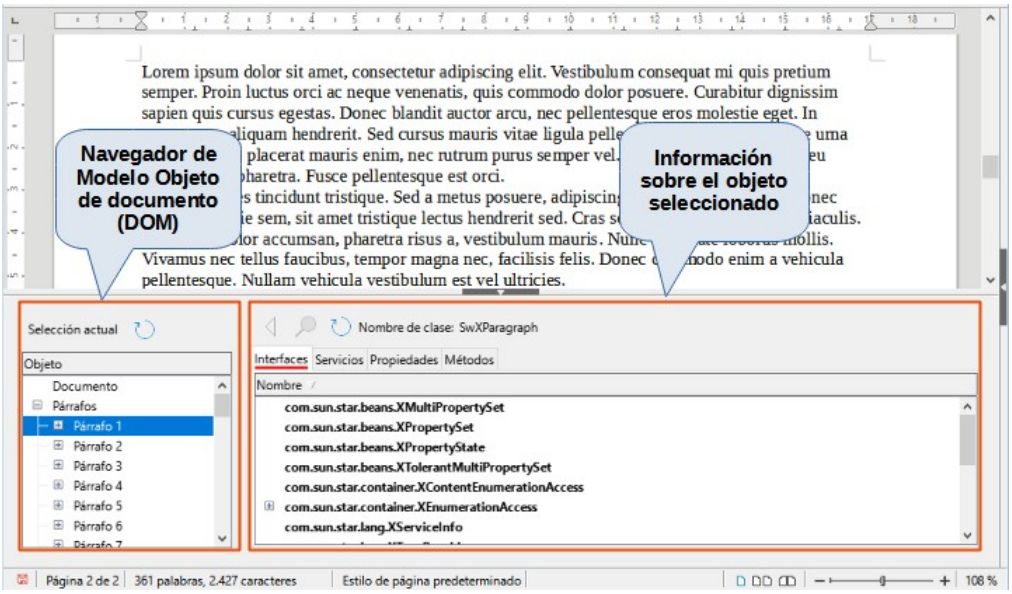


Figura 18: El Inspector de objetos UNO en un documento de Writer

La parte izquierda del *Inspector de objetos* consiste en el navegador de *Modelo de objetos del documento (DOM)*, que permite al usuario navegar por todos los objetos del documento. Cuando se selecciona un objeto, en la parte derecha de la ventana del *Inspector de objetos* se muestra la información sobre ese objeto en función de la pestaña activada:

- Los nombres de todas las *Interfaces* implementadas.
- Los nombres de todos los *Servicios* admitidos por el objeto.
- Los nombres y tipos de todas las *Propiedades* disponibles en el objeto.
- Los nombres, argumentos y tipos de retorno de todos los *Métodos* que pueden llamar el objeto.

En lugar de inspeccionar todos los objetos utilizando el navegador DOM, es posible inspeccionar directamente un objeto seleccionado en el documento al alternar el botón *Selección actual*.

Por ejemplo, suponga que desea cambiar el color de fondo del texto seleccionado en un documento de Writer. Seleccione una parte del texto. Abra el *Inspector de objetos* y pulse en *Selección actual*. Inspeccione las propiedades del objeto en busca de una propiedad que coincida con el efecto deseado. La figura 19 muestra la propiedad `CharBackColor` seleccionada, que es la propiedad utilizada para establecer el color de fondo del texto.

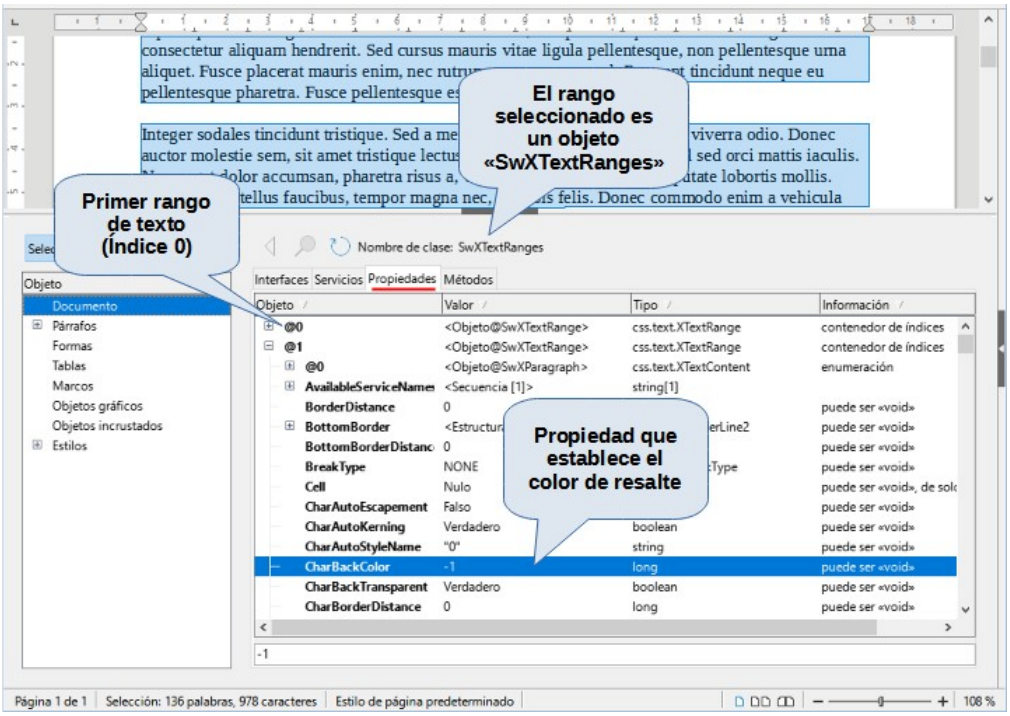


Figura 19: Uso del Inspector de objetos para buscar una propiedad

Ahora puede escribir una macro usando esta propiedad para cambiar el color de fondo del texto seleccionado. El listado 8 muestra el código para esta macro.

Listado 8: Macro que cambia el color de fondo de un fragmento de texto

```
Sub CambiarColorFondo
  Dim oSel as Object
  Set oSel = ThisComponent.getCurrentSelection()
  oSel(0).CharBackColor = RGB(255, 127, 127)
End Sub
```

Tenga en cuenta que es posible tener varios fragmentos o rangos de texto seleccionados a la vez, por lo que, en la macro, `oSel(0)` accede al primer fragmento.

Visión general de macros en Python, BeanShell y JavaScript

Es posible que muchos programadores no estén familiarizados con *LibreOffice Basic*, por lo que LibreOffice admite macros escritas en otros tres lenguajes que pueden ser más familiares: Python, BeanShell y JavaScript.

Las macros se organizan de la misma manera para los cuatro lenguajes de secuencias de comandos. El contenedor de bibliotecas *Macros y diálogos de LibreOffice* contiene todas las macros que se proporcionan en la instalación de LibreOffice. El contenedor de las bibliotecas *Mis macros* contiene las macros que están disponibles para cualquiera de sus documentos de LibreOffice. Cada documento también puede contener sus macros (solamente disponibles para ese mismo documento).

La *Grabadora de macros* solo puede crear macros para *LibreOffice Basic*. Para usar otros lenguajes de secuencias de comandos disponibles, debe escribir el código usted mismo y luego cópielo a la carpeta `Scripts` en su directorio de usuario. Para más información, consulte «¿Dónde se almacenan las macros?» más atrás.

Cuando selecciona ejecutar una macro usando **Herramientas > Macros > Ejecutar macro**, LibreOffice muestra el *Selector de macros*. Este diálogo permite seleccionar y ejecutar cualquier macro disponible, codificado en cualquiera de los lenguajes admitidos (figura 20).

Cuando edita una macro utilizando **Herramientas > Macros > Editar macros**, LibreOffice muestra el Entorno de desarrollo integrado de Basic que permite seleccionar y editar cualquier macro de *LibreOffice Basic* disponible, pero este editor no se puede usar en macros de otros lenguajes.

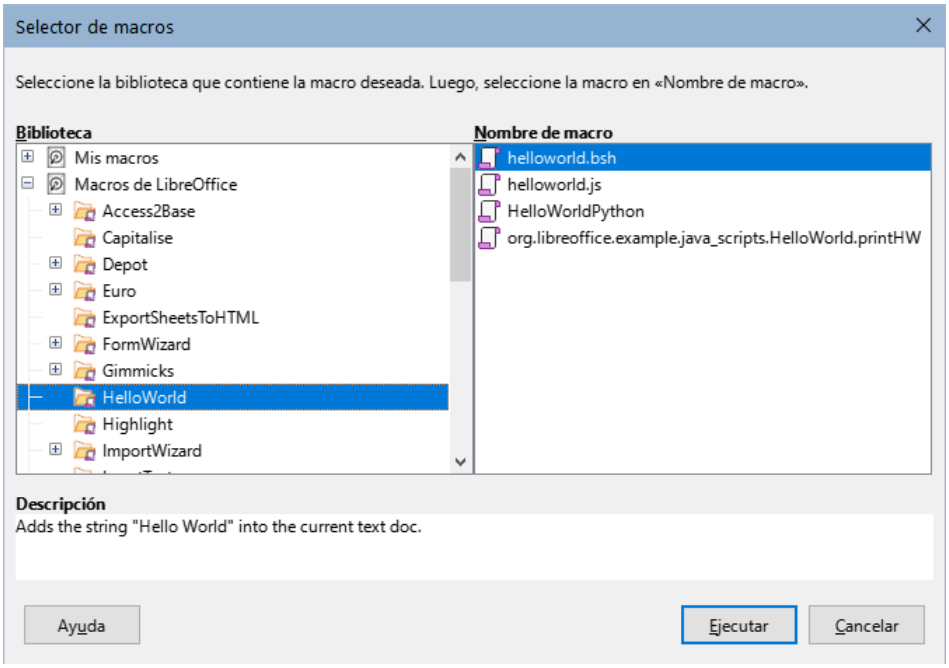


Figura 20: Diálogo Selector de macros

El modelo de componentes utilizado en LibreOffice se conoce como Universal Network Objects o `UNO`. Las macros de LibreOffice en cualquier lenguaje de secuencias de comandos utiliza la interfaz de programación de aplicaciones (`API`) en tiempo de ejecución de `UNO`.

La interfaz `XSCRIPTCONTEXT` está disponible para cualquiera de los cuatro lenguajes y proporciona algunos métodos de acceso a las diversas interfaces que pueden necesitar para realizar una acción en un documento.

Consejo

Si desea más información sobre la API de LibreOffice y los objetos UNO, consulte la documentación oficial de la API en https://api.libreoffice.org/

Macros en Python

Python es un lenguaje de programación de propósito general y alto nivel que se publicó por primera vez en 1991.

Cuando selecciona **Herramientas > Macros > Organizar macros > Python**, LibreOffice muestra el diálogo *Macros en Python* (figura 21).

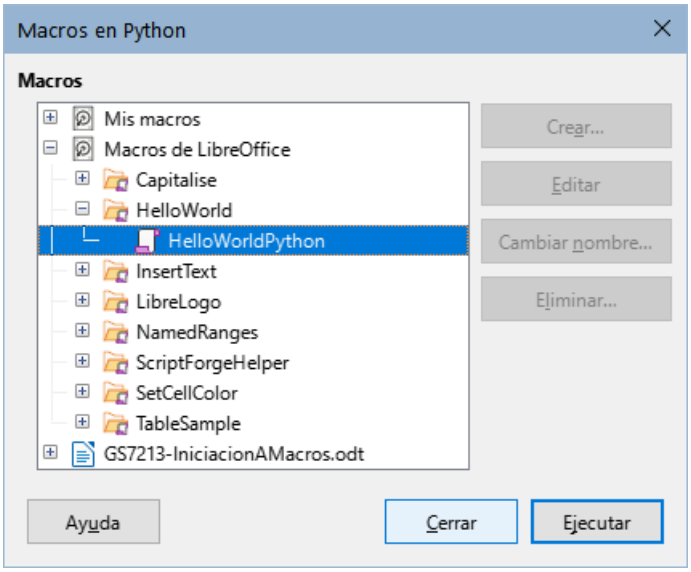


Figura 21: Diálogo Macros en Python

Las funciones para editar y depurar secuencias de comandos de Python no están integradas actualmente en la interfaz de usuario estándar de LibreOffice. Sin embargo, puede usar cualquier editor de Python para crear scripts y luego copiar estos archivos en el subdirectorio `Scripts` dentro de su directorio de usuario. Para más información, consulte «¿Dónde se almacenan las macros?» más atrás.

El listado 9 presenta un ejemplo de macro en Python que escribe el texto «Hello World (in Python)» al final de un documento de Writer.

Listado 9: Ejemplo de macro en Python

```
def HelloWorldPython():
    # Obtiene el documento mediante el contexto.
    desktop = XSCRIPTCONTEXT.getDesktop()
    model = desktop.getCurrentComponent()

    # Si no hay abierto un documento crea una nuevo
    if not hasattr(model, "Text"):
        model = desktop.loadComponentFromURL(
            "private:factory/swriter", "_blank", 0, ())
    # obtiene la interfaz de XText
    text = model.Text
    # creat un rango de texto(XtextRange)al final del documento
    tRange = text.End
    # y escribe la cadena de texto
    tRange.String = "Hello World (in Python)"
    return None
```

Sugerencia

La extensión «Alternative Python Script Organizer» (APSO) facilita la edición y organización de scripts de Python, en particular cuando están incrustados en un documento. Con APSO, puede configurar su editor de código preferido, iniciar el shell de Python integrado y depurar scripts de Python.
Visite https://gitlab.com/jmzambon/apso para descargar APSO y obtener más información sobre cómo usarlo.

Información

Para más información sobre las secuencias de comandos de Python en LibreOffice, puede consultar la Wiki https://wiki.documentfoundation.org/Macros/Python_Basics, donde encontrará algunas explicaciones detalladas y ejemplos sobre cómo empezar con las secuencias de comandos de Python. (en inglés por el momento)

Macros en BeanShell

BeanShell es un intérprete de código fuente de Java que se lanzó por primera vez en 1999.

Cuando selecciona **Herramientas > Macros > Organizar macros > BeanShell**, LibreOffice muestra el diálogo *Macros en BeanShell* (figura 22).

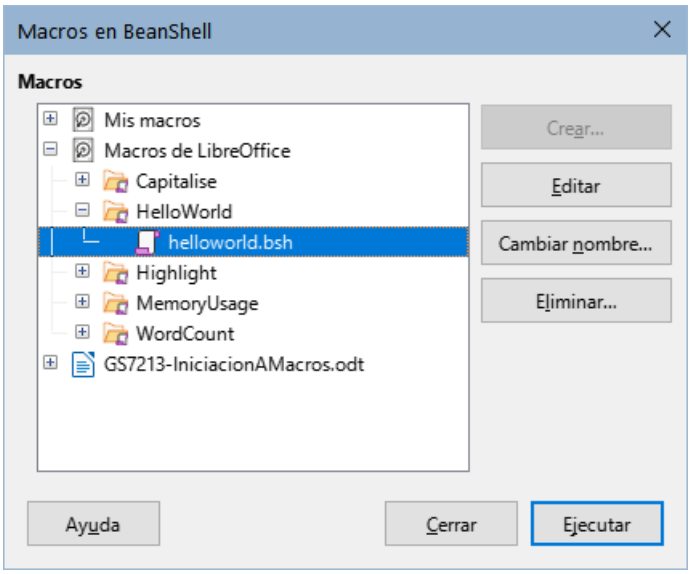


Figura 22: Diálogo Macros BeanShell

Pulse el botón *Editar* para acceder al editor y depurador de BeanShell (figura 23).

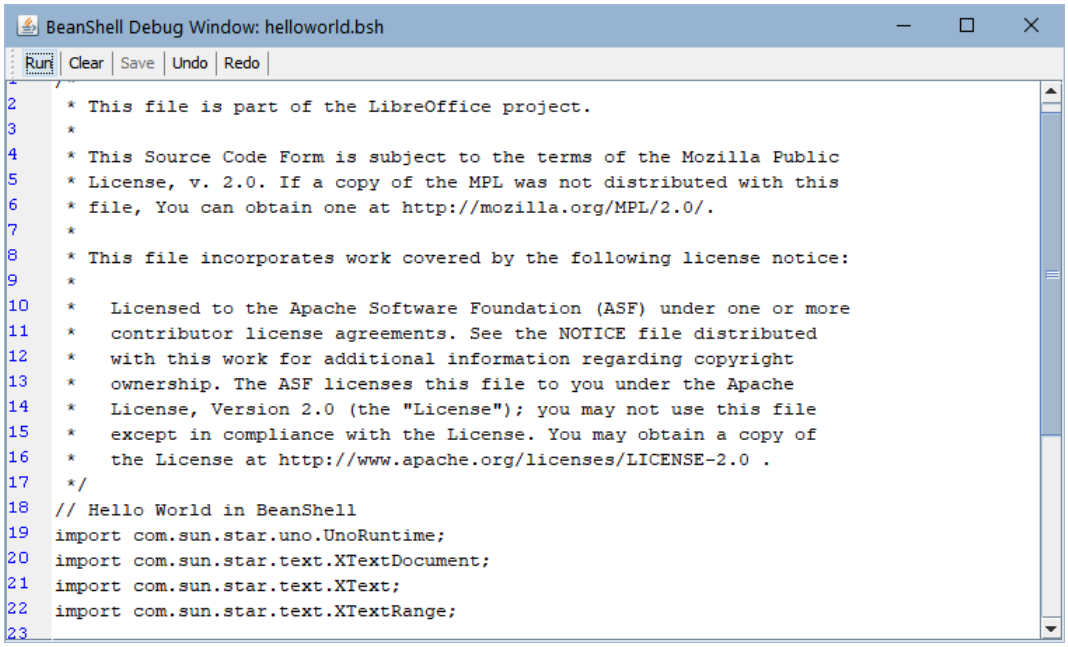


Figura 23: Ventana de depuración BeanShell

El listado 10 es un ejemplo de una macro BeanShell que inserta el texto «Hello World (in BeanShell)» al final de un documento de Writer.

Listado 10: Ejemplo de macro en BeanShell

```
import com.sun.star.uno.UnoRuntime;
import com.sun.star.text.XTextDocument;
import com.sun.star.text.XText;
import com.sun.star.text.XTextRange;

// Obtiene el documento mediante el contexto
oDoc = XSCRIPTCONTEXT.getDocument();
// Obtiene la interfaz XTextDocument
xTextDoc = (XTextDocument)
UnoRuntime.queryInterface(XTextDocument.class,oDoc);
//obtiene la interfaz XText
xText = xTextDoc.getText();
// Obtiene el rango de texto vacío (XtextRange) al final del documento
xTextRange = xText.getEnd();
// Inserta la cadena de texto
xTextRange.setString( "Hello World (in BeanShell)" );
```

Macros en JavaScript

JavaScript es un lenguaje de secuencias de comandos de alto nivel que se lanzó por primera vez en 1995.

Cuando selecciona **Herramientas > Macros > Organizar macros > JavaScript**, LibreOffice muestra diálogo *Macros en JavaScript* (figura 24).

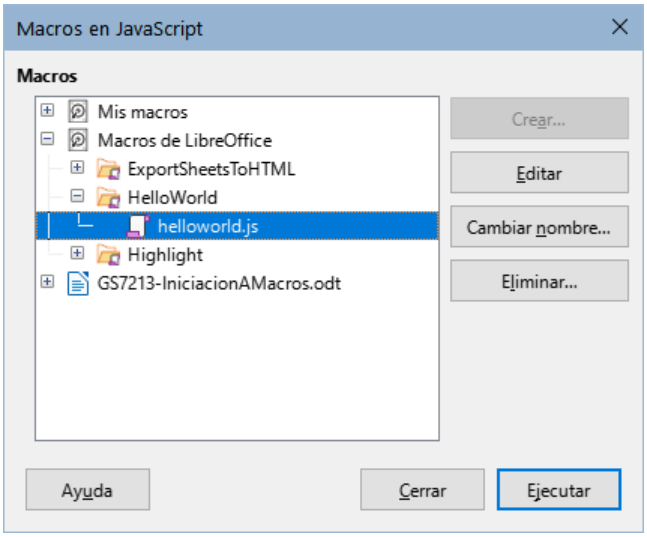


Figura 24: Diálogo macros en JavaScript

Pulse el botón *Editar* para acceder al editor y depurador de JavaScript «Rhino» (figura 25). Las instrucciones para usar esta herramienta se encuentran en las páginas web archivadas de Mozilla https://www-archive.mozilla.org/rhino/debugger.

El listado 11 es un ejemplo de una macro de JavaScript que inserta el texto «Hello World (in JavaScript)» al final de un documento de Writer.

Listado 11: Ejemplo de macro en JavaScript

```
importClass(Packages.com.sun.star.uno.UnoRuntime);
importClass(Packages.com.sun.star.text.XTextDocument);
```

```
importClass(Packages.com.sun.star.text.XText);
importClass(Packages.com.sun.star.text.XTextRange);

//Obtiene el documento mediante el contexto
oDoc = XSCRIPTCONTEXT.getDocument();
//Obtiene la interfaz XTextDocument
xTextDoc = UnoRuntime.queryInterface(XTextDocument,oDoc);
//Obtiene la interfaz XText
xText = xTextDoc.getText();
//Obtiene un rango de texto vacio (XtextRange) al final del documento
xTextRange = xText.getEnd();
//Escribe la cadena de texto en el rango
xTextRange.setString( "Hello World (in JavaScript)" );
```

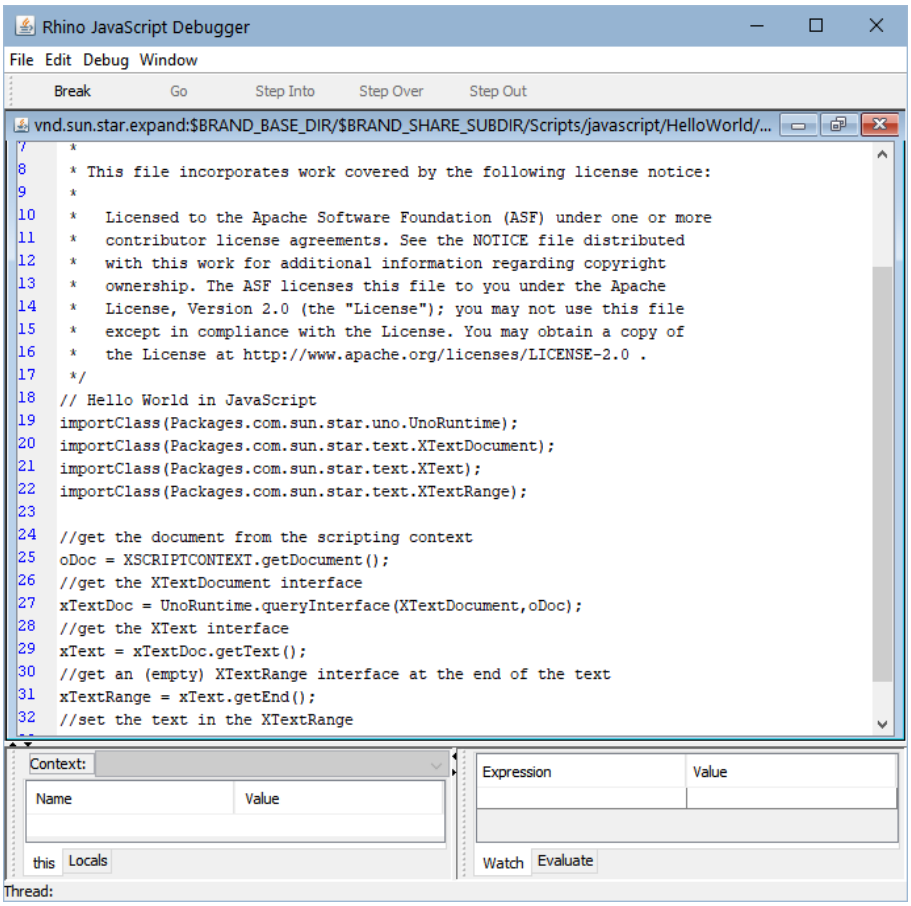


Figura 25: Depurador Rhino JavaScript

Encontrar información

Existen numerosos recursos disponibles le ayudarán en la escritura de macros, uno de ellos es la ayuda de LibreOffice. Vaya a **Ayuda > Ayuda de LibreOffice** para abrir el sistema de ayuda de LibreOffice en su navegador. En la esquina superior izquierda de la página de inicio encontrará una lista desplegable que le permite seleccionar el contenido para un determinado programa de LibreOffice. Para la ayuda de Basic, seleccione *LibreOffice Basic* en esta lista.

Material incluido

LibreOffice incluye la biblioteca *Tools* en la que se incluyen bastantes macros y funciones que puede utlizar como material auxiliar Vaya a **Herramientas > Macros > Organizar macros >**

LibreOffice Basic para abrir el diálogo de macros. En el contenedor de bibliotecas *Macros de LibreOffice*, expanda la biblioteca *Tools* e Inspeccione su contenido. Por ejemplo: en el módulo *Debug*, la macro `WritedbgInfo` proporciona información sobre métodos, propiedades e interfaces de un objeto (crea un documento de texto) y `printdbgInfo` la muestra en pantalla;

Uso: `WritedbgInfo(<Nombre_de_variable_Tipo_OBJETO>).`

En la página oficial de LibreOffice, también puede descargar el Kit de desarrollo de software (SDK) como un componente de apoyo que instala las páginas de la API en su ordenador.

Recursos en línea

LibreOffice surgió como una bifurcación del código liberado de OpenOffice y este, a su vez, del extinto StarOffice. Si bien las aplicaciones de LibreOffice se encuentran en una fase de desarrollo mucho más avanzada y poseen unas características más potentes, el lenguaje de macros Basic es prácticamente igual para estas suites.

Es por esto que la información y ejemplos de macros creadas para las otras aplicaciones son válidas para LibreOffice Basic aunque a veces tendrá que hacer pequeñas adaptaciones debido a las distintas evoluciones de sus interfaces de programación de aplicaciones (API).

Desafortunadamente, existe poco material elaborado o traducido al español, por lo que gran parte de la información que obtenga será en inglés.

Páginas web

Un buen punto de partida es la página de macros de nuestra wiki, encontrará multitud de enlaces y ejemplos de macros: https://wiki.documentfoundation.org/Macros/es.

La wiki de Apache OpenOffice también tienen unas páginas dedicadas a las macros https://wiki.openoffice.org/wiki/ES/Manuales/GuiaAOO/TemasAvanzados/Macros

En la web en español https://wiki.open-office.es encontrará una sección dedicada a Basic con ejemplos de macros y diálogos (https://wiki.open-office.es/Basic).

En la página del SDK de LibreOffice encontrará también varios ejemplos de macros para los distintos lenguajes de secuencias de comandos. https://api.libreoffice.org/

Foros:

Los foros son también de gran ayuda puedes consultar tus dudas y seguro que obtendrás respuesta. Aunque la gran parte de las colaboraciones se hacen en inglés también tienen subforos para distintos idiomas y aplicaciones.

https://ask.libreoffice.org/ (foro de la comunidad de LibreOffice).

http://forum.openoffice.org/forum/ (foro de la comunidad de Apache OpenOffice).

Libros en formato electrónico e impresos

El libro de Andrew Pitonyak *OpenOffice.org Macros Explained* está considerado como una referencia indispensable en la programación de macros Basic es de libre distribución y puede descargarse tanto en formato pdf como odt de su web https://www.pitonyak.org/, donde también encontrará información muy valiosa sobre macros.

El libro de Mauricio Baeza *Aprendiendo Ooobasic* es también una muy buena referencia para empezar a escribir macros y su contenido está en español. Puede descargar su versión en pdf desde: https://git.cuates.net/elmau/libreoffice-books/raw/branch/master/books/es/AprendiendoOOoBasic.pdf.

Actualmente, se encuentran muy pocas publicaciones en papel debido a la evolución de las comunicaciones por internet y al coste elevado de su lanzamiento. Dos libros considerados como referencia del lenguaje Basic son: *Learn OpenOffice.org Spreadsheet Macro Programming*

del Dr. Mark Alexander Bain y *Database Programming with OpenOffice.org Base & Basic* de Roberto Benítez (ambos en inglés) pero han sido descatalogados en sus respectivas editoriales, con lo que es difícil conseguirlos a no ser de segunda mano. Busque en internet si está interesado.