LibreOffice Calc マクロ Basic プログラミング

目次

第１章　LibreOffice Calc マクロBasicとは ・・・・・・・・・・・・ 3
１．基礎知識 ・・・・・・・・・・・・ 3
２．マクロの構成 ・・・・・・・・・・・・ 4
2-1. マクロを動かすためのプログラムを組む ・・・・・・・・・・・・ 4
2-2　マクロの実行 ・・・・・・・・・・・・ 4
３．プログラムを書き込む手順 ・・・・・・・・・・・・ 4
3-1. LibreOffice マクロエディタ―を起動 ・・・・・・・・・・・・ 4
3-2. Basicマクロのダイアログ ・・・・・・・・・・・・ 4
3-3. Basic Editorにマクロのプログラムコード ・・・・・・・・・・・・ 5
3-4. マクロの実行 ・・・・・・・・・・・・ 6
４．Basicプログラムを組む ・・・・・・・・・・・・ 6
4-1. シートの上をダブルクリックしてプログラムを動かす。・・・・・・・・ 6
4-2. ダイアログを使ってプログラミング
ダイアログを使ってマクロを動かす。 ・・・・・・・・・・・・ 8

第２章　プログラミング（モジュール編） ・・・・・・・・・・・・ 10
１．セルの操作 ・・・・・・・・・・・・ 10
１－１．セルをアクティブにする ・・・・・・・・・・・・ 10
１－２．セルにデーターを入力する ・・・・・・・・・・・・ 11
１－３．セルをコピー、切り取り、貼り付け ・・・・・・・・・・・・ 12
１－４．書式のコピー ・・・・・・・・・・・・ 13
１－５．セル範囲を削除する ・・・・・・・・・・・・ 13
１－６．セル範囲をクリアー ・・・・・・・・・・・・ 14
１－７．セルを挿入 ・・・・・・・・・・・・ 14
１－８．セルの幅を設定 ・・・・・・・・・・・・ 15
１－９．セルの高さを設定 ・・・・・・・・・・・・ 16
１-10. フォントを設定 ・・・・・・・・・・・・ 16
１-11.　セルの背景色を設定 ・・・・・・・・・・・・ 17
１-12. 数値や日付の表示形式の設定 ・・・・・・・・・・・・ 17
１-13. セル内の配置を指定 ・・・・・・・・・・・・ 19
１）文字の配置 ・・・・・・・・・・・・ 19
２）セル内文字列の制御 ・・・・・・・・・・・・ 19
２．シートの操作 ・・・・・・・・・・・・ 21
２-1. シートを追加する ・・・・・・・・・・・・ 21
２－２．シートをアクティブにする ・・・・・・・・・・・・ 21
２－３．シートの名前を変更する ・・・・・・・・・・・・ 21
２－４．シートを移動する ・・・・・・・・・・・・ 21

２−５．シートをコピーする ・・・・・・・・・・・ 22
２−６．シートを削除する ・・・・・・・・・・・ 22
２−７．シートを保護する ・・・・・・・・・・・ 22
３．ブック（ドキュメント）の操作 ・・・・・・・・・・・ 23
３−１．ブック（ドキュメント）を作成する ・・・・・・・・・・・ 23
３−２．ブック（ドキュメント）を開く ・・・・・・・・・・・ 24
３−３．組み込みダイアログボックスを使ってドキュメントを開く ・・・ 24
３−４．複数のドキュメントを開く ・・・・・・・・・・・ 24
３−５．開いたドキュメントを操作 ・・・・・・・・・・・ 25
４．ファイルの操作 ・・・・・・・・・・・ 25
４−１．ファイル、フォルダーを探す ・・・・・・・・・・・ 25
４−２．フォルダーの作成・削除 ・・・・・・・・・・・ 27
４−３．ファイルのコピー・移動 ・・・・・・・・・・・ 27
４−４．ファイルの削除 ・・・・・・・・・・・ 27
４−５．ファイルの書き込み（シーケンシャルファイル）・・・・・・・・・・ 28
４−６．ファイルの読み込み（シーケンシャルファイル）・・・・・・・・・・ 28
４−７．CSV ファイルの書き込み ・・・・・・・・・・・ 29
４−８．CSV ファイルを開く ・・・・・・・・・・・ 29
４−９．ファイルの書き込み（ランダムアクセスファイル）・・・・・・・・・ 31
４-10．ファイルの読み込み（ランダムアクセスファイル）・・・・・・・・・ 31
４-11．テキストファイルを開く ・・・・・・・・・・・ 32
５．印刷 ・・・・・・・・・・・ 32
６．罫線 ・・・・・・・・・・・ 33
７．　データー ・・・・・・・・・・・ 35
７−１．レコードの数を数える ・・・・・・・・・・・ 35
７−２．データの並べ替え ・・・・・・・・・・・ 35

第３章　プログラミング（Sheet 編） ・・・・・・・・・・・ 36

第４章　プログラミング（ユーザーフォーム編） ・・・・・・・・・・・ 39

参照　罫線の種類 ・・・・・・・・・・・ 44

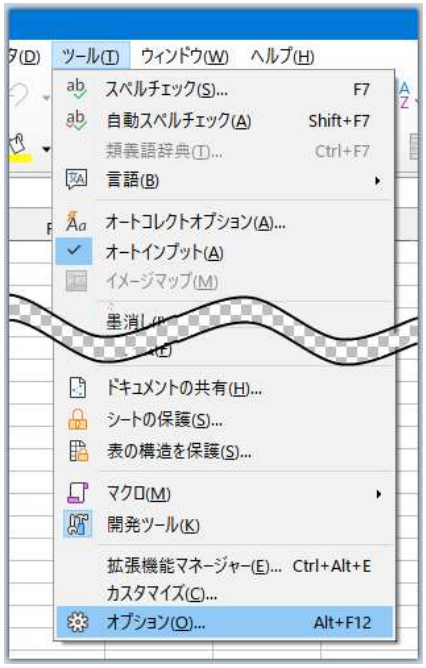
第１章 LibreOffice Calc マクロ Basic とは

マクロはコンピューターの作業を自動化するための機能を言います。
その機能をプログラミングする手段として Basic を使います。
LibreOffice には他に BeanShell、JavaScript、Python が用意されています。

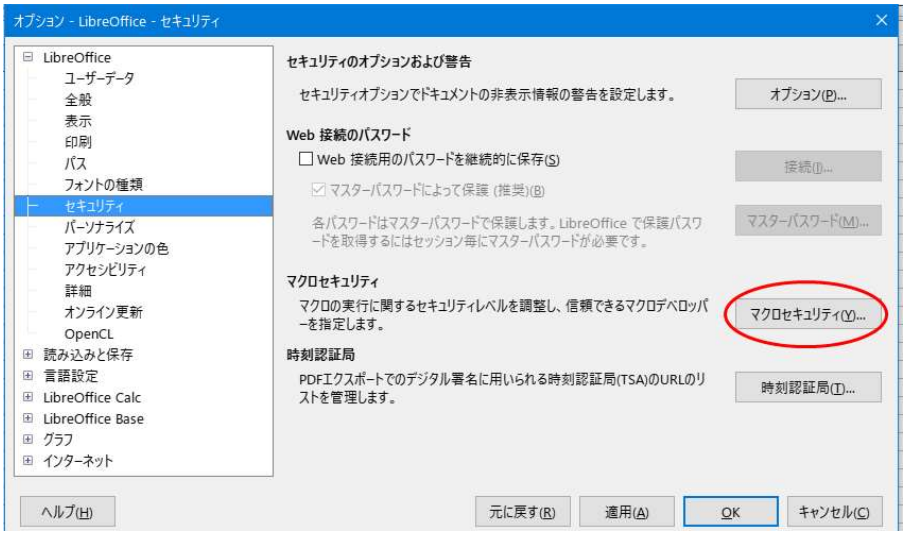
１．基礎知識

Calc の Basic を使うにはマクロを有効にする必要があります。

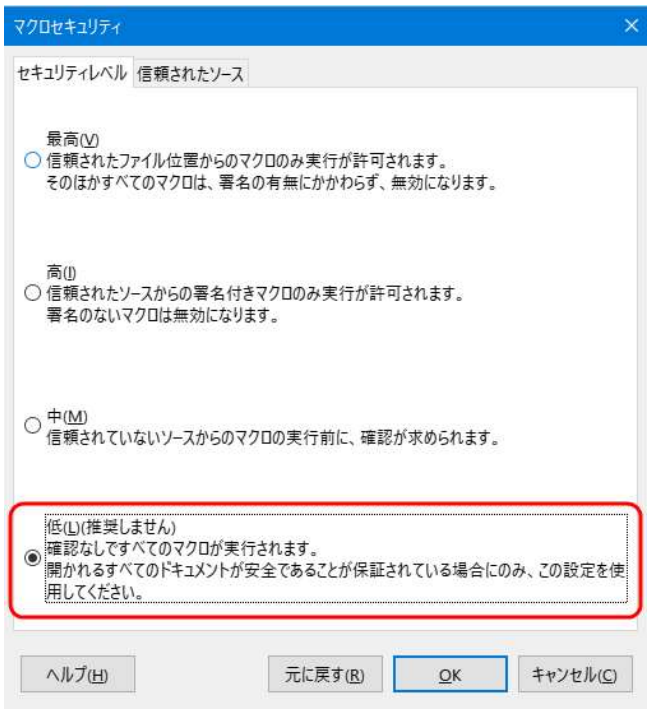
・メニューバーのツール・・・オプション



オプション　LibreOffice　セキュリティのダイアログ
・・・セキュリティ・・・マクロセキュリティ



マクロセキュリティのダイアログ・・・
低い（推奨しません）を選択・・・ＯＫ
マクロセキュリティのダイアログが消え、オプション　LibreOffice　セキュリティのダイアログの画面の右下のＯＫをクリックして終了。



２．マクロの構成

2-1　マクロを動かすためのプログラムを組む

・マクロを動かすためにプログラムが必要であり、マクロエディターを開き、
Basic エディターにそれぞれの作業をするプログラムを書き込んでいきます。

2-2　マクロの実行

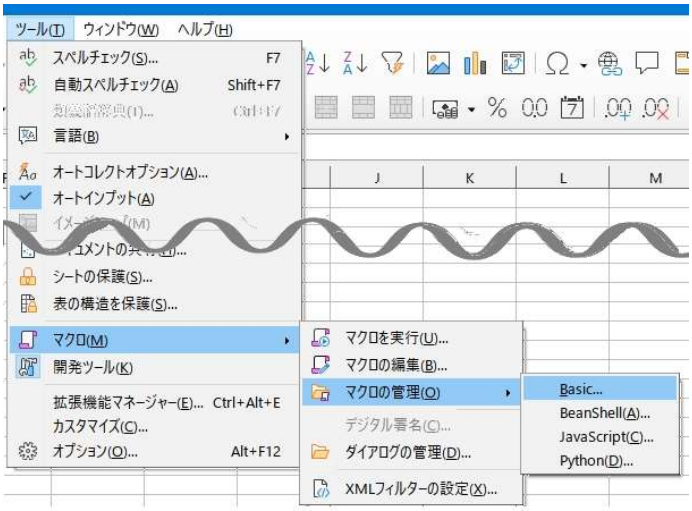
・ツールバーのマクロからマクロを実行をクリックしてプログラムを動かす。
・シートにイベントを発生させてプログラムを動かす。
・ダイアログ（フォーム）を作り挿入したコントロールにイベントを発生させてプログラムを動かす。

＊プログラムを組んでいるときもマクロエディターのメニューバーから実行ができます。
エラー確認ができるので便利です。

３．プログラムを書き込む手順

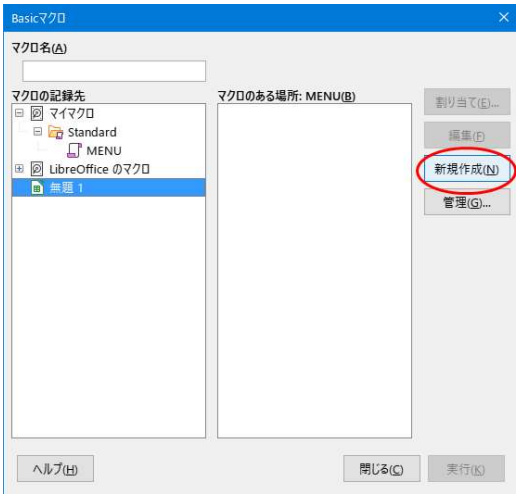
3-1. LibreOffice マクロエディター―を起動

ツール・・・マクロ・・・マクロの管理・・・Basic をクリック

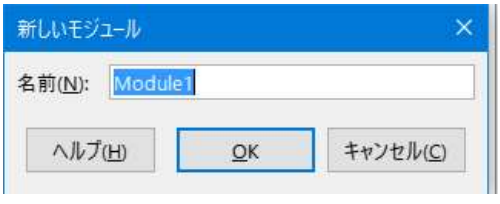


3-2.　Basic マクロのダイアログ

マクロの記録先・・・無題１（Calc に名前を付けて保存すると保存した名前になります）・・・新規作成をクリック

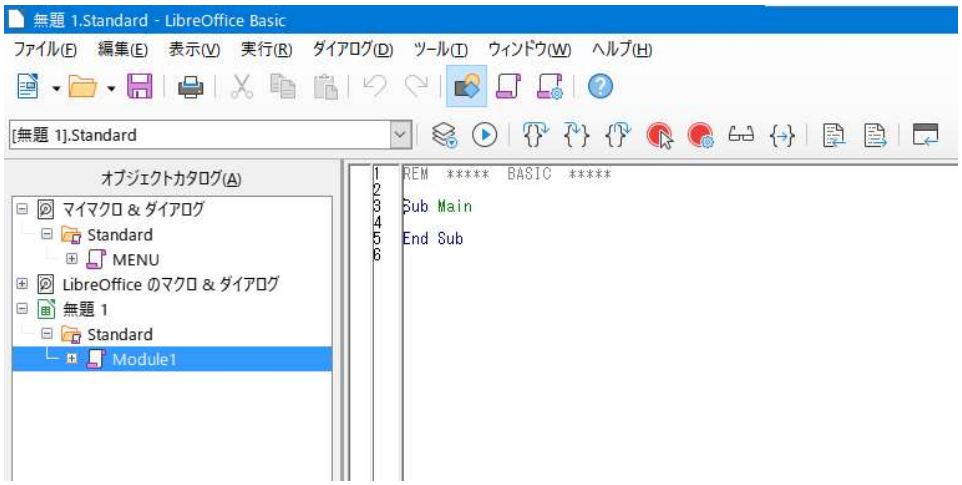


新しいモジュールのダイアログ
名前「Module1」・・・OK
＊　名前は変更可能ですが、ここでは Module1



で説明します。

左側にプロジェクト、右側にマクロエディタが表示



3-3. Basic Editor にマクロのプログラムコードを書いていきます。

プログラムは Sub 処理名（処理名は日本語不可）から始まり End Sub で終わります。

これをプロシージャ（Procedure）といいます。

【例】

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

Sub ENZAN
        Sheet1 = ThisComponent.Sheets(0)
        CellS1 = Sheet1.GetCellByPosition(0,0)
        CellR1 = Sheet1.GetCellRangeByName("A2")
          A=10
          B=2
          CellS1.Value = A*B
          CellR1.Value = A/B
End Sub
```

最初から Sub Main～End Sub のプロシージャが書かれていますが、Main は必要に応じて変更可能

変数 Sheet1 に Calc の Sheet1（Sheets(0)を設定

変数 CellS1 に Calc の Cell を ByPosition で設定

変数 CellR1 に Calc の Cell を RangeByName で設定

変数 A に 10 を代入、変数 B に 2 を代入

Cells1（セルの 1 行目、1 列目）に A×B の値を表示

Cellr1（A2）に A÷B の値を表示

Cell のプロパティ

数字　　：　Value 文字列　：　String 数字、文字列　：　Formula

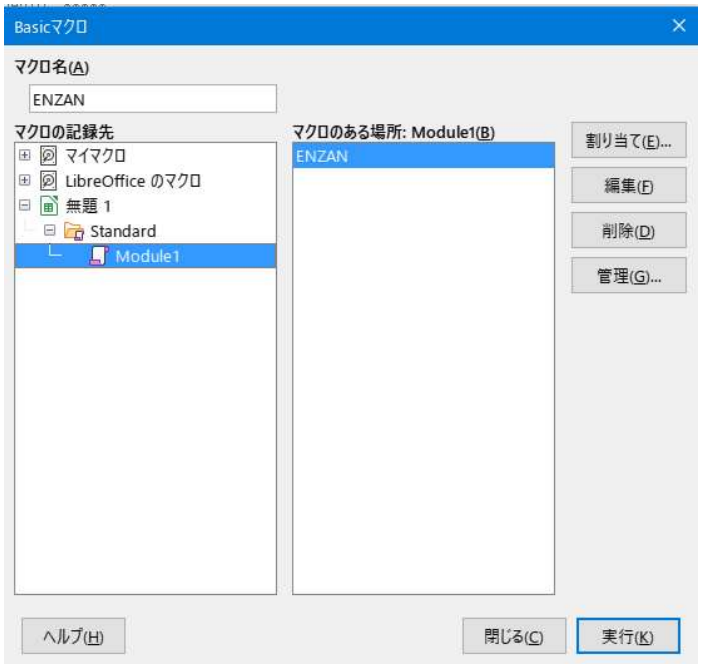
3-4.　マクロの実行

（マクロを実行する方法はシートからとマクロからの二通りあります。）

● シートの上のメニューバーからツールをクリック

マクロ・・・マクロを実行をクリック

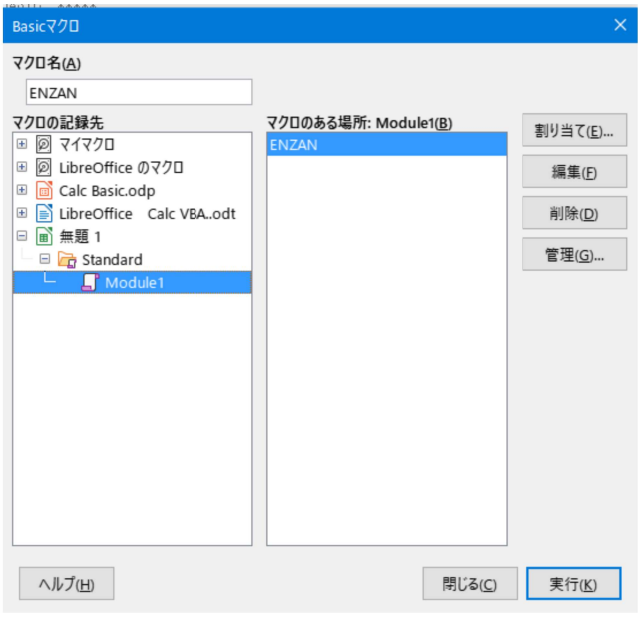
マクロセレクターのダイアログ・・・左側のライブラリーの下の無題１・・・Standard・・・Module1・・・右側のマクロ名の下の Enzan を選択・・・実行をクリック



● マクロの上のメニューバーから実行をクリック

プログラムを書いた後やプログラムのエディタをクリックした後はそのままプログラムが実行されます。

が、一度 Basic エディタを閉じて、再度開いて実行ボタンを押すと Basic マクロのダイアログが表示されます。　左側のマクロの記録先の下の無題１・・・Standard・・・Module1・・・右側のマクロのある場所の下の Enzan を選択・・・実行をクリック



４．Basic プログラムを組む

4-1.　シートの上をダブルクリックしてプログラムを動かす。

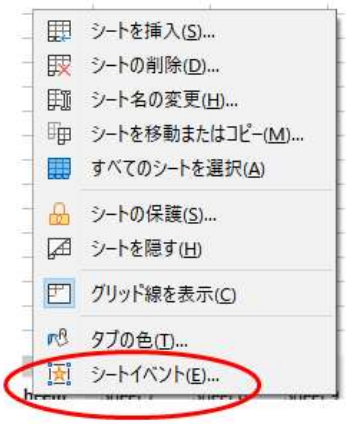
まず先にプログラムを作成します。

【例】

```
Sub W_Click
        Sheet1 = ThisComponent.Sheets(0)
        Cell1 = Sheet1.GetCellByPosition(0,2)
        oCell = ThisComponent.CurrentSelection
        CellR = oCell.CellAddress.Row
        CellC = oCell.CellAddress.Column
if CellC = 0 And CellR = 0 then
        Cell1.Value = 2000
end if

if CellC = 1 And CellR = 0 then
        Cell1.ClearContents(1)
end if

End Sub
```

＊　Sheet1 に Sheets(0)、（Sheet1)を設定
Cell1 にセルを選択する位置 CellByPosition(0,2)、セル A3 を設定
CellC に列（Column）、CellR に行（Row）の値を取得
Sheets1 のセル A1（列=0,行=0）をダブルクリックしたときはセル A3 に 2000 を表示、セル B1（列=1,行=0）をダブルクリックしたときはセル A3 はクリアーされる。

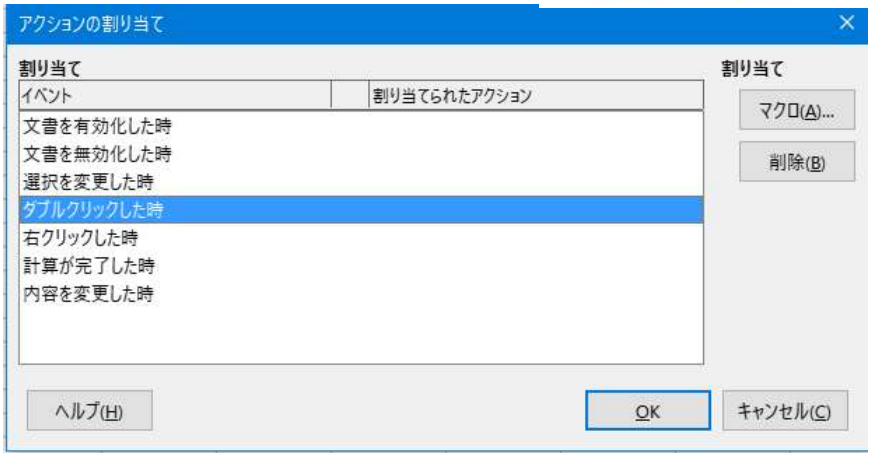
シートの下　”Sheet1”を右クリック・・・
シートイベントをクリック



アクションの割り当てのダイアログ表示

・・・　文書を有効化した時、文書を無効化した時、選択を変更した時、ダブルクリックした時、右クリックした時、計算が完了した時、内容を変更した時、（ここではダブルクリックした時を選択）

・・・　マクロをクリック



マクロセレクターダイアログ

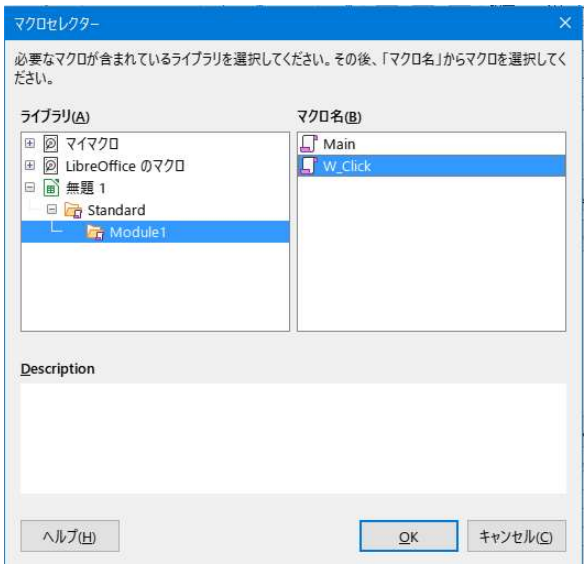
・・・無題１

・・・Standard

・・・Module1

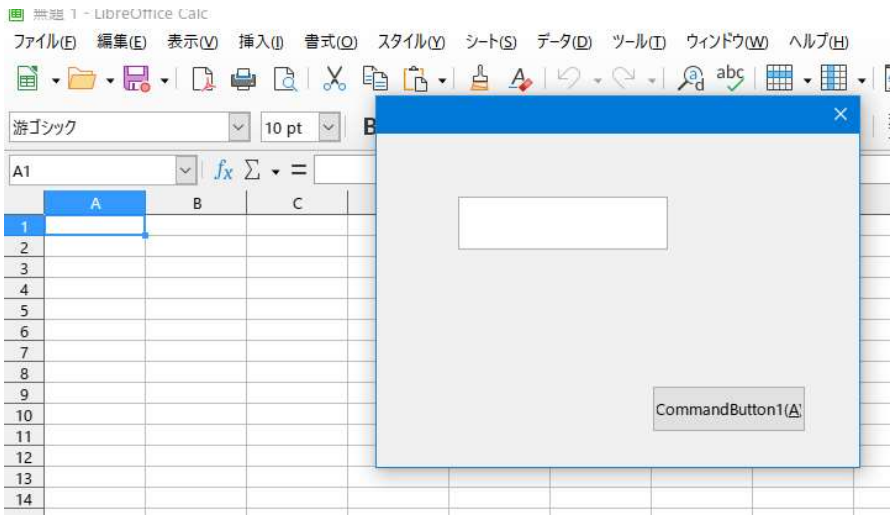
・・・右側の W_Click を
選択・・・OK（閉じます）

アクションの割り当ても OK で閉じます。



4-2.　ダイアログを使ってプログラミング

ダイアログを使ってマクロを動かす。



【例】TextBox に文字を入れて CommandButton を押し、
セル A1 に文字を挿入するプログラム

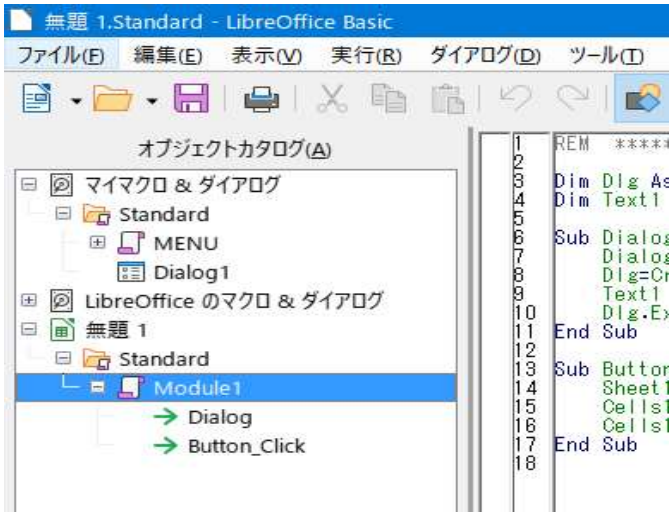
```
Dim Dlg As Object
Dim Text1 As Object

Sub Dialog
 DialogLibraries.LoadLibrary("Standard")
 Dlg=CreateUnoDialog(DialogLibraries.Standard.Dialog1)
 Text1 = Dlg.GetControl("TextField1")
 Dlg.Execute()
End Sub
```

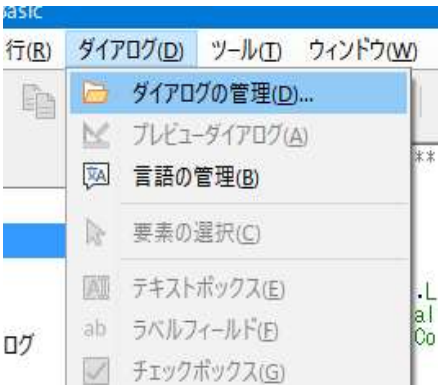
```
Sub Button_Click
 Sheet1 = ThisComponent.Sheets(0)
 Cells1 = Sheet1.GetCellByPosition(0,0)
 Cells1.Formula = Text1.Text
End Sub
```

● ダイアログの挿入

マクロエディタのメニューバーのダイアログをクリック・・・

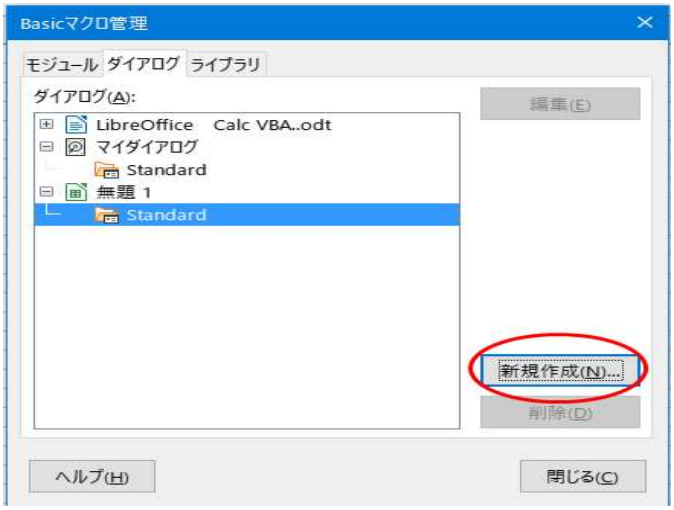
ダイアログの管理をクリック





Basic マクロ管理のダイアログ

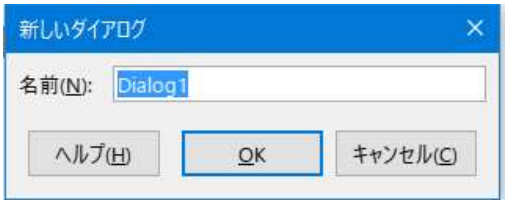
無題１・・・新規作成をクリック



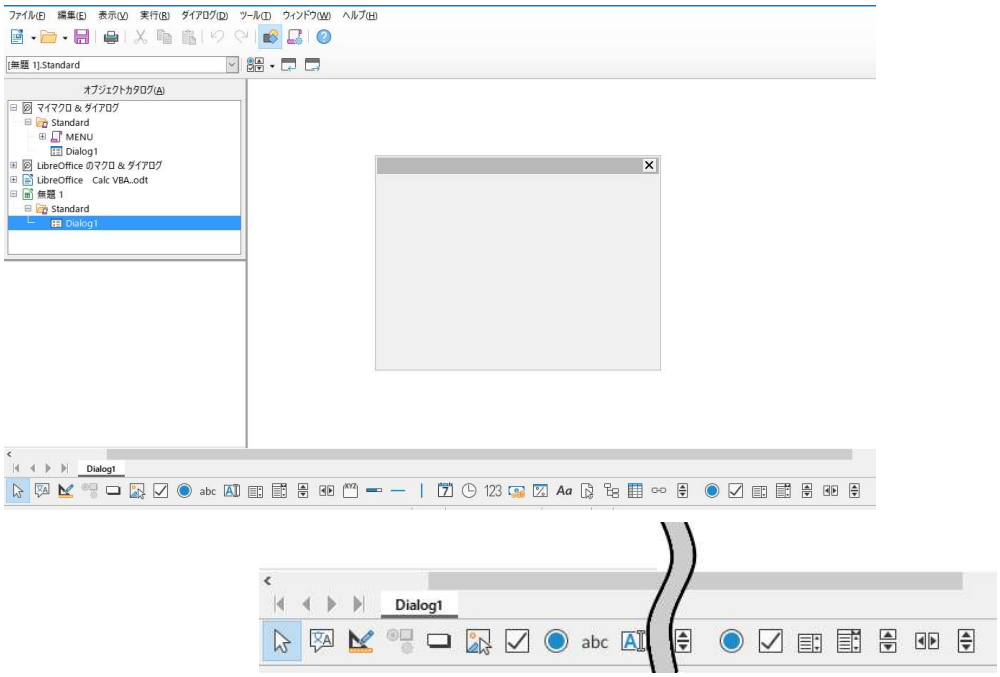
新しいダイアログ

名前は変更できますが、ここでも Module の時と同様そのまま Dialog1 にして

・・・ＯＫをクリック



Basic マクロ管理のダイアログを閉る。
新しいダイアログ「Dialog1」が表示され・・・OK
Standard の下に Dialog1 が表示されます。
上の無題１または、Standard をクリック（右上の編集が黒くなります）・・・編集をクリック
ダイアログエディタが表示（左側にオブジェクトカタログ、右側にダイアログが表示）、下にツールボックス（ダイアログ作成の部品が並んでします）



ダイアログに下のツールボックスから選択した TextBox と CommandButton を配置
左側のオブジェクトカタログの無題１の下の Standard をクリック
右側の Basic エディタにプログラムを書いていきます。

第２章　プログラミング（モジュール編）

１．セルの操作

セルを指定するコードは（Cells1 と Sheets1 は変数です）

セル名を使用する場合

```
Cells1 = Sheets1.GetCellRangeByName("A1")
```

セルの位置を使用する場合

```
Cells1 = Sheets1.GetCellByPosition(0,0)
```

セル名を使って範囲を指定する場合は

（例）`Cells1 = Sheets1.GetCellRangeByName("A1 : C1")`

セルの位置を使って範囲指定をする場合は

（例）`Cells1 = Sheets1.GetCellRangeByPosition(0,0,2,0)`

１－１．セルをアクティブにする

M1.　セル B5 にカーソルが移動して入力可能になります。

```
Sub Active_Cell
            oController = ThisComponent.getCurrentController()
            Sheet1=ThisComponent.Sheets(0)
            Cells1=Sheet1.GetCellByPosition(1,4)
            oController.Select(Cells1)
End Sub
```

１－２．セルにデーターを入力する

M2.　セルの名前を使った場合

```
'セル名を使用
Sub CellName_Input
            Sheet1 =ThisComponent.Sheets(0)
            Sheet1.GetCellRangeByName("A1").Formula=123456

            Cells1 = Sheet1.GetCellRangeByName("B1")
            Cells1.Formula=123456

Dim R_data(0) As Variant
            R_data(0)=Array("高岡市","高岡市","高岡市")
            Sheet1.GetCellRangeByName("A2:C2").setDataArray(R_data)

            Cells_R=Sheet1.GetCellRangeByName("A3:C3")
            Cells_R.setDataArray(R_data)
End Sub
```

セル（“A1”）に数字 123456 を入力
セル（“B1”）を変数 Cells1 に置き換え 123456 を入力
セル（“A2”）からセル（“C2”）に文字　“高岡市”を入力
セル（“A2”）からセル（“C2”）を変数 Cells_R に置き換え“高岡市”を入力

セルの位置を使った場合

```
'セル位置を使用
Sub CellPosition_Input
            Sheet1 =ThisComponent.Sheets(0)
            Sheet1.GetCellByPosition(0,0).Formula=123456

            Cells1 = Sheet1.GetCellByPosition(1,0)
            Cells1.Formula = 123456

Dim R_data(0) As Variant
            R_data(0)=Array("高岡市","高岡市","高岡市")
            Sheet1.GetCellRangeByPosition(0,1,2,1).setDataArray(R_data)
```

```
        Cells_R=Sheet1.GetCellRangeByPosition(0,2,2,2)
        Cells_R.setDataArray(R_data)
End Sub
```

１－３．セルをコピー、切り取り、貼り付け

「A1にプロパティをValueで数字　123456　A2にプロパティをStringで文字あいうえお　A3にプロパティをFormulaで数字　123456　A4にプロパティをFormulaで文字　あいうえお」と入力

＊　セルのプロパティ

Value・・・　書式設定されていない数値

String・・・ 文字

Formula・・・数字も文字も認識

セルA1:A4をコピーしてセルB1に貼り付け

M3.　コピー(Copy)＆貼り付け(Paste)

```
'セルのコピー　＆　貼り付け
Sub Copy_Paste
    Sheet1=ThisComponent.Sheets(0)
    Cell1=Sheet1.GetCellRangeByName("A1")
    Cell2=Sheet1.GetCellRangeByName("A2")
    Cell3=Sheet1.GetCellRangeByName("A3")
    Cell4=Sheet1.GetCellRangeByName("A4")

    Cell1.Value = 123456
    Cell2.String = "あいうえお"
    Cell3.Formula = 123456
    Cell4.Formula = "あいうえお"

    C_Cell=Sheet1.GetCellRangeByName("A1:A4").GetRangeAddress()
    P_Cell=Sheet1.GetCellRangeByName("B1").GetCellAddress()

    Sheet1.CopyRange(P_Cell,C_Cell)
End Sub
```

M4.　セルのクリア

値	フラグ	説明
1	VALUE	日付や時刻として書式設定されていない数値
2	DATETIME	日付や時刻として書式設定されている数値
4	STRING	文字列
8	ANNOTATION	セルに付けられたコメント

16	FORMULA	計算式
32	HARDATTR	セルに直接指定された書式
64	STYLES	間接的に指定された書式
128	OBJECT	セルに配置された描画オブジェクト
256	EDITATTR	セル内の一部のテキストに対してのみ指定された書式
511	全指定	

```
'セルのクリア
Sub Cell_Clear
    Sheet1=ThisComponent.Sheets(0)
    Sheet1.GetCellRangeByName("B1").ClearContents(com.sun.star.sheet.CellFlags.FORMULA)
    Sheet1.GetCellRangeByName("C1").ClearContents(4)
    Sheet1.GetCellRangeByName("D1").ClearContents(17)
End Sub
```

セルB1の数式をクリア（フラグを大文字で書くこと！！）
セルC1の文字列をクリア
セルD1の数字と数式をクリア
セルB1はClearContents の後に(com.sun.star.sheet.CellFlags.FORMULA)
セルC1、D1はClearContents の後に(値)

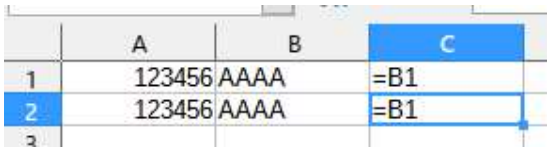
1-4. 書式のコピー

M5.　セルのコピー＆書式の貼り付け

A2にA1、B2にB1、C2にC1の書式を張り付け。

```
Sub Copy_Format
    Sheet=ThisComponent.Sheets(0)
    Sheet.GetCellRangeByName("A2").value=Sheet.GetCellRangeByName("A1").Value
    Sheet.GetCellRangeByName("B2").String=Sheet.GetCellRangeByName("B1").String
    Sheet.GetCellRangeByName("C2").Formula=Sheet.GetCellRangeByName("C1").Formula
End Sub
```

A2は数字（Value）
B2は文字（String）
C2は数字は数字、文字は文字、数式は数式でコピーできる（Formula）



1-5. セル範囲を削除する

M6.　セルA1からB2まで削除して左へシフトします。

```
Sub Cell_Delete
        Sheet1=ThisComponent.Sheets(0)
        C_Cell=Sheet1.GetCellRangeByName("A1:b2").GetRangeAddress()
```

```
        Sheet1.removeRange(C_Cell, com.sun.star.sheet.CellDeleteMode.LEFT)
End Sub
```

com.sun.star.sheet.CellDeleteMode.LEFT	左方向にシフト
com.sun.star.sheet.CellDeleteMode.UP	上方向にシフト

列を削除

M7.　列を削除

```
Sub Column_Delete
        Sheet1=ThisComponent.Sheets(0)
        C_Cell=Sheet1.GetCellRangeByName("A1:B2").GetRangeAddress()
        Sheet1.removeRange(C_Cell, com.sun.star.sheet.CellDeleteMode.COLUMNS)
End Sub
```

A1：B2 でも A 列から B 列までの全てが削除されます。

M8.　行を削除　A3:B4 でも 3 行から 4 行まで全てが削除されます。

```
Sub Row_Delete
        Sheet1=ThisComponent.Sheets(0)
        C_Cell=Sheet1.GetCellRangeByName("A3:B4").GetRangeAddress()
        Sheet1.removeRange(C_Cell, com.sun.star.sheet.CellDeleteMode.ROWS)
End Sub
```

1-6. セル範囲をクリアー

M9.　セル A1:C5 までの範囲をクリアーします。（M4 の応用）

```
Sub Cell_Clear
    Sheet1=ThisComponent.Sheets(0)
    Sheet1.GetCellRangeByName("A3:D5").ClearContents(511)
End Sub
```

1-7. セルを挿入

M10.　セルの挿入：セル A5:C8 まで空白を挿入して右側にシフトする

列の挿入：A 列～B 列（A3:B4)を挿入

```
'セルの挿入
Sub Insert_Right
    Sheet1=ThisComponent.Sheets(0)
    C_Cell=Sheet1.GetCellRangeByName("A3:B4").GetRangeAddress()
    Sheet1.InsertCells(C_Cell, com.sun.star.sheet.CellInsertMode.RIGHT)
End Sub

'列の挿入
Sub Insert_Column
```

```
    Sheet1=ThisComponent.Sheets(0)
    C_Cell=Sheet1.GetCellRangeByName("A3:B4").GetRangeAddress()
    Sheet1.InsertCells(C_Cell, com.sun.star.sheet.CellDeleteMode.COLUMNS)
End Sub
```

C_Cell, com.sun.star.sheet.CellInsertMode.RIGHT	セルを右方向にシフト
C_Cell, com.sun.star.sheet.CellInsertMode.DOWN	セルを下方向にシフト
C_Cell, com.sun.star.sheet.CellDeleteMode.COLUMNS	列全体を右方向にシフト
C_Cell, com.sun.star.sheet.CellDeleteMode.ROWS	行全体を右方向にシフト

１－８．セルの幅を設定

M11.　セルのＡ列目～Ｂ列目までの幅を2cmに、またＣ列目の幅を一番多い文字列の長さに合わせる。

＊　RangeByName ("A1:B1")でもいいが、("C")はエラーが発生 ("C1")または("C:C")と入力

```
Sub Width
                Sheet1=ThisComponent.Sheets(0)
                Cell1=Sheet1.GetCellRangeByName("A:B")
                Cell2=Sheet1.GetCellRangeByName("C:C")
                Cell1.GetColumns().Width=2000
                Cell2.getColumns.OptimalWidth = true
End Sub
```

１－９．セルの高さを設定

M12.　セルの１行目 (A1)の高さを1cmに、また２行目 (A2)の高さを一番大きい文字列の高さに合わせる。

```
Sub Height
                Sheet1=ThisComponent.Sheets(0)
                Cell1=Sheet1.GetCellRangeByName("A1")
                Cell2=Sheet1.GetCellRangeByName("A2")
                Cell1.GetRows().Height=1000
                Cell2.getRows.OptimalHeight = true
End Sub
```

１-10. フォントを設定

M13.　セルA1の文字をHG正楷書体-PRO、文字のサイズを20ポイント、
文字を太字、斜体、文字の色を赤、文字列に２重線のアンダーライン

```
Sub Font
        Sheet1=ThisComponent.Sheets(0)
```

```
        Cell1=Sheet1.GetCellRangeByName("A1")

        With Cell1
          .CharFontName = "HG正楷書体-PRO"
          .CharFontNameAsian = "HG正楷書体-PRO"
          .CharHeight = 20
          .CharHeightAsian = 20
          .CharWeight = com.sun.star.awt.FontWeight.BOLD
          .CharWeightAsian = com.sun.star.awt.FontWeight.BOLD
          .CharPosture = com.sun.star.awt.FontSlant.ITALIC
          .CharPostureAsian = com.sun.star.awt.FontSlant.ITALIC
          .CharColor = RGB(255,0,0)
          .CharUnderline = com.sun.star.awt.FontUnderline.DOUBLE
        End With

End Sub
```

内容	
CharFontName = "HG行書体" CharFontNameAsian = "HG行書体"	西洋諸言語用フォント アジア諸言語用フォント
CharHeight = 15	フォントサイズをポイント単位で設定
CharWeight = com.sun.star.awt.FontWeight.BOLD	BOLD：太字 DONTKNOW：標準(細字)
CharPosture = com.sun.star.awt.FontSlant.ITALIC	ITALIC：斜体 NONE：標準
＊ FontName（フォント名）、Height（フォントサイズ）、Weight（フォント太字）、 Posture(斜体) は西洋諸言語用フォントをアジア諸言語用フォントそれぞれに設定	
CharColor = RGB(255,0,0)	色をRGBで設定 RGB(255,0,0) 左から 赤、緑、青
CharUnderline = com.sun.star.awt.FontUnderline.DOUBLE	UnderLine NONE ・・・何もしない SINGLE ・・・1本線 DOUBLE ・・・2重線 DOTTED ・・・点線 DONTKNOW ・・・線の取り消し＝NONE

DASH	・・・破線
LONGDASH	・・・長い破
DASHDOT	・・・1点付き破線
DASHDOTDOT	・・・2点付き破線
SMALLWAVE	・・・小さい波線
WAVE	・・・波線
DOUBLEWAVE	・・・波線2重
BOLD	・・・太線
BOLDDOTTED	・・・太い点線
BOLDDASH	・・・太い破線
BOLDLONGDASH	・・・太く長い破線
BOLDDASHDOT	・・・太い1点付き破線
BOLDDASHDOTDOT	・・太い2点付き破線
BOLDWAVE	・・・太い波線

1-11. セルの背景色を設定

M14. セルA1背景を黄色

```
Sub Cell_Color
        Sheet1 = ThisComponent.Sheets(0)
        Cell1 = Sheet1.GetCellRangeByName("A1")

        Cell1.CellBackColor = RGB(255,255,0)
End Sub
```

1-12. 数値や日付の表示形式の設定 (NumberFormats)

M15. あらかじめ、セルA1、B1数字 123456を入力、C1,D1に＝Now()を入力

A1は12,3456円・・・3桁ごとに句点と最後に円を表示

B1は12,3456 ・・・3桁ごとに句点を表示

C1は現在の日付 令和y年mm月dd日(w曜日)・・・令和年月日（曜日）

D1は現在の日付Ry年mm月dd日(w) hh:mm:ss・・・（現在の日付）と入力された値をR年月日(曜)時刻を表示

年月日、曜日、時刻はすべて大文字で入力します。

西暦を4桁表示はYYYY 下2桁表示はYY 和暦（令和yy）GGEE （令和y）GGGE （令yy）GGEE （令y）GGE （Ryy）GEE （Ry）GE

セルのフォーマットはNumberFormats を使う事で、文字, 数値, 通貨表示にできます。

NumberFormats.queryKey を使って、指定したフォーマットが存在するか確認します。

該当する場合は、そのID番号を返します。該当しない場合は、-1 を返します。

該当しない場合は、新たに追加しないといけませんので、NumberFormats.addNew でフォーマットを追加します。

```
Sub NumberFmt
        Dim NumberFormats As Object
        Dim oLocale As New com.sun.star.lang.Locale
        Dim DF(3) As String
        Dim KeyNo(3) As Long
        Dim i As Integer
                Doc=ThisComponent
                Sheet1=Doc.Sheets(0)
                Cell1=Sheet1.getCellRangeByName("A1")
                Cell2=Sheet1.getCellRangeByName("B1")
                Cell3=Sheet1.getCellRangeByName("C1")
                Cell4=Sheet1.getCellRangeByName("D1")

                NumberFormats = Doc.NumberFormats
                  DF(0)="#,##0.#0 円"
                  DF(1)="#,##0"
                  DF(2)="GGGE 年 MM 月 DD 日(AAAA)"
                  DF(3)="GE 年 MM 月 DD 日(AAA) HH:MM:SS"

                For i=0 to 3
                  KeyNo(i)=NumberFormats.queryKey(DF(i), oLocale, False)
                          If KeyNo(i) = -1 Then
                              KeyNo(i) = NumberFormats.addNew(DF(i), oLocale)
                          End If
                Next

                Cell1.NumberFormat = KeyNo(0)
                Cell2.NumberFormat = KeyNo(1)
                Cell3.NumberFormat = KeyNo(2)
                Cell4.NumberFormat = KeyNo(3)
End Sub
```

M16.　Now()関数を日付変数Day1にする

セル(A1)は、日付や時刻、yyyy 年 mm 月 mm 日　h 時 mm 分 ss 秒で表示

＊　Format(Day1,"ggge 年 mm 月 dd 日 (aaaa) hh:mm:ss")は小文字でも可能です

セル(B1)は、数字、###,##0.00 で表示

```
Sub FormatString()
Dim Day1 As Date
        Sheet1 = ThisComponent.Sheets(0)
        Cell1 = Sheet1.GetCellRangeByName("A1")
```

```
        Cell2 = Sheet1.GetCellRangeByName("B1")
        Day1 = Now()
        Cell1.String = Format(Day1,"ggge 年 mm 月 dd 日 (aaaa) hh:mm:ss")
        Cell2.String = Format("123456.789","###,###.00")
End Sub
```

1-13. セル内の配置を指定

1）文字の配置

横方向に揃えるプロパティ　HoriJustify

縦方向に揃えるプロパティ　VertJustify

M17.　セル A1 の文字が横中央揃え縦中央揃えにする

```
Sub Cell_HoriJustfy
        Sheet1 = ThisComponent.Sheets(0)
        Cell1 = Sheet1.GetCellRangeByName("A1")
        Cell1.String = "アイウエオ"
        Cell1.HoriJustify = com.sun.star.table.CellHoriJustify.CENTER
        Cell1.VertJustify = com.sun.star.table.CellVertJustify.CENTER
End Sub
```

＊ HoriJustify（横方向）のプロパティで指定できる定数

定数	内容
com.sun.star.table.CellHoriJustify.STANDART	デフォルト
com.sun.star.table.CellHoriJustify.LEFT	左揃え
com.sun.star.table.CellHoriJustify.CENTER	中央揃え
com.sun.star.table.CellHoriJustify.RIGHT	右揃え

＊ VertJustfy（縦方向）のプロパティで指定できる定数

定数	内容
com.sun.star.table.CellVertJustify.STANDART	デフォルト
com.sun.star.table.CellVertJustify.TOP	上詰め
com.sun.star.table.CellVertJustify.CENTER	中央揃え
com.sun.star.table.CellVertJustify.BOTTOM	下詰め

2）セル内文字列の制御

M18.　セル A1 は折り返して全体を表示する

セル A2 は縮小して全体を表示

セル A3：B3 のセルを結合して A3（A3:B3）を中央揃えで表示

セル A4 は文字列を 45 度傾けて表示

セル A5 を縦書きで表示

```
Sub Cell_Shoshiki
```

```
        Sheet1 = ThisComponent.Sheets(0)
        Cell1 = Sheet1.GetCellRangeByName("A1")
        Cell2 = Sheet1.GetCellRangeByName("A2")
        RCell1 = Sheet1.GetCellRangeByName("A3:B3")
        Cell3 = Sheet1.GetCellRangeByName("A3")
        Cell4 = Sheet1.GetCellRangeByName("A4")
        Cell5 = Sheet1.GetCellRangeByName("A5")
        Cell1.Formula = "文字を折り返して全体を表示します。"
        Cell2.Formula = "文字を縮小して全体を表示します。"
        Cell1.IsTextWrapped = True
        Cell2.ShrinkToFit = True
        RCell1.merge(true)
        RCell1.HoriJustify = com.sun.star.table.CellHoriJustify.CENTER
        Cell3.Formula = 1234567
        Cell4.Formula = "45° 傾けます。"
        Cell4.RotateAngle = -4500
        With Cell5
          .Formula = "この文書は縦書きで表示します。"
          .IsTextWrapped = True
          .Orientation = 3
          .AsianVerticalMode = True
         End With
End Sub
```

＊　セルの書式設定

プロパティ	内容
IsTextWrapped = True	True で折り返して全体を表示する（False で解除）
ShrinkToFit = True	True で縮小して全体を表示する（False で解除）
Merge(true)	True でセルを結合する（False で解除）
RotateAngle = 4500	文字列を傾ける 0～9000（0° ～90° ）　-1～-9000（-1° ～-90° ）
Orientation = 3 AsianVerticalMode = True	Orientation = 3 AsianVerticalMode = True で縦書きを設定する（False で解除）
HoriJustify = com.sun.star.table.CellHoriJustify.CENTER 文字の配置 ： STANDARD （標準。文字列の場合は左寄せ、数字の場合は右寄せ） RIGHT　右寄せ CENTER　中央揃え LEFT　左寄せ BLOCK　折り返して全体を表示 REPEAT　縮小して全体を表示	

２．シートの操作

２-1. シートを追加する

M20.　シート数を取得（0から始まります）、シート名をSheet & シート数 + 1、順番をシート数（最後）、1番先頭は0

シート名、順番は適宜に指定

```
Sub New_Sheet
Dim L_St As Integer
        L_St = Thiscomponent.Sheets.Count
        ThisComponent.Sheets.insertNewByName("Sheet" & Str(L_St+1),L_St)
End Sub
```

２-２．シートをアクティブにする

M21.　Sheet2をアクティブにする

```
'***** シートをアクティブにする　*****
'シートの変数をシート名で指定
Sub Sheet_Active1
        oCTRL = ThisComponent.getCurrentController()
        Sheet1 = ThisComponent.Sheets()
        oCTRL.setActiveSheet(Sheet1.getByName("Sheet2"))
End Sub

'シートの変数をシートの順番で指定（0から始まり）
Sub Sheet_Active2
        oCTRL = ThisComponent.getCurrentController()
        Sheet1 = ThisComponent.Sheets(0)
        oCTRL.setActiveSheet(Sheet1)
End Sub
```

２-３．シートの名前を変更する

M22.　Sheet1をNewNameに変更

```
Sub Rename_Sheet
        Sheet1 = ThisComponent.Sheets(0)
        Sheet1.Name="NewName"
End Sub
```

＊　シートの順番をシート名を指定する場合はThisConponent.の後のSheets(0)をSheets.getByName("シート名")にする

２-４．シートを移動する

M23.　Sheet1を一番後ろに移動する（移動するシート番号はシートカウントの変数L_St＋1を数字に置き換える）

```
Sub Rename_Sheet
Dim L_St As Integer
        L_St = Thiscomponent.Sheets.Count + 1
        ThisComponent.Sheets.moveByName("Sheet1", L_St)
End Sub
```

２-５．シートをコピーする

M24.　Sheet1 を一番後ろに C_Sheet と名前を付けてコピーする

```
Sub Sheet_CopyIndex
Dim L_St As Integer
        L_St = Thiscomponent.Sheets.Count + 1
        ThisComponent.Sheets.copyByName("Sheet1", "C_Sheet", L_St)
End Sub
```

２-６．シートを削除する

M25.　C_Sheet を削除する

```
Sub Remove_Sheet
        Sheet = ThisComponent.getSheets()
        Sheet.removeByName("C_Sheet")
End Sub
```

２-７．シートを保護する

M26.　Sheet1 の保護を解除（パスワード　password）
Sheet1 のセル全体のロック解除（規定ではシートはロックされている）
セル A1～C1 をロックする
Sheet1 の保護を設定（パスワード　password）
パスワードは省略可能です

```
'シートのセルの一部を保護
Sub SheetCell_Protect
        oCellProct = createUnoStruct("com.sun.star.util.CellProtection")
        oCTRL = ThisComponent.CurrentController
        oCTRL.selection.CellProtection=oCellProct
        Sheet1 = ThisComponent.Sheets(0)

  'シートの保護解除
        Sheet1.UnProtect("password")

  'シートのロック解除
        oCTRL.Select(Sheet1)
        oCellProct.IsLocked = False
```

```
  '保護するセルのロック設定
        oRange = Sheet1.getCellRangeByName("A1:C1")
        oRange.setPropertyToDefault("CellProtection")
        oCTRL.Select(oRange)
        oCellProct.IsLocked = True

  'シートの保護設定
        Sheet1.Protect("password")
End Sub
```

３．ブック（ドキュメント）の操作

３-１．ブック（ドキュメント）を作成する

ブックを開く（名前はフルパスで、拡張子（ods)もつけて）

```
   oName = ""C:¥TEST¥名前.ods""
   oUrl = ConvertToURL(oName)
   oDoc = StarDesktop.loadComponentFromURL(oUrl, "_blank", 0, Dummy())
```

新しいブックを作成

```
   oDoc = StarDesktop.loadComponentFromURL( "private:factory/scalc", "_blank", 0, Dummy())
```

名前を付ける（名前はフルパスで、拡張子（ods)もつけて）

```
   oCName = ConvertToUrl("C:¥TEST¥新しい名前.ods")
```

保存する

```
   oDoc.storeAsURL(oCName, Dummy())
```

ブックを閉じる（oDoc.dispose、oDoc.close(True)も変わらなかった）

```
   oDoc.dispose
   oDoc.close(True)
```

M27.　新しいブックを作成してNewCalc.odsと名前を付けて保存

```
'新しいブックを作成新しい名前で保存
Sub oCalcOpen_Save
Dim Dummy()
  oDoc = StarDesktop.loadComponentFromURL( "private:factory/scalc", "_blank", 0, Dummy())
  oCName = ConvertToUrl("D:¥TEST¥NewCalc.ods")
  oDoc.storeAsURL(oCName, Dummy())
  oDoc.dispose
End Sub
```

３-２．ブック（ドキュメント）を開く

M28.　DのTESTのフォルダのTEST.odsを開く

```
'ブックを開く
Sub oCalcOpen_Name
```

```
        Dim Dummy()
        oName = "D:¥TEST¥TEST.ods"
        oUrl = ConvertToURL(oName)
        oDoc = StarDesktop.loadComponentFromURL(oUrl, "_blank", 0, Dummy())
End Sub
```

３−３．組み込みダイアログボックスを使ってドキュメントを開く

M29.　エクスプローラーのダイアログボックスを開いてファイルを選択して開く

```
Sub OpenByDialog
Dim Dummy()
        FilePicker=createUnoService("com.sun.star.ui.dialogs.FilePicker")
        FilePicker.execute()
        Files=FilePicker.getFiles()
        FileOpenDialog=Files(0)
        FName=FileOpenDialog
        oUrl = ConvertToURL(FName)
        oDoc = StarDesktop.loadComponentFromURL(oUrl, "_blank", 0, Dummy())
End Sub
```

＊ MicrosoftのExcellはCalc、WordはWriter、写真はDrowで開くことを確認しました。

３−４．複数のドキュメントを開く

M30.　D:¥TEST¥TEST1.odsとD:¥TEST¥TEST2.odsを開き各ドキュメント（マクロのあるドキュメントを含む）ファイル名（Title）をメッセージボックスに表示し、TEST1.odsとTEST2.odsを閉じる

```
'複数のドキュメントを開く
Sub Open_Documents
Dim Dummy()
    oName1 = "D:¥TEST¥TEST1.ods"
    oName2 = "D:¥TEST¥TEST2.ods"
    oUrl1 = ConvertToURL(oName1)
    oUrl2 = ConvertToURL(oName2)
    oDoc1 = StarDesktop.loadComponentFromURL(oUrl1, "_blank", 0, Dummy())
    oDoc2 = StarDesktop.loadComponentFromURL(oUrl2, "_blank", 0, Dummy())
    oDoc0 = ThisComponent
    FName0 = oDoc0.Title
    FName1 = oDoc1.Title
    FName2 = oDoc2.Title
    MsgBox(FName0 & Chr$(13) & FName1 & Chr$(13) & FName2)
    oDoc1.dispose
```

```
    oDoc2.Dispose
End Sub
```

３-５．開いたドキュメントを操作

M31.　ドキュメント（プログラミング.ods）からドキュメント（D:¥TEST¥TEST.ods）を開きシート１のセル（A1)にABCDEを入力、保存して閉じる

```
'ドキュメント　開く　保存　閉じる
Sub Open_Save_Close
Dim Dummy()
    oName = "D:¥TEST¥TEST.ods"
    oUrl = ConvertToURL(oName)
    oDoc = StarDesktop.loadComponentFromURL(oUrl, "_blank", 0, Dummy())
    Sheet1=oDoc.Sheets(0)
    Cell1=Sheet1.getCellRangeByName("A1")
    Cell1.String="ABCDE"
    oDoc.store()
    oDoc.close(True)
End Sub
```

４．ファイルの操作

４-１．ファイル、フォルダーを探す

M32.　フォルダーの存在の確認

```
'フォルダーの確認
Sub Folder_Exist
        oFolder="D:¥TEST¥NewFolder"
        If FileExists(oFolder)=True Then
                Message=oFolder & "が存在します"
        Else Message=oFolder & "が存在しません"
        End if
        MsgBox(Message)
End Sub
```

M33.　ファイルの存在を確認

```
'ファイルの確認
Sub File_Exist
        oFile="D:¥TEST¥TEST.txt"
        If FileExists(oFile)=True Then
                Message=oFile & "が存在します"
        Else Message=oFile & "が存在しません"
```

```
        End if
        MsgBox(Message)
End Sub
```

M34.　C:¥TESTのフォルダーの中ファイル名を取得し、MsgBoxに表示する。

```
'ファイル名を取得
Sub NewFolder
        oFolder=Dir("D:¥TEST¥",0)

        Do While oFolder <> ""
        strMessage = strMessage & oFolder & Chr$(13)
        oFolder = Dir()
    Loop

    Msgbox(strMessage)
End Sub
```

M35.　D:¥TESTのフォルダーの中のフォルダー名を取得し、MsgBoxに表示する。

```
'フォルダー名を取得
Sub NewFolder
        oFolder=Dir("D:¥TEST¥",16)

        Do While oFolder <> ""
        strMessage = strMessage & oFolder & Chr$(13)
        oFolder = Dir()
    Loop

    Msgbox(strMessage)
End Sub
```

＊ Dir（フォルダー）内のデーターのパラメーターの値

Dir（フォルダー）内のデーター	パラメーター
通常のファイル	0
サブディレクトリ（フォルダー）	16

M36.　カレントディレクトリ（ワークブックのディレクトリ）を取得

```
'カレントディレクトリの取得
Sub CurrentFolder1
    oUrl = ThisComponent.getURL
    oFold=ConvertFromUrl(oUrl)
```

```
    i=1
    Do
        ui = InStr(i,oFold,"¥")
        if ui=0 then Exit Do
        li=ui-1
        i=i+1
    Loop

    MsgBox Left(oFold,li)

End Sub
```

＊　ThisComponent.getURL にすると、D:¥TEST¥bbbb.fil とファイルまで取得するので InStr 関数で￥の最後の位置を取得して Left 関数で位置-1 を指定して抽出

４－２．フォルダーの作成・削除

M37.　D:¥TEST のフォルダーにフォルダー（NewFolder）を作成

＊ MkDir と Path の間にスペースを入れる（Basic のシンタックスエラーが発生）

```
'フォルダーの作成
Sub New_Folder
   MkDir "D:¥TESR¥NewFolder"
End Sub
```

M38.　D:¥TEST の中の NewFolder というフォルダーを削除する

＊ RmDir と Path の間にスペースを入れる（Basic のシンタックスエラーが発生）

```
'フォルダーの削除
Sub Remove_Folder
   RmDir "D:¥TEST¥NewFolder"
End Sub
```

４－３．ファイルのコピー・移動

M39.　D:¥TEST¥TEST1 のフォルダーの libtest1.odt のファイルを TEST2 のフォルダーに libtest2.odt と名前を変更してコピー

＊ Filecopy と Path の間にスペースを入れる（Basic のシンタックスエラーが発生）

```
'ファイルを別名でコピー
Sub File_Copy
    Filecopy "D:¥TEST¥TEST1¥libtest1.odt","D:¥TEST¥TEST2¥libtest2.odt"
End Sub
```

４－４．ファイルの削除

M40.　D:¥TEST のフォルダーの TEST.txt ファイルを削除する

```
'ファイル削除
Sub File_Kill
    Kill "D:¥TEST¥TEST.txt"
End Sub
```

４－５．ファイルの書き込み（シーケンシャルファイル）

M41.　シーケンシャルファイルの書き込み

D:¥TEST¥Libo_fileにTTEST.txtにデーターを書き込みます。

ファイルがあればFor Output（出力モード）データーを書き込み、ファイルがなければFor Append（追加モード）を使ってデーターを書き込みます。

Freefile関数 ： 現在開いているファイルの次の番号を与えます。

FileExists ： ファイルの有無を確認する。

Print #ファイルNo. ： ファイルにデーターを書き込む

```
'シーケンシャルファイル(書き込み)
Sub Seqential_file_Writing
Dim d_file As String
Dim f_No As Integer

   d_file = "D:¥TEST¥Libo_file¥TEST.txt"
   f_No = Freefile

   if FileExists(d_file) then
        Open d_file For Append As #f_No
   Else Open d_file For Output As #f_No
   End If

    data1="ABCDE"
    data2="あいうえお"
    data3=12345
        Print #f_No, "1行目",data1,data2,data3
        Print #f_No, "2行目",data1,data2,data3
       Close #f_No
End Sub
```

＊ Print #f_No,とWrite #f_No,で書き込んだTEST.txtの内容

Print #f_No,	ABCDE　　あいうえお　　12345
Write #f_No,	"ABCDE","あいうえお",12345

４－６．ファイルの読み込み（シーケンシャルファイル）

M42.　シーケンシャルファイルの読み込み

Eof(ファイルNo.)（エンド　オブ　ファイル）・・・データーの最後まで読み込む

```
'シーケンシャルファイル(読み込み)
Sub Seqential_file_Reading
Dim f_Line As Variant
Dim d_file As String
Dim f_No As Integer
   d_file = "D:¥TEST¥Libo_file¥TEST.txt"
   f_No = Freefile
   if FileExists(d_file) then
            Open d_File For Input As #f_No
       Else MsgBox d_file & " がありません" : Exit Sub
   End If
   While not Eof(f_No)
           Line Input #f_No,f_Line
           MsgBox f_line
   Wend
   Close #f_No
End Sub
```

４－７．CSV ファイルの書き込み

M43. CSV ファイルの書き込み

```
'CSV ファイル(書き込み)
Sub CSV_file_Writing
Dim d_file As String
Dim f_No As Integer
   d_file = "D:¥TEST¥Libo_file¥TEST.csv"
   f_No = Freefile
   if FileExists(d_file) then
               Open d_file For Append As #f_No
   Else Open d_file For Output As #f_No
   End If
    data1="ABCDE"
    data2="あいうえお"
    data3=12345
        Write #f_No, data1,data2,data3
        Close #f_No
End Sub
```

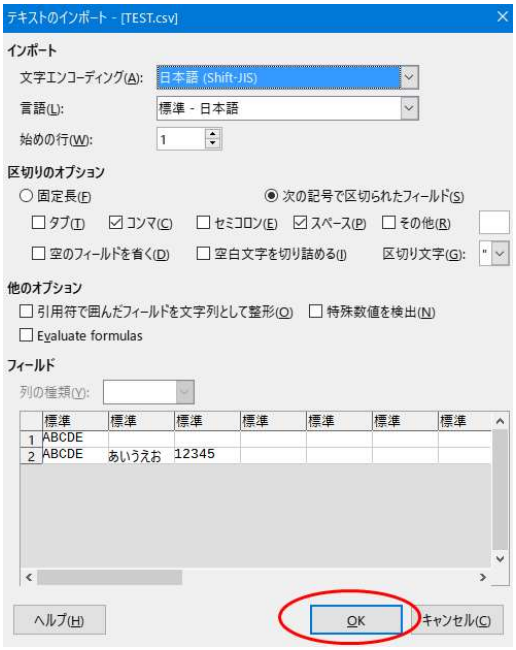
４－８．CSV ファイルを開く

M44.　CSV で書き込まれた D:¥TEST¥Libo_file¥TEST.csv を開く(Calc で開かれます)

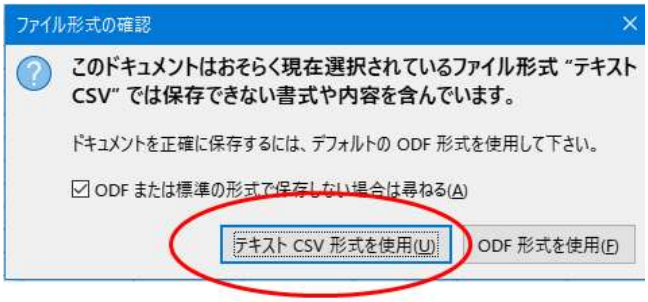
```
'CSV ファイルを開く
```

```
Sub CSV_file_Open
 Dim Dummy()
        oName = "D:¥TEST¥Libo_file¥TEST.csv"
        oUrl = ConvertToURL(oName)
        oDoc = StarDesktop.loadComponentFromURL(oUrl, "_blank", 0, Dummy())
End Sub
```

＊　csv ファイル起動時テキストのインポートと言うダイアログボックスが表示、ＯＫボタンを押すと Calc で表示されます。



＊　メニューバーのファイルの閉じるや右上の X（閉じるボタン）を押すとすうっと閉じる。
保存（名前を付けて保存）をクリックするとファイル形式確認のメッセージボックスが表示テキスト形式を使用をクリックして終了



＊　Print #f_No, と Write #f_No, で書き込んだ TEST.csv の内容

Print #f_No,	＊　Calc の場合 セルが 10 個ほど離れて表示される ABCDE　　あいうえお　　12345 ＊　Excel の場合 1 つのセルに表示される
Write #f_No,	1 つのセルに 1 つずつ表示される ABCDE　あいうえお　12345

従って、Write は csv ファイルの書き込みに有効です

４－９．ファイルの書き込み（ランダムアクセスファイル）

M45.　D:¥TEST¥Libo_file¥RndTEST.txt にフ Seek1 と Seek20 にそれぞれデーターを書き込み

PUT #1,フィールド No で書込みする

```
'ランダムアクセスファイル書込み
Sub RandomAccess_Put
Dim iNumber As Integer
Dim aFile As String
    aFile = "D:¥TEST¥Libo_file¥RndTEST.txt"
    iNumber = Freefile
    Open aFile For Random As #iNumber Len=32
    Seek #iNumber,1
    Put #iNumber,, "ここはテキストの始めの行です。"
    Put #iNumber,, "This is the second line of text"
    Put #iNumber,, "This is the third line of text"
    Seek #iNumber,20
    Put #iNumber,,"This is the text in record 20"
    Close #iNumber
End Sub
```

４-10. ファイルの読み込み（ランダムアクセスファイル）

M46.　D:¥TEST¥Libo_file¥RndTEST.txt のデーターを Get #1,フィールド No で読込み

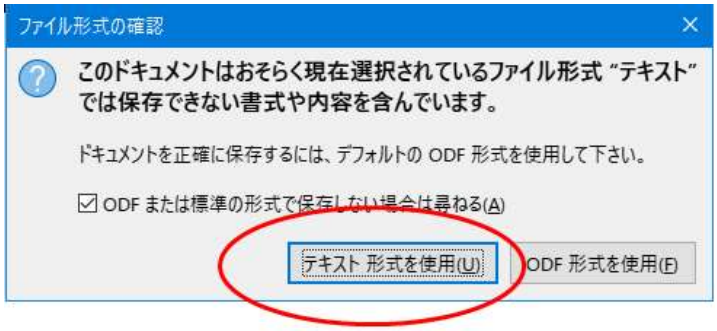
```
'ランダムアクセスファイル読み込み
Sub RandomAccess_Get
Dim iNumber As Integer
Dim sText As Variant
Dim aFile As String
    aFile = "D:¥TEST¥Libo_file¥RndTEST.txt"
    iNumber = Freefile
    Open aFile For Random As #iNumber Len=32
    Get #iNumber,,sText
    Print sText
    Get #iNumber,,sText
    Print sText
    Get #iNumber,,sText
    Print sText
    Get #iNumber,20,sText
    Print Lof(#iNumber)
    Close #iNumber
End Sub
```

4-11. テキストファイルを開く

M47.　D:¥TEST¥Libo_file¥TEST.txt を開く（Writer で開かれます）

```
'テキストファイルを開く
Sub Text_file_Open
 Dim Dummy()
        oName = "D:¥TEST¥Libo_file¥TEST.txt"
        oUrl = ConvertToURL(oName)
        oDoc = StarDesktop.loadComponentFromURL(oUrl, "_blank", 0,
Dummy())
End Sub
```

Writer で開かれる。　メニューバーのファイルの閉じるや右上の X（閉じるボタン）を押すとすうっと閉じる。保存（名前を付けて保存）をクリックするとファイル形式確認のメッセージボックスが表示テキスト形式を使用するをクリックして終了



5．印刷

M48.　プリンター、用紙規格、用紙方向、ページ、ページ数を指定して印刷

```
'プリンター、用紙規格、用紙方向、ページ、ページ数を指定して印刷
Sub Printer_Page
        Dim Doc as Object
                Doc=ThisComponent
        Dim oProps(5) as new com.sun.star.beans.PropertyValue
                oProps(0).Name = "PaperFormat"
                oProps(0).Value = com.sun.star.view.PaperFormat.A5
                oProps(1).Name="PaperOrientation"
                oProps(1).Value=com.sun.star.view.PaperOrientation.PORTRAIT
                oProps(2).Name= "Name"
                oProps(2).Value= "HP Officejet 4630 series (ネットワーク)"
                Doc.Printer=oProps()

                oProps(3).Name = "DuplexMode"
                oProps(3).Value = com.sun.star.view.DuplexMode.OFF
                oProps(4).Name = "Pages"
                oProps(4).Value = "1-2"
                oProps(5).Name = "CopyCount"
                oProps(5).Value = 2
                Doc.print(oProps())
```

```
End Sub
```

new com.sun.star.beans.PropertyValue		
Name	Value	説明
PaperFormat	com.sun.star.view.PaperFormat.用紙規格	用紙規格（A4、A3、B5、B4）
PaperOrientation	com.sun.star.view.PaperOrientation.PORTRAIT	縦向き
	com.sun.star.view.PaperOrientation.LANDSCAPE	横向き
Name	プリンター名	プリンター名
PaperSize	設定したPaperSize Dim PaperSize As New com.sun.star.awt.Size PaperSize.Width = 20000 PaperSize.Height = 20000	Width(幅)、Height(高さ)のサイズは1cm=1/1000
DuplexMode	com.sun.star.view.DuplexMode.UNKNOWN	
	com.sun.star.view.DuplexMode.OFF	両面印刷OFF
	com.sun.star.view.DuplexMode.LONGEDGE	用紙の長い方を軸に両面印刷
	com.sun.star.view.DuplexMode.SHORTEDGE	用紙の短い方を軸に両面印刷
Pages	x-z	(例)　1-3;5;8
CopyCount	Integer（整数）	整数で入力

M49.　ダイアログボックスを表示して印刷

```
'ダイアログボックスを表示して印刷
Sub ShowPrinterSettingDialog
   GlobalScope.BasicLibraries.LoadLibrary("Tools")
   DispatchSlot(5504)
End Sub
```

ダイアログボックスの種類	
DispatchSlot(5302)	プリンター設定
DispatchSlot(5504)	印刷ダイアログボックス

６．罫線

M50.　セルに罫線を引く

```
'指定された範囲のセルに罫線を引く
Sub Calc_Line
  'シート＆セルの設定
  oSheet=ThisComponent.Sheets(0)
  oRange=oSheet.getCellRangeByName("A1:B2")

  '罫線の設定
  oBorderLine = oRange.LeftBorder
  oBorderLine.Color = RGB(0,0,0)
  oBorderLine.LineWidth = 50
  oBorderLine.LineStyle = 0

  '罫線を引く
  With oRange
        .BottomBorder = oBorderLine
        .TopBorder = oBorderLine
        .LeftBorder = oBorderLine
        .RightBorder = oBorderLine
  End With
End Sub
```

M51.　罫線の種類

```
'罫線の種類
Sub Calc_Line
        Dim oRange(30) As Object
        Dim oBorderLine(30) As Object
        Sheet1 = ThisComponent.Sheets(0)
        LineW=100  '罫線の太さ  100  100：1mm
For Li=0 To 17
        oRange(Li)=Sheet1.getCellRangeByPosition(1,Li*2+1,3,Li*2+1)
                                          'Cell("B2:D2")からLi=17まで
        Cell= Sheet1.getCellByPosition(0,Li*2+1)
        Cell.Value=Li
        oBorderLine(Li)=oRange(Li).RightBorder
        oBorderLine(Li).Color = RGB(0, 0, 0)
        oBorderLine(Li).LineWidth = LineW
        oBorderLine(Li).LineStyle = Li
    With oRange(Li)
        .BottomBorder = oBorderLine(Li)
        .TopBorder = oBorderLine(Li)
```

```
        .LeftBorder = oBorderLine(Li)
        .RightBorder = oBorderLine(Li)
    End With
Next
  End Sub
```

＊　LineStyle　0：Line（実線）　1：Dot（点線）　2：Dash（破線）　3：二重線

罫線の種類（LineStyle）と罫線の太さ（LineWidth)の比較

（罫線の太さについては最後のページにに表が添付してあります。）

７．　データー

７－１．レコードの数を数える

M52.　セルＡの列１行目からデーターのあるの最後までの行の数と列の数をMsgBoxに出力
　　つまり、最終行と列の取得になる

```
'データーの最後の列と行を取得
Sub Last_Data
        oSheet=ThisComponent.Sheets(0)
        oRange = oSheet.getCellRangeByName("A1")
    oCursor = oSheet.createCursorByRange(oRange)
    oCursor.gotoEndOfUsedArea(True)
    Msgbox("最終行 : " & oCursor.Rows.Count & Chr$(13) & _
           "最終列 : " & oCursor.Columns.Count)
End Sub
```

７－２．データの並べ替え

M53.　セルB2:D32のデータをD2をキーに降順（小さいもの）で並べ替えます

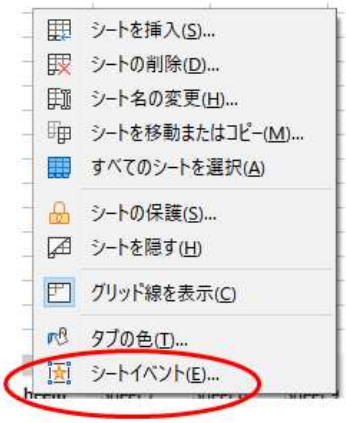
```
Sub ExampleSort()
        Dim oSortDesc(1) As New com.sun.star.beans.PropertyValue
        Dim oSortFields(1) As New com.sun.star.util.SortField
        oSheet = ThisComponent.CurrentController.ActiveSheet
        oRange = oSheet.getCellRangeByName("A1:D47")

        '並べ替えの第１キーをＡ列にして昇順で並べ替えを行う
        oSortFields(0).Field  = 0       '0:A列  1:B列  …
        oSortFields(0).SortAscending  = True   'True 昇順  False 降順
        '並べ替えの第２キーをＢ列にして昇順で並べ替えを行う
        oSortFields(1).Field  = 1
        oSortFields(1).SortAscending  = True

        '並べ替えキーの設定
        oSortDesc(0).Name = "SortFields"
```

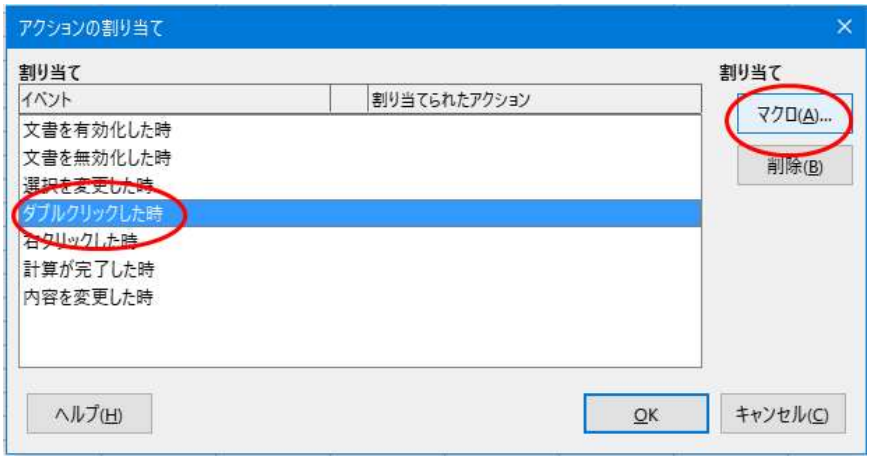
```
        oSortDesc(0).Value = oSortFields()

        '見出しの有無
        oSortDesc(1).Name = "ContainsHeader"
        oSortDesc(1).Value = False

        '並べ替えの実行
        oRange.sort(oSortDesc())
End Sub
```

第３章　プログラミング（Sheet 編）

1. 前準備・・・新しいモジュールを作成

シート名を右クリック・・・シートイベントをクリック

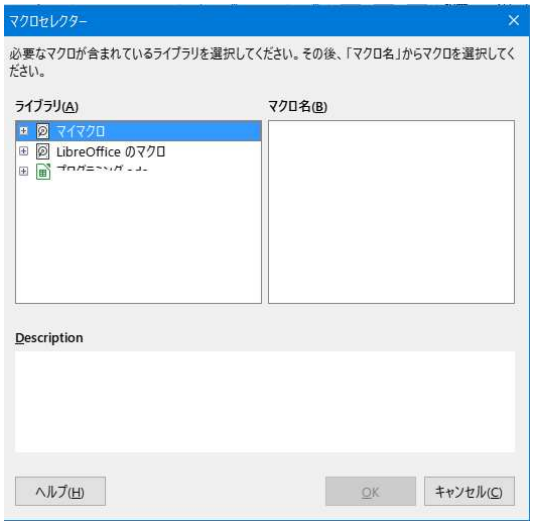


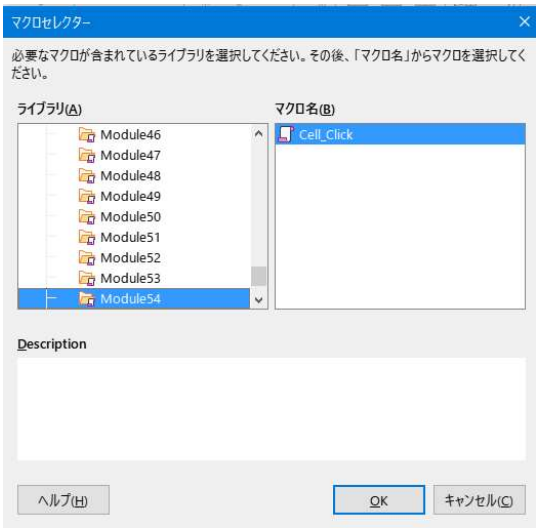


アクションの割り当てのダイアログボックスからイベントを選択（ダブルクリックした時をクリック）・・・マクロボタンをクリック

マクロセレクターのダイアログボックスが表示・・・ライブラリから Calc のファイル名をクリック・・・ライブラリに作成されたモジュールが表示・・・新しく作成したモジュールをクリック・・・OK

アクションの割り当てのダイアログボックスに戻る・・・OK をクリック





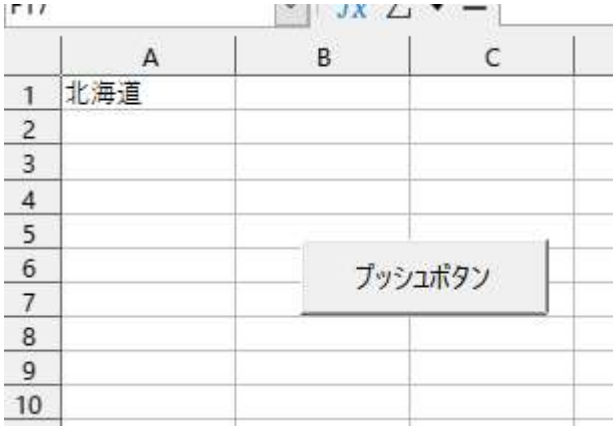
M54. シートのセルをクリックしてイベントを発生させ、メッセージボックスでセルの位置を表示。　Calk の列と行は 0 から始まるので　oRow、oColumn に各々 1 を加えて表示。

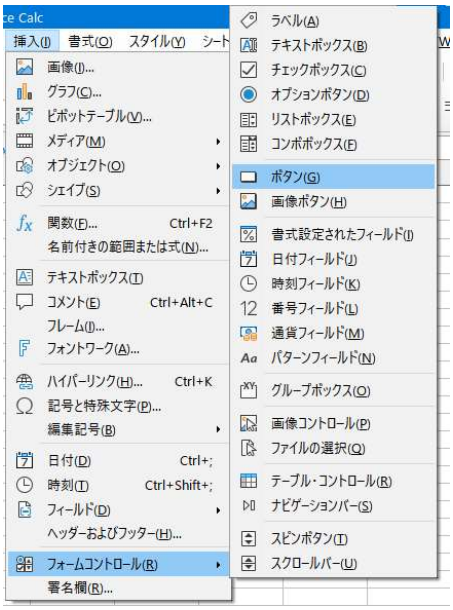
```
'シートのセルをクリック
Sub Cell_Click
        oCell = ThisComponent.CurrentSelection
        oRow = oCell.CellAddress.Row
        oColumn = oCell.CellAddress.Column
        MsgBox(oColumn+1 & "列目" & oRow+1 & "行目")
End Sub
```

M55. プッシュボタンを押して A1 の北海道を B1 にコピー

```
Sub Push_Button
        oSheet = ThisComponent.CurrentController.ActiveSheet
        oCell1 = oSheet.getCellRangeByName("A1")
        oCell2 = oSheet.getCellRangeByName("B1")
        oCell2.Formula = oCell1.Formula
End Sub
```

メニューバーの挿入・・・フォームコントロール・・・ボタンをクリック

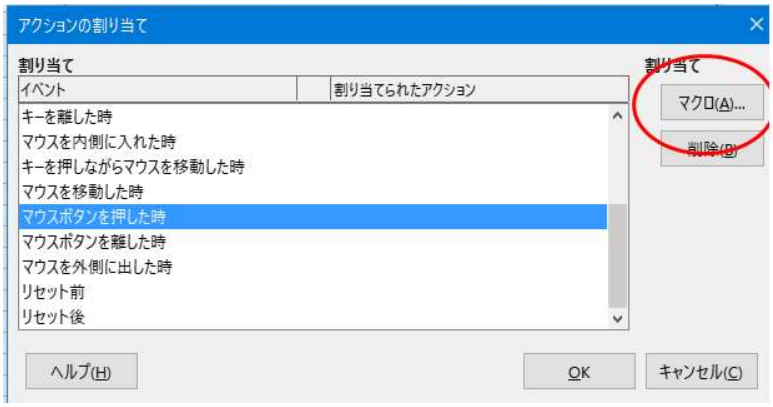




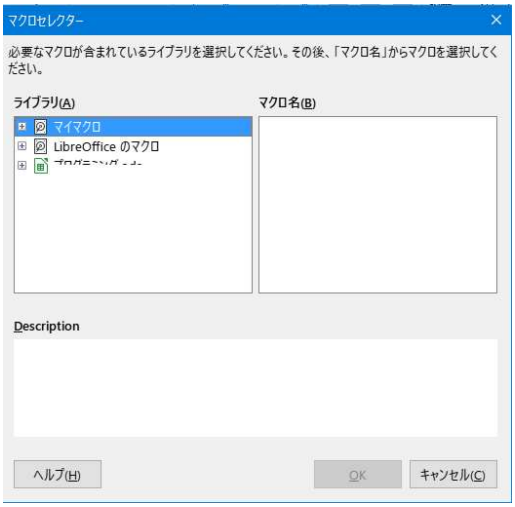
シートの上をドラッグ・・・プッシュボタンが表示
プッシュボタンを右クリック・・・コントロールのプロパティをクリック

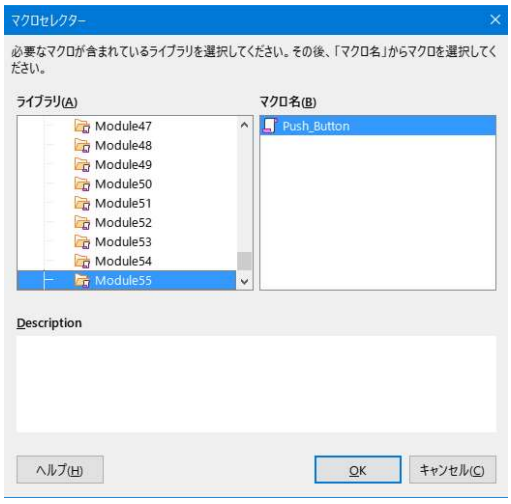
・・・属性プッシュボタンのダイアログボックス・・・イベント・・・マウスボタンを押したときの右側（…）をクリック

アクションの割り当てのダイアログボックスからイベントを選択（マウスボタンを押した時を確認）・・・マクロボタンをクリック



マクロセレクターのダイアログボックスが表示・・・ライブラリからCalcのファイル名をクリック・・・ライブラリに作成されたモジュールが表示・・・新しく作成したモジュールをクリック・・・OK

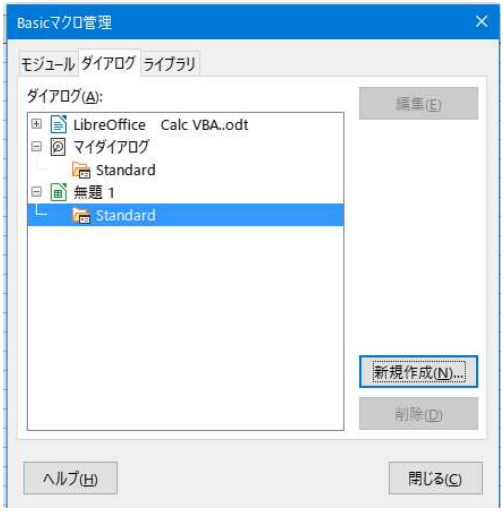


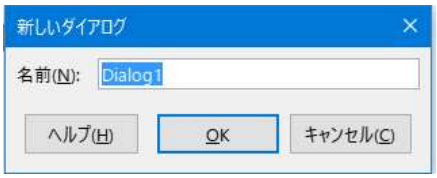


アクションの割り当てのダイアログボックスに戻り・・・ＯＫ
属性のプッシュボタンのダイアログボックスを終了
ツール・・・フォーム・・・デザインモードをＯＦＦ

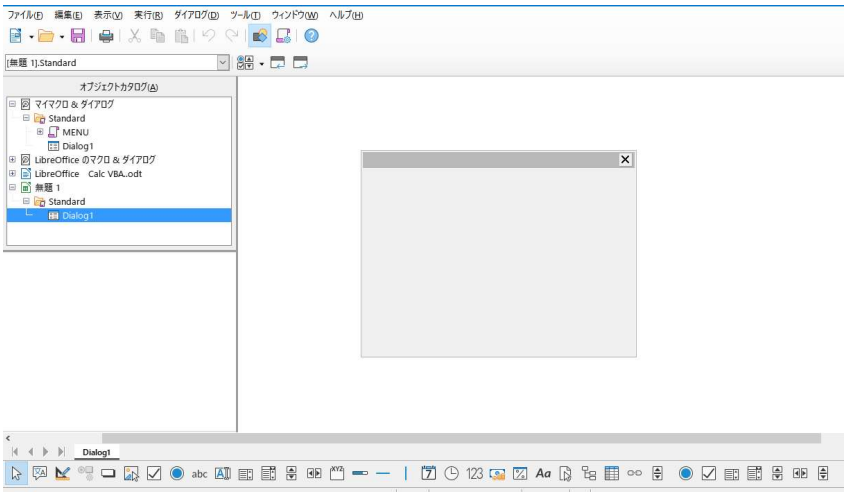
右のようにツールボックスのコントロール Textbox、CommandButton を挿入

1. LibreOffice マクロ Editor を起動する（ツール・・・マクロ・・・ダイアログの管理・・・無題１（マクロと同様 名前を付けて保存すると保存した名前になります）・・・Standard・・・新規作成をクリック
   新しいダイアログ「Dialog1」が表示され・・・OK

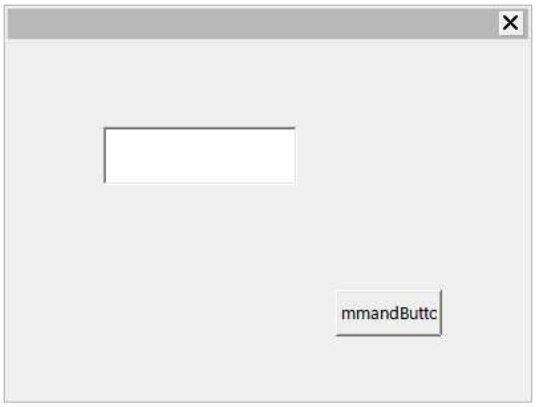




Standard の下に Dialog1 が表示されます。
上の無題１または、Standard をクリック（右上の編集が黒くなります）・・・編集をクリック
ダイアログエディタが表示（左側にオブジェクトカタログ、右側にダイアログが表示）、下にツールボックス（ダイアログ作成の部品が並んでします）
ダイアログに下のツールボックスから選択した TextBox と CommandButton を配置
ダイアログに下のツールボックスから選択した TextBox と CommandButton を配置
左側のオブジェクトカタログの無題１の下の Standard をクリック
右側の IDE Editor にプログラムを書いていきます。



M56.　Textbox を使ったダイアログボックス




```
'TextBoxの内容をセルに入力
Dim Dlg As Object
Dim Text1 As Object

Sub Dialog
        DialogLibraries.LoadLibrary("Standard")
        Dlg=CreateUnoDialog(DialogLibraries.Standard.Dialog1)
        Text1 = Dlg.GetControl("TextField1")
        Dlg.Execute()
End Sub

Sub Button_Click
        Sheet1 = ThisComponent.Sheets(0)
        Cells1 = Sheet1.GetCellByPosition(0,1)
        Cells1.HoriJustify = com.sun.star.table.CellHoriJustify.CENTER
        Cells1.CharColor = RGB(0,255,0)
        Cells1.CharHeight = 20
        Cells1.Formula = Text1.Text
        Dlg.EndExecute()
End Sub
```

＊　Dimにより変数Dlg1、変数Text1をDialog1の一部として設定
プロシージャMainにダイアログを開くためのプログラムを書きます。
プロシージャButton_clickにはTextboxに任意の数字を入力。　CommandButtonをクリックするとセル（“A2”）に入力した数字や文字が表示されるプログラムを書きます。

＊　実行はエディタのプロジェクトカタログで作成したModuleを選択・・・上のメニューバーの実行をクリック・・・実行を選択・・・Basicマクロのダイアログ、またはSheetの上のツール・・・マクロ・・・マクロを実行・・・マクロセレクターのダイアログ・・・Calk名・・・Standardをクリック・・・右側のMainをクリック

M57.　Label を使ったダイアログボックス

```
'Label
Dim Dlg2 As Object
Dim oLabel1 As Object

Sub Label
        DialogLibraries.LoadLibrary("Standard")
        Dlg2=CreateUnoDialog(DialogLibraries.Standard.Dialog2)
       oLabel1 = Dlg2.GetControl("Label1")
        Dlg2.Execute()
End Sub

Sub Button_Click
        oLabel1.SetText("プログラミング　テスト")
End Sub
```

M58.　Listbox を使ったダイアログボックス

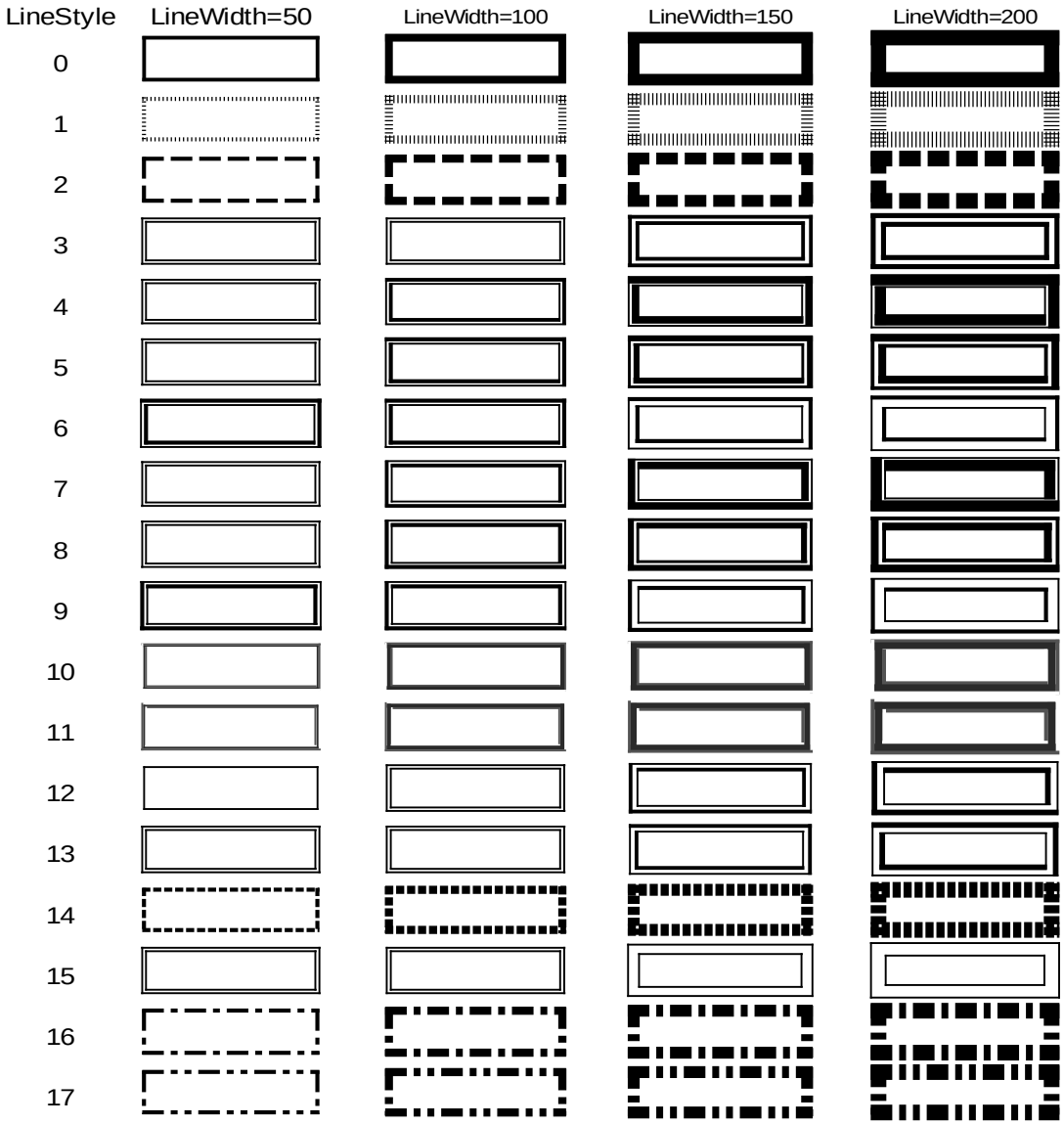
```
REM  *****  BASIC  *****
'ListBox
Dim Dlg3 As Object
Dim oListBox1 As Object
Dim SelItems() As String
Dim SelMessage As String

Sub ListBox
        DialogLibraries.LoadLibrary("Standard")
        Dlg3=CreateUnoDialog(DialogLibraries.Standard.Dialog3)
        oListBox1 = Dlg3.GetControl("ListBox1")
        With oListBox1
          .AddItem("北海道",0)
          .AddItem("青森県",1)
          .AddItem("秋田県",3)
          .AddItem("岩手県",4)
          .AddItem("新潟県",5)
          .AddItem("宮城県",6)
          .AddItem("福島県",7)
        End With
        Dlg3.Execute()
End Sub

Sub Button_Click
        oSheet1 = ThisComponent.Sheets(0)
        Cell11 = oSheet1.getCellRangeByName("A1")
        Cell12 = oSheet1.getCellRangeByName("B1")
        Cell21 = oSheet1.getCellRangeByName("A2")
        Cell22 = oSheet1.getCellRangeByName("B2")
        Cell31 = oSheet1.getCellRangeByName("A3")
        Cell32 = oSheet1.getCellRangeByName("B3")
        Cell41 = oSheet1.getCellRangeByName("A4")
        Cell42 = oSheet1.getCellRangeByName("B4")

        Cell11.String = "ItemCount"
        Cell12.Value = olistBox1.ItemCount
        Cell21.String = "SelectedItem"
        Cell22.String = olistBox1.SelectedItem

        SelItems = oListBox1.SelectedItems
```

```
        ListUBound = UBound(SelItems)
        for UIndex = 0 To ListUBound
                SelMessage = SelMessage & SelItems(UIndex)
                if UIndex<ListUBound then SelMessage = SelMessage & Chr$(13)
        Next

        Cell31.GetRows().Height = 450*(ListUBound+1)
        Cell31.VertJustify = com.sun.star.table.CellVertJustify.CENTER
        Cell32.VertJustify = com.sun.star.table.CellVertJustify.CENTER
        Cell31.String = "SelectedItems"
        Cell32.String = SelMessage
        Cell41.String = "SelectedItemPos"
        Cell42.Value = oListbox1.SelectedItemPos
        oListbox1.RemoveItems(1,3)
End Sub
```

・A1 は ItemCount（Item 数）
・A2 は選択した Item の先頭の項目
・A3 は選択した Item（セルの高さを選択された Item に 450（4.5mm）を掛けた数字にし、セルの縦位置を中央揃えにしてあります。
・A4 は選択した Item の先頭の項目のポジション Index 番号（0 から始まる）
・ListBox の Item 削除
RemoveItems(Index 番号,個数)

	A	B	C
1	ItemCount	7	
2	SelectedItem	秋田県	
3	SelectedItems	秋田県 新潟県	
4	SelectedItemPos	2	
5			
6			
7			

参照　罫線の種類


LineStyle
LineWidth=50
LineWidth=100
LineWidth=150
LineWidth=200
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17