OpenOBfuscatOr

L'obfuscateur des macros LibreOffice

Fichier LibreOffice Calc pour l'obfuscation des macros LibreOffice Basic

Obfuscation : techniques mises en œuvre pour transformer le code source afin de le rendre illisible pour un être humain tout en le conservant entièrement opérationnel.

OpenObfuscatOr

L'obfuscateur des macros LibreOffice

Fichier LibreOffice Calc pour l'obfuscation des macros LibreOffice Basic

1- Définition de l'obfuscation

En programmation informatique, on appelle obfuscation les techniques mises en œuvre pour transformer le code source afin de le rendre illisible pour un être humain après une forme de compilation, tout en le conservant entièrement opérationnel. L'objectif est de protéger les droits de propriété sur une application informatique en empêchant la reconstitution ou la transformation d'un code source compréhensible à partir du code d'origine.

Synonyme d'obfuscation : brouillage

OpenObfuscatOr pour LibreOffice Calc

	Le dossier de travail est le dossier où se trouve le classeur OpenObfuscatOr
Dossier de travail :	C:/A_Joe_Dev/A-DEV-MATH-BUT/LibreOffice-Org/OpenObfuscatOr/OpenObfuscatOr/Version_2.0/
Nom du fichier .bas à crypter :	demo_encoding.bas
	Le fichier .bas à crypter doit se trouver dans le même dossier que celui où se trouve le classeur OpenObfuscatOr.
Lancer OpenObfuscatOr	Opération terminée avec succès.
	Ouvrir Fichier Initial / Ouvrir Fichier Crypté / Ouvrir la Liste des Variables
Fichier Initial :	demo_encoding.bas
Fichier Crypté :	demo_encoding_Crypt.bas
Liste des Variables :	demo_encoding_Joob_Variables.csv

Mode d'emploi du classeur OpenObfuscatOr pour LibreOffice Calc

1/ Exporter le code Basic du module dans un fichier .bas en actionnant le menu : "Fichier/Exporter le Basic" de l'Editeur Basic de LibreOffice.
2/ Placer ce fichier .bas à obfusquer (crypter) dans le même dossier que celui où se trouve le classeur OpenObfuscatOr sur le disque dur.
3/ Saisir le nom de ce fichier .bas dans la cellule G6.

Cliquer sur le bouton "Lancer OpenObfuscatOr" pour obfusquer (crypter) le fichier .bas
On obtient ainsi un fichier .bas don't le mot "Crypt" a été ajouté au nom initial.

Remarques

1/ Une copie de sauvegarde du fichier .bas est automatiquement créé dans le même dossier.
2/ Un fichier d'extension .csv contient la liste des noms des variables des macros ainsi que les noms obfusqués (cryptés) correspondants.

Ouvrir le fichier obfusqué (crypté) qui se trouve dans le dossier en cours à l'aide d'un éditeur de texte (notepad.exe) et copier l'ensemble du code Basic obfusqué (crypté).
Coller le code Basic obfusqué (crypté) dans le module concerné en utilisant l'Editeur Basic de LibreOffice.

Pour information, la liste des noms des variables du code Basic et des noms obfusqués (cryptés) est disponible dans un fichier .csv

2- Description du classeur LibreOffice Calc

OpenObfuscatOr est un classeur LibreOffice Calc permettant de protéger le code source Basic d'une macro LibreOffice en le rendant quasiment illisible. Pour ce faire, cet outil effectue un cryptage du nom des variables, transforme les chaînes de caractères et supprime les commentaires ainsi que l'indentation des lignes de code. Toutes ces opérations rendent le code source de la macro quasiment illisible.

De plus OpenObfuscatOr sauvegarde le code source initial dans un fichier au format texte (.bas) ainsi que le code source après obfuscation dans un fichier séparé (.bas). Il dresse également la liste des variables cryptées et des chaînes de caractères transformées (fichier d'extension .csv).

Ainsi toutes les macros du fichier basic (.bas) sont archivées dans des fichiers séparés (macros initiales et macros traitées). Cet archivage permet d'avoir une sauvegarde des macros d'un projet donné.

Exemple

Un exemple d'une procédure (Sub) et de deux fonctions (Function) LibreOffice Basic avant et après le traitement pas OpenObfuscatOr.


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

'----------------
Sub Demo_Video ()
'----------------
'Ceci est une procédure de démonstration
Dim RepMessage As Boolean
Dim ValMessage As Integer

 RepMessage = Demo_Message
 If RepMessage = True then
     ValMessage = 1
     MsgBox "Un message a été affiché, 64, "Résultats"
     Ecrire_Valeur ("Feuille1", "D5", ValMessage )
 Else
     ValMessage = 0
     MsgBox "Aucun message n'a été affiché, 64, "Résultats"
     Ecrire_Valeur ("Feuille1", "D5", ValMessage )
 End If
'------
End Sub

'---------------------------------
Function Demo_Message () As Boolean
'---------------------------------
'-- Ceci est une macro de démonstration.
'-- Déclaration des variables
Dim Message1 As String
Dim Message2 As String
Dim MessageFinal As String
Dim Titre As String
Dim MesON As Boolean
Dim Kp As Integer
Dim Rep As Integer

'-- Fin de déclaration des variables
```

```
 Message1 = "OpenObfuscatOr" & chr(13) & chr(10)
 Message2 = "Ceci est une démonstration du classeur OpenObfuscatOr." & chr(13) &
chr(10)
'-- chr(13) : Carriage return et chr(10) : Line feed
 Kp = Len( Message1 )

    If Kp = 16 Then
'-- On met la variable MesON à True
        MesON = True
    Else
'-- On met la variable MesON à False
        MesON = False
    End If

'-- Contenu du message :
 Titre = "OpenObfuscatOr : Vidéo de démonstration"
 MessageFinal = Message1
 MessageFinal = MessageFinal & Message2
 MessageFinal = MessageFinal & "Téléchargez la version originale sur www.joseouin.fr" &
chr(13) & chr(10)
 MessageFinal = MessageFinal & "Auteur : JoséOUIN - Site Internet : www.joseouin.fr"

'On teste la valeur de la variable MesON
    If  MesON = True Then
        Rep = MsgBox( MessageFinal , 64 , Titre )
    End If
'------
 Demo_Message = MesON
'-----------
End Function

'-------------------------------------------------------
Function Ecrire_Valeur (ZNomFeuille,ZNomCellule,ZValeur)
'-------------------------------------------------------
Dim OOODoc As Object
Dim OOOFeuille As Object
Dim OOOCellule As Object

'Définition d'un objet Classeur
 OOODoc = ThisComponent
'Définition d'un objet Feuille de ce classeur
 OOOFeuille = OOODoc.getSheets.getByName(ZNomFeuille)

'Identification par l'adresse de la cellule
 OOOCellule = OOOFeuille.getCellRangeByName(ZNomCellule)
'Affectation de la valeur dans la cellule
 OOOCellule.Value = ZValeur
'-----------
End Function
```

Remarque :
Le code source ci-dessus comporte des commentaires, des lignes vierges ainsi qu'une indentation des lignes de codes. Tout ceci participe à la lisibilité du code source et c'est pour cette raison qu'OpenObfuscatOr retire tous ces éléments.

Résultat : La procédure et les deux fonctions LibreOffice Basic ont été cryptées par OpenObfuscatOr :

```
Sub Demo_Video ()
Dim JOOBetjKaoqPNcV8RrWPTaHdFqb As Boolean
Dim JOOBetjKaoALJcV8RrWPTaHdFqb As Integer

JOOBetjKaoqPNcV8RrWPTaHdFqb = JOOBetjKaocPKQPpDP9LUKTaHdFqb
If JOOBetjKaoqPNcV8RrWPTaHdFqb = True then
     JOOBetjKaoALJcV8RrWPTaHdFqb = 1
     MsgBox Chr(85) & Chr(110) & Chr(32) & Chr(109) & Chr(101) & Chr(115) & Chr(115) &
Chr(97) & Chr(103) & Chr(101) & Chr(32) & Chr(97) & Chr(32) & Chr(233) & Chr(116) &
Chr(233) & Chr(32) & Chr(97) & Chr(102) & Chr(102) & Chr(105) & Chr(99) & Chr(104) &
Chr(233) & Chr(46), 64, Chr(82) & Chr(233) & Chr(115) & Chr(117) & Chr(108) & Chr(116)
& Chr(97) & Chr(116) & Chr(115)
     JOOBetjKaodNPK9TrgQWS13TaHdFqb (Chr(70) & Chr(101) & Chr(117) & Chr(105) &
Chr(108) & Chr(108) & Chr(101) & Chr(49), Chr(68) & Chr(53),
JOOBetjKaoALJcV8RrWPTaHdFqb )
Else
     JOOBetjKaoALJcV8RrWPTaHdFqb = 0
     MsgBox Chr(65) & Chr(117) & Chr(99) & Chr(117) & Chr(110) & Chr(32) & Chr(109) &
Chr(101) & Chr(115) & Chr(115) & Chr(97) & Chr(103) & Chr(101) & Chr(32) & Chr(110) &
Chr(39) & Chr(97) & Chr(32) & Chr(233) & Chr(116) & Chr(233) & Chr(32) & Chr(97) &
Chr(102) & Chr(102) & Chr(105) & Chr(99) & Chr(104) & Chr(233) & Chr(46), 64, Chr(82) &
Chr(233) & Chr(115) & Chr(117) & Chr(108) & Chr(116) & Chr(97) & Chr(116) & Chr(115)
     JOOBetjKaodNPK9TrgQWS13TaHdFqb (Chr(70) & Chr(101) & Chr(117) & Chr(105) &
Chr(108) & Chr(108) & Chr(101) & Chr(49), Chr(68) & Chr(53),
JOOBetjKaoALJcV8RrWPTaHdFqb )
End If
End Sub

Function JOOBetjKaocPKQPpDP9LUKTaHdFqb () As Boolean
Dim JOOBetjKaolPQURVDETaHdFqb As String
Dim JOOBetjKaolPQURVDFTaHdFqb As String
Dim JOOBetjKaolPQURVDZYYORTaHdFqb As String
Dim JOOBetjKaosTRTVTaHdFqb As String
Dim JOOBetjKaolPQesTaHdFqb As Boolean
Dim JOOBetjKaoj1TaHdFqb As Integer
Dim JOOBetjKaoqPNTaHdFqb As Integer


JOOBetjKaolPQURVDETaHdFqb = Chr(79) & Chr(112) & Chr(101) & Chr(110) & Chr(79) &
Chr(98) & Chr(102) & Chr(117) & Chr(115) & Chr(99) & Chr(97) & Chr(116) & Chr(79) &
Chr(114) & chr(13) & chr(10)
JOOBetjKaolPQURVDFTaHdFqb = Chr(67) & Chr(101) & Chr(99) & Chr(105) & Chr(32) &
Chr(101) & Chr(115) & Chr(116) & Chr(32) & Chr(117) & Chr(110) & Chr(101) & Chr(32) &
Chr(100) & Chr(233) & Chr(109) & Chr(111) & Chr(110) & Chr(115) & Chr(116) & Chr(114) &
Chr(97) & Chr(116) & Chr(105) & Chr(111) & Chr(110) & Chr(32) & Chr(100) & Chr(117) &
Chr(32) & Chr(99) & Chr(108) & Chr(97) & Chr(115) & Chr(115) & Chr(101) & Chr(117) &
Chr(114) & Chr(32) & Chr(79) & Chr(112) & Chr(101) & Chr(110) & Chr(79) & Chr(98) &
Chr(102) & Chr(117) & Chr(115) & Chr(99) & Chr(97) & Chr(116) & Chr(79) & Chr(114) &
Chr(46) & chr(13) & chr(10)
JOOBetjKaoj1TaHdFqb = Len( JOOBetjKaolPQURVDETaHdFqb )

If JOOBetjKaoj1TaHdFqb = 16 Then
JOOBetjKaolPQesTaHdFqb = True
Else
JOOBetjKaolPQesTaHdFqb = False
End If
```

```
JOOBetjKaosTRTVTaHdFqb = Chr(79) & Chr(112) & Chr(101) & Chr(110) & Chr(79) & Chr(98) &
Chr(102) & Chr(117) & Chr(115) & Chr(99) & Chr(97) & Chr(116) & Chr(79) & Chr(114) &
Chr(32) & Chr(58) & Chr(32) & Chr(86) & Chr(105) & Chr(100) & Chr(233) & Chr(111) &
Chr(32) & Chr(100) & Chr(101) & Chr(32) & Chr(100) & Chr(233) & Chr(109) & Chr(111) &
Chr(110) & Chr(115) & Chr(116) & Chr(114) & Chr(97) & Chr(116) & Chr(105) & Chr(111) &
Chr(110)
JOOBetjKaolPQURVDZYYORTaHdFqb = JOOBetjKaolPQURVDETaHdFqb
JOOBetjKaolPQURVDZYYORTaHdFqb = JOOBetjKaolPQURVDZYYORTaHdFqb &
JOOBetjKaolPQURVDFTaHdFqb
JOOBetjKaolPQURVDZYYORTaHdFqb = JOOBetjKaolPQURVDZYYORTaHdFqb & Chr(84) & Chr(233) &
Chr(108) & Chr(233) & Chr(99) & Chr(104) & Chr(97) & Chr(114) & Chr(103) & Chr(101) &
Chr(122) & Chr(32) & Chr(108) & Chr(97) & Chr(32) & Chr(118) & Chr(101) & Chr(114) &
Chr(115) & Chr(105) & Chr(111) & Chr(110) & Chr(32) & Chr(111) & Chr(114) & Chr(105) &
Chr(103) & Chr(105) & Chr(110) & Chr(97) & Chr(108) & Chr(101) & Chr(32) & Chr(115) &
Chr(117) & Chr(114) & Chr(32) & Chr(119) & Chr(119) & Chr(119) & Chr(46) & Chr(106) &
Chr(111) & Chr(115) & Chr(101) & Chr(111) & Chr(117) & Chr(105) & Chr(110) & Chr(46) &
Chr(102) & Chr(114) & chr(13) & chr(10)
JOOBetjKaolPQURVDZYYORTaHdFqb = JOOBetjKaolPQURVDZYYORTaHdFqb & Chr(65) & Chr(117) &
Chr(116) & Chr(101) & Chr(117) & Chr(114) & Chr(32) & Chr(58) & Chr(32) & Chr(74) &
Chr(111) & Chr(115) & Chr(233) & Chr(32) & Chr(79) & Chr(85) & Chr(73) & Chr(78) &
Chr(32) & Chr(45) & Chr(32) & Chr(83) & Chr(105) & Chr(116) & Chr(101) & Chr(32) &
Chr(73) & Chr(110) & Chr(116) & Chr(101) & Chr(114) & Chr(110) & Chr(101) & Chr(116) &
Chr(32) & Chr(58) & Chr(32) & Chr(119) & Chr(119) & Chr(119) & Chr(46) & Chr(106) &
Chr(111) & Chr(115) & Chr(101) & Chr(111) & Chr(117) & Chr(105) & Chr(110) & Chr(46) &
Chr(102) & Chr(114)

If  JOOBetjKaolPQesTaHdFqb = True Then
JOOBetjKaoqPNTaHdFqb = MsgBox( JOOBetjKaolPQURVDZYYORTaHdFqb , 64 ,
JOOBetjKaosTRTVTaHdFqb )
End If
JOOBetjKaocPKQPpDP9LUKTaHdFqb = JOOBetjKaolPQesTaHdFqb
End Function


Function JOOBetjKaodNPK9TrgQWS13TaHdFqb (ZNomFeuille,ZNomCellule,ZValeur)
Dim JOOBetjKaonna36RTaHdFqb As Object
Dim JOOBetjKaonna5VaHI2PTaHdFqb As Object
Dim JOOBetjKaonna2V1KR2PTaHdFqb As Object

JOOBetjKaonna36RTaHdFqb = ThisComponent
JOOBetjKaonna5VaHI2PTaHdFqb = JOOBetjKaonna36RTaHdFqb.getSheets.getByName(ZNomFeuille)

JOOBetjKaonna2V1KR2PTaHdFqb =
JOOBetjKaonna5VaHI2PTaHdFqb.getCellRangeByName(ZNomCellule)
JOOBetjKaonna2V1KR2PTaHdFqb.Value = ZValeur
End Function
```

Remarques :

Les noms des fonctions et des variables ont été cryptées. Le nom de la procédure (Sub) n'a pas été modifié : ainsi l'affectation d'une procédure (Sub) à un bouton reste inchangée.

Les chaînes de caractères ont été remplacées par le code Ascii de chacune des lettres composant ces chaînes.

3- Propriétés de OpenObfuscatOr

Cet outil effectue les actions suivantes :

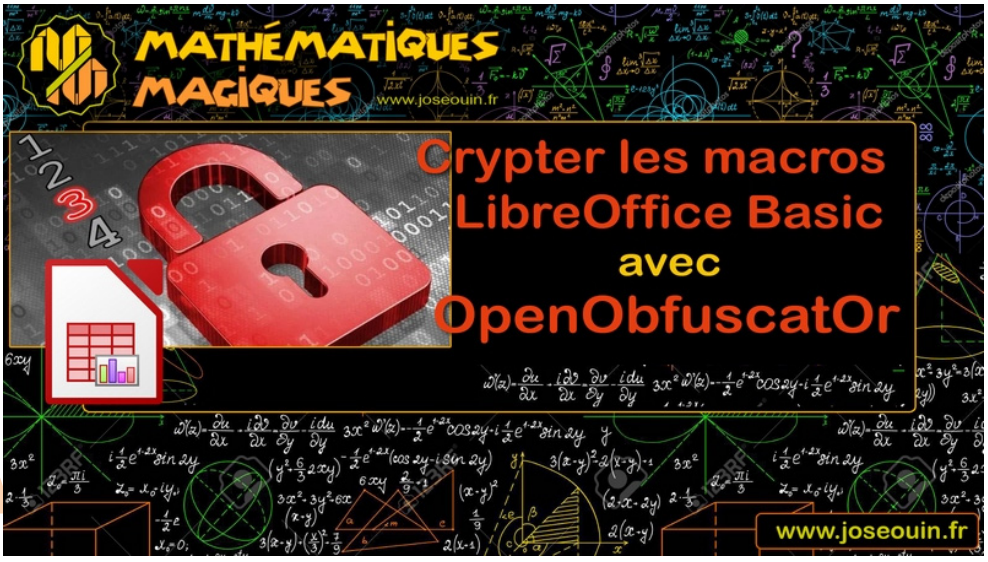
1. Cryptage du nom des variables ;
2. Transformation des chaînes de caractères ;
3. Suppression de l'indentation des lignes de code ;
4. Sauvegarde du code source initial dans un fichier au format texte (.bas) ;
5. Sauvegarde du code source après transformation dans un fichier séparé (.bas) ;
6. Listage du nom des variables initiales et transformées (fichier .csv).

Vidéo de présentation :

Une vidéo de présentation et de mini-formation à l'utilisation de ce classeur LibreOffice Calc est disponible sur Youtube :

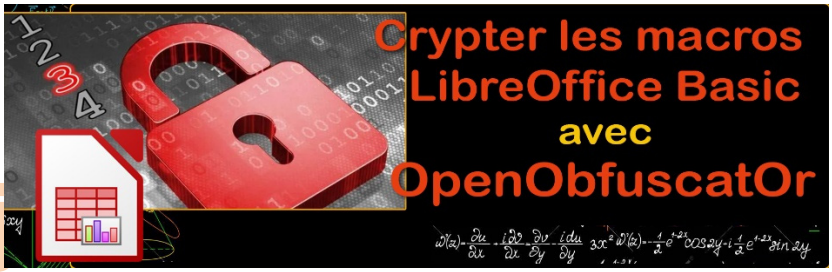
Lien N°1 : http://bit.ly/openobfuscator2

Lien N°2 : https://www.youtube.com/watch?v=F_Ia4QZkSy8



Article sur le site www.joseouin.fr

Un article complet se trouve sur le site Internet http://www.joseouin.fr



Lien : http://joseouin.fr/logiciels/logiciel-openobfuscator

4- Conseils et remarques sur l'utilisation de OpenObfuscatOr

Limiter la longueur des chaînes de caractères au niveau de l'éditeur Basic

OpenObfuscator transforme les chaînes de caractères à l'aide de la fonction Chr() et des codes Ascii des caractères de la chaîne. Par exemple, le mot **VBA** sera remplacé par **Chr(86) & Chr(66) & Chr(65).**

Ces **3** caractères génèreront : 3 x **7** caractères + 4 espaces + 2 **&** = **27** caractères.

Dans l'éditeur Basic, une ligne de code ne peut pas dépasser 1024 caractères (à moins d'utiliser l'underscore « _ » touches AltGr + 8). Il ne faut donc pas dépasser 1024/**9**, soit 110 caractères environ par ligne de code.
Mon conseil : limiter les lignes de codes à 90 caractères.

Attention ! La ligne de code suivante (suite à l'obfuscation) dépassera les 1024 caractères si vous saisissez ceci :

Oh non !

```
Message = "OpenObfuscatOr : Ceci est un message qui assez long qu'il faudra couper en deux au niveau de votre code car la transformation en Chr() augmente le nombre de caractères de la ligne de code."
```

Solutions

Solution 1 :
Couper votre ligne en 2, voire plus, de cette manière :

Ah oui !

```
Message = "JO-Obfuscator : Ceci est un message qui assez long qu'il faudra couper en deux au niveau"
Message = Message & " de votre code car la transformation en Chr() augmente le nombre de caractères de la ligne de code."
```

Solution 2 :
Couper votre ligne en 2, voire plus, en utilisant plusieurs variables :

Ah oui !

```
Ligne1 = "JO-Obfuscator : Ceci est un message qui assez long qu'il faudra couper en deux au niveau"
Ligne2 = " de votre code car la transformation en Chr() augmente le nombre de caractères de la ligne de code."
Message = Ligne1 & Ligne2
```

Solution 3 :
Couper votre ligne en 2, voire plus, en utilisant l'underscore (« AltGr + 8 » : tiret bas de la touche « 8 » : "_") :

Ah oui !

```
Message = "JO-Obfuscator : Ceci est un message qui assez long qu'il faudra couper en deux au niveau" & _
" de votre code car la transformation en Chr() augmente le nombre de caractères de la ligne de code."
```

5- Les règles à respecter pour l'utilisation du classeur OpenObfuscatOr

Dans toute la suite, on symbolise le caractère « espace » par un carré de couleur magenta ■

5-1 : Déclarer les variables et les chaînes de caractères

Il faut déclarer vos variables si vous souhaitez qu'elles soient cryptées. Autrement dit toutes les variables non déclarées par le mot clé **Dim** ne seront pas obfusquées (cryptées) par OpenObfuscatOr.

Oh non ! (rappel : le carré ■ symbolise un espace)

```
Message■= "Bonjour à tous"
Longueur■ = 7
```

Solution :
Déclarez les variables, sinon elles ne seront pas cryptées par OpenObfuscatOr :

Ah oui !

```
Dim Message As String
Dim Longueur As Integer
Message■= "Bonjour à tous"
Longueur■= 7
```

IMPORTANT !

Si une variable déclarée par **Dim** porte le même nom qu'une variable constituant un paramètre d'une fonction, OpenObfuscatOr cryptera les deux de la même manière. Ceci occasionnera des dysfonctionnements au niveau de la fonction.

Exemple :

```
Function■Demo1■()
Dim X As Double
.....
.....
End Function

Function■Demo2■(X) As Double
■Demo2■=■X■^2
End Function
```

On obtiendra alors la variable X cryptée dans fonction Demo2 (ce qui n'est pas souhaité par le programmeur) :

```
Function■Demo2■(X) As Double
■Demo2■=■JOOBetjKaobJVUTXTaHdFqb■^2
End Function
```

La fonction Demo2 ne donnera plus les bons résultats.

Solution :
Il faut adopter une règle d'écriture pour les variables qui sont des paramètres de fonctions. On peut par exemple décider que les noms de ces variables commenceront tous par « Z ». Ainsi il n'y aura plus de doublons entre les noms des variables déclarées par Dim et ceux constituant les paramètres des fonctions.

```
Function■Demo2■(ZX) As Double
■Demo2■=■ZX■^2
End Function
```

5-2 : Ecrire les Instructions Dim sur le bord gauche de l'éditeur Basic (sans espaces)

Dim doit se trouver tout à fait à gauche de la page de l'éditeur Basic. Il ne doit y avoir aucun espace avant ce mot clé. On symbolise le caractère « espace » par un carré de couleur magenta ■.

Exemple :

Oh non !

```
Sub Demo_Fichier_Output2 ()
■■Dim OOONumF As Integer
■Dim NomFichier As String
■■Dim Tampon As String

 NomFichier = "C:\temp\demo.txt"
 Tampon = "Bonjour"
' Détermination d'un Numéro de fichier libre
 OOONumF = FreeFile
' Ouverture du fichier en Output
Open NomFichier for Output As #OOONumF
Print #OOONumF , Tampon
'....
' Fermeture du fichier
Close OOONumF

End Sub
```

Ah oui !

```
Sub Demo_Fichier_Output2 ()
Dim OOONumF As Integer
Dim NomFichier As String
Dim Tampon As String

 NomFichier = "C:\temp\demo.txt"
 Tampon = "Bonjour"
' Détermination d'un Numéro de fichier libre
 OOONumF = FreeFile
' Ouverture du fichier en Output
Open NomFichier for Output As #OOONumF
Print #OOONumF , Tampon
'....
' Fermeture du fichier
Close OOONumF

End Sub
```

5-3 : Insérer un espace après les noms des variables

Tous les noms des variables déclarées par **Dim** doivent avoir un espace ■**après** au niveau du code Basic.

Exemple :

Oh non !

```
Sub Demo_Fichier_Output2 ()
Dim OOONumF As Integer
Dim NomFichier As String
Dim Tampon As String

Dim■Tableau■(1 to 10, 1 to 10) As Integer

NomFichier= "C:\temp\demo.txt"
Tampon= "Bonjour"
' Détermination d'un Numéro de fichier libre

OOONumF■= FreeFile
' Ouverture du fichier en Output

Open■NomFichier■for Output as #OOONumF■

Print #OOONumF■,■Tampon■
'....
' Fermeture du fichier

Close■OOONumF■

End Sub
```

Ah oui !

```
Sub Demo_Fichier_Output2 ()
Dim OOONumF As Integer
Dim NomFichier As String
Dim Tampon As String

Dim■Tableau■(1 to 10, 1 to 10) As Integer

NomFichier■= "C:\temp\demo.txt"

Tampon■= "Bonjour"
' Détermination d'un Numéro de fichier libre

OOONumF■= FreeFile
' Ouverture du fichier en Output

Open■NomFichier■for Output As #OOONumF■

Print #OOONumF■,■Tampon■
'....
' Fermeture du fichier

Close■OOONumF■

End Sub
```


Remarque importante :

Un espace doit être placé entre le nom de la variable pour une matrice et la parenthèse ouvrante :

```
Dim■Tableau■(1 to 10, 1 to 10) As Integer
```

Autre exemple : il faut mettre un espace ■ après le nom des variables déclarées par Dim.

Oh non !

```
Dim angle_a As Double
Dim d_a As Double
Dim d_b As Double
Dim d_c As Double
Dim Tableau(1 to 10, 1 to 10) As Double

angle_a=arccos((d_b^2 +d_c^2 -d_a^2)/(2*d_b*d_c))
Tableau(1,1)=angle_a
```

Ah oui !

```
Dim angle_a As Double
Dim d_a As Double
Dim d_b As Double
Dim d_c As Double

Dim■Tableau■(1 to 10, 1 to 10) As Double

angle_a■=arccos■((d_b■^2 +d_c■^2 -d_a■^2)/(2*d_b■*d_c■))

Tableau■(1,1)■=angle_a■
```

Remarque :
On peut également insérer un espace avant et après chaque nom de variable ou de function.

```
angle_a■=■arccos■((■d_b■^2 +■d_c■^2 -■d_a■^2)/(2*■d_b■*■d_c■))

Tableau■(1,1)■= ■angle_a■
```

5-4 : Attribuer un nom particulier aux variables ne devant pas comporter d'espaces dans le code Basic crypté

Si un nom de variable ne doit pas avoir d'espace dans sa version cryptée, son nom doit commencer par "**OOO**". C'est le cas des variables "Object" ou des numéros de fichier. Dans ce cas, **il ne faut pas ajouter d'espace** dans le code basic initial.

Exemple :

Oh non !

```
Sub Demo_Fichier_Output2 ()
Dim OOONumF As Integer
Dim NomFichier As String
Dim Tampon As String

NomFichier = "C:\temp\demo.txt"
Tampon = "Bonjour"
' Détermination d'un Numéro de fichier libre
OOONumF = FreeFile
' Ouverture du fichier en Output
Open NomFichier for Output As #OOONumF
Print #OOONumF , Tampon
'....
' Fermeture du fichier
Close OOONumF

End Sub
```

Ah oui !

```
Sub Demo_Fichier_Output2 ()
Dim OOONumF As Integer
Dim NomFichier As String
Dim Tampon As String

NomFichier = "C:\temp\demo.txt"
Tampon = "Bonjour"
' Détermination d'un Numéro de fichier libre
OOONumF = FreeFile
' Ouverture du fichier en Output
Open NomFichier for Output As #OOONumF
Print #OOONumF , Tampon
'....
' Fermeture du fichier
Close OOONumF

End Sub
```

Autre exemple :
Les attributs des variables "objet" sont séparés par des points "." **sans espaces**. Pour que cette disposition soit respectée après l'obfuscation, il faut que les noms des variables commencent par "**OOO**".

```
Function Ecrire_Chaine (ZNomFeuille,ZNomCellule,ZChaine)
Dim OOODoc As Object
Dim OOOFeuille As Object
Dim OOOCellule As Object

' Definition d'un objet Classeur

OOODoc = ThisComponent
' Définition d'un objet Feuille de ce classeur

OOOFeuille = OOODoc.getSheets.getByName(ZNomFeuille)
' Identification par l'adresse de la cellule

OOOCellule = OOOFeuille.getCellRangeByName(ZNomCellule)
' Affectation de la valeur dans la cellule
OOOCellule.String = ZChaine
End Function
```

Par exemple, il ne doit pas y avoir d'espace à droite de `OOODoc` dans cette ligne : `OOODoc.getSheets.getByName`

Suite à l'obfuscation, on obtiendra :

```
Function JOOBetjKaodNPK9TrgQWS13NTaHdFqb (ZNomFeuille,ZNomCellule,ZChaine)

Dim JOOBetjKaonna36RTaHdFqb As Object
Dim JOOBetjKaonna5VaHI2PTaHdFqb As Object
Dim JOOBetjKaonna2V1KR2PTaHdFqb As Object

JOOBetjKaonna36RTaHdFqb = ThisComponent
JOOBetjKaonna5VaHI2PTaHdFqb = JOOBetjKaonna36RTaHdFqb.getSheets.getByName(ZNomFeuille)
JOOBetjKaonna2V1KR2PTaHdFqb = JOOBetjKaonna5VaHI2PTaHdFqb.getCellRangeByName(ZNomCellule)
JOOBetjKaonna2V1KR2PTaHdFqb.Value = ZChaine
```

5-5 : Insérer un espace ■ avant et après les noms des fonctions “Function” et des procedures “Sub”

Le nom des fonctions “Function” doit comporter un espace avant et après au niveau du code Basic.
Le nom des procedures “Sub” n’est pas crypté. Ceci permet d’affecter les procédures à des boutons sans devoir changer le nom de cette affectation suite au cryptage du code Basic.

Exemple :

Oh non !

```
Function■Ecrire_Fichier() As Boolean
'--
Code ici …
End Function
```

Ah oui !

```
Function■Ecrire_Fichier■() As Boolean
'--
Code ici …
End Function


Sub Main ()
'--
Code ici …
Rep■=■Ecrire_Fichier■
'--
Code ici …
End Sub
```

Remarque importante :

Il ne faut pas oublier d’ajouter un espace après le nom de la fonction : ■Rep■=■Ecrire_Fichier■

Autre exemple :

```
Function■Split_Chaine■(ZMaChaineSC As String)
'Fonction de découpage des caractères d'une chaine
Dim■i■As Integer
Dim■Tableau■(len(ZMaChaineSC))■As■String

For■i■=■1■to■len(ZMaChaineSC)
   Tableau■(■i■-1) = Mid(ZMaChaineSC,■i■,1)
next■i■

Split_Chaine■=■Tableau■()
End Function
```

Remarque importante :

Il ne faut pas oublier d’ajouter un espace après le **i** ici : **next■i■**

5-6 : Solution aux problèmes liées aux noms des variables et des mots clés du langage Basic

Les noms des variables peuvent parfois interférer avec les nom des mots clés du langage Basic.

Exemple :

La variable “p” sera cryptée par OpenObfuscatOr car elle a été déclarée par Dim.
Problème : le “p” du mot clé “**step**” sera remplacé par le nom crypté de “p” (car il y a un espace après le “p” de “step”).
Même chose pour la variable “ext” : le “ext” de “**next**” sera remplacé par le nom crypté de “ext” (car il y a un espace après le “ext” de “next”).

```
Dim i As Integer
Dim p As Integer
Dim ext As Integer

Dim Tableau (len(ZMaChaineSC)) As String

For i = 1 to len(ZMaChaineSC) step 2
   Tableau (i -1) = Mid(ZMaChaineSC, i ,1)
next i
```

Solution :

1/ Ne pas déclarer les variables “p” et “ext”. Mais dans ce cas elles ne seront pas cryptées, ce qui peut atténuer l’illisibilité du code source.
2/ Choisir une lettre réservée (par exemple W) comme première lettre pour toutes les variables du code source.

```
Dim Wi As Integer
Dim Wp As Integer
Dim Wext As Integer
Dim WTableau (len(ZMaChaineSC)) As String

For Wi = 1 to len(ZMaChaineSC) step 2
   WTableau (Wi -1) = Mid(ZMaChaineSC, Wi ,1)
next Wi
```

5-7 : Solution aux problèmes liées aux noms des variables

Il peut y avoir des problèmes liées aux noms des variables les uns vis-à-vis des autres.

Exemple :
La variable “Temp” sera cryptée par OpenObfuscatOr car elle a été déclarée par Dim.
Problème : le “Temp” du nom “MatTemp” sera remplacé par le nom crypté de “Temp” (car il y a un espace après le “Temp” de “MatTemp”).

```
Dim i As Integer
Dim Temp As Integer
Dim MatTemp As Integer
Dim Tableau (len(ZMaChaineSC)) As String

For i = 1 to len(ZMaChaineSC) step 2
   Tableau ( Wi -1) = Mid(ZMaChaineSC, Wi ,1)
   MatTemp = i
next i
```

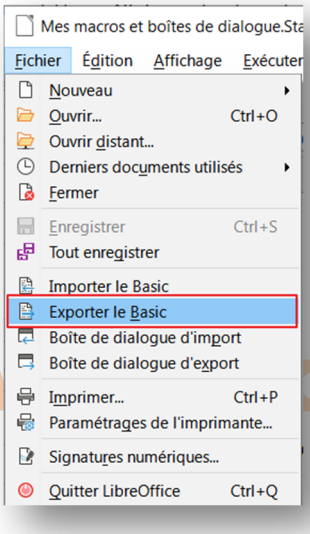
Solution :
Choisir une lettre réservée (par exemple W) comme première lettre pour tous les noms des variables du code source.

```
Dim Wi As Integer
Dim WTemp As Integer
Dim WMatTemp As Integer
Dim WTableau (len(ZMaChaineSC)) As String

For Wi = 1 to len(ZMaChaineSC) step 2
   WTableau ( Wi -1) = Mid(ZMaChaineSC, Wi ,1)
   WMatTemp = Wi

next Wi
```


En conclusion :
Ces 7 règles d’écriture peuvent sembler contraignantes mais elles assurent un cryptage réussi des macros Basic de LibreOffice.

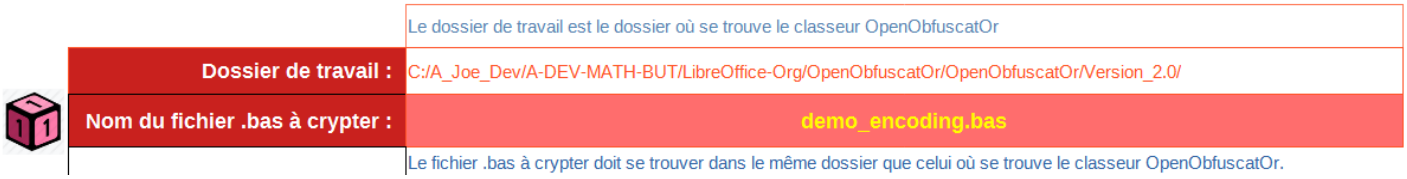
6- Mode d'emploi de OpenObfuscatOr

Indiquer le nom du code Basic à obfusquer (à crypter)



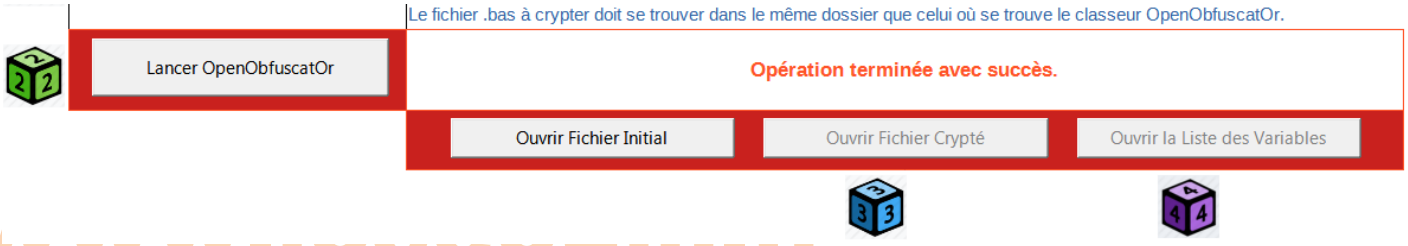
Exporter le code Basic du module dans un fichier .bas en actionnant le menu : "**Fichier/Exporter le Basic**" de l'Editeur Basic de LibreOffice.

Placer ce fichier .bas à obfusquer (crypter) **dans le même dossier** que celui où se trouve le classeur OpenObfuscatOr sur le disque dur puis saisir le nom de ce fichier .bas dans la cellule G6 (ici « **demo_encoding.bas** »).



Lancer OpenObfuscatOr afin de crypter le code Basic

Cliquer sur le bouton "**Lancer OpenObfuscatOr**" pour obfusquer (crypter) le fichier .bas
On obtient ainsi un fichier .bas dont le mot "Crypt" a été ajouté au nom initial.



Remarques :
1/ Une copie de sauvegarde du fichier .bas est automatiquement créé dans le même dossier.
2/ Un fichier d'extension .csv contient la liste des noms des variables des macros ainsi que les noms obfusqués (cryptés) correspondants.

Fichier Initial :	demo_encoding.bas
Fichier Crypté :	demo_encoding_Crypt.bas
Liste des Variables :	demo_encoding_Joob_Variables.csv

Récupération du code Basic obfusqué (crypté)

Ouvrir le fichier obfusqué (crypté) qui se trouve dans le dossier en cours à l'aide d'un éditeur de texte (notepad.exe) et copier l'ensemble du code Basic obfusqué (crypté). Coller le code Basic obfusqué (crypté) dans le module concerné en utilisant l'Editeur Basic de LibreOffice.

Consultation du nom des variables du code basic et des noms cryptés

Pour information, la liste des noms des variables du code Basic et des noms obfusqués (cryptés) est disponible dans un fichier .csv qui se trouve dans le dossier en cours.

Rappel :
JO-OBfuscator effectue une sauvegarde du code source du module (nom du fichier : « nom_du_code_Save.bas »).
Ainsi les code basic du module se trouvent en deux exemplaires sur le disque dur (nom des fichiers : « nom_du_code.bas » et « nom_du_code_Save.bas »).
Aucune donnée n'est effacée par OpenObfuscatOr. Aucune macro n'est effacée, elles sont toutes archivées dans des fichiers.

Toutes vos remarques et vos suggestions sont les bienvenues et m'aideront à faire évoluer cet outil. Si vous appréciez cette application tableur OpenObfuscatOr, vous pouvez faire un don pour m'encourager à créer les prochaines mises à jour. Vos dons permettront également de régler les frais de mise en ligne et de maintenance du site Internet.
Très belle journée à vous.

José OUIN - www.joseouin.fr