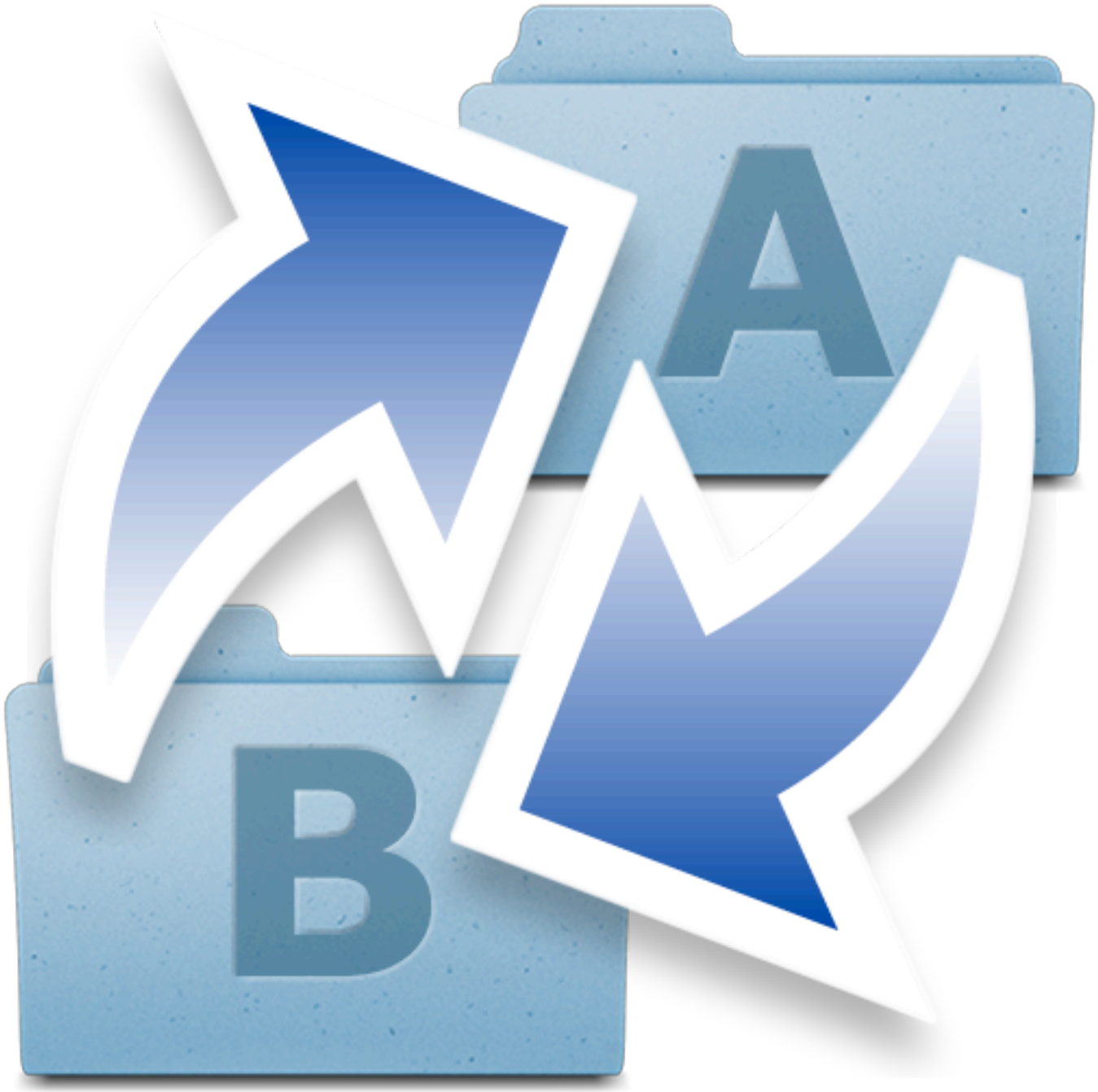
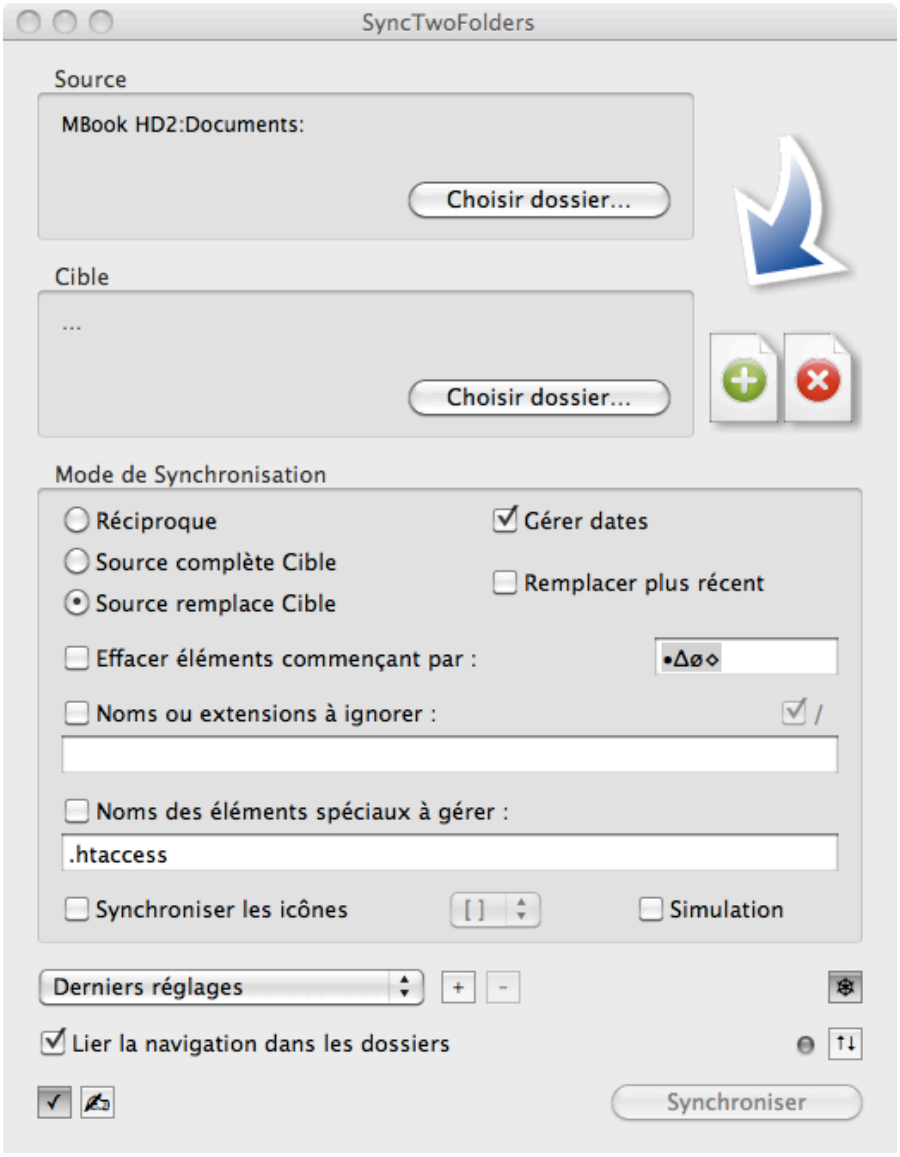
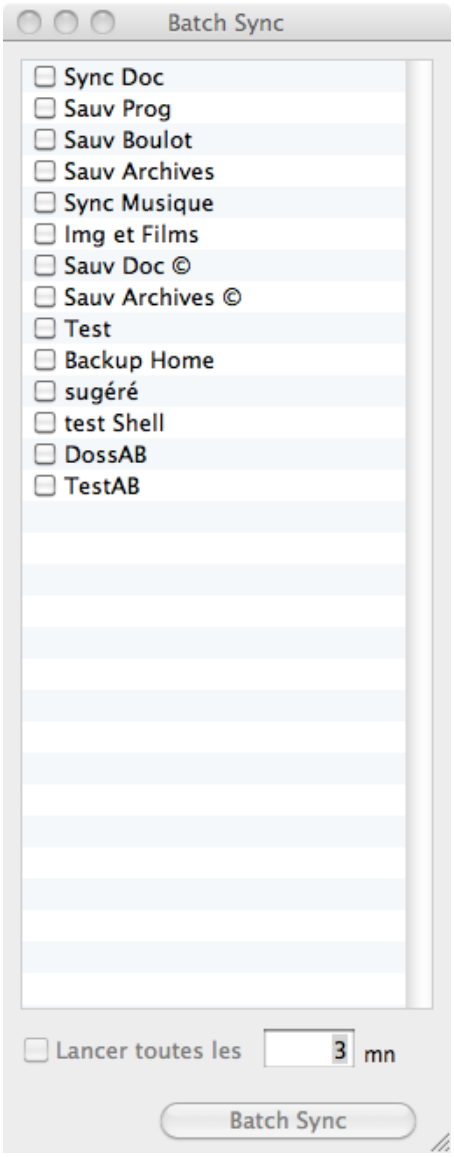
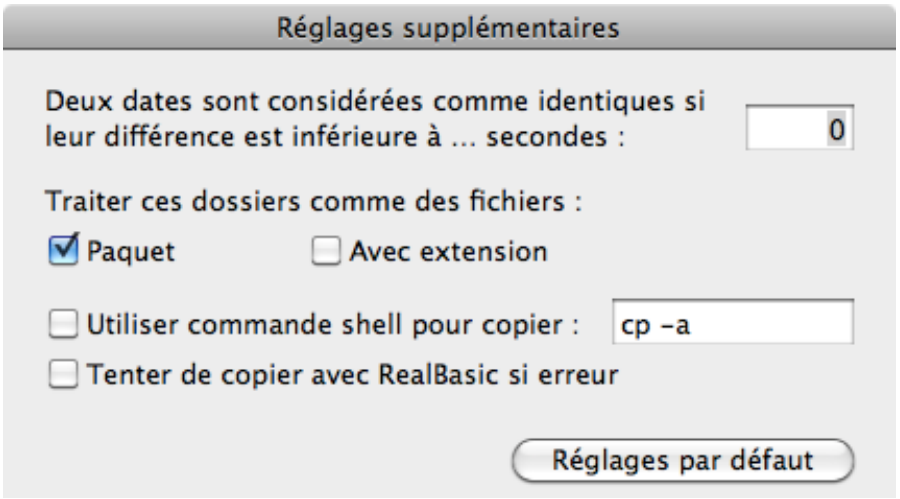
Code source de SyncTwoFolders v.1.8.6

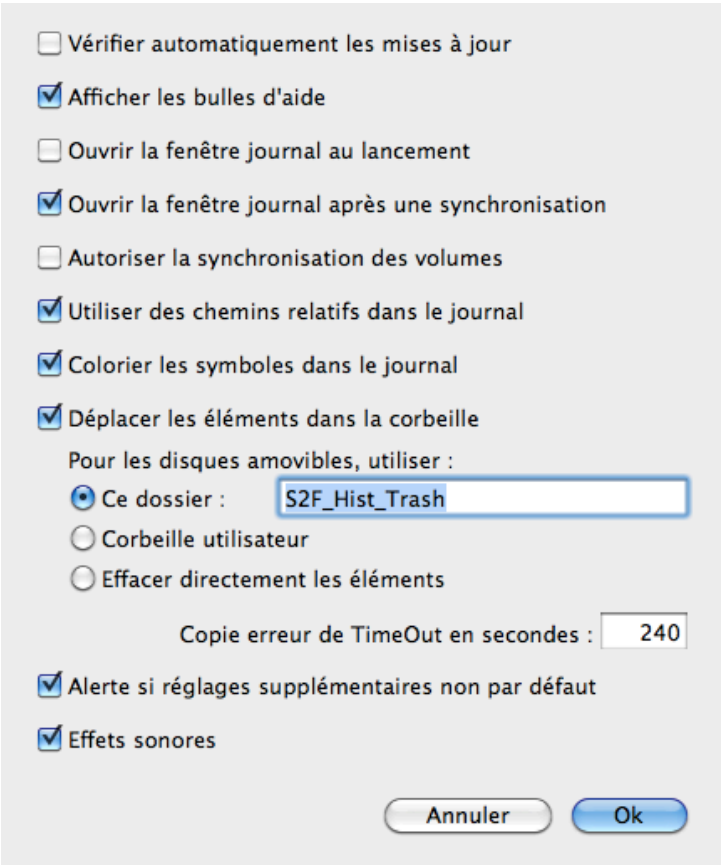


Fenêtres SyncTwoFolders 1.8.6.rbp

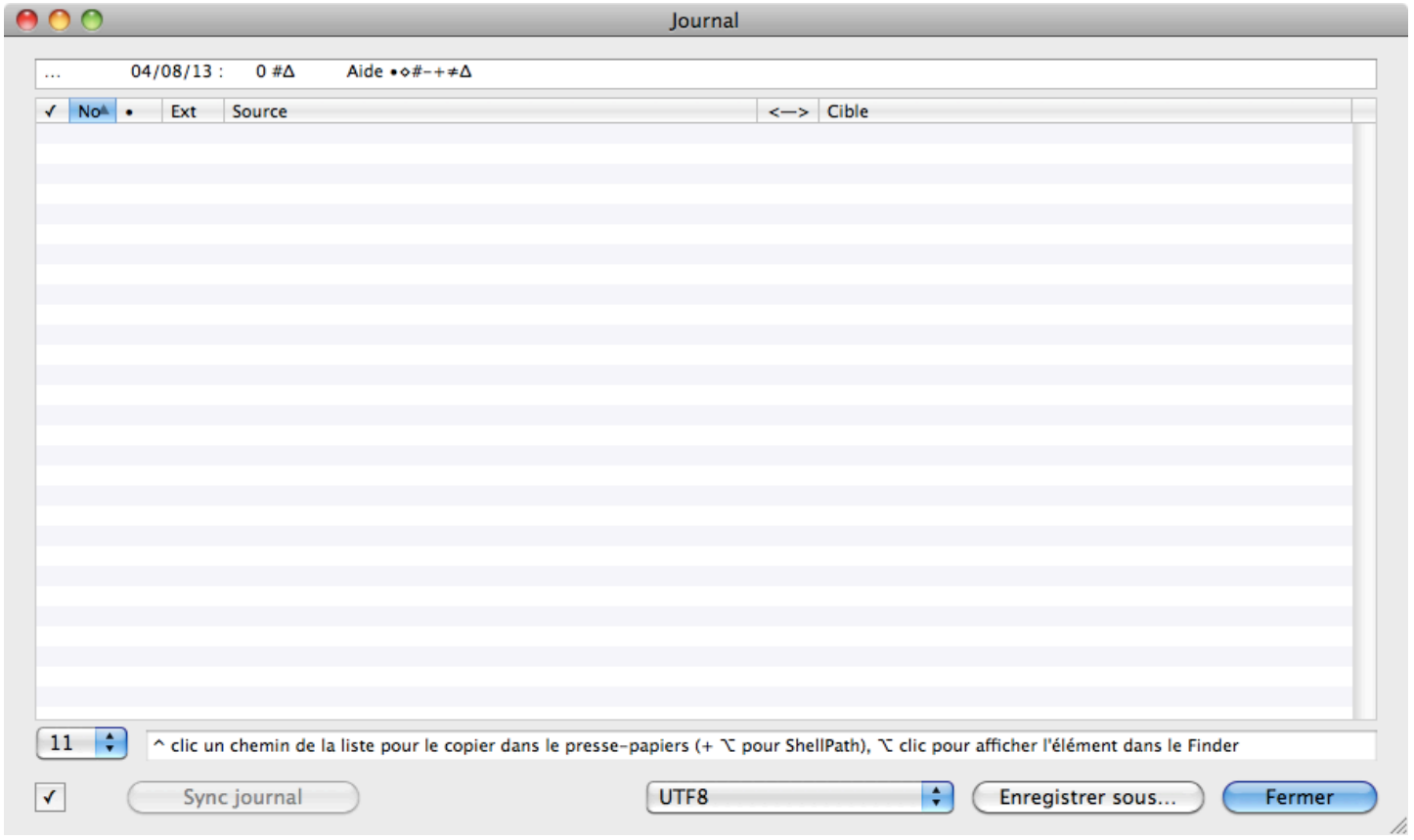







Vérifier automatiquement les mises à jour
Afficher les bulles d'aide
Ouvrir la fenêtre journal au lancement
Ouvrir la fenêtre journal après une synchronisation
Autoriser la synchronisation des volumes
Utiliser des chemins relatifs dans le journal
Colorier les symboles dans le journal
Déplacer les éléments dans la corbeille
Pour les disques amovibles, utiliser :
Ce dossier :
S2F_Hist_Trash
Corbeille utilisateur
Effacer directement les éléments
Copie erreur de TimeOut en secondes :
240
Alerte si réglages supplémentaires non par défaut
Effets sonores
Annuler
Ok


À propos de SyncTwoFolders
SyncTwoFolders
v.1.8.6
©2006, Th. Robisson et Ph. Galmel, tous droits réservés.
Ce programme synchronise deux dossiers. Les fichiers les plus anciens seront remplacés par les plus récents, dans certains cas, des fichiers seront effacés.
Utilisez les help tags pour vous familiariser avec les options.
Soyez sûrs de ce que vous faites quand vous lancez une synchronisation. Dans le doute, faites une Simulation, aucun fichier ne sera ni remplacé ni effacé, seules les icônes seront placées si l'option a été choisie. Les alias seront traités eux-mêmes, si un alias pointe vers un dossier, son contenu ne sera pas synchronisé. Les éléments invisibles et/ou précédés d'un "." (point) ne seront
#App.LienSitePerso
http://phg-home.com
th.rob@orange.fr
ph.galmel@orange.fr
Faire un don...
made with XOJO
Ok


Journal
... 04/08/13 : 0 #Δ Aide •◇#-+≠Δ
✓ No Ext Source <—> Cible
11
^ clic un chemin de la liste pour le copier dans le presse-papiers (+ ⌥ pour ShellPath), ⌥ clic pour afficher l'élément dans le Finder
✓
Sync journal
UTF8
Enregistrer sous...
Fermer

Project: SyncTwoFolders 1.8.6.rbp

Date: dimanche 4 août 2013 13:22:27

Project Info:

Mac (Carbon PEF) App Name: SyncTwoFolders

Mac (Carbon Mach-O) App Name: SyncTwoFolders

Windows App Name: SyncTwoFolders.exe

Linux App Name: SyncTwoFolders

Long Version: 1.8.6 - ©2006, Th. Robisson & Ph. Galmel

Major Version: 1

Minor Version: 8

Sub Version: 6

Release: 3

Non-Release:

Mac Creator Code: TrSf

Windows MDI Caption: SyncTwoFolders

Minimum Memory Size: 2048

Standard Memory Size: 4096

Class App

Inherits Application

Const ColBackList = &cF3F6FA

```
Const DefCdeCpShell = "cp -a "

Const DefCopyRBerr = False

Const DefDossExtFich = False

Const DefDossPackFich = True

Const DefMargeTps = 0

Const DefUseShell = False

Const DelaisAlerte = 10

Const DelaisAvQuit = 18

Private Const IndVers = "02"

Const kEditClear = "&Delete"

Const kFileQuit = "&Quit"

Const kFileQuitShortcut = ""

Const ListBoxLogColWidths = "22,30,30,40,*,40,*"

Const NomProg = "SyncTwoFolders"

Const SepPref = " :@: "
```

App.EnableMenuItems:

```
Sub EnableMenuItems()

   ' MsgBox "On est dans EnableMenuItems"

   AppleApropos.Enabled = DrapMenu
   AppleVMaJ.Enabled = DrapMenu
   FichierDon.Enabled = DrapMenu
   FichierPrefs.Enabled = DrapMenu
   FichierQuitter.Enabled = DrapMenu
```

```
  FichierChDossS.Enabled = DrapMenu
  FichierChDossC.Enabled = DrapMenu
  FichierSynchro.Enabled = DrapMenu and WinMain.DrapSyncEnab and (not LogSEnCo
  urs) ' Pas la peine de faire WinMain.UpdtSyncEnab car a été fait quand nécessaire
  If WinMain.BevButtAffBatch.Value Then ' Fenêtre WinBatch ouverte
    FichierBatchSync.Enabled = DrapMenu and WinBatch.DrapBatchEnab ' Pas la pei
    ne de faire WinBatch.UpdtBatchEnab car ça a été fait a chaque modif
  Else ' Menu inactif si fenêtre fermée
    FichierBatchSync.Enabled = False
  End If
  FichierStop.Enabled = (not DrapMenu) and (SyncEnCours or LogSEnCours) ' pas not
  DrapMenu tout seul car quand on ouvre WinPrefs…
  FichierFermer.Enabled = DrapMenu

  ' EditCut.Enabled = DrapMenu
  ' EditCopy.Enabled = DrapMenu
  ' EditPaste.Enabled = DrapMenu
  ' EditClear.Enabled = DrapMenu

  WinProg.Enabled = True ' DrapMenu ' Reste actif
  WinLogMenu.Enabled = True ' DrapMenu ' Reste actif

  AideAide.Enabled = DrapMenu

End Sub
```

App.Open:

```
Sub Open()

  ' #If TargetMacOS Then
  ' DefAppEncod = Encodings.MacRoman ' On ne peut pas le définir en Constante (pa
  s un TextEncoding)
  ' #Else
  ' #If DebugBuild Then
  ' MsgBox "Problème Tom, tu devrais être en TargetMacOS !"
  ' #EndIf
  ' DefAppEncod = Encodings.WindowsLatin1 '  Ne pas mettre  WindowsANSI  car plan
  te aussi sec sous Windows98
  ' #EndIf
  ' Note "ISOLatin1" = "ISO-8859-1"  mais Attention, ce n'est pas "WindowsLatin1"
  DefAppEncod = Encodings.UTF8 ' Plante aussi sec sous Windows98 ?
```

```
  ' MsgBox "On est dans App event Open"
  DragDropTicks = Ticks

  ' initProg = False ' Il vaut mieux ne pas le faire ici, et les valeurs sont de de toute fa
  çon mises par défaut
  ' DragDropAlerte = False
  ' DragDropOuv = False

End Sub
```

App.OpenDocument:

```
Sub OpenDocument(item As FolderItem)

  ' Pour qu'une application accepte les Drag and Drop il faut créer un  FileTypes , puis
   aller dans l'onglet  Project,
  '  App et sélectionner le  FileType  dans les property à droite

  ' Mac OS X résoud automatiquement les alias quand Drag and Drop. Il est donc imp
  ossible de traiter l'alias lui-même
  '  si on lâche un alias sur l'application, c'est le fichier pointé par l'alias qui sera trait
  é. Si l'alias ne pointe sur rien il
  '  est impossible de le lâcher sur l'application

  ' On pourrait faire le test si  SyncEnCours  mais bon ... et pis il doit être possible d'y
   faire en même temps

  ' MsgBox "On est dans App event OpenDocument, sur l'élément : '" + Item.AbsPath_
  f + "'"
  If not initProg Then DragDropOuv = True ' L'application a été ouverte par Drag&Dro
  p

  If Ticks > (DragDropTicks + DelaisAlerte) Then DragDropAlerte = False ' Si jamais l'
  application n'a pas
  '  été quittée (car elle avait été ouverte normalement) il faut le remettre à False car c
  eci est un autre DragDrop

  If not DragDropAlerte Then ' Pour ne pas boucler sur 50 éléments lâchés sur appli
     If DragDropOuv Then
        AddTxtTrash(item, WinMain.CfTextTrash(0), WinMain.CfTrashIfBegin(0))
        ' On n'a pas encore fait ChangeConfig donc  GTextTrash  et  CheckBoxTras
        hIfBegin  n'ont pas été initialisées
```

```
    Else
      AddTxtTrash(item, WinMain.GTextTrash, WinMain.CheckBoxTrashIfBegin.V
      alue)
      '                                        = EditFieldTrashBegin.Text
    End If
  End If

  DragDropTicks = Ticks ' On attendra avant de refaire l'alerte pour ne pas en faire 5
  0 le cas échéant

End Sub
```

App.AideAide:

```
Function AideAide() As Boolean

  Dim MonTexteAide as String
  Dim rep as Int16

  MonTexteAide = FracTexte(Txt_AideNonDispo, 1) + NomProg + FracTexte(Txt_Aide
  NonDispo, 2) + Menu_AppleApropos + FracTexte(Txt_AideNonDispo, 3)

  MonTexteAide = MonTexteAide + TypeCompil

  rep = MyDialogBox(MonTexteAide, "", Btn_LireAbout, Btn_Fermer, Btn_FAQ, 0)
  If rep = 1 Then
    WinInfos.ShowModal
  ElseIf rep = 3 Then
    ShowURL(LienSitePerso + Lien_PrgRb + NomProg + "_FAQ.html")
    ' Else rep = 2 Fermer
    '  rien
  End If

End Function
```

App.AppleApropos:

```
Function AppleApropos() As Boolean

  WinInfos.ShowModal
```

```
End Function
```

App.AppleVMaJ:

```
Function AppleVMaJ() As Boolean

  WinMain.Socket_Ver_MaJ.Get(LienSitePerso + Lien_PrgRb + NomProg + "_V.txt")

End Function
```

App.FichierBatchSync:

```
Function FichierBatchSync() As Boolean

  If WinMain.BevButtAffBatch.Value Then
    WinBatch.ActButtBatchSync
  Else
    MsgBox "Error ! Please contact the authors." + EndOfLine + "FichierBatchSync, T
    his menu shouldn't be active while BevButtAffBatch is False!"
  End If

End Function
```

App.FichierChDossC:

```
Function FichierChDossC() As Boolean

  WinMain.ActButtCible(Nil)

End Function
```

App.FichierChDossS:

```
Function FichierChDossS() As Boolean

  WinMain.ActButtSource(Nil)

End Function
```

App.FichierDon:

```
Function FichierDon() As Boolean

   DejaDonne = True
   ShowURL(LienSitePerso + Lien_PayPage + NomProg)

End Function
```

App.FichierFermer:

```
Function FichierFermer() As Boolean

   If FenetrFront = Nil Then
      Beep
      MsgBox "Error ! Please contact the authors." + EndOfLine + "FicherFermer, Clos
      e which window?"
   Else
      FenetrFront.Close ' Soit WinMain soit WinLog, pas les autres qui sont Modales n
      i WinBatch
      ' On gére les checkmark du menu fenêtre dans l'event Close de WinLog, et si on
       quitte on s'en fout
   End If

End Function
```

App.FichierPrefs:

```
Function FichierPrefs() As Boolean

   WinPrefs.ShowModalWithin WinMain

End Function
```

App.FichierQuitter:

```
Function FichierQuitter() As Boolean

   ' MsgBox "On est dans menu Quitter"
```

```
  ' Pour être sûr de fermer WinMain en 1er, car on a besoin que les fenêtres satellites (Log, etc.)
  '  soient ouvertes pour enregistrer leur coordonées
  ' Les fenêtres sont de toute façon toutes fermées quand on quitte. Simplement là o n précise l'ordre

  WinMain.Close ' Pas Self car ce menu pourrait être dans App ou ailleurs
  ' Note : Fermée 2 fois si on quitte par le Dock lorsque SyncEnCours

  ' WinLog.Close ' Intile car fait dans Quit automatiquement

End Function
```

App.FichierStop:

```
Function FichierStop() As Boolean

  WinMain.ActButtSync ' Fera Stop vu que ce menu est activé quand Synchro en cours

End Function
```

App.FichierSynchro:

```
Function FichierSynchro() As Boolean

  WinMain.ActButtSync

End Function
```

App.WinLogMenu:

```
Function WinLogMenu() As Boolean

  If WinMain.BevButtAffLog.Value Then ' Fenêtre déjà visible  Note : Si  LogSEnCours  a lors forcément visible
    WinLog.Show ' CheckWinMenu fait dans l'event Activate   Ca ne fait que la reme ttre au 1er plan si déjà ouverte
  Else ' Voir BevButtAffLog pour ouvrir la fenêtre WinLog
    ' If SyncEnCours Then WinMain.ThreadSync.Suspend ' Car il pourrait y avoir des  loupés dans le journal si la synchro continue,
```

```
    '  des fichiers seront copiés et ajoutés au journal alors qu'on n'aura pas fini  Wi
    nLog.AfficherJournal
    WinLog.Show ' CheckWinMenu fait dans l'event Activate
    ' If SyncEnCours Then WinMain.ThreadSync.Resume
  End If

End Function
```

App.WinProg:

```
Function WinProg() As Boolean

  WinMain.Show ' CheckWinMenu fait dans l'event Activate   Ca ne fait que la remettre
   au 1er plan si déjà ouverte

End Function
```

App.AddTxtTrash:

```
Sub AddTxtTrash(Elt as FolderItem, LTextTrash as String, DrapChBox as Boolean)

  Dim Nom, NouvNom as String
  Dim rep as Int16

  If not(Elt = Nil) Then ' Normalement impossible mais …
    If not DrapChBox Then LTextTrash = "" ' On n'utilise pas  WinMain.GTextTrash
    des fois que lancé en même temps que Synchro

    If LTextTrash = "" Then ' Si "" alors DrapChBox devait être à False, on aurait pu
    se contenter du test ci-dessus
      Beep
      DragDropAlerte = (MyDialogBox(Txt_ErrActiverEff, "", Btn_Arreter, "", "", 0)
      = 1) ' Obligé rep = 1 donc Vrai

    Else
      Nom = Elt.Name
      If InStr(Nom, LTextTrash) = 1 Then
        Beep
        rep = MyDialogBox(FracTexte(Txt_AdejaTextTrash, 1) + Nom + FracTe
        xte(Txt_AdejaTextTrash, 2) + LTextTrash + FracTexte(Txt_AdejaTextT
        rash, 3), "", Btn_Cont, Btn_Arreter, Btn_Supp, 0)
        If rep = 3 Then NouvNom = Mid(Nom, Len(LTextTrash) + 1) ' Mid(sour
        ce, start, [length])
```

```
      Else
        rep = 3 ' Comme ci-dessus pour test ci-dessous
        NouvNom = LTextTrash + Nom
      End If
      If rep = 3 Then
        If Elt.Parent.TrueChild(NouvNom).Exists Then
          Beep
          rep = MyDialogBox(FracTexte(Txt_NomExiste, 1) + NouvNom + Fr
          acTexte(Txt_NomExiste, 2), "", Btn_Cont, Btn_Arreter, "", 0)
        Else
          Elt.Name = NouvNom
          ' rep  reste à 3
        End If
      End If
      DragDropAlerte = (rep = 2)

    End If
    ' Else
    '  DragDropAlerte = True ' Bof non je laisse continuer
  End If

End Sub
```

App.AScriptCopy:

```
Function AScriptCopy(CopyFSource as FolderItem, CopyFDest as FolderItem, DossTrash a
s FolderItem, DrapRemplDest as Boolean, DrapUseShell as Boolean, TextCdeShell as Strin
g, DrapCopyRBerr as Boolean) As Boolean

  Dim DrapErr as Boolean
  Dim TampNbre as Int16 ' rep
  Dim NbEltsVisSrc, NbEltsVisCib as UInt32 ' Même que Type de retour de  CountVis_f
  SH
  Dim RepScript as String
  Dim VerifDest as FolderItem
  Dim CdeShell as New Shell

  ' Cette procédure lance l'AppleScript pour copier un élément (fichier ou dossier)
  ' Si  CopyFSource  est un alias cette procédure copie l'alias (et non l'élément pointé
  par l'alias)
  ' ATTENTION, Il y a un problème, AppleScript ne retourne pas "Annule" quand l'utilis
  ateur a annulé la copie d'un dossier ???!
```

```
#If DebugBuild Then
   ' Test ci-dessus équivalent au test ci-dessous
   If not CopyFDest.Directory Then
      MsgBox "Pb Tom, CopyFDest n'est pas un dossier." + EndOfLine + CopyFD
      est.Name + EndOfLine + "Package : " + Cstr(CopyFDest.IsPackage_f)
      CopyFDest = CopyFDest.Parent
   End If
#EndIf
VerifDest = CopyFDest.TrueChild(CopyFSource.Name) ' Ne sera pas à Nil puisque  C
opyFDest  existe
' MsgBox "VerifDest : '" + VerifDest.AbsPath_f + "'" + EndOfLine + "Effacer fichier de
stination (remplacement) : " + Cstr(DrapRemplDest)
' Si c'est un remplacement, on doit effacer le fichier existant dans destination
If DrapRemplDest Then
   If VerifDest.Exists Then
      ' Le Replacing Yes d'AppleScript fonctionne mal, surtout avec disque en rés
      eau. Le nom de l'éléments copié est indexé
      DrapErr = DeleteFileOrFolder(VerifDest, DossTrash) ' On efface  VerifDest
      car il faut l'effacer avant de le copier
      '  Il me semble que le  replacing yes  d'AppleScript ne marche pas sur les fi
      chiers d'ordinateur distant (réseau)
      '  DrapErr  à Vrai si erreur (éléments verrouillés ou pas d'autorisations ou e
      tc.)
   Else
      DrapErr = False ' Sera mis à Vrai si erreur
   End If
   ' Else ' DrapRemplDest à False
   ' DrapErr = VerifDest.Exists ' Si le fichier existe, on aura une erreur plus bas pui
   squ'on n'a pas  Replacing Yes
End If

' MsgBox "DrapErr = " + CStr(DrapErr) + EndOfLine + "On copie : " + CopyFSource.A
bsPath_f + EndOfLine + "vers : " + CopyFDest.AbsPath_f
' Note : CopyFSource.AbsPath_f  envoie le path de l'alias lui même si  CopyFSource
est un alias
If not DrapErr Then
   ' #If TargetMacOS Then CopyItem  NON car pas prévu autre que MacOS, comme
    ça Compil plantera
   If DrapUseShell Then
      ' MsgBox TextCdeShell + CopyFSource.ShellPath + " " + CopyFDest.ShellPa
      th
      ' PressPapTxt(TextCdeShell + CopyFSource.ShellPath + " " + CopyFDest.Sh
      ellPath)
      CdeShell.Execute TextCdeShell + CopyFSource.ShellPath + " " + CopyFDest
      .ShellPath ' NE PAS utiliser  "" + MonFolderItem.ShellPath_fAS + "". On n'ut
```

```
   ilise pas le Quotedform avec ' ' car s'il y a un ' dans le nom ça merde
   ' cp -a MonShellPathSource MonShellPathCible  =  cp -pPR MonShellPathSo
   urce MonShellPathCible  =  cp -p -P -R MonShellPathSource MonShellPath
   Cible
   TampNbre = CdeShell.ErrorCode
   If TampNbre = 0 Then
      RepScript = "Realise"
   Else
      RepScript = "Err n° " + str(TampNbre)
   End If
Else
   RepScript = CopyItemNo(CopyFSource.ShellPath_fAS, CopyFDest.ShellPath_
   fAS, Format(TimeOutCopy, F_MBillionSsEsp)) ' TimeOut en secondes
End If
' MsgBox "RepScript = '" + RepScript + "'"
Select Case Left(RepScript, 7)
Case "Realise"
   If CopyFDest.TrueChild(CopyFSource.Name).Exists Then
      If CopyFSource.Directory Then ' Si  CopyFSource  est un dossier alors
      CopyFDest.TrueChild(CopyFSource.Name)  est aussi un dossier
         ' Je compare nb élts si tout le contenu a bien été copié, car si ann
         ulé par utilisateur AppleScript ne renvoie pas d'erreur
         NbEltsVisSrc = CopyFDest.TrueChild(CopyFSource.Name).CountVi
         s_fSH ' Retourne 1 si dossier vide, 0 si erreur
         NbEltsVisCib = CopyFSource.CountVis_fSH ' Retourne 1 si dossier
         vide, 0 si erreur
         If (NbEltsVisSrc = 0) or (not(NbEltsVisSrc = NbEltsVisCib)) Then Go
         to ErreurCopie ' or ((NbEltsVisCib = 0)  mais alors (NbEltsVisSrc ≠
         NbEltsVisCib)
      End If
   Else
      Goto ErreurCopie
   End If
   ' Si copie OK  DrapErr  reste à False et on continue

Case "Annule '" Voir note dans l'AppleScript  CopyItem , dans l'AppleScript, on
n'a pas d'erreur (pas même err -128)
   Goto ErreurCopie '  si l'utilisateur clique  Arrêter  de la fenêtre  Copier du Fi
   nder
   '     Mais des fois que ça marche je fais tout de même le test   Et sinon ce s
   era  Else  plus bas

Case "Err n° "
   ' Beep
```

```
        ' rep = MyDialogBox(RepScript, "", Btn_Ok, "", "", 0) ' Je ne fais pas de mess
        age d'alerte pour ce cas sinon je
        '  devrais en faire pour les autres (erreur effacement etc.), et ça me fait chi
        er
        If DrapCopyRBerr Then ' Left(RepScript, 10) = "Err n° -17" Then ' Left(RepSc
        ript, 12) = "Err n° -1700" Erreur si Cible est un Package sous Leopard
          CopyFSource.CopyFileTo CopyFDest ' RealBasic peut désormais copier
          un dossier, avant il fallait le créer et copier
          DrapErr = not(CopyFSource.LastErrorCode = 0) '  ce qu'il y avait dedan
          s. Voir SyncTwoFolders 1.0.3
          ' Note : CopyFileTo ne copie que si la destination ne contient pas déjà
          le fichier à copier, s'il existe une erreur est générée, le fichier n'est pas
           remplacé
        Else
          DrapErr = True
        End If

      Else ' Si on a éjecté le disque pendant la copie ou … ???  on a annulé (voir note
      Case "Annule")
        ' DrapErr = True ' La copie ne s'est pas faite
        ' Beep ' Là par contre ce n'est pas normal
        ' MsgBox "Error ! Please contact the authors." + EndOfLine + "Unknown Err
        or in Script CopyItem !" + EndOfLine + """ + RepScript + """
        Goto ErreurCopie

      End Select
    End If

    Return DrapErr ' Vrai si Erreur


Exception Err ' Je ne vois pas trop quoi mais bon, problème d'autorisations ?

    ErreurCopie: ' Goto

    AppliAuPplan ' On rappelle SyncTwoFolders au 1er plan
    WinMain.ArretUrg = True ' On arrête tout
    ' Inutile DrapErr = True ' La copie ne s'est pas faite

    Return True ' Il y a eu une erreur

End Function
```

App.CaptSyncSimu:

```
Sub CaptSyncSimu(DrapSyncSimu as Boolean, DrapTestAff as Boolean, DrapTestEnab as B
oolean)

  Dim BtnCaptSyncSimu, MenuTextSyncSimu as String

  ' MsgBox "On change PushButtSync.Caption et FichierSynchro.Text"
  If DrapSyncSimu Then ' CheckBoxSimu ou CfSimul()
    MenuTextSyncSimu = Btn_Simu
  Else
    MenuTextSyncSimu = Btn_Sync
  End If

  If SyncEnCours Then
    BtnCaptSyncSimu = Btn_Arreter
    If DrapTestEnab Then
      If FichierSynchro.Enabled Then MsgBox "Error !" + EndOfLine + "Please cont
      act the authors. CaptSyncSimu" + EndOfLine + EndOfLine + "PushButtSync.
      Caption is '" + Btn_Arreter + "'" + " FichierSynchro is Enabled, it shouldn't."
    End If
  Else
    BtnCaptSyncSimu = MenuTextSyncSimu
    If DrapTestEnab Then
      If not((WinMain.DrapSyncEnab = WinMain.PushButtSync.Enabled) and (WinM
      ain.DrapSyncEnab = FichierSynchro.Enabled)) Then MsgBox "Error !" + End
      OfLine + "Please contact the authors. CaptSyncSimu, they should be equal
      :" + EndOfLine + EndOfLine + "DrapSyncEnab = " + Cstr(WinMain.DrapSync
      Enab) + EndOfLine + "PushButtSync.Enabled = " + Cstr(WinMain.PushButtSy
      nc.Enabled) + EndOfLine + "FichierSynchro.Enabled = " + Cstr(FichierSynch
      ro.Enabled)
    End If
  End If

  If DrapTestAff Then
    If not((WinMain.PushButtSync.Caption = BtnCaptSyncSimu) and (FichierSynchro.
    Text = MenuTextSyncSimu)) Then MsgBox "Error !" + EndOfLine + "Please conta
    ct the authors. CaptSyncSimu" + EndOfLine + EndOfLine + "PushButtSync.Capti
    on should be '" + BtnCaptSyncSimu + "'" + EndOfLine + "FichierSynchro.Text sh
    ould be '" + MenuTextSyncSimu + "'"
  Else
    WinMain.PushButtSync.Caption = BtnCaptSyncSimu
    FichierSynchro.Text = MenuTextSyncSimu
    EnableMenuItems ' Pour menu FichierSynchro , pas forcément utile car le menu
    se mettra automatiquement à jour dès qu'on l'affichera, mais je préfère qu'il so
```

```
    it à jour
    '                    En plus ça fait merder ma vérification c-dessus lors du proc
    hain appel de  CaptSyncSimu  avec  DrapTestEnab  à True
  End If

  ' Note : On ne vient pas ici si LogSEnCours, on désactive simplement PushButtSync

End Sub
```

App.CheckWinMenu:

```
Sub CheckWinMenu(DrapCheckW as Boolean)

  WinProg.Checked = DrapCheckW
  WinLogMenu.Checked = not DrapCheckW

  ' Pourrait être utile si plus de 2 fenêtres mais ne marche pas, il faudrait comparer le
   Title ou autre mais si 2 fenêtres ont le même titre … ?!
  ' If FenetrFront = WinProg Then
  ' Beep
  ' Else
  ' If Sfx Then RadioBeep_t.Play
  ' End If

End Sub
```

App.CreerTabEncod:

```
Sub CreerTabEncod()

  #If DebugBuild Then ' EXACTEMENT même Methods dans MyPopBarrier, TextBatchC
  onv, MemoDate et SyncTwoFolders
    Dim iNbre, jNbre, Fin_iNbre as Int16
    Dim TampAlert as String

    If not(Ubound(EncodFich) = -1) Then MsgBox "CreerTabEncod" + EndOfLine + "
    EncodFich n'est pas vide, " + str(Ubound(EncodFich)) + " élts"
  #EndIf

  ' ATTENTION, toute modification doit être fait en corrélation avec  FairePopupEnc  p
  our que ça corresponde
  ' Coller ce qui suit depuis le fichier Excel 'Liste Encodage.xlsx' dans 'Communs'
```

```
' ATTENTION : UTF8 doit être le premier, voir SauvFichier
EncodFich.Append Encodings.UTF8 ' n° 0
EncodFich.Append Encodings.UTF16 ' n° 1
EncodFich.Append Encodings.UTF16BE ' n° 2
EncodFich.Append Encodings.UTF16LE ' n° 3
EncodFich.Append Encodings.UTF32 ' n° 4
EncodFich.Append Encodings.UTF32BE ' n° 5
EncodFich.Append Encodings.UTF32LE ' n° 6
EncodFich.Append Encodings.ASCII ' n° 7
EncodFich.Append Encodings.DOSArabic ' n° 8
EncodFich.Append Encodings.DOSBalticRim ' n° 9
EncodFich.Append Encodings.DOSCanadianFrench ' n° 10
EncodFich.Append Encodings.DOSChineseSimplif ' n° 11
' EncodFich.Append Encodings.DOSChineseTrad
EncodFich.Append Encodings.DOSCyrillic ' n° 12
EncodFich.Append Encodings.DOSGreek ' n° 13
EncodFich.Append Encodings.DOSGreek1 ' n° 14
EncodFich.Append Encodings.DOSGreek2 ' n° 15
EncodFich.Append Encodings.DOSHebrew ' n° 16
EncodFich.Append Encodings.DOSIcelandic ' n° 17
' EncodFich.Append Encodings.DOSJapanese
' EncodFich.Append Encodings.DOSKorean
EncodFich.Append Encodings.DOSLatin1 ' n° 18
EncodFich.Append Encodings.DOSLatin2 ' n° 19
EncodFich.Append Encodings.DOSLatinUS ' n° 20
EncodFich.Append Encodings.DOSNordic ' n° 21
EncodFich.Append Encodings.DOSPortuguese ' n° 22
EncodFich.Append Encodings.DOSRussian ' n° 23
' EncodFich.Append Encodings.DOSThai
EncodFich.Append Encodings.DOSTurkish ' n° 24
EncodFich.Append Encodings.ISOLatin1 ' n° 25
EncodFich.Append Encodings.ISOLatin2 ' n° 26
EncodFich.Append Encodings.ISOLatin3 ' n° 27
EncodFich.Append Encodings.ISOLatin4 ' n° 28
EncodFich.Append Encodings.ISOLatin5 ' n° 29
EncodFich.Append Encodings.ISOLatin6 ' n° 30
EncodFich.Append Encodings.ISOLatin7 ' n° 31
EncodFich.Append Encodings.ISOLatin8 ' n° 32
EncodFich.Append Encodings.ISOLatin9 ' n° 33
EncodFich.Append Encodings.ISOLatinArabic ' n° 34
EncodFich.Append Encodings.ISOLatinCyrillic ' n° 35
EncodFich.Append Encodings.ISOLatinGreek ' n° 36
EncodFich.Append Encodings.ISOLatinHebrew ' n° 37
EncodFich.Append Encodings.KOI8_R ' n° 38
```

```
EncodFich.Append Encodings.MacArabic ' n° 39
EncodFich.Append Encodings.MacArmenian ' n° 40
EncodFich.Append Encodings.MacBengali ' n° 41
EncodFich.Append Encodings.MacBurmese ' n° 42
EncodFich.Append Encodings.MacCeltic ' n° 43
EncodFich.Append Encodings.MacCentralEurRoman ' n° 44
EncodFich.Append Encodings.MacChineseSimp ' n° 45
EncodFich.Append Encodings.MacChineseTrad ' n° 46
EncodFich.Append Encodings.MacCroatian ' n° 47
EncodFich.Append Encodings.MacCyrillic ' n° 48
EncodFich.Append Encodings.MacDevanagari ' n° 49
EncodFich.Append Encodings.MacDingbats ' n° 50
EncodFich.Append Encodings.MacEthiopic ' n° 51
EncodFich.Append Encodings.MacExtArabic ' n° 52
EncodFich.Append Encodings.MacGaelic ' n° 53
EncodFich.Append Encodings.MacGeorgian ' n° 54
EncodFich.Append Encodings.MacGreek ' n° 55
EncodFich.Append Encodings.MacGujarati ' n° 56
EncodFich.Append Encodings.MacGurmukhi ' n° 57
EncodFich.Append Encodings.MacHebrew ' n° 58
EncodFich.Append Encodings.MacIcelandic ' n° 59
EncodFich.Append Encodings.MacJapanese ' n° 60
EncodFich.Append Encodings.MacKannada ' n° 61
EncodFich.Append Encodings.MacKhmer ' n° 62
EncodFich.Append Encodings.MacKorean ' n° 63
EncodFich.Append Encodings.MacLaotian ' n° 64
EncodFich.Append Encodings.MacMalayalam ' n° 65
EncodFich.Append Encodings.MacMongolian ' n° 66
EncodFich.Append Encodings.MacOriya ' n° 67
EncodFich.Append Encodings.MacRoman ' n° 68
EncodFich.Append Encodings.MacRomanian ' n° 69
' EncodFich.Append Encodings.MacRomanLatin1
EncodFich.Append Encodings.MacSinhalese ' n° 70
EncodFich.Append Encodings.MacSymbol ' n° 71
EncodFich.Append Encodings.MacTamil ' n° 72
EncodFich.Append Encodings.MacTelugu ' n° 73
EncodFich.Append Encodings.MacThai ' n° 74
EncodFich.Append Encodings.MacTibetan ' n° 75
EncodFich.Append Encodings.MacTurkish ' n° 76
EncodFich.Append Encodings.MacVietnamese ' n° 77
' EncodFich.Append Encodings.ShiftJIS
' EncodFich.Append Encodings.WindowsANSI
EncodFich.Append Encodings.WindowsArabic ' n° 78
EncodFich.Append Encodings.WindowsBalticRim ' n° 79
```

```
EncodFich.Append Encodings.WindowsCyrillic ' n° 80
EncodFich.Append Encodings.WindowsGreek ' n° 81
EncodFich.Append Encodings.WindowsHebrew ' n° 82
EncodFich.Append Encodings.WindowsKoreanJohab ' n° 83
EncodFich.Append Encodings.WindowsLatin1 ' n° 84
EncodFich.Append Encodings.WindowsLatin2 ' n° 85
EncodFich.Append Encodings.WindowsLatin5 ' n° 86
EncodFich.Append Encodings.WindowsVietnamese ' n° 87
' EncodFich.Append Encodings.SystemDefault
' EncodFich.Append Encodings.UCS4 (2012r1 and earlier)

' EncodFich.Append GetTextEncoding(&h7E)
EncodFich.Append GetTextEncoding(&h8C) ' n° 88
' EncodFich.Append GetTextEncoding(&h98)
EncodFich.Append GetTextEncoding(&hEC) ' n° 89
' EncodFich.Append GetTextEncoding(&hFC)
' EncodFich.Append GetTextEncoding(&hFF)
' EncodFich.Append GetTextEncoding(&h0100)
EncodFich.Append GetTextEncoding(&h0101) ' n° 90
' EncodFich.Append GetTextEncoding(&h0101)
' EncodFich.Append GetTextEncoding(&h0103)
' EncodFich.Append GetTextEncoding(&h0103)
' EncodFich.Append GetTextEncoding(&h0201)
' EncodFich.Append GetTextEncoding(&h0202)
' EncodFich.Append GetTextEncoding(&h0203)
' EncodFich.Append GetTextEncoding(&h0204)
' EncodFich.Append GetTextEncoding(&h0205)
' EncodFich.Append GetTextEncoding(&h0206)
' EncodFich.Append GetTextEncoding(&h0207)
' EncodFich.Append GetTextEncoding(&h0208)
' EncodFich.Append GetTextEncoding(&h0209)
' EncodFich.Append GetTextEncoding(&h0400)
' EncodFich.Append GetTextEncoding(&h0405)
' EncodFich.Append GetTextEncoding(&h0405)
' EncodFich.Append GetTextEncoding(&h0410)
' EncodFich.Append GetTextEncoding(&h0411)
' EncodFich.Append GetTextEncoding(&h0412)
' EncodFich.Append GetTextEncoding(&h0413)
' EncodFich.Append GetTextEncoding(&h0414)
' EncodFich.Append GetTextEncoding(&h0415)
' EncodFich.Append GetTextEncoding(&h0416)
' EncodFich.Append GetTextEncoding(&h0417)
' EncodFich.Append GetTextEncoding(&h0418)
' EncodFich.Append GetTextEncoding(&h0419)
```

```
' EncodFich.Append GetTextEncoding(&h041A)
' EncodFich.Append GetTextEncoding(&h041B)
' EncodFich.Append GetTextEncoding(&h041C)
' EncodFich.Append GetTextEncoding(&h041D)
' EncodFich.Append GetTextEncoding(&h0420)
' EncodFich.Append GetTextEncoding(&h0421)
' EncodFich.Append GetTextEncoding(&h0422)
' EncodFich.Append GetTextEncoding(&h0423)
' EncodFich.Append GetTextEncoding(&h0500)
' EncodFich.Append GetTextEncoding(&h0500)
' EncodFich.Append GetTextEncoding(&h0501)
' EncodFich.Append GetTextEncoding(&h0502)
' EncodFich.Append GetTextEncoding(&h0503)
' EncodFich.Append GetTextEncoding(&h0504)
' EncodFich.Append GetTextEncoding(&h0505)
' EncodFich.Append GetTextEncoding(&h0506)
' EncodFich.Append GetTextEncoding(&h0507)
' EncodFich.Append GetTextEncoding(&h0508)
' EncodFich.Append GetTextEncoding(&h0510)
' EncodFich.Append GetTextEncoding(&h0600)
EncodFich.Append GetTextEncoding(&h0620) ' n° 91
' EncodFich.Append GetTextEncoding(&h0621)
EncodFich.Append GetTextEncoding(&h0622) ' n° 92
EncodFich.Append GetTextEncoding(&h0623) ' n° 93
EncodFich.Append GetTextEncoding(&h0624) ' n° 94
EncodFich.Append GetTextEncoding(&h0630) ' n° 95
' EncodFich.Append GetTextEncoding(&h0631)
EncodFich.Append GetTextEncoding(&h0640) ' n° 96
' EncodFich.Append GetTextEncoding(&h0641)
' EncodFich.Append GetTextEncoding(&h0651)
' EncodFich.Append GetTextEncoding(&h0652)
' EncodFich.Append GetTextEncoding(&h0653)
EncodFich.Append GetTextEncoding(&h0820) ' n° 97
EncodFich.Append GetTextEncoding(&h0821) ' n° 98
EncodFich.Append GetTextEncoding(&h0830) ' n° 99
EncodFich.Append GetTextEncoding(&h0831) ' n° 100
EncodFich.Append GetTextEncoding(&h0840) ' n° 101
EncodFich.Append GetTextEncoding(&h0920) ' n° 102
' EncodFich.Append GetTextEncoding(&h0930)
EncodFich.Append GetTextEncoding(&h0931) ' n° 103
' EncodFich.Append GetTextEncoding(&h0940)
' EncodFich.Append GetTextEncoding(&h0A01)
' EncodFich.Append GetTextEncoding(&h0A02)
' EncodFich.Append GetTextEncoding(&h0A03)
```

```
  ' EncodFich.Append GetTextEncoding(&h0A04)
  EncodFich.Append GetTextEncoding(&h0A05) ' n° 104
  EncodFich.Append GetTextEncoding(&h0B01) ' n° 105
  ' EncodFich.Append GetTextEncoding(&h0C01)
  EncodFich.Append GetTextEncoding(&h0C02) ' n° 106
  ' EncodFich.Append GetTextEncoding(&h0FFF)

  #If DebugBuild Then
    Fin_iNbre = Ubound(EncodFich)
    For iNbre = 0 to Fin_iNbre
      TampAlert = ""
      ' If EncodFich(iNbre).internetName = "ISO-2022-JP" Then MsgBox "ISO-202
      2-JP à iNbre = " + str(iNbre)
      For jNbre = (iNbre + 1) to Fin_iNbre
        If (EncodFich(iNbre).code = EncodFich(jNbre).code) or (EncodFich(iNbre
        ).internetName = EncodFich(jNbre).internetName) Then
          If TampAlert = "" Then
            TampAlert = "CreerTabEncod" + EndOfLine + "Il y a plusieurs
            encodages identiques" + EndOfLine + EncodFich(iNbre).intern
            etName + " - " + str(EncodFich(iNbre).code) + ",  iNbre = " +
            str(iNbre)
          End If
          TampAlert = TampAlert + EndOfLine + EncodFich(jNbre).internet
          Name + " - " + str(EncodFich(jNbre).code) + ",  jNbre = " + str(jNb
          re)
        End If
      Next jNbre
      If not(TampAlert = "") Then MsgBox TampAlert
    Next iNbre
  #EndIf

End Sub
```

App.CtrlOuvFenetres:

```
Private Sub CtrlOuvFenetres()

  Dim Jfenetr, FinJfenetr as Int16
  Dim TampText as String

  FinJfenetr = WindowCount - 1
  TampText = ""
  If FinJfenetr > 0 Then
```

```
    For Jfenetr = 0 to FinJfenetr
      TampText= EndOfLine + " • '" + Window(Jfenetr).Title + "'" + TampText
    Next Jfenetr
    MsgBox "Error ! Please contact the authors, PrefsCharger, Windows launched be
    fore read AffHelpTag !" + EndOfLine + " " + EndOfLine + str(FinJfenetr + 1) + "
    windows :" + TampText
  End If

End Sub
```

App.DeleteFileOrFolder:

```
Function DeleteFileOrFolder(DelElement as FolderItem, DossTrash as FolderItem) As Bool
ean

  Dim DrapErr as Boolean
  Dim TampNom, NouvNom as String
  ' Cette procédure déplace dans le dossier DossTrash  l'élément ou l'efface directem
  ent si  DossTrash  à Nil
  ' S'il s'agit d'un alias cette procédure efface l'alias (et non l'élément pointé par l'alias
  )

  If DossTrash = Nil Then ' On efface directement
    DrapErr = RemoveFileOrFolder(DelElement)

  Else
    ' MsgBox "On efface (déplace dans corbeille en l'indexant si nécessaire) :" + En
    dOfLine + DelElement.AbsPath_f + EndOfLine + "-> " + DossTrash.AbsPath_f
    DrapErr = False ' Sera mis à Vrai si erreur
    TampNom = DelElement.Name
    If DossTrash.Child(TampNom).Exists Then
      NouvNom = NomIndex_Doss(TampNom, DelElement.Extension_f, DossTra
      sh)
      If NouvNom = TampNom Then ' Pas trouvé de nom libre même indexé
        DrapErr = True
      Else
        DelElement.Name = NouvNom
        ' MsgBox "On vient de renommer '" + TampNom + "' en '" + NouvNom
        + "'" + EndOfLine + DelElement.AbsPath_f
      End If
    Else
      NouvNom = TampNom
    End If
```

```
    If not DrapErr Then ' Il n'y avait pas besoin de l'indexer ou on a pu l'indexer
      #If DebugBuild Then
        If DossTrash.TrueChild(NouvNom).Exists Then MsgBox "Tu t'es gourré
        Tom, DeleteFileOrFolder, cet élément aurait dû être renommé dans No
        mIndex_Doss :" + EndOfLine + DossTrash.TrueChild(NouvNom).AbsPa
        th_f
      #EndIf
      DelElement.MoveFileTo DossTrash
      If DossTrash.TrueChild(NouvNom).Exists Then ' Il est bien dans la corbeille
        If not(DelElement = Nil) Then ' Car comme il a été effacé ou déplacé il
        peut être à Nil
          DrapErr = DrapErr or (not(DelElement.LastErrorCode = 0)) ' or DelE
          lement.Exists ' Pas tout à fait même test quand dans RemoveFileO
          rFolder
          ' MsgBox DelElement.AbsPath_f + EndOfLine + "Error : " + str(Del
          Element.LastErrorCode) + EndOfLine + "Existe : " + Cstr(DelEleme
          nt.Exists)
          If (not DrapErr) and DelElement.Exists Then DrapErr = RemoveFile
          OrFolder(DelElement) ' Car sur autre volume ça fait une copie, pas
           un Move
        End If
      Else ' In n'est pas dans la corbeille
        ' If DelElement.Exists Then ' Donc  DelElement  est forcément resté là
        où il était (si jamais il y a eu un bug ben ça plantera)
        If not(NouvNom = TampNom) Then DelElement.Name = TampNom ' O
        n remet le nom d'avant puisque pas déplacé dans DossTrash
        ' MsgBox DelElement.AbsPath_f + EndOfLine + "existe toujours et n'a
        pas été déplacé dans la corbeille."
        DrapErr = True ' DrapErr or (not DossTrash.TrueChild(NouvNom).Exist
        s)
      End If

    End If

  End If ' DossTrash = Nil

  ' MsgBox "On a fini d'effacer : " + DelElement.AbsPath_f + EndOfLine + "Taille : " +
  Format(SizeTotal, "###\ ###\ ###\ ###\ ###\ ##0")

  Return DrapErr ' Vrai si Erreur


Exception Err
```

```
  ' Inutile DrapErr = True
  WinMain.ArretUrg = True ' On arrête tout
  Return True ' Il y a eu une erreur, sûrement qu'on a voulu boucler dans un dossier p
  our lequel on n'avait pas les
  '  autorisations ou on a éjecté le disque pendant l'effacement ou … ???

End Function
```

App.EnableBtnsA:

```
Sub EnableBtnsA()

  WinMain.CheckBoxRemplRecent.Enabled = DrapMenu and (WinMain.CheckBoxGerer
  Date.Value and (not(WinMain.RadioButtSyncMode(0).Value)))
  WinMain.CheckBoxPreNiv.Enabled = DrapMenu and WinMain.CheckBoxIgnorElt.Valu
  e

  ' Pas de CheckBoxPremNiv pour CheckBoxPasIgnor

  #If TargetMacOS Then ' Si Windows restent à False  !!! Que Mac OS !!!
    ' Ci-dessous on pourrait tester  WinMain.CheckBoxSyncIc.Enabled
    WinMain.PopupBarOut.Enabled = (DrapMenu or SyncEnCours) and WinMain.Che
    ckBoxSyncIc.Value '  Ca ne gêne en rien si  LogSEnCours  mais c'est plus logiqu
    e si désactivé
  #Else
    #If DebugBuild Then
      MsgBox "Problème Tom, tu devrais être en TargetMacOS !"
    #EndIf
  #EndIf

  WinMain.BevButtDelConfig.Enabled = DrapMenu and (not(WinMain.PopupConfig.ListI
  ndex = 0))

End Sub
```

App.EnableMenuBoutons:

```
Sub EnableMenuBoutons(TampDrap as Boolean)

  Dim iNbre as Int16
```

```
If TampDrap Then ' On met tout à jour
   If LogSEnCours Then
      EnableMenuItems
   Else
      ' EnableMenuItems NON car est appelé dans  CaptSyncSimu ci-dessous
      ' WinMain.PushButtSync.Enabled = DrapMenu NON puisque devient Arrêter
      , il doit rester Enabled
      CaptSyncSimu(WinMain.CheckBoxSimu.Value, False, False) ' Pas test pour te
      xte affiché car  PushButtSync.Caption  et FichierSynchro.Text  étaient sans
      doute différents
      '                                          Pas test pour Enabled car on n'a pas
      encore appelé  EnableMenuBoutons  (et UpdtSyncEnab)
      '
                           qui est appelé à la fin de  WinMain.CaptSyncSimu
   End If

   If WinMain.BevButtAffLog.Value Then ' Si Fenêtre Journal affichée
      WinLog.ListBoxLog.Enabled = DrapMenu ' Pour empêcher de checker Chec
      kBox
      WinLog.BevButtCheck.Enabled = DrapMenu ' and (UBound(JournCol1) > -1)
       ou not(WinMain.Journal = "")
      WinLog.PushButtEnregF.Enabled = DrapMenu ' and (UBound(JournCol1) > -
      1) ou not(WinMain.Journal = "")
      ' WinLog.PopupEncodSrc.Enabled = DrapMenu ' Ne contient pas de code
      ' WinLog.PushButtFermer = DrapMenu ' Reste toujours actif, on peut ferme
      r pendant la synchro normal et ça donne une alerte si synchro du Journal
      ' WinLog.PushButtLogSync.Enabled = DrapMenu and (not(WinLog.NbCellsC
      heck = 0)) ' and (UBound(JournCol1) > -1) ' Forcément si NbCellsCheck >
      0  Fait Ailleurs
   End If

   ' If WinMain.BevButtAffRegl.Value Then ' Sinon c'est que fenêtre est invisible ma
   is on disable quand même
   WinRegl.PushButtReglDef.Enabled = DrapMenu
   WinRegl.EdFieldMargeTps.Enabled = DrapMenu
   WinRegl.CheckBoxDossPack.Enabled = DrapMenu
   WinRegl.CheckBoxDossExt.Enabled = DrapMenu
   WinRegl.CheckBoxUseShell.Enabled = DrapMenu
   WinRegl.TextFieldCpShell.Enabled = DrapMenu
   WinRegl.CheckBoxCopyRBerr.Enabled = DrapMenu
   ' End If

   If WinMain.BevButtAffBatch.Value Then ' Sinon c'est que fenêtre est fermée  Pas
   la peine de faire WinBatch.UpdtBatchEnab car ça a été fait a chaque modif
```

```
    WinBatch.ListBoxBatch.Enabled = DrapMenu
    WinBatch.CheckBoxBatchAuto.Enabled = DrapMenu and WinBatch.DrapBatc
    hEnab
    WinBatch.EdFieldTimer.Enabled = DrapMenu
    WinBatch.PushButtBatchSync.Enabled = DrapMenu and WinBatch.DrapBatch
    Enab
End If

WinMain.PushButtSource.Enabled = DrapMenu
WinMain.PushButtCible.Enabled = DrapMenu

For iNbre = 0 to 2
    WinMain.RadioButtSyncMode(iNbre).Enabled = DrapMenu
Next iNbre
WinMain.CheckBoxGererDate.Enabled = DrapMenu
WinMain.CheckBoxTrashIfBegin.Enabled = DrapMenu ' Ca fait pas Mac  and (no
t(WinMain.GTextTrash = "")) ' EditFieldTrashBegin.Text   Si je réactive, à déplac
er dans EnableBtnsA
WinMain.EditFieldTrashBegin.Enabled = DrapMenu
WinMain.CheckBoxIgnorElt.Enabled = DrapMenu ' Ca fait pas Mac  and (Ubound
(WinMain.MemIgnorElt Nom et Ext) > -1) ' not(EditFieldIgnorElt.Text = ""))   Si j
e réactive, à déplacer dans EnableBtnsA
WinMain.EditFieldIgnorElt.Enabled = DrapMenu
WinMain.CheckBoxPasIgnor.Enabled = DrapMenu ' Ca fait pas Mac  and (Uboun
d(WinMain.MemPasIgnor) > -1) ' not(EditFieldPasIgnor.Text = ""))   Si je réactive
, à déplacer dans EnableBtnsA
WinMain.EditFieldPasIgnor.Enabled = DrapMenu

#If TargetMacOS Then ' Si Windows restent à False !!! Que Mac OS !!!
    ' (DrapMenu and (not LogSEnCours)) or SyncEnCours ' Si  SyncEnCours  alor
    s forcément pas  LogSEnCours  et si  LogSEnCours  alors pas  DrapMenu
    WinMain.CheckBoxSyncIc.Enabled = DrapMenu or SyncEnCours ' On peut s
    e mettre à synchroniser icônes en cours de synchro
#Else
    #If DebugBuild Then
        MsgBox "Problème Tom, tu devrais être en TargetMacOS !"
    #EndIf
#EndIf
WinMain.CheckBoxSimu.Enabled = DrapMenu

WinMain.CheckBoxLierNav.Enabled = DrapMenu ' On pourrait toutefois le chan
ger en cours de route sans problème
WinMain.BevButtInvDoss.Enabled = DrapMenu
```

```
  WinMain.PopupConfig.Enabled = DrapMenu
  WinMain.BevButtAddConfig.Enabled = DrapMenu
  WinMain.BevButtAffBatch.Enabled = DrapMenu ' Ca gênerait car si fenêtre ferm
  ée au départ alors son contenu n'a pas été désactivé
  ' WinMain.BevButtAffRegl.Enabled = DrapMenu ' Ca ne gêne pas car contenu dé
  sactivé plus haut

  ' WinMain.BevButtAffLog.Enabled = DrapMenu ' On peut afficher le Journal en c
  ours de route

  EnableBtnsA ' J'ai sorti ces boutons là car on peut les Enabled à d'autres mome
  nts (depuis ChangeConfig)

  ' !!! On ne peut pas annuler un  AcceptFileDrop' If DrapMenu Then !!!
  ' MsgBox "On valide DragDrop sur GroupBox"
  ' #If TargetMacOS Then
  ' WinMain.GroupBoxSource.AcceptFileDrop "folder" ' On aurait pu ajouter MACS
   en créateur mais ça pourrait être
  ' WinMain.GroupBoxCible.AcceptFileDrop "folder" '  autre chose tout en étant u
  n dossier ?!
  ' #Else
  ' #If DebugBuild Then
  ' MsgBox "Problème Tom, tu devrais être en TargetMacOS !"
  ' #EndIf
  ' WinMain.GroupBoxSource.AcceptFileDrop "any"
  ' WinMain.GroupBoxCible.AcceptFileDrop "any"
  ' #EndIf
  ' Else
  ' MsgBox "On annule DragDrop sur GroupBox"
  ' WinMain.GroupBoxSource.AcceptFileDrop ""
  ' WinMain.GroupBoxCible.AcceptFileDrop ""
  ' WinMain.CheckBoxTrashIfBegin.AcceptFileDrop ""
  ' WinMain.CheckBoxIgnorElt.AcceptFileDrop ""
  ' WinMain.CheckBoxPasIgnor.AcceptFileDrop ""
  ' End If

  #If DebugBuild Then
    If not LogSEnCours Then CaptSyncSimu(WinMain.CheckBoxSimu.Value, Tru
    e, True) ' Je teste tout car tout a dû être mis à jour
  #EndIf

Else ' On ne met à jour QUE les menus
  EnableMenuItems
  ' Ca marche également si cette procédure est dans WinMain, même sans mettre
   App. ??? Pourtant si on fait EnableMenuItems dans WinMain on appelle celui de
```

```
      WinMain ?
    '  qui est vide ?  Donc j'appelle  EnableMenuItems  depuis une procédure de Ap
    p pour être sûr car ça ne me plait pas

  End If ' TampDrap

End Sub
```

App.NomIndex_Doss:

```
Function NomIndex_Doss(NomElt as String, ExtElt as String, DossDestElt as FolderItem) A
s String

  Dim EltIndex as UInt16 ' 0 to 65535  ATTENTION, si change alors changer nb de dig
  it dans Format plus bas
  Dim TampNom, NomEltSsExt as String ' Cette procédure index  NomElt  de manière
  à ce que ce nom n'existe pas dans DossDestElt

  NomEltSsExt = Left(NomElt, Len(NomElt) - Len(ExtElt))

  #If DebugBuild Then
    If not(NomElt = (NomEltSsExt + ExtElt)) Then MsgBox "Tu t'es gourré Tom, dans
     NomIndex_Doss :" + EndOfLine + TampNom + EndOfLine + NomEltSsExt + Ex
    tElt
    If not DossDestElt.Exists Then MsgBox "Tu t'es gourré Tom, dans NomIndex_Do
    ss, DossDestElt  doit exister et donc ne pas être à Nil non plus !"
    If not DossDestElt.Child(NomElt).Exists Then MsgBox "Tu t'es gourré Tom, dans
    NomIndex_Doss, tu dois testé AVANT d'appeler cette procédure que cet éléme
    nt n'existe pas :" + EndOfLine + DossDestElt.Child(NomElt).AbsPath_f
  #EndIf

  EltIndex = 0

  ' While DossDestElt.Child(TampNom).Exists etc.   Wend  Marcherait aussi

  Do
    EltIndex = EltIndex + 1
    TampNom = NomEltSsExt + " " + Format(EltIndex, "####0") + ExtElt
  Loop Until (EltIndex = 0) or (not DossDestElt.Child(TampNom).Exists) ' 5 digits maxi
  puisque < 65536

  If EltIndex = 0 Then TampNom = NomElt ' = (NomEltSsExt + ExtElt)  Si EltIndex = 0
  c'est qu'on a fait le tour et pas trouvé de nom même indexé libre (pas testé avec ind
  ex 0 car je veux pas 'NomElt 0'
```

```
    '                       Alors on retourne le même nom envoyé, non indexé, et on gère
    dans procédure appelante

    Return TampNom

End Function
```

App.PrefsCharger:

```
Function PrefsCharger() As Boolean

    Dim iNbre, Fin_iNbre, nNbre, WMainL, WMainT, WMainW, WMainH, ModeSync, BarOu
    tFenetr, NoConfig as Int16
    Dim PrefsArray(-1), Tampon, NomConfigT, DossSourceT, DossCibleT, LTextTrash, I
    gnorEltList, PasIgnorList, CdeCpShell as String
    Dim MargeTps as UInt16
    Dim GererDate, RemplRecent, TrashIfBegin, SyncIcones, Simul, DossPackFich, DossE
    xtFich, UseShell, CopyRBerr, BatchSync, DrapFRegl, DrapFBatch as Boolean
    Dim DrapPrefs, IgnorElt, PasIgnor, PremNiv, TampAffLog as Boolean
    Dim vTtVar as Variant
    Dim FichierPref as FolderItem
    Dim Stream as TextInputStream

    FichierPref = DossierPref.Child(NomProg + " Prefs.txt") ' METTRE le MÊME Nom dans
     SavePrefs

    ' NOTE IMPORTANTE : Toutes les préférences sont définies par les valeurs par défau
    t AVANT de charger les préférences.
    ' Comme ça si l'une d'elle n'était pas lu dans le FichierPref  (ajout de nouvelle variab
    le) elle serait à sa valeur par défaut
    vTtVar = DateDuJour.SQLDateTime ' IL FAUT passer par un String  (SQLDateTime)  si
    non les dates sont liées "1904-01-02 12:00:00" ' 2ème jour, note 1er jour = 12:00
    AM, January 1, 1904
    DateLaunch = vTtVar.DateValue
    DejaDonne = False
    AutoCheckUpdt = False
    AffHelpTag = True
    OuvLogLch = True
    OuvLogApS = True
    AlertReglDef = True
    AutorisVol = False
    RelPathLog = False
    CoulDsListLog = True
```

```
DelTrash = False
AmovTrash = 0
NomDossHistTrash = "S2F_Hist_Trash"
TimeOutCopy = 240 ' 240 secondes = 4 minutes
Sfx = True
WMainL = -1
WMainT = -1 ' Coord hors norme comme ça si non lues dans prefs elles seront
ChgWinMainWH = True ' Est mis à Faux en bidouillant pref si le gars veut modifier l
a taille de WinMain dans les prefs, si reste à vrai ça se calcule suivant taille écran
WLogL = -1 '  mises à leur position pas défaut à la fin de cette procédure
WLogT = -1
WLogLargCol = ListBoxLogColWidths
WLogTailText = 2 ' Même chose que dans Properties de PopupTailleTextListe
WLogEncIndex = 0
' Aff F Regl
DrapFRegl = False
' Aff F Batch
DrapFBatch = False
WinMain.TabSymb = Array("◇", "•", "#", "?", "¿", "Δ", "+", "-", "≠", "<—>", "—>", "<—",
 "—X", "X—", "ƒ", "»")
'                                0   1   2   3   4  5   6   7   8     9     10    11
  12    13   14  15
' FichLog = ??? ' On laisse à Nil
' Aff F Journal
' Lier Navigation
NoConfig = 0 ' Pour tout ce qui suit, mettre la même chose plus bas dans cette proc
édure
NomConfigT = Txt_Attente
DossSourceT = Txt_Attente
DossCibleT = Txt_Attente
ModeSync = 0
GererDate = True
RemplRecent = False
TrashIfBegin = False
LTextTrash = "•Δø◇"
IgnorElt = False
PremNiv = True
IgnorEltList = ""
PasIgnor = False
PasIgnorList = ".htaccess"
SyncIcones = False
BarOutFenetr = 0
Simul = True
' Voir pour réglages ci-dessous WinRegl.PushButtReglDef et WinMain.AlertPasReglD
ef
```

```
MargeTps = DefMargeTps
DossPackFich = DefDossPackFich ' Paquet géré comme fichier par défaut
DossExtFich = DefDossExtFich ' Dossier avec extension géré comme dossier par déf
aut
UseShell = DefUseShell
CdeCpShell = DefCdeCpShell
CopyRBerr = DefCopyRBerr ' On n'utilisera l'instruction de copie RealBasic si l'instru
ction AppleScript ou Shell cp retourne une erreur
' WinBatch.CheckBoxBatchAuto.Value = False
BatchTempo = 30 ' Minutes
BatchSync = False

' MsgBox "Début chargement Pref"

If FichierPref.Exists Then
   Stream = TextInputStream.Open(FichierPref)
   Stream.Encoding = DefAppEncod

   ' Lecture des lignes de préfs.
   While not Stream.eof
      PrefsArray.AppEnd Stream.ReadLine
   Wend
   Stream.Close

   Fin_iNbre = UBound(PrefsArray)
   For iNbre = 0 to Fin_iNbre
      nNbre = InStr(PrefsArray(iNbre), SepPref)
      Tampon = Mid(PrefsArray(iNbre), nNbre + Len(SepPref))
      ' MsgBox ""“" + Left(PrefsArray(iNbre), nNbre - 1) + ""    “" + Tampon + ""”"
      Select Case Left(PrefsArray(iNbre), nNbre - 1)
      Case "Date Launch"
         vTtVar = Tampon
         Try ' Obligé de faire  Try  car  VarType(vTtVar.DateValue)  retourne tou
         jours 8 (String) et non 7 (Date)
            DateLaunch = vTtVar.DateValue ' Activer ligne ci-dessous pour te
            st
            ' DateLaunch.TotalSeconds = DateLaunch.TotalSeconds - 86400 '
            1 jour = 24 x 3600
         Catch ' err as UnsupportedFormatException ' On s'en fous si c'est une
         autre erreur mais je ne vois pas laquelle
            ' Normalement DateLaunch a gardé sa valeur d'initialisation du dé
            but de cette méthode
            #If DebugBuild Then '  sinon (si pas en DebugBuild) une erreur ci-
            dessous nous enverra à Exception Err
```

```
        MsgBox "Error ! Please contact the authors, PrefsCharger" + E
        ndOfLine + "DateLaunch = " + DateLaunch.SQLDateTime + E
        ndOfLine + "vTtVar.Date = " + vTtVar.DateValue.SQLDateTim
        e
      #EndIf
    End Try

Case "Deja Envoye Don " + IndVers ' Pour indexer et faire rappel de deman
de de don
    DejaDonne = (Tampon = "True")

Case "Auto Check Update"
    AutoCheckUpdt = (Tampon = "True")

Case "Afficher HelpTag"
    AffHelpTag = (Tampon = "True")
    CtrlOuvFenetres

Case "Ouvre Log at Launch"
    OuvLogLch = (Tampon = "True")

Case "Ouvre Log after Sync"
    OuvLogApS = (Tampon = "True")

Case "Alerte si pas Regl Def"
    AlertReglDef = (Tampon = "True")

Case "Autorise Sync Volume"
    AutorisVol = (Tampon = "True")

Case "Relatif Path Log"
    RelPathLog = (Tampon = "True")

Case "Colorier Liste Journal"
    CoulDsListLog = (Tampon = "True")

Case "Del Trash"
    DelTrash = (Tampon = "True")

Case "VolAmovible Trash"
    AmovTrash = Val(Tampon)

Case "Nom Doss HistTrash"
    NomDossHistTrash = Tampon
```

```
Case "TimeOut Copy"
   TimeOutCopy = Val(Tampon)

Case "Effets sonores"
   Sfx = (Tampon = "True")

Case "Fenetre Win"
   WMainL = Val(NthField(Tampon, ",", 1))
   WMainT = Val(NthField(Tampon, ",", 2))
   WMainW = Val(NthField(Tampon, ",", 3)) ' C'est fixe sauf si gars modifie
   WMainH = Val(NthField(Tampon, ",", 4))

Case "AutoSet WinMain Width/Height"
   ChgWinMainWH = (Tampon = "True")

Case "Fenetre Log"
   WLogL = Val(NthField(Tampon, ",", 1))
   WLogT = Val(NthField(Tampon, ",", 2))
   WLogW = Val(NthField(Tampon, ",", 3))
   WLogH = Val(NthField(Tampon, ",", 4))

Case "Larg Colonnes Liste Log"
   ' MsgBox str(CountFields(Tampon, ",")) + " = " + str(CountFields(ListBo
   xLogColWidths, ","))
   If CountFields(Tampon, ",") = CountFields(ListBoxLogColWidths, ",") Th
   en WLogLargCol = Tampon

Case "Taille Texte ListeLog"
   WLogTailText = Val(Tampon)

Case "Encodage Log"
   WLogEncIndex = Max(0, Min(Val(Tampon), Ubound(EncodFich)))

Case "Aff F Regl"
   WinMain.BevButtAffRegl.Value = (Tampon = "True")

Case "Fenetre Regl"
   DrapFRegl = True
   WinRegl.Left = Val(NthField(Tampon, ",", 1))
   WinRegl.Top = Val(NthField(Tampon, ",", 2))

Case "Aff F Batch"
   WinMain.BevButtAffBatch.Value = (Tampon = "True")
```

```
Case "Fenetre Batch"
   DrapFBatch = True
   WinBatch.Left = Val(NthField(Tampon, ",", 1))
   WinBatch.Top = Val(NthField(Tampon, ",", 2))
   ' WinBatch.Width = Val(NthField(Tampon, ",", 3))
   WinBatch.Height = Val(NthField(Tampon, ",", 4))

Case "Tab Symbols"
   ' MsgBox str(CountFields(Tampon, CarTab)) + " = " + str(UBound(Win
   Main.TabSymb) + 1)
   If CountFields(Tampon, CarTab) = (UBound(WinMain.TabSymb) + 1) Th
   en WinMain.TabSymb = Split(Tampon, CarTab)
   ' MsgBox Join(WinMain.TabSymb, CarTab)

Case "Fichier Log"
   FichLog = GetFitemAbsPath(Tampon, False)

Case "Aff F Journal"
   TampAffLog = (Tampon = "True")

Case "Lier Navigation"
   WinMain.CheckBoxLierNav.Value = (Tampon = "True")

Case "Batch Auto"
   WinBatch.CheckBoxBatchAuto.Value = (Tampon = "True")

Case "Batch Tempo"
   BatchTempo = Val(Tampon)

Case "Nom Config"
   NomConfigT = Tampon

Case "Dossier Source"
   DossSourceT = Tampon

Case "Dossier Cible"
   DossCibleT = Tampon

Case "Mode Synchro"
   ModeSync = Val(Tampon)

Case "Gerer Date"
   GererDate = (Tampon = "True")
```

```
Case "Rempl Recent"
   ReplRecent = (Tampon = "True")

Case "Trash If Beg"
   TrashIfBegin = (Tampon = "True")

Case "Text Trash"
   LTextTrash = Tampon

Case "Ignore Elts"
   IgnorElt = (Tampon = "True")

Case "Premier Niv Uniq"
   PremNiv = (Tampon = "True")

Case "Ignored Elts List"
   IgnorEltList = Tampon

Case "Not Ignore"
   PasIgnor = (Tampon = "True")

Case "Not Ignored List"
   PasIgnorList = Tampon

Case "Sync Icones"
   SyncIcones = (Tampon = "True")

Case "Barre Outils Fenetre"
   BarOutFenetr = Val(Tampon)

Case "Simulation"
   Simul = (Tampon = "True")

Case "Marge Temps"
   MargeTps = Val(Tampon)

Case "Doss Pack = Fich"
   DossPackFich = (Tampon = "True")

Case "Doss Ext = Fich"
   DossExtFich = (Tampon = "True")

Case "Utiliser Shell"
```

```
    UseShell = (Tampon = "True")

  Case "Cde cp shell"
    CdeCpShell = Tampon

  Case "Copie RB si Erreur"
    CopyRBerr = (Tampon = "True")

  Case "Batch Sync"
    BatchSync = (Tampon = "True")

  Case "@##@ @##@ @##@ @##@ @##@ @##@" ' Fin Config
    If Tampon = "@##@" Then ' Pas vraiment utile mais bon, pour garder
    même logique
      ' On valide enregistre Config chargée
      WinMain.CfNomConfig.Append NomConfigT
      WinMain.CfDossSource.Append DossSourceT
      WinMain.CfDossCible.Append DossCibleT
      WinMain.CfModeSync.Append ModeSync
      WinMain.CfGererDate.Append GererDate
      WinMain.CfRemplRecent.Append RemplRecent
      WinMain.CfTrashIfBegin.Append TrashIfBegin
      WinMain.CfTextTrash.Append LTextTrash
      WinMain.CfIgnorElt.Append IgnorElt
      WinMain.CfPremNiv.Append PremNiv
      WinMain.CfIgnorEltList.Append IgnorEltList
      WinMain.CfPasIgnor.Append PasIgnor
      WinMain.CfPasIgnorList.Append PasIgnorList
      WinMain.CfSyncIcones.Append SyncIcones
      WinMain.CfBarOutils.Append BarOutFenetr
      WinMain.CfSimul.Append Simul
      WinMain.CfMargeTps.Append MargeTps
      WinMain.CfDossPackFich.Append DossPackFich
      WinMain.CfDossExtFich.Append DossExtFich
      WinMain.CfUseShell.Append UseShell
      WinMain.CfCdeCpShell.Append CdeCpShell
      WinMain.CfCopyRBerr.Append CopyRBerr
      WinMain.CfBatch.Append BatchSync
      NoConfig = NoConfig + 1
      ' MsgBox "Config(" + str(NoConfig - 1) + ") : " + WinMain.CfNomC
      onfig(NoConfig - 1) +" = " + NomConfigT + EndOfLine + "Prochai
      n NoConfig : " + str(NoConfig)
      ' On réinitialise pour prochaine Config, mettre la même chose qu'
      au début de cette procédure
```

```
            NomConfigT = Txt_Attente
            DossSourceT = Txt_Attente
            DossCibleT = Txt_Attente
            ModeSync = 0
            GererDate = True
            RemplRecent = False
            TrashIfBegin = False
            LTextTrash = "•Δø◇"
            IgnorElt = False
            PremNiv = True
            IgnorEltList = ""
            PasIgnor = False
            PasIgnorList = ".htaccess"
            SyncIcones = False
            BarOutFenetr = 0
            Simul = True
            MargeTps = DefMargeTps
            DossPackFich = DefDossPackFich
            DossExtFich = DefDossExtFich
            UseShell = DefUseShell
            CdeCpShell = DefCdeCpShell
            CopyRBerr = DefCopyRBerr
            BatchSync = False
          End If '  Tampon = "@##@"

          ' Else ' Rien à faire
        End Select
    Next iNbre

    ' Else  Rien  NoConfig  reste à 0
End If ' FichierPref  existe

If OuvLogLch Then WinMain.BevButtAffLog.Value = TampAffLog ' On ouvrira la fenêt
re à la fin de WinMain.Open

' Replacement de la fenêtre
If (WMainL < 0) or (WMainT < 38) or (WMainW < 50) or (WMainH < 50) or ((WMainL +
 WMainW) > Screen(0).Width) or ((WMainT + WMainH) > Screen(0).Height) Then
    WMainL = 20 ' Ci-dessus je mets  < 50  et pas  < MinWidth MinHeight car calcu
    l au-dessous, juste pour éviter que 0 en largeur ou hauteur
    WMainW = WinMain.Width ' Ne sera pas plus large que l'écran
    WMainH = Min(WinMain.Height, Screen(0).Height - 38) ' Si vraiment trop petit et
     inférieur à WinMain.MinHeight ce sera remis plus bas
    WMainT = Min(60, Screen(0).Height - WMainH) ' Donc 38 si tout petit écran
```

```
End If
WinMain.Left = WMainL
WinMain.Top = WMainT
If (not ChgWinMainWH) or (WinMain.Height > (Screen(0).Height - 38)) Then ' Je ne te
ste pas  WMainH  car si la hauteur a changé entre 2 versions de SyncTwoFolders …
   WMainW = Max(Min(WinMain.MaxWidth, WMainW), WinMain.MinWidth) ' 38 = (2
   2 hauteur barre menu et 16 hauteur barre titre)
   WMainH = Max(Min(WinMain.MaxHeight, WMainH), WinMain.MinHeight)
   WinMain.RedimWin(True, WinMain.MaxHeight - WMainH) ' A initialiser AVANT d
   e changer les dimensions
   WinMain.Width = WMainW
   WinMain.Height = WMainH
End If ' Sinon  WinMain.Width  et  WinMain.Height  restent à leur valeur par défaut

WinRegl.RePosFenetr(False, not DrapFRegl) ' On ne l'a pas déjà appelé dans l'event W
inFenetr.Moved puisque InitProg encore à False
WinBatch.RePosFenetr(False, not DrapFBatch) ' On ne l'a pas déjà appelé dans l'event
 WinFenetr.Moved puisque InitProg encore à False

If NoConfig = 0 Then ' Pas de pref ou problème, pas trouvé  "@##@ @##@ @##@ @
##@ @##@ @##@"
   WinMain.CfNomConfig.Append "LastConfig" ' NomConfigT
   WinMain.CfDossSource.Append DossSourceT
   WinMain.CfDossCible.Append DossCibleT
   WinMain.CfModeSync.Append ModeSync
   WinMain.CfGererDate.Append GererDate
   WinMain.CfRemplRecent.Append RemplRecent
   WinMain.CfTrashIfBegin.Append TrashIfBegin
   WinMain.CfTextTrash.Append LTextTrash
   WinMain.CfIgnorElt.Append IgnorElt
   WinMain.CfPremNiv.Append PremNiv
   WinMain.CfIgnorEltList.Append IgnorEltList
   WinMain.CfPasIgnor.Append PasIgnor
   WinMain.CfPasIgnorList.Append PasIgnorList
   WinMain.CfSyncIcones.Append SyncIcones
   WinMain.CfBarOutils.Append BarOutFenetr
   WinMain.CfSimul.Append Simul
   WinMain.CfMargeTps.Append MargeTps
   WinMain.CfDossPackFich.Append DossPackFich
   WinMain.CfDossExtFich.Append DossExtFich
   WinMain.CfUseShell.Append UseShell
   WinMain.CfCdeCpShell.Append CdeCpShell
   WinMain.CfCopyRBerr.Append CopyRBerr
   WinMain.CfBatch.Append BatchSync
```

```
    DrapPrefs = True ' On affichera WinInfos.ShowModal
  Else ' Il y a un FichierPref
    DrapPrefs = False

  End If ' FichierPref  existe

  If not(WinMain.CfNomConfig(0) = "LastConfig") Then BeepNbT(3) ' C'est qu'il y a eu u
  n problème (bidouille des Prefs)
  WinMain.CfNomConfig(0) = Txt_LastConfig ' Je le remet à sa valeur localisée suivant
   langue

  ' MsgBox "Fin Chargement Pref" + EndOfLine + "N° dernier Réglage = " + str(UBoun
  d(WinMain.CfNomConfig))
  Return DrapPrefs


Exception Err

  BeepNbT(3)
  MsgBox "Error ! Please contact the authors." + EndOfLine + "PrefsCharger, problem
  while reading file!"
  Return True ' On affichera WinInfos.ShowModal

End Function
```

App.PrefsSauver:

```
Sub PrefsSauver(DrapAffWLog as Boolean)

  Dim iNbre, TampInt, NoConfig, Fin_CfNconfig as Int16
  Dim FichierPref as FolderItem
  Dim Stream as TextOutputStream
  Dim StreamText as String

  If PasNil_Existe_Alias(DossierPref, True) Then ' Si  DossierPref = Nil  c'est qu'on a qui
  tté avant de le définir
    FichierPref = DossierPref.Child(NomProg + " Prefs.txt") ' METTRE le MÊME Nom
    dans LoadPrefs
    ' MsgBox FichierPref.AbsPath_f

    StreamText = ""
```

```
StreamText = StreamText + "Date Launch" + SepPref + DateDuJour.SQLDateTi
me + EndOfLine

If DejaDonne Then ' Il y a Cstr mais …
   StreamText = StreamText + "Deja Envoye Don " + IndVers + SepPref + "Tr
   ue" + EndOfLine
Else
   StreamText = StreamText + "Deja Envoye Don " + IndVers + SepPref + "Fal
   se" + EndOfLine
End If

If AutoCheckUpdt Then ' Il y a Cstr mais …
   StreamText = StreamText + "Auto Check Update" + SepPref + "True" + En
   dOfLine
Else
   StreamText = StreamText + "Auto Check Update" + SepPref + "False" + En
   dOfLine
End If

If AffHelpTag Then ' Il y a Cstr mais …
   StreamText = StreamText + "Afficher HelpTag" + SepPref + "True" + EndOf
   Line
Else
   StreamText = StreamText + "Afficher HelpTag" + SepPref + "False" + EndO
   fLine
End If

If OuvLogLch Then ' Il y a Cstr mais …
   StreamText = StreamText + "Ouvre Log at Launch" + SepPref + "True" + E
   ndOfLine
Else
   StreamText = StreamText + "Ouvre Log at Launch" + SepPref + "False" + E
   ndOfLine
End If

If OuvLogApS Then ' Il y a Cstr mais …
   StreamText = StreamText + "Ouvre Log after Sync" + SepPref + "True" + E
   ndOfLine
Else
   StreamText = StreamText + "Ouvre Log after Sync" + SepPref + "False" + E
   ndOfLine
End If

If AlertReglDef Then ' Il y a Cstr mais …
```

```
   StreamText = StreamText + "Alerte si pas Regl Def" + SepPref + "True" + E
   ndOfLine
Else
   StreamText = StreamText + "Alerte si pas Regl Def" + SepPref + "False" +
   EndOfLine
End If

If AutorisVol Then ' Il y a Cstr mais …
   StreamText = StreamText + "Autorise Sync Volume" + SepPref + "True" + E
   ndOfLine
Else
   StreamText = StreamText + "Autorise Sync Volume" + SepPref + "False" +
   EndOfLine
End If

If RelPathLog Then ' Il y a Cstr mais …
   StreamText = StreamText + "Relatif Path Log" + SepPref + "True" + EndOfL
   ine
Else
   StreamText = StreamText + "Relatif Path Log" + SepPref + "False" + EndOf
   Line
End If

If CoulDsListLog Then ' Il y a Cstr mais …
   StreamText = StreamText + "Colorier Liste Journal" + SepPref + "True" + E
   ndOfLine
Else
   StreamText = StreamText + "Colorier Liste Journal" + SepPref + "False" + E
   ndOfLine
End If

If DelTrash Then ' Il y a Cstr mais …
   StreamText = StreamText + "Del Trash" + SepPref + "True" + EndOfLine
Else
   StreamText = StreamText + "Del Trash" + SepPref + "False" + EndOfLine
End If

StreamText = StreamText + "VolAmovible Trash" + SepPref + str(AmovTrash)
+ EndOfLine

StreamText = StreamText + "Nom Doss HistTrash" + SepPref + NomDossHistTr
ash + EndOfLine

StreamText = StreamText + "TimeOut Copy" + SepPref + Format(TimeOutCopy
, F_MBillionSsEsp) + EndOfLine
```

```
If Sfx Then ' Il y a Cstr mais …
   StreamText = StreamText + "Effets sonores" + SepPref + "True" + EndOfLi
   ne
Else
   StreamText = StreamText + "Effets sonores" + SepPref + "False" + EndOfLi
   ne
End If

StreamText = StreamText + "Fenetre Win" + SepPref + Str(WinMain.Left)+","+St
r(WinMain.Top)+","+Str(WinMain.Width)+","+Str(WinMain.Height) + EndOfLine

If ChgWinMainWH Then ' Il y a Cstr mais …
   StreamText = StreamText + "AutoSet WinMain Width/Height" + SepPref + "
   True" + EndOfLine
Else
   StreamText = StreamText + "AutoSet WinMain Width/Height"+ SepPref + "
   False" + EndOfLine
End If

StreamText = StreamText + "Fenetre Log" + SepPref + Str(WLogL)+","+Str(WLo
gT)+","+Str(WLogW)+","+Str(WLogH) + EndOfLine

StreamText = StreamText + "Larg Colonnes Liste Log" + SepPref + WLogLargC
ol + EndOfLine

StreamText = StreamText + "Taille Texte ListeLog" + SepPref + str(WLogTailTe
xt) + EndOfLine

StreamText = StreamText + "Encodage Log" + SepPref + str(WLogEncIndex) +
EndOfLine

If WinMain.BevButtAffRegl.Value Then ' Il y a Cstr mais …
   StreamText = StreamText + "Aff F Regl" + SepPref + "True" + EndOfLine
Else
   StreamText = StreamText + "Aff F Regl" + SepPref + "False" + EndOfLine
End If

StreamText = StreamText + "Fenetre Regl" + SepPref + Str(WinRegl.Left)+","+St
r(WinRegl.Top)+","+Str(WinRegl.Width)+","+Str(WinRegl.Height) + EndOfLine

If WinMain.BevButtAffBatch.Value Then ' Il y a Cstr mais …
   StreamText = StreamText + "Aff F Batch" + SepPref + "True" + EndOfLine
Else
```

```
   StreamText = StreamText + "Aff F Batch" + SepPref + "False" + EndOfLine
End If

StreamText = StreamText + "Fenetre Batch" + SepPref + Str(WinBatch.Left)+","
+Str(WinBatch.Top)+","+Str(WinBatch.Width)+","+Str(WinBatch.Height) + EndOfL
ine

StreamText = StreamText + "Tab Symbols" + SepPref +  Join(WinMain.TabSymb
, CarTab) + EndOfLine

If PasNil_Existe_Alias(FichLog, False) Then
   StreamText = StreamText + "Fichier Log" + SepPref + FichLog.AbsPath_f +
    EndOfLine
Else
   StreamText = StreamText + "Fichier Log" + SepPref + "" + EndOfLine ' GetF
   itemAbsPath("", ) retourne Nil
   '  "?@#%:@Aucun Fichier Log@"  renverra  Nil  car dossier  "?@#%"  n'existe
   pas
End If

If DrapAffWLog Then ' Il y a Cstr mais … WinMain.BevButtAffLog.Value a été mis
 à Faux quand on a fermé la fenêtre
   StreamText = StreamText + "Aff F Journal" + SepPref + "True" + EndOfLine
Else
   StreamText = StreamText + "Aff F Journal" + SepPref + "False" + EndOfLin
   e
End If

If WinMain.CheckBoxLierNav.Value Then ' Il y a Cstr mais …
   StreamText = StreamText + "Lier Navigation" + SepPref + "True" + EndOfLi
   ne
Else
   StreamText = StreamText + "Lier Navigation" + SepPref + "False" + EndOfL
   ine
End If

If WinBatch.CheckBoxBatchAuto.Value Then ' Il y a Cstr mais …
   StreamText = StreamText + "Batch Auto" + SepPref + "True" + EndOfLine
Else
   StreamText = StreamText + "Batch Auto" + SepPref + "False" + EndOfLine
End If

StreamText = StreamText + "Batch Tempo" + SepPref + Format(BatchTempo, F
_MBillionSsEsp) + EndOfLine ' WinBatch.EdFieldTimer.Text
```

```
' On sauvegarde Dernière Config '  WinMain.CfNomConfig(NoConfig) = Txt_Last
Config
NoConfig = 0
StreamText = StreamText + "Nom Config" + SepPref + "LastConfig" + EndOfLin
e ' On ne met pas  Txt_LastConfig  car on pourrait relancer l'app dans une autr
e langue

StreamText = StreamText + "Dossier Source" + SepPref + WinMain.StTextLire(
WinMain.StTextDossSource.Text) + EndOfLine

StreamText = StreamText + "Dossier Cible" + SepPref + WinMain.StTextLire(Wi
nMain.StTextDossCible.Text) + EndOfLine

TampInt = -1 ' On aura une Nil objection si problème lors de chargement Pref
For iNbre = 0 to 2
   If WinMain.RadioButtSyncMode(iNbre).Value Then TampInt = iNbre
Next iNbre
StreamText = StreamText + "Mode Synchro" + SepPref + str(TampInt) + EndOf
Line

If WinMain.CheckBoxGererDate.Value Then ' Il y a Cstr mais …
   StreamText = StreamText + "Gerer Date" + SepPref + "True" + EndOfLine
Else
   StreamText = StreamText + "Gerer Date" + SepPref + "False" + EndOfLine
End If

If WinMain.CheckBoxRemplRecent.Value Then ' Il y a Cstr mais …
   StreamText = StreamText + "Rempl Recent" + SepPref + "True" + EndOfLin
   e
Else
   StreamText = StreamText + "Rempl Recent" + SepPref + "False" + EndOfLi
   ne
End If

If WinMain.CheckBoxTrashIfBegin.Value Then ' Il y a Cstr mais …
   StreamText = StreamText + "Trash If Beg" + SepPref + "True" + EndOfLine
Else
   StreamText = StreamText + "Trash If Beg" + SepPref + "False" + EndOfLine
End If

StreamText = StreamText + "Text Trash" + SepPref + WinMain.GTextTrash + E
ndOfLine ' EditFieldTrashBegin.Text
' StreamText = StreamText + "Text CarAccept" + SepPref + WinMain.CarAccept
 + EndOfLine
```

```
If WinMain.CheckBoxIgnorElt.Value Then ' Il y a Cstr mais …
   StreamText = StreamText + "Ignore Elts" + SepPref + "True" + EndOfLine
Else
   StreamText = StreamText + "Ignore Elts" + SepPref + "False" + EndOfLine
End If

If WinMain.CheckBoxPreNiv.Value Then ' Il y a Cstr mais …
   StreamText = StreamText + "Premier Niv Uniq" + SepPref + "True" + EndOf
   Line
Else
   StreamText = StreamText + "Premier Niv Uniq" + SepPref + "False" + EndO
   fLine
End If

StreamText = StreamText + "Ignored Elts List" + SepPref + WinMain.TextIgnorE
lts(True) + EndOfLine ' EditFieldIgnorElt.Text SANS l'éventuel  SepAbsPath  à la f
in

If WinMain.CheckBoxPasIgnor.Value Then ' Il y a Cstr mais …
   StreamText = StreamText + "Not Ignore" + SepPref + "True" + EndOfLine
Else
   StreamText = StreamText + "Not Ignore" + SepPref + "False" + EndOfLine
End If

StreamText = StreamText + "Not Ignored List" + SepPref + WinMain.TextIgnorE
lts(False) + EndOfLine ' EditFieldPasIgnor.Text SANS l'éventuel  SepAbsPath  à la
 fin

If WinMain.CheckBoxSyncIc.Value Then ' Il y a Cstr mais …
   StreamText = StreamText + "Sync Icones" + SepPref + "True" + EndOfLine
Else
   StreamText = StreamText + "Sync Icones" + SepPref + "False" + EndOfLine
End If

StreamText = StreamText + "Barre Outils Fenetre" + SepPref + str(WinMain.Pop
upBarOut.ListIndex) + EndOfLine

If WinMain.CheckBoxSimu.Value Then ' Il y a Cstr mais …
   StreamText = StreamText + "Simulation" + SepPref + "True" + EndOfLine
Else
   StreamText = StreamText + "Simulation" + SepPref + "False" + EndOfLine
End If
```

```
StreamText = StreamText + "Marge Temps" + SepPref + Format(WinMain.Mem
MargeTps, F_MBillionSsEsp) + EndOfLine ' WinRegl.EdFieldMargeTps.Text

If WinRegl.CheckBoxDossPack.Value Then ' Il y a Cstr mais …
   StreamText = StreamText + "Doss Pack = Fich" + SepPref + "True" + EndO
   fLine
Else
   StreamText = StreamText + "Doss Pack = Fich" + SepPref + "False" + EndO
   fLine
End If

If WinRegl.CheckBoxDossExt.Value Then ' Il y a Cstr mais …
   StreamText = StreamText + "Doss Ext = Fich" + SepPref + "True" + EndOfL
   ine
Else
   StreamText = StreamText + "Doss Ext = Fich" + SepPref + "False" + EndOf
   Line
End If

If WinRegl.CheckBoxUseShell.Value Then ' Il y a Cstr mais …
   StreamText = StreamText + "Utiliser Shell" + SepPref + "True" + EndOfLine
Else
   StreamText = StreamText + "Utiliser Shell" + SepPref + "False" + EndOfLin
   e
End If

StreamText = StreamText + "Cde cp shell" + SepPref + WinRegl.TextFieldCpSh
ell.Text + EndOfLine

If WinRegl.CheckBoxCopyRBerr.Value Then ' Il y a Cstr mais …
   StreamText = StreamText + "Copie RB si Erreur" + SepPref + "True" + End
   OfLine
Else
   StreamText = StreamText + "Copie RB si Erreur" + SepPref + "False" + End
   OfLine
End If

StreamText = StreamText + "Batch Sync" + SepPref + "False" + EndOfLine ' Jam
ais BatchSync pour Dernière Config

StreamText = StreamText + "@##@ @##@ @##@ @##@ @##@ @##@" + SepPr
ef + "@##@" + EndOfLine

' On sauvegarde les Config supplémentaires
```

```
Fin_CfNconfig = UBound(WinMain.CfNomConfig)
For NoConfig = 1 To Fin_CfNconfig ' Ou n'importe quel autre  La "Config 0" a ét
é sauvegardée ci-dessus
   StreamText = StreamText + "Nom Config" + SepPref + WinMain.CfNomCo
   nfig(NoConfig) + EndOfLine

   StreamText = StreamText + "Dossier Source" + SepPref + WinMain.CfDoss
   Source(NoConfig) + EndOfLine

   StreamText = StreamText + "Dossier Cible" + SepPref + WinMain.CfDossCi
   ble(NoConfig) + EndOfLine

   StreamText = StreamText + "Mode Synchro" + SepPref + str(WinMain.CfMo
   deSync(NoConfig)) + EndOfLine

   If WinMain.CfGererDate(NoConfig) Then ' Il y a Cstr mais …
      StreamText = StreamText + "Gerer Date" + SepPref + "True" + EndOfL
      ine
   Else
      StreamText = StreamText + "Gerer Date" + SepPref + "False" + EndOfL
      ine
   End If

   If WinMain.CfRemplRecent(NoConfig) Then ' Il y a Cstr mais …
      StreamText = StreamText + "Rempl Recent" + SepPref + "True" + End
      OfLine
   Else
      StreamText = StreamText + "Rempl Recent" + SepPref + "False" + End
      OfLine
   End If

   If WinMain.CfTrashIfBegin(NoConfig) Then ' Il y a Cstr mais …
      StreamText = StreamText + "Trash If Beg" + SepPref + "True" + EndOf
      Line
   Else
      StreamText = StreamText + "Trash If Beg" + SepPref + "False" + EndOf
      Line
   End If

   StreamText = StreamText + "Text Trash" + SepPref + WinMain.CfTextTras
   h(NoConfig) + EndOfLine

   If WinMain.CfIgnorElt(NoConfig) Then ' Il y a Cstr mais …
      StreamText = StreamText + "Ignore Elts" + SepPref + "True" + EndOfLi
      ne
```

```
Else
   StreamText = StreamText + "Ignore Elts" + SepPref + "False" + EndOfL
   ine
End If

If WinMain.CfPremNiv(NoConfig) Then ' Il y a Cstr mais …
   StreamText = StreamText + "Premier Niv Uniq" + SepPref + "True" + E
   ndOfLine
Else
   StreamText = StreamText + "Premier Niv Uniq" + SepPref + "False" + E
   ndOfLine
End If

StreamText = StreamText + "Ignored Elts List" + SepPref + WinMain.CfIgn
orEltList(NoConfig) + EndOfLine

If WinMain.CfPasIgnor(NoConfig) Then ' Il y a Cstr mais …
   StreamText = StreamText + "Not Ignore" + SepPref + "True" + EndOfLi
   ne
Else
   StreamText = StreamText + "Not Ignore" + SepPref + "False" + EndOfL
   ine
End If

StreamText = StreamText + "Not Ignored List" + SepPref + WinMain.CfPasI
gnorList(NoConfig) + EndOfLine

If WinMain.CfSyncIcones(NoConfig) Then ' Il y a Cstr mais …
   StreamText = StreamText + "Sync Icones" + SepPref + "True" + EndOf
   Line
Else
   StreamText = StreamText + "Sync Icones" + SepPref + "False" + EndOf
   Line
End If

StreamText = StreamText + "Barre Outils Fenetre" + SepPref + str(WinMain
.CfBarOutils(NoConfig)) + EndOfLine

If WinMain.CfSimul(NoConfig) Then ' Il y a Cstr mais …
   StreamText = StreamText + "Simulation" + SepPref + "True" + EndOfLi
   ne
Else
   StreamText = StreamText + "Simulation" + SepPref + "False" + EndOfL
   ine
```

```
End If

StreamText = StreamText + "Marge Temps" + SepPref + Format(WinMain.
CfMargeTps(NoConfig), F_MBillionSsEsp) + EndOfLine

If WinMain.CfDossPackFich(NoConfig) Then ' Il y a Cstr mais …
   StreamText = StreamText + "Doss Pack = Fich" + SepPref + "True" + E
   ndOfLine
Else
   StreamText = StreamText + "Doss Pack = Fich" + SepPref + "False" +
   EndOfLine
End If

If WinMain.CfDossExtFich(NoConfig) Then ' Il y a Cstr mais …
   StreamText = StreamText + "Doss Ext = Fich" + SepPref + "True" + En
   dOfLine
Else
   StreamText = StreamText + "Doss Ext = Fich" + SepPref + "False" + E
   ndOfLine
End If

If WinMain.CfUseShell(NoConfig) Then ' Il y a Cstr mais …
   StreamText = StreamText + "Utiliser Shell" + SepPref + "True" + EndOf
   Line
Else
   StreamText = StreamText + "Utiliser Shell" + SepPref + "False" + EndO
   fLine
End If

StreamText = StreamText + "Cde cp shell" + SepPref + WinMain.CfCdeCpS
hell(NoConfig) + EndOfLine

If WinMain.CfCopyRBerr(NoConfig) Then ' Il y a Cstr mais …
   StreamText = StreamText + "Copie RB si Erreur" + SepPref + "True" +
   EndOfLine
Else
   StreamText = StreamText + "Copie RB si Erreur" + SepPref + "False" +
   EndOfLine
End If

If WinMain.CfBatch(NoConfig) Then ' Il y a Cstr mais …
   StreamText = StreamText + "Batch Sync" + SepPref + "True" + EndOfLi
   ne
Else
```

```
        StreamText = StreamText + "Batch Sync" + SepPref + "False" + EndOfL
        ine
      End If

      StreamText = StreamText + "@##@ @##@ @##@ @##@ @##@ @##@" + S
      epPref + "@##@" + EndOfLine

    Next NoConfig

    #If TargetWin32 Then
      If FichierPref.Exists Then
        If not FichierPref.Visible Then FichierPref.Visible = True ' Car ça plante
        sous Windows si on recrée (et écrit) un fichier invisible
      End If
    #EndIf
    Stream = TextOutputStream.Create(FichierPref)
    Stream.Write ConvertEncoding(StreamText, DefAppEncod)
    Stream.Close

  End If


Exception Err

  BeepNbT(3)
  MsgBox "Error ! Please contact the authors." + EndOfLine + "PrefsSauver, problem w
  hile writing file!"

End Sub
```

App.RemoveFileOrFolder:

```
Function RemoveFileOrFolder(DelElement as FolderItem) As Boolean

  Dim DrapErr as Boolean
  Dim iNbre, NbElts as UInt16
  ' Cette procédure efface un fichier ou un dossier (avec son contenu)  Même chose q
  ue dans MyPopBarrier à part ArretUrg
  ' On ne peut pas effacer directement un dossier non vide
  ' S'il s'agit d'un alias cette procédure efface l'alias (et non l'élément pointé par l'alias
  )

  ' MsgBox "On efface : " + DelElement.AbsPath_f
```

```
DrapErr = False ' Sera mis à Vrai si erreur

If DelElement.Directory Then '  DelElement  is a folder
   NbElts = DelElement.Count ' NE SURTOUT PAS faire  For i = 1 to DelElement.Co
   unt  car  DelElement.Count est décrémenté en cours de boucle
   ' MsgBox DelElement.AbsPath_f + " contient" + EndOfLine + str(NbElts) + " = "
   + str(DelElement.Count) + " éléments"
   For iNbre = NbElts downto 1 ' Go through each item
      ' MsgBox "On boucle sur " + str(NbElts) + " élts contenus, on efface l'élt ("
      + str(iNbre) + ") : " + DelElement.TrueItem(iNbre).AbsPath_f
      ' NE SURTOUT PAS aller dans le sens normal, ou alors en prenant item(1) à
      chaque fois et non l' item(iNbre) , car sinon
      '  lorsqu'on supprime l' item(iNbre)  l' item(iNbre+1)  n'existe plus car il est
       devenu l' item(iNbre)
      If DelElement.TrueItem(iNbre).Directory Then ' It's a folder
         DrapErr = DrapErr or RemoveFileOrFolder(DelElement.TrueItem(iNbre))
          ' Recursively call this routine passing it the folder

      Else
         DelElement.TrueItem(iNbre).Delete ' On l'efface
         If not(DelElement.TrueItem(iNbre) = Nil) Then ' Car comme il a été effa
         cé il peut être à Nil
            DrapErr = DrapErr or (not(DelElement.TrueItem(iNbre).LastErrorCo
            de = 0)) ' Bof non  or DelElement.TrueItem(iNbre).Exists ' Ca ralent
            i l'exécution, detoute façon
            ' MsgBox "Error : " + str(DelElement.TrueItem(iNbre).LastErrorCod
            e) '   si l'élément existe encore et pas de CodeErreur alors on aura
            un problème en effaçant dossier parent
         End If

      End If
   Next iNbre
   ' While (DelElement.Count > 0) NE PAS faire ça car si un élément n'a pas pu être
    effacé on boucle sans fin
   '  Ou alors il faudrait faire un exit repeat ce qui annulerait l'effacement des autr
   es éléments

End If ' Soit  DelElement  est un dossier vide soit c'est un fichier

' MsgBox "Fin, on efface : " + DelElement.AbsPath_f
DelElement.Delete ' On l'efface

If not(DelElement = Nil) Then ' Car comme il a été effacé ou déplacé il peut être à Nil
   DrapErr = DrapErr or (not(DelElement.LastErrorCode = 0)) or DelElement.Exists '
    Pas tout à fait même test quand dans DeleteFileOrFolder
```

```
      ' MsgBox DelElement.AbsPath_f + EndOfLine + "Error : " + str(DelElement.LastE
      rrorCode) + EndOfLine + "Existe : " + Cstr(DelElement.Exists)
    End If

    ' MsgBox "On a fini d'effacer : " + DelElement.AbsPath_f + EndOfLine + "Taille : " +
    Format(SizeTotal, "###\ ###\ ###\ ###\ ###\ ##0")

    Return DrapErr ' Vrai si Erreur


Exception Err

    ' Inutile DrapErr = True
    WinMain.ArretUrg = True ' On arrête tout
    Return True ' Il y a eu une erreur, sûrement qu'on a voulu boucler dans un dossier p
    our lequel on n'avait pas les
    '  autorisations ou on a éjecté le disque pendant l'effacement ou … ???

End Function

AffHelpTag As Boolean


AlertReglDef As Boolean


AmovTrash As Int16


AutoCheckUpdt As Boolean


AutorisVol As Boolean


BatchTempo As UInt16


Private ChgWinMainWH As Boolean


CoulDsListLog As Boolean
```

DateDuJour As Date

DateLaunch As Date

DejaDonne As Boolean

DelTrash As Boolean

DossierPref As FolderItem

DragDropAlerte As Boolean

DragDropOuv As Boolean

DragDropTicks As Double

EncodFich(-1) As TextEncoding

FenetrFront As Window

FichLog As FolderItem

initProg As Boolean

JourDcoulPos(4) As Int16

JournCol1(-1) As String

JournCol3(-1) As String

JournCol4(-1) As String

JournCol5(-1) As String

JournCol5b(-1) As String

JournCol6(-1) As String

JournCol7(-1) As String

JournCol7b(-1) As String

LogCdeCpShell(-1) As String

LogCopyRBerr(-1) As Boolean

LogDossTrash(-1) As FolderItem

LogEltCib(-1) As FolderItem

LogEltSrc(-1) As FolderItem

LogSEnCours As Boolean

LogUseShell(-1) As Boolean

NomDossHistTrash As String

OuvLogApS As Boolean

OuvLogLch As Boolean

RelPathLog As Boolean

SelLBoxBatch As Boolean

Sfx As Boolean

SyncEnCours As Boolean

TampCloseLog As Boolean

TimeOutCopy As UInt16

WLogEncIndex As Int16

WLogH As Int16

WLogL As Int16

WLogLargCol As String

WLogT As Int16

WLogTailText As Int16

WLogW As Int16

App Note: Mes Notes

Mes Notes

Avant je vérifié que DossierSource et DossierCible PasNil_Existe car au chargement d
es prefs
on pouvait avoir n'importe quoi si un truc pointait vers un disque qui n'était pus monté
.
Mais ce n'est plus indispensable car je ne sauvegarde plus DossierSource.AbsPath_f et
DossierCible.AbsPath_f
alors que quand je les sauvegardais de cette manière et qu'un des deux pointait vers u
ne disquette éjecté on
obtenais un truc du genre "Mondisque:MonDossier::MonBidulle" IL Y AvAIT :: (2 fois :) q
ui se suivaient

```
If PasNil_Existe_Alias(WinMain.DossierSource, True) Then ' On ne teste pas si à Nil car ris
que d'enregistrer n'importe quoi
Stream.WriteLine ConvertEncoding("Dossier Source" + SepPref + WinMain.DossierSource
.AbsPath_f, DefAppEncod)
Else
Stream.WriteLine ConvertEncoding("Dossier Source" + SepPref + Txt_Attente, DefAppEnc
od)
End If

If PasNil_Existe_Alias(WinMain.DossierCible, True) Then ' On ne teste pas si à Nil car risq
ue d'enregistrer n'importe quoi
Stream.WriteLine ConvertEncoding("Dossier Cible" + SepPref + WinMain.DossierCible.Ab
sPath_f, DefAppEncod)
Else
Stream.WriteLine ConvertEncoding("Dossier Cible" + SepPref + Txt_Attente, DefAppEnco
d)
End If



End Class
```

Class WinMain

Inherits Window

```
Private Const DefStTextSize = 12
```

```
Private Const SymbAlias = 15

Const SymbAncien = 5

Const SymbCopy = 6

Private Const SymbDoss = 14

Const SymbEff = 7

Const SymbFlecheD = 10

Const SymbFlecheG = 11

Const SymbFleches = 9

Const SymbFlEffD = 12

Const SymbFlEffG = 13

Const SymbOpErrDiv = 2

Const SymbOpErrInc = 3

Const SymbOpErrOvF = 4

Const SymbOpNon = 0

Const SymbOpOui = 1

Const SymbRempl = 8

Private Const TicksInt = 2

Private Const VmajTsLesXj = 0
```

WinMain.Activate:

```
Sub Activate()

   App.FenetrFront = Me
```

```
  ' A plus sa place dans event Paint  CoulPtitRond -> RefreshPtitRond ' Car si on mas
  que SyncTwoFolders le petit rond disparaît

  App.CheckWinMenu(True)

End Sub
```

WinMain.CancelClose:

```
Function CancelClose(appQuitting as Boolean) As Boolean

  Dim DrapAnnulFerme as Boolean

  If not appQuitting Then ' Car (voir FichierQuitter) si on quitte pendant synchro via le
   Dock on ferme 2 fois et
    '  on pose cette question 2 fois. appQuitting est vrai lorsqu'on a fini la procédu
    re de FichierQuitter
    If App.SyncEnCours or App.LogSEnCours Then
      Beep
      If MyDialogBox(Txt_SyncEnCours, "", Btn_Cont, Btn_Arreter, "", 0) = 2 Then '
       Retourne 1 pour Cont et 2 pour Arreter
        ArretUrg = True ' Note : Le thread est en pause pendant l'attente que l'
        utilisateur clique un bouton de la DialogBox
        If App.Sfx Then
          Zoum.Play ' Pour signifier prise en compte  ArretUrg     On ne l'ent
          ends pas si coup de frein juste après, mais des fois
          While Zoum.IsPlaying '                                    que ce soi
          t plus long pour stopper tout
          Wend
        End If
        ' While (App.SyncEnCours or App.LogSEnCours) ' Ca marche ici mais pa
        s dans WinLog ???
        ' Wend '                                                Même sans DoEvents
        Pause(10, False) ' Pour éviter 2 appuie trop vif
        ' If App.Sfx Then Coup_de_freins.Play ' A été fait dans le Thread de Syn
        chro
      End If
      DrapAnnulFerme = True ' NON, le gars refermera un seconde fois : App.Sy
      ncEnCours or App.LogSEnCours
    ElseIf not(DrapMenu or App.DragDropOuv) Then
      Beep
      DrapAnnulFerme = True
    Else
```

```
        DrapAnnulFerme = False
      End If
    Else
      DrapAnnulFerme = App.SyncEnCours or App.LogSEnCours or (not(DrapMenu or
      App.DragDropOuv))
    End If

    Return DrapAnnulFerme ' Close annulé si  True
    ' Normalement  SyncEnCours  et  DrapMenu  sont toujours inversés donc inutile de t
    ester les 2
    '  sauf quand on affiche  WinLog ou seul  DrapMenu  à Faux
    ' Ou on a ouvert l'application par Drag&Drop auquel cas aucun menu n'a été activé (
    et aucune synchro lancée)

End Function
```

WinMain.Close:

```
Sub Close()

    Dim TempDrap as Boolean

    ' Pas de sauvegarde si ouvert par Drag&Drop
    If App.initProg and (not App.DragDropOuv) Then ' On n'a pas ouvert l'application pa
    r Drag&Drop et on n'a pas quitté dès le open
      ZTestA ' Je teste que je n'ai pas fait de connerie en remettant les variables à leu
      r valeur  Mem…
      TempDrap = BevButtAffLog.Value ' Obligé de mettre ce flag car  BevButtAffLogs
      .Value  est mis à False
      '  quand on ferme  WinLog (et il faut la fermer AVANT pour avoir les coordonné
      es)
      If TempDrap Then WinLog.Close ' Et là BevButtAffLog.Value est mis à False
      ' MsgBox "On sauve Prefs"
      App.PrefsSauver(TempDrap)
    End If

    Quit

End Sub
```

WinMain.KeyDown:

```
Function KeyDown(Key As String) As Boolean

  ' Même chose dans chaque  event KeyDown  des autres fenêtres et que dans ActBut
  tSync

  If App.SyncEnCours or App.LogSEnCours Then
    ' If (Keyboard.AsynckeyDown(&h35) or (Key = Chr(27))) and (not ArretUrg) Then
     ' Touche Escape , pour pas faire 2 fois Zoum (procédure Synchro)
    ' Touche  Escape dans les 2 cas, je faisais les 2 tests !
    If (Key = Chr(27)) and (not ArretUrg) Then ' Touche Escape , pour pas faire 2 foi
    s Zoum (procédure Synchro)
      ArretUrg = True
      If App.Sfx Then
        Zoum.Play ' Pour signifier prise en compte  ArretUrg     On ne l'entend
        s pas si coup de frein juste après, mais des fois
        While Zoum.IsPlaying '                                   que ce soit plu
        s long pour stopper tout
        Wend
      End If
      ' NON, NE PAS faire boucle ci-dessous, sinon cet event (clic sur Bouton Syn
      chro/Arrêter) ne se fait QU'à la fin de la sychro, donc redémarre et arrête a
      ussi sec
      ' While (App.SyncEnCours or App.LogSEnCours) ' Si ça marche !  (NE PAS fai
      re ça car garde la main et le Thread ne se fini pas)
      ' Wend '                                                Même sans DoEvents
      Pause(10, False) ' Pour éviter 2 appuie trop vif
      ' If App.Sfx Then Coup_de_freins.Play ' A été fait dans le Thread de Synchro
    End If
  End If

  Return (not(Key = Chr(9))) ' Car si Tab on passe d'un TextField à l'autre

End Function
```

WinMain.Moved:

```
Sub Moved()

  If App.InitProg Then
    WinBatch.RePosFenetr(True, False)
    WinRegl.RePosFenetr(True, False)
  End If
```

```
End Sub
```

WinMain.Open:

```
Sub Open()

  Dim DrapPrefs as Boolean

  DecPosCtrl(Self) ' Décalage des controls ( Label , TextField , Slider , etc. )

  #If DebugBuild Then
    ' If App.VersionProg <> App.ShortVersion Then MsgBox "Tu t'es gourré dans le
    s versions Tom !" ' J'utilisais  App.VersionProg  car  App.ShortVersion  merde so
    us Windows98
    If (App.NomProg + ".debug") <> App.ExecutableFile.Name Then MsgBox "Tu t'e
    s gourré dans le nom du prog Tom !"
    If App.DelaisAvQuit <= App.DelaisAlerte Then MsgBox "DelaisAvQuit = " + str(
    App.DelaisAvQuit) + " <= DelaisAlerte = " + str(App.DelaisAlerte) + EndOfLine
    + "Il devrait être >"
    If not((StTextDossSource.TextSize = DefStTextSize) and (StTextDossCible.TextS
    ize = DefStTextSize)) Then MsgBox "Tu t'es gourré Tom, DefStTextSize doit fair
    e la taille par défaut des StaticText !"
  #EndIf
  ' MsgBox "Nom de l'application : " + App.ExecutableFile.Name + " - " + App.Execut
  ableFile.DisplayName + EndOfLine +  App.ExecutableFile.AbsPath_f
  ' Et oui, le .app n'y est pas ?!!!
  ' MsgBox "CetteAppli : '" + App.ExecutableFile.Parent.Parent.Parent.AbsPath_f + "'"
  + EndOfLine + "CetteAppli : '" + GetFolderItem("").Child(App.ExecutableFile.Name).
  AbsPath_f + ".app"

  ' La Date
  App.DateDuJour = New Date ' Date et heure actuelle
  ' MsgBox "DateDuJour : " + App.DateDuJour.SQLDateTime + EndOfLine + Format(Ap
  p.DateDuJour.TotalSeconds, "00000000000000000000.000000000")
  #If TargetWin32 Then
    Dim vTtVar as Variant
    vTtVar = App.DateDuJour.SQLDateTime ' Enlève les chiffres après la virgule, on
    pourrait faire  Round(), car sous Windows il y a des chiffres après la virgule
    App.DateDuJour = vTtVar.DateValue '  et ça merde quand compare date exacte
    ment (voir Neuronyx)
    ' MsgBox "DateDuJour : " + App.DateDuJour.SQLDateTime + EndOfLine + Form
    at(App.DateDuJour.TotalSeconds, "00000000000000000000.000000000")
    PushButtSource.TextSize = 12
```

```
    PushButtCible.TextSize = 12
    PushButtSync.TextSize = 12
    CheckBoxSyncIc.Value = False
    CheckBoxSyncIc.Enabled = False
    PopupBarOut.ListIndex = 0
    PopupBarOut.Enabled = False
#EndIf

' Le logo de SyncTwoFolders
Icone128 = New Picture(128, 128, 32)
Icone128.Graphics.DrawPicture SyncTwoFolders_Ic128, 0, 0
Icone128.Mask.Graphics.DrawPicture SyncTwoFolders_Ic128m, 0, 0

StTextDossSource.Text = Txt_Attente
StTextDossCible.Text = Txt_Attente

' Les petits points de couleurs
PtitPtGris = New Picture(16, 16, 32)
PtitPtGris.Graphics.DrawPicture PtitPointGris, 0, 0
PtitPtGris.Mask.Graphics.DrawPicture PtitPointMask, 0, 0
PtitPtJaune = New Picture(16, 16, 32)
PtitPtJaune.Graphics.DrawPicture PtitPointJaune, 0, 0
PtitPtJaune.Mask.Graphics.DrawPicture PtitPointMask, 0, 0
PtitPtVert = New Picture(16, 16, 32)
PtitPtVert.Graphics.DrawPicture PtitPointVert, 0, 0
PtitPtVert.Mask.Graphics.DrawPicture PtitPointMask, 0, 0

' Les SyncMode
SyncMode0P = New Picture(90, 210, 32)
SyncMode0P.Graphics.DrawPicture Mode0Img, 0, 0
SyncMode0P.Mask.Graphics.DrawPicture Mode0Mask, 0, 0
SyncMode1P = New Picture(90, 210, 32)
SyncMode1P.Graphics.DrawPicture Mode1Img, 0, 0
SyncMode1P.Mask.Graphics.DrawPicture Mode1Mask, 0, 0
SyncMode2P = New Picture(90, 210, 32)
SyncMode2P.Graphics.DrawPicture Mode2Img, 0, 0
SyncMode2P.Mask.Graphics.DrawPicture Mode2Mask, 0, 0

App.CreerTabEncod ' Pour PopupMenuEncod de WinLog

App.SyncEnCours = False
App.LogSEnCours = False
ArretUrg = False
DrapBatch = False
```

```
App.SelLBoxBatch = True ' Servira dans Change Config pour sélectionner ligne de Li
stBoxBatch ou non
DateSync = New Date
#If TargetWin32 Then
   ' Dim vTtVar as Variant ' Fait plus haut
   vTtVar = DateSync.SQLDateTime ' Enlève les chiffres après la virgule, on pourra
   it faire  Round(), car sous Windows il y a des chiffres après la virgule
   DateSync = vTtVar.DateValue '  et ça merde quand compare date exactement (v
   oir Neuronyx)
   ' MsgBox "DateSync : " + DateSync.SQLDateTime + EndOfLine + Format(DateSy
   nc.TotalSeconds, "00000000000000000000.000000000")
#EndIf
NomDerConfig = "" ' Sera utilisé comme nom par défaut pour l'enregistrement d'un
e config

' On définie le dossier Préférences
If PasNil_Existe_Alias(SpecialFolder.Preferences, True) Then
   App.DossierPref = SpecialFolder.Preferences
Else
   App.DossierPref = GetFolderItem("").Child(App.NomProg + "Prefs")
End If
If not(App.DossierPref = Nil) Then ' Des fois que vraiment merdé ci-dessus
   If not App.DossierPref.Exists Then App.DossierPref.CreateAsFolder
End If
If PasNil_Existe_Alias(App.DossierPref, True) Then
   DrapPrefs = App.PrefsCharger ' A faire avant  App.OpenDocument

   Pause(5, True) ' Pour laisser le temps  à event App.OpenDocument de se faire
   ' MsgBox "On est dans WinMain event open"

   If App.DragDropOuv Then ' On a ouvert l'application par Drag&Drop
      ' MsgBox "On est dans WinMain event open  Then"
      App.initProg = True
      While Ticks < (App.DragDropTicks + App.DelaisAvQuit) ' Plus grand que D
      elaisAlerte qu'on ait le temps d'afficher l'alerte
         App.DoEvents
      Wend
      Self.Close ' On quitte automatiquement
      ' La fenêtre WinMain n'aura même pas été affichée

   Else ' L'application a été ouverte par double-clic
      ' MsgBox "On est dans WinMain event open  Else"
      AffMasqHelpTag("", "")
      App.CaptSyncSimu(CfSimul(0), False, False)
```

```
' Voir note dans  EnableMenuBoutons  On ne peut pas annuler un  AcceptFi
leDrop
#If TargetMacOS Then
   GroupBoxSource.AcceptFileDrop "folder" ' On aurait pu ajouter MACS e
   n créateur mais ça pourrait être
   GroupBoxCible.AcceptFileDrop "folder" '  autre chose tout en étant un
   dossier ?!
#Else
   #If DebugBuild Then
      MsgBox "Problème Tom, tu devrais être en TargetMacOS !"
   #EndIf
   GroupBoxSource.AcceptFileDrop "any"
   GroupBoxCible.AcceptFileDrop "any"
#EndIf
CheckBoxTrashIfBegin.AcceptFileDrop "any"
' EditFieldTrashBegin.AcceptFileDrop "any" ' Plus de ragDrop sur EditField c
ar insère le texte quand on y lâche des fichiers liens ou Clipboard car conti
ennent du texte
CheckBoxIgnorElt.AcceptFileDrop "any"
CheckBoxPasIgnor.AcceptFileDrop "any"

InitJournalSync ' JournalDd  sera modifié suivant les cas,  ATTENTION ci-de
ssous, esp + 2 symboles d'erreurs = 3 caractères, voir  WinLog.EcrEdFieldL
og
JournalDf = (" " + TabSymb(SymbOpErrDiv) + TabSymb(SymbAncien) + Car
Tab + "" + CarTab + Menu_Aide + " " + TabSymb(SymbOpOui) + TabSymb(
SymbOpNon) + TabSymb(SymbOpErrDiv) + TabSymb(SymbEff) + TabSymb(
SymbCopy) + TabSymb(SymbRempl) + TabSymb(SymbAncien))
JournalD_HT = ("" + CarTab + TabSymb(SymbOpOui) + CarTab + SymbOp
Oui_HT)
JournalD_HT = JournalD_HT + EndOfLine + ("" + CarTab + TabSymb(Symb
OpNon) + CarTab + SymbOpNon_HT)
JournalD_HT = JournalD_HT + EndOfLine + ("" + CarTab + " " + TabSymb(S
ymbOpErrDiv) + CarTab + SymbOpErrDiv_HT)
JournalD_HT = JournalD_HT + EndOfLine + ("" + CarTab + "  " + TabSymb(
SymbEff) + CarTab + SymbEff_HT)
JournalD_HT = JournalD_HT + EndOfLine + ("" + CarTab + "  " + TabSymb(
SymbCopy) + CarTab + SymbCopy_HT)
JournalD_HT = JournalD_HT + EndOfLine + ("" + CarTab + "  " + TabSymb(
SymbRempl) + CarTab + SymbRempl_HT)
JournalD_HT = JournalD_HT + EndOfLine + ("" + CarTab + "  " + TabSymb(
SymbAncien) + CarTab + SymbAncien_HT)
WinBatch.FenetrInit ' Après avoir lu les Prefs
If BevButtAffLog.Value Then WinLog.Show ' A faire avant WinMain.Show car
cette dernière doit être active pour l'utilisateur
```

```
If BevButtAffRegl.Value Then WinRegl.Visible = True ' Avant Self.Show pour
que la fenêtre principale ait le focus
If BevButtAffBatch.Value Then WinBatch.Visible = True ' Avant Self.Show po
ur que la fenêtre principale ait le focus
Self.Show ' Invisible par défaut pour le Drag&Drop
' App.CheckWinMenu(True) ' Sera appelé dans event Activate
If DrapPrefs Then
  WinInfos.ShowModal
ElseIf not App.DejaDonne Then
  ' WinInfos.TexteInfo.TextSize = 12
  WinInfos.TexteInfo.Text = Btn_FaireDon + EndOfLine + EndOfLine + I
  nfos_Text_Don
  WinInfos.TexteInfo.SelStart = 0
  WinInfos.TexteInfo.SelLength = Len(Btn_FaireDon)
  WinInfos.TexteInfo.SelBold = True
  WinInfos.TexteInfo.SelUnderline = True
  WinInfos.TexteInfo.SelTextSize = WinInfos.TexteInfo.TextSize + 2 ' 16
  ' WinInfos.TexteInfo.SelTextColor =
  WinInfos.TexteInfo.SelLength = 0
  WinInfos.BevButtDejDon.Visible = True
  WinInfos.ShowModal
End If

If App.initProg Then
  Beep
  MsgBox "Error ! Please contact the authors." + EndOfLine + "initProg s
  hould be False!"
End If
FairePopupConfig ' ChangeConfig ne sera pas fait car  initProg  à False
App.initProg = True
PopupConfig.ListIndex = 0 '  On initialise le Popup  A FAIRE AVANT Enable
MenuBoutons et AVEC  initProg à True
' ChangeConfig a été fait ci-dessus car  initProg  à True mais on n'a pas to
uché à ListBoxBatch car ListIndex à 0
#If DebugBuild Then
  If BevButtAffBatch.Value and WinBatch.CheckBoxBatchAuto.Value and
  WinBatch.DrapBatchEnab and (not(WinBatch.TimerBatch.Mode = Timer.
  ModeSingle)) Then MsgBox "Pb Tom, TimerBatch devrait être lancé" ' Je
   le lance dans WinBatch.Open, je préfère
#EndIf
DrapMenu = True ' Car si erreur et que pas à True on ne pourra pas quitter
 (voir event CancelClose)
App.EnableMenuBoutons(True) ' Menu affichés par défaut sauf pour menu
FichierStop mais il faut faire les boutons et checkbox
```

```
        ' If BevButtAffLog.Value Then
        ' WinLog.Show ' Si on veut l'afficher par dessus WinMain, l'écrire ici plutôt q
        ue plus haut
        ' ' App.CheckWinMenu(False) ' Sera appelé dans event Activate
        ' Else
        ' ' App.CheckWinMenu(True) ' Sera appelé dans event Activate
        ' End If

        ' DateTest.TotalSeconds = Floor(DateTest.TotalSeconds / 86400) * 86400 '
         1 jour = 24 h x 3600 s = 86400 s. Met l'heure à 00:00:00 et non à 12:00:
        00, comme dit l'aide RealBasic : seconds since 12:00 AM, January 1, 1904
        ' MsgBox "DateDuJour = " + App.DateDuJour.SQLDateTime + EndOfLine +
        "DateLaunch = " + App.DateLaunch.SQLDateTime + EndOfLine + "Nb jour
        depuis last launch = " + Str(Floor(App.DateDuJour.TotalSeconds / 86400)
        - Floor(App.DateLaunch.TotalSeconds / 86400)) ' + EndOfLine + "DateTest
         = " + DateTest.SQLDateTime
        If App.AutoCheckUpdt and ((Floor(App.DateDuJour.TotalSeconds / 86400)
        - Floor(App.DateLaunch.TotalSeconds / 86400)) > VmajTsLesXj) Then
          AlertVerAjour = False ' Sera mis à Vrai dans Socket_Ver_MaJ , pour ne
          pas avoir d'alerte au lancement si la version est à jour
          Socket_Ver_MaJ.Get(LienSitePerso + Lien_PrgRb + App.NomProg + "_V.
          txt")
        Else
          AlertVerAjour = True
        End If

      End If

    Else
      Beep
      MsgBox "Error !"+EndOfLine+"Please contact the authors."+EndOfLine+EndOfLi
      ne+"WinMain.Open, Impossible to create the Preferences folder."
      DrapMenu = True ' Sinon on ne pourrait jamais quitter à cause du CancelClose
      ' App.EnableMenuBoutons ' Inutile
      Self.Close ' Main  ' Quit ' Il ne faut pas mettre Quit car ... voir note dans Event Cl
      ose de Main
    End If

    ' MsgBox "Fin WinMain Event open"

End Sub
```

WinMain.Resized:

```
Sub Resized()

   If App.InitProg Then
      WinBatch.RePosFenetr(True, False)
      WinRegl.RePosFenetr(True, False)
   End If

End Sub
```

WinMain.ActButtCible:

```
Sub ActButtCible(f_elt as FolderItem)

   Dim e_elt as FolderItem
   Dim rep as Int16

   If f_elt = Nil Then
      If DossierCible = Nil Then
         e_elt = Nil
      ElseIf DossierCible.Parent = Nil Then ' Volume
         e_elt = DossierCible
      Else
         e_elt = DossierCible.Parent
      End If
      f_elt = MyDialogFolder(e_elt, Txt_ChoisirUnDoss) ' SelectFolder ' Retourne  Nil
      si l'utilisateur clique Annuler
   End If

   If not(f_elt = Nil) Then
      ' MsgBox f_elt.Type + EndOfLine + "Parent existe : " + Cstr(not(f_elt.Parent = N
      il))
      If not f_elt.Directory Then ' If f_elt.Alias Then ' On ne peut normalement choisir
      qu'un dossier, mais on peut choisir un alias de dossier sans cible
         Beep
         rep = MyDialogBox(Txt_SelectQueDoss, "", Btn_Ok, "", "", 0)

      ElseIf (not App.AutorisVol) and (f_elt.Parent = Nil) Then ' Ou f_elt.AbsPath_f = f_
      elt.Volume_f.AbsPath_f Then ' Il ne faut pas sélectionner un volume (racine)
         Beep
         rep = MyDialogBox(Txt_PasVolume, "", Btn_Ok, "", "", 0)
```

```
  Else
    e_elt = DefDossTrash(f_elt, False) ' e_elt sert juste en TampElt pour vérifier
    dossier Corbeille ci-dessous
    If (e_elt = Nil) or (not(f_elt.ShellPath = e_elt.ShellPath)) Then ' On vérifie qu'
    on n'a pas sélectionné le dossier Corbeille, mais si le dossier Corbeille n'a
    pu être créé on continue quand même
      f_elt = VerifFolders(f_elt, DossierSource, True)
      If not(f_elt = Nil) Then ' Si  f_elt = Nil  le message d'erreur a été donné
      dans  VerifFolders
        If CheckBoxLierNav.Value Then
          e_elt = LierFolders(DossierCible, f_elt, DossierSource) ' J'aurai
          s pu faire en une seule ligne avec celle ci-dessous mais c'est
          plus clair comme ça et en plus LierFolders peut retourner Nil
          If PasNil_Existe_Alias(e_elt, True) Then
            e_elt = VerifFolders(e_elt, f_elt, False) ' Pas d'alerte car au
            rait été donnée plus haut. Mais je re-vérifie car si  Dossie
            rSource  et  DossierCible  dans même dossier parent on
            peut
            If PasNil_Existe_Alias(e_elt, True) Then '  re-sélectionner l
            e même dossier
              DossierSource = e_elt
            End If
          End If
        End If
        DossierCible = f_elt ' A faire APRES  LierFolders
        ChangeConfig(-1)

        ' Else
        '    Alerte a été donnée dans  VerifFolders
      End If

      ' Else
      '    Alerte a été donnée dans  DefDossTrash
    End If
  End If
End If

' Note : On fait ConvertEncoding car quand j'enregistre puis lis les prefs je convertie
 en App.DefAppEncod pour
'  CfDossSource (Cible) et ces derniers peuvent me donner des comparaisons fausse
s quand je les compare aux
'  StaticTextDoss (Cible) s'il y a des accents (e´ -> é)
' Problème réglé différement avec ma fonction GetFitemAbsPath qui retransforme (é
 -> e´)
```

```
End Sub
```

WinMain.ActButtSource:

```
Sub ActButtSource(f_elt as FolderItem)

  Dim e_elt as FolderItem
  Dim rep as Int16

  If f_elt = Nil Then
    If DossierSource = Nil Then
      e_elt = Nil
    ElseIf DossierSource.Parent = Nil Then ' Volume
      e_elt = DossierSource
    Else
      e_elt = DossierSource.Parent
    End If
    f_elt = MyDialogFolder(e_elt, Txt_ChoisirUnDoss) ' SelectFolder ' Retourne  Nil
    si l'utilisateur clique Annuler
  End If

  If not(f_elt = Nil) Then
    ' MsgBox f_elt.Type + EndOfLine + "Parent existe : " + Cstr(not(f_elt.Parent = N
    il))
    If not f_elt.Directory Then ' If f_elt.Alias Then ' On ne peut normalement choisir
    qu'un dossier, mais on peut choisir un alias de dossier sans cible
      Beep
      rep = MyDialogBox(Txt_SelectQueDoss, "", Btn_Ok, "", "", 0)

    ElseIf (not App.AutorisVol) and (f_elt.Parent = Nil) Then ' Ou f_elt.AbsPath_f = f_
    elt.Volume_f.AbsPath_f Then ' Il ne faut pas sélectionner un volume (racine)
      Beep
      rep = MyDialogBox(Txt_PasVolume, "", Btn_Ok, "", "", 0)

    Else
      e_elt = DefDossTrash(f_elt, False) ' e_elt sert juste en TampElt pour vérifier
      dossier Corbeille ci-dessous
      If (e_elt = Nil) or (not(f_elt.ShellPath = e_elt.ShellPath)) Then ' On vérifie qu'
      on n'a pas sélectionné le dossier Corbeille, mais si le dossier Corbeille n'a
      pu être créé on continue quand même
        f_elt = VerifFolders(f_elt, DossierCible, True)
        If not(f_elt = Nil) Then ' Si  f_elt = Nil  le message d'erreur a été donné
        dans  VerifFolders
```

```
            If CheckBoxLierNav.Value Then
               e_elt = LierFolders(DossierSource, f_elt, DossierCible) ' J'aurai
               s pu faire en une seule ligne avec celle ci-dessous mais c'est
               plus clair comme ça et en plus LierFolders peut retourner Nil
               If PasNil_Existe_Alias(e_elt, True) Then
                  e_elt = VerifFolders(e_elt, f_elt, False) ' Pas d'alerte car au
                  rait été donnée plus haut. Mais je re-vérifie car si  Dossie
                  rSource  et  DossierCible  dans même dossier parent on
                  peut
                  If PasNil_Existe_Alias(e_elt, True) Then '  re-sélectionner l
                  e même dossier
                     DossierCible = e_elt
                  End If
               End If
            End If
            DossierSource = f_elt ' A faire APRES  LierFolders
            ChangeConfig(-1)

            ' Else
            '    Alerte a été donnée dans  VerifFolders
         End If

         ' Else
         '    Alerte a été donnée dans  DefDossTrash
      End If
   End If
End If

' Note : On fait ConvertEncoding car quand j'enregistre puis lis les prefs je convertie
 en App.DefAppEncod pour
'  CfDossSource (Cible) et ces derniers peuvent me donner des comparaisons fausse
s quand je les compare aux
'  StaticTextDoss (Cible) s'il y a des accents (e´ -> é)
' Problème réglé différement avec ma fonction GetFitemAbsPath qui retransforme (é
 -> e´)

End Sub
```

WinMain.ActButtSync:

```
Sub ActButtSync()

   Dim rep as Int16 ' On peut venir ici depuis PushButtSync ou depuis menu  FichierSyn
   chro  ou  FichierStop (donc si LogSync en cours)
```

```
If App.SyncEnCours or App.LogSEnCours Then ' Même chose que dans event KeyDo
wn. Note ' Btn_Sync désactivé si LogSEnCours
   If not ArretUrg Then ' On n'a pas encore appuyé, si pas de test je ferai autant de
    fois ce qui suit que de
      ArretUrg = True '  clics sur  Arrêter  (ou menu  Arrêter ou appuie sur Esc)
      ' Quoique là je ne fais rien d'autres que Zoum mais si un jour …
      If App.Sfx Then
         Zoum.Play ' Pour signifier prise en compte  ArretUrg     On ne l'entend
         s pas si coup de frein juste après, mais des fois
         While Zoum.IsPlaying '                                    que ce soit plu
         s long pour stopper tout
         Wend
      End If
      ' NON, NE PAS faire boucle ci-dessous, sinon cet event (clic sur Bouton Syn
      chro/Arrêter) ne se fait QU'à la fin de la sychro, donc redémarre et arrête a
      ussi sec
      ' While (App.SyncEnCours or App.LogSEnCours) ' Si ça marche !  (NE PAS fai
      re ça car garde la main et le Thread ne se fini pas)
      ' Wend '                                                  Même sans DoEvents
      Pause(10, False) ' Pour éviter 2 appuie trop vif
      ' If App.Sfx Then Coup_de_freins.Play ' A été fait dans le Thread de Synchro
   End If

Else
   DrapBatch = False
   UpdtSyncEnab(False)
   CanvPtitRond.Refresh
   If DrapSyncEnab Then ' Bouton non actif si  DossierSource  et  DossierCible  no
   n des dossiers, mais je vérifie quand même car on a pu éjecter une disquette p
   ar exemple
      ' PushButtSync et FichierSynchro étaient déjà Enabled
      ' MsgBox DossierSource.AbsPath_f + EndOfLine + DossierCible.AbsPath_f

      If DrapBatch Then ' Une BatchSynchro était déjà en cours
         MsgBox "Error ! Please contact the authors." + EndOfLine + "ActButtSy
         nc, DrapBatch should be False."
         DrapBatch = False
      End If
      L_ThreadSync ' Launch ThreadSync (.Run) ' On lance la Synchro dans un Th
      read

   Else
      Beep
```

```
        rep = MyDialogBox(Txt_Select2Doss, "", Btn_Ok, "", "", 0)
        PushButtSync.Enabled = False
        App.EnableMenuBoutons(False) ' Pour FichierSynchro , pas forcément utile
        car le menu se mettra automatiquement à jour dès qu'on l'affichera, mais j
        e préfère qu'il soit à jour

      End If '  DrapSyncEnab

    End If

End Sub
```

WinMain.AffMasqHelpTag:

```
Sub AffMasqHelpTag(CeCtrlNomId as String, CeCtrlHlpTg as String)

    Dim iNbre, Fin_iNbre as Int16 ' Envoyer  RectControl.Name  avec éventuellement son
      Index  (si tableau de RadioButton par exemple) pour ne mettre que lui à jour
    Dim TampNom as String '              Exemple : AffMasqHelpTag(StaticTextNbSpam.Na
    me, "TextDuHelpTag")
    Dim CeCtrl as Control '          ou si index 1 : AffMasqHelpTag(StaticTextNbSpam.Na
    me + "1", "TextDuHelpTag") ' ou  + str(index)
    Dim CeRectCtrl as RectControl
    Static MemRctCtrlNom(-1), MemHlpTgTxt(-1) as String

    Fin_iNbre = ControlCount - 1
    If UBound(MemHlpTgTxt) = -1 Then ' = UBound(MemRctCtrlNom)
      ' MsgBox "On mémorise HelpTag de " + str(Fin_iNbre + 1) + " controls." + End
      OfLine + "CeCtrlNomId = '" + CeCtrlNomId + "'"
      #If DebugBuild Then
        If not(CeCtrlNomId = "") Then MsgBox "Tu t'es gourré Tom, CeCtrlNomId de
        vrait être vide " !" + EndOfLine + "'" + CeCtrlNomId + "'"
      #EndIf
      For iNbre = 0 to Fin_iNbre
        CeCtrl = Self.Control(iNbre)
        ' MsgBox CeCtrl.Name + EndOfLine + str(CeCtrl.Index) + EndOfLine + "Rec
        tControl ? : " + CStr(CeCtrl isa RectControl)
        If CeCtrl isa RectControl Then
          CeRectCtrl = RectControl(CeCtrl) ' Et NON RectControl(iNbre)
          TampNom = CeCtrl.Name
          #If DebugBuild Then
            If (not(CeRectCtrl.Index = CeCtrl.Index) and (CeRectCtrl.Name = T
            ampNom)) Then MsgBox "Bizarre Tom, RectCtrl.Index ≠ Ctrl.Index
             :" + EndOfLine + str(CeRectCtrl.Index) + " ≠ " + str(CeCtrl.Index)
```

```
        + EndOfLine + "RectCtrl.Name ≠ Ctrl.Name :" + EndOfLine + str(C
        eRectCtrl.Name) + " ≠ " + str(TampNom)
      #EndIf
      If CeCtrl.Index > -1 Then TampNom = TampNom + str(CeCtrl.Index) '
      ATTENTION Int32, si pas un tableau de control alors = -2147483648
      ' MsgBox " id = " + str(CeCtrl.Index) + " = " + str(CeRectCtrl.Index) +
      EndOfLine + TampNom
      MemRctCtrlNom.Append TampNom
      MemHlpTgTxt.Append CeRectCtrl.HelpTag ' On mémorise qu'il y en ait
       ou non
      If not App.AffHelpTag Then CeRectCtrl.HelpTag = ""
      ' Si à Vrai alors l'helptag reste à sa valeur
    End If ' RectControl
  Next iNbre

Else ' On les a déjà mémorisés
  ' MsgBox "Nbre de Controls : " + str(Fin_iNbre + 1) + EndOfLine + "Nbre de Rec
  tControls : " + str(UBound(MemHlpTgTxt) + 1) + EndOfLine + "CeCtrlNomId = '
  " + CeCtrlNomId + "'"
  If CeCtrlNomId = "" Then ' On applique à tous
    For iNbre = 0 to Fin_iNbre ' Pas forcément = à  UBound(MemHlpTgTxt)  car
     tous ne sont pas RectControl, mais QUE Control
      CeCtrl = Self.Control(iNbre)
      ' MsgBox CeCtrl.Name + EndOfLine + str(CeCtrl.Index) + EndOfLine +
      "RectControl ? : " + CStr(CeCtrl isa RectControl)
      If CeCtrl isa RectControl Then
        CeRectCtrl = RectControl(CeCtrl) ' Et NON RectControl(iNbre)
        ' MsgBox str(CeRectCtrl.Index) + " = " + EndOfLine + str(CeCtrl.In
        dex) + EndOfLine + CeRectCtrl.Name + EndOfLine + CeRectCtrl.H
        elpTag
        If App.AffHelpTag Then
          CeRectCtrl.HelpTag = MemHlpTgTxt(iNbre) ' S'il n'y en avait p
          as ça reste à ""
        Else
          CeRectCtrl.HelpTag = ""
        End If
      End If ' RectControl
    Next iNbre

  Else ' On applique qu'au control indiqué
    iNbre = MemRctCtrlNom.IndexOf(CeCtrlNomId)
    If iNbre = -1 Then
      MsgBox "Error ! Please contact the authors, AffMasqHelpTag :" + EndO
      fLine + CeCtrlNomId + EndOfLine + "iNbre = -1"
```

```
      Else
        CeCtrl = Self.Control(iNbre)
        ' MsgBox CeCtrlNomId + EndOfLine + "iNbre = " + str(iNbre)
        ' Forcément : If CeCtrl isa RectControl Then
        CeRectCtrl = RectControl(CeCtrl) ' Et NON RectControl(iNbre)
        MemHlpTgTxt(iNbre) = CeCtrlHlpTg ' On met à jour la mémoire du hel
        ptag
        If App.AffHelpTag Then
          CeRectCtrl.HelpTag = CeCtrlHlpTg ' On met à jour le helptag lui-
          même
        Else
          If not(CeRectCtrl.HelpTag = "") Then MsgBox  "Error ! Please contac
          t the authors, AffMasqHelpTag :" + EndOfLine + CeCtrlNomId + "'
          helptag should be empty " :" + EndOfLine + """ + CeRectCtrl.HelpT
          ag + """
        End If
      End If

    End If ' On applique à tous ou au contrôle indiqué  CeCtrlNomId

  End If ' UBound(MemHlpTgTxt) = -1

End Sub
```

WinMain.ChangeConfig:

```
Sub ChangeConfig(NoConfig as Int16)

  Dim iLigne, FinLigne as Int16

  App.initProg = False ' Pour ne pas faire les tests à l'intérieur des  Events Action  des
   boutons CheckBox etc.
  ' NON If App.SyncEnCours Then ThreadSync.Suspend ' C'est que forcément on l'app
  elle depuis ThreadSync donc ce code se déroule comme s'il était dans ThreadSync
  ' MsgBox "Config(" + str(NoConfig) + ") : " + CfNomConfig(NoConfig)
  If BevButtAffBatch.Value and App.SelLBoxBatch Then ' Voir notes plus bas
    FinLigne = UBound(CfNomConfig) - 1 ' Ou n'importe quel autre, Ligne 0 corres
    pond à réglage 1 puisque réglage 0 est le réglage Dernière Config
    For iLigne = 0 to FinLigne
      WinBatch.ListBoxBatch.Selected(iLigne) = False
    Next iLigne
  End If
```

```
If NoConfig = -1 Then
   If not(PopupConfig.ListIndex = 0) Then
      NomDerConfig = PopupConfig.Text ' On le mémorise car il sera utilisé co
      mme nom par défaut pour l'enregistrement d'une config
      PopupConfig.ListIndex = 0 ' On remet simplement le PopupMenu à pointer
       sur les derniers réglages
      If App.Sfx Then
         Bop.Play
         While Bop.IsPlaying
         Wend
      End If
   End If

Else ' On remet à jour toute la configuration (Dossiers, CheckBox etc.)
   '  CfNomConfig(NoConfig) ' Rien à Faire le PopupConfig  est censé être comme i
   l faut
   If CfDossSource(NoConfig) = Txt_Attente Then
      DossierSource = Nil
   Else
      DossierSource = GetFitemAbsPath(CfDossSource(NoConfig), False)
   End If
   If CfDossCible(NoConfig) = Txt_Attente Then
      DossierCible = Nil
   Else
      DossierCible = GetFitemAbsPath(CfDossCible(NoConfig), False)
   End If
   ' @ For iLigne = 0 to 2 ' Les autres sont à faux au lancement de l'appli mais pas
    si charge Config
   ' @ RadioButtSyncMode(iLigne).Value = (iLigne = CfModeSync(NoConfig))
   ' @ Next iLigne
   ' Il est inutile de mettre les autres à Faux comme ci-dessus, c'est automatique
   quand on en met un autre à Vrai
   RadioButtSyncMode(CfModeSync(NoConfig)).Value = True
   CheckBoxGererDate.Value = CfGererDate(NoConfig)
   CheckBoxRemplRecent.Value = CfRemplRecent(NoConfig)
   CheckBoxTrashIfBegin.Value = CfTrashIfBegin(NoConfig)
   EditFieldTrashBegin.Text = CfTextTrash(NoConfig)
   EditFieldTrashBegin.SelStart = Len(EditFieldTrashBegin.Text)
   GTextTrash = CfTextTrash(NoConfig)
   CheckBoxIgnorElt.Value = CfIgnorElt(NoConfig)
   CheckBoxPreNiv.Value = CfPremNiv(NoConfig)
   EditFieldIgnorElt.Text = CfIgnorEltList(NoConfig)
   EditFieldIgnorElt.SelStart = Len(EditFieldIgnorElt.Text)
   TabIgnorElts(CfIgnorEltList(NoConfig), True)
```

```
  CheckBoxPasIgnor.Value = CfPasIgnor(NoConfig)
  EditFieldPasIgnor.Text = CfPasIgnorList(NoConfig)
  EditFieldPasIgnor.SelStart = Len(EditFieldPasIgnor.Text)
  TabIgnorElts(CfPasIgnorList(NoConfig), False)
  CheckBoxSyncIc.Value = CfSyncIcones(NoConfig)
  PopupBarOut.ListIndex = CfBarOutils(NoConfig)
  CheckBoxSimu.Value = CfSimul(NoConfig)
  WinRegl.EdFieldMargeTps.Text = Format(CfMargeTps(NoConfig), F_MBillionSsE
  sp)
  WinRegl.EdFieldMargeTps.SelStart = Len(WinRegl.EdFieldMargeTps.Text)
  MemMargeTps = CfMargeTps(NoConfig)
  WinRegl.CheckBoxDossPack.Value = CfDossPackFich(NoConfig)
  WinRegl.CheckBoxDossExt.Value = CfDossExtFich(NoConfig)
  WinRegl.CheckBoxUseShell.Value = CfUseShell(NoConfig)
  WinRegl.TextFieldCpShell.Text = CfCdeCpShell(NoConfig)
  WinRegl.CheckBoxCopyRBerr.Value = CfCopyRBerr(NoConfig)
  ' ... = CfBatch(NoConfig) ' Il n'y a rien à mettre à jour

  If not(NoConfig = PopupConfig.ListIndex) Then MsgBox "Error ! Please contact t
  he authors. ChangeConfig" + EndOfLine + str(NoConfig) + " ≠ " + str(PopupCo
  nfig.ListIndex)
  If NoConfig = 0 Then
    NomDerConfig = ""
  Else
    NomDerConfig = PopupConfig.Text ' Il sera utilisé comme nom par défaut
    pour l'enregistrement d'une config
    If BevButtAffBatch.Value and App.SelLBoxBatch Then WinBatch.ListBoxBatch
    .Selected(NoConfig - 1) = True
    ' Si  App.SelLBoxBatch  à False c'est qu'on vient de ListBoxBatch donc inutil
    e de resélectionner ligne (et en plus ça déconne car décale de 2 au lieu de
    1)
  End If

  If App.AlertReglDef and (not BevButtAffRegl.Value) Then VrifAlertReglDef
End If

App.EnableBtnsA
' Plus bas désormais car test  initProg : CanvSyncMode.Refresh

UpdtSyncEnab(True) ' On a affiché les StaticText dans  UpdtSyncEnab  car (True)
If not DrapBatch Then VmemeNomVol ' Sinon l'alerte bloquerait la synchro automati
que
PushButtSync.Enabled = DrapSyncEnab
App.EnableMenuBoutons(False) ' Pour FichierSynchro , pas forcément utile car le me
nu se mettra automatiquement à jour dès qu'on l'affichera, mais je préfère qu'il soit
```

```
à jour
'                    En plus ça fair merdait ma vérification c-dessus lors du prochain
appel de  CaptSyncSimu  avec  DrapTestEnab  à True

  CanvPtitRond.Refresh
  CanvSyncMode.Refresh
  App.initProg = True
  ' NON If App.SyncEnCours Then ThreadSync.Resume ' Voir note au début

End Sub
```

WinMain.DefDossTrash:

```
Private Function DefDossTrash(CeDoss as FolderItem, DrapDefTrash as Boolean) As Folde
rItem

  Dim CeDossTrash, CeDossVol as FolderItem
  Dim rep as Int16
  Dim TampText as String
  Dim DrapErr as Boolean

  If App.DelTrash Then
    CeDossTrash = SpecialFolder.Trash ' Avant deprecated CeDoss.TrashFolder ' Se
    ra sur le même volume si disque non réseau, donc disque local ou clef USB, etc
    .  Voir version 1.7.4
    ' MsgBox "DefDossTrash, App.AmovTrash = " + str(App.AmovTrash) + EndOfLi
    ne + "CeDoss Vol : " + CeDoss.AbsPath_f + EndOfLine + "CeDossTrash Vol : "
    + CeDossTrash.AbsPath_f

    If CeDoss.Volume_f.URLPath = CeDossTrash.Volume_f.URLPath Then ' On est su
    r le disque de démarrage puisque la corbeille utilisateur est sur le même volum
    e
      ' MsgBox "Cas classique, on prend la corbeille du fichier (même volume) vi
      a l'instruction SpecialFolder.Trash"
      DrapErr = not(PasNil_Existe_Alias(CeDossTrash, True))

    Else ' La corbeille n'est pas sur le même volume donc Cas d'un disque amovible
      ' MsgBox "Cas d'un disque amovible"
      Select Case App.AmovTrash
      Case 0 ' Dossier à la racine du volume dont le nom a été choisi dans les Pr
      efs
        CeDossVol = CeDoss.Volume_f
        CeDossTrash = CeDossVol.Child(App.NomDossHistTrash)
```

```
        If DrapDefTrash Then
          If not CeDossTrash.Exists Then CeDossTrash.CreateAsFolder
          DrapErr = not(PasNil_Existe_Alias(CeDossTrash, True))
        Else
          DrapErr = False ' On s'en fout qu'on puisse le créer ou non, on veu
          t juste vérifier qu'on n'a pas sélectionné le dossier Corbeille
        End If

      Case 1 ' Corbeille utilisateur
        ' Rien, ça reste à CeDossTrash = SpecialFolder.Trash
        If DrapDefTrash Then
          DrapErr = not(PasNil_Existe_Alias(CeDossTrash, True))
        Else
          DrapErr = False ' On s'en fout qu'il existe ou non, on veut juste vé
          rifier qu'on n'a pas sélectionné le dossier Corbeille
        End If

      Case 2 ' On efface directement les éléments sur volume amovible
        CeDossTrash = Nil
        DrapErr = False

      Else
        MsgBox "Error ! Please contact the authors." + EndOfLine + "DefDossT
        rash, App.AmovTrash = " + str(App.AmovTrash)
        DrapErr = True

      End Select

    End If

Else
    CeDossTrash = Nil
    DrapErr = False

End If

If DrapDefTrash Then ' On a lancé la synchro, on vient de InitSynchro (donc du Thre
ad)
    rep = 1 ' icône Attention
    TampText = Btn_Arreter
Else
    rep = 0 ' icône tête parle
    TampText = Btn_Ok
End If
```

```
  If DrapErr Then
    Beep
    rep = MyDialogBox(Txt_DefDossTrash_impo, "", TampText, "", "", rep)
  ElseIf not(CeDossTrash = Nil) Then
    If CeDoss.ShellPath = CeDossTrash.ShellPath Then
      Beep
      rep = MyDialogBox(Txt_PasDossTrash, "", TampText, "", "", rep)
      ' NON Justement, car sert au test en retour : CeDossTrash = Nil
      DrapErr = True
    End If
  End If
  If DrapDefTrash Then ArretUrg = ArretUrg or DrapErr ' Comme la synchro est lancée
   on gère ArretUrg

  ' MsgBox "On a choisi pour le dossier : " + CeDoss.AbsPath_f + EndOfLine + "La cor
  beille : " + CeDossTrash.AbsPath_f
  Return CeDossTrash

End Function
```

WinMain.EcrireJournal:

```
Private Sub EcrireJournal(Lcol1 as String, Lcol3 as String, Lcol4 as String, Lcol5 as String,
Lcol6 as String, Lcol7 as String)

  Dim DerLign as Int32 ' Pas UInt16 si Ubound -1 car vide
  ' Parce qu'on ne peut pas ajouter de ligne à un tableau à multi dimension, donc je c
  rée un tableau par colonne (par dimension)

  ' ThreadSync.Sleep 100, False ' Pour test

  App.JournCol1.Append Lcol1 ' Opération à effectuer (si en mode simu sinon elle a d
  éjà été effectuée)

  ' App.JournCol2.Append Ubound(App.JournCol1) + 1 ' Indice n° de ligne

  App.JournCol3.Append Lcol3 ' Indication opération réalisée ou non, si erreur etc.

  App.JournCol4.Append Lcol4 ' Extension ou symbole Dossier ou Alias

  If Lcol5 = "" Then
    App.JournCol5b.Append ""
    App.JournCol5.Append ""
```

```
Else
   #If DebugBuild Then
      If not(DossSrcPath = Left(Lcol5, LgPathSrc - 1)) Then MsgBox "Problème To
      m, dans EcrireJournal, devraient être égaux :" + EndOfLine + DossSrcPath
      + EndOfLine + Left(Lcol5, LgPathSrc - 1)
   #EndIf
   Lcol5 = Mid(Lcol5, LgPathSrc)
   App.JournCol5b.Append DossSrcPath ' Pour les Batch Sync car il change
   App.JournCol5.Append Lcol5 ' Path Source
End If

App.JournCol6.Append Lcol6 ' Indication sens copie

If Lcol7 = "" Then
   App.JournCol7b.Append ""
   App.JournCol7.Append ""
Else
   #If DebugBuild Then
      If not(DossCibPath = Left(Lcol7, LgPathCib - 1)) Then MsgBox "Problème To
      m, dans EcrireJournal, devraient être égaux :" + EndOfLine + DossCibPath
      + EndOfLine + Left(Lcol7, LgPathCib - 1)
   #EndIf
   Lcol7 = Mid(Lcol7, LgPathCib)
   App.JournCol7b.Append DossCibPath ' Pour les Batch Sync car il change
   App.JournCol7.Append Lcol7 ' Path Cible
End If

DerLign = UBound(App.JournCol1) ' Ou de n'importe quel autre

If BevButtAffLog.Value Then
   WinLog.FaireLigne(Lcol1, DerLign, Lcol3, Lcol4, Lcol5, Lcol6, Lcol7)
   '  Lcol1 sera CheckBox   Lcol2 = Ubound(Journal) est le n° d'opération et +1 car
    quand on commence Ubound = -1 et première opération est indice 0
   App.DoEvents ' Pour afficher l'écriture ci-dessus, et laisse plus de temps pour é
   ventuel clic sur bouton Arrêter
   WinLog.ListBoxLog.ScrollPosition = DerLign ' + 10
   WinLog.ListBoxLog.Refresh ' Semble inutile mais je le fais quand même car dan
   s Synchro, quand j'écris dans EditFieldPath je suis obligé de le Refresh
   ' WinLog.UpdateNow ' J'ai fait Refresh ci-dessus
End If ' Je fais le  App.DoEvents  que journal affiché ou non dès qu'un fichier est cop
ié, le temps perdu est négigeable par rapport au temps de la copie
App.DoEvents ' Sinon la fenêtre n'est pas rafraichie et le scroll se fait mal, et l'évent
uel clic du bouton Arrêter ne se fait pas non plus
' ThreadSync.Sleep 1, False ' En fait c'est inutile, le refresh se fait
```

```
  ' Note : Avant je faisais ici, dans cette procédure, le test si item à Nil alors son Path
  est "" mais désormais j'ai besoin de mémoriser AbsPath AVANT de le renommer
  '  avec  App.DeleteFileOrFolder , normalement je sais quand j'envoie des Nil mais au
   cas où … Pour pas faire planter Synchro

End Sub
```

WinMain.EffaceReglage:

```
Private Sub EffaceReglage(NoConfig as Int16)

  If (not(PopupConfig.Text = CfNomConfig(NoConfig))) or (NoConfig = 0) Then
    ' C'est normalement impossible mais je vérifie pour ne pas faire tout planter le
    bazar des fois que …
    Beep
    MsgBox "Error ! Please contact the authors." + EndOfLine + PopupConfig.Text
    + " ≠ " + CfNomConfig(NoConfig) + "   index = " + str(NoConfig)

  Else
    CfNomConfig.Remove NoConfig
    CfDossSource.Remove NoConfig
    CfDossCible.Remove NoConfig
    CfModeSync.Remove NoConfig
    CfGererDate.Remove NoConfig
    CfRemplRecent.Remove NoConfig
    CfTrashIfBegin.Remove NoConfig
    CfTextTrash.Remove NoConfig
    CfIgnorElt.Remove NoConfig
    CfPremNiv.Remove NoConfig
    CfIgnorEltList.Remove NoConfig
    CfPasIgnor.Remove NoConfig
    CfPasIgnorList.Remove NoConfig
    CfSyncIcones.Remove NoConfig
    CfBarOutils.Remove NoConfig
    CfSimul.Remove NoConfig
    CfMargeTps.Remove NoConfig
    CfDossPackFich.Remove NoConfig
    CfDossExtFich.Remove NoConfig
    CfUseShell.Remove NoConfig
    CfCdeCpShell.Remove NoConfig
    CfCopyRBerr.Remove NoConfig
    CfBatch.Remove NoConfig
```

```
  End If

End Sub
```

WinMain.FairePopupConfig:

```
Sub FairePopupConfig()

  Dim iLigne, Fin_CfNconfig as Int16

  If App.initProg Then ' ATTENTION, mettre  App.initProg à False  AVANT d'appeler ce
  tte procédure
    MsgBox "Error ! Please contact the authors." + EndOfLine + "FairePopupConfig,
    initProg should be False."
    App.initProg = False
  End If

  PopupConfig.DeleteAllRows
  Fin_CfNconfig = UBound(CfNomConfig)
  For iLigne = 0 To Fin_CfNconfig ' Ou n'importe quel autre
    PopupConfig.AddRow CfNomConfig(iLigne)
  Next iLigne

End Sub
```

WinMain.InitJournalSync:

```
Private Sub InitJournalSync()

  Redim App.JournCol1(-1)
  ' Pour App.JournCol2 je prend UBound()
  Redim App.JournCol3(-1)
  Redim App.JournCol4(-1)
  Redim App.JournCol5(-1)
  Redim App.JournCol6(-1)
  Redim App.JournCol7(-1)
  ReDim App.JournCol5b(-1)
  ReDim App.JournCol7b(-1)

  ReDim App.LogEltSrc(-1)
```

```
  ReDim App.LogEltCib(-1)
  ReDim App.LogDossTrash(-1) ' On mémorise aussi car si BatchSync, il peut y avoir p
  lusieurs disques différents, donc différentes corbeilles
  ReDim App.LogUseShell(-1) ' Idem, ce réglage peut être différent en BatchSync
  ReDim App.LogCdeCpShell(-1) ' Idem, ce réglage peut être différent en BatchSync
  ReDim App.LogCopyRBerr(-1) ' Idem, ce réglage peut être différent en BatchSync
  ' Je mémorise le fichier et non le dossier pour les remplacements car sinon ça merd
  e dans ThreadLogSync quand je vérifie si le Path correspond au texte de la ListBox

  DateSync = New Date ' Des fois que l'Appli tourne jour et nuit et qu'on est changé d
  e jour depuis lancement
  #If TargetWin32 Then
     Dim vTtVar as Variant
     vTtVar = DateSync.SQLDateTime ' Enlève les chiffres après la virgule, on pourra
     it faire  Round(), car sous Windows il y a des chiffres après la virgule
     DateSync = vTtVar.DateValue '  et ça merde quand compare date exactement (v
     oir Neuronyx)
     ' MsgBox "DateSync : " + DateSync.SQLDateTime + EndOfLine + Format(DateSy
     nc.TotalSeconds, "00000000000000000000.000000000")
  #EndIf

  JournalDd = (Txt_Attente + CarTab + "" + CarTab + DateSync.ShortDate + " : ") ' Pui
  s Nombre d'Erreurs (= 0 au départ)
  App.JourDcoulPos(0) = 0 ' Rien fait, on ne touchera pas aux couleurs du texte
  App.JourDcoulPos(1) = Len(Txt_Attente) ' Couleur ci-dessus de 0 à la fin du 1er mo
  t
  App.JourDcoulPos(2) = 0 ' Rien fait, on ne touchera pas aux couleurs du texte
  App.JourDcoulPos(3) = Len(JournalDd) ' Couleur ci-dessus de cette position
  App.JourDcoulPos(4) = 0 '   à la fin des mots suivants "Att nb erreurs" (mais là il n'y
  en a pas)
  ' JournalDf  ne change pas
  ' Mettre la même chose dans ThreadCopy.Run (avec  NbErrRemplAnc  à la place de
  0)
  ' Pourquoi je faisais comme ci-dessous ????  (car ce n'est pas ça en format Anglais)
  ' JournalD.Append str(DateSync.Day) + "/" + Format(DateSync.Month, "00") + "/" +
  str(DateSync.Year)

  ' A faire juste après appel de cette procédure InitJournalSync, remettre ordre tri par
  défaut avec AfficherJournal

End Sub
```

WinMain.InitSynchro:

```
Private Sub InitSynchro(DrapInitTt as Boolean)

  ZTestA ' Je teste que je n'ai pas fait de connerie en remettant les variables  Mem...
  à leur valeur

  If not CheckBoxTrashIfBegin.Value Then GTextTrash = "" ' On utilise  GTextTrash  le
  temps de la  Synchro
  '  On utilisera  Len(GTextTrash)  pour savoir si on l'utilise ou non (pas utilisé si "")

  If not CheckBoxIgnorElt.Value Then
     ReDim MemIgnorEltNom(-1) ' On utilise  MemIgnorEltNom
     ReDim MemIgnorEltExt(-1) '  et  MemIgnorEltExt  le temps de la  Synchro
     '  MemIgnorElt.IndexOf("abcd")  retourne -1 si  MemIgnorElt  est vide
  End If

  If not CheckBoxPasIgnor.Value Then ReDim MemPasIgnor(-1) ' On utilise  MemPasIg
  nor  le temps de la  Synchro

  If DrapInitTt Then ' Je revérifie (en plus de définir) le dossier Corbeille car on a pu c
  hanger les Prefs entre le choix des dossiers et le lancement de la synchro
     DossSrcTrash = DefDossTrash(DossierSource, True) ' Alerte donnée dedans
     DossCibTrash = DefDossTrash(DossierCible, True) '  et ArretUrg  mis à vrai ded
     ans (si problème)
  Else '  DrapSyncEnab  est False
     DossSrcTrash = Nil
     DossCibTrash = Nil
  End If

  If App.RelPathLog and DrapInitTt Then ' On gardera les derniers : puisqu'on comme
  ncera à ces : avec Mid (voir EcrireJournal)
     ' DossSrcPath , LgPathSrc , DossCibPath  et  LgPathCib  ont été mis à leur valeur
      dans UpdtSyncEnab
     If not((DossSrcPath = DossierSource.AbsPath_f) and (DossCibPath = DossierCibl
     e.AbsPath_f) and (LgPathSrc = Len(DossSrcPath)) and (LgPathCib = Len(DossCib
     Path))) Then
        MsgBox "Error ! Please contact the authors." + EndOfLine + "InitSynchro, D
        ossSrcPath , DossCibPath , LgPathSrc and LgPathCib are not to their good v
        alue !  Should have benn done in UpdtSyncEnab !"
        DossSrcPath = DossierSource.AbsPath_f
        DossCibPath = DossierCible.AbsPath_f
        LgPathSrc = Len(DossierSource.AbsPath_f)
        LgPathCib = Len(DossierCible.AbsPath_f)
     End If
     DossSrcPath = Left(DossSrcPath, LgPathSrc - 1) ' On enlève le  :  à la fin car on l
     e laisse au début des relatifs paths
```

```
    DossCibPath = Left(DossCibPath, LgPathCib - 1) ' Idem
  Else
    DossSrcPath = "" ' Comme ça on n'ajoute rien aux paths relatifs
    DossCibPath = "" ' Idem
    LgPathSrc = 1 ' Car dans  EcrireJournal  on écrit Mid(…, LgPath) ce qui donne to
    ute la chaîne
    LgPathCib = 1 ' Idem
  End If

  TicksEvent = Ticks + TicksInt

End Sub
```

WinMain.LierFolders:

```
Private Function LierFolders(CeDossAv as FolderItem, CeDoss as FolderItem, AutreDoss a
s FolderItem) As FolderItem

  Dim LongAv, LongAp, Long as Int16 ' Cette procédure cherche AutreDoss en suivant
   navigation de CeDossAv à CeDoss
  Dim CeDossAvPath, CeDossPath, AutreDossPath, PathDiff, NouvPath as String ' Voir
  VerifFolders

  NouvPath = ""
  If PasNil_Existe_Alias(CeDossAv, True) and PasNil_Existe_Alias(AutreDoss, True) The
  n '  CeDoss  ne peut pas être à Nil, testé avant
    CeDossAvPath = CeDossAv.AbsPath_f
    CeDossPath = CeDoss.AbsPath_f
    AutreDossPath = AutreDoss.AbsPath_f

    If not(CeDossAvPath = CeDossPath) Then ' Même dossier
      LongAv = Len(CeDossAvPath)
      LongAp = Len(CeDossPath)
      Long = Min(LongAv, LongAp)
      If Left(CeDossAvPath, Long) = Left(CeDossPath, Long) Then ' Il s'agit bien d
      u même path, on aurait pu choisir complètement un autre dossier
        If LongAp > LongAv Then ' On est descendu dans l'arborescence du ré
        pertoire
          ' If Mid(CeDossPath, LongAv, 1) = SepAbsPath Then ' Inutile c'est
          obligé
          PathDiff = Mid(CeDossPath, LongAv + 1)
          NouvPath = AutreDossPath + PathDiff ' On rajoute le chemin rajou
          té, sans ajouter 2 fois : au début
```

```
      ' MsgBox "AutreDossPath = " + AutreDossPath + EndOfLine + "Pat
      hDiff = " + PathDiff + EndOfLine + "NouvPath = " + NouvPath

    ElseIf LongAp < LongAv Then ' On est remonté dans l'arborescence du
     répertoire
      ' If Mid(CeDossAvPath, LongAp, 1) = SepAbsPath Then ' Inutile c'e
      st obligé
      PathDiff = Mid(CeDossAvPath, LongAp + 1) ' Chemin retiré sans le
      s : au début
      #If DebugBuild Then
        If not(Len(PathDiff) = (LongAv - LongAp)) Then MsgBox "LierF
        olders, Pb Tom" + EndOfLine + PathDiff + " (" + str(Len(Path
        Diff)) + ") ≠ " + str(LongAv - LongAp)
      #EndIf
      NouvPath = Left(AutreDossPath, Len(AutreDossPath) - Len(PathDif
      f))
      ' MsgBox "AutreDossPath = " + AutreDossPath + EndOfLine + "Pat
      hDiff = " + PathDiff + EndOfLine + "NouvPath = " + NouvPath
      ' MsgBox str(StrComp(AutreDossPath, NouvPath + PathDiff, 1)) +
      EndOfLine + Cstr(AutreDossPath = NouvPath + PathDiff)
      If not(AutreDossPath = NouvPath + PathDiff) Then ' On n'a pas reti
      ré une partie existante
        NouvPath = "" ' On le remet à ""
      End If

      ' Else LongAp = LongAv ' On a resélectionné le même dossier
      '  On ne fait rien, NouvPath reste à ""

    End If

  Else ' On a pu choisir un dossier à coté
    If not((CeDoss.Parent = Nil) or (CeDossAv.Parent = Nil) or (AutreDoss.P
    arent = Nil)) Then
      CeDossAvPath = CeDossAv.Parent.AbsPath_f
      CeDossPath = CeDoss.Parent.AbsPath_f
      AutreDossPath = AutreDoss.Parent.AbsPath_f
      If CeDossPath = CeDossAvPath Then ' Effectivement même dossie
      r parent
        NouvPath = AutreDossPath + CeDoss.Name + SepAbsPath
      End If

    End If

  End If
```

```
    End If
  End If

  If NouvPath = "" Then ' Sinon renvoie le lien du dossier de l'application
    AutreDoss = Nil ' J'aurais pu laisser tel quel, on aurait repris le même dossier d
    ans ActButt
  Else
    AutreDoss = GetFitemAbsPath(NouvPath, False) ' J'aurais pu faire TrueItem et re
    mettre à Nil si Alias
  End If

  Return AutreDoss

End Function
```

WinMain.L_ThreadSync:

```
Sub L_ThreadSync()

  Dim NoConfig, Fin_CfNconfig as Int16

  ' Avant je faisais ça dans le Thread, je préfère le faire avant
  If App.LogSEnCours Then
    MsgBox "Error ! Please contact the authors." + EndOfLine + "ThreadSync, LogSy
    EnCours should be False."
    ' ThreadSync.Kill ' On arrête ici  Pas encore lancé

  ElseIf App.SyncEnCours Then
    MsgBox "Error ! Please contact the authors." + EndOfLine + "ThreadSync, SyncE
    nCours should be False."
    ' Me.Kill ' Je laisse, je ne fais rien

  Else
    App.SyncEnCours = True
    If ArretUrg Then
      #If DebugBuild Then
        MsgBox "A vérifier Tom." + EndOfLine + "ThreadSync, ArretUrg devrait
         être False mais peut être que Arrêt d'urgence fait juste à la fin de la s
        ynchro ?"
      #EndIf
      ArretUrg = False
    End If
    DrapMenu = False
```

```
App.EnableMenuBoutons(True)

RemplReglage(0) ' On sauvegarde comme défaut les réglages actuels

InitJournalSync
If BevButtAffLog.Value Then ' Si Fenêtre Journal affiché, on la réinitialise
   If WinLog.TimerCellCheck.Mode = Timer.ModeSingle Then WinLog.TimerCe
   llCheck.Mode = Timer.ModeOff ' Puisqu'on reset la liste ci-dessous
   WinLog.NbCellsCheck = 0
   WinLog.ListBoxLog.DeleteAllRows
   WinLog.PushButtLogSync.Enabled = False
   WinLog.EditFieldPath.Text = "" ' EditFieldPath_Note
   WinLog.EcrEdFieldLog(JournalDd, "", JournalDf)
   WinLog.AfficherJournal(True) ' Ou False comme  Journal  est vide on n'affic
   he que remettre tri par défaut
End If

ProgWheelSync.Visible = True
' ### Début ancien Thread ThreadTrait.Run ' S'il tournait déjà on aura un bug
NbErrRemplAnc = 0 ' Pas d'erreur ou remplacement d'ancien pour l'instant
If DrapBatch Then
   ' MsgBox "Début BatchSync" ' MsgBox plante dans Thread
   ' On est obligé de faire cette boucle dans le thread sinon ça relancerait un t
   hread du réglage suivant avant de finir le précédent
   ' De la même manière on ferait la fin de Synchro avant d'avoir tout fini si o
   n la plaçait et appelait la fin de ce thread ailleurs
   Fin_CfNconfig = UBound(CfNomConfig)
   For NoConfig = 1 to Fin_CfNconfig ' Ou n'importe quel autre, on démarre à
    1 car réglage 0 est le réglage par défaut
      ' MsgBox "NoConfig = " + str(NoConfig) + "  CfBatch = " + Cstr(CfBatc
      h(NoConfig))
      If CfBatch(NoConfig) Then
         PopupConfig.ListIndex = NoConfig
         ' Pause(180, False) ' Me.Sleep 3000, False ' Pour test, avoir le tem
         ps de voir ce qui change
         InitSynchro(DrapSyncEnab) ' = PushButtSync.Enabled  Si dossiers n
         on valide a été mis à faux dans  ChangeConfig  via  UpdtSyncEnab
         '  ArretUrg  mis à vrai ci-dessus si problème lors de la création du
          dossier corbeille
         If DrapSyncEnab and (not ArretUrg) Then
            Synchro(DossierSource, DossierCible) ' On lance la Synchro de
            s 2 dossiers
         Else ' Pas d'alerte, on écrit une ligne dans le journal
            NbErrRemplAnc = NbErrRemplAnc + 1
```

```
            App.LogEltSrc.Append Nil ' DossSource
            App.LogEltCib.Append Nil ' DossCible
            App.LogDossTrash.Append Nil
            App.LogUseShell.Append False ' Ou True on s'en fout
            App.LogCdeCpShell.Append " " ' N'importe quoi on s'en fout c
            ar inutilé également
            App.LogCopyRBerr.Append False ' Ou True on s'en fout
            EcrireJournal("NeRienFaire", TabSymb(SymbOpNon) + TabSym
            b(SymbOpErrDiv), TabSymb(SymbDoss), " " + TabSymb(Symb
            OpNon) + " " + CfNomConfig(NoConfig) + " " + TabSymb(Sym
            bOpNon) + "  " + CfDossSource(NoConfig), TabSymb(SymbFle
            ches), " " + TabSymb(SymbOpNon) + " " + CfNomConfig(NoCo
            nfig) + " " + TabSymb(SymbOpNon) + "  " + CfDossCible(NoC
            onfig))
            ' NON PushButtSync.Enabled = True ' Sera remis à True au pr
            ochain tour, et si pas de prochain tour reste à False
          End If
        End If
        If ArretUrg Then Exit
      Next NoConfig
      DrapBatch = False ' Pas forcément utile mais je préfére
    Else ' Pas DrapBatch
      If not DrapSyncEnab Then
        MsgBox "Error ! Please contact the authors." + EndOfLine + "ThreadSy
        nc, DrapSyncEnab should be True."
      Else
        ' MsgBox "Début Synchro" ' MsgBox plante dans Thread
      End If
      InitSynchro(True) ' On initialise tout
      If not ArretUrg Then '  ArretUrg  mis à vrai ci-dessus si problème lors de la
      création du dossier Corbeille
        ' MsgBox "DossSrcTrash : " + DossSrcTrash.AbsPath_f + EndOfLine + "
        DossCibTrash : " + DossCibTrash.AbsPath_f
        Synchro(DossierSource, DossierCible) ' On lance la Synchro des 2 dossi
        ers
      End If
    End If ' DrapBatch

    ' SyncEnCours = False ' Je le fais tout en bas
    If ArretUrg Then
      If App.Sfx Then
        Coup_de_freins.Play
        While Coup_de_freins.IsPlaying
        Wend
```

```
    End If
    JournalDd = Txt_Arret
    ' ArretUrg = False ' Fait plus bas des fois qu'on réappuie encore sur Esc le t
    emps de finir ici
    App.JourDcoulPos(0) = 2 ' pb rouge
Else
    If App.Sfx Then
        Bop.Play
        While Bop.IsPlaying
        Wend
    End If
    JournalDd = Txt_fin
    App.JourDcoulPos(0) = 1 ' ok vert
End If
App.JourDcoulPos(1) = Len(JournalDd)
JournalDd = (JournalDd + CarTab + "" + CarTab + DateSync.ShortDate + " : ")
App.JourDcoulPos(3) = Len(JournalDd)
If NbErrRemplAnc = 0 Then
    ' JournalDd ne change pas
    App.JourDcoulPos(2) = 1 ' ok vert
Else
    Pause(10, False) ' Note Pause(60) = 1 s = 1000 ms
    If App.Sfx Then
        RadioBeep_t.Play ' Buzzer.Play
        While RadioBeep_t.IsPlaying ' While Buzzer.IsPlaying
        Wend
    End If
    JournalDd = (JournalDd + Txt_Attention + " : ")
    App.JourDcoulPos(2) = 2 ' pb rouge
End If
App.JourDcoulPos(4) = Len(JournalDd) - App.JourDcoulPos(3)
' JournalDd = (JournalDd + NbErrRemplAnc) ' Mettre la même chose dans InitJo
urnalSync pour DateSync
' JournalDf = (" " + SymbOpErrDiv + SymbAncien) ' Déjà à cette valeur depuis In
itJournalSync

If not CheckBoxTrashIfBegin.Value Then
    ' NON If not(GTextTrash = "") Then MsgBox "Error ! Please contact the auth
    ors." + EndOfLine + "ThreadSync, GTextTrash should be ""."
    '  Car en DrapBatch, il se peut qu'on ne soit pas entré dans InitSynchro si d
    ossier non valide
    GTextTrash = EditFieldTrashBegin.Text ' On remet  GTextTrash  à sa valeu
    r
End If
```

```
If (not CheckBoxIgnorElt.Value) or (CheckBoxIgnorElt.Value and CheckBoxPreNiv
.Value) Then ' On remet  MemIgnorElt  à sa valeur
   ' NON If UBound(MemIgnorEltNom) > -1 Then MsgBox "Error ! Please conta
   ct the authors." + EndOfLine + "ThreadSync, MemIgnorElt should be empty
   ."
   '  Car en DrapBatch, il se peut qu'on ne soit pas entré dans InitSynchro si d
   ossier non valide
   ' End If
   TabIgnorElts(EditFieldIgnorElt.Text, True)
End If
If not CheckBoxPasIgnor.Value Then ' On remet  MemPasIgnor  à sa valeur
   ' NON If UBound(MemPasIgnor) > -1 Then MsgBox "Error ! Please contact t
   he authors." + EndOfLine + "ThreadSync, MemPasIgnor should be empty."
   '  Car en DrapBatch, il se peut qu'on ne soit pas entré dans InitSynchro si d
   ossier non valide
   ' End If
   TabIgnorElts(EditFieldPasIgnor.Text, False)
End If
ZTestA ' Je teste que je n'ai pas fait de connerie en remettant les variables à leu
r valeur d'avant synchro

Pause(10, False) ' Je préfére afin d'être sûr que tout soit fini dans le Finder avan
t de relancer une Synchro car parfois
'  j'ai des Nil Objection quand je reclique  Synchroniser  aussi sec que synchro p
récédente terminée

If App.TampCloseLog Then ' On a demandé WinLog.Close mais on l'a cancélé
   If WinLog.TampClose Then
      WinLog.TampClose = False ' On continue de le canceler puiqu'on l'ouvr
      e ci-dessous, mais normalement  WinLog.TampClose  est resté à False
      MsgBox "Error ! Please contact the authors." + EndOfLine + "ThreadSy
      nc, WinLog.TampClose should be False."
   End If
   App.TampCloseLog = False ' On ne demande plus sa fermeture puisque je
   l'ouvre ci-dessous
End If

' Et à faire AVANT de réinitialiser ArretUrg car on pourrait le remettre à Vrai sin
on
If BevButtAffLog.Value Then ' Si n'était pas déjà affiché, tout ceci sera fait dans
event open de WinLog
   WinLog.EditFieldPath.Text = EditFieldPath_Note ' Car on affichait le path du
    dossier Source en cours de traitement
   ' WinLog.EditFieldPath.Refresh ' Inutile ici
```

```
      WinLog.AfficherJournal(False) ' On ne fait que le trie puisque WinLog était d
      éjà ouverte donc remplie
      WinLog.EcrEdFieldLog(JournalDd, Format(NbErrRemplAnc, F_MBillionAvEsp)
      , JournalDf)
      ' JournalD a changé plus haut
    End If

    ' MsgBox "Synchro terminée, on réactive menus boutons" ' MsgBox plante dans
    Thread
    TimerFinSync.Mode = Timer.ModeSingle ' 1
    ' ### Fin ancien Thread

  End If

End Sub
```

WinMain.RedimWin:

```
Sub RedimWin(DrapInit as Boolean, ScrBarWMVal as Int16)

  Dim iNbre, Fin_iNbre as Int16 ' Voir  AffMasqHelpTag
  Dim CeCtrl as Control
  Dim CeRectCtrl as RectControl
  Static AncScrBarWMVal as Int16

  If DrapInit Then
    #If DebugBuild Then
      MsgBox "Attention Tom, soit du test 'RedimWin' soit tu as changé la largeu
      r de WinMain !" + EndOfLine + EndOfLine + "Initialisation de RedimWin."
    #EndIf
    If ScrBarWMVal < 0 Then
      MsgBox "Error ! Please contact the authors." + EndOfLine + "ScrBarWMVal
      = ScrollBarWM.Maximum shouldn't be < 0 !" + EndOfLine + "ScrBarWMVal
      = " + str(ScrBarWMVal)
      ScrBarWMVal = 0
    End If
    ScrollBarWM.Maximum = ScrBarWMVal ' Dans ce cas c'est  Self.MaxHeight - Self
    .Height ' A placer également dans l'event Resized de WinMain  si jamais je la re
    nd Resizeable
    ' Self.Resizeable = True ' Lecture seulement
    ' Self.LiveResize = True ' Inutile puisque pas Resizeable
    ScrollBarWM.Visible = True
    If not(ScrollBarWM.Value = 0) Then
```

```
      MsgBox "Error ! Please contact the authors." + EndOfLine + "ScrollBarWM.V
      alue should be 0 !" + EndOfLine + "ScrollBarWM.Value = " + str(ScrollBarW
      M.Value)
      ScrollBarWM.Value = 0
    End If
    ScrBarWMVal = 0 ' Pour remettre  AncScrBarWMVal  à 0 en fin de procédure
End If ' DrapInit

Fin_iNbre = ControlCount - 1
For iNbre = 0 to Fin_iNbre
    CeCtrl = Self.Control(iNbre)
    ' MsgBox CeCtrl.Name + EndOfLine + str(CeCtrl.Index) + EndOfLine + "RectCo
    ntrol ? : " + CStr(CeCtrl isa RectControl)
    If CeCtrl isa RectControl Then
      CeRectCtrl = RectControl(CeCtrl) ' Et NON RectControl(iNbre)
      #If DebugBuild Then
        If (not(CeRectCtrl.Index = CeCtrl.Index) and (CeRectCtrl.Name = CeCtrl
        .Name)) Then MsgBox "Bizarre Tom, RectCtrl.Index ≠ Ctrl.Index :" + En
        dOfLine + str(CeRectCtrl.Index) + " ≠ " + str(CeCtrl.Index) + EndOfLin
        e + "RectCtrl.Name ≠ Ctrl.Name :" + EndOfLine + str(CeRectCtrl.Name)
         + " ≠ " + str(CeCtrl.Name)
      #EndIf

      If DrapInit Then ' On réinitialise chaque RectControl pour qu'ils soient align
      és par Top
        If not CeRectCtrl.LockTop Then
          If CeRectCtrl.LockBottom Then CeRectCtrl.LockBottom = False
          CeRectCtrl.LockTop = True
          ' ElseIf CeRectCtrl.LockBottom Then
          ' MsgBox CeCtrl.Name + " est LockTop & LockBottom."
        End If

      Else ' On bouge ScrollBarWM
        ' MsgBox str(ScrBarWMVal)
        If CeRectCtrl.Parent = Nil Then ' Sinon c'est qu'on est dans une GroupB
        ox ou un TabPanel est l'objet sera déplacé avec son Parent
          If not CeRectCtrl.LockBottom Then CeRectCtrl.Top = CeRectCtrl.To
          p + (AncScrBarWMVal - ScrBarWMVal)
        End If

      End If ' DrapInit
    End If ' RectControl
Next iNbre
AncScrBarWMVal = ScrBarWMVal
```

```
End Sub
```

WinMain.RemplReglage:

```
Private Sub RemplReglage(NoConfig as Int16)

   Dim iNbre, TampInt as Int16

   '  CfNomConfig  Rien à faire car on remplacé, le nom reste le même
   CfDossSource(NoConfig) = StTextLire(StTextDossSource.Text)
   CfDossCible(NoConfig) = StTextLire(StTextDossCible.Text)
   TampInt = -1 ' On aura une Nil objection si problème
   For iNbre = 0 to 2
      If RadioButtSyncMode(iNbre).Value Then TampInt = iNbre ' RadioButtSyncMode.
      ListIndex n'existe pas ?!!!
   Next iNbre
   CfModeSync(NoConfig) = TampInt
   CfGererDate(NoConfig) = CheckBoxGererDate.Value
   CfRemplRecent(NoConfig) = CheckBoxRemplRecent.Value
   CfTrashIfBegin(NoConfig) = CheckBoxTrashIfBegin.Value
   CfTextTrash(NoConfig) = GTextTrash ' EditFieldTrashBegin.Text
   CfIgnorElt(NoConfig) = CheckBoxIgnorElt.Value
   CfPremNiv(NoConfig) = CheckBoxPreNiv.Value
   CfIgnorEltList(NoConfig) = TextIgnorElts(True) ' EditFieldIgnorElt.Text SANS l'éventu
   el  SepAbsPath  à la fin
   CfPasIgnor(NoConfig) = CheckBoxPasIgnor.Value
   CfPasIgnorList(NoConfig) = TextIgnorElts(False) ' EditFieldPasIgnor.Text SANS l'éven
   tuel  SepAbsPath  à la fin
   CfSyncIcones(NoConfig) = CheckBoxSyncIc.Value
   CfBarOutils(NoConfig) = PopupBarOut.ListIndex
   CfSimul(NoConfig) = CheckBoxSimu.Value
   CfMargeTps(NoConfig) = MemMargeTps ' WinRegl.EdFieldMargeTps.Text
   CfDossPackFich(NoConfig) = WinRegl.CheckBoxDossPack.Value
   CfDossExtFich(NoConfig) = WinRegl.CheckBoxDossExt.Value
   CfUseShell(NoConfig) = WinRegl.CheckBoxUseShell.Value
   CfCdeCpShell(NoConfig) = WinRegl.TextFieldCpShell.Text
   CfCopyRBerr(NoConfig) = WinRegl.CheckBoxCopyRBerr.Value
   ' CfBatch(NoConfig) = False ' Reste comme il était

End Sub
```

WinMain.StTextEcrire:

```
Private Sub StTextEcrire(CeStaticText as Label, CeDossPath as String)

  Dim iNbre, NbCarCeDossPath, LargStText, TampTxtSize as Int16
  Dim TampPic as Picture ' Avant je faisais  Graphics.TextSize  et  Graphics.StringWidt
  h()  mais c'est Deprecated
  Dim RetPath as String

  LargStText = StTextDossSource.Width
  #If DebugBuild Then
    Dim TestRetPath as String

    TestRetPath = CeDossPath
    If not(LargStText = StTextDossCible.Width) Then MsgBox "EcrireStText : " + str(
    LargStText) + " ≠ " + str(StTextDossCible.Width)
  #EndIf
  LargStText = LargStText - 5 ' Pour être sûr de ne pas avoir un écart et du coup un r
  etour à la ligne avant mon EndOfLine
  NbCarCeDossPath = Len(CeDossPath) '            Note : J'ai agrandi les StaticText Sou
  rce et Cible de 5 vu que je retire 5 ici

  TampPic = New Picture(CeStaticText.Width, CeStaticText.Height, 32) ' Pas du tout ut
  ile de la définir à la taille du StaticText en fait, mais faut bien une taille
  TampTxtSize = TampPic.Graphics.TextSize ' Car il reste à sa nouvelle valeur tout le
  temps, donc si j'utilise Graphics ailleurs dans le programme le texte aura cette taille
  ' MsgBox "Graphics.TextSize = " + str(TampTxtSize)
  TampPic.Graphics.TextSize = DefStTextSize
  ' MsgBox CeDossPath + EndOfLine + "Nb caracts = " + str(NbCarCeDossPath) + End
  OfLine + "GrLenght = " + str(Graphics.StringWidth(CeDossPath)) + " > 2 * ? " + str(L
  argStText)
  If TampPic.Graphics.StringWidth(CeDossPath) > (2 * LargStText) Then ' Il faut desce
  ndre de taille 12 à 9 pour passer à 3 lignes
    CeStaticText.TextSize = DefStTextSize - 3
    TampPic.Graphics.TextSize = DefStTextSize - 3
  Else
    CeStaticText.TextSize = DefStTextSize ' Car on ne sait pas à combien il était av
    ant
    ' Graphics.TextSize = DefStTextSize ' Inchangé
  End If

  iNbre = 2 ' Car - 1 plus bas et il faut prendre au moins un caractère, mais il n'est pa
  s possible que 2 caractères dépassent du cadre
  RetPath = ""
```

```
Do
   If TampPic.Graphics.StringWidth(Left(CeDossPath, iNbre)) > LargStText Then ' O
   n a forcément iNbre ≤ NbCarCeDossPath
      ' MsgBox "av '" + RetPath + "' '" + CeDossPath + "'" + EndOfLine + "GrLeng
      ht = " + str(Graphics.StringWidth(Left(CeDossPath, iNbre))) + " > ? " + str(L
      argStText)
      RetPath = RetPath + Left(CeDossPath, iNbre - 1) + EndOfLine ' - 1 donc il r
      estera forcément au moins 1 caractère
      #If DebugBuild Then
         If not(Mid(CeDossPath, iNbre) = Mid(CeDossPath, iNbre, (NbCarCeDoss
         Path + 1) - iNbre)) Then MsgBox "EcrireStText, devrait être =" + EndOf
         Line + Mid(CeDossPath, iNbre) + EndOfLine + Mid(CeDossPath, iNbre,
         (NbCarCeDossPath + 1) - iNbre)
      #EndIf
      CeDossPath = Mid(CeDossPath, iNbre) ' , (NbCarCeDossPath + 1) - iNbre) '
      Jusqu'à la fin
      NbCarCeDossPath = Len(CeDossPath)
      ' MsgBox "ap '" + RetPath + "' '" + CeDossPath + "'" + EndOfLine + "GrLeng
      ht = " + str(Graphics.StringWidth(CeDossPath)) + " > ? " + str(LargStText)
      If CeDossPath = "" Then MsgBox "Error ! Please contact the authors." + End
      OfLine + "EcrireStText, CeDossPath shouldn't be '' !" ' Soit  NbCarCeDossPa
      th = 0
      iNbre = 2
   Else
      iNbre = iNbre + 1
      If iNbre > NbCarCeDossPath Then
         RetPath = RetPath + CeDossPath
         CeDossPath = ""
      End If
   End If
Loop Until (CeDossPath = "")

#If DebugBuild Then
   If not(StTextLire(RetPath) = TestRetPath) Then MsgBox "EcrireStText Terminé, d
   evrait être =" + EndOfLine + StTextLire(RetPath) + EndOfLine + TestRetPath
#EndIf
TampPic.Graphics.TextSize = TampTxtSize ' Je remet à sa taille par défaut le texte
Graphics

CeStaticText.Text = RetPath ' CeDossPath

' Pour test
' CeDossPath = "MBook HD2:Applications:Utilities:StuffIt:StuffItExpander:Tampon:T
ampôn éè:IMac HD2 ImgDisk.dmg"
```

```
  ' CeDossPath = "MBook HD2:MBook HD1:Applications:Utilities:StuffIt:StuffIt Expand
  er:Tampon:Tampôn éè:IMac HD2 ImgDisk.dmg"
  ' CeDossPath = "MBook HD2:MBook HD1:Applications:Utilities:StuffIt:StuffIt Expand
  er:Tampon:Tampôn éè:IMac HD2 ImgDisk:MBook HD2:MBook HD1:Applications:Utili
  ties:StuffIt:StuffIt Expander:Tampon:Tampôn éè:IMac HD2 ImgDisk.dmg"

  ' Autre méthode plus simple mais s'il y a un espace après la 3ième lettre du Path et
  pas pendant 30 lettres, on se retrouve avec 3 lettres sur la première ligne et tout le
  reste au dessous
  ' If NbCarCeDossPath < 100 Then
  ' CeStaticText.TextSize = DefStTextSize
  ' ' ElseIf LgCeDossPath < 200 Then
  ' ' CeStaticText.TextSize = DefStTextSize - 2
  ' Else
  ' CeStaticText.TextSize = DefStTextSize - 3
  ' End If

End Sub
```

WinMain.StTextLire:

```
Function StTextLire(CeTextStText as String) As String

  Return ReplaceAll(CeTextStText, EndOfLine, "") ' Je préfére faire une méthode que j'u
  tilise pour vérifier le résultat dans  StTextEcrire  comme ça si je modifie
  '  quelque chose plus tard ce sera plus simple

End Function
```

WinMain.Synchro:

```
Private Sub Synchro(DossSource as FolderItem, DossCible as FolderItem)

  Dim IndSource, IndCible, NoDerElt, iNbre, TampInd as Int32 ' Car UInt16 ailleurs mai
  s ici peut-être -1 si pas trouvé avec IndexOf
  Dim LgGTextTrash as Int16
  Dim NomS, NomC, NomSource(-1), NomCible(-1), ExtSource(-1), ExtCible(-1), Nom
  Doss(-1), NomAlias(-1), TampPath, TampOp, TampJ, TampExt, TampText as String '
   TypeSource(-1), TypeCible(-1)
  Dim EltSource(-1), EltCible(-1), EltDoss(-1), EltAlias(-1), TampElt as FolderItem
  Dim IsDossSource(-1), IsDossCible(-1), Drap_eff, Drap_err as Boolean
  Dim DateSource, DateCible as UInt64 ' Avant as Double mais il semble que ça merda
  it aléatoirement ' Pas as Date car j'utilise TotalSeconds pour ajouter MemMargeTps
```

```
'                    Date commence 1/01/1900 qui fait en 2040 -> 51138 jours ->
 x 365 j x 24 h x 3600 s = 1 612 687 968 000 s et UInt32 va jusqu'à 4,294,967,29
5 donc UInt64 bcp plus

If BevButtAffLog.Value Then ' Que pour sous-dossiers, pas pour chaque élément. De
 plus, pas de DoEvent à chaque tour pour permettre de rafraichir
   WinLog.EditFieldPath.Text = DossSource.AbsPath_f ' Il existe alors inutile de tes
   ter
   WinLog.EditFieldPath.Refresh ' Sinon rafraichi seulement quand Thread laisse le
    temps (plus un Thread désormais, mais je laisse quand même)
End If
Drap_err = False ' Sera mis à Vrai si erreur, si impossibilité de lire noms éléments c
ontenus dans dossiers
LgGTextTrash = Len(GTextTrash)
' MsgBox "Nb Source = " + str(UBound(NomSource)) + EndOfLine + "Nb Cible = " +
str(UBound(NomCible))
NoDerElt = DossSource.Count ' Pas besoin de MyCount puisqu'il existe
For IndSource = 1 to NoDerElt ' ATTENTION car Item(0) est le dossier lui-même
   If Ticks > TicksEvent Then
      App.DoEvents
      TicksEvent = Ticks + TicksInt
   End If
   Try ' Si jamais dossier pour lequel on n'a pas d'autorisation ou etc.
      TampElt = DossSource.TrueItem(IndSource)
      NomS = TampElt.Name
      TampExt = TampElt.Extension_f
      ' MsgBox "NomS : " + NomS + EndOfLine + "Path : " + TampElt.AbsPath_f
      + EndOfLine + "Ext : " + TampExt
      If (TampElt.Visible and (not(Left(NomS, 1) = ".")) and (MemIgnorEltNom.Inde
      xOf(NomS) = -1) and (MemIgnorEltExt.IndexOf(TampExt) = -1)) or (MemPa
      sIgnor.IndexOf(NomS) > -1) Then
         If TampElt.Directory Then ' Que package ou non
            If TraitComDoss(TampElt, TampExt) Then
               ' MsgBox "'" + NomS + "' est un dossier."
               NomDoss.Append NomS
               EltDoss.Append TampElt
               ' ExtSource ' Ajouté plus bas
               ' IsDossSource.Append True ' TypeSource ajouté plus bas
            Else ' Fichier (Package)
               ' MsgBox "'" + NomS + "' est un fichier (Package)."
               NomSource.Append NomS
               ExtSource.Append TampExt
               EltSource.Append TampElt
               IsDossSource.Append False ' TypeSource.Append SymbFich
```

```
            End If
          ElseIf TampElt.Alias Then
            ' MsgBox """ + NomS + "' est un alias."
            NomAlias.Append NomS
            EltAlias.Append TampElt
            ' ExtSource ' Ajouté plus bas
            ' IsDossSource.Append False ' TypeSource ajouté plus bas
          Else ' Fichier
            ' MsgBox """ + NomS + "' est un fichier."
            NomSource.Append NomS
            ExtSource.Append TampExt
            EltSource.Append TampElt
            IsDossSource.Append False ' TypeSource.Append SymbFich
          End If
          ' RevealThisItem(TampElt.ShellPath_fAS)
        End If ' Si on ignore ce dossier/fichier je n'écris rien dans le journal, trop c
        hiant et pas utile
      Catch
        Drap_err = True
        Exit
      End Try
    Next IndSource
    NoDerElt = UBound(NomDoss) ' = Ubound(EltDoss)
    For iNbre = 0 to NoDerElt ' On n'entre pas dans la boucle si pas de dossier
      NomSource.Append NomDoss(iNbre)
      ExtSource.Append TabSymb(SymbDoss)
      EltSource.Append EltDoss(iNbre)
      IsDossSource.Append True ' TypeSource.Append SymbDoss
    Next iNbre
    NoDerElt = UBound(NomAlias) ' = Ubound(EltAlias)
    For iNbre = 0 to NoDerElt ' On n'entre pas dans la boucle si pas d'Alias
      NomSource.Append NomAlias(iNbre)
      ExtSource.Append TabSymb(SymbAlias)
      EltSource.Append EltAlias(iNbre)
      IsDossSource.Append False ' TypeSource.Append SymbAlias
    Next iNbre
    ReDim NomDoss(-1)
    ReDim EltDoss(-1)
    ReDim NomAlias(-1)
    ReDim EltAlias(-1)

    ' On met, dans les tableaux Source et Cible, tous les fichiers, puis tous les dossiers,
     puis enfin tous les alias afin de les traiter en dernier (voir Notes)
    ' Ainsi, les alias seront traités en derniers puisqu'on aura traité tous les fichiers puis
     les sous-dossiers avant
```

```
' Va sans doute plus vite de faire comme ça que faire .Insert au lieu de .Append

' On pourrait ne pas faire la boucle sur les éléments Cible si  Drap_err  à Vrai
NoDerElt = DossCible.Count ' Pas besoin de MyCount puisqu'il existe
For IndCible = 1 to NoDerElt ' ATTENTION car Item(0) est le dossier lui-même
   If Ticks > TicksEvent Then
      App.DoEvents
      TicksEvent = Ticks + TicksInt
   End If
   Try ' Si jamais dossier pour lequel on n'a pas d'autorisation ou etc.
      TampElt = DossCible.TrueItem(IndCible)
      NomC = TampElt.Name
      TampExt = TampElt.Extension_f
      ' MsgBox "NomC : " + NomC + EndOfLine + "Path : " + TampElt.AbsPath_f
      + EndOfLine + "Ext : " + TampExt
      If (TampElt.Visible and (not(Left(NomC, 1) = ".")) and (MemIgnorEltNom.Inde
      xOf(NomC) = -1) and (MemIgnorEltExt.IndexOf(TampExt) = -1)) or (MemPa
      sIgnor.IndexOf(NomC) > -1) Then
         If TampElt.Directory Then ' Que package ou non
            If TraitComDoss(TampElt, TampExt) Then
               ' MsgBox "'" + NomC + "' est un dossier."
               NomDoss.Append NomC
               EltDoss.Append TampElt
               ' ExtCible ' Ajouté plus bas
               ' IsDossCible.Append True ' TypeCible ajouté plus bas
            Else ' Fichier (Package)
               ' MsgBox "'" + NomC + "' est un fichier (Package)."
               NomCible.Append NomC
               ExtCible.Append TampExt
               EltCible.Append TampElt
               IsDossCible.Append False ' TypeCible.Append SymbFich
            End If
         ElseIf TampElt.Alias Then
            ' MsgBox "'" + NomC + "' est un alias."
            NomAlias.Append NomC
            EltAlias.Append TampElt
            ' ExtCible ' Ajouté plus bas
            ' IsDossCible.Append False ' TypeCible ajouté plus bas
         Else ' Fichier
            NomCible.Append NomC
            ExtCible.Append TampExt
            EltCible.Append TampElt
            IsDossCible.Append False ' TypeCible.Append SymbFich
         End If
```

```
        ' RevealThisItem(TampElt.ShellPath_fAS)
      End If ' Si on ignore ce dossier/fichier je n'écris rien dans le journal, trop c
      hiant et pas utile
    Catch
      Drap_err = True
      Exit
    End Try
Next IndCible
' On pourrait ne pas faire ces boucles si  Drap_err  à Vrai
NoDerElt = UBound(NomDoss) ' = Ubound(EltDoss)
For iNbre = 0 to NoDerElt ' On n'entre pas dans la boucle si pas de dossier
    NomCible.Append NomDoss(iNbre)
    ExtCible.Append TabSymb(SymbDoss)
    EltCible.Append EltDoss(iNbre)
    IsDossCible.Append True ' TypeCible.Append SymbDoss
Next iNbre
NoDerElt = UBound(NomAlias) ' = Ubound(EltAlias)
For iNbre = 0 to NoDerElt ' On n'entre pas dans la boucle si pas d'Alias
    NomCible.Append NomAlias(iNbre)
    ExtCible.Append TabSymb(SymbAlias)
    EltCible.Append EltAlias(iNbre)
    IsDossCible.Append False ' TypeCible.Append SymbAlias
Next iNbre
ReDim NomDoss(-1) ' On ne s'en sert plus et ça libère de la place pour éviter la Stac
kOverflowException error
ReDim EltDoss(-1)
ReDim NomAlias(-1)
ReDim EltAlias(-1)

' MsgBox "Nb Source = " + str(UBound(NomSource)) + " = " + str(UBound(ExtSource
)) + " = " + str(UBound(EltSource)) + " = " + str(UBound(IsDossSource)) + EndOfLine
 + "Nb Cible = " + str(UBound(NomCible)) + " = " + str(UBound(ExtCible)) + " = " +
str(UBound(EltCible)) + " = " + str(UBound(IsDossCible)) ' TypeSource , TypeCible
If Drap_err Then
    NbErrRemplAnc = NbErrRemplAnc + 1
    ' En mode normal (non Simu), on ne s'occupe pas des LogSrc, LogCib, LogDoss
    Trash et LogCopyRBerr mais on vérifie que même nombre que ligne de Journal
    '  Et en BatchSync, il peut y avoir des réglages en mode normal et d'autres en m
    ode simulé
    App.LogEltSrc.Append DossSource ' Pas Nil pour pouvoir afficher l'élément dan
    s le Log avec un clic dans la liste
    App.LogEltCib.Append DossCible ' Pas Nil pour pouvoir afficher l'élément dans
    le Log avec un clic dans la liste
    App.LogDossTrash.Append Nil
```

```
  App.LogUseShell.Append False ' Ou True on s'en fout
  App.LogCdeCpShell.Append " " ' N'importe quoi on s'en fout car inutilé égaleme
  nt
  App.LogCopyRBerr.Append False ' Ou True on s'en fout
  TampOp = "NeRienFaire" ' On ne fera rien car erreur
  TampJ = TabSymb(SymbOpNon) + TabSymb(SymbOpErrDiv)
  TampExt = TabSymb(SymbDoss)
  EcrireJournal(TampOp, TampJ, TampExt, DossSource.AbsPath_f, TabSymb(Symb
  Fleches), DossCible.AbsPath_f)
  ReDim NomSource(-1) ' On ne fera pas les boucles ci-dessous
  ReDim NomCible(-1) ' On aurait pu mettre Else et sautait tout ça mais ça me dé
  cale toutes mes lignes d'un cran à droite
  ReDim ExtSource(-1)
  ReDim ExtCible(-1)
  ReDim EltSource(-1)
  ReDim EltCible(-1)
  ReDim IsDossSource(-1) ' TypeSource(-1)
  ReDim IsDossCible(-1) ' TypeCible(-1)
End If
If CheckBoxPreNiv.Value and (UBound(MemIgnorEltNom) > -1) Then ReDim MemIgn
orEltNom(-1) ' On ne l'utilise qu'au premier niveau, pas contre  MemIgnorEltExt  est
utilisé à tous
'                                                                    MemP
asIgnor  est également utilisé à tous les niveaux

' Début de la Synchro # # # # # # # # # # # # # # # # # # # # # # # # # # # # #
  # # # # # # # # # # # # # # # # # # # # # #
NoDerElt = UBound(NomSource) ' = UBound(ExtSource) = UBound(EltSource) = UBo
und(IsDossSource) ' TypeSource
For IndSource = 0 to NoDerElt
  If Ticks > TicksEvent Then
    App.DoEvents
    TicksEvent = Ticks + TicksInt
  End If
  NomS = NomSource(IndSource)
  #If DebugBuild Then
    If not(DossSource.TrueChild(NomS).AbsPath_f = EltSource(IndSource).AbsP
    ath_f) Then MsgBox "Synchro , Problème Tom" + EndOfLine + DossSource.
    TrueChild(NomS).AbsPath_f + EndOfLine + EltSource(IndSource).AbsPath_f
  #EndIf
  ' MsgBox "IndSource = " + str(IndSource) + "  NomS : " + NomS + EndOfLine +
  "Path : " + EltSource(IndSource).AbsPath_f + EndOfLine + "Instr(" + GTextTrash
  + ") = " + str(InStr(NomS, GTextTrash))
  TampInd = NomCible.IndexOf(NomS) ' Parce qu'on peut le chercher 3 fois, pour
   gain de temps
```

```
Drap_eff = False
If LgGTextTrash > 0 Then
  If InStr(NomS, GTextTrash) = 1 Then ' NomS de Source commence par G
  TextTrash
    Drap_eff = True
    TampPath = EltSource(IndSource).AbsPath_f ' Parce qu'il risque d'être r
    enommé ci-dessous avant d'être déplacé à la corbeille
    App.LogEltSrc.Append EltSource(IndSource)
    App.LogEltCib.Append Nil
    App.LogDossTrash.Append DossSrcTrash
    App.LogUseShell.Append False ' Ou True on s'en fout
    App.LogCdeCpShell.Append " " ' N'importe quoi on s'en fout car inutilé
     également
    App.LogCopyRBerr.Append False ' Ou True on s'en fout
    If CheckBoxSimu.Value Then
      TampOp = "Effacer_Src"
      TampJ = TabSymb(SymbOpNon)
    Else
      TampOp = "NeRienFaire" ' Car l'opération aura été effectuée ci-de
      ssous
      If App.DeleteFileOrFolder(EltSource(IndSource), DossSrcTrash) The
      n
        TampJ = TabSymb(SymbOpErrDiv)
        NbErrRemplAnc = NbErrRemplAnc + 1
      Else
        TampJ = TabSymb(SymbOpOui)
      End If
    End If
    TampJ = TampJ + TabSymb(SymbEff)
    TampExt = ExtSource(IndSource)
    EcrireJournal(TampOp, TampJ, TampExt, TampPath, TabSymb(SymbFlE
    ffG), "")

    IndCible = TampInd ' = NomCible.IndexOf(NomS) ' Note : Si NomS in
    visible dans DossCible il n'est pas dans NomCible
    ' MsgBox NomS + EndOfLine + "IndCible = " + str(IndCible)
    If IndCible >= 0 Then ' NomS (commençant par GTextTrash ) est aus
    si dans NomCible
      #If DebugBuild Then
        If not(DossCible.TrueChild(NomS).AbsPath_f = EltCible(IndCib
        le).AbsPath_f) Then MsgBox "Synchro , Problème Tom" + End
        OfLine + DossCible.TrueChild(NomS).AbsPath_f + EndOfLine
        + EltCible(IndCible).AbsPath_f
      #EndIf
```

```
    TampPath = EltCible(IndCible).AbsPath_f ' Parce qu'il risque d'être
    renommé ci-dessous avant d'être déplacé à la corbeille
    App.LogEltSrc.Append Nil
    App.LogEltCib.Append EltCible(IndCible)
    App.LogDossTrash.Append DossCibTrash
    App.LogUseShell.Append False ' Ou True on s'en fout
    App.LogCdeCpShell.Append " " ' N'importe quoi on s'en fout car in
    utilé également
    App.LogCopyRBerr.Append False ' Ou True on s'en fout
    If CheckBoxSimu.Value Then
      TampOp = "Effacer_Cib"
      TampJ = TabSymb(SymbOpNon)
    Else
      TampOp = "NeRienFaire" ' Car l'opération aura été effectuée c
      i-dessous
      If App.DeleteFileOrFolder(EltCible(IndCible), DossCibTrash) Th
      en
        TampJ = TabSymb(SymbOpErrDiv)
        NbErrRemplAnc = NbErrRemplAnc + 1
      Else
        TampJ = TabSymb(SymbOpOui)
      End If
    End If
    TampJ = TampJ + TabSymb(SymbEff)
    TampExt = ExtCible(IndCible)
    EcrireJournal(TampOp, TampJ, TampExt, "", TabSymb(SymbFlEffD),
     TampPath)
    NomCible.Remove IndCible ' A été traité
    ExtCible.Remove IndCible
    EltCible.Remove IndCible
    IsDossCible.Remove IndCible ' TypeCible

  Else ' On regarde si  NomS  (ne commençant pas par  GTextTrash ) exi
  ste dans DossCible
    If not DossCible.TrueChild(NomS).Exists Then ' On ne vient ici que
     si  NomS  (commençant par  GTextTrash ) n'existe pas dans  Dos
    sCible  (visible ou non), pas à Nil puisque  DossCible  existe
      NomC = Mid(NomS, LgGTextTrash + 1)
      If not(NomC = "") Then ' On regarde si  NomC = NomS ne com
      mençant pas par  GTextTrash est dans  Cible
        If NomSource.IndexOf(NomC) = -1 Then ' Sinon il a ou il
        sera traité à un autre moment
          IndCible = NomCible.IndexOf(NomC) ' Note : Si  Nom
          C  invisible il n'est pas dans  NomCible
```

```
' MsgBox NomC + EndOfLine + "IndCible = " + str(In
dCible)
If IndCible >= 0 Then '  NomC  est dans  NomCible
  #If DebugBuild Then
    If not(DossCible.TrueChild(NomC).AbsPath_f
     = EltCible(IndCible).AbsPath_f) Then MsgBo
    x "Synchro , Problème Tom" + EndOfLine +
    DossCible.TrueChild(NomC).AbsPath_f + En
    dOfLine + EltCible(IndCible).AbsPath_f
  #EndIf
  TampPath = EltCible(IndCible).AbsPath_f ' Parce
  qu'il risque d'être renommé ci-dessous avant d'
  être déplacé à la corbeille
  App.LogEltSrc.Append Nil
  App.LogEltCib.Append EltCible(IndCible)
  App.LogDossTrash.Append DossCibTrash
  App.LogUseShell.Append False ' Ou True on s'en
   fout
  App.LogCdeCpShell.Append " " ' N'importe quoi
  on s'en fout car inutilé également
  App.LogCopyRBerr.Append False ' Ou True on s'
  en fout
  If CheckBoxSimu.Value Then
    TampOp = "Effacer_Cib"
    TampJ = TabSymb(SymbOpNon)
  Else
    TampOp = "NeRienFaire" ' Car l'opération a
    ura été effectuée ci-dessous
    If App.DeleteFileOrFolder(EltCible(IndCible),
    DossCibTrash) Then
      TampJ = TabSymb(SymbOpErrDiv)
      NbErrRemplAnc = NbErrRemplAnc + 1
    Else
      TampJ = TabSymb(SymbOpOui)
    End If
  End If
  TampJ = TampJ + TabSymb(SymbEff)
  TampExt = ExtCible(IndCible)
  EcrireJournal(TampOp, TampJ, TampExt, "", TabS
  ymb(SymbFlEffD), TampPath)
  NomCible.Remove IndCible ' A été traité
  ExtCible.Remove IndCible
  EltCible.Remove IndCible
  IsDossCible.Remove IndCible ' TypeCible
```

```
      Else ' On ne fait rien
        If DossCible.TrueChild(NomC).Exists Then ' Ce n'
        est pas normal qu'il soit invisible car visible dan
        s  DossSource, pas à Nil puisque  DossCible  exi
        ste
          ' Non If (not DossCible.TrueChild(NomC).Vis
          ible) or (Left(NomC, 1) = ".") Then ' Je revéri
          fie si invisible car si en mode Simu on ne l'a
           pas supprimé réellement
          NbErrRemplAnc = NbErrRemplAnc + 1
          ' En mode normal (non Simu), on ne s'occup
          e pas des LogSrc, LogCib, LogDossTrash et
          LogCopyRBerr mais on vérifie que même no
          mbre que ligne de Journal
          '  Et en BatchSync, il peut y avoir des réglag
          es en mode normal et d'autres en mode sim
          ulé
          App.LogEltSrc.Append Nil
          App.LogEltCib.Append DossCible.TrueChild(
          NomC) ' Pas Nil pour pouvoir afficher l'élém
          ent dans le Log avec un clic dans la liste
          App.LogDossTrash.Append Nil
          App.LogUseShell.Append False ' Ou True on
           s'en fout
          App.LogCdeCpShell.Append " " ' N'importe q
          uoi on s'en fout car inutilé également
          App.LogCopyRBerr.Append False ' Ou True
          on s'en fout
          TampOp = "NeRienFaire" ' On ne fera rien c
          ar erreur
          TampJ = TabSymb(SymbOpNon) + TabSymb
          (SymbOpErrDiv)
          TampExt = DossCible.TrueChild(NomC).Ext
          ension_f
          EcrireJournal(TampOp, TampJ, TampExt, "",
          TabSymb(SymbOpErrDiv), DossCible.TrueCh
          ild(NomC).AbsPath_f)
          ' Non End If ' Revérifie si invisible
        End If

      End If
    End If '  NomS  sans  GTextTrash  existe et a été ou sera t
    raité à un autre moment
End If ' Si  NomC = ""  on ne fait rien (c'est que  NomS = GTex
tTrash )
```

```
      Else '  DossCible.TrueChild(NomS)  existe et ce n'est pas normal q
      u'il soit invisible car visible dans  DossSource
         ' Non If (not DossCible.TrueChild(NomS).Visible) or (Left(Nom
         S, 1) = ".") Then ' Je revérifie si invisible car si en mode Simu
         on ne l'a pas supprimé réellement
         NbErrRemplAnc = NbErrRemplAnc + 1
         ' En mode normal (non Simu), on ne s'occupe pas des LogSrc,
          LogCib, LogDossTrash et LogCopyRBerr mais on vérifie que
         même nombre que ligne de Journal
         '  Et en BatchSync, il peut y avoir des réglages en mode norm
         al et d'autres en mode simulé
         App.LogEltSrc.Append Nil
         App.LogEltCib.Append DossCible.TrueChild(NomS) ' Pas Nil p
         our pouvoir afficher l'élément dans le Log avec un clic dans la
          liste
         App.LogDossTrash.Append Nil
         App.LogUseShell.Append False ' Ou True on s'en fout
         App.LogCdeCpShell.Append " " ' N'importe quoi on s'en fout c
         ar inutilé également
         App.LogCopyRBerr.Append False ' Ou True on s'en fout
         TampOp = "NeRienFaire" ' On ne fera rien car erreur
         TampJ = TabSymb(SymbOpNon) + TabSymb(SymbOpErrDiv)
         TampExt = DossCible.TrueChild(NomS).Extension_f
         EcrireJournal(TampOp, TampJ, TampExt, "", TabSymb(SymbOp
         ErrDiv), DossCible.TrueChild(NomS).AbsPath_f)
         ' Non End If ' Revérifie si invisible

      End If '  DossCible.TrueChild(NomS)  existe

   End If '  NomS  ou  NomC  existent dans  DossCible
End If '  NomS  de  Source  commence par  GTextTrash

If not Drap_eff Then
   NomC = GTextTrash + NomS
   If (TampInd = -1) and (NomSource.IndexOf(NomC) = -1) Then ' Sinon i
   l a ou il sera traité à un autre moment,  TampInd = NomCible.IndexOf(
   NomS)
      IndCible = NomCible.IndexOf(NomC) ' Note : Si  NomC  invisible il
       n'est pas dans  NomCible
      ' MsgBox NomC + EndOfLine + "IndCible = " + str(IndCible)
      If IndCible >= 0 Then '  NomC = GTextTrash + NomS  est dans Ci
      ble
         #If DebugBuild Then
```

```
    If not(DossCible.TrueChild(NomC).AbsPath_f = EltCible(In
    dCible).AbsPath_f) Then MsgBox "Synchro , Problème To
    m" + EndOfLine + DossCible.TrueChild(NomC).AbsPath_f
     + EndOfLine + EltCible(IndCible).AbsPath_f
#EndIf
Drap_eff = True
TampPath = EltSource(IndSource).AbsPath_f ' Parce qu'il risqu
e d'être renommé ci-dessous avant d'être déplacé à la corbeil
le
App.LogEltSrc.Append EltSource(IndSource)
App.LogEltCib.Append Nil
App.LogDossTrash.Append DossSrcTrash
App.LogUseShell.Append False ' Ou True on s'en fout
App.LogCdeCpShell.Append " " ' N'importe quoi on s'en fout c
ar inutilé également
App.LogCopyRBerr.Append False ' Ou True on s'en fout
If CheckBoxSimu.Value Then
    TampOp = "Effacer_Src"
    TampJ = TabSymb(SymbOpNon)
Else
    TampOp = "NeRienFaire" ' Car l'opération aura été effect
    uée ci-dessous
    If App.DeleteFileOrFolder(EltSource(IndSource), DossSrcTr
    ash) Then
        TampJ = TabSymb(SymbOpErrDiv)
        NbErrRemplAnc = NbErrRemplAnc + 1
    Else
        TampJ = TabSymb(SymbOpOui)
    End If
End If
TampJ = TampJ + TabSymb(SymbEff)
TampExt = ExtSource(IndSource)
EcrireJournal(TampOp, TampJ, TampExt, TampPath, TabSymb
(SymbFlEffG), "")

TampPath = EltCible(IndCible).AbsPath_f ' Parce qu'il risque d'
être renommé ci-dessous avant d'être déplacé à la corbeille
App.LogEltSrc.Append Nil
App.LogEltCib.Append EltCible(IndCible)
App.LogDossTrash.Append DossCibTrash
App.LogUseShell.Append False ' Ou True on s'en fout
App.LogCdeCpShell.Append " " ' N'importe quoi on s'en fout c
ar inutilé également
App.LogCopyRBerr.Append False ' Ou True on s'en fout
```

```
    If CheckBoxSimu.Value Then
      TampOp = "Effacer_Cib"
      TampJ = TabSymb(SymbOpNon)
    Else
      TampOp = "NeRienFaire" ' Car l'opération aura été effect
      uée ci-dessous
      If App.DeleteFileOrFolder(EltCible(IndCible), DossCibTras
      h) Then
        TampJ = TabSymb(SymbOpErrDiv)
        NbErrRemplAnc = NbErrRemplAnc + 1
      Else
        TampJ = TabSymb(SymbOpOui)
      End If
    End If
    TampJ = TampJ + TabSymb(SymbEff)
    TampExt = ExtCible(IndCible)
    EcrireJournal(TampOp, TampJ, TampExt, "", TabSymb(SymbFlE
    ffD), TampPath)
    NomCible.Remove IndCible ' A été traité
    ExtCible.Remove IndCible
    EltCible.Remove IndCible
    IsDossCible.Remove IndCible ' TypeCible

  Else ' On ne fait rien
    If DossCible.TrueChild(NomC).Exists Then ' Ce n'est pas norm
    al qu'il soit invisible car visible dans  DossSource
      ' Non If (not DossCible.TrueChild(NomC).Visible) or (Left(
      NomC, 1) = ".") Then ' Je revérifie si invisible car si en mo
      de Simu on ne l'a pas supprimé réellement
      NbErrRemplAnc = NbErrRemplAnc + 1
      ' En mode normal (non Simu), on ne s'occupe pas des Lo
      gSrc, LogCib, LogDossTrash et LogCopyRBerr mais on vé
      rifie que même nombre que ligne de Journal
      '  Et en BatchSync, il peut y avoir des réglages en mode n
      ormal et d'autres en mode simulé
      App.LogEltSrc.Append Nil
      App.LogEltCib.Append DossCible.TrueChild(NomC) ' Pas
      Nil pour pouvoir afficher l'élément dans le Log avec un cl
      ic dans la liste
      App.LogDossTrash.Append Nil
      App.LogUseShell.Append False ' Ou True on s'en fout
      App.LogCdeCpShell.Append " " ' N'importe quoi on s'en f
      out car inutilé également
      App.LogCopyRBerr.Append False ' Ou True on s'en fout
```

```
            TampOp = "NeRienFaire" ' On ne fera rien car erreur
            TampJ = TabSymb(SymbOpNon) + TabSymb(SymbOpErrD
            iv)
            TampExt = DossCible.TrueChild(NomC).Extension_f
            EcrireJournal(TampOp, TampJ, TampExt, "", TabSymb(Sy
            mbOpErrDiv), DossCible.TrueChild(NomC).AbsPath_f)
            ' Non End If ' Revérifie si invisible

          End If

        End If
      End If ' GTextTrash + NomS existe et a été ou sera traité à un autre
      moment
    End If ' Drap_eff
End If ' LgGTextTrash > 0

If not Drap_eff Then ' EltSource(IndSource) n'a pas été effacé
    IndCible = TampInd ' = NomCible.IndexOf(NomS) ' Note : Si NomS invisibl
    e dans DossCible il n'est pas dans NomCible
    ' MsgBox NomS + EndOfLine + "IndCible = " + str(IndCible)
    If IndCible >= 0 Then ' NomS existe dans les 2 dossiers Source et Cible
      #If DebugBuild Then
        If not(DossCible.TrueChild(NomS).AbsPath_f = EltCible(IndCible).A
        bsPath_f) Then MsgBox "Synchro , Problème Tom" + EndOfLine +
        DossCible.TrueChild(NomS).AbsPath_f + EndOfLine + EltCible(Ind
        Cible).AbsPath_f
        ' If ((TypeSource(IndSource) = SymbDoss) and (TypeCible(IndCible)
         = SymbDoss)) Xor (TraitComDoss(EltSource(IndSource), EltSource(
        IndSource).Extension_f) and TraitComDoss(EltCible(IndCible), EltCi
        ble(IndCible).Extension_f)) Then MsgBox "Synchro , Problème Tom
        , avec les packages traités comme des dossiers !"
        If (IsDossSource(IndSource) and IsDossCible(IndCible)) Xor ((EltSou
        rce(IndSource).Directory and TraitComDoss(EltSource(IndSource),
        EltSource(IndSource).Extension_f)) and (EltCible(IndCible).Director
        y and TraitComDoss(EltCible(IndCible), EltCible(IndCible).Extensio
        n_f))) Then MsgBox "Synchro , Problème Tom, avec les packages tr
        aités comme des dossiers !"
      #EndIf
      If IsDossSource(IndSource) and IsDossCible(IndCible) Then ' TraitComD
      oss(EltSource(IndSource), EltSource(IndSource).Extension_f) and TraitC
      omDoss(EltCible(IndCible), EltCible(IndCible).Extension_f) Then
        Try ' Dans certaines versions l'erreur se produit ici, dans d'autres
        ça crée une erreur dans la Method Synchro appelée et ça saute à l
        a fin de cette méthode
          Synchro(EltSource(IndSource), EltCible(IndCible)) ' Recursively
          call this routine passing it the folders
```

```
    Catch ' C'est sûrement une StackOverFlowException. Voir à la fin d
    e cette méthode car on y fait la même chose
      NbErrRemplAnc = NbErrRemplAnc + 1
      ' En mode normal (non Simu), on ne s'occupe pas des LogSrc,
       LogCib, LogDossTrash et LogCopyRBerr mais on vérifie que
      même nombre que ligne de Journal
      '  Et en BatchSync, il peut y avoir des réglages en mode norm
      al et d'autres en mode simulé
      App.LogEltSrc.Append EltSource(IndSource) ' Pas Nil pour pou
      voir afficher l'élément dans le Log avec un clic dans la liste
      App.LogEltCib.Append EltCible(IndCible) ' Pas Nil pour pouvoi
      r afficher l'élément dans le Log avec un clic dans la liste
      App.LogDossTrash.Append Nil
      App.LogUseShell.Append False ' Ou True on s'en fout
      App.LogCdeCpShell.Append " " ' N'importe quoi on s'en fout c
      ar inutilé également
      App.LogCopyRBerr.Append False ' Ou True on s'en fout
      TampOp = "NeRienFaire" ' On ne fera rien car erreur
      TampJ = TabSymb(SymbOpNon) + TabSymb(SymbOpErrDiv)
      TampExt = ExtSource(IndSource)
      TampText = ExtCible(IndCible)
      If not(TampExt = TampText) Then TampExt = TampExt + " "
      + TampText ' Si un fichier remplace un alias par exemple
      EcrireJournal(TampOp, TampJ, TampExt, EltSource(IndSource).
      AbsPath_f, TabSymb(SymbFleches), EltCible(IndCible).AbsPath
      _f)
    End Try

ElseIf CheckBoxGererDate.Value and (not(EltSource(IndSource).Alias an
d EltCible(IndCible).Alias)) Then ' Si les 2 sont des Alias on ne s'occupe
 pas des dates de modif
    ' Si seulement l'un des 2 est un dossier (donc l'autre est un fichier
     sans extension), on fait comme si c'était 2 fichiers
    DateSource = EltSource(IndSource).ModificationDate.TotalSeconds
    DateCible = EltCible(IndCible).ModificationDate.TotalSeconds
    ' Voir notes Sync_Alias, mais j'ai fait plus simple ci-dessus. J'ignor
    e si les 2 sont des Alias
    If DateSource > (DateCible + MemMargeTps) Then ' Fichier de  So
    urce  plus récent que celui de  Cible
      App.LogEltSrc.Append EltSource(IndSource)
      App.LogEltCib.Append EltCible(IndCible) ' .Parent = DossCible
      App.LogDossTrash.Append DossCibTrash
      App.LogUseShell.Append WinRegl.CheckBoxUseShell.Value
      App.LogCdeCpShell.Append WinRegl.TextFieldCpShell.Text
```

```
    App.LogCopyRBerr.Append WinRegl.CheckBoxCopyRBerr.Valu
    e
    If CheckBoxSimu.Value Then
      TampOp = "Rempl_Src->Cib"
      TampJ = TabSymb(SymbOpNon)
    Else
      TampOp = "NeRienFaire" ' Car l'opération aura été effect
      uée ci-dessous
      If App.AScriptCopy(EltSource(IndSource), DossCible, Doss
      CibTrash, True, WinRegl.CheckBoxUseShell.Value, WinReg
      l.TextFieldCpShell.Text, WinRegl.CheckBoxCopyRBerr.Val
      ue) Then ' On remplace  EltCible(IndCible)  par  EltSource(
      IndSource)
        TampJ = TabSymb(SymbOpErrDiv)
        NbErrRemplAnc = NbErrRemplAnc + 1
      Else
        TampJ = TabSymb(SymbOpOui)
      End If
    End If
    TampJ = TampJ + TabSymb(SymbRempl)
    TampExt = ExtSource(IndSource)
    TampText = ExtCible(IndCible)
    If not(TampExt = TampText) Then TampExt = TampExt + " "
    + TampText ' Si un fichier remplace un alias par exemple
    EcrireJournal(TampOp, TampJ, TampExt, EltSource(IndSource).
    AbsPath_f, TabSymb(SymbFlecheD), EltCible(IndCible).AbsPat
    h_f)

  ElseIf DateCible > (DateSource + MemMargeTps) Then ' Fichier de
   Cible  plus récent que celui de  Source
    If RadioButtSyncMode(0).Value Then '  Réciproque
      App.LogEltSrc.Append EltSource(IndSource) ' .Parent = D
      ossSource
      App.LogEltCib.Append EltCible(IndCible)
      App.LogDossTrash.Append DossSrcTrash
      App.LogUseShell.Append WinRegl.CheckBoxUseShell.Valu
      e
      App.LogCdeCpShell.Append WinRegl.TextFieldCpShell.Te
      xt
      App.LogCopyRBerr.Append WinRegl.CheckBoxCopyRBerr.
      Value
      If CheckBoxSimu.Value Then
        TampOp = "Rempl_Cib->Src"
        TampJ = TabSymb(SymbOpNon)
```

```
    Else
      TampOp = "NeRienFaire" ' Car l'opération aura été ef
      fectuée ci-dessous
      If App.AScriptCopy(EltCible(IndCible), DossSource, D
      ossSrcTrash, True, WinRegl.CheckBoxUseShell.Value,
       WinRegl.TextFieldCpShell.Text, WinRegl.CheckBoxC
      opyRBerr.Value) Then ' On remplace  EltSource(IndSo
      urce)  par  EltCible(IndCible)
        TampJ = TabSymb(SymbOpErrDiv)
        NbErrRemplAnc = NbErrRemplAnc + 1
      Else
        TampJ = TabSymb(SymbOpOui)
      End If
    End If
    TampJ = TampJ + TabSymb(SymbRempl)
    TampExt = ExtSource(IndSource)
    TampText = ExtCible(IndCible)
    If not(TampExt = TampText) Then TampExt = TampExt +
     " " + TampText ' Si un fichier remplace un alias par exem
    ple
    EcrireJournal(TampOp, TampJ, TampExt, EltSource(IndSo
    urce).AbsPath_f, TabSymb(SymbFlecheG), EltCible(IndCibl
    e).AbsPath_f)

  Else '  Source vers Cible  ou  Source sur Cible  On remplace un
   fichier plus récent
    NbErrRemplAnc = NbErrRemplAnc + 1
    App.LogEltSrc.Append EltSource(IndSource)
    App.LogEltCib.Append EltCible(IndCible) ' .Parent = Doss
    Cible
    App.LogDossTrash.Append DossCibTrash
    App.LogUseShell.Append WinRegl.CheckBoxUseShell.Valu
    e
    App.LogCdeCpShell.Append WinRegl.TextFieldCpShell.Te
    xt
    App.LogCopyRBerr.Append WinRegl.CheckBoxCopyRBerr.
    Value
    If CheckBoxRemplRecent.Value Then
      If CheckBoxSimu.Value Then
        TampOp = "Rempl_Src->Cib"
        TampJ = TabSymb(SymbOpNon)
      Else
        TampOp = "NeRienFaire" ' Car l'opération aura é
        té effectuée ci-dessous
```

```
                If App.AScriptCopy(EltSource(IndSource), DossCi
                ble, DossCibTrash, True, WinRegl.CheckBoxUseS
                hell.Value, WinRegl.TextFieldCpShell.Text, WinR
                egl.CheckBoxCopyRBerr.Value) Then ' On rempla
                ce  EltCible(IndCible)  par  EltSource(IndSource)
                  TampJ = TabSymb(SymbOpErrDiv)
                  ' NbErrRemplAnc = NbErrRemplAnc + 1 ' Se
                  ra de toute façon comptabilisé ci-dessous c
                  ar rempl fich plus récent
                Else
                  TampJ = TabSymb(SymbOpOui)
                End If
              End If
            Else ' On a choisi de ne pas remplacer un fichier plus réc
            ent
              TampOp = "NeRienFaire" ' Il n'y aura pas de case à c
              ochée même si en mode Simulation, ça évite de faire
               la copie sans faire attention en cliquant  Tout coche
              r
              TampJ = TabSymb(SymbOpNon)
            End If
            TampJ = TampJ + TabSymb(SymbAncien)
            TampExt = ExtSource(IndSource)
            TampText = ExtCible(IndCible)
            If not(TampExt = TampText) Then TampExt = TampExt +
             " " + TampText ' Si un fichier remplace un alias par exem
            ple
            EcrireJournal(TampOp, TampJ, TampExt, EltSource(IndSo
            urce).AbsPath_f, TabSymb(SymbFlecheD), EltCible(IndCibl
            e).AbsPath_f)

          End If

        End If ' DateCible <> DateSource
      End If ' ( DossSource  et  DossCible  Directory ) ou  CheckBoxGererDat
      e
      NomCible.Remove IndCible ' A été traité
      ExtCible.Remove IndCible
      EltCible.Remove IndCible
      IsDossCible.Remove IndCible ' TypeCible

    Else ' IndCible = -1 ,  NomS  n'est pas dans dans DossCible (ou alors il est
     invisible)
      If not DossCible.TrueChild(NomS).Exists Then
        ' MsgBox "Cet élément n'existe pas : " + DossCible.AbsPath_f + N
        omS
```

```
    App.LogEltSrc.Append EltSource(IndSource)
    App.LogEltCib.Append DossCible
    App.LogDossTrash.Append DossCibTrash
    App.LogUseShell.Append WinRegl.CheckBoxUseShell.Value
    App.LogCdeCpShell.Append WinRegl.TextFieldCpShell.Text
    App.LogCopyRBerr.Append WinRegl.CheckBoxCopyRBerr.Value
    If CheckBoxSimu.Value Then
      TampOp = "Copie_Src->Cib"
      TampJ = TabSymb(SymbOpNon)
    Else
      TampOp = "NeRienFaire" ' Car l'opération aura été effectuée c
      i-dessous
      If App.AScriptCopy(EltSource(IndSource), DossCible, DossCibT
      rash, False, WinRegl.CheckBoxUseShell.Value, WinRegl.TextFi
      eldCpShell.Text, WinRegl.CheckBoxCopyRBerr.Value) Then ' O
      n copie  EltSource(IndSource)  dans  DossCible
        TampJ = TabSymb(SymbOpErrDiv)
        NbErrRemplAnc = NbErrRemplAnc + 1
      Else
        TampJ = TabSymb(SymbOpOui)
      End If
    End If
    TampJ = TampJ + TabSymb(SymbCopy)
    TampExt = ExtSource(IndSource)
    EcrireJournal(TampOp, TampJ, TampExt, EltSource(IndSource).Abs
    Path_f, TabSymb(SymbFlecheD), DossCible.AbsPath_f)

  Else ' Ce n'est pas normal qu'il soit invisible car visible dans  DossSour
  ce
    ' Non If (not DossCible.TrueChild(NomS).Visible) or (Left(NomS, 1)
    = ".") Then ' Je revérifie si invisible car si en mode Simu on n'a pu
    ne le Remove mais pas le supprimer réellement
    NbErrRemplAnc = NbErrRemplAnc + 1
    ' En mode normal (non Simu), on ne s'occupe pas des LogSrc, Log
    Cib, LogDossTrash et LogCopyRBerr mais on vérifie que même no
    mbre que ligne de Journal
    '  Et en BatchSync, il peut y avoir des réglages en mode normal et
    d'autres en mode simulé
    App.LogEltSrc.Append EltSource(IndSource) ' Pas Nil pour pouvoir
    afficher l'élément dans le Log avec un clic dans la liste
    App.LogEltCib.Append DossCible.TrueChild(NomS) ' Pas Nil pour
    pouvoir afficher l'élément dans le Log avec un clic dans la liste
    App.LogDossTrash.Append Nil
    App.LogUseShell.Append False ' Ou True on s'en fout
```

```
        App.LogCdeCpShell.Append " " ' N'importe quoi on s'en fout car in
        utilé également
        App.LogCopyRBerr.Append False ' Ou True on s'en fout
        TampOp = "NeRienFaire" ' On ne fera rien car erreur
        TampJ = TabSymb(SymbOpNon) + TabSymb(SymbOpErrDiv)
        TampExt = ExtSource(IndSource)
        TampText = DossCible.TrueChild(NomS).Extension_f
        If not(TampExt = TampText) Then TampExt = TampExt + " " + Ta
        mpText ' Si un fichier remplace un alias par exemple
        EcrireJournal(TampOp, TampJ, TampExt, EltSource(IndSource).Abs
        Path_f, TabSymb(SymbOpErrDiv), DossCible.TrueChild(NomS).Abs
        Path_f) ' Avant je mettais Nil à la place de EltSource(IndSource)

      End If '  DossCible.TrueChild(NomS)  existe

    End If
End If

' Même chose dans les boucles IndSource et IndCible
If App.TampCloseLog Then ' On a demandé WinLog.Close mais on l'a cancélé
    WinLog.TampClose = True ' Cette fois on ne le cancelera pas
    ' MsgBox "On avait empêché fermeture de WinLog, maintenant on ferme"
    WinLog.Close
    ' Pause(1, True) ' ThreadSync.Sleep 10, False ' Inutile, voir note dans WinLo
    g.Close, les lignes de codes de l'event WinLog.Close sont exécutées comm
    e étant dans ce Thread
    App.TampCloseLog = False '  donc on ne passe à la ligne suivante qu'après
     avoir fini l'event WinLog.Close
End If

If Ticks > TicksEvent Then
    App.DoEvents
    TicksEvent = Ticks + TicksInt
End If
If ArretUrg Then Exit ' A pu être mis à  Vrai  par ailleurs
#If DebugBuild Then
    iNbre = UBound(NomSource)
    If not((iNbre = UBound(EltSource)) and (iNbre = UBound(ExtSource)) and (iN
    bre = UBound(IsDossSource))) Then MsgBox "Synchro , Problème Tom" + E
    ndOfLine + str(iNbre) + " ≠ " + str(UBound(ExtSource)) + " ≠ " + str(UBoun
    d(EltSource)) + " ≠ " + str(UBound(IsDossSource)) ' TypeSource
    iNbre = UBound(NomCible)
    If not((iNbre = UBound(EltCible)) and (iNbre = UBound(ExtCible)) and (iNbre
     = UBound(IsDossCible))) Then MsgBox "Synchro , Problème Tom" + EndOf
    Line + str(iNbre) + " ≠ " + str(UBound(ExtCible)) + " ≠ " + str(UBound(EltCi
```

```
      ble)) + " ≠ " + str(UBound(IsDossCible)) ' TypeCible
    #EndIf
Next IndSource ' UBound(NomSource) = UBound(ExtSource) = UBound(EltSource) =
UBound(IsDossSource) ' TypeSource
' MsgBox "On a fini phase 1" + EndOfLine + DossSource.AbsPath_f + EndOfLine + D
ossCible.AbsPath_f

If not ArretUrg Then
  NoDerElt = UBound(NomCible) ' = UBound(EltCible) = UBound(IsDossCible) ' Ty
  peCible
  For IndCible = 0 to NoDerElt ' On traite les éléments restant dans  DossCible
    If Ticks > TicksEvent Then
      App.DoEvents
      TicksEvent = Ticks + TicksInt
    End If
    NomC = NomCible(IndCible)
    ' #If DebugBuild Then
    ' Je ne fais pas car donne une alerte si un fichier remplace un alias par exe
    mple
    ' If not(DossCible.TrueChild(NomC).AbsPath_f = EltCible(IndCible).AbsPath
    _f) Then MsgBox "Synchro , Problème Tom" + EndOfLine + DossCible.True
    Child(NomC).AbsPath_f + EndOfLine + EltCible(IndCible).AbsPath_f
    ' #EndIf
    Drap_eff = False
    If LgGTextTrash > 0 Then
      If InStr(NomC, GTextTrash) = 1 Then
        Drap_eff = True
        TampPath = EltCible(IndCible).AbsPath_f ' Parce qu'il risque d'être
        renommé ci-dessous avant d'être déplacé à la corbeille
        App.LogEltSrc.Append Nil
        App.LogEltCib.Append EltCible(IndCible)
        App.LogDossTrash.Append DossCibTrash
        App.LogUseShell.Append False ' Ou True on s'en fout
        App.LogCdeCpShell.Append " " ' N'importe quoi on s'en fout car in
        utilé également
        App.LogCopyRBerr.Append False ' Ou True on s'en fout
        If CheckBoxSimu.Value Then
          TampOp = "Effacer_Cib"
          TampJ = TabSymb(SymbOpNon)
        Else
          TampOp = "NeRienFaire" ' Car l'opération aura été effectuée c
          i-dessous
          If App.DeleteFileOrFolder(EltCible(IndCible), DossCibTrash) Th
          en
```

```
      TampJ = TabSymb(SymbOpErrDiv)
      NbErrRemplAnc = NbErrRemplAnc + 1
   Else
      TampJ = TabSymb(SymbOpOui)
   End If
End If
TampJ = TampJ + TabSymb(SymbEff)
TampExt = ExtCible(IndCible)
EcrireJournal(TampOp, TampJ, TampExt, "", TabSymb(SymbFlEffD),
 TampPath)

' On ne fait rien d'autre que vérifier
If DossSource.TrueChild(NomC).Exists and (NomSource.IndexOf(N
omC) = -1) Then ' Ce n'est pas normal qu'il soit invisible car visibl
e dans  DossCible
   ' Non If (not DossSource.TrueChild(NomC).Visible) or (Left(No
   mC, 1) = ".") Then ' Je revérifie si invisible car si en mode Sim
   u on ne l'a pas supprimé réellement
   NbErrRemplAnc = NbErrRemplAnc + 1
   ' En mode normal (non Simu), on ne s'occupe pas des LogSrc,
    LogCib, LogDossTrash et LogCopyRBerr mais on vérifie que
   même nombre que ligne de Journal
   '  Et en BatchSync, il peut y avoir des réglages en mode norm
   al et d'autres en mode simulé
   App.LogEltSrc.Append DossSource.TrueChild(NomC) ' Pas Nil
   pour pouvoir afficher l'élément dans le Log avec un clic dans l
   a liste
   App.LogEltCib.Append Nil
   App.LogDossTrash.Append Nil
   App.LogUseShell.Append False ' Ou True on s'en fout
   App.LogCdeCpShell.Append " " ' N'importe quoi on s'en fout c
   ar inutilé également
   App.LogCopyRBerr.Append False ' Ou True on s'en fout
   TampOp = "NeRienFaire" ' On ne fera rien car erreur
   TampJ = TabSymb(SymbOpNon) + TabSymb(SymbOpErrDiv)
   TampExt = DossSource.TrueChild(NomC).Extension_f
   EcrireJournal(TampOp, TampJ, TampExt, DossSource.TrueChil
   d(NomC).AbsPath_f, TabSymb(SymbOpErrDiv), "")
   ' Non End If ' Revérifie si invisible

Else
   NomS = Mid(NomC, LgGTextTrash + 1)
   If not(NomS = "") Then ' On regarde si  NomS = NomC ne com
   mençant pas par  GTextTrash  est dans  Source
```

```
            If DossSource.TrueChild(NomS).Exists and (NomSource.In
            dexOf(NomS) = -1) Then ' Ce n'est pas normal qu'il soit i
            nvisible car visible dans  DossCible, pas à Nil puisque  D
            ossSource  existe
              ' Non If (not DossSource.TrueChild(NomS).Visible) or
              (Left(NomS, 1) = ".") Then ' Je revérifie si invisible car
               si en mode Simu on ne l'a pas supprimé réellement
              NbErrRemplAnc = NbErrRemplAnc + 1
              ' En mode normal (non Simu), on ne s'occupe pas de
              s LogSrc, LogCib, LogDossTrash et LogCopyRBerr ma
              is on vérifie que même nombre que ligne de Journal
              '  Et en BatchSync, il peut y avoir des réglages en mo
              de normal et d'autres en mode simulé
              App.LogEltSrc.Append DossSource.TrueChild(NomS) '
               Pas Nil pour pouvoir afficher l'élément dans le Log a
              vec un clic dans la liste
              App.LogEltCib.Append Nil
              App.LogDossTrash.Append Nil
              App.LogUseShell.Append False ' Ou True on s'en fout
              App.LogCdeCpShell.Append " " ' N'importe quoi on s'
              en fout car inutilé également
              App.LogCopyRBerr.Append False ' Ou True on s'en fo
              ut
              TampOp = "NeRienFaire" ' On ne fera rien car erreur
              TampJ = TabSymb(SymbOpNon) + TabSymb(SymbO
              pErrDiv)
              TampExt = DossSource.TrueChild(NomS).Extension_
              f
              EcrireJournal(TampOp, TampJ, TampExt, DossSource
              .TrueChild(NomS).AbsPath_f, TabSymb(SymbOpErrDi
              v), "")
              ' Non End If ' Revérifie si invisible
            End If
          End If

        End If

    End If ' '  NomC  de  Cible  commence par  GTextTrash
End If ' LgGTextTrash > 0

If not Drap_eff Then '  EltCible(IndCible)  n'a pas été effacé
    If RadioButtSyncMode(0).Value Then '  Réciproque
        If not DossSource.TrueChild(NomC).Exists Then
            App.LogEltSrc.Append DossSource
            App.LogEltCib.Append EltCible(IndCible)
```

```
    App.LogDossTrash.Append DossSrcTrash
    App.LogUseShell.Append WinRegl.CheckBoxUseShell.Value
    App.LogCdeCpShell.Append WinRegl.TextFieldCpShell.Text
    App.LogCopyRBerr.Append WinRegl.CheckBoxCopyRBerr.Valu
    e
    If CheckBoxSimu.Value Then
      TampOp = "Copie_Cib->Src"
      TampJ = TabSymb(SymbOpNon)
    Else
      TampOp = "NeRienFaire" ' Car l'opération aura été effect
      uée ci-dessous
      If App.AScriptCopy(EltCible(IndCible), DossSource, DossS
      rcTrash, False, WinRegl.CheckBoxUseShell.Value, WinRegl
      .TextFieldCpShell.Text, WinRegl.CheckBoxCopyRBerr.Val
      ue) Then ' On copie  EltCible(IndCible)  dans  DossSource
        TampJ = TabSymb(SymbOpErrDiv)
        NbErrRemplAnc = NbErrRemplAnc + 1
      Else
        TampJ = TabSymb(SymbOpOui)
      End If
    End If
    TampJ = TampJ + TabSymb(SymbCopy)
    TampExt = ExtCible(IndCible)
    EcrireJournal(TampOp, TampJ, TampExt, DossSource.AbsPath
    _f, TabSymb(SymbFlecheG), EltCible(IndCible).AbsPath_f)

  Else ' Ce n'est pas normal qu'il soit invisible car visible dans  Doss
  Cible
    ' Non If (not DossSource.TrueChild(NomC).Visible) or (Left(No
    mC, 1) = ".") Then ' Je revérifie si invisible car si en mode Sim
    u on ne l'a pas supprimé réellement
    NbErrRemplAnc = NbErrRemplAnc + 1
    ' En mode normal (non Simu), on ne s'occupe pas des LogSrc,
     LogCib, LogDossTrash et LogCopyRBerr mais on vérifie que
    même nombre que ligne de Journal
    '  Et en BatchSync, il peut y avoir des réglages en mode norm
    al et d'autres en mode simulé
    App.LogEltSrc.Append DossSource.TrueChild(NomC) ' Pas Nil
    pour pouvoir afficher l'élément dans le Log avec un clic dans l
    a liste
    App.LogEltCib.Append EltCible(IndCible) ' Pas Nil pour pouvoi
    r afficher l'élément dans le Log avec un clic dans la liste
    App.LogDossTrash.Append Nil
    App.LogUseShell.Append False ' Ou True on s'en fout
```

```
      App.LogCdeCpShell.Append " " ' N'importe quoi on s'en fout c
      ar inutilé également
      App.LogCopyRBerr.Append False ' Ou True on s'en fout
      TampOp = "NeRienFaire" ' On ne fera rien car erreur
      TampJ = TabSymb(SymbOpNon) + TabSymb(SymbOpErrDiv)
      TampExt = ExtCible(IndCible)
      TampText = DossSource.TrueChild(NomC).Extension_f
      If not(TampText = TampExt) Then TampExt = TampText + " "
      + TampExt ' Si un fichier remplace un alias par exemple
      EcrireJournal(TampOp, TampJ, TampExt, DossSource.TrueChil
      d(NomC).AbsPath_f, TabSymb(SymbOpErrDiv), EltCible(IndCib
      le).AbsPath_f) ' Avant je mettais Nil à la place de EltCible(IndC
      ible)
      ' Non End If
    End If

ElseIf RadioButtSyncMode(2).Value Then '  Source sur Cible
    TampPath = EltCible(IndCible).AbsPath_f ' Parce qu'il risque d'être
    renommé ci-dessous avant d'être déplacé à la corbeille
    App.LogEltSrc.Append Nil
    App.LogEltCib.Append EltCible(IndCible)
    App.LogDossTrash.Append DossCibTrash
    App.LogUseShell.Append False ' Ou True on s'en fout
    App.LogCdeCpShell.Append " " ' N'importe quoi on s'en fout car in
    utilé également
    App.LogCopyRBerr.Append False ' Ou True on s'en fout
    If CheckBoxSimu.Value Then
      TampOp = "Effacer_Cib"
      TampJ = TabSymb(SymbOpNon)
    Else
      TampOp = "NeRienFaire" ' Car l'opération aura été effectuée c
      i-dessous
      If App.DeleteFileOrFolder(EltCible(IndCible), DossCibTrash) Th
      en
        TampJ = TabSymb(SymbOpErrDiv)
        NbErrRemplAnc = NbErrRemplAnc + 1
      Else
        TampJ = TabSymb(SymbOpOui)
      End If
    End If
    TampJ = TampJ + TabSymb(SymbEff)
    TampExt = ExtCible(IndCible)
    EcrireJournal(TampOp, TampJ, TampExt, "", TabSymb(SymbFlEffD),
     TampPath)
```

```
          ' Else ' If RadioButtSyncMode(1).Value Then '  Source vers Cible
          '  On ne fait rien, on laisse EltCible(IndCible) dans DossCible

        End If
      End If

      ' Même chose dans les boucles IndSource et IndCible
      If App.TampCloseLog Then ' On a demandé WinLog.Close mais on l'a cancé
      lé
        WinLog.TampClose = True ' Cette fois on ne le cancelera pas
        ' MsgBox "On avait empêché fermeture de WinLog, maintenant on fer
        me"
        WinLog.Close
        ' Pause(1, True) ' ThreadSync.Sleep 10, False ' Inutile, voir note dans W
        inLog.Close, les lignes de codes de l'event WinLog.Close sont exécuté
        es comme étant dans ce Thread
        App.TampCloseLog = False '  donc on ne passe à la ligne suivante qu'a
        près avoir fini l'event WinLog.Close
      End If

      If Ticks > TicksEvent Then
        App.DoEvents
        TicksEvent = Ticks + TicksInt
      End If
      If ArretUrg Then Exit ' A pu être mis à  Vrai  par ailleurs
      ' #If DebugBuild Then ' Inutile de vérifier car pas de Remove
      ' If not((UBound(NomSource) = UBound(EltSource)) and (UBound(NomSourc
      e) = UBound(ExtSource)) and (UBound(NomSource) = UBound(IsDossSource
      )) and (UBound(NomCible) = UBound(EltCible)) and (UBound(NomCible) = U
      Bound(ExtCible)) and (UBound(NomCible) = UBound(IsDossCible))) Then Ms
      gBox "Synchro , Problème Tom" + EndOfLine + str(UBound(NomSource)) +
      " ≠ " + str(UBound(ExtSource)) + " ≠ " + str(UBound(EltSource)) + " ≠ " + st
      r(UBound(IsDossSource)) + EndOfLine + str(UBound(NomCible)) + " ≠ " + s
      tr(UBound(ExtCible)) + " ≠ " + str(UBound(EltCible)) + " ≠ " + str(UBound(Is
      DossCible)) ' TypeSource , TypeCible
      ' #EndIf
    Next IndCible ' UBound(NomCible) = UBound(ExtCible) = UBound(EltCible) = UB
    ound(IsDossCible) ' TypeCible
  End If ' ArretUrg

  #If TargetMacOS Then
    If (not ArretUrg) and (not Drap_err) and CheckBoxSyncIc.Value Then
      Select Case PopupBarOut.ListIndex
```

```
      Case 0
        TampJ = "LeaveIt"
      Case 1
        TampJ = "False"
      Case 2
        TampJ = "True"
      Else
        MsgBox "Error ! Please contact the authors." + EndOfLine + "PopupBar
        Out.ListIndex = " + str(PopupBarOut.ListIndex) + " : " + PopupBarOut.
        Text
        ArretUrg = True ' On arrête tout
      End Select
      SyncIcons(DossSource.ShellPath_fAS, DossCible.ShellPath_fAS, TampJ) ' S'il
      y a une erreur dans le script, celui-ci s'arrête et RealBasic continue
    End If
  #Else
    #If DebugBuild Then
      MsgBox "Synchro , Problème Tom, tu devrais être en TargetMacOS !"
    #EndIf
  #EndIf


Exception Err ' C'est sûrement une Nil Objection Error sur un des fichiers de  DossSource
  ou  DossCible

  ' Inutile Drap_err = True '                  On a dû ejecter l'un des disques pendant le
  process ou StackOverflowException ou ???
  NbErrRemplAnc = NbErrRemplAnc + 1
  ' En mode normal (non Simu), on ne s'occupe pas des LogSrc, LogCib, LogDossTras
  h et LogCopyRBerr mais on vérifie que même nombre que ligne de Journal
  '  Et en BatchSync, il peut y avoir des réglages en mode normal et d'autres en mode
  simulé
  App.LogEltSrc.Append DossSource ' Pas Nil pour pouvoir afficher l'élément dans le L
  og avec un clic dans la liste
  App.LogEltCib.Append DossCible ' Pas Nil pour pouvoir afficher l'élément dans le Lo
  g avec un clic dans la liste
  App.LogDossTrash.Append Nil
  App.LogUseShell.Append False ' Ou True on s'en fout
  App.LogCdeCpShell.Append " " ' N'importe quoi on s'en fout car inutilé également
  App.LogCopyRBerr.Append False ' Ou True on s'en fout
  TampOp = "NeRienFaire" ' On ne fera rien car erreur
  If Err IsA StackOverflowException Then ' Voir plus haut le Try Catch  Synchro()  puisq
  u'on met le même symbole d'erreur puisque StackOverflowException
    ' MsgBox "The stack has overflowed!"
```

```
    TampJ = TabSymb(SymbOpNon) + TabSymb(SymbOpErrOvF)
  Else
    TampJ = TabSymb(SymbOpNon) + TabSymb(SymbOpErrInc) ' Ou SymbOpErrDiv
    ArretUrg = True ' On arrête tout
  End If
  TampExt = TabSymb(SymbDoss)
  EcrireJournal(TampOp, TampJ, TampExt, DossSource.AbsPath_f, TabSymb(SymbFlec
  hes), DossCible.AbsPath_f)

End Sub
```

WinMain.TabIgnorElts:

```
Private Sub TabIgnorElts(CeTexte as String, DrapIgnorPas as Boolean)

  Dim iNbre, Fin_iNbre as Int16 ' Je mémorise  MemIgnorEltNom  et  MemIgnorEltExt
  en retirant les doublons, par contre je laisse si par exemple
  Dim TampText as String '  il y a  MonFichier.rtf  et l'extension  .rtf  car le gars peut v
  ouloir ignorer  MonFichier.rtf  au 1er niveau  uniquement

  #If DebugBuild Then
    If not(Len(SepAbsPath) = 1) Then MsgBox "Problème Tom, dans TabIgnorElts." +
     EndOfLine + "Len(SepAbsPath) = " + str(Len(SepAbsPath)) + " et devrait être 1 !
    "
    ' Il faut aussi corriger dans CheckBoxIgnorElt.DropObject
  #EndIf

  If Right(CeTexte, 1) = SepAbsPath Then CeTexte = Left(CeTexte, Len(CeTexte) - 1) '
   Voir notes dans  EditFieldIgnorElt.TextChange  et  EditFieldPasIgnor.TextChange , Il
   peut déjà y avoir : à la fin
  ' Inutile, on aura un élément vide plus bas : If Left(CeTexte, 1) = SepAbsPath Then C
  eTexte = Mid(CeTexte, 2) ' Normalement impossible car vérifié avant

  If DrapIgnorPas Then ' On s'occupe des éléments ignorés
    If UBound(MemIgnorEltNom) > -1 Then ReDim MemIgnorEltNom(-1) ' Ne fait pa
    s de bug si  MemIgnorElt  non vide mais par contre ne lui affecte pas sa nouvell
    e valeur
    If not(CeTexte = "") Then MemIgnorEltNom = Split(CeTexte, SepAbsPath) ' Sinon
     créré un tableau {""}
    If UBound(MemIgnorEltExt) > -1 Then ReDim MemIgnorEltExt(-1) ' Là aussi, le t
    ableau doit être vide puisqu'on réajoute les extensions avec  .Append

    Fin_iNbre = UBound(MemIgnorEltNom)
```

```
    iNbre = 0
    Do Until (iNbre > Fin_iNbre)
      TampText = MemIgnorEltNom(iNbre)
      If TampText = "" Then
        MemIgnorEltNom.Remove iNbre
        Fin_iNbre = Fin_iNbre - 1
        MsgBox "Error ! Please contact the authors." + EndOfLine + "TabIgnorE
        lts, EditFieldIgnorElt should not begin with ':' or have two ':' following l
        ike '::' !" ' or end with  or (Right(EditFieldIgnorElt.Text, 1) = SepAbsPath
        )
      ElseIf (iNbre < Fin_iNbre) and MemIgnorEltNom.IndexOf(TampText, iNbre
      + 1) > -1 Then
        MemIgnorEltNom.Remove iNbre
        Fin_iNbre = Fin_iNbre - 1
        ' Pas d'alerte, mais cet élément est supprimé vu qu'on traitera son dou
        ble plus loin
      ElseIf Left(TampText, 1) = "." Then ' Des fois que le gars ait entré un simple
       '.'  and (not(TampText = "."))  mais on s'en fou, il n'y aura jamais d'extensi
      on simplement '.'
        MemIgnorEltExt.Append TampText ' Mais je ne supprime pas un simpl
        e . car quand le gars tape l'extension il tape d'abord le .
        MemIgnorEltNom.Remove iNbre
        Fin_iNbre = Fin_iNbre - 1
      Else
        iNbre = iNbre + 1
      End If
    Loop
    ' MsgBox "MemIgnorEltNom(" + str(Fin_iNbre + 1) + ") = '" + Join(MemIgnorElt
    Nom, SepAbsPath) + "'" + EndOfLine + "MemIgnorEltExt(" + str(UBound(MemIg
    norEltExt) + 1) + ") = '" + Join(MemIgnorEltExt, SepAbsPath) + "'"

  Else ' On s'occupe des éléments pas ignorés
    If UBound(MemPasIgnor) > -1 Then ReDim MemPasIgnor(-1) ' Ne fait pas de bu
    g si  MemPasIgnor  non vide mais par contre ne lui affecte pas sa nouvelle vale
    ur
    If not(CeTexte = "") Then MemPasIgnor = Split(CeTexte, SepAbsPath) ' Sinon cré
    ré un tableau {""}
    ' MsgBox "MemPasIgnor(" + str(UBound(MemPasIgnor) + 1) + ") = '" + Join(Me
    mPasIgnor, SepAbsPath) + "'"

  End If

End Sub
```

WinMain.TextIgnorElts:

```
Function TextIgnorElts(DrapIgnorPas as Boolean) As String

  Dim TampText as String ' Remet en texte les 2 tableaux  MemIgnorEltNom  et  MemI
  gnorEltExt  ou le tableau  MemPasIgnor

  If DrapIgnorPas Then ' On s'occupe des éléments ignorés
    If UBound(MemIgnorEltNom) = -1 Then
      If UBound(MemIgnorEltExt) = -1 Then
        TampText = "" ' Il y est déjà
      Else
        TampText = Join(MemIgnorEltExt, SepAbsPath) ' De toute façon si UBo
        und(MemIgnorEltExt) = -1 donc {} ça retourne ""
      End If
    Else
      If UBound(MemIgnorEltExt) = -1 Then
        TampText = Join(MemIgnorEltNom, SepAbsPath) ' De toute façon si UB
        ound(MemIgnorEltNom) = -1 donc {} ça retourne ""
      Else
        TampText = Join(MemIgnorEltNom, SepAbsPath) + SepAbsPath + Join(
        MemIgnorEltExt, SepAbsPath)
      End If
    End If

  Else ' On s'occupe des éléments pas ignorés
    TampText = Join(MemPasIgnor, SepAbsPath) ' Si UBound(MemIgnorEltNom) = -
    1 donc {} ça retourne ""

  End If

  Return TampText

End Function
```

WinMain.TraitComDoss:

```
Private Function TraitComDoss(CeDossier as FolderItem, CetteExt as String) As Boolean

  ' Cette fonction retourne Vrai si CeDossier doit être traité comme un dossier, Faux s
  i comme un fichier
```

```
' Avant je vérifiais si le dossier n'avait pas d'extension mais suite à la nouvelle méth
ode d'obtention d'extension j'ai dû changer
' Je vérifie si Package mais aussi si extension car apparemment certains dossiers av
ec extension ne sont pas vus comme Package

' DossPackFich à Faux si on considère les dossiers Package comme des fichiers
'  donc à Vrai si on ne s'occupe pas de si Package ou non --> un Package est un do
ssier

' DossExtFich à Faux si on considère les dossiers ayant une Extension comme des fi
chiers
'  donc à Vrai si on ne s'occupe pas de si Extension ou non --> un dossier avec une
 Ext est un dossier

' MsgBox CeDossier.AbsPath_f + EndOfLine + "Dossier : " + Cstr(CeDossier.Director
y) + EndOfLine + "Paquet : " + Cstr(CeDossier.IsPackage_f) + EndOfLine + "Ext : '"
+ CetteExt + " = " + CeDossier.Extension_f + "'" + EndOfLine + "Result : " + Cstr(C
eDossier.Directory and (DossPackFich or (not CeDossier.IsPackage_f)) and (DossExtF
ich or (CetteExt = "")))

#If DebugBuild Then
   If not CeDossier.Directory Then MsgBox "Problème Tom, TraitComDoss , tu ne
   dois y tester que des dossiers !"
   If not(CetteExt = CeDossier.Extension_f) Then MsgBox "Problème Tom, TraitCo
   mDoss , pb extension :" + EndOfLine + CetteExt + " ≠ " + CeDossier.Extension
   _f
#EndIf

If CeDossier = Nil Then ' Ne devrait pas arriver
   ArretUrg = True ' On arrête tout
   Return False ' On devrait avoir une NilObjectException dans Synchro en regarda
   nt si Alias (si gère date)
   ' Ou (autre solution)
   ' Return True ' On rappellera Synchro en récursif et on aura le bug mais avec un
   e ligne dans le journal
   ' En faisant comme ça on a peut-être une ligne dans le journal donc on ne sait
   pas qui est fautif mais c'est risqué

Else
   ' Simplifié ci-dessous : CeDossier.Directory and ((not WinRegl.CheckBoxDossPa
   ck.Value) or (not CeDossier.IsPackage_f)) and ((not WinRegl.CheckBoxDossExt.V
   alue) or (CetteExt = ""))
   ' Return CeDossier.Directory and (not(WinRegl.CheckBoxDossPack.Value and Ce
   Dossier.IsPackage_f)) and (not(WinRegl.CheckBoxDossExt.Value and (not(CetteE
   xt = ""))))
```

```
    ' Avant je faisais comme ci-dessus car j'envoyais tout type d'élément à cette fo
    nction, mais désormais je n'envois que des dossiers
    Return ((not(WinRegl.CheckBoxDossExt.Value and (not(CetteExt = "")))) and (not(
    WinRegl.CheckBoxDossPack.Value and CeDossier.IsPackage_f)))

  End If

End Function
```

WinMain.UpdtSyncEnab:

```
Private Sub UpdtSyncEnab(DrapAffTexte as Boolean)

  Dim DrapS, DrapC as Boolean ' Voir notes VerifFolders  Cette procédure vérifie que l
  es dossiers Source et Cible sont toujours valides
  Dim Long as Int16 ' Il peut y avoir des problèmes en cas d'éjection de disquette.  Dr
  apAffTexte  à Vrai si on veut réafficher chemin même si dossier bon

  DrapS = PasNil_Existe_Alias(DossierSource, True)
  If DrapS Then DrapS = DossierSource.DossPasPack_f ' (DossierSource.Directory and
  (not DossierSource.IsPackage_f)) ' Des fois que pointe sur autre chose
  If DrapS and (not App.AutorisVol) Then DrapS = not(DossierSource.Parent = Nil) ' Ou
   not(DossierSource.AbsPath_f = DossierSource.Volume_f.AbsPath_f) ' Ne doit pas êt
  re un volume (racine)
  If DrapS Then
    DossSrcPath = DossierSource.AbsPath_f
    LgPathSrc = Len(DossSrcPath)
    If DrapAffTexte Then StTextEcrire(StTextDossSource, DossSrcPath)
  Else
    DossierSource = Nil ' Des fois que pas à Nil mais qu'il n'existe plus
    DossSrcPath = ""
    LgPathSrc = 0
    StTextEcrire(StTextDossSource, Txt_Attente)
  End If

  DrapC = PasNil_Existe_Alias(DossierCible, True)
  If DrapC Then DrapC = DossierCible.DossPasPack_f ' (DossierCible.Directory and (n
  ot DossierCible.IsPackage_f)) ' Des fois que pointe sur autre chose
  If DrapC and (not App.AutorisVol) Then DrapC = not(DossierCible.Parent = Nil) ' Ou
  not(DossierCible.AbsPath_f = DossierCible.Volume_f.AbsPath_f) ' Ne doit pas être u
  n volume (racine)
  If DrapC Then ' Car si  DrapS  à False il faut quand même définir  DossCibPath  et  L
  gPathCib
```

```
    DossCibPath = DossierCible.AbsPath_f
    LgPathCib = Len(DossCibPath)
    If DrapS Then ' If DrapS and DrapC Then
      ' MsgBox "Src Vol : '" + DossSrcPath + "'" + EndOfLine + "Cibl Vol : '" + Dos
      sCibPath + "'" + EndOfLine + str(StrComp(Left(DossSrcPath, Long), Left(Do
      ssCibPath, Long), 1))
      If DossierSource.Volume_f.URLPath = DossierCible.Volume_f.URLPath Then
       ' Les deux dossiers sont sur le même volume, on regarde si égaux ou incl
      us l'un dans l'autre
        Long = Min(LgPathSrc, LgPathCib) ' Note : Il y a les 2 points : à la fin d
        e l'AbsPath_f puisque dossier
        DrapC = not(Left(DossSrcPath, Long) = Left(DossCibPath, Long))
        ' Voir notes dans  VerifFolders
        ' DrapC = (InStr(DossSrcPath, DossCibPath) = 0) and (InStr(DossCibPat
        h, DossSrcPath) = 0)
      End If
    End If ' DrapS
  End If ' DrapC

  If DrapC Then ' S'il est resté à Vrai ci-dessus, car en cas de pb d'imbrication, c'est lu
  i qui saute
    If DrapAffTexte Then StTextEcrire(StTextDossCible, DossCibPath)
  Else ' Si problème l'un dans l'autre ou égal
    DossierCible = Nil ' C'est lui qu'on annule
    DossCibPath = ""
    LgPathCib = 0
    StTextEcrire(StTextDossCible, Txt_Attente)
  End If

  ' MsgBox "DrapSyncEnab = " + CStr(DrapSyncEnab)
  DrapSyncEnab = (DrapS and DrapC)

End Sub
```

WinMain.VerifFolders:

```
Private Function VerifFolders(CeDoss as FolderItem, AutreDoss as FolderItem, DrapVFold
Alert as Boolean) As FolderItem

  Dim rep, Long, LgCeDoss, LgAutrDoss as Int16 ' Cette procédure vérifie que les 2 d
  ossiers ne sont ni identiques ni imbriqués l'un dans l'autre
  Dim CeDossPath, AutreDossPath as String ' Ressemble à UpdtSyncEnab
  ' Quand on fait GetFolderItem et qu'on a 2 volumes du même nom, RealBasic cherch
  e uniquement dans le 1er volume trouvé ayant ce nom. Donc il se peut que soit il n
```

```
e
'  trouve pas l'élément cherché (puisque sur l'autre volume), soit que les 2 items poi
ntent vers le même dossier (même volume) alors qu'à l'origine il s'agissait de 2 volu
mes différents

' MsgBox CeDoss.AbsPath_f
If PasNil_Existe_Alias(AutreDoss, True) Then '  CeDoss  ne peut pas être à Nil, testé
avant
   ' MsgBox "CeDoss : '" + CeDoss.AbsPath_f + "'" + EndOfLine + "AutreDoss : '" +
    AutreDoss.AbsPath_f
   If CeDoss.Volume_f.URLPath = AutreDoss.Volume_f.URLPath Then ' Les deux do
   ssiers sont sur le même volume, on regarde si égaux ou inclus l'un dans l'autre
      CeDossPath = CeDoss.AbsPath_f
      AutreDossPath = AutreDoss.AbsPath_f
      LgCeDoss = Len(CeDossPath)
      LgAutrDoss = Len(AutreDossPath)
      ' If CeDoss.AbsPath_f = AutreDoss.AbsPath_f Then ' Car  CeDoss = AutreD
      oss  jamais égal si on compare les items
      ' "Tom" = "tom" retourne Vrai, "Toma" = "Tomà" retourne Faux
      ' StrComp("Tom", "tom") retourne -1, StrComp("Toma", "Tomà") retourne -
      1, (0 si égal)
      ' MsgBox Cstr("Tom" = "tom") + " / " + Cstr("Toma" = "Tomà") + EndOfLine
       + str(StrComp("Tom", "tom", 1)) + " / " + str(StrComp("Toma", "Tomà", 1))
      + EndOfLine + str(StrComp("Tom", "tom", 0)) + " / " + str(StrComp("Toma",
       "Tomà", 0))
      ' MsgBox "Src : '" + CeDossPath + "'" + EndOfLine + "Cibl : '" + AutreDossP
      ath + "'" + EndOfLine + str(StrComp(CeDossPath, AutreDossPath, 1))
      Long = Min(LgCeDoss, LgAutrDoss) ' Note : Il y a les 2 points : à la fin de l'
      AbsPath_f puisque dossier
      If Left(CeDossPath, Long) = Left(AutreDossPath, Long) Then ' Donc 2 dossi
      ers avec le même path avec 1 qui aurait
         CeDoss = Nil '  une lettre de plus serait bien différent : "MonDisk:Doss
         ierA:" et "MonDisk:DossierAa:"
         If DrapVFoldAlert Then
            Beep
            If LgCeDoss = LgAutrDoss Then
               rep = MyDialogBox(Txt_PasMemeDoss, "", Btn_Ok, "", "", 0)
            Else
               rep = MyDialogBox(Txt_PasUnDsAutre, "", Btn_Ok, "", "", 0)
            End If
         End If
      End If
   End If
End If
```

```
  ' If PasNil_Monte(CeDoss) Then MsgBox CeDoss.AbsPath_f

  Return CeDoss

  ' Quand je faisais :   (mais maintenant que je contrôle les volumes et il n'y a plus de
  problème)
  ' If (InStr(CeDoss.AbsPath_f, TampText) > 0) or (InStr(TampText, CeDoss.AbsPath_f)
   > 0) Then
  ' Ca me sortait une erreur si je mettais un disque qui avait le nom d'un dossier.
  ' Exemple : CeDoss = "EMac HD2:Documents:"   et  AutreDoss = "Documents:"  ben
  ça bloquait

End Function
```

WinMain.VmemeNomVol:

```
Private Sub VmemeNomVol()

  Dim iNbre, jNbre, NbVol, jMemVol, rep as Int16 ' Affiche une alerte si plusieurs volu
  mes ont le même nom (le même Path)
  Dim DrapMemeNomVol, TampDrap as Boolean
  Dim VolAbsPath(-1), Vol_i_AbsPath as String

  TampDrap = False
  DrapMemeNomVol = False
  NbVol = VolumeCount - 1
  ' MsgBox "Nb volume : " + str(NbVol + 1) + EndOfLine + "Vol ayant déjà généré aler
  te :" + EndOfLine + Join(MemAbsPathVol, EndOfLine)
  For iNbre = 0 to NbVol ' Je génére un tableau des noms de volumes car je fais un In
  dexOf dedans plus loin
    VolAbsPath.Append Volume(iNbre).AbsPath_f
    ' MsgBox VolAbsPath(iNbre) + EndOfLine + Volume(iNbre).AbsPath_f + EndOfLi
    ne + Volume(iNbre).Volume_f.AbsPath_f
  Next iNbre

  ' For iNbre = 1 to NbVol ' On ne commence pas au 1er, il sera testé plus bas par rap
  port au 2ième
  iNbre = 0
  While iNbre < NbVol ' Pas la peine de tester Volume(NbVol) puisque comparé avec s
  uivant (jNbre = (iNbre = NbVol) + 1) qui n'existe donc pas
    Vol_i_AbsPath = VolAbsPath(iNbre)
    jNbre = VolAbsPath.IndexOf(Vol_i_AbsPath, iNbre + 1)
```

```
jMemVol = MemAbsPathVol.IndexOf(Vol_i_AbsPath) ' A t'on déjà généré alerte s
ur ce volume lors d'un ChangeConfig précédent ?
If jNbre > -1 Then ' Il y a un autre volume du même nom
   If not(DossierSource = Nil) Then ' Inutile de faire  PasNil_Monte(DossierSour
   ce)  car on vient de faire  UpdtSyncEnab
      If (DossierSource.Volume_f.AbsPath_f = Vol_i_AbsPath) and (jMemVol
      = -1) Then TampDrap = True
   End If '                                                    Si > -1 a
   lors on avait généré alerte lors d'un ChangeConfig précédent
   ' On ne fait alerte 2 volumes de même nom que si  DossierSource  ou  Dos
   sierCible  concernés
   If not(DossierCible = Nil) Then ' Inutile de faire  PasNil_Monte(DossierCible)
    car on vient de faire  UpdtSyncEnab
      If (DossierCible.Volume_f.AbsPath_f = Vol_i_AbsPath) and (jMemVol =
      -1) Then TampDrap = True
   End If '                                                    Si > -1 alo
   rs on avait généré alerte lors d'un ChangeConfig précédent
   If TampDrap Then
      DrapMemeNomVol = True ' On affichera l'alerte en fin de procédure
      TampDrap = False
      MemAbsPathVol.Append Vol_i_AbsPath ' On mémorise qu'on a généré
      une alerte sur ce volume puisque n'était pas listé dedans puisque  jMe
      mVol  était -1
      jMemVol = UBound(MemAbsPathVol)
   End If
   Do ' A l'entrée dans cette boucle on a forcément jNbre > -1 puisque testé
   plus haut
      ' MsgBox "On va supprimer VolAbsPath(" + str(jNbre) + ") = " + VolAbs
      Path(jNbre)
      VolAbsPath.Remove jNbre ' Qu'il ait ou non été ajouté à  MemAbsPath
      Vol  il a été traité puisque même nom que volume(iNbre)
      NbVol = NbVol - 1
      If iNbre < NbVol Then ' Sinon  IndexOf  plante si on commence recherc
      he après dernier élément
         jNbre = VolAbsPath.IndexOf(Vol_i_AbsPath, iNbre + 1)
      Else
         jNbre = -1
      End If
   Loop Until jNbre = -1

Else ' Il n'y a pas de volume du même nom
   If jMemVol > -1 Then ' Mais il y en avait lors du ChangeConfig précédent (l
   e volume en question a été éjecté ou renommé)
      ' MsgBox "On va supprimer MemAbsPathVol(" + str(jMemVol) + ") = "
      + MemAbsPathVol(jMemVol)
```

```
        MemAbsPathVol.Remove jMemVol
        jMemVol = -1 ' Mais inutile vu que remis à IndexOf la prochaine boucl
        e sur  iNbre
      End If

    End If ' jNbre > -1

    iNbre = iNbre + 1
  Wend ' iNbre

  If DrapMemeNomVol Then
    Beep
    rep = MyDialogBox(Txt_VolMemeNom, "", Btn_Ok, "", "", 0)
  End If

End Sub
```

WinMain.VrifAlertReglDef:

```
Private Sub VrifAlertReglDef()

  Dim iNbre as Int16 ' Joue un son et fait bouger la CheckMar de BevButtAffRegl si les
  réglages ne sont pas ceux par défaut
  Const AttBouge = 6

  If WinRegl.VrifReglDef(False) Then ' On ne vérifie pas si  WinRegl.TextFieldCpShell.T
  ext = App.DefCdeCpShell
    ' Ou WinRegl.EdFieldMargeTps.Text = Format(App.DefMargeTps, App.F_MBillio
    nSsEsp)                                On s'en fou de la ligne de commande Shell par
    défaut ou pas, si case pas coché alors inutile de faire l'alerte
    If (not App.SyncEnCours) and App.Sfx Then Beep3.play ' Sinon c'est qu'on est en
     Batch synchro et que changement de réglages automatique
    For iNbre = 0 to 2
      Pause(AttBouge, False)
      BevButtAffRegl.Bold = True
      BevButtAffRegl.Italic = True
      BevButtAffRegl.Refresh
      Pause(AttBouge, False)
      BevButtAffRegl.Bold = False
      BevButtAffRegl.Italic = False
      BevButtAffRegl.Refresh
    Next iNbre
    ' Pause(AttBouge, False)
```

```
    If (not App.SyncEnCours) and App.Sfx Then
      While Beep3.IsPlaying
      Wend
    End If
  End If

End Sub
```

WinMain.ZTestA:

```
Private Sub ZTestA()

  Dim TampTextA, TampTextB, TampTab(-1) as String ' Il s'agit de tests que je fais po
  ur vérifier les cohérences et avoir une alerte si je modifie des trucs qui en affectent
  d'autres

  If not(GTextTrash = EditFieldTrashBegin.Text) Then MsgBox "Error ! Please contact t
  he authors." + EndOfLine + "ZTestA, GTextTrash = " + GTextTrash + " ≠ EditFieldTr
  ashBegin = " + EditFieldTrashBegin.Text

  If not(Format(MemMargeTps, F_MBillionSsEsp) = WinRegl.EdFieldMargeTps.Text) Th
  en MsgBox "Error ! Please contact the authors." + EndOfLine + "ZTestA, MemMargeT
  ps = " + Format(MemMargeTps, F_MBillionSsEsp) + " ≠ EdFieldMargeTps = " + WinR
  egl.EdFieldMargeTps.Text

  TampTextA = TextIgnorElts(True)
  If not(TampTextA = "") Then ' Sinon créré un tableau {""}
    TampTab = Split(TampTextA, SepAbsPath) ' De plus, si on faisait directement
    TampTab = MemIgnorElt  les tableaux seraient liés
    TampTab.Sort
    TampTextA = Join(TampTab, SepAbsPath)
    ReDim TampTab(-1)
  End If
  TampTextB = EditFieldIgnorElt.Text
  If Right(TampTextB, 1) = SepAbsPath Then TampTextB = Left(TampTextB, Len(Tam
  pTextB) - 1) ' Voir notes dans  EditFieldIgnorElt.TextChange , Il peut déjà y avoir : à
  la fin
  If not(TampTextB = "") Then ' Sinon créré un tableau {""}
    TampTab = Split(TampTextB, SepAbsPath)
    TampTab.Sort
    TampTextB = Join(TampTab, SepAbsPath)
    ReDim TampTab(-1)
  End If
```

```
    If not(TampTextA = TampTextB) Then MsgBox "Error ! Please contact the authors."
    + EndOfLine + "ZTestA, Ignore, those should be equal :" + EndOfLine + "MemIgnor
    Elt = '" + TampTextA + "'" + EndOfLine + "EditFieldIgnorElt = '" + TampTextB + "'"

    ' Pour  IgnorElt  (ci-dessus) je trie car je remets les extensions à la fin. Pour  PasIgn
    or  (ci-dessous) c'est inutile de trier

    TampTextA = TextIgnorElts(False)
    TampTextB = EditFieldPasIgnor.Text
    If Right(TampTextB, 1) = SepAbsPath Then TampTextB = Left(TampTextB, Len(Tam
    pTextB) - 1) ' Voir notes dans  EditFieldPasIgnor.TextChange , Il peut déjà y avoir : à
     la fin
    If not(TampTextA = TampTextB) Then MsgBox "Error ! Please contact the authors."
    + EndOfLine + "ZTestA, PasIgnore, those should be equal :" + EndOfLine + "MemPa
    sIgnor = '" + TampTextA + "'" + EndOfLine + "EditFieldPasIgnor = '" + TampTextB
    + "'"


    ' AVANT je testais mais je le fais désormais dans TabIgnorElts
    ' If (Left(EditFieldIgnorElt.Text, 1) = SepAbsPath) Then MsgBox "Error ! Please contac
    t the authors." + EndOfLine + "ZTestA, EditFieldIgnorElt should not begin :" ' or end
     with  or (Right(EditFieldIgnorElt.Text, 1) = SepAbsPath)
    ' If not(Instr(EditFieldIgnorElt.Text, SepAbsPath + SepAbsPath) = 0) Then MsgBox "E
    rror ! Please contact the authors." + EndOfLine + "ZTestA, EditFieldIgnorElt should
    not contains ::"
    ' Je pourrais tester si aucun des éléments de  MemIgnorElt  à "" mais c'est impossibl
    e vu que commence pas par : , n'a pas :: à la suite, et ne se finit pas par :

End Sub

Private AlertVerAjour As Boolean


ArretUrg As Boolean


CfBarOutils(-1) As Int16


CfBatch(-1) As Boolean


CfCdeCpShell(-1) As String
```

CfCopyRBerr(-1) As Boolean

CfDossCible(-1) As String

CfDossExtFich(-1) As Boolean

CfDossPackFich(-1) As Boolean

CfDossSource(-1) As String

CfGererDate(-1) As Boolean

CfIgnorElt(-1) As Boolean

CfIgnorEltList(-1) As String

CfMargeTps(-1) As UInt16

CfModeSync(-1) As Int16

CfNomConfig(-1) As String

CfPasIgnor(-1) As Boolean

CfPasIgnorList(-1) As String

CfPremNiv(-1) As Boolean

CfRemplRecent(-1) As Boolean

CfSimul(-1) As Boolean

CfSyncIcones(-1) As Boolean

CfTextTrash(-1) As String

CfTrashIfBegin(-1) As Boolean

CfUseShell(-1) As Boolean

DateSync As Date

Private DossCibPath As String

DossCibTrash As FolderItem

DossierCible As FolderItem

DossierSource As FolderItem

Private DossSrcPath As String

DossSrcTrash As FolderItem

DrapBatch As Boolean

DrapSyncEnab As Boolean

GTextTrash As String

Icone128 As Picture

JournalDd As String

JournalDf As String

JournalD_HT As String

Private LgPathCib As Int16

Private LgPathSrc As Int16

Private MemAbsPathVol(-1) As String

MemIgnorEltExt(-1) As String

MemIgnorEltNom(-1) As String

MemMargeTps As UInt16

MemPasIgnor(-1) As String

NbErrRemplAnc As UInt16

NomConfig As String

Private NomDerConfig As String

Private PtitPtGris As Picture

Private PtitPtJaune As Picture

Private PtitPtVert As Picture

Private SyncMode0P As Picture

Private SyncMode1P As Picture

Private SyncMode2P As Picture

TabSymb(13) As String

Private TicksEvent As Double

WinMain Note: Keyboard_AsynckeyDown

Keyboard_AsynckeyDown

Je n'utilise plus Keyboard.AsynckeyDown(&h35) (Escape) ou autre car RealBasic détecte l' appuie sur la touche même si l'appli est en arrière plan.
Donc si on appuie Escape pour stopper un autre programme ça arrête aussi le program me RealBasic.
C'était de toute façon une double sécurité inutile

J'avais mis dans Method Synchro et LogSyncchro :

If Keyboard.AsynckeyDown(&h35) and (not ArretUrg) Then ' Touche Escape , pour pas fai re 2 fois Zoum (event KeyDown)
ArretUrg = True
If App.Sfx Then
Zoum.Play ' Pour signifier prise en compte ArretUrg On ne l'entends pas si coup de fr ein juste après, mais des fois

While Zoum.IsPlaying ' que ce soit plus long pour stopp
er tout
Wend
End If
' If App.Sfx Then Coup_de_freins.Play ' Fait plus tard quand Synchro finie A été fait dans
 le Thread de Synchro
End If

WinMain Note: Note_Appli

Note_Appli

Voir Note dans Localization

Si version Windows prévue, voir Neuronyx pour définir Variable Encodings
  et voir pour copier les fichiers sans utiliser Script (sur ancienne version de SyncTwoFol
ders)

Il ne faut pas cocher, dans le menu, MenuModifier , quand on a un racourci clavier local
isé car sous Windows ça met
 2 fois Ctrl --> (Ctrl-Ctrl-Z)

Il faut mettre Transparent-1-White pour la carte CarteCadre de Medias-Images

Quitter, qui s'ajoute automatiquement reste dans la langue (pas celle de Edit - Project S
etting)
Il n'est pas utile de localiser QuitMenuItem, il se mets tout seul à la bonne place dans la
bonne langue avec
le bon raccourci clavier (s'il n'est pas déjà pris)
Pour PrefsMenuItem il faut localiser le texte (car il ne se met pas tout seul) mais il est in
utile de localiser le raccourci clavier

CancelClose

Je mémorise DrapMenu dans DrapMenuTemp car normalement il est à Vrai (menus a
ctifs) mais il pourrait
déjà être à False si je venais d'un autre MovableModal
Les menus sont désactivés automatiquement mais on peut quitter par le Dock. Sous Ma
c, il y avait peu de
problème car il essayait de fermer uniquement la première fenêtre et si CancelClose il
arrêtait là, par contre

sous Windows il ferme les autres fenêtres, dont WinMain, et c'est le bordel

WinMain Note: Sync_Alias

Sync_Alias

A voir si je remet ça dans Synchro (là où j'ai écrit Voir note Sync_Alias) :

```
If DossSource.TrueChild(NomS).Alias and TampF.Alias Then ' Cas spécial 2 Alias
MsgBox DossSource.Child(NomS).AbsPath_f + EndOfLine + DossCible.Child(NomS).AbsP
ath_f
' NON If DossSource.Child(NomS).AbsPath_f = DossCible.Child(NomS).AbsPath_f Then '
On ne fait rien
' Car pointe sans doute dans son propre volume
If DossSource.Child(NomS).Name = DossCible.Child(NomS).Name Then ' On ne fait rien
' Note : TampF = DossCible.TRUEChild(NomS)
' Si l'élément pointé n'existe pas retourne le path de l'alias (voir notes PasNil_Existe_Alia
s()
DateCible = DossSource.TrueChild(NomS).ModificationDate ' = DateSource mais les 2 d
ates seraient liées
End If
End If
```

Bug de la copie avec AppleScript concernant les alias :

Lane Roathe, auteur de HexEdit et autres jeux, ma envoyé un email pour m'expliquer ce ça plantait quand on copiait l'alias avant l'élément vers lequel il pointait.
J'ai modifié Synchro afin de lister d'abord les fichiers, puis les dossiers, et enfin les alias afin de les traiter en dernier. Toutefois, si on descend dans un sous-dossier contenant des alias, ces derniers seront traités avant qu'on ne traite un autre sous-dossier placé dans le même dossier que lui-même contenant des alias.
Il aurait fallu mettre tous les alias dans un tableau global, puis traiter ce tableau après avoir fait tout le reste. Mais ça pausait problème dans le cas ou on regarde si un fichier du même nom existe, si jamais un fichier et un alias ont le même nom. Bref, c'était trop chiant.

Une autre méthode consistait à afficher dans le journal le type de chaque élément et de trier suivant ces types, mais ça alourdit et ça me fait chier

SymbDoss : ▷
SymbAlias : ↶
SymbFich : ▪

WinMain Control GroupBoxSource:

```
Sub DropObject(obj As DragItem, action As Integer)

  Dim f_elt, e_elt as FolderItem
  Dim rep as Int16

  ' Voir note dans  EnableMenuBoutons  On ne peut pas annuler un  AcceptFileDrop
  If DrapMenu Then ' Ou  If not(App.SyncEnCours or App.LogSEnCours) Then  Mais c'e
  st plus logique d'utiliser même variable que pour menu et bouton Choisir
    f_elt = Nil ' Inutile mais …
    ' MsgBox "obj.FolderItemAvailable : " + CStr(obj.FolderItemAvailable)
    If obj.FolderItemAvailable Then ' AcceptFileDrop à folder
      e_elt = Obj.FolderItem
      ' MsgBox "e est alias : " + CStr(e_elt.Alias) + EndOfLine + e_elt.AbsPath_f
      If e_elt.Alias Then e_elt = e_elt.GetTargetItem_f ' Mais bon, on ne peut pas
      drag drop des Alias de toute façon, à cause des AcceptFileType
      If (not(obj.NextItem or e_elt.Alias)) and e_elt.DossPasPack_f Then f_elt = e_
      elt ' Il ne doit y avoir qu'un élément, et si c'est resté un Alias, c'est qu'il n'a
      pas de cible
      ' Avant je ne testais pas si  DossPasPack_f  mais depuis Cocoa, AcceptFileD
      rop à folder ne suffit plus
    End If

    If f_elt = Nil Then ' Vu que Accept FileType.Folder  on ne devrait pas voir autre
    chose qu'un dossier (sauf sous Windows)
      Beep
      rep = MyDialogBox(Txt_DropQueDoss, "", Btn_Ok, "", "", 0)
    Else
      ActButtSource(f_elt)
    End If

  Else
    Beep

  End If

End Sub
```

WinMain Control PushButtSource:

```
Sub Action()

  Dim CeDossTrash as FolderItem

  If Keyboard.AsyncAltKey Then ' and  ' Keyboard.AsyncOSKey = Keyboard.AsyncCom
  mandKey
    If PasNil_Existe_Alias(DossierSource, False) Then ' On pourrait mettre True car ç
    a n'est surement pas un alias
      If Keyboard.AsyncCommandKey Then
        CeDossTrash = DossierSource.Volume_f.Child(App.NomDossHistTrash
        )
        If PasNil_Existe_Alias(CeDossTrash, False) Then ' On pourrait mettre Tr
        ue car ça n'est surement pas un alias
          CeDossTrash.Parent.Launch ' Launch le dossier parent donc l'ouvr
          e dans le Finder, et ne place que cette fenêtre au 1er plan, pas to
          utes celles du Finder comme
          '  Activate d'AppleScript. Mais le Reveal d'AppleScript ci-dessous n
          e fait qu'ouvrir la fenêtre du dossier, ça ne la place pas au 1er pla
          n
          RevealThisItem(CeDossTrash.ShellPath_fAS) ' Mieux que ci-dessus
          , en plus d'ouvrir la fenêtre du dossier, ça sélectionne l'élément d
          ans la fenêtre, n'existe pas dans RealBasic
        Else
          Beep
        End If

      Else
        DossierSource.Parent.Launch ' Launch le dossier parent donc l'ouvre d
        ans le Finder, et ne place que cette fenêtre au 1er plan, pas toutes cell
        es du Finder comme
        '  Activate d'AppleScript. Mais le Reveal d'AppleScript ci-dessous ne fai
        t qu'ouvrir la fenêtre du dossier, ça ne la place pas au 1er plan
        RevealThisItem(DossierSource.ShellPath_fAS) ' Mieux que ci-dessus, en
         plus d'ouvrir la fenêtre du dossier, ça sélectionne l'élément dans la fe
        nêtre, n'existe pas dans RealBasic
      End If

    Else
      Beep
    End If
```

```
  Else
    ActButtSource(Nil)

  End If

End Sub
```

WinMain Control GroupBoxCible:

```
Sub DropObject(obj As DragItem, action As Integer)

  Dim f_elt, e_elt as FolderItem
  Dim rep as Int16

  ' Voir note dans  EnableMenuBoutons  On ne peut pas annuler un  AcceptFileDrop
  If DrapMenu Then ' Ou  If not(App.SyncEnCours or App.LogSEnCours) Then  Mais c'e
  st plus logique d'utiliser même variable que pour menu et bouton Choisir
    f_elt = Nil ' Inutile mais …
    ' MsgBox "obj.FolderItemAvailable : " + CStr(obj.FolderItemAvailable)
    If obj.FolderItemAvailable Then ' AcceptFileDrop à folder
      e_elt = Obj.FolderItem
      ' MsgBox "e est alias : " + CStr(e_elt.Alias) + EndOfLine + e_elt.AbsPath_f
      If e_elt.Alias Then e_elt = e_elt.GetTargetItem_f ' Mais bon, on ne peut pas
      drag drop des Alias de toute façon, à cause des AcceptFileType
      If (not(obj.NextItem or e_elt.Alias)) and e_elt.DossPasPack_f Then f_elt = e_
      elt ' Il ne doit y avoir qu'un élément, et si c'est resté un Alias, c'est qu'il n'a
      pas de cible
      ' Avant je ne testais pas si  DossPasPack_f  mais depuis Cocoa, AcceptFileD
      rop à folder ne suffit plus
    End If

    If f_elt = Nil Then ' Vu que Accept FileType.Folder  on ne devrait pas voir autre
    chose qu'un dossier (sauf sous Windows)
      Beep
      rep = MyDialogBox(Txt_DropQueDoss, "", Btn_Ok, "", "", 0)
    Else
      ActButtCible(f_elt)
    End If

  Else
    Beep
```

```
  End If

End Sub
```

WinMain Control PushButtCible:

```
Sub Action()

  Dim CeDossTrash as FolderItem

  If Keyboard.AsyncAltKey Then ' and  ' Keyboard.AsyncOSKey = Keyboard.AsyncCom
  mandKey
    If PasNil_Existe_Alias(DossierCible, False) Then ' On pourrait mettre True car ça
    n'est surement pas un alias
      If Keyboard.AsyncCommandKey Then
        CeDossTrash = DossierCible.Volume_f.Child(App.NomDossHistTrash)
        If PasNil_Existe_Alias(CeDossTrash, False) Then ' On pourrait mettre Tr
        ue car ça n'est surement pas un alias
          CeDossTrash.Parent.Launch ' Launch le dossier parent donc l'ouvr
          e dans le Finder, et ne place que cette fenêtre au 1er plan, pas to
          utes celles du Finder comme
          '  Activate d'AppleScript. Mais le Reveal d'AppleScript ci-dessous n
          e fait qu'ouvrir la fenêtre du dossier, ça ne la place pas au 1er pla
          n
          RevealThisItem(CeDossTrash.ShellPath_fAS) ' Mieux que ci-dessus
          , en plus d'ouvrir la fenêtre du dossier, ça sélectionne l'élément d
          ans la fenêtre, n'existe pas dans RealBasic
        Else
          Beep
        End If

      Else
        DossierCible.Parent.Launch ' Launch le dossier parent donc l'ouvre dan
        s le Finder, et ne place que cette fenêtre au 1er plan, pas toutes celles
        du Finder comme
        '  Activate d'AppleScript. Mais le Reveal d'AppleScript ci-dessous ne fai
        t qu'ouvrir la fenêtre du dossier, ça ne la place pas au 1er plan
        RevealThisItem(DossierCible.ShellPath_fAS) ' Mieux que ci-dessus, en
        plus d'ouvrir la fenêtre du dossier, ça sélectionne l'élément dans la fen
        être, n'existe pas dans RealBasic
      End If

    Else
```

```
      Beep
    End If

  Else
    ActButtCible(Nil)

  End If

End Sub
```

WinMain Control RadioButtSyncMode:

```
Sub Action(index as Integer)

  ' MsgBox str(Index)

  If App.initProg Then
    ChangeConfig(-1)
  End If '  initProg

End Sub
```

WinMain Control CheckBoxGererDate:

```
Sub Action()

  If App.initProg Then
    ChangeConfig(-1)
  End If '  initProg

End Sub
```

WinMain Control CheckBoxRemplRecent:

```
Sub Action()

  If App.initProg Then
    ChangeConfig(-1)
  End If '  initProg
```

```
End Sub
```

WinMain Control CheckBoxTrashIfBegin:

```
Sub Action()

  If App.initProg Then
    ChangeConfig(-1)
  End If '  initProg

End Sub

Sub DropObject(obj As DragItem, action As Integer)

  ' Voir note dans  EnableMenuBoutons  On ne peut pas annuler un  AcceptFileDrop
  ' Plus de DragDrop sur EditField car insère le texte quand on y lâche des fichiers lie
  ns ou Clipboard car contiennent du texte

  If DrapMenu Then ' Ou  If not(SyncEnCours or App.LogSEnCours) Then  Mais c'est pl
  us logique d'utiliser même variable que pour menu et bouton Choisir
    If obj.FolderItemAvailable Then ' On accepte "any"
      Do
        App.AddTxtTrash(Obj.FolderItem, GTextTrash, CheckBoxTrashIfBegin.
        Value)
      Loop Until (not obj.NextItem) or App.DragDropAlerte
    End If

  Else
    Beep

  End If

  If App.DragDropAlerte Then App.DragDropAlerte = False ' Inutile d'utiliser DragDro
  pTicks

End Sub
```

WinMain Control EditFieldTrashBegin:

```
Sub TextChange()
```

```
  Dim iNbre, rep as Int16 ' Quasiment la même chose que WinPrefs.EditFieldNomHistT
  rash
  Dim DrapErr as Boolean
  Const MaxCaracts = 6
  Const CarAccept = """!#$%&'()*+,-/ 0123456789;<=>?@ABCDEFGHIJKLMNOPQRST
  UVWXYZabcdefghijklmnopqrstuvwxyz[\]_{|}~†°¢£§•¶ß®©™≠ÆØ∞±≤≥¥µ∂∑∏π∫ªº
  Ωæø¿¡¬√ƒ≈Δ«»… Œœ-—“”‘’÷◇€"
  ' Je n'autorise pas le point .  Tous ces caractères n'ont pas de caractères combinés, i
  ls font tous 1 de Len(). Il y a espace normal et espace insécable

  If App.initProg Then
    If Me.Text = "" Then ' Len(Me.Text) = 0
      DrapErr = False
    ElseIf (Me.Text = " ") or (Len(Me.Text) > MaxCaracts) Then ' Je n'autorise pas esp
    ace seul parce que trop risqué d'effacer élément commençant par espace car il
    peut y avoir des erreurs
      DrapErr = True
    Else
      DrapErr = False
      For iNbre = 1 to Len(Me.Text)
        If InStr(CarAccept, Mid(Me.Text, iNbre, 1)) = 0 Then
          DrapErr = True
          Exit
        End If
      Next iNbre
    End If

    If DrapErr Then ' Il y a un caractère interdit
      Beep
      rep = MyDialogBox(FracTexte(Txt_CarAccept, 1) + str(MaxCaracts) + FracT
      exte(Txt_CarAccept, 2) + CarAccept, "", Btn_Ok, "", "", 0)
      Me.Text = GTextTrash ' On remet à la valeur d'avant
    Else
      If GTextTrash = "" Then CheckBoxTrashIfBegin.Value = True ' On le checke
      automatiquement si était vide avant
      GTextTrash = Me.Text
      ' If Me.Text = "" Then CheckBoxTrashIfBegin.Value = False ' Mais pas forcé
      ment checké si non vide
      ChangeConfig(-1)
    End If

  End If '  initProg

End Sub
```

WinMain Control EditFieldIgnorElt:

```
Sub TextChange()

  Dim CarNonAccept, CeCaract, TampText as String ' Je pourrais utiliser  CarAccept  d
  e EditFieldTrashBegin  mais non, si jamais le gars veut entrer le nom d'un dossier a
  vec des caractères à la con
  Dim iNbre, Fin_iNbre as Int16 ' On ignorera les dossiers portant ces noms, et si on r
  entre un nom avec un . on ignorera les fichiers portant ce nom

  If App.initProg Then
    CarNonAccept = chr(0) + chr(9) + chr(10) + chr(13) ' Ca foutrait la merde dans
    la sauvegarde des prefs
    Fin_iNbre = Len(CarNonAccept)
    #If DebugBuild Then
      If not(Len(SepAbsPath) = 1) Then MsgBox "Problème Tom, Dans TextChang
      e EditFieldIgnorElt on prend Right(..., 1) alors que SepAbsPath fait " + str(L
      en(SepAbsPath)) + " de long."
    #EndIf

    TampText = Me.Text
    For iNbre = 1 to Fin_iNbre ' Je pourrais peut-être boucler direct sur CeCaract
      CeCaract = Mid(CarNonAccept, iNbre, 1)
      TampText = ReplaceAll(TampText, CeCaract, "") ' ReplaceAll(sourceString,
      oldString, newString)
    Next iNbre
    While not(Instr(TampText, SepAbsPath + SepAbsPath) = 0) ' Sinon supprime pas
     plusieurs ::::::: si on colle le texte depuis le presse-papiers
      TampText = ReplaceAll(TampText, SepAbsPath + SepAbsPath, SepAbsPath
      )
    Wend
    If Left(TampText, 1) = SepAbsPath Then
      TampText = Mid(TampText, 2)
    End If
    ' If Right(TampText, 1) = SepAbsPath Then ' NON car c'est chiant quand on veu
    t ajouter un  :  puis du texte
    ' TampText = Left(TampText, Len(TampText) - 1) ' Alors j'enlève les  :  à la fin
    ailleurs
    ' End If

    If (Ubound(MemIgnorEltNom) + Ubound(MemIgnorEltExt)) = -2 Then CheckBoxI
    gnorElt.Value = True ' On le checke automatiquement si était vide avant
```

```
    iNbre = Len(Me.Text)
    If Right(Me.Text, 1) = SepAbsPath Then iNbre = iNbre - 1 ' Car il n'y aura pas d
    e  SepAbsPath  à la fin ci-dessous
    TabIgnorElts(TampText, True)
    TampText = TextIgnorElts(True)
    ' If not((Me.Text = TampText) or (Me.Text = (TampText + SepAbsPath))) Then
    If not(iNbre = Len(TampText)) Then ' C'est qu'on a supprimé des  :  ou carréme
    nt un nom ou une extension en double
      Beep
      Me.Text = TampText
      ' Else
      '  Rien, même si l'ordre changera plus tard (extensions à la fin) je laisse tel
      quel car c'est chiant ça remet le curseur au début
    End If
    ChangeConfig(-1)

  End If '  initProg

End Sub
```

WinMain Control CheckBoxIgnorElt:

```
Sub DropObject(obj As DragItem, action As Integer)

  Dim TampText as String ' Voir note dans  EnableMenuBoutons  On ne peut pas annu
  ler un  AcceptFileDrop
  ' Plus de DragDrop sur EditField car insère le texte quand on y lâche des fichiers lie
  ns ou Clipboard car contiennent du texte
  '  Quoique là il n'y ait pas de problème comme avec  TrashIfBegin  mais bon

  If DrapMenu Then ' Ou  If not(SyncEnCours or App.LogSEnCours) Then  Mais c'est pl
  us logique d'utiliser même variable que pour menu et bouton Choisir
    If obj.FolderItemAvailable Then ' On accepte "any"
      Do
        If Keyboard.AsyncAltKey Then ' Option
          TampText = Obj.FolderItem.Extension_f
        Else
          TampText = Obj.FolderItem.Name
        End If
        If not(TampText = "") Then ' Si le gars demande l'extension d'un dossie
        r -> Pas d'extension
          If not((EditFieldIgnorElt.Text = "") or (Right(EditFieldIgnorElt.Text,
          1) = SepAbsPath)) Then EditFieldIgnorElt.Text = EditFieldIgnorElt.
          Text + SepAbsPath
```

```
            '                    Voir notes dans  EditFieldIgnorElt.TextChange
            , Il peut déjà y avoir : à la fin
            ' Mais comme l'event  EditFieldIgnorElt.TextChange  va se faire il l'
            enléverait
            EditFieldIgnorElt.Text = EditFieldIgnorElt.Text + TampText ' Chan
            geConfig(-1) fait dans  EditFieldIgnorElt.TextChange
            ' Else
            ' Beep ' Je pourrais mais faudrait mettre un flag et ne faire qu'un b
            eep à la fin
          End If
        Loop Until (not obj.NextItem)
      End If

    Else
      Beep

    End If

End Sub

Sub Action()

    If App.initProg Then
      ChangeConfig(-1)
    End If '  initProg

End Sub
```

WinMain Control CheckBoxSyncIc:

```
Sub Action()

    If App.initProg Then
      ChangeConfig(-1)
    End If '  initProg

End Sub
```

WinMain Control CheckBoxSimu:

```
Sub Action()
```

```
  If App.initProg Then ' A été changé par clique sur la CheckBox
    ChangeConfig(-1)
    If (not DrapMenu) or App.SyncEnCours Then MsgBox "Error !"+EndOfLine+"Pleas
    e contact the authors. CheckBoxSimu.Action"+EndOfLine+EndOfLine+"DrapMe
    nu = " + Cstr(DrapMenu) + " should be True" + EndOfLine +"App.SyncEnCours
    = " + Cstr(App.SyncEnCours) + " should be False"
    App.CaptSyncSimu(Me.Value, False, True) ' Pas test pour texte affiché car  Push
    ButtSync.Caption  et FichierSynchro.Text  car justement on les change
    '                                          Test pour Enabled car doivent avoir été m
    is comme il faut dans ChangeConfig appelé plus haut

  Else ' A été changé par  ChangeConfig  via  PopupConfig  ou via  ThreadSync
    If DrapMenu or ((not DrapMenu and App.SyncEnCours)) Then ' On a changé cett
    e CheckBox depuis  ChangeConfig  appelé soit depuis  PopupConfig  soit depui
    s  ThreadSync
      App.CaptSyncSimu(Me.Value, False, False) ' Pas test pour texte affiché car
      PushButtSync.Caption  et FichierSynchro.Text  car justement on les change
      '                                           Pas test pour Enabled car on n'a pas
      encore appelé  App.EnableMenuBoutons  (et UpdtSyncEnab)   Sera appelé à
       la fin de  ChangeConfig
    Else ' On vient d'initialiser cette CheckBox depuis ChangeConfig appelé depuis
    WinMain.Open
      App.CaptSyncSimu(Me.Value, True, False) ' Test pour texte affiché car  Push
      ButtSync.Caption  et FichierSynchro.Text  doivent déjà être aux bonnes val
      eurs car on a fait  CaptSyncSimu  au début de WinMain.Open
      '                                         Pas test pour Enabled car on n'a pas
      encore appelé  App.EnableMenuBoutons  (et UpdtSyncEnab)
    End If

  End If '  initProg

  ' Note : Désactivé si App.LogSEnCours

End Sub
```

WinMain Control CheckBoxPreNiv:

```
Sub Action()

  If App.initProg Then
    ChangeConfig(-1)
  End If '  initProg
```

```
End Sub
```

WinMain Control PopupBarOut:

```
Sub Change()

  If App.initProg Then
    ChangeConfig(-1)
  End If '  initProg

End Sub
```

WinMain Control EditFieldPasIgnor:

```
Sub TextChange()

  Dim CarNonAccept, CeCaract, TampText as String ' Je pourrais utiliser  CarAccept  d
  e EditFieldTrashBegin  mais non, si jamais le gars veut entrer le nom d'un dossier a
  vec des caractères à la con
  Dim iNbre, Fin_iNbre as Int16 ' On n'ignorera PAS les éléments portant ces noms (uti
  le car par défaut on ignore elts invisibles, commençant par . etc.)

  If App.initProg Then
    CarNonAccept = chr(0) + chr(9) + chr(10) + chr(13) ' Ca foutrait la merde dans
    la sauvegarde des prefs
    Fin_iNbre = Len(CarNonAccept)
    #If DebugBuild Then
      If not(Len(SepAbsPath) = 1) Then MsgBox "Problème Tom, Dans TextChang
      e EditFieldPasIgnor on prend Right(..., 1) alors que SepAbsPath fait " + str(
      Len(SepAbsPath)) + " de long."
    #EndIf

    TampText = Me.Text
    For iNbre = 1 to Fin_iNbre ' Je pourrais peut-être boucler direct sur CeCaract
      CeCaract = Mid(CarNonAccept, iNbre, 1)
      TampText = ReplaceAll(TampText, CeCaract, "") ' ReplaceAll(sourceString,
      oldString, newString)
    Next iNbre
    While not(Instr(TampText, SepAbsPath + SepAbsPath) = 0) ' Sinon supprime pas
     plusieurs ::::::: si on colle le texte depuis le presse-papiers
      TampText = ReplaceAll(TampText, SepAbsPath + SepAbsPath, SepAbsPath
      )
```

```
    Wend
    If Left(TampText, 1) = SepAbsPath Then
      TampText = Mid(TampText, 2)
    End If
    ' If Right(TampText, 1) = SepAbsPath Then ' NON car c'est chiant quand on veu
    t ajouter un  :  puis du texte
    ' TampText = Left(TampText, Len(TampText) - 1) ' Alors j'enlève les  :  à la fin
    ailleurs
    ' End If

    If Ubound(MemPasIgnor) = -1 Then CheckBoxPasIgnor.Value = True ' On le che
    cke automatiquement si était vide avant
    iNbre = Len(Me.Text)
    If Right(Me.Text, 1) = SepAbsPath Then iNbre = iNbre - 1 ' Car il n'y aura pas d
    e  SepAbsPath  à la fin ci-dessous
    TabIgnorElts(TampText, False)
    TampText = TextIgnorElts(False)
    ' If not((Me.Text = TampText) or (Me.Text = (TampText + SepAbsPath))) Then
    If not(iNbre = Len(TampText)) Then ' C'est qu'on a supprimé des  :  , je pourrais
     tester  Me.Text = TampText +? SepAbsPath  mais bon je fais comme EditFieldI
    gnorElt
      Beep
      Me.Text = TampText
      ' Else
      '  Rien
    End If
    ChangeConfig(-1)

  End If '  initProg

End Sub
```

WinMain Control CheckBoxPasIgnor:

```
Sub DropObject(obj As DragItem, action As Integer)

  Dim TampText as String ' Voir note dans  EnableMenuBoutons  On ne peut pas annu
  ler un  AcceptFileDrop
  ' Plus de DragDrop sur EditField car insère le texte quand on y lâche des fichiers lie
  ns ou Clipboard car contiennent du texte
  '  Quoique là il n'y ait pas de problème comme avec  TrashIfBegin  mais bon

  If DrapMenu Then ' Ou  If not(SyncEnCours or App.LogSEnCours) Then  Mais c'est pl
  us logique d'utiliser même variable que pour menu et bouton Choisir
```

```
    If obj.FolderItemAvailable Then ' On accepte "any"
      Do
        TampText = Obj.FolderItem.Name
        If not(TampText = "") Then ' Normalement impossible
          If not((EditFieldPasIgnor.Text = "") or (Right(EditFieldPasIgnor.Text
          , 1) = SepAbsPath)) Then EditFieldPasIgnor.Text = EditFieldPasIgn
          or.Text + SepAbsPath
          '                       Voir notes dans  EditFieldPasIgnor.TextChange
          , Il peut déjà y avoir : à la fin
          ' Mais comme l'event  EditFieldPasIgnor.TextChange  va se faire il
          l'enléverait
          EditFieldPasIgnor.Text = EditFieldPasIgnor.Text + TampText ' Cha
          ngeConfig(-1) fait dans  EditFieldPasIgnor.TextChange
          ' Else
          ' Beep ' Je pourrais mais faudrait mettre un flag et ne faire qu'un b
          eep à la fin
        End If
      Loop Until (not obj.NextItem)
    End If

  Else
    Beep

  End If

End Sub

Sub Action()

  If App.initProg Then
    ChangeConfig(-1)
  End If '  initProg

End Sub
```

WinMain Control PushButtSync:

```
Sub Action()

  ActButtSync

End Sub
```

WinMain Control BevButtAffLog:

```
Sub Action()

   ' Me.Value sera changé dans les events open et close de WinLog

   If Me.Value Then ' Si Sync journal en cours ( App.LogSEnCours  à vrai), ça affichera d
   ialogue du CancelClose
      ' Me.Value = False ' Fait dans event Close de WinLog
      WinLog.Close ' Voir TimerBatch si on modifie des trucs ici lors de la fermeture
   Else ' Voir menu WinLogMenu
      ' Me.Value = True ' Fait dans event Open de WinLog
      ' If App.SyncEnCours Then ThreadSync.Suspend ' Car il pourrait y avoir des lou
      pés dans le journal si la synchro continue,
      '  des fichiers seront copiés et ajoutés au journal alors qu'on n'aura pas fini  Wi
      nLog.AfficherJournal
      WinLog.Show
      ' If App.SyncEnCours Then ThreadSync.Resume
   End If

End Sub
```

WinMain Control PopupConfig:

```
Sub Change()

   If App.initProg Then
      ChangeConfig(Me.ListIndex)
   End If '  initProg

End Sub
```

WinMain Control BevButtAddConfig:

```
Sub Action()

   Dim iLign, Fin_CfNconfig, TampInt, NoConfig, rep as Int16

   NomConfig = NomDerConfig
   Do
```

```
WinNomConfig.ShowModal ' Within Non car ce n'est pas une Sheet Window
If NomConfig = "" Then ' Annulé
   rep = 2 ' Pour sortir de la boucle (Annuler)

Else
   Fin_CfNconfig = UBound(CfNomConfig)
   NoConfig = Fin_CfNconfig + 1 ' Ou n'importe quel autre CfDossSource etc.
   For iLign = 0 To Fin_CfNconfig ' La boucle s'arrêtera AVANT si on efface un
    Réglage dans ce  For Next
      If CfNomConfig(iLign) = NomConfig Then
         If NoConfig = (Fin_CfNconfig + 1) Then ' C'est le premier réglage
         qu'on trouve avec ce nom
            NoConfig = iLign
         Else ' Putain c'est quoi ce bordel il y a déjà un réglage avec ce no
         m !!!  Impossible (bidouille des prefs)
            BeepNbT(3)
            TampInt = PopupConfig.ListIndex
            EffaceReglage(iLign) ' Je l'efface Na, l'avait qu'à pas bidouiller
            Pref
            App.initProg = False
            FairePopupConfig
            If TampInt = iLign Then ' On pointait sur celui effacé du mêm
            e nom que  NoConfig
               TampInt = NoConfig
            ElseIf TampInt > iLign Then ' Note :  TampInt  ne peut pas êtr
            e > à  Fin_CfNconfig = UBound(CfNomConfig)
               TampInt = TampInt - 1
            End If
            PopupConfig.ListIndex = TampInt
            App.initProg = True
            If BevButtAffBatch.Value Then WinBatch.FaireListe
            ' Note : Il était impossible qu'il n'y ait plus qu'un Réglage puis
            que 2 avaient le même nom
         End If
      End If
   Next iLign

   If NoConfig = (Fin_CfNconfig + 1) Then ' Ce réglage n'existait pas, on l'ajo
   ute
      CfNomConfig.Append NomConfig
      Fin_CfNconfig = Fin_CfNconfig + 1 ' = Ubound(CfNomConfig)
      CfDossSource.Append StTextLire(StTextDossSource.Text)
      CfDossCible.Append StTextLire(StTextDossCible.Text)
      TampInt = -1 ' On aura une Nil objection si problème
```

```
    For iLign = 0 to 2
      If RadioButtSyncMode(iLign).Value Then TampInt = iLign
    Next iLign
    CfModeSync.Append TampInt
    CfGererDate.Append CheckBoxGererDate.Value
    CfRemplRecent.Append CheckBoxRemplRecent.Value
    CfTrashIfBegin.Append CheckBoxTrashIfBegin.Value
    CfTextTrash.Append GTextTrash ' EditFieldTrashBegin.Text
    CfIgnorElt.Append CheckBoxIgnorElt.Value
    CfPremNiv.Append CheckBoxPreNiv.Value
    CfIgnorEltList.Append TextIgnorElts(True) ' EditFieldIgnorElt.Text SANS
     l'éventuel  SepAbsPath  à la fin
    CfPasIgnor.Append CheckBoxPasIgnor.Value
    CfPasIgnorList.Append TextIgnorElts(False) ' EditFieldPasIgnor.Text SA
    NS l'éventuel  SepAbsPath  à la fin
    CfSyncIcones.Append CheckBoxSyncIc.Value
    CfBarOutils.Append PopupBarOut.ListIndex
    CfSimul.Append CheckBoxSimu.Value
    CfMargeTps.Append MemMargeTps ' WinRegl.EdFieldMargeTps.Text
    CfDossPackFich.Append WinRegl.CheckBoxDossPack.Value
    CfDossExtFich.Append WinRegl.CheckBoxDossExt.Value
    CfUseShell.Append WinRegl.CheckBoxUseShell.Value
    CfCdeCpShell.Append WinRegl.TextFieldCpShell.Text
    CfCopyRBerr.Append WinRegl.CheckBoxCopyRBerr.Value
    CfBatch.Append False

    App.initProg = False
    PopupConfig.AddRow NomConfig
    PopupConfig.ListIndex = NoConfig
    App.initProg = True
    NomDerConfig = NomConfig ' Puisque  ChangeConfig  n'a pas été app
    elé
    If BevButtAffBatch.Value Then WinBatch.FaireListe
    rep = 4 ' Pour sortir de la boucle (Ajouter)

  Else ' Ce réglage existe
    Beep
    rep = MyDialogBox(FracTexte(Txt_ReglageExiste, 1) + NomConfig + F
    racTexte(Txt_ReglageExiste, 2), "", Btn_Remplacer, Btn_Annuler, Btn_R
    ecommencer, 0)
    If rep = 1 Then ' Remplacer
      RemplReglage(NoConfig) ' Utile de faire ci-dessous car on n'était r
      evenu sur Réglages par défaut
      App.initProg = False
```

```
            PopupConfig.ListIndex = NoConfig
            App.initProg = True
            NomDerConfig = NomConfig ' Puisque  ChangeConfig  n'a pas été
             appelé
            If BevButtAffBatch.Value Then WinBatch.FaireListe ' Utile aussi car
            peut-être changé  ListBoxBatch (checkbox de la liste Batch)

            ' Else  On sortira de la boucle ou on recommencera suivant  rep
          End If '  rep
        End If ' NoConfig = (UBound(CfNomConfig) + 1)  Réglage existait ou non
      End If '  NomConfig = ""
    Loop Until (not(rep = 3)) ' Recommencer

End Sub
```

WinMain Control BevButtDelConfig:

```
Sub Action()

    Dim rep as Int16

    If PopupConfig.ListIndex = 0 Then ' On ne peut pas supprimer les réglages par défa
    ut
        MsgBox "Error ! Please contact the authors." + EndOfLine + "You can't remove t
        he default settings! This button should be disabled!"

    Else
        rep = MyDialogBox(FracTexte(Txt_SuppReglage, 1) + PopupConfig.Text + Frac
        Texte(Txt_SuppReglage, 2), "", Btn_Supp, Btn_Annuler, "", 0)
        '       Ou                                          CfNomConfig(PopupConfig.ListInd
        ex)
        If rep = 1 Then ' Supprimer
            RemplReglage(0) ' On sauvegarde comme défaut les réglages actuels
            EffaceReglage(PopupConfig.ListIndex)
            App.initProg = False
            FairePopupConfig
            App.initProg = True
            PopupConfig.ListIndex = 0 ' InitProg à True pour remettre à jour Me.Enable
            d à False. Avant : initProg  reste à False pour cette instruction car pas beso
            in de remettre à jour
            If BevButtAffBatch.Value Then WinBatch.FaireListe
        End If
```

```
  End If

End Sub
```

WinMain Control BevButtInvDoss:

```
Sub Action()

  Dim f_elt as FolderItem
  Dim TampTextSource, TampTextCible as String

  TampTextSource = StTextLire(StTextDossSource.Text)
  TampTextCible = StTextLire(StTextDossCible.Text)

  f_elt = DossierSource
  DossierSource = DossierCible
  DossierCible = f_elt

  UpdtSyncEnab(True) ' On a affiché les StaticText dans  UpdtSyncEnab  car (True)
  If not DrapSyncEnab Then
    If not((TampTextSource = StTextLire(StTextDossCible.Text)) and (TampTextCibl
    e = StTextLire(StTextDossSource.Text))) Then Beep
    ' Je fais le test ci-dessus car on a pu échanger un dossier avec vide auquel cas
    pas de Beep même si ReVerifDossSetC False
  End If

  ChangeConfig(-1)

End Sub
```

WinMain Control Socket_Ver_MaJ:

```
Sub Error(code as integer)

  Dim rep as Int16

  If AlertVerAjour Then ' Pas d'alerte si check update au démarrage
    Beep
    rep = MyDialogBox(Txt_Ver_MaJ_impo, "Server Error " + str(code), Btn_Ok, "", "",
     1)
  Else
```

```
    AlertVerAjour = True
  End If

End Sub

Sub PageReceived(url as string, httpStatus as integer, headers as internetHeaders, conten
t as string)

  Dim TexteRecu, TexteTitre, TexteSiteWeb, TexteVersionNo, TexteHistoric, MaJverEo
  F as String
  Dim rep, LEoF as Int16

  MaJverEoF = EndOfLine ' Je pourrais mettre .Macintosh mais cela marcherait-il bien
  sous Windows ?
  LEof = Len(MaJverEoF) ' Normalement = 1 sous Mac et 2 sous Windows
  TexteRecu = DefineEncoding(content, DefAppEncod) ' = Encodings.UTF8  Avant je
  mettais GetTextEncoding(&h0201)) ' De ' "ISOLatin1" = "ISO-8859-1"
  ' TexteRecu = ConvertEncoding(TexteRecu, App.DefAppEncod) ' = Encodings.UTF8 '
   Pas forcément utile avant et de toute façon inutile maintenant
  TexteRecu = ReplaceLineEndings(TexteRecu, MaJverEoF) ' Pas sûr que ce soit utile
  mais pour être sûr de trouver EndOfLine
  TexteRecu = NthField(TexteRecu, "Older versions", 1) ' NthField(source, separator, fi
  eldNumber)
  ' NthField retourne une chaîne vide si fieldnumber en dehors du champs
  ' Après "Old Version" il y a l'historique des autres versions et on s'en fou ici
  TexteTitre = NthField(TexteRecu, MaJverEoF, 1) ' 1ère ligne
  TexteSiteWeb = NthField(TexteRecu, MaJverEoF, 2) ' 2ème ligne
  TexteVersionNo = NthField(TexteRecu, MaJverEoF, 3) ' 3ème ligne
  TexteHistoric = Mid(TexteRecu, Len(TexteTitre) + LEof + Len(TexteSiteWeb) + LEof
  + Len(TexteVersionNo) + LEof + 1) ' Tout ce qui suit
  While Right(TexteHistoric, LEoF) = MaJverEoF
    TexteHistoric = Left(TexteHistoric, Len(TexteHistoric) - LEoF)
  Wend ' On a retiré tous les sauts de lignes en fin de texte
  ' MsgBox """ + TexteTitre + """ + EndOfLine + """ + TexteSiteWeb + """ + EndOfLine
  + """ + TexteVersionNo + """ + EndOfLine + """ + TexteHistoric + """

  If (Left(TexteTitre, 13) = "Release Notes") and (Left(TexteSiteWeb, 7) = "http://") and
  (Left(TexteVersionNo, 2) = "v.") Then
    ' On pourrait vérifier que  TexteTitre  est suivi de  App.NomProg  etc. mais bon
    … on a l'URL avec son nom
    If TexteVersionNo = App.ShortVersion Then ' App.VersionProg
      If AlertVerAjour Then ' Pas d'alerte si check update au démarrage
        rep = MyDialogBox(Txt_Ver_a_jour, TexteVersionNo, Btn_Ok, Btn_Hist
        oric, "", 0) + 1 ' Pour correspondre à ci-dessous
```

```
      Else
        rep = 2
        AlertVerAjour = True
      End If
    Else ' On dit qu'elle est plus récente mais en fait elle est différente, mais faudrai
    t être con pour remettre une
      If not AlertVerAjour Then
        AlertVerAjour = True
        Beep ' On fait un Beep car vérification auto, pas demandé via menu Vé
        rif si version à jour
      End If
      rep = MyDialogBox(Txt_Ver_Nouv, TexteVersionNo + EndOfLine + TexteHi
      storic, Btn_SiteWeb, Btn_Fermer, Btn_Historic, 0) '  ancienne version ;-)
    End If
    If rep = 1 Then
      ShowURL(TexteSiteWeb)
    ElseIf rep = 3 Then
      ShowURL(url) ' Celle qu'on vient de télécharger
      ' Else ' rep = 2 rien
    End If

  Else
    ' Si doc non trouvé ça renvoie "!< Doc has moved etc." ou alors j'ai merdé dans
    l'écriture de mon fichier texte d'historique
    If AlertVerAjour Then ' Pas d'alerte si check update au démarrage
      Beep
      rep = MyDialogBox(Txt_Ver_MaJ_impo, "", Btn_Ok, "", "", 1)
    Else
      AlertVerAjour = True
    End If

  End If

End Sub
```

WinMain Control CanvPtitRond:

```
Sub Paint(g As Graphics, areas() As REALbasic.Rect)

  ' Colorie le petit rond en vert si les noms des dossiers Sources et Cibles sont identi
  ques, en orange si différents, en blanc sinon
  ' Cette Method doit être appelé chaque fois que l'on modifie  StaticTextDoss  Sourc
  e ou Cible
```

```
  ' Beep
  ' Pause(20, False)
  If PasNil_Existe_Alias(DossierSource, True) and PasNil_Existe_Alias(DossierCible, Tru
  e) Then
    If DossierSource.Name = DossierCible.Name Then
      g.DrawPicture PtitPtVert, 0, 0
    Else
      g.DrawPicture PtitPtJaune, 0, 0
    End If
  Else
    g.DrawPicture PtitPtGris, 0, 0
  End If

  ' Note : Je convertie encodage car si un des dossier est lu depuis les prefs et que je
  viens de choisir l'autre (bouton ou
  '  DragDrop) je n'aurais pas l'égalité pour un même nom comportant des accents
  ' Problème réglé différement avec ma fonction GetFitemAbsPath qui retransforme (é
   -> e´)

End Sub
```

WinMain Control BevButtAffBatch:

```
Sub Action()

  If Me.Value Then
    Me.Value = False
    WinBatch.Visible = False
    WinBatch.DrapBatchEnab = False
    ' Me.Value = False ' Fait dans event Close de WinBatch
    ' AffDrawWin("WinBatch", False) ' WinBatch.Close
    #If DebugBuild Then
      Dim iLigne, FinLigne as Int16

      FinLigne = UBound(CfNomConfig) - 1 ' Ou n'importe quel autre, Ligne 0 co
      rrespond à réglage 1 puisque réglage 0 est le réglage Dernière Config
      For iLigne = 0 to FinLigne
        If not(WinBatch.ListBoxBatch.CellCheck(iLigne, 0) = CfBatch(iLigne + 1)
        ) Then MsgBox "Erreur Tom. WinBatch.Close" + EndOfLine + Cstr(WinB
        atch.ListBoxBatch.CellCheck(iLigne, 0)) + " ≠ " + Cstr(CfBatch(iLigne +
         1))
        If not(WinBatch.ListBoxBatch.Cell(iLigne, 1) = CfNomConfig(iLigne + 1)
        ) Then MsgBox "Erreur Tom. WinBatch.Close" + EndOfLine + WinBatch.
```

```
            ListBoxBatch.Cell(iLigne, 1) + " ≠ " + CfNomConfig(iLigne + 1)
        Next iLigne
        ' MsgBox "On fermeWinBatch"
      #EndIf
    Else
      Me.Value = True
      ' WinBatch.RePosFenetr(False, False) ' Aura pour effet de l'ouvrir comme si  Win
      Batch.Show
      WinBatch.Visible = True
      WinBatch.DrapBatchEnab = True
      #If DebugBuild Then
        ' MsgBox "Self.Height = " +str(Self.Height)
        If not ScrollBarWM.Visible Then ' Sinon c'est normal que c'est la merde
          ' If WinBatch.Top <> (Self.Top + (636 - WinBatch.Height)) Then MsgBo
          x "Pb Tom, les boutons Synchroniser et Batch Sync ne seront pas align
          és."
          If WinBatch.ListBoxBatch.Width <> (PopupConfig.Width + 11) Then Ms
          gBox "Pb Tom, ListBoxBatch.Width devrait faire " + str(PopupConfig.Wi
          dth + 11) + " pour que même largeur que PopupMenuConfig."
        End If
      #EndIf
      ' Me.Value = True ' Fait dans event Open de WinBatch
      ' AffDrawWin("WinBatch", True)
    End If

    WinBatch.UpdtBatchEnabTim(False, True) ' On remet le Timer à On ou Off suivant le
    cas
    App.EnableMenuBoutons(False) ' Pour FichierBatchSync , pas forcément utile car le
    menu se mettra automatiquement à jour dès qu'on l'affichera, mais je préfère qu'il
    soit à jour

End Sub
```

WinMain Control BevButtAffRegl:

```
Sub Action()

    If Me.Value Then
      Me.Value = False
      WinRegl.Visible = False
      ' Me.Value = False ' Fait ci-dessous
      ' AffDrawWin("WinRegl", False) ' WinRegl.Visible = False
    Else
```

```
    Me.Value = True
    ' WinRegl.RePosFenetr(False, False)
    WinRegl.Visible = True
    ' Me.Value = True ' Fait ci-dessous
    ' AffDrawWin("WinRegl", True)
  End If

End Sub
```

WinMain Control CanvSyncMode:

```
Sub Paint(g As Graphics, areas() As REALbasic.Rect)

  ' Dessine le mode choisi, cette méthode doit être appelée à chaque modification de
  RadioButtSyncMode

  ' Beep
  ' Pause(20, False)
  If RadioButtSyncMode(0).Value Then '  Réciproque
    g.DrawPicture SyncMode0P, 0, 0

  ElseIf RadioButtSyncMode(1).Value Then '  Réciproque
    g.DrawPicture SyncMode1P, 0, 0

  ElseIf RadioButtSyncMode(2).Value Then '  Réciproque
    g.DrawPicture SyncMode2P, 0, 0

  Else ' @@@ NON car comme ça fait Refresh sur le Canvas ça rappelle cette méthode
  et ça tourne en boucle
    MsgBox "Error !" + EndOfLine + "Please contact the authors. CanSyncMode" + E
    ndOfLine + EndOfLine + "Which sync mode ?"

  End If

End Sub
```

WinMain Control TimerFinSync:

```
Sub Action()

  ' Je fini mes Threads par un timer car si j'Enable et je remet tout (menus, boutons, e
  tc.) à leur valeur dans le Thread ça peut changer en même temps dans un event
```

```
  ' En lançant un Timer en fin de Thread, je bloque les events

  ' Fin du Thread : ThreadSync

  ProgWheelSync.Visible = False
  If ArretUrg and (not(Me.Period = 1000)) Then
    Me.Period = 1000 ' Je ne le fais pas à la fin de L_ThreadTrait car parfois on re-c
    liquait le bouton pendant l'attente Timer
    Me.Mode = Timer.ModeSingle ' On relance le Timer avec une pause plus longue
     comme ça les events (pleins de clics sur bouton Stop) ont le temps de se faire
  Else
    If App.OuvLogApS Then
      WinLog.Show ' Même si déjà affiché je la remet au 1er plan, et si ne l'était
      pas on fait AfficherJournal
      WinLog.PushButtLogSync.Enabled = not(WinLog.NbCellsCheck = 0) ' and (U
      Bound(App.JournCol1) > -1) ' Forcément si NbCellsCheck > 0
    End If
    DrapMenu = True
    App.SyncEnCours = False
    App.EnableMenuBoutons(True)
    If ArretUrg Then
      ArretUrg = False
      If not(Me.Period = 1000) Then Me.Period = 10 ' Val par défaut, à faire tout
      à la fin car Me.Mode est encore à Timer.ModeSingle (1), et si le code restan
      t à exécuter est plus long que 10 ms ça relance le Timer
    End If
  End If

End Sub
```

WinMain Control ScrollBarWM:

```
Sub ValueChanged()

  RedimWin(False, Me.Value)

End Sub

End Class
```

Class WinLog

```
Inherits Window

Private Const CellSsCheckBox = " "

Private Const CoulNoir = &c000000

Private Const CoulOrang = &cFF7F00

Private Const CoulRouge = &cFF0000

Private Const CoulVert = &c00C800

Private Const CoulViolet = &c7F007F
```

WinLog.Activate:

```
Sub Activate()

   App.FenetrFront = Me
   ' A plus sa place dans event Paint  WinMain.CoulPtitRond ' Car si on masque SyncTw
   oFolders le petit rond disparaît

   App.CheckWinMenu(False)

End Sub
```

WinLog.CancelClose:

```
Function CancelClose(appQuitting as Boolean) As Boolean

   Dim DrapAnnulFerme as Boolean

   If App.LogSEnCours Then
      Beep
      If MyDialogBox(Txt_SyncEnCours, "", Btn_Cont, Btn_Arreter, "", 0) = 2 Then ' Ret
      ourne 1 pour Cont et 2 pour Arreter
         WinMain.ArretUrg = True ' Note : Le Thread est en pause pendant l'attente
         que l'utilisateur clique un bouton de la DialogBox
         If App.Sfx Then
            Zoum.Play ' Pour signifier prise en compte  ArretUrg     On ne l'entend
            s pas si coup de frein juste après, mais des fois
```

```
      While Zoum.IsPlaying '                                    que ce soit plu
      s long pour stopper tout
      Wend
    End If
    ' While App.LogSEnCours ' Ca marche dans WinMain mais pas là ??? Sans d
    oute parce que dans le thread je lis ListBoxLog
    ' Wend '                                                   Même sans DoEvents
    Pause(10, False) ' Pour éviter 2 appuie trop vif
    ' If App.Sfx Then Coup_de_freins.Play ' A été fait dans le Thread de Synchro
  End If
  DrapAnnulFerme = True ' NON, le gars refermera un seconde fois : App.LogSEn
  Cours ' On ne fermera pas si synchro journal toujours en cours, si n'a pas eu le
  temps de s'arrêter avec pause ci-dessus

ElseIf App.SyncEnCours Then ' Voir note dans  WinMain.Synchro
  App.TampCloseLog = DrapMenu or (not appQuitting) ' = (not((not DrapMenu) a
  nd appQuitting)) Note :  DrapMenu  est False puisque Synchro en cours
  ' Si  App.TampCloseLog  était déjà à True on s'en fou, c'est qu'on vient ici en fe
  rmant via WinMain.Synchro,  App.TampCloseLog  sera remis à False dedans
  ' MsgBox "On ne ferme pas WinLog car peut-être qu'écriture journal en cours.
  On demandera sa fermeture dans Synchro." + EndOfLine + "On a demandé fer
  meture depuis WinMain.Synchro : " + Cstr(TampClose) + EndOfLine + "On va d
  emander fermeture depuis WinMain.Synchro : " + Cstr(App.TampCloseLog)
  DrapAnnulFerme = (not TampClose) ' TampClose  est à False à l'ouverture de
  WinLog  et il y reste pour Canceler la fermeture jusqu'à ce qu'on ferme WinLog
  depuis WinMain.Synchro
  '                                  On demande fermeture du Log depuis Synchro pour
  la gérer et éviter ainsi que sa fermeture ne se fasse pendant l'écriture d'une lig
  ne du journal

Else ' Note : Si  App.LogSEnCours  et  App.SyncEnCours  à False alors  DrapMenu  es
t sûrement à vrai, sauf si une Movable Windows est ouverte
  ' MsgBox "On ferme WinLog si False : " + Cstr(((not DrapMenu) and appQuitting
  ) + EndOfLine + "DrapMenu = " + Cstr(DrapMenu) + EndOfLine + "appQuitting
  = " + Cstr(appQuitting)
  DrapAnnulFerme = ((not DrapMenu) and appQuitting) ' Si on retourne Vrai la fe
  nêtre ne se fermera pas
  ' Voir  WinMain.CancelClose  Note : On n'utilise pas  App.DragDropOuv  car cett
  e fenêtre Log pas ouverte si DragDropOuv
  ' Si on quitte par le Dock on annulerait la fermeture de WinMain mais pas celle-
  ci qui se fermerait
  '  donc on perdrait les coordonnées de la fenêtre
End If

If DrapAnnulFerme and (TimerCellCheck.Mode = Timer.ModeSingle) Then TimerCell
Check.Mode = Timer.ModeOff ' Puisqu'on ferme la fenêtre, sinon le timer se fera ju
```

```
    ste après la fin de cet event
    Return DrapAnnulFerme

    ' Explication sur  ((not DrapMenu) and appQuitting)  : Si on est en train de fermer al
    ors que  appQuitting  est True et que  DrapMenu  à False, c'est forcément
    '  qu'on a quitté par le Dock vu que le menu Quitter est désactivé

End Function
```

WinLog.Close:

```
Sub Close()

    ' Avant je suspendais ici ThreadSync puis je le relançais plus bas mais je ne fais plu
    s comme ça car ça déconnait. Il fallait faire gaffe à ne pas suspendre ce Thread si
    '  on avait demandé la fermeture depuis le Thread (et non en cliquant le bouton Fer
    mer de la fenêtre) car le Thread ne redémarrait jamais

    App.WLogL = Self.Left
    App.WLogT = Self.Top
    App.WLogW = Self.Width
    App.WLogH = Self.Height

    App.WLogLargCol = ListBoxLog.ColumnWidths

    App.WLogTailText = PopupTailleTextListe.ListIndex

    App.WLogEncIndex = PopupEncodSrc.ListIndex

    WinMain.BevButtAffLog.Value = False ' Il était forcément à Vrai puisque cette fenêtre
     était ouverte
    App.CheckWinMenu(True)

End Sub
```

WinLog.KeyDown:

```
Function KeyDown(Key As String) As Boolean

    ' Même chose dans chaque  event KeyDown  des autres fenêtres et que dans ActBut
    tSync
```

```
  If App.SyncEnCours or App.LogSEnCours Then
    ' If (Keyboard.AsynckeyDown(&h35) or (Key = Chr(27))) and (not WinMain.Arret
    Urg) Then ' Touche Escape , pour pas faire 2 fois Zoum (procédure Synchro)
    ' Touche  Escape dans les 2 cas, je faisais les 2 tests !
    If (Key = Chr(27)) and (not WinMain.ArretUrg) Then ' Touche Escape , pour pas f
    aire 2 fois Zoum (procédure Synchro)
      WinMain.ArretUrg = True
      If App.Sfx Then
        Zoum.Play ' Pour signifier prise en compte  ArretUrg     On ne l'entend
        s pas si coup de frein juste après, mais des fois
        While Zoum.IsPlaying '                                   que ce soit plu
        s long pour stopper tout
        Wend
      End If
      ' NON, NE PAS faire boucle ci-dessous, sinon cet event (clic sur Bouton Syn
      chro/Arrêter) ne se fait QU'à la fin de la sychro, donc redémarre et arrête a
      ussi sec
      ' While (App.SyncEnCours or App.LogSEnCours) ' Si ça marche !  (NE PAS fai
      re ça car garde la main et le Thread ne se fini pas)
      ' Wend '                                                Même sans DoEvents
      Pause(10, False) ' Pour éviter 2 appuie trop vif
      ' If App.Sfx Then Coup_de_freins.Play ' A été fait dans le Thread de Synchro
    End If
  End If

  Return (not(Key = Chr(9))) ' Car si Tab on passe d'un TextField à l'autre

End Function
```

WinLog.Open:

```
Sub Open()

  DecPosCtrl(Self) ' Décalage des controls ( Label , TextField , Slider , etc. )
  #If TargetWin32 Then ' Note : Je suspend le Thread depuis le Bouton AffLog
    PushButtFermer.TextSize = 12
    PushButtEnregF.TextSize = 12
    PushButtLogSync.TextSize = 12
  #EndIf

  ' MsgBox "Début WinLog.Open"

  ' Replacement de la fenêtre
```

```
If (App.WLogL < 0) Or (App.WLogT < 38) or (App.WLogW < Self.MinWidth) or (App.W
LogH < Self.MinHeight) or ((App.WLogL + App.WLogW) > Screen(0).Width) or ((App.
WLogT + App.WLogH) > Screen(0).Height) Then
   App.WLogL = 20
   App.WLogT = 60
   App.WLogW = 740 ' Voir  WinLog.MinWidth
   App.WLogH = 500 ' Voir  WinLog.MinWidth
End If
Self.Left = App.WLogL
Self.Top = App.WLogT
Self.Width = App.WLogW
Self.Height = App.WLogH

ListBoxLog.ColumnWidths = App.WLogLargCol

#If DebugBuild Then
   If not(Val(PopupTailleTextListe.Text) = ListBoxLog.TextSize) Then MsgBox "Pop
   upTailleTextListes et Listes.TextSize doivent correspondrent Tom !"
   If not(Val(PopupTailleTextListe.Text) = EditFieldPath.TextSize) Then MsgBox "Po
   pupTailleTextListes et EditFieldPath.TextSize doivent correspondrent Tom !"
#EndIf
PopupTailleTextListe.ListIndex = App.WLogTailText ' ListBoxLog.TextSize et EditFiel
dPath.TextSize mis à leur valeur dans Change de

FairePopupEnc(False)
PopupEncodSrc.ListIndex = App.WLogEncIndex

WinMain.BevButtAffLog.Value = True ' Il était forcément à Faux puisque cette fenêtr
e était fermée
' App.CheckWinMenu(False) ' Sera appelé dans event Activate

EcrEdFieldLog(WinMain.JournalDd, Format(WinMain.NbErrRemplAnc, F_MBillionAvEs
p), WinMain.JournalDf)
EditFieldLog.HelpTag = WinMain.JournalD_HT + EndOfLine + EndOfLine + EditField
Log_HT

AffMasqHelpTag("", "") ' APRES avoir défini le helptag ci-dessus

NbCellsCheck = 0
AfficherJournal(True)

If DrapMenu Then ' Ou not App.SyncEnCours  Si App.LogSEnCours alors on ne peut
pas être ici car cette fenêtre est forcément ouverte
   ListBoxLog.Enabled = True
```

```
    BevButtCheck.Enabled = True ' and (UBound(App.JournCol1) > -1) ou not(WinM
    ain.Journal = "")
    PushButtEnregF.Enabled = True ' and (UBound(App.JournCol1) > -1) ou not(Win
    Main.Journal = "")
    PushButtLogSync.Enabled = not(NbCellsCheck = 0) ' and (UBound(App.JournCol
    1) > -1) ' Forcément si NbCellsCheck > 0  A faire après  AfficherJournal  pour a
    ctualiser  NbCellsCheck
  End If
  ' Les autres boutons sont Enabled par défaut, le bouton Fermer reste actif

  ' MsgBox "Fin WinLog open"

End Sub
```

WinLog.AbsPathSiExist:

```
Private Function AbsPathSiExist(TestElt as FolderItem) As String

  Dim TestElt_AbsPath as String ' Cette procédure retourne l'AbsoluthPath d'un item,
  mais retourne "" s'il est à Nil (au lieu de planter) ou si n'existe pas
  ' J'ai remarqué que parfois ça plante si un élément n'existe pas.
  ' ATTENTION, Ne Pas mettre  Extends TestElt as FolderItem  car plante si on fail  Nil.
  AbsPathSiExist  , il faut faire  AbsPathSiExist(Nil)
  ' Dans SyncTwoFolders, j'ai un tableau qui mémorise les FolderItems. Si je déplace u
  n des éléments du tableau à la corbeille, puis vide la corbeille, l'elt n'existe plus

  If TestElt = Nil Then
    TestElt_AbsPath = ""
  Else
    If TestElt.Exists Then
      TestElt_AbsPath = TestElt.AbsPath_f
    Else ' Car ça me fait planter l'appli, je n'ai même pas de Nil Objection mais la ro
    ue multicolor (si  TestElt  n'existe pas)
      TestElt_AbsPath = ""
    End If
  End If

  Return TestElt_AbsPath

End Function
```

WinLog.AfficherJournal:

```
Sub AfficherJournal(DrapFaireJ as Boolean)

  Dim iLign, DerLign as Int32 ' Pas UInt16 si Ubound -1 car vide

  DerLign = UBound(App.JournCol1)
  ' MsgBox "DrapFaireJ = " + CStr(DrapFaireJ) + EndOfLine + "DerLign = " + str(DerLi
  gn)
  #If DebugBuild Then
    If not((DerLign = UBound(App.JournCol3)) and (DerLign = UBound(App.JournCol
    4)) and (DerLign = UBound(App.JournCol5b)) and (DerLign = UBound(App.Journ
    Col5)) and (DerLign = UBound(App.JournCol6)) and (DerLign = UBound(App.Jou
    rnCol7b)) and (DerLign = UBound(App.JournCol7))) Then
      MsgBox "Tu t'es planté dans Journal Tom, pas même nb d'éléments dans c
      haque :" + EndOfLine + str(DerLign) + " = " + str(UBound(App.JournCol3))
      + " = " + str(UBound(App.JournCol4)) + " = " + str(UBound(App.JournCol5b
      )) + " = " + str(UBound(App.JournCol5)) + " = " + str(UBound(App.JournCol
      6)) + " = " + str(UBound(App.JournCol7b)) + " = " + str(UBound(App.JournC
      ol7)) ' @
    End If
  #EndIf

  If DrapFaireJ Then ' Liste vide, on rempli la liste avec le Journal
    If not(ListBoxLog.ListCount = 0) Then MsgBox "Error ! Please contact the author
    s." + EndOfLine + "AfficherJournal, ListBoxLog should be empty!"

    For iLign = 0 to DerLign ' Si Ubound = -1 (pas de journal), on n'entre pas dans l
    a boucle
      FaireLigne(App.JournCol1(iLign), iLign, App.JournCol3(iLign), App.JournCol
      4(iLign), App.JournCol5(iLign), App.JournCol6(iLign), App.JournCol7(iLign))
      ' Normalement la liste est trié dans l'ordre habituel mais on s'en fout car d
      e toute façon on écrit toute la ligne donc pas de décalage possible
    Next iLign

    ' For iLign = (DerLign + 1) to (DerLign + 30) ' Pour test, pour remplir le journal
    ' FaireLigne("NeRienFaire", iLign, "b", "c", "d", "e")
    ' Next iLign

  End If

  ' On trie la liste suivant colonne 1
  ListBoxLog.HeadingIndex = 1 ' On montre que c'est sur cette colonne que s'effectue
   le tri
  ListBoxLog.ColumnSortDirection(1) = ListBoxLog.SortAscending ' On défini ordre tri
   de cette colonne
```

```
  ListBoxLog.SortedColumn = 1 ' On trie cette colonne par ordre défini ci-dessus
  ListBoxLog.Sort ' Liste créée dans l'ordre plus haut mais nécessaire quand même si non la liste reste triée par
  '  rapport à la colonne et au sens de tri précédemment sélectionné
  ListBoxLog.Refresh ' Car quand j'appele ceci depuis un Thread j'ai l'impression que des fois ça ne rafraichi pas bien quand il faut. J'avais une erreur au test ci-dessous ( je l'ai encore ?)
  ' Self.UpdateNow ' J'ai fait Refresh ci-dessus

  If not(ListBoxLog.ListCount = (DerLign + 1)) Then MsgBox "Error ! Please contact the authors." + EndOfLine + "AfficherJournal, ListBoxLog should have " + str(DerLign + 1) + " lines!" + EndOfLine + "DrapFaireJ : " + Cstr(DrapFaireJ)

End Sub
```

WinLog.AffMasqHelpTag:

```
Sub AffMasqHelpTag(CeCtrlNomId as String, CeCtrlHlpTg as String)

  Dim iNbre, Fin_iNbre as Int16 ' Envoyer  RectControl.Name  avec éventuellement son Index  (si tableau de RadioButton par exemple) pour ne mettre que lui à jour
  Dim TampNom as String '              Exemple : AffMasqHelpTag(StaticTextNbSpam.Name, "TextDuHelpTag")
  Dim CeCtrl as Control '            ou si index 1 : AffMasqHelpTag(StaticTextNbSpam.Name + "1", "TextDuHelpTag") ' ou  + str(index)
  Dim CeRectCtrl as RectControl
  Static MemRctCtrlNom(-1), MemHlpTgTxt(-1) as String

  Fin_iNbre = ControlCount - 1
  If UBound(MemHlpTgTxt) = -1 Then ' = UBound(MemRctCtrlNom)
    ' MsgBox "On mémorise HelpTag de " + str(Fin_iNbre + 1) + " controls." + EndOfLine + "CeCtrlNomId = '" + CeCtrlNomId + "'"
    #If DebugBuild Then
      If not(CeCtrlNomId = "") Then MsgBox "Tu t'es gourré Tom, CeCtrlNomId devrait être vide " !" + EndOfLine + "'" + CeCtrlNomId + "'"
    #EndIf
    For iNbre = 0 to Fin_iNbre
      CeCtrl = Self.Control(iNbre)
      ' MsgBox CeCtrl.Name + EndOfLine + str(CeCtrl.Index) + EndOfLine + "RectControl ? : " + CStr(CeCtrl isa RectControl)
      If CeCtrl isa RectControl Then
        CeRectCtrl = RectControl(CeCtrl) ' Et NON RectControl(iNbre)
        TampNom = CeCtrl.Name
```

```
        #If DebugBuild Then
          If (not(CeRectCtrl.Index = CeCtrl.Index) and (CeRectCtrl.Name = T
          ampNom)) Then MsgBox "Bizarre Tom, RectCtrl.Index ≠ Ctrl.Index
           :" + EndOfLine + str(CeRectCtrl.Index) + " ≠ " + str(CeCtrl.Index)
          + EndOfLine + "RectCtrl.Name ≠ Ctrl.Name :" + EndOfLine + str(C
          eRectCtrl.Name) + " ≠ " + str(TampNom)
        #EndIf
        If CeCtrl.Index > -1 Then TampNom = TampNom + str(CeCtrl.Index) '
        ATTENTION Int32, si pas un tableau de control alors = -2147483648
        ' MsgBox " id = " + str(CeCtrl.Index) + " = " + str(CeRectCtrl.Index) +
        EndOfLine + TampNom
        MemRctCtrlNom.Append TampNom
        MemHlpTgTxt.Append CeRectCtrl.HelpTag ' On mémorise qu'il y en ait
         ou non
        If not App.AffHelpTag Then CeRectCtrl.HelpTag = ""
        ' Si à Vrai alors l'helptag reste à sa valeur
      End If ' RectControl
    Next iNbre

Else ' On les a déjà mémorisés
    ' MsgBox "Nbre de Controls : " + str(Fin_iNbre + 1) + EndOfLine + "Nbre de Rec
    tControls : " + str(UBound(MemHlpTgTxt) + 1) + EndOfLine + "CeCtrlNomId = '
    " + CeCtrlNomId + "'"
    If CeCtrlNomId = "" Then ' On applique à tous
      For iNbre = 0 to Fin_iNbre ' Pas forcément = à  UBound(MemHlpTgTxt)  car
       tous ne sont pas RectControl, mais QUE Control
        CeCtrl = Self.Control(iNbre)
        ' MsgBox CeCtrl.Name + EndOfLine + str(CeCtrl.Index) + EndOfLine +
        "RectControl ? : " + CStr(CeCtrl isa RectControl)
        If CeCtrl isa RectControl Then
          CeRectCtrl = RectControl(CeCtrl) ' Et NON RectControl(iNbre)
          ' MsgBox str(CeRectCtrl.Index) + " = " + EndOfLine + str(CeCtrl.In
          dex) + EndOfLine + CeRectCtrl.Name + EndOfLine + CeRectCtrl.H
          elpTag
          If App.AffHelpTag Then
            CeRectCtrl.HelpTag = MemHlpTgTxt(iNbre) ' S'il n'y en avait p
            as ça reste à ""
          Else
            CeRectCtrl.HelpTag = ""
          End If
        End If ' RectControl
      Next iNbre

    Else ' On applique qu'au control indiqué
```

```
        iNbre = MemRctCtrlNom.IndexOf(CeCtrlNomId)
        If iNbre = -1 Then
          MsgBox "Error ! Please contact the authors, AffMasqHelpTag :" + EndO
          fLine + CeCtrlNomId + EndOfLine + "iNbre = -1"
        Else
          CeCtrl = Self.Control(iNbre)
          ' MsgBox CeCtrlNomId + EndOfLine + "iNbre = " + str(iNbre)
          ' Forcément : If CeCtrl isa RectControl Then
          CeRectCtrl = RectControl(CeCtrl) ' Et NON RectControl(iNbre)
          MemHlpTgTxt(iNbre) = CeCtrlHlpTg ' On met à jour la mémoire du hel
          ptag
          If App.AffHelpTag Then
            CeRectCtrl.HelpTag = CeCtrlHlpTg ' On met à jour le helptag lui-
            même
          Else
            If not(CeRectCtrl.HelpTag = "") Then MsgBox  "Error ! Please contac
            t the authors, AffMasqHelpTag :" + EndOfLine + CeCtrlNomId + "'
            helptag should be empty " :" + EndOfLine + """ + CeRectCtrl.HelpT
            ag + """
          End If
        End If

      End If ' On applique à tous ou au contrôle indiqué  CeCtrlNomId

    End If ' UBound(MemHlpTgTxt) = -1

End Sub
```

WinLog.EcrEdFieldLog:

```
Sub EcrEdFieldLog(CeTexteD as String, CeNbErrT as String, CeTexteF as String)

    ' MsgBox Cstr(EditFieldLog.SelTextColor) + ", " + Cstr(EditFieldLog.TextColor) + End
    OfLine + str(App.JourDcoulPos(0)) + ", " + str(App.JourDcoulPos(1)) + ", " + str(App.
    JourDcoulPos(2)) + ", " + str(App.JourDcoulPos(3)) + ", " + str(App.JourDcoulPos(4))

    EditFieldLog.Text = CeTexteD + CeNbErrT + CeTexteF

    If App.CoulDsListLog Then
      Select Case App.JourDcoulPos(0)
      Case 1 ' ok vert
        EditFieldLog.SelStart = 0
        EditFieldLog.SelLength = App.JourDcoulPos(1)
```

```
      EditFieldLog.SelTextColor = CoulVert

    Case 2 ' pb rouge
      EditFieldLog.SelStart = 0
      EditFieldLog.SelLength = App.JourDcoulPos(1)
      EditFieldLog.SelTextColor = CoulRouge

      ' Else ' 0 Rien, le texte reste noir

    End Select

    Select Case App.JourDcoulPos(2)
    Case 1 ' ok vert
      EditFieldLog.SelStart = App.JourDcoulPos(3)
      EditFieldLog.SelLength = App.JourDcoulPos(4) + Len(CeNbErrT) + 3 ' Voir i
      nit  JournalDf  dans  WinMain.Open
      EditFieldLog.SelTextColor = CoulVert

    Case 2 ' pb rouge
      EditFieldLog.SelStart = App.JourDcoulPos(3)
      EditFieldLog.SelLength = App.JourDcoulPos(4) + Len(CeNbErrT) + 3 ' Voir i
      nit  JournalDf  dans  WinMain.Open
      EditFieldLog.SelTextColor = CoulRouge

      ' Else ' 0 Rien, le texte reste noir

    End Select

    EditFieldLog.SelStart = 0 ' Réinitialise sinon ça peut merder le coup d'après
    EditFieldLog.SelLength = 0 ' Déselectionne le texte
    EditFieldLog.SelTextColor = CoulNoir
  End If

End Sub
```

WinLog.FaireLigne:

```
Sub FaireLigne(Lcol1 as String, iLign as Int32, Lcol3 as String, Lcol4 as String, Lcol5 as Str
ing, Lcol6 as String, Lcol7 as String)

  ListBoxLog.AddRow "" ' Ne marche pas d'ajouter direct  True ou False

  #If DebugBuild Then
```

```
    If not(iLign = ListBoxLog.LastIndex) Then MsgBox "iLign = " + str(iLign) + EndO
    fLine + "ListBoxLog.LastIndex = " + str(ListBoxLog.LastIndex)
  #EndIf

  If Lcol1 = "NeRienFaire" Then ' Donc erreur ou opération déjà réalisée, impossible d
  e faire une opération sur cette ligne
    ListBoxLog.CellCheck(iLign, 0) = False ' L'est par défaut mais je préfère
    ' ListBoxLog.CellCheck(iLign, 0).Enabled = False  Impossible
    ListBoxLog.CellType(iLign, 0) = ListBoxLog.TypeNormal ' 1 ' Si on écrit 0 c'est C
    olumnType soit 2
    ListBoxLog.Cell(iLign, 0) = CellSsCheckBox ' Je met un espace car c'est impossi
    ble de lire CellType  pour CompareRow
  Else
    ListBoxLog.CellCheck(iLign, 0) = False ' Ou True ou voir pour mettre BevButtCh
    eck  comme valeur par défaut mais c'est bof
    ' ATTENTION : NbCellsCheck = NbCellsCheck + 1  à compter si mis à vrai ci-de
    ssus
    ' ListBoxLog.Cell(iLign, 0) = "" ' Inutile
  End If
  ' MsgBox "ListBoxLog.CellType(iLign, 0) = " + str(ListBoxLog.CellType(iLign, 0))

  ListBoxLog.Cell(iLign, 1) = Format(iLign + 1, F_MBillionSsEsp) + " " ' Lcol2

  ListBoxLog.Cell(iLign, 2) = Lcol3

  ListBoxLog.Cell(iLign, 3) = Lcol4

  ListBoxLog.Cell(iLign, 4) = Lcol5

  ListBoxLog.Cell(iLign, 5) = Lcol6

  ListBoxLog.Cell(iLign, 6) = Lcol7

  ' Ca ne fait pas grande différence de mettre en gras. Pour comparer faire : If (iLign
  mod 2) = 0 Then
  ' ListBoxLog.CellBold(iLign, 2) = True
  ' ListBoxLog.CellBold(iLign, 5) = True

End Sub
```

WinLog.FairePopupEnc:

```
Private Sub FairePopupEnc(TampDrap as Boolean)
```

```
' EXACTEMENT même Methods dans MyPopBarrier, TextBatchConv, MemoDate et Sy
ncTwoFolders  sauf App.EncodFich
If TampDrap Then MsgBox "Error ! Please contact the authors." + EndOfLine + "Faire
PopupEnc" + EndOfLine + "TampDrap is True and it should be False !"

' ATTENTION, toute modification doit être fait en corrélation avec  CreerTabEncod
pour que ça corresponde
PopupEncodSrc.DeleteAllrows
' Coller ce qui suit depuis le fichier Excel 'Liste Encodage.xlsx' dans 'Communs'
PopupEncodSrc.AddRow "UTF8" ' n° 0
PopupEncodSrc.AddRow "UTF16" ' n° 1
PopupEncodSrc.AddRow "UTF16BE" ' n° 2
PopupEncodSrc.AddRow "UTF16LE" ' n° 3
PopupEncodSrc.AddRow "UTF32" ' n° 4
PopupEncodSrc.AddRow "UTF32BE" ' n° 5
PopupEncodSrc.AddRow "UTF32LE" ' n° 6
PopupEncodSrc.AddRow "ASCII" ' n° 7
PopupEncodSrc.AddRow "DOSArabic" ' n° 8
PopupEncodSrc.AddRow "DOSBalticRim" ' n° 9
PopupEncodSrc.AddRow "DOSCanadianFrench" ' n° 10
PopupEncodSrc.AddRow "DOSChineseSimplif" ' n° 11
' PopupEncodSrc.AddRow "DOSChineseTrad"
PopupEncodSrc.AddRow "DOSCyrillic" ' n° 12
PopupEncodSrc.AddRow "DOSGreek" ' n° 13
PopupEncodSrc.AddRow "DOSGreek1" ' n° 14
PopupEncodSrc.AddRow "DOSGreek2" ' n° 15
PopupEncodSrc.AddRow "DOSHebrew" ' n° 16
PopupEncodSrc.AddRow "DOSIcelandic" ' n° 17
' PopupEncodSrc.AddRow "DOSJapanese"
' PopupEncodSrc.AddRow "DOSKorean"
PopupEncodSrc.AddRow "DOSLatin1" ' n° 18
PopupEncodSrc.AddRow "DOSLatin2" ' n° 19
PopupEncodSrc.AddRow "DOSLatinUS" ' n° 20
PopupEncodSrc.AddRow "DOSNordic" ' n° 21
PopupEncodSrc.AddRow "DOSPortuguese" ' n° 22
PopupEncodSrc.AddRow "DOSRussian" ' n° 23
' PopupEncodSrc.AddRow "DOSThai"
PopupEncodSrc.AddRow "DOSTurkish" ' n° 24
PopupEncodSrc.AddRow "ISOLatin1" ' n° 25
PopupEncodSrc.AddRow "ISOLatin2" ' n° 26
PopupEncodSrc.AddRow "ISOLatin3" ' n° 27
PopupEncodSrc.AddRow "ISOLatin4" ' n° 28
PopupEncodSrc.AddRow "ISOLatin5" ' n° 29
```

```
PopupEncodSrc.AddRow "ISOLatin6" ' n° 30
PopupEncodSrc.AddRow "ISOLatin7" ' n° 31
PopupEncodSrc.AddRow "ISOLatin8" ' n° 32
PopupEncodSrc.AddRow "ISOLatin9" ' n° 33
PopupEncodSrc.AddRow "ISOLatinArabic" ' n° 34
PopupEncodSrc.AddRow "ISOLatinCyrillic" ' n° 35
PopupEncodSrc.AddRow "ISOLatinGreek" ' n° 36
PopupEncodSrc.AddRow "ISOLatinHebrew" ' n° 37
PopupEncodSrc.AddRow "KOI8_R" ' n° 38
PopupEncodSrc.AddRow "MacArabic" ' n° 39
PopupEncodSrc.AddRow "MacArmenian" ' n° 40
PopupEncodSrc.AddRow "MacBengali" ' n° 41
PopupEncodSrc.AddRow "MacBurmese" ' n° 42
PopupEncodSrc.AddRow "MacCeltic" ' n° 43
PopupEncodSrc.AddRow "MacCentralEurRoman" ' n° 44
PopupEncodSrc.AddRow "MacChineseSimp" ' n° 45
PopupEncodSrc.AddRow "MacChineseTrad" ' n° 46
PopupEncodSrc.AddRow "MacCroatian" ' n° 47
PopupEncodSrc.AddRow "MacCyrillic" ' n° 48
PopupEncodSrc.AddRow "MacDevanagari" ' n° 49
PopupEncodSrc.AddRow "MacDingbats" ' n° 50
PopupEncodSrc.AddRow "MacEthiopic" ' n° 51
PopupEncodSrc.AddRow "MacExtArabic" ' n° 52
PopupEncodSrc.AddRow "MacGaelic" ' n° 53
PopupEncodSrc.AddRow "MacGeorgian" ' n° 54
PopupEncodSrc.AddRow "MacGreek" ' n° 55
PopupEncodSrc.AddRow "MacGujarati" ' n° 56
PopupEncodSrc.AddRow "MacGurmukhi" ' n° 57
PopupEncodSrc.AddRow "MacHebrew" ' n° 58
PopupEncodSrc.AddRow "MacIcelandic" ' n° 59
PopupEncodSrc.AddRow "MacJapanese" ' n° 60
PopupEncodSrc.AddRow "MacKannada" ' n° 61
PopupEncodSrc.AddRow "MacKhmer" ' n° 62
PopupEncodSrc.AddRow "MacKorean" ' n° 63
PopupEncodSrc.AddRow "MacLaotian" ' n° 64
PopupEncodSrc.AddRow "MacMalayalam" ' n° 65
PopupEncodSrc.AddRow "MacMongolian" ' n° 66
PopupEncodSrc.AddRow "MacOriya" ' n° 67
PopupEncodSrc.AddRow "MacRoman" ' n° 68
PopupEncodSrc.AddRow "MacRomanian" ' n° 69
' PopupEncodSrc.AddRow "MacRomanLatin1"
PopupEncodSrc.AddRow "MacSinhalese" ' n° 70
PopupEncodSrc.AddRow "MacSymbol" ' n° 71
PopupEncodSrc.AddRow "MacTamil" ' n° 72
```

```
PopupEncodSrc.AddRow "MacTelugu" ' n° 73
PopupEncodSrc.AddRow "MacThai" ' n° 74
PopupEncodSrc.AddRow "MacTibetan" ' n° 75
PopupEncodSrc.AddRow "MacTurkish" ' n° 76
PopupEncodSrc.AddRow "MacVietnamese" ' n° 77
' PopupEncodSrc.AddRow "ShiftJIS"
' PopupEncodSrc.AddRow "WindowsANSI"
PopupEncodSrc.AddRow "WindowsArabic" ' n° 78
PopupEncodSrc.AddRow "WindowsBalticRim" ' n° 79
PopupEncodSrc.AddRow "WindowsCyrillic" ' n° 80
PopupEncodSrc.AddRow "WindowsGreek" ' n° 81
PopupEncodSrc.AddRow "WindowsHebrew" ' n° 82
PopupEncodSrc.AddRow "WindowsKoreanJohab" ' n° 83
PopupEncodSrc.AddRow "WindowsLatin1" ' n° 84
PopupEncodSrc.AddRow "WindowsLatin2" ' n° 85
PopupEncodSrc.AddRow "WindowsLatin5" ' n° 86
PopupEncodSrc.AddRow "WindowsVietnamese" ' n° 87
' PopupEncodSrc.AddRow "SystemDefault"
' PopupEncodSrc.AddRow "UCS4 (2012r1 and earlier)"

' PopupEncodSrc.AddRow "MacUnicode"
PopupEncodSrc.AddRow "MacFarsi" ' n° 88
' PopupEncodSrc.AddRow "MacUkrainian"
PopupEncodSrc.AddRow "MacInuit" ' n° 89
' PopupEncodSrc.AddRow "MacVT100"
' PopupEncodSrc.AddRow "MacHFS"
' PopupEncodSrc.AddRow "UnicodeDefault"
PopupEncodSrc.AddRow "UnicodeV1_1" ' n° 90
' PopupEncodSrc.AddRow "ISO10646_1933"
' PopupEncodSrc.AddRow "UnicodeV2_0"
' PopupEncodSrc.AddRow "Unicodev2_1"
' PopupEncodSrc.AddRow "ISOLatin1"
' PopupEncodSrc.AddRow "ISOLatin2"
' PopupEncodSrc.AddRow "ISOLatin3"
' PopupEncodSrc.AddRow "ISOLatin4"
' PopupEncodSrc.AddRow "ISOLatinCyrillic"
' PopupEncodSrc.AddRow "ISOLatinArabic"
' PopupEncodSrc.AddRow "ISOLatinGreek"
' PopupEncodSrc.AddRow "ISOLatinHebrew"
' PopupEncodSrc.AddRow "ISOLatin5"
' PopupEncodSrc.AddRow "DOSLatinUS"
' PopupEncodSrc.AddRow "DOSGreek"
' PopupEncodSrc.AddRow "DOSBalticRim"
' PopupEncodSrc.AddRow "DOSLatin1"
```

```
' PopupEncodSrc.AddRow "DOSGreek1"
' PopupEncodSrc.AddRow "DOSLatin2"
' PopupEncodSrc.AddRow "DOSCyrillic"
' PopupEncodSrc.AddRow "DOSTurkish"
' PopupEncodSrc.AddRow "DOCPortuguese"
' PopupEncodSrc.AddRow "DOSIslandic"
' PopupEncodSrc.AddRow "DOSHebrew"
' PopupEncodSrc.AddRow "DOSCanadianFrench"
' PopupEncodSrc.AddRow "DOSArabic"
' PopupEncodSrc.AddRow "DOSNordic"
' PopupEncodSrc.AddRow "DOSRussian"
' PopupEncodSrc.AddRow "DOSGreek2"
' PopupEncodSrc.AddRow "DOSThai"
' PopupEncodSrc.AddRow "DOSJapanese"
' PopupEncodSrc.AddRow "DOSChineseSimplif"
' PopupEncodSrc.AddRow "DOSKorean"
' PopupEncodSrc.AddRow "DOSChineseTrad"
' PopupEncodSrc.AddRow "WindowsLatin1"
' PopupEncodSrc.AddRow "WindowsANSI"
' PopupEncodSrc.AddRow "WindowsLatin2"
' PopupEncodSrc.AddRow "WindowsCryillic"
' PopupEncodSrc.AddRow "WindowsGreek"
' PopupEncodSrc.AddRow "WindowsLatin5"
' PopupEncodSrc.AddRow "WindowsHebrew"
' PopupEncodSrc.AddRow "WindowsArabic"
' PopupEncodSrc.AddRow "WindowsBalticRim"
' PopupEncodSrc.AddRow "WindowsVietnamese"
' PopupEncodSrc.AddRow "WindowsKoreanJohab"
' PopupEncodSrc.AddRow "US_ASCII"
PopupEncodSrc.AddRow "JIS_X0201_76" ' n° 91
' PopupEncodSrc.AddRow "JIS_X0208_83"
PopupEncodSrc.AddRow "JIS_X0208_90" ' n° 92
PopupEncodSrc.AddRow "JIS_X0212_90" ' n° 93
PopupEncodSrc.AddRow "JIS_C6226_78" ' n° 94
PopupEncodSrc.AddRow "GB_2312_80" ' n° 95
' PopupEncodSrc.AddRow "GBK_95"
PopupEncodSrc.AddRow "KSC_5601_87" ' n° 96
' PopupEncodSrc.AddRow "KSC_5601_92_Johab"
' PopupEncodSrc.AddRow "CNS_11643_92_P1"
' PopupEncodSrc.AddRow "CNS_11643_92_P2"
' PopupEncodSrc.AddRow "CNS_11643_92_P3"
PopupEncodSrc.AddRow "ISO_2022_JP" ' n° 97
PopupEncodSrc.AddRow "ISO_2022_JP_2" ' n° 98
PopupEncodSrc.AddRow "ISO_2022_CN" ' n° 99
```

```
  PopupEncodSrc.AddRow "ISO_2022_CN_EXT" ' n° 100
  PopupEncodSrc.AddRow "ISO_2022_KR" ' n° 101
  PopupEncodSrc.AddRow "EUC_JP" ' n° 102
  ' PopupEncodSrc.AddRow "EUC_CN"
  PopupEncodSrc.AddRow "EUC_TW" ' n° 103
  ' PopupEncodSrc.AddRow "EUC_KR"
  ' PopupEncodSrc.AddRow "ShiftJIS"
  ' PopupEncodSrc.AddRow "KOI8_R"
  ' PopupEncodSrc.AddRow "Big5"
  ' PopupEncodSrc.AddRow "MacRomanLatin1"
  PopupEncodSrc.AddRow "HZ_GB_2312" ' n° 104
  PopupEncodSrc.AddRow "NextStepLatin" ' n° 105
  ' PopupEncodSrc.AddRow "EBCDIC_US"
  PopupEncodSrc.AddRow "EBCDIC_CP037" ' n° 106
  ' PopupEncodSrc.AddRow "MultiRun"

  If not(PopupEncodSrc.ListCount = (Ubound(App.EncodFich) + 1)) Then
    MsgBox "Error ! Please contact the authors. FairePopupEnc" + EndOfLine + "Pop
    upEncodSrc.ListCount = " + str(PopupEncodSrc.ListCount) + EndOfLine + "Ubo
    und(EncodFich) + 1 = " + str(Ubound(App.EncodFich) + 1)
  End If

End Sub
```

WinLog.LogSynchro:

```
Private Sub LogSynchro()

  Dim iLign, DerLign, jNbre as Int32 ' Pas UInt16 si Ubound -1 car vide
  Dim SymbText, SymbA, EltSrcPath, EltCibPath, TampSrc, TampCib as String
  Dim DrapErr as Boolean ' Voir notes dans InitSynchro

  DerLign = UBound(App.JournCol1) ' On a vérifié dans  AfficherJournal  si même nb d
  'élément dans  JournCol3, JournCol4, JournCol5, JournCol6, JournCol7, JournCol5b
  et JournCinPath
  '
        et que ListBoxLog.ListCount - 1
  If not((DerLign = UBound(App.LogEltSrc)) and (DerLign = UBound(App.LogEltCib)) an
  d (DerLign = UBound(App.LogDossTrash)) and (DerLign = UBound(App.LogUseShell)
  ) and (DerLign = UBound(App.LogCdeCpShell)) and (DerLign = UBound(App.LogCop
  yRBerr)) and (DerLign = (ListBoxLog.ListCount - 1))) Then
    MsgBox "Error ! Please contact the authors." + EndOfLine + "LogSynchro, ListBo
    xLog have " + str(ListBoxLog.ListCount) + " lines!" + EndOfLine + "UBound(Win
    Main.Journal Col 1) + 1 = " + str(DerLign+1) + "  and -1 =" + EndOfLine + str(
```

```
  UBound(App.LogEltSrc)) + " = " + str(UBound(App.LogEltCib)) + " = " + str(UBo
  und(App.LogDossTrash)) + " = " + str(UBound(App.LogUseShell)) + " = " + str(U
  Bound(App.LogCdeCpShell)) + " = " + str(UBound(App.LogCopyRBerr))
  DerLign = -1 ' Pour ne pas entrer dans la boucle ci-dessous
Else
  If (DerLign = -1) or (NbCellsCheck = 0) Then MsgBox "Error ! Please contact the
   authors." + EndOfLine + "LogSynchro, Synchronize button should be disabled!
  " + EndOfLine + "DerLign = " + str(DerLign) + EndOfLine + "NbCellsCheck = "
  + str(NbCellsCheck)
End If

For iLign = 0 to DerLign ' Si Ubound = -1 (pas de journal), on n'entre pas dans la bo
ucle
  ' Pause(100, False) ' Pour test
  jNbre = Val(ListBoxLog.Cell(iLign, 1)) - 1 ' Indice de la ligne
  ' MsgBox "iLign = " + str(iLign) + EndOfLine + "jNbre = " + str(jNbre)
  ' MsgBox "App.LogOper(" + str(jNbre) + ") = " + str(App.LogOper(jNbre)) + End
  OfLine + AbsPathSiExist(App.LogEltSrc(jNbre)) + EndOfLine + AbsPathSiExist(A
  pp.LogEltCib(jNbre))
  If ListBoxLog.CellCheck(iLign, 0) Then ' Si pas de CheckBox alors à False car je
  mets à False dans ce cas (NE PAS oublier sinon ça peut mémoriser que valeur ét
  ait vrai)
    SymbText = App.JournCol3(jNbre)
    SymbA = Left(SymbText, 1)
    If (SymbText = ListBoxLog.Cell(iLign, 2)) and (SymbA = WinMain.TabSymb(
    WinMain.SymbOpNon)) and (Len(SymbText) = 2) Then
      TampSrc = ListBoxLog.Cell(iLign, 4)
      If TampSrc = App.JournCol5(jNbre) Then
        TampSrc = App.JournCol5b(jNbre) + TampSrc
      Else
        TampSrc = ":" ' Erreur test plus bas
      End If
      TampCib = ListBoxLog.Cell(iLign, 6)
      If TampCib = App.JournCol7(jNbre) Then
        TampCib = App.JournCol7b(jNbre) + TampCib
      Else
        TampCib = ":" ' Erreur test plus bas
      End If
      EltSrcPath = AbsPathSiExist(App.LogEltSrc(jNbre)) ' Voir note dans  Abs
      PathSiExist
      EltCibPath = AbsPathSiExist(App.LogEltCib(jNbre)) ' Voir note dans  Ab
      sPathSiExist
      If (EltSrcPath = TampSrc) and (EltCibPath = TampCib) Then
        SymbText = Right(SymbText, 1) ' car il n'y a que 2 caractères  sino
        n faire  Mid(TampText, 2)
```

```
Select Case App.JournCol1(jNbre)
Case "NeRienFaire"
   DrapErr = True
   MsgBox "Error ! Please contact the authors." + EndOfLine + "L
   ogSynchro, the checkbox of this line shouldn't be checked!"
   + EndOfLine + "jNbre = " + str(jNbre)
Case "Effacer_Src"
   #If DebugBuild Then ' "Error ! Please contact the authors."
      If not(SymbText = WinMain.TabSymb(WinMain.SymbEff))
      Then MsgBox "Problème Tom" + EndOfLine + "LogSynchr
      o," + EndOfLine + "App.JournCol1(" + str(jNbre) + ") = "
      + str(App.JournCol1(jNbre)) + EndOfLine + "Symbol = "
      + SymbText
   #EndIf
   DrapErr = App.DeleteFileOrFolder(App.LogEltSrc(jNbre), App.
   LogDossTrash(jNbre))
Case "Effacer_Cib"
   #If DebugBuild Then ' "Error ! Please contact the authors."
      If not(SymbText = WinMain.TabSymb(WinMain.SymbEff))
      Then MsgBox "Problème Tom" + EndOfLine + "LogSynchr
      o," + EndOfLine + "App.JournCol1(" + str(jNbre) + ") = "
      + str(App.JournCol1(jNbre)) + EndOfLine + "Symbol = "
      + SymbText
   #EndIf
   DrapErr = App.DeleteFileOrFolder(App.LogEltCib(jNbre), App.
   LogDossTrash(jNbre))
Case "Rempl_Src->Cib" ' On remplace  EltSource  dans  EltCible
   #If DebugBuild Then ' "Error ! Please contact the authors."
      If not((SymbText = WinMain.TabSymb(WinMain.SymbRem
      pl)) and (App.JournCol6(jNbre) = WinMain.TabSymb(WinM
      ain.SymbFlecheD))) Then MsgBox "Problème Tom" + End
      OfLine + "LogSynchro," + EndOfLine + "App.JournCol1("
      + str(jNbre) + ") = " + str(App.JournCol1(jNbre)) + EndOf
      Line + "Symbol = " + SymbText + "  " + App.JournCol6(jN
      bre) + EndOfLine + " " + EndOfLine + "Mais si symbole "
      + WinMain.TabSymb(WinMain.SymbAncien) + " et Rempla
      cer plus récent alors c'est bon."
   #EndIf
   DrapErr = App.AScriptCopy(App.LogEltSrc(jNbre), App.LogElt
   Cib(jNbre).Parent, App.LogDossTrash(jNbre), True, App.LogUs
   eShell(jNbre), App.LogCdeCpShell(jNbre), App.LogCopyRBerr(j
   Nbre))
Case "Rempl_Cib->Src" ' On remplace  EltCible  dans  EltSource
   #If DebugBuild Then ' "Error ! Please contact the authors."
      If not((SymbText = WinMain.TabSymb(WinMain.SymbRem
      pl)) and (App.JournCol6(jNbre) = WinMain.TabSymb(WinM
```

```
      ain.SymbFlecheG))) Then MsgBox "Problème Tom" + End
      OfLine + "LogSynchro," + EndOfLine + "App.JournCol1("
      + str(jNbre) + ") = " + str(App.JournCol1(jNbre)) + EndOf
      Line + "Symbol = " + SymbText + "  " + App.JournCol6(jN
      bre) + EndOfLine + " " + EndOfLine + "Mais si symbole "
      + WinMain.TabSymb(WinMain.SymbAncien) + " et Rempla
      cer plus récent alors c'est bon."
    #EndIf
    DrapErr = App.AScriptCopy(App.LogEltCib(jNbre), App.LogElt
    Src(jNbre).Parent, App.LogDossTrash(jNbre), True, App.LogUs
    eShell(jNbre), App.LogCdeCpShell(jNbre), App.LogCopyRBerr(j
    Nbre))
Case "Copie_Src->Cib" ' On copie  EltSource  dans  EltCible
    #If DebugBuild Then ' "Error ! Please contact the authors."
      If not((SymbText = WinMain.TabSymb(WinMain.SymbCop
      y)) and (App.JournCol6(jNbre) = WinMain.TabSymb(WinM
      ain.SymbFlecheD))) Then MsgBox "Problème Tom" + End
      OfLine + "LogSynchro," + EndOfLine + "App.JournCol1("
      + str(jNbre) + ") = " + str(App.JournCol1(jNbre)) + EndOf
      Line + "Symbol = " + SymbText + "  " + App.JournCol6(jN
      bre)
    #EndIf
    DrapErr = App.AScriptCopy(App.LogEltSrc(jNbre), App.LogElt
    Cib(jNbre), App.LogDossTrash(jNbre), False, App.LogUseShell(
    jNbre), App.LogCdeCpShell(jNbre), App.LogCopyRBerr(jNbre))
Case "Copie_Cib->Src" ' On copie  EltCible  dans  EltSource
    #If DebugBuild Then ' "Error ! Please contact the authors."
      If not((SymbText = WinMain.TabSymb(WinMain.SymbCop
      y)) and (App.JournCol6(jNbre) = WinMain.TabSymb(WinM
      ain.SymbFlecheG))) Then MsgBox "Problème Tom" + End
      OfLine + "LogSynchro," + EndOfLine + "App.JournCol1("
      + str(jNbre) + ") = " + str(App.JournCol1(jNbre)) + EndOf
      Line + "Symbol = " + SymbText + "  " + App.JournCol6(jN
      bre)
    #EndIf
    DrapErr = App.AScriptCopy(App.LogEltCib(jNbre), App.LogElt
    Src(jNbre), App.LogDossTrash(jNbre), False, App.LogUseShell(
    jNbre), App.LogCdeCpShell(jNbre), App.LogCopyRBerr(jNbre))
Else
    DrapErr = True
    MsgBox "Error ! Please contact the authors." + EndOfLine + "L
    ogSynchro, Which case?" + EndOfLine + "App.JournCol1(" + st
    r(jNbre) + ") = " + str(App.JournCol1(jNbre))
End Select
If DrapErr Then
```

```
            SymbText = WinMain.TabSymb(WinMain.SymbOpErrDiv) + Sy
            mbText
            WinMain.NbErrRemplAnc = WinMain.NbErrRemplAnc + 1
        Else
            SymbText = WinMain.TabSymb(WinMain.SymbOpOui) + Symb
            Text
        End If

    Else ' On ne fait ce test que si opération non réalisée car sinon il peut
    y avoir problème de vérification avec élément supprimé (par exemple l
    e path d'un dossier supprimé n'a plus les : à la fin)
        SymbText = WinMain.TabSymb(WinMain.SymbOpNon) + WinMain.
        TabSymb(WinMain.SymbOpErrDiv) ' Ou SymbOpErrInc
        WinMain.NbErrRemplAnc = WinMain.NbErrRemplAnc + 1
        ' "Error ! Please contact the authors."
        ' MsgBox "Problème Tom" + EndOfLine + "LogSynchro, Ces valeur
        s doivent être égales!" + EndOfLine + "iLign = " + str(iLign) + "    j
        Nbre = " + str(jNbre) + EndOfLine + (App.JournCol5b(jNbre) + Ap
        p.JournCol5(jNbre)) + EndOfLine + (App.JournCol5b(jNbre) + ListB
        oxLog.Cell(iLign, 4)) + EndOfLine + EltSrcPath + EndOfLine + "an
        d" + EndOfLine + (App.JournCol7b(jNbre) + App.JournCol7(jNbre))
         + EndOfLine + (App.JournCol7b(jNbre) + ListBoxLog.Cell(jNbre, 6
        )) + EndOfLine + EltCibPath

    End If
Else
    SymbText = WinMain.TabSymb(WinMain.SymbOpNon) + WinMain.TabS
    ymb(WinMain.SymbOpErrDiv) ' Ou SymbOpErrInc
    WinMain.NbErrRemplAnc = WinMain.NbErrRemplAnc + 1
    ' "Error ! Please contact the authors."
    ' MsgBox "Problème Tom" + EndOfLine + "LogSynchro," + EndOfLine
    + "App.JournCol1(" + str(jNbre) + ") = " + str(App.JournCol1(jNbre)) +
    EndOfLine + "Symbol = " + SymbText + EndOfLine + "ListBoxLog.Cell(
    " + str(iLign) + ", 2) = " + ListBoxLog.Cell(iLign, 2)
End If

' Même chose que plus bas à Exception Err
NbCellsCheck = NbCellsCheck - 1
App.JournCol3(jNbre) = SymbText
ListBoxLog.Cell(iLign, 2) = SymbText
App.JournCol1(jNbre) = "NeRienFaire" ' Que erreur ou non, cette ligne ne p
ourra pas être à nouveau traitée
ListBoxLog.CellCheck(iLign, 0) = False ' Car reste à vrai même si CheckBox
supprimée
' ListBoxLog.CellCheck(iLign, 0).Enabled = False  Impossible
```

```
            ListBoxLog.CellType(iLign, 0) = ListBoxLog.TypeNormal ' 1 ' Si on écrit 0 c'
            est ColumnType soit 2
            ListBoxLog.Cell(iLign, 0) = CellSsCheckBox ' Je met un espace car c'est imp
            ossible de lire CellType  pour CompareRow
            ' ListBoxLog.Refresh ' Fait plus bas
            ' Self.UpdateNow ' J'ai fait Refresh ci-dessus

        End If ' Il y a une CheckBox
        ' EditFieldPath.Text = str(iLign) ' Pour test
        App.DoEvents ' Pour afficher l'effacement de la CheckBox ci-dessus, et laisse pl
        us de temps pour éventuel clic sur bouton Arrêter
        ' If iLign > 5 Then ListBoxLog.ScrollPosition = iLign - 5 ' Sinon ça ne plante pas
         mais bon. ScrollPosition à été mis à 0 au lancement du Thread
        ListBoxLog.ScrollPosition = Max(0, iLign - 5) ' Idem ci-dessus
        ' MsgBox "iLign = " + Str(iLign)
        ListBoxLog.Refresh ' Semble inutile mais je le fais quand même car dans Synchr
        o, quand j'écris danEditFieldPath je suis obligé de le Refresh
        ' Self.UpdateNow ' J'ai fait Refresh ci-dessus
        App.DoEvents ' Sinon la fenêtre n'est pas rafraichie et le scroll se fait mal, et l'é
        ventuel clic du bouton Arrêter ne se fait pas non plus
        ' ThreadLogSync.Sleep 1, False ' En fait c'est inutile, le refresh se fait

        If WinMain.ArretUrg Then Exit ' A pu être mis à  Vrai  par ailleurs
    Next iLign


Exception Err ' C'est sûrement des fichiers (dossiers parents) du journal qui ont été supp
rimés

    ' Inutile Drap_err = True '                On a dû ejecter l'un des disques pendant le
    process ou StackOverflowException ou ???
    ' "Error ! Please contact the authors."
    ' MsgBox "Problème Tom" + EndOfLine + "LogSynchro," + EndOfLine + "Exception E
    rr" + EndOfLine + "iLign = " + str(iLign) + "    jNbre = " + str(jNbre)
    SymbText = WinMain.TabSymb(WinMain.SymbOpNon) + WinMain.TabSymb(WinMain
    .SymbOpErrInc) ' Ou SymbOpErrDiv
    WinMain.NbErrRemplAnc = WinMain.NbErrRemplAnc + 1
    WinMain.ArretUrg = True ' On arrête tout

    If (iLign >= 0) and (iLign <= DerLign) and (jNbre >= 0) and (jNbre <= DerLign) The
    n ' Pour être sûr de ne pas faire une erreur dans la gestion d'erreur
        ' Même chose que plus haut
        NbCellsCheck = NbCellsCheck - 1
        App.JournCol3(jNbre) = SymbText
```

```
    ListBoxLog.Cell(iLign, 2) = SymbText
    App.JournCol1(jNbre) = "NeRienFaire" ' Que erreur ou non, cette ligne ne pourr
    a pas être à nouveau traitée
    ListBoxLog.CellCheck(iLign, 0) = False ' Car reste à vrai même si CheckBox sup
    primée
    ' ListBoxLog.CellCheck(iLign, 0).Enabled = False  Impossible
    ListBoxLog.CellType(iLign, 0) = ListBoxLog.TypeNormal ' 1 ' Si on écrit 0 c'est C
    olumnType soit 2
    ListBoxLog.Cell(iLign, 0) = CellSsCheckBox ' Je met un espace car c'est impossi
    ble de lire CellType  pour CompareRow
  End If

End Sub
```

WinLog.L_ThreadLogSync:

```
Private Sub L_ThreadLogSync()

  Dim TampNbErrRemplAnc as UInt16

  ' Avant je faisais ça dans le Thread, je préfère le faire avant
  If App.SyncEnCours Then
    MsgBox "Error ! Please contact the authors." + EndOfLine + "ThreadLogSync, Sy
    ncEnCours should be False."
    ' ThreadLogSync.Kill ' On arrête ici  Pas encore lancé

  ElseIf App.LogSEnCours Then
    MsgBox "Error ! Please contact the authors." + EndOfLine + "ThreadLogSync, Lo
    gSEnCours should be False."
    ' Me.Kill ' Je laisse, je ne fais rien

  Else
    App.LogSEnCours = True
    If WinMain.ArretUrg Then
      #If DebugBuild Then
        MsgBox "A vérifier Tom." + EndOfLine + "ThreadLogSync, ArretUrg dev
        rait être False mais peut être que Arrêt d'urgence fait juste à la fin de l
        a synchro ?"
      #EndIf
      WinMain.ArretUrg = False
    End If
    DrapMenu = False
    WinMain.PushButtSync.Enabled = False ' AVANT  EnableMenuBoutons  car test d
    ans  CaptSyncSimu
```

```
App.EnableMenuBoutons(True)
PushButtLogSync.Caption = Btn_Arreter
EditFieldPath.Text = "" ' EditFieldPath_Note
ListBoxLog.ScrollPosition = 0

ProgWheelLog.Visible = True
' ### Début ancien Thread ThreadTrait.Run ' S'il tournait déjà on aura un bug
' MsgBox "Début LogSync" ' MsgBox plante dans Thread
If TimerCellCheck.Mode = Timer.ModeSingle Then Pause(1, True) ' Me.Sleep 2, F
alse ' Des fois que, afin que le Timer se fasse et que le compte de  NbCellsChec
k  soit juste
TampNbErrRemplAnc = WinMain.NbErrRemplAnc

LogSynchro ' La synchro est faîte dedans

If NbCellsCheck < 0 Then MsgBox "Error ! Please contact the authors." + EndOf
Line + "ThreadLogSync, shouldn't be negative!" + EndOfLine + "NbCellsCheck
= " + str(NbCellsCheck)

EditFieldPath.Text = EditFieldPath_Note

If WinMain.ArretUrg Then
   If App.Sfx Then
      Coup_de_freins.Play
      While Coup_de_freins.IsPlaying
      Wend
   End If
   ' ArretUrg = False ' Fait plus bas des fois qu'on réappuie encore sur Esc le t
   emps de finir ici
   ' Non car je n'écris pas 'Non terminé'  App.JourDcoulPos(0) = 2 ' pb rouge ,
    y était peut-être déjà  Et il faudrait rafraichir avec  EcrEdFieldLog
Else
   If App.Sfx Then
      Bop.Play
      While Bop.IsPlaying
      Wend
   End If
   ' Non car je n'écris pas 'Non terminé'  App.JourDcoulPos(0) = 1 ' ok vert , y
   était peut-être déjà  Et il faudrait rafraichir avec  EcrEdFieldLog
End If
If WinMain.NbErrRemplAnc > TampNbErrRemplAnc Then ' Ne peut pas être infé
rieur mais égal si pas d'erreur en plus
   If TampNbErrRemplAnc = 0 Then ' Sinon c'était déjà en rouge et il y avait T
   xt_Attention
```

```
        App.JourDcoulPos(2) = 2 ' pb rouge
        WinMain.JournalDd = (WinMain.JournalDd + Txt_Attention + " : ")
        App.JourDcoulPos(4) = Len(WinMain.JournalDd) - App.JourDcoulPos(3)
      End If
      EcrEdFieldLog(WinMain.JournalDd, Format(WinMain.NbErrRemplAnc, F_MBil
      lionAvEsp), WinMain.JournalDf)
      Pause(10, False) ' Me.Sleep 200, False ' Note Pause(60) = 1 s = 1000 ms
      If App.Sfx Then
        RadioBeep_t.Play ' Buzzer.Play
        While RadioBeep_t.IsPlaying ' While Buzzer.IsPlaying
        Wend
      End If
    Else
      Pause(10, False) ' Me.Sleep 100, False ' Inutile mais c'est pour faire comme
      dans ThreadSync (où c'est inutile aussi ;-) )
    End If

    Pause(10, False) ' Me.Sleep 200, False ' Je préfére afin d'être sûr que tout soit fi
    ni dans le Finder avant de relancer une Synchro car parfois
    '  j'ai des Nil Objection quand je reclique  Synchroniser  aussi sec que synchro p
    récédente terminée
    ' J'aurais pu mettre un délai plus long pour lancer le Timer, mais je recopie le Ti
    merFinThread d'un programme à l'autre alors je garde .Period = 1

    ' MsgBox "Synchro terminée, on réactive menus boutons" ' MsgBox plante dans
    Thread
    TimerFinLogSync.Mode = Timer.ModeSingle
    ' ### Fin ancien Thread

  End If

End Sub

NbCellsCheck As Int32


Private TampCellCheck As Boolean


TampClose As Boolean


Private TampRow As Int32
```

WinLog Note: CancelClose

CancelClose

NON car on ne peut pas fermer cette fenêtre (Journal) tant que la synchro est en cours p
uisque DrapMenu à False

Return (not DrapMenu) ' Si on retourne Vrai la fenêtre ne se fermera pas
' Mettre la même chose que dans WinMain.CancelClose
' Si on quitte par le Dock on annulerait la fermeture de WinMain mais pas celle-ci qui se
fermerait
' donc on perdrait les coordonnées de la fanêtre

WinLog Note: DoubleClick

DoubleClick

A mettre dans CellClick

LBoxRow = row
LBoxCol = column

A mettre dans Event DoubleClick, mais ça reste à voir, notamment la gestion du symbol
e SymbAncien

Dim jNbre as Int32 ' Pas UInt16 si Ubound -1 car vide

If not(App.SyncEnCours or App.LogSEnCours) Then ' Car si oui alors EditFieldPath.Text ≠
 Me.Cell(row, column) de toute façon ListBox disable quand SyncEnCours
' Si App.LogSEnCours on fait quand même mais ListBox disable donc on ne peut pas (
mais sinon je ferai pour compter NbCellsCheck)
EditFieldPath.Text = str(LBoxRow) + ", " + str(LBoxCol)

If LBoxCol = 4 Then ' Colonne des flèches sens copie
jNbre = Val(ListBoxLog.Cell(LBoxRow, 1)) - 1 ' Indice de la ligne
Select Case App.JournCol1(jNbre)
Case "Rempl_Src->Cib" ' On remplace EltSource dans EltCible
App.JournCol1(jNbre) = "Rempl_Cib->Src"

```
App.JournCol6(jNbre) = WinMain.SymbFlecheG
Me.Cell(LBoxRow, LBoxCol) = WinMain.SymbFlecheG
' Me.Refresh

Case "Rempl_Cib->Src" ' On ou remplace  EltCible  dans  EltSource
App.JournCol1(jNbre) = "Rempl_Src->Cib"
App.JournCol6(jNbre) = WinMain.SymbFlecheD
Me.Cell(LBoxRow, LBoxCol) = WinMain.SymbFlecheD
' Me.Refresh

' Case "Copie_Src->Cib" ' On copie (ou remplace)  EltSource  dans  EltCible
'   PAS copie car il n'y pas de fichier à remplacer donc pas de sens inverse
' Case "Copie_Cib->Src" ' On copie (ou remplace)  EltCible  dans  EltSource

Else
Beep

' VOIR pour symbole Orange plus récent à inverser

End Select
' Self.UpdateNow

End If

End If
```

WinLog Control PushButtFermer:

```
Sub Action()

  Self.Close

End Sub
```

WinLog Control PushButtEnregF:

```
Sub Action()

  Dim FichierLog as FolderItem
  Dim iLign, jCol, DerLign, DerCol as Int32 ' Pas UInt16 si Ubound -1 car vide
```

```
Dim TampLog, TampText as String
Dim Stream as TextOutputStream

FichierLog = MyDialogSave(App.FichLog, Txt_SelectDossier, "text/plain", "Log Sync.t
xt") ' Voir FileTypes !!!

If not(FichierLog = Nil) Then ' A  Nil  si cliqué  Annuler
  ' MsgBox "f_Elt : " + f_Elt.AbsPath_f + EndOfLine + "Nom : " + f_Elt.Name
  App.FichLog = FichierLog
  DerLign = ListBoxLog.ListCount - 1
  DerCol = ListBoxLog.ColumnCount - 1

  TampLog = WinMain.JournalD_HT + EndOfLine
  ' TampLog = TampLog + EndOfLine ' Saut de ligne, mais maintenant j'écris no
  m et version prog
  TampLog = TampLog + CarTab + CarTab+ App.NomProg + " " + App.ShortVer
  sion + EndOfLine ' App.VersionProg
  TampLog = TampLog + EditFieldLog.Text + EndOfLine
  TampLog = TampLog + EndOfLine ' Saut de ligne

  #If TargetWin32 Then
    If FichierLog.Exists Then
      If not FichierLog.Visible Then FichierLog.Visible = True ' Car ça plante s
      ous Windows si on recrée (et écrit) un fichier invisible
    End If
  #EndIf

  For iLign = 0 to DerLign ' On sauvegarde dans l'ordre de tri actuel de la liste
    ' ListBoxLog.Cell(iLign, 0) ' On ignore la colonne des CheckBox
    TampText = ListBoxLog.Cell(iLign, 1)
    For jCol = 2 to DerCol
      TampText = TampText + CarTab + ListBoxLog.Cell(iLign, jCol)
    Next jCol
    TampLog = TampLog + TampText + EndOfLine
  Next iLign

  Stream = TextOutputStream.Create(FichierLog) ' Ou  FichPartie
  Stream.Write ConvertEncoding(TampLog, App.EncodFich(PopupEncodSrc.ListInd
  ex)) ' App.DefAppEncod
  Stream.Close

  ' #If TargetMacOS Then ' Car sinon  ttxt  est mis comme créateur
  ' FichierLog.MacCreator = "ttxt" ' ATTENTION  Mettre la même chose dans  App
   Créateur
```

```
    ' #Else
    ' #If DebugBuild Then
    ' MsgBox "Problème Tom, tu devrais être en TargetMacOS !"
    ' #EndIf
    ' #EndIf

  End If


Exception Err

  BeepNbT(3)
  MsgBox "Error ! Please contact the authors." + EndOfLine + "Save Log, problem whil
  e writing file!"

End Sub
```

WinLog Control ListBoxLog:

```
Function CompareRows(row1 as Integer, row2 as Integer, column as Integer, ByRef result
as Integer) As Boolean

  Dim MaCell1, MaCell2 as String

  If App.SyncEnCours or App.LogSEnCours Then ' On est en train de synchroniser, mai
  s de toute façon la ListBox est disable quand synchro en cours
    result = 0 ' On ne trie pas, les 2 lignes sont considérées égales

  Else
    If column = 0 Then
      If Me.Cell(row1, 0) = CellSsCheckBox Then ' Me.CellType(row1, 0) = Me.Typ
      eCheckbox Then ' soit 2
        MaCell1 = "" ' C
      Else
        If Me.CellCheck(row1, 0) Then
          MaCell1 = "B"
        Else
          MaCell1 = "A"
        End If
      End If
      ' Je fais selon la logique mais ce serait plus pratique dans l'ordre inverse (c
      ase cochée plus petite donc en haut si ordre croissant)
      If Me.Cell(row2, 0) = CellSsCheckBox Then ' Me.CellType(row2, 0) = Me.Typ
      eCheckbox Then ' soit 2
```

```
        MaCell2 = "" ' C
      Else
        If Me.CellCheck(row2, 0) Then
          MaCell2 = "B"
        Else
          MaCell2 = "A"
        End If
      End If
    Else
      MaCell1 = Me.Cell(row1, column)
      MaCell2 = Me.Cell(row2, column)
    End If
    ' If row1 = 1 Then MsgBox str(Me.CellType(row1, column)) + " : '" + Me.Cell(ro
    w1, column) + "' : " + MaCell1 + EndOfLine + str(Me.CellType(row2, column)) +
     " : '" + Me.Cell(row2, column) + "' : " + MaCell2
    If column = 1 Then
      #If DebugBuild Then
        If (Val(MaCell1) < 1) or (Val(MaCell2) < 1) or (Val(MaCell1) = Val(MaCell
        2)) Then MsgBox "Ca merde Tom, Evenmt colonne 1 doit être un nomb
        re > 0 et on doit avoir" + EndOfLine + MaCell1 + " ≠ " + MaCell2
      #EndIf
      If Val(MaCell1) > Val(MaCell2) Then ' Si on comparait String on aurait 3 > 2
      1
        result = 1
      Else ' Note si c'est 2 lignes d'événements alors ce test jamais = car incréme
      nt à chaque événement
        result = -1
      End If
    Else ' On trie une autre colonne que 1  Pas de problème si on trie par rapport à
    colonne 0 CheckBox
      If MaCell1 > MaCell2 Then ' On compare String
        result = 1
      Else
        result = -1
      End If
    End If


  End If

  Return True ' Return False si on utilisait le trie par défaut de RealBasic

End Function
```

```
Sub Open()

   ' Aligner à gauche par défaut
   Me.ColumnAlignment(1) = 3 ' Droite
   Me.ColumnAlignment(5) = 2 ' Centre

   ' Me.ColumnType(1) = ListBox.TypeDefault ' Same as its column type, comme ici on
    est dans colonne type
   Me.ColumnType(0) = ListBox.TypeCheckbox
   Me.ColumnType(1) = ListBox.TypeNormal ' Indice n° de ligne
   Me.ColumnType(2) = ListBox.TypeNormal ' Symbole de l'opération effectuée ou à ef
   fectuer
   Me.ColumnType(3) = ListBox.TypeNormal ' Extension
   Me.ColumnType(4) = ListBox.TypeNormal ' ListBox.TypeEditable ' Path du Source
   Me.ColumnType(5) = ListBox.TypeNormal ' Symbole du sens de la copie
   Me.ColumnType(6) = ListBox.TypeNormal ' ListBox.TypeEditable ' Path du Cible

End Sub

Sub MouseMove(X As Integer, Y As Integer)

   Dim row, column, jNbre as Int32 ' MaListRow, MaListeColumn

   If not(App.SyncEnCours or App.LogSEnCours) Then ' Car EditFieldPath affiche le path
    du dossier en cours de synchro
      '       pas  or App.LogSEnCours  car là il n'y a pas de problème avec EditFieldPat
      h
      row = Me.RowFromXY(X, Y) ' MaListRow
      column = Me.ColumnFromXY(X, Y) ' MaListeColumn
      ' MsgBox "mouve (" + str(row) + ", " + str(column) + ")"
      If column = 3 Then
         EditFieldPath.Text = Me.Cell(row, column) ' MaListRow, MaListeColumn

      ElseIf column = 4 Then ' Même test que dans CellClick
         jNbre = Val(ListBoxLog.Cell(row, 1)) - 1 ' Indice de la ligne
         EditFieldPath.Text = App.JournCol5b(jNbre) + Me.Cell(row, column) ' = Ap
         p.JournCol5(jNbre)  MaListRow, MaListeColumn
         ' Note :  Me.Cell(row, column)  peut être une cellule vide (si on supprime u
         n fichier d'un dossier l'autre colonne est vide)
         ' AffMasqHelpTag(Me.Name, Me.Cell(row, column))
         ' J'affiche dans un EditField car c'était long d'attendre l'HelpTag de la liste s
         ans bouger la souris

      ElseIf column = 6 Then ' Même test que dans CellClick
```

```
      jNbre = Val(ListBoxLog.Cell(row, 1)) - 1 ' Indice de la ligne
      EditFieldPath.Text = App.JournCol7b(jNbre) + Me.Cell(row, column) ' = Ap
      p.JournCol7(jNbre) MaListRow, MaListeColumn
      ' Note :  Me.Cell(row, column)  peut être une cellule vide (si on supprime u
      n fichier d'un dossier l'autre colonne est vide)
      ' AffMasqHelpTag(Me.Name, Me.Cell(row, column))
      ' J'affiche dans un EditField car c'était long d'attendre l'HelpTag de la liste s
      ans bouger la souris

    Else ' Autre colonne
      EditFieldPath.Text = EditFieldPath_Note

    End If

    ' Else
    ' Si SyncEnCours on le laisse à ""
  End If

End Sub

Function CellClick(row as Integer, column as Integer, x as Integer, y as Integer) As Boolea
n

  Dim f_elt as FolderItem
  Dim jNbre as Int32

  ' MsgBox "clic (" + str(row) + ", " + str(column) + ")" + EndOfLine + "Clic Droit : " +
  Cstr(IsContextualClick) + EndOfLine + "Ctrl : " + Cstr(Keyboard.AsyncControlKey) +
   EndOfLine + "Alt-Option : " + Cstr(Keyboard.AsyncAltKey) + EndOfLine + "Cde : "
  + Cstr(Keyboard.AsyncOSKey) + EndOfLine + "Shift : " + Cstr(Keyboard.AsyncShiftK
  ey) ' Keyboard.AsyncOSKey = Keyboard.AsyncCommandKey
  If not(App.SyncEnCours or App.LogSEnCours) Then ' Car si oui alors EditFieldPath.Te
  xt ≠ Me.Cell(row, column)  de toute façon ListBox disable quand SyncEnCours
    ' Si  App.LogSEnCours  on fait quand même mais ListBox disable donc on ne pe
    ut pas (mais sinon je ferai pour compter NbCellsCheck)
    If column = 0 Then
      If TimerCellCheck.Mode = Timer.ModeSingle Then
        MsgBox "TimerCellCheck is running, you click to fast !"
        Do Until (TimerCellCheck.Mode = Timer.ModeOff)
          App.DoEvents ' Que le Timer se fasse
        Loop
      End If
      #If DebugBuild Then
        jNbre = Val(Me.Cell(row, 1)) - 1 ' Indice de la ligne
```

```
      If (Me.Cell(row, 0) = CellSsCheckBox) Xor (App.JournCol1(jNbre) = "NeR
      ienFaire") Then
        MsgBox "row = " + str(row) + EndOfLine + "jNbre = " + str(jNbre)
        + EndOfLine + """ + Me.Cell(row, 0) + """ + App.JournCol1(jNbre)
      End If
    #EndIf
    TampRow = row
    ' TampColumn = column ' 0
    If Me.Cell(row, 0) = CellSsCheckBox Then ' App.JournCol1(jNbre) = "NeRien
    Faire"
      TampCellCheck = False
    Else
      TampCellCheck = Me.CellCheck(row, 0)
    End If
    ' MsgBox "CellCheck(" + str(row) + ", 0) = " + Cstr(TampCellCheck)
    TimerCellCheck.Mode = Timer.ModeSingle ' Car la checkbox de la cellule n
    'a pas encore changé d'état, il changera le temps que la tempo se lance

ElseIf column = 4 Then ' Même test que dans MouseMove mais le test plus bas
suffit puisqu' EditFieldPath  doit être à "" si sur autre colonne
    jNbre = Val(ListBoxLog.Cell(row, 1)) - 1 ' Indice de la ligne
    #If DebugBuild Then
      If not(EditFieldPath.Text = App.JournCol5b(jNbre) + Me.Cell(row, colu
      mn)) Then MsgBox "Ca déconne Tom, EditFieldPath pas sur bonne case
       !"
    #EndIf
    If not((EditFieldPath.Text = "") or (EditFieldPath.Text = EditFieldPath_Note))
    Then ' Car il se peut que colonne 4 ou 6 vide quand on efface élément de l'
    autre colonne
      ' MsgBox "row = " + str(row) + EndOfLine + "jNbre = " + str(jNbre)
      If ((App.JournCol5b(jNbre) + App.JournCol5(jNbre)) = EditFieldPath.Te
      xt) and (EditFieldPath.Text = AbsPathSiExist(App.LogEltSrc(jNbre))) The
      n
        f_elt = App.LogEltSrc(jNbre)
      Else
        f_elt = Nil
        #If DebugBuild Then
          MsgBox "Problème Tom" + EndOfLine + "Ces valeurs doivent
          être égales!" + EndOfLine + "row = " + str(row) + "    jNbre =
          " + str(jNbre) + EndOfLine + EditFieldPath.Text + EndOfLine
          + (App.JournCol5b(jNbre) + App.JournCol5(jNbre)) + EndOfLi
          ne + AbsPathSiExist(App.LogEltSrc(jNbre))
        #EndIf
      End If
```

```
  End If

ElseIf column = 6 Then ' Même test que dans MouseMove mais le test plus bas
suffit puisqu' EditFieldPath  doit être à "" si sur autre colonne
  jNbre = Val(ListBoxLog.Cell(row, 1)) - 1 ' Indice de la ligne
  #If DebugBuild Then
    If not(EditFieldPath.Text = (App.JournCol7b(jNbre) + Me.Cell(row, colu
    mn))) Then MsgBox "Ca déconne Tom, EditFieldPath pas sur bonne cas
    e !"
  #EndIf
  If not((EditFieldPath.Text = "") or (EditFieldPath.Text = EditFieldPath_Note))
  Then ' Car il se peut que colonne 4 ou 6 vide quand on efface élément de l'
  autre colonne
    ' MsgBox "row = " + str(row) + EndOfLine + "jNbre = " + str(jNbre)
    If ((App.JournCol7b(jNbre) + App.JournCol7(jNbre)) = EditFieldPath.Te
    xt) and (EditFieldPath.Text = AbsPathSiExist(App.LogEltCib(jNbre))) Th
    en
      f_elt = App.LogEltCib(jNbre)
    Else
      f_elt = Nil
      #If DebugBuild Then
        MsgBox "Problème Tom" + EndOfLine + "Ces valeurs doivent
        être égales!" + EndOfLine + "row = " + str(row) + "    jNbre =
        " + str(jNbre) + EndOfLine + EditFieldPath.Text + EndOfLine
        + (App.JournCol7b(jNbre) + App.JournCol7(jNbre)) + EndOfLi
        ne + AbsPathSiExist(App.LogEltCib(jNbre))
      #EndIf
    End If
  End If

  ' Else ' Autre colonne
  '  Beep

End If

If (column = 4) or (column = 6) Then ' Même test que dans MouseMove mais le
test plus bas suffit puisqu' EditFieldPath  doit être à "" si sur autre colonne
  ' AVANT : f_elt = GetFitemAbsPath(EditFieldPath.Text, True) ' Note :  Me.Ce
  ll(row, column)  peut être une cellule vide (si on supprime
  '  un fichier d'un dossier l'autre colonne est vide) et GetFolderItem("") affich
  e le dossier de l'application
  If Keyboard.AsyncControlKey or IsContextualClick Then
    If PasNil_Existe_Alias(f_elt, False) Then ' On s'en fou si Alias que cible e
    xiste ou non
```

```
            ' #If DebugBuild Then
            ' If not(f_elt.AbsPath_f = EditFieldPath.Text = Me.Cell(row, column
            )) Then MsgBox "Ca déconne Tom, f_elt pas sur bonne case !" ' Pa
            s forcément pareil si  RelPathLog
            ' #EndIf
            If Keyboard.AsyncAltKey Then ' Pas Shift ni Command car servent
            aussi à sélectionner plusieurs lignes : Keyboard.AsyncShiftKey Key
            board.AsyncCommandKey
              PressPapTxt(f_elt.ShellPath) ' PAS le  _fAS  qui est celui pour A
              ppleScript, mais le VRAI pour le Terminal
            Else
              PressPapTxt(f_elt.AbsPath_f) ' Copie le path dans le presse-pa
              piers
            End If
            ' Else
            '  Beep
            '  PressPapTxt("")
          End If
        ElseIf Keyboard.AsyncAltKey Then
          If PasNil_Existe_Alias(f_elt, False) Then ' On s'en fou si Alias que cible e
          xiste ou non
            ' MsgBox """ + EditFieldPath.Text + """ + EndOfLine + """ + f_AbsPa
            th + """
            f_elt.Parent.Launch ' Launch le dossier parent donc l'ouvre dans le
             Finder, et ne place que cette fenêtre au 1er plan, pas toutes celle
            s du Finder comme
            '  Activate d'AppleScript. Mais le Reveal d'AppleScript ci-dessous n
            e fait qu'ouvrir la fenêtre du dossier, ça ne la place pas au 1er pla
            n
            RevealThisItem(f_elt.ShellPath_fAS) ' Mieux que ci-dessus, en plus
             d'ouvrir la fenêtre du dossier, ça sélectionne l'élément dans la fe
            nêtre, n'existe pas dans RealBasic
            ' Else
            '  Beep
          End If
        End If
      End If

    End If
End Function

Function CellTextPaint(g As Graphics, row As Integer, column As Integer, x as Integer, y a
s Integer) As Boolean
```

```
Dim TampText, CeCar1, CeCar2 as String

If App.CoulDsListLog Then
  ' Beep
  If column = 2 Then
    TampText = Me.Cell(row, 2)
    CeCar1 = Left(TampText, 1) ' = Mid(TampText, 1, 1)
    CeCar2 = Mid(TampText, 2, 1)
    ' MsgBox "Il ne peut y avoir qu'un de ces 3 symboles en 1er :" + EndOfLine
    + WinMain.SymbOpErrDiv + WinMain.SymbOpNon + WinMain.SymbOpOui
    ' MsgBox "et un ces 6 symboles en 2ième :" + EndOfLine + WinMain.Symb
    OpErrInc + WinMain.SymbOpErrOvF + WinMain.SymbRempl + WinMain.Sy
    mbEff + WinMain.SymbCopy + WinMain.SymbAncien
    If (CeCar1 = WinMain.TabSymb(WinMain.SymbOpErrDiv)) or (CeCar2 = Win
    Main.TabSymb(WinMain.SymbOpErrDiv)) or (CeCar2 = WinMain.TabSymb(W
    inMain.SymbOpErrInc)) or (CeCar2 = WinMain.TabSymb(WinMain.SymbOpEr
    rOvF)) Then
      g.ForeColor = CoulRouge
    ElseIf CeCar2 = WinMain.TabSymb(WinMain.SymbAncien) Then
      g.ForeColor = CoulOrang
    Else
      g.ForeColor = CoulVert
      #If DebugBuild Then
        If not((CeCar1 = WinMain.TabSymb(WinMain.SymbOpNon)) or (CeC
        ar1 = WinMain.TabSymb(WinMain.SymbOpOui))) Then MsgBox "Lis
        tBox.CellTextPaint, the characters are " + TampText + " : CeCar1
        = " + CeCar1 + " !"
        If not((CeCar2 = WinMain.TabSymb(WinMain.SymbRempl)) or (CeC
        ar2 = WinMain.TabSymb(WinMain.SymbEff)) or (CeCar2 = WinMain
        .TabSymb(WinMain.SymbCopy))) Then MsgBox "ListBox.CellTextPai
        nt, the characters are " + TampText + " : CeCar2 = " + CeCar2 + "
         !"
        ' App.CoulDsListLog = False ' Si je ne veux pas avoir 50 alertes
      #EndIf
    End If

  ElseIf column = 5 Then
    g.ForeColor = CoulViolet

  End If
End If

Return False ' PAS True ou ça n'écrit rien
```

```
  'ElseIf CeCar1 = WinMain.SymbOpNon Then
  'If (CeCar2 = WinMain.SymbRempl) or (CeCar2 = WinMain.SymbEff) or (CeCar2 = Wi
  nMain.SymbCopy ) Then
  'g.ForeColor = CoulVert_N
  'ElseIf CeCar2 = WinMain.SymbAncien Then
  'g.ForeColor = CoulOrang_N
  'Else
  'MsgBox "Error ! Please contact the authors." + EndOfLine + "ListBox.CellTextPaint,
  the characters are " + TampText + " : " + CeCar1 + CeCar2 + " !"
  'End If
  'ElseIf CeCar1 = WinMain.SymbOpOui Then
  'If (CeCar2 = WinMain.SymbRempl) or (CeCar2 = WinMain.SymbEff) or (CeCar2 = Wi
  nMain.SymbCopy ) Then
  'g.ForeColor = CoulVert_O
  'ElseIf CeCar2 = WinMain.SymbAncien Then
  'g.ForeColor = CoulOrang_O
  'Else
  'MsgBox "Error ! Please contact the authors." + EndOfLine + "ListBox.CellTextPaint,
  the characters are " + TampText + " : " + CeCar1 + CeCar2 + " !"
  'End If
  'Else
  'MsgBox "Error ! Please contact the authors." + EndOfLine + "ListBox.CellTextPaint,
  the characters are " + TampText + " : " + CeCar1 + CeCar2 + " !"
  'End If

End Function

Function CellBackgroundPaint(g As Graphics, row As Integer, column As Integer) As Bool
ean

  If (row mod 2) = 0 Then
    g.ForeColor = App.ColBackList ' J'ai mis comme iTunes, avant c'était RGB(232,2
    35,255)
    g.FillRect 0, 0, g.Width, g.Height
    ' Else
    ' Reste à sa valeur (blanc)
  End If

End Function
```

WinLog Control PopupTailleTextListe:

```
Sub Change()
```

```
  Dim TampNbre, TtSize as Int16 ' ATTENTION, les tailles doivent être à la bonne vale
  ur au lancement du programme, Voir PrefsCharger

  TampNbre = ListBoxLog.TextSize ' = EditFieldPath.TextSize
  TtSize = Val(Me.Text)

  ListBoxLog.TextSize = TtSize
  EditFieldPath.TextSize = TtSize

  ' L'instruction ci-dessous sert non seulement à garder l'ascenseur à peu près au mê
  me niveau par rapport au texte,
  '  mais surtout à le rafraîchir car les flèches ne disparaissent / n'apparaissent pas
  ListBoxLog.ScrollPosition = Round(ListBoxLog.ScrollPosition * (TtSize / TampNbre))

End Sub
```

WinLog Control EditFieldLog:

```
Sub MouseUp(X As Integer, Y As Integer)

  If Keyboard.AsyncAltKey and Keyboard.AsyncOSKey Then ' Keyboard.AsyncOSKey =
  Keyboard.AsyncCommandKey
    ' Beep
    ListBoxLog.ColumnWidths = App.ListBoxLogColWidths
  End If

End Sub

Function MouseDown(X As Integer, Y As Integer) As Boolean

  Return True ' Pour pouvoir faire  MouseUp

End Function
```

WinLog Control PushButtLogSync:

```
Sub Action()

  If App.LogSEnCours Then ' Même chose que dans event KeyDown
    If not WinMain.ArretUrg Then ' On n'a pas encore appuyé, si pas de test je ferai
    autant de fois ce qui suit que de
```

```
      WinMain.ArretUrg = True '  clics sur  Arrêter  (ou menu  Arrêter ou appuie s
      ur Esc)
      ' Quoique là je ne fais rien d'autres que Zoum mais si un jour …
      If App.Sfx Then
        Zoum.Play ' Pour signifier prise en compte  ArretUrg     On ne l'entend
        s pas si coup de frein juste après, mais des fois
        While Zoum.IsPlaying '                                    que ce soit plu
        s long pour stopper tout
        Wend
      End If
      ' NON, NE PAs faire boucle ci-dessous, sinon cet event (clic sur Bouton Syn
      chro/Arrêter) ne se fait QU'à la fin de la sychro, donc redémarre et arrête a
      ussi sec
      ' While App.LogSEnCours ' Si ça marche !  (NE PAS faire ça car garde la mai
      n et le Thread ne se fini pas)
      ' Wend '                                                Même sans DoEvents
      Pause(10, False) ' Pour éviter 2 appuie trop vif
      ' If App.Sfx Then Coup_de_freins.Play ' A été fait dans le Thread de Synchro
    End If

  Else
    L_ThreadLogSync ' Launch ThreadLogSync (.Run) ' On lance la Synchro dans un
    Thread

  End If

End Sub
```

WinLog Control BevButtCheck:

```
Sub Action()

  Dim iLign, DerLign, NbCellsSel as Int32 ' Pas UInt16 si Ubound -1 car vide
  Dim CochCell, TtesLesLignes as Boolean ' A False si appuie sur Alt pour tout décoch
  er au lieu de tout cocher
  #If DebugBuild Then
    Dim jNbre as Int32
  #EndIf

  If TimerCellCheck.Mode = Timer.ModeSingle Then TimerCellCheck.Mode = Timer.M
  odeOff ' Puisqu'on va tout checker ou déchecker
  CochCell = not Keyboard.AsyncAltKey ' and  ' Keyboard.AsyncOSKey = Keyboard.As
  yncCommandKey
```

```
' Me.Value = not Me.Value ' Je ne fais pas un bouton à 2 état, juste un PushButton.
Voir note plus bas
NbCellsCheck = 0
DerLign = UBound(App.JournCol1) ' Ou n'importe quel autre

NbCellsSel = 0 ' L'est déjà mais comme d'hab, j'initialise
For iLign = 0 to DerLign ' Si Ubound = -1 (pas de journal), on n'entre pas dans la bo
ucle
   If ListBoxLog.Selected(iLign) Then NbCellsSel = NbCellsSel + 1
Next iLign
TtesLesLignes = (NbCellsSel = 0) ' Si aucune ligne n'était sélectionné, on applique à
 toute la liste

For iLign = 0 to DerLign ' Si Ubound = -1 (pas de journal), on n'entre pas dans la bo
ucle
   #If DebugBuild Then
      jNbre = Val(ListBoxLog.Cell(iLign, 1)) - 1 ' Indice de la ligne
      If (ListBoxLog.Cell(iLign, 0) = CellSsCheckBox) Xor (App.JournCol1(jNbre) =
       "NeRienFaire") Then
         MsgBox "iLign = " + str(iLign) + EndOfLine + "jNbre = " + str(jNbre) +
         EndOfLine + """ + ListBoxLog.Cell(iLign, 0) + """ + App.JournCol1(jNbre
         )
      End If
   #EndIf
   If not(ListBoxLog.Cell(iLign, 0) = CellSsCheckBox) Then ' not(App.JournCol1(jNb
   re) = "NeRienFaire")
      If TtesLesLignes or ListBoxLog.Selected(iLign) Then ListBoxLog.CellCheck(iL
      ign, 0) = CochCell ' Me.Value
      If ListBoxLog.CellCheck(iLign, 0) Then NbCellsCheck = NbCellsCheck + 1
   End If
Next iLign

PushButtLogSync.Enabled = not(NbCellsCheck = 0) ' and (UBound(App.JournCol1) >
-1) ' Forcément si NbCellsCheck > 0

' Voir pour retrier colonne mais si on coche tout il ne devrait pas y avoir de change
ment

' Mettre ce HelpTag si en mode 2 états
' Cocher / décocher tous
' Check / uncheck all
' Marcar / Desmarcar todos
' Aktivieren bzw. deaktivieren Sie alle
```

```
End Sub
```

WinLog Control TimerFinLogSync:

```
Sub Action()

  ' Je fini mes Threads par un timer car si j'Enable et je remet tout (menus, boutons, e
  tc.) à leur valeur dans le Thread ça peut changer en même temps dans un event
  ' En lançant un Timer en fin de Thread, je bloque les events

  ' Fin du Thread : ThreadLogSync

  ProgWheelLog.Visible = False
  If WinMain.ArretUrg and (not(Me.Period = 1000)) Then
    Me.Period = 1000 ' Je ne le fais pas à la fin de L_ThreadTrait car parfois on re-c
    liquait le bouton pendant l'attente Timer
    Me.Mode = Timer.ModeSingle ' On relance le Timer avec une pause plus longue
     comme ça les events (pleins de clics sur bouton Stop) ont le temps de se faire
  Else
    DrapMenu = True
    App.LogSEnCours = False
    WinMain.PushButtSync.Enabled = True ' AVANT  EnableMenuBoutons  car test d
    ans  CaptSyncSimu
    App.EnableMenuBoutons(True)
    PushButtLogSync.Caption = Btn_LogSync
    PushButtLogSync.Enabled = not(NbCellsCheck = 0) ' and (UBound(App.JournCol
    1) > -1) ' Forcément si NbCellsCheck > 0
    If WinMain.ArretUrg Then
      WinMain.ArretUrg = False
      If not(Me.Period = 1000) Then Me.Period = 10 ' Val par défaut, à faire tout
      à la fin car Me.Mode est encore à Timer.ModeSingle (1), et si le code restan
      t à exécuter est plus long que 10 ms ça relance le Timer
    End If
  End If

End Sub
```

WinLog Control TimerCellCheck:

```
Sub Action()

  If TampCellCheck Then ' La cellule était cochée
```

```
    If not ListBoxLog.CellCheck(TampRow, 0) Then NbCellsCheck = NbCellsCheck -
     1 ' On l'a décochée
Else ' La cellule était décochée
    If ListBoxLog.CellCheck(TampRow, 0) Then NbCellsCheck = NbCellsCheck + 1 '
    On l'a cochée
End If

If ListBoxLog.SortedColumn = 0 Then ListBoxLog.Sort ' On retrie la liste car si triée
selon checkbox ça a changé, pas sinon car c'est chiant

' MsgBox "NbCellsCheck = " + str(NbCellsCheck)
If NbCellsCheck < 0 Then MsgBox "Error ! Please contact the authors." + EndOfLine
+ "TimerCellCheck, shouldn't be negative!" + EndOfLine + "NbCellsCheck = " + str(
NbCellsCheck)
PushButtLogSync.Enabled = not(NbCellsCheck = 0) ' and (UBound(App.JournCol1) >
-1) ' Forcément si NbCellsCheck > 0

' Plutôt que cet astuce de Tempo j'aurais pu compter les cellules cochées mais mêm
e problème pour les compter après un click sur une checkbox de la liste
' CompterCellsCheck
' ===========

'Dim iLign, DerLign as Int32 ' Pas UInt16 si Ubound -1 car vide
'#If DebugBuild Then
'Dim jNbre as Int32
'#EndIf
'
'NbCellsCheck = 0
'DerLign = UBound(App.JournCol1)
'
'For iLign = 0 to DerLign ' Si Ubound = -1 (pas de journal), on n'entre pas dans la b
oucle
'#If DebugBuild Then
'jNbre = Val(Me.Cell(row, 1)) - 1 ' Indice de la ligne
'If (Me.Cell(row, 0) = CellSsCheckBox) Xor (App.JournCol1(jNbre) = "NeRienFaire") T
hen
'MsgBox "row = " + str(row) + EndOfLine + "jNbre = " + str(jNbre) + EndOfLine + "'
" + Me.Cell(row, 0) + "'" + App.JournCol1(jNbre)
'End If
'#EndIf
'If not(Me.Cell(row, 0) = CellSsCheckBox) Then ' not(App.JournCol1(jNbre) = "NeRie
nFaire")
'If ListBoxLog.CellCheck(iLign, 0) Then NbCellsCheck = NbCellsCheck + 1 ' Sinon il
n'y a pas de checkbox
```

```
    'End If
    'Next iLign
    '
    'Return NbCellsCheck

End Sub

End Class
```

Class WinInfos

Inherits Window

WinInfos.CancelClose:

```
Function CancelClose(appQuitting as Boolean) As Boolean

    ' C'est une  MovableModal  ou  SheetWindows  donc normalement on ne peut pas q
    uitter car menu désactivé

    If DrapMovableModal Then Beep ' On a tenté de quitter via le Dock

    Return DrapMovableModal ' Close annulé si  True


    ' Je préfère créé une nouvelle variable rien que pour ça car moins risqué en cas de
    modif futur

    ' If not DrapMenu Then Beep ' On a tenté de quitter via le Dock

    ' Return (not DrapMenu) ' Close annulé si  True

End Function
```

WinInfos.Close:

```
Sub Close()

    DrapMenu = DrapMenuTemp
    ' @Menu@ EnableMenuItems ' Inutile
```

```
End Sub
```

WinInfos.Open:

```
Sub Open()

  DecPosCtrl(Self) ' Décalage des controls ( Label , TextField , Slider , etc. )
  #If TargetWin32 Then
    BoutonOk.TextSize = 12
  #EndIf

  DrapMovableModal = True
  DrapMenuTemp = DrapMenu ' Voir Notes dans WinMain
  DrapMenu = False
  ' @Menu@ EnableMenuItems ' Inutile

  ' StaticTextNomProg.Text = App.NomProg
  StaticTextVersion.Text = App.ShortVersion '  = App.VersionProg  Ca marche pas de
   mettre direct  #App.ShortVersion  dans les properties

  #If RBVersion < 2013 Then ' RealStudio , If InStr(1, TypeCompil, "2012r") > 0 Then '
  RealStudio
    DevLogo = New Picture(146, 22, 32)
    DevLogo.Graphics.DrawPicture RS_Logo, 0, 0
    DevLogo.Mask.Graphics.DrawPicture RS_LogoMask, 0, 0
  #Else ' Xojo
    DevLogo = New Picture(128, 22, 32)
    DevLogo.Graphics.DrawPicture Xojo_Logo, 0, 0
    DevLogo.Mask.Graphics.DrawPicture Xojo_LogoMask, 0, 0
  #EndIf

  OnWeb1 = False
  OnWeb2 = False
  OnEmail1 = False
  OnEmail2 = False

  ' If not((FondBtnDon.Width = BevButtDon.Width) and (FondBtnDon.Height = BevButt
  Don.Height)) Then
  ' MsgBox "Error ! Please contact the authors." + EndOfLine + "FondBtnDon' size and
   BevButtDon' size must be identical !" + EndOfLine + "FondBtnDon : " + str(FondBtn
  Don.Width) + ", " + str(FondBtnDon.Height) + EndOfLine + "BevButtDon : " + str(Bev
  ButtDon.Width) + ", " + str(BevButtDon.Height)
  ' End If
```

```
End Sub
```

WinInfos.Paint:

```
Sub Paint(g As Graphics, areas() As REALbasic.Rect)

  g.DrawPicture WinMain.Icone128, (Self.Width/2) - 64, 14
  g.DrawPicture DevLogo, 20, Self.Height - 41

End Sub
```

WinInfos.FermeFentr:

```
Private Sub FermeFentr()

  DrapMovableModal = False

  Self.Close

End Sub

Private DevLogo As Picture


Private DrapMenuTemp As Boolean


Private DrapMovableModal As Boolean


Private OnEmail1 As Boolean


Private OnEmail2 As Boolean


Private OnWeb1 As Boolean


Private OnWeb2 As Boolean
```

WinInfos Control BoutonOk:

```
Sub Action()

   #If DebugBuild Then
      If Keyboard.AsyncAltKey Then App.DejaDonne = False
   #EndIf
   FermeFentr

End Sub
```

WinInfos Control Web2:

```
Sub MouseExit()

   If Me.Enabled Then Self.MouseCursor = System.Cursors.StandardPointer

End Sub

Sub MouseEnter()

   If Me.Enabled Then Self.MouseCursor = System.Cursors.FingerPointer

End Sub

Sub MouseUp(X As Integer, Y As Integer)

   ' MsgBox "(" + str(X) + "," + str(Y) + ")"
   If OnWeb2 Then
      OnWeb2 = False ' Car si on est sorti pendant maintien clic on aura le curseur m
      ain lors du relâché
      Self.MouseCursor = System.Cursors.StandardPointer ' Et si on est toujours dess
      us on partira sur la page web, dans les 2 cas il faut remettre curseur flèche
      If (X >= 0) and (X <= Me.Width) and (Y >= 0) and (Y <= Me.Height) Then Show
      URL(Me.Text) ' PAS EncodeURLComponent() car remplace aussi les '/' et ':'
   End If

End Sub
```

```
Function MouseDown(X As Integer, Y As Integer) As Boolean

  OnWeb2 = True

  Return True ' Pour pouvoir faire  MouseUp

End Function
```

WinInfos Control Web1:

```
Sub MouseExit()

  If Me.Enabled Then Self.MouseCursor = System.Cursors.StandardPointer

End Sub

Sub MouseEnter()

  If Me.Enabled Then Self.MouseCursor = System.Cursors.FingerPointer

End Sub

Sub MouseUp(X As Integer, Y As Integer)

  ' MsgBox "(" + str(X) + "," + str(Y) + ")"
  If OnWeb1 Then
    OnWeb1 = False ' Car si on est sorti pendant maintien clic on aura le curseur m
    ain lors du relâché
    Self.MouseCursor = System.Cursors.StandardPointer ' Et si on est toujours dess
    us on partira sur la page web, dans les 2 cas il faut remettre curseur flèche
    If (X >= 0) and (X <= Me.Width) and (Y >= 0) and (Y <= Me.Height) Then Show
    URL(Me.Text) ' PAS EncodeURLComponent() car remplace aussi les '/' et ':'
  End If

End Sub

Function MouseDown(X As Integer, Y As Integer) As Boolean

  OnWeb1 = True

  Return True ' Pour pouvoir faire  MouseUp

End Function
```

WinInfos Control Email2:

```
Sub MouseExit()

   If Me.Enabled Then Self.MouseCursor = System.Cursors.StandardPointer

End Sub

Sub MouseEnter()

   If Me.Enabled Then Self.MouseCursor = System.Cursors.FingerPointer

End Sub

Sub MouseUp(X As Integer, Y As Integer)

   ' MsgBox "(" + str(X) + "," + str(Y) + ")"
   If OnEmail2 Then
      OnEmail2 = False ' Car si on est sorti pendant maintien clic on aura le curseur
      main lors du relâché
      Self.MouseCursor = System.Cursors.StandardPointer ' Et si on est toujours dess
      us on partira sur la page web, dans les 2 cas il faut remettre curseur flèche
      If (X >= 0) and (X <= Me.Width) and (Y >= 0) and (Y <= Me.Height) Then
         ShowURL("mailto:" + EncodeURLComponent(Me.Text) + "?Subject=" + Enco
         deURLComponent("A propos de " + App.NomProg + " " + App.ShortVersion
          + " " + TypeCompil)) ' Fin ligne
      End If
   End If

End Sub

Function MouseDown(X As Integer, Y As Integer) As Boolean

   OnEmail2 = True

   Return True ' Pour pouvoir faire  MouseUp

End Function
```

WinInfos Control Email1:

```
Sub MouseExit()

   If Me.Enabled Then Self.MouseCursor = System.Cursors.StandardPointer

End Sub

Sub MouseEnter()

   If Me.Enabled Then Self.MouseCursor = System.Cursors.FingerPointer

End Sub

Function MouseDown(X As Integer, Y As Integer) As Boolean

   OnEmail1 = True

   Return True ' Pour pouvoir faire  MouseUp

End Function

Sub MouseUp(X As Integer, Y As Integer)

   ' MsgBox "(" + str(X) + "," + str(Y) + ")"
   If OnEmail1 Then
      OnEmail1 = False ' Car si on est sorti pendant maintien clic on aura le curseur
      main lors du relâché
      Self.MouseCursor = System.Cursors.StandardPointer ' Et si on est toujours dess
      us on partira sur la page web, dans les 2 cas il faut remettre curseur flèche
      If (X >= 0) and (X <= Me.Width) and (Y >= 0) and (Y <= Me.Height) Then
         ShowURL("mailto:" + EncodeURLComponent(Me.Text) + "?Subject=" + Enco
         deURLComponent("A propos de " + App.NomProg + " " + App.ShortVersion
          + " " + TypeCompil)) ' Fin ligne
      End If
   End If

End Sub
```

WinInfos Control BevButtDon:

```
Sub MouseEnter()

   If Me.Enabled Then Self.MouseCursor = System.Cursors.FingerPointer
```

```
End Sub

Sub MouseExit()

  If Me.Enabled Then Self.MouseCursor = System.Cursors.StandardPointer

End Sub

Sub Action()

  App.DejaDonne = True
  ShowURL(LienSitePerso + Lien_PayPage + App.NomProg)
  FermeFentr

End Sub
```

WinInfos Control BevButtDejDon:

```
Sub Action()

  App.DejaDonne = True
  FermeFentr

End Sub

End Class
```

Class WinNomConfig

```
Inherits Window

Private Const MaxCarNomConfig = 25
```

WinNomConfig.CancelClose:

```
Function CancelClose(appQuitting as Boolean) As Boolean

  ' C'est une  MovableModal  ou  SheetWindows  donc normalement on ne peut pas q
  uitter car menu désactivé
```

```
  If DrapMovableModal Then Beep ' On a tenté de quitter via le Dock

  Return DrapMovableModal ' Close annulé si  True


  ' Je préfère créé une nouvelle variable rien que pour ça car moins risqué en cas de
  modif futur

  ' If not DrapMenu Then Beep ' On a tenté de quitter via le Dock

  ' Return (not DrapMenu) ' Close annulé si  True

End Function
```

WinNomConfig.Close:

```
Sub Close()

  DrapMenu = DrapMenuTemp
  ' @Menu@ EnableMenuItems ' Inutile

End Sub
```

WinNomConfig.Open:

```
Sub Open()

  DecPosCtrl(Self) ' Décalage des controls ( Label , TextField , Slider , etc. )
  #If TargetWin32 Then
    PushButtEnreg.TextSize = 12
    PushButtAnnul.TextSize = 12
  #EndIf

  DrapMovableModal = True
  DrapMenuTemp = DrapMenu ' Voir Notes dans WinMain
  DrapMenu = False
  ' @Menu@ EnableMenuItems ' Inutile

  EditFieldNomConfig.Text = WinMain.NomConfig
  EditFieldNomConfig.SelStart = 0
  EditFieldNomConfig.SelLength = Len(EditFieldNomConfig.Text)
```

```
   EditFieldNomConfig.SetFocus

End Sub

Private DrapMenuTemp As Boolean


Private DrapMovableModal As Boolean
```

WinNomConfig Control EditFieldNomConfig:

```
Sub TextChange()

   If Me.Text = "" Then
      PushButtEnreg.Enabled = False
   Else
      PushButtEnreg.Enabled = True
      If Len(Me.Text) > MaxCarNomConfig Then
         Beep
         Me.Text = Left(Me.Text, MaxCarNomConfig)
      End If
   End If

End Sub
```

WinNomConfig Control PushButtAnnul:

```
Sub Action()

   WinMain.NomConfig = ""

   DrapMovableModal = False

   Self.Close

End Sub
```

WinNomConfig Control PushButtEnreg:

```
Sub Action()

  WinMain.NomConfig = EditFieldNomConfig.Text
  ' NON car on a fait MyLenUTF8 : Left(…, MaxCarNomConfig) ' Des fois que merdé d
  ans Change EditField

  DrapMovableModal = False

  Self.Close

End Sub

End Class
```

Class WinPrefs

Inherits Window

WinPrefs.CancelClose:

```
Function CancelClose(appQuitting as Boolean) As Boolean

  ' C'est une  MovableModal  ou  SheetWindows  donc normalement on ne peut pas q
  uitter car menu désactivé

  If DrapMovableModal Then Beep ' On a tenté de quitter via le Dock

  Return DrapMovableModal ' Close annulé si  True


  ' Je préfère créé une nouvelle variable rien que pour ça car moins risqué en cas de
  modif futur

  ' If not DrapMenu Then Beep ' On a tenté de quitter via le Dock

  ' Return (not DrapMenu) ' Close annulé si  True

End Function
```

WinPrefs.Close:

```
Sub Close()

   DrapMenu = DrapMenuTemp
   ' @Menu@ EnableMenuItems ' Inutile

End Sub
```

WinPrefs.Open:

```
Sub Open()

   DecPosCtrl(Self) ' Décalage des controls ( Label , TextField , Slider , etc. )
   #If TargetWin32 Then
      PushButtOk.TextSize = 12
      PushButtAnnuler.TextSize = 12
   #EndIf

   DrapMovableModal = True
   DrapMenuTemp = DrapMenu ' Voir Notes dans WinMain
   DrapMenu = False
   ' @Menu@ EnableMenuItems ' Inutile

   CheckBoxAutoUpdt.Value = App.AutoCheckUpdt

   CheckBoxAffHlpTg.Value = App.AffHelpTag

   CheckBoxOuvLogLch.Value = App.OuvLogLch
   CheckBoxOuvLogApS.Value = App.OuvLogApS

   CheckBoxAlertReglDef.Value = App.AlertReglDef

   CheckBoxAutorisVol.Value = App.AutorisVol

   CheckBoxRelPathLog.Value = App.RelPathLog

   CheckBoxCoulListLog.Value = App.CoulDsListLog

   CheckBoxDelTrash.Value = App.DelTrash ' NON, plus maintenant car je fais plus ba
   s : CheckBoxDelTrash  est à vrai par défaut pour être sûr de faire son event Action s
   i faux pour les  RadioButtAmovTrash(iNbre).Enabled
   ' For iNbre = 0 to 2 ' Les autres sont à faux à l'ouverture
   ' RadioButtAmovTrash(iNbre).Value = (iNbre = RadioButtAmovTrash(App.AmovTras
   h).Value)
```

```
  ' Il est inutile de mettre les autres à Faux comme ci-dessus, c'est automatique quan
  d on en met un autre à Vrai
  ' Inutile  RadioButtAmovTrash(iNbre).Enabled = App.DelTrash  Fait dans l'event Acti
  on de  CheckBoxDelTrash
  ' Next iNbre
  RadioButtAmovTrash(App.AmovTrash).Value = True
  If App.NomDossHistTrash = "" Then
    App.NomDossHistTrash = "Shouldn't be empty"
    BeepNbT(3)
    ' NON, la fenêtre n'est pas encore affichée et ça ouvre SheetWindows sur MsgBo
    x : MsgBox "Error ! Please contact the authors." + EndOfLine +"WinPrefs.Open,
    App.NomDossHistTrash shouldn't be empty "."
  End If
  MemNomHistTrash = App.NomDossHistTrash
  EditFieldNomHistTrash.Text = App.NomDossHistTrash

  DisEnabRdButtAmovTrash(CheckBoxDelTrash.Value) ' = App.DelTrash

  EdFieldTimeOutCopy.Text = Format(App.TimeOutCopy, F_MBillionSsEsp)

  CheckBoxSfx.Value = App.Sfx

  InitWinPref = True

End Sub
```

WinPrefs.DisEnabRdButtAmovTrash:

```
Private Sub DisEnabRdButtAmovTrash(TampDrap as Boolean)

  Dim iNbre as Int16

  For iNbre = 0 to 2
    RadioButtAmovTrash(iNbre).Enabled = TampDrap
  Next iNbre

End Sub

Private DrapMenuTemp As Boolean


Private DrapMovableModal As Boolean
```

```
Private InitWinPref As Boolean


Private MemNomHistTrash As String
```

WinPrefs Control PushButtOk:

```
Sub Action()

   Dim iNbre as Int16

   App.AutoCheckUpdt = CheckBoxAutoUpdt.Value

   If CheckBoxAffHlpTg.Value Xor App.AffHelpTag Then ' Pas même valeur
      App.AffHelpTag = CheckBoxAffHlpTg.Value
      WinMain.AffMasqHelpTag("", "") ' Jamais fermé
      WinRegl.AffMasqHelpTag("", "") ' Jamais fermé mais invisible
      If WinMain.BevButtAffBatch.Value Then WinBatch.AffMasqHelpTag("", "")
      If WinMain.BevButtAffLog.Value Then WinLog.AffMasqHelpTag("", "")
   End If ' Sinon c'est que pas changé

   App.OuvLogLch = CheckBoxOuvLogLch.Value
   App.OuvLogApS = CheckBoxOuvLogApS.Value

   App.AlertReglDef = CheckBoxAlertReglDef.Value

   App.AutorisVol = CheckBoxAutorisVol.Value

   App.RelPathLog = CheckBoxRelPathLog.Value ' On ne récrée pas tout le journal ave
   c les path tronqués, ça changera à la procgaine synchro

   If not(App.CoulDsListLog = CheckBoxCoulListLog.Value) Then ' Else tout reste pareil
      App.CoulDsListLog = CheckBoxCoulListLog.Value
      If WinMain.BevButtAffLog.Value Then
         ' Beep
         ' WinLog.ListBoxLog.Refresh
         ' WinLog.UpdateNow
         ' App.DoEvents
         WinLog.Show ' Rien n'y fait, il n'y a QUE .Show qui redessine les couleurs
      End If
```

```
  End If

  App.DelTrash = CheckBoxDelTrash.Value
  For iNbre = 0 to 2
    If RadioButtAmovTrash(iNbre).Value Then App.AmovTrash = iNbre
  Next iNbre
  App.NomDossHistTrash = EditFieldNomHistTrash.Text

  App.TimeOutCopy = Val(EdFieldTimeOutCopy.Text)

  App.Sfx = CheckBoxSfx.Value

  DrapMovableModal = False
  Self.Close

End Sub
```

WinPrefs Control PushButtAnnuler:

```
Sub Action()

  DrapMovableModal = False
  Self.Close

End Sub
```

WinPrefs Control CheckBoxDelTrash:

```
Sub Action()

  If InitWinPref Then DisEnabRdButtAmovTrash(Me.Value)

End Sub
```

WinPrefs Control EditFieldNomHistTrash:

```
Sub TextChange()

  Dim iNbre, rep as Int16 ' Quasiment la même chose que WinMain.EditFieldTrashBegi
  n
```

```
  Dim DrapErr as Boolean
  Const MaxCaracts = 30 ' C'est bien assez long
  Const CarAccept = """!#$%&'()*+,-/ 0123456789;<=>?@ABCDEFGHIJKLMNOPQRST
  UVWXYZabcdefghijklmnopqrstuvwxyz[\]_{|}~†°¢£§•¶ß®©™≠ÆØ∞±≤≥¥µ∂∑∏π∫ªº
  Ωæø¿¡¬√ƒ≈Δ«»… Œœ-—“”‘’÷◇€"
  ' Je n'autorise pas le point .  Tous ces caractères n'ont pas de caractères combinés, i
  ls font tous 1 de Len(). Il y a espace normal et espace insécable

  If InitWinPref Then
    If Me.Text = "" Then ' Len(Me.Text) = 0
      DrapErr = True
    ElseIf (Me.Text = " ") or (Len(Me.Text) > MaxCaracts) Then ' Je n'autorise pas esp
    ace seul parce que trop risqué d'effacer élément commençant par espace car il
    peut y avoir des erreurs
      DrapErr = True
    Else
      DrapErr = False
      For iNbre = 1 to Len(Me.Text)
        If InStr(CarAccept, Mid(Me.Text, iNbre, 1)) = 0 Then
          DrapErr = True
          Exit
        End If
      Next iNbre
    End If

    If DrapErr Then ' Il y a un caractère interdit ou pas du tout ou trop de caractère
    s
      Beep
      rep = MyDialogBox(FracTexte(Txt_CarAccept, 1) + str(MaxCaracts) + FracT
      exte(Txt_CarAccept, 2) + CarAccept, "", Btn_Ok, "", "", 0)
      Me.Text = MemNomHistTrash ' On remet à la valeur d'avant
    Else
      MemNomHistTrash = Me.Text
    End If
  End If ' InitWinPref

End Sub
```

WinPrefs Control EdFieldTimeOutCopy:

```
Sub TextChange()

  Dim TampNbre as Int32 ' Des fois qu'un nombre négatif soit entré mais on ne doit p
  as pouvoir car mask
```

```
  Dim TampText as String ' Le mask autorise 6 caractères pour automatiquement écri
  re 65535 si on entre un plus grand nombre
  Const MinVal = 1 ' Exactement la même chose que EdFieldMargeTps et EdFieldTime
  r sauf pour valeur mini et maxi
  Const MaxVal = 65535 ' Max UInt16

  If InitWinPref Then
    TampText = Me.Text
    While (Left(TampText, 1) = "0") and (Len(TampText) > 1)
      TampText = Mid(TampText, 2)
    Wend

    TampNbre = Val(TampText)
    ' MsgBox "TampNbre = " + str(TampNbre)
    If TampNbre < MinVal Then ' Au moins 1 seconde de Timeout
      Beep ' Pour dire que truc pas normal, fais chier de faire une DisplayDialog
      encore !
      Me.Text = Format(App.TimeOutCopy, F_MBillionSsEsp) ' Valeur d'avant
    ElseIf TampNbre > MaxVal Then ' Pas même valeur que Mask (999999) car UInt
    16 0 à 65535
      Beep ' Pour dire que truc pas normal, fais chier de faire une DisplayDialog
      encore !
      Me.Text = Format(MaxVal, F_MBillionSsEsp) ' Val maxi
      ' NON, que si valide Pref par Btn_Ok : App.TimeOutCopy = MaxVal
    ElseIf not(Me.Text = TampText) Then
      ' If App.Sfx Then Bop.Play
      Me.Text = TampText
      ' NON, que si valide Pref par Btn_Ok : App.TimeOutCopy = TampNbre ' = V
      al(TampText)
    Else ' L'entrée est bonne
      ' NON, que si valide Pref par Btn_Ok : App.TimeOutCopy = TampNbre ' = V
      al(TampText)
    End If
    ' #If DebugBuild Then
    ' NON, que si valide Pref par Btn_Ok : If not(Format(App.TimeOutCopy, App.F_M
    BillionSsEsp) = Me.Text) Then MsgBox "Problème Tom, EdFieldTimeOutCopy.Te
    xtChange, TimeOutCopy = " + Format(App.TimeOutCopy, App.F_MBillionSsEsp)
     + " ≠ EdFieldTimeOutCopy = " + Me.Text
    ' #EndIf

  End If '  InitWinPref

End Sub
```

WinPrefs Control CheckBoxAutorisVol:

```
Sub Action()

  If InitWinPref and Me.Value Then ' Pas d'alerte si on décoche
    Beep
    If MyDialogBox(Txt_AlertAutorisVol + EndOfLine + Txt_VoirFAQ_HT, "", Btn_An
    nuler, Btn_Cont, "", 1) = 1 Then Me.Value = False ' On re-décoche
  End If

End Sub

End Class
```

Class WinBatch

Inherits Window

```
Private Const NbMsDsMin = 60000
```

WinBatch.CancelClose:

```
Function CancelClose(appQuitting as Boolean) As Boolean

  Return ((not DrapMenu) and appQuitting) ' Si on retourne Vrai la fenêtre ne se ferme
  ra pas
  ' Voir  WinMain.CancelClose  Note : On n'utilise pas  App.DragDropOuv  car cette fe
  nêtre Log pas ouverte si DragDropOuv
  ' Si on quitte par le Dock on annulerait la fermeture de WinMain mais pas celle-ci q
  ui se fermerait
  '  donc on perdrait les coordonnées de la fenêtre

End Function
```

WinBatch.KeyDown:

```
Function KeyDown(Key As String) As Boolean

  ' Même chose dans chaque  event KeyDown  des autres fenêtres et que dans ActBut
  tSync
```

```
  If App.SyncEnCours or App.LogSEnCours Then
    ' If (Keyboard.AsynckeyDown(&h35) or (Key = Chr(27))) and (not WinMain.Arret
    Urg) Then ' Touche Escape , pour pas faire 2 fois Zoum (procédure Synchro)
    ' Touche  Escape dans les 2 cas, je faisais les 2 tests !
    If (Key = Chr(27)) and (not WinMain.ArretUrg) Then ' Touche Escape , pour pas f
    aire 2 fois Zoum (procédure Synchro)
      WinMain.ArretUrg = True
      If App.Sfx Then
        Zoum.Play ' Pour signifier prise en compte  ArretUrg     On ne l'entend
        s pas si coup de frein juste après, mais des fois
        While Zoum.IsPlaying '                                   que ce soit plu
        s long pour stopper tout
        Wend
      End If
      ' NON, NE PAS faire boucle ci-dessous, sinon cet event (clic sur Bouton Syn
      chro/Arrêter) ne se fait QU'à la fin de la sychro, donc redémarre et arrête a
      ussi sec
      ' While (App.SyncEnCours or App.LogSEnCours) ' Si ça marche !  (NE PAS fai
      re ça car garde la main et le Thread ne se fini pas)
      ' Wend '                                                Même sans DoEvents
      Pause(10, False) ' Pour éviter 2 appuie trop vif
      ' If App.Sfx Then Coup_de_freins.Play ' A été fait dans le Thread de Synchro
    End If
  End If

  Return (not(Key = Chr(9))) ' Car si Tab on passe d'un TextField à l'autre

End Function
```

WinBatch.Moved:

```
Sub Moved()

  If App.InitProg Then RePosFenetr(False, Keyboard.AsyncAltKey) ' Inutile car je ne fais
   plus le DoEvents  and DrapRePosFenetr

End Sub
```

WinBatch.Open:

```
Sub Open()
```

```
  DecPosCtrl(Self) ' Décalage des controls ( Label , TextField , Slider , etc. )
  #If TargetWin32 Then ' Note : WinBatch ne peut pas être ouverte pendant une synch
  ro car BevButtAffBatch désactivé
    PushButtBatchSync.TextSize = 12
  #EndIf

  AffMasqHelpTag("", "")

End Sub
```

WinBatch.ActButtBatchSync:

```
Sub ActButtBatchSync()

  #If DebugBuild Then
    Dim TampDrapBatchEnab as Boolean

    TampDrapBatchEnab = DrapBatchEnab
    UpdtBatchEnabTim(True, False) ' On ne touche PAS à TimerBatch car fout la mer
    de au lancement de l'application sinon (remet Timer à mode 2 trop tôt)
    If not(TampDrapBatchEnab = DrapBatchEnab) Then MsgBox "Erreur Tom. ActBut
    tBatchSync" + EndOfLine + "DrapBatchEnab = " + Cstr(DrapBatchEnab) + EndOf
    Line + "TampDrapBatchEnab = " + Cstr(TampDrapBatchEnab)
    App.EnableMenuBoutons(False) ' Pour FichierBatchSync , pas forcément utile car
     le menu se mettra automatiquement à jour dès qu'on l'affichera, mais je préfèr
    e qu'il soit à jour
  #EndIf
  ' ListBoxBatch.Selected() = False ' Inutile car on ChangeConfig dans ThreadSync

  If DrapBatchEnab Then
    WinMain.DrapBatch = True
    WinMain.L_ThreadSync ' Launch ThreadSync (.Run) ' On lance la Synchro dans u
    n Thread
  Else
    MsgBox "Error ! Please contact the authors." + EndOfLine + "ActButtBatchSync s
    houldn't have been runned because DrapBatchEnab is False."
  End If

End Sub
```

WinBatch.AffMasqHelpTag:

```
Sub AffMasqHelpTag(CeCtrlNomId as String, CeCtrlHlpTg as String)

  Dim iNbre, Fin_iNbre as Int16 ' Envoyer  RectControl.Name  avec éventuellement son
    Index  (si tableau de RadioButton par exemple) pour ne mettre que lui à jour
  Dim TampNom as String '              Exemple : AffMasqHelpTag(StaticTextNbSpam.Na
  me, "TextDuHelpTag")
  Dim CeCtrl as Control '          ou si index 1 : AffMasqHelpTag(StaticTextNbSpam.Na
  me + "1", "TextDuHelpTag") ' ou  + str(index)
  Dim CeRectCtrl as RectControl
  Static MemRctCtrlNom(-1), MemHlpTgTxt(-1) as String

  Fin_iNbre = ControlCount - 1
  If UBound(MemHlpTgTxt) = -1 Then ' = UBound(MemRctCtrlNom)
    ' MsgBox "On mémorise HelpTag de " + str(Fin_iNbre + 1) + " controls." + End
    OfLine + "CeCtrlNomId = '" + CeCtrlNomId + "'"
    #If DebugBuild Then
      If not(CeCtrlNomId = "") Then MsgBox "Tu t'es gourré Tom, CeCtrlNomId de
      vrait être vide " !" + EndOfLine + "'" + CeCtrlNomId + "'"
    #EndIf
    For iNbre = 0 to Fin_iNbre
      CeCtrl = Self.Control(iNbre)
      ' MsgBox CeCtrl.Name + EndOfLine + str(CeCtrl.Index) + EndOfLine + "Rec
      tControl ? : " + CStr(CeCtrl isa RectControl)
      If CeCtrl isa RectControl Then
        CeRectCtrl = RectControl(CeCtrl) ' Et NON RectControl(iNbre)
        TampNom = CeCtrl.Name
        #If DebugBuild Then
          If (not(CeRectCtrl.Index = CeCtrl.Index) and (CeRectCtrl.Name = T
          ampNom)) Then MsgBox "Bizarre Tom, RectCtrl.Index ≠ Ctrl.Index
           :" + EndOfLine + str(CeRectCtrl.Index) + " ≠ " + str(CeCtrl.Index)
          + EndOfLine + "RectCtrl.Name ≠ Ctrl.Name :" + EndOfLine + str(C
          eRectCtrl.Name) + " ≠ " + str(TampNom)
        #EndIf
        If CeCtrl.Index > -1 Then TampNom = TampNom + str(CeCtrl.Index) '
        ATTENTION Int32, si pas un tableau de control alors = -2147483648
        ' MsgBox " id = " + str(CeCtrl.Index) + " = " + str(CeRectCtrl.Index) +
        EndOfLine + TampNom
        MemRctCtrlNom.Append TampNom
        MemHlpTgTxt.Append CeRectCtrl.HelpTag ' On mémorise qu'il y en ait
         ou non
        If not App.AffHelpTag Then CeRectCtrl.HelpTag = ""
        ' Si à Vrai alors l'helptag reste à sa valeur
      End If ' RectControl
    Next iNbre
```

```
Else ' On les a déjà mémorisés
  ' MsgBox "Nbre de Controls : " + str(Fin_iNbre + 1) + EndOfLine + "Nbre de Rec
  tControls : " + str(UBound(MemHlpTgTxt) + 1) + EndOfLine + "CeCtrlNomId = '
  " + CeCtrlNomId + """
  If CeCtrlNomId = "" Then ' On applique à tous
    For iNbre = 0 to Fin_iNbre ' Pas forcément = à  UBound(MemHlpTgTxt)  car
     tous ne sont pas RectControl, mais QUE Control
      CeCtrl = Self.Control(iNbre)
      ' MsgBox CeCtrl.Name + EndOfLine + str(CeCtrl.Index) + EndOfLine +
      "RectControl ? : " + CStr(CeCtrl isa RectControl)
      If CeCtrl isa RectControl Then
        CeRectCtrl = RectControl(CeCtrl) ' Et NON RectControl(iNbre)
        ' MsgBox str(CeRectCtrl.Index) + " = " + EndOfLine + str(CeCtrl.In
        dex) + EndOfLine + CeRectCtrl.Name + EndOfLine + CeRectCtrl.H
        elpTag
        If App.AffHelpTag Then
          CeRectCtrl.HelpTag = MemHlpTgTxt(iNbre) ' S'il n'y en avait p
          as ça reste à ""
        Else
          CeRectCtrl.HelpTag = ""
        End If
      End If ' RectControl
    Next iNbre

  Else ' On applique qu'au control indiqué
    iNbre = MemRctCtrlNom.IndexOf(CeCtrlNomId)
    If iNbre = -1 Then
      MsgBox "Error ! Please contact the authors, AffMasqHelpTag :" + EndO
      fLine + CeCtrlNomId + EndOfLine + "iNbre = -1"
    Else
      CeCtrl = Self.Control(iNbre)
      ' MsgBox CeCtrlNomId + EndOfLine + "iNbre = " + str(iNbre)
      ' Forcément : If CeCtrl isa RectControl Then
      CeRectCtrl = RectControl(CeCtrl) ' Et NON RectControl(iNbre)
      MemHlpTgTxt(iNbre) = CeCtrlHlpTg ' On met à jour la mémoire du hel
      ptag
      If App.AffHelpTag Then
        CeRectCtrl.HelpTag = CeCtrlHlpTg ' On met à jour le helptag lui-
        même
      Else
        If not(CeRectCtrl.HelpTag = "") Then MsgBox  "Error ! Please contac
        t the authors, AffMasqHelpTag :" + EndOfLine + CeCtrlNomId + "'
        helptag should be empty " :" + EndOfLine + """ + CeRectCtrl.HelpT
        ag + """
```

```
        End If
      End If

    End If ' On applique à tous ou au contrôle indiqué  CeCtrlNomId

  End If ' UBound(MemHlpTgTxt) = -1

End Sub
```

WinBatch.FaireListe:

```
Sub FaireListe()

  Dim iLigne, FinLigne as Int16

  ListBoxBatch.DeleteAllRows

  FinLigne = UBound(WinMain.CfNomConfig) - 1 ' Ou n'importe quel autre, Ligne 0 co
  rrespond à réglage 1 puisque réglage 0 est le réglage Dernière Config
  For iLigne = 0 to FinLigne
    ListBoxBatch.AddRow "" ' Ne marche pas d'ajouter direct  WinMain.CfBatch(iLign
    e)
    ListBoxBatch.CellCheck(iLigne, 0) = WinMain.CfBatch(iLigne + 1)
    ListBoxBatch.Cell(iLigne, 1) = WinMain.CfNomConfig(iLigne + 1)
  Next iLigne

  If WinMain.PopupConfig.ListIndex > 0 Then ListBoxBatch.Selected(WinMain.PopupC
  onfig.ListIndex - 1) = True

  UpdtBatchEnabTim(True, InitWinBatch) ' A faire APRES boucle ci-dessus
  PushButtBatchSync.Enabled = DrapBatchEnab
  CheckBoxBatchAuto.Enabled = DrapBatchEnab
  ' MsgBox "FichierBatchSync.Enabled = " + Cstr(FichierBatchSync.Enabled)
  App.EnableMenuBoutons(False) ' Pour FichierBatchSync , pas forcément utile car le
  menu se mettra automatiquement à jour dès qu'on l'affichera, mais je préfère qu'il
  soit à jour
  '  If DrapMenu Then  Je pourrais mettre ce test ci-dessus car si Faux c'est qu'on vie
  nt de WinMain.open et  EnableMenuItems  sera appelé via
  '  App.EnableMenuBoutons  à la fin de WinMain.Open  Mais je préfère l'appeler 2 foi
  s que de l'oublier si jamais j'appele FaireListe alors que DrapMenu à Faux (DialogBo
  x, etc.)

End Sub
```

WinBatch.FenetrInit:

```
Sub FenetrInit()

  #If DebugBuild Then ' Après avoir lu les Prefs car on initialise la Liste avec FaireListe
    avec les réglages
     If not(NbMsDsMin = 60000) Then
        StTextMn.Text = "sec"
        StTextMn.TextColor = &cFF0000
     End If
     If not(TimerBatch.Mode = Timer.ModeOff) Then MsgBox "Problème Tom." + End
     OfLine + "WinBatch.Open, TimerBatch.Mode devrait être Timer.ModeOff = 0 !"
     If not(TimerBatch.Period = 200) Then MsgBox "Problème Tom." + EndOfLine + "
     WinBatch.Open, TimerBatch.Period devrait être 200 !"
  #Else
     If not(NbMsDsMin = 60000) Then MsgBox "Tu t'es gourré Tom, NbMsDsMin dev
     rait faire 60000 et non " + Format(NbMsDsMin, F_MBillionSsEsp) + " mn/s !"
  #EndIf

  ListBoxBatch.ColumnType(0) = 2

  EdFieldTimer.Text = Format(App.BatchTempo, F_MBillionSsEsp) ' Minutes
  FaireListe '  UpdtBatchEnabTim  fait dedans

  If DrapMenu Then MsgBox "Error ! Please contact the authors." + EndOfLine + "WinB
  atch.Open, DrapMenu should be False because event WinMain.Open shouldn't be fi
  nished !"
  ' Voir note dans Timer de pourquoi Period à 200
  If CheckBoxBatchAuto.Value and DrapBatchEnab Then
     TimerBatch.Mode = Timer.ModeSingle ' 1 ' Sera mis à 2 dans le TimerBatch, et l
     a Période y sera également réglée
  Else
     TimerBatch.Period = App.BatchTempo * NbMsDsMin ' Car 1 période = 1 milliè
     me de sec, et EditFieldTimer est en minute
     ' TimerBatch.Mode reste à Timer.ModeOff ' 0
  End If
  ' App.EnableMenuBoutons(False) = EnableMenuItems  est appelé dans WinMain.Ope
  n apres avoir fait ce code ci

  ' MsgBox "Fin FenetrInit" + EndOfLine + "TimerBatch.Mode = " + str(TimerBatch.Mo
  de) + EndOfLine + "TimerBatch.Period = " + str(TimerBatch.Period) + EndOfLine + "
  App.BatchTempo = " + str(App.BatchTempo)
```

```
  InitWinBatch = True

End Sub
```

WinBatch.RePosFenetr:

```
Sub RePosFenetr(DrapMoveWM as Boolean, DrapDefPos as Boolean)

  Dim FenetrL, FenetrT as Int16 ' En Cocoa les Events se font en Live et ça merde si j'a
  ffecte directement  Self.Left  et  Self.Top
  Dim DrapMemDec, DrapPosFenetr as Boolean ' Pas vraiment utile ces Flags mais j'ai
  me autant ne pas refaire des calculs inutiles, de plus il y a le pb ci-dessus
  Static DecalageL, DecalageT, MemWMw, MemWMh as Int16 ' Static garde en mémoir
  e la valeur de la variable d'un appel de à l'autre de la Method

  ' Inutile car je ne fais plus le DoEvents : DrapRePosFenetr = False ' Voir note plus ba
  s au App.DoEvents
  If DrapMoveWM Then ' On a déplacé ou redimensionné WinMain, on vient donc de W
  inMain.Moved ou WinMain.Resized
    DrapMemDec = False ' On pourrait remémoriser  DecalageL  et  DecalageT  car
    on retrouverait le même nombre mais je n'aime pas
    DrapPosFenetr = True
    FenetrL = WinMain.Left + DecalageL
    FenetrT = WinMain.Top + DecalageT
    If Self.Left > (WinMain.Left + (MemWMw / 2)) Then FenetrL = FenetrL + (WinMai
    n.Width - MemWMw)
    If Self.Top > (WinMain.Top + (MemWMh / 2)) Then FenetrT = FenetrT + (WinMai
    n.Height - MemWMh)

  Else ' On vient de Self.Moved (ou des Prefs)
    DrapMemDec = True
    If DrapDefPos Then
      DrapPosFenetr = True
      FenetrL = WinMain.Left + WinMain.Width + 10 ' Seules ces 2 lignes sont à
      personnaliser suivant la position par défaut de la fenêtre
      FenetrT = WinMain.Top + (636 - Self.Height) ' Seules ces 2 lignes sont à pe
      rsonnaliser suivant la position par défaut de la fenêtre
    Else
      DrapPosFenetr = False
      FenetrL = Self.Left
      FenetrT = Self.Top
    End If
```

```
  End If

  If FenetrL < 0 Then
    DrapMemDec = True
    DrapPosFenetr = True
    FenetrL = 0
  ElseIf (FenetrL + Self.Width) > Screen(0).Width Then
    DrapMemDec = True
    DrapPosFenetr = True
    FenetrL = Screen(0).Width - Self.Width
  End If
  If FenetrT < 38 Then
    DrapMemDec = True
    DrapPosFenetr = True
    FenetrT = 38
  ElseIf (FenetrT + Self.Height) > Screen(0).Height Then
    DrapMemDec = True
    DrapPosFenetr = True
    FenetrT = Screen(0).Height - Self.Height
  End If

  ' Non, pas utile : App.DoEvents ' Comme ça l'event Moved de Self se fait alors que
  DrapRePosFenetr  est encore False
  ' Inutile car je ne fais plus le DoEvents : DrapRePosFenetr = True
  ' MsgBox "On vient de WinMain : " + Cstr(DrapMoveWM) + EndOfLine + "WinMain : "
   + str(WinMain.Left) + ", " + str(WinMain.Top) + EndOfLine + "Self : " + str(FenetrL)
  + ", " + str(FenetrT) + EndOfLine + "Mem décalage : " + Cstr(DrapMemDec) ' + "  "
  + Cstr(DrapRePosFenetr) ' Fin
  If DrapMemDec Then
    DecalageL = FenetrL - WinMain.Left
    DecalageT = FenetrT - WinMain.Top
    MemWMw = WinMain.Width
    MemWMh = WinMain.Height
  End If
  If DrapPosFenetr Then ' On va immédiatement revenir ici via l'event Self.Moved donc
   on va re-mémoriser  DecalageL  et  DecalageT
    Self.Left = FenetrL
    Self.Top = FenetrT
  End If

End Sub
```

WinBatch.UpdtBatchEnabTim:

```
Sub UpdtBatchEnabTim(TampDrapA as Boolean, TampDrapB as Boolean)

  Dim iLigne, FinLigne as Int16 ' Si un réglage checké ne se fait pas car bouton Synchr
  o non actif (pb avec dossier) le bouton BatchSync sera quand même actif,
  '  il y aura une ligne indiquant le problème dans le journal

  If TampDrapA Then ' On remet à jour  DrapBatchEnab
    DrapBatchEnab = False
    FinLigne = UBound(WinMain.CfNomConfig) - 1 ' Ou n'importe quel autre, Ligne
    0 correspond à réglage 1 puisque réglage 0 est le réglage Dernière Config
    For iLigne = 0 to FinLigne
      #If DebugBuild Then
        If not(ListBoxBatch.CellCheck(iLigne, 0) = WinMain.CfBatch(iLigne + 1))
         Then MsgBox "Erreur Tom. BtnBatchSEnab" + EndOfLine + Cstr(ListBo
        xBatch.CellCheck(iLigne, 0)) + " ≠ " + Cstr(WinMain.CfBatch(iLigne + 1
        ))
        If not(ListBoxBatch.Cell(iLigne, 1) = WinMain.CfNomConfig(iLigne + 1))
         Then MsgBox "Erreur Tom. BtnBatchSEnab" + EndOfLine + ListBoxBatc
        h.Cell(iLigne, 1) + " ≠ " + WinMain.CfNomConfig(iLigne + 1)
      #EndIf
      DrapBatchEnab = DrapBatchEnab or (ListBoxBatch.CellCheck(iLigne, 0))
    Next iLigne
  End If

  If TampDrapB Then ' Sinon on est dans test DebugBuild de ActButtBatchSync ou dan
  s WinBatch.Open et on le gère car peut être mis en mode 1
    If CheckBoxBatchAuto.Value and DrapBatchEnab Then
      TimerBatch.Mode = Timer.ModeMultiple ' 2
    Else
      TimerBatch.Mode = Timer.ModeOff ' 0 ' Si le Timer était sur le point de se
      déclencher (arrivé au terme de la tempo), ça annule l'Action
    End If
  End If

End Sub

DrapBatchEnab As Boolean


Private InitWinBatch As Boolean
```

WinBatch Control ListBoxBatch:

```
Function DragReorderRows(newPosition as Integer, parentRow as Integer) As Boolean

  Dim NouvPos, AncPos as Int16

  If not(Me.ListCount = Ubound(WinMain.CfNomConfig)) Then MsgBox "Error ! Please
  contact the authors." + EndOfLine + "ReorderCpte Pas bon nb de Regl dans liste." '
  Ca va planter plus loin
  NouvPos = newPosition + 1 ' Parce que le 1er élément qui est Dernier Réglages n'es
  t pas dans la ListBox
  AncPos = Me.ListIndex + 1
  ' MsgBox "newPosition = " + str(newPosition) + " + 1 = " + str(NouvPos) + EndOfLin
  e + "Me.ListIndex = " + str(Me.ListIndex) + " + 1 = " + str(AncPos) + EndOfLine + "
  parentRow = " + str(parentRow) + " (pour hierarchical ListBox)"

  If NouvPos > AncPos Then NouvPos = NouvPos + 1 ' Car la ligne où on se trouve n'e
  xistera plus, sinon en se déplaçant d'une ligne vers le haut on se retrouverait au mê
  me endroit

  If NouvPos > Ubound(WinMain.CfNomConfig) Then
    WinMain.CfNomConfig.Append WinMain.CfNomConfig(AncPos)
    WinMain.CfDossSource.Append WinMain.CfDossSource(AncPos)
    WinMain.CfDossCible.Append WinMain.CfDossCible(AncPos)
    WinMain.CfModeSync.Append WinMain.CfModeSync(AncPos)
    WinMain.CfGererDate.Append WinMain.CfGererDate(AncPos)
    WinMain.CfRemplRecent.Append WinMain.CfRemplRecent(AncPos)
    WinMain.CfTrashIfBegin.Append WinMain.CfTrashIfBegin(AncPos)
    WinMain.CfTextTrash.Append WinMain.CfTextTrash(AncPos)
    WinMain.CfIgnorElt.Append WinMain.CfIgnorElt(AncPos)
    WinMain.CfPremNiv.Append WinMain.CfPremNiv(AncPos)
    WinMain.CfIgnorEltList.Append WinMain.CfIgnorEltList(AncPos)
    WinMain.CfPasIgnor.Append WinMain.CfPasIgnor(AncPos)
    WinMain.CfPasIgnorList.Append WinMain.CfPasIgnorList(AncPos)
    WinMain.CfSyncIcones.Append WinMain.CfSyncIcones(AncPos)
    WinMain.CfBarOutils.Append WinMain.CfBarOutils(AncPos)
    WinMain.CfSimul.Append WinMain.CfSimul(AncPos)
    WinMain.CfMargeTps.Append WinMain.CfMargeTps(AncPos)
    WinMain.CfDossPackFich.Append WinMain.CfDossPackFich(AncPos)
    WinMain.CfDossExtFich.Append WinMain.CfDossExtFich(AncPos)
    WinMain.CfUseShell.Append WinMain.CfUseShell(AncPos)
    WinMain.CfCdeCpShell.Append WinMain.CfCdeCpShell(AncPos)
    WinMain.CfCopyRBerr.Append WinMain.CfCopyRBerr(AncPos)
```

```
    WinMain.CfBatch.Append WinMain.CfBatch(AncPos)
Else
    WinMain.CfNomConfig.Insert NouvPos, WinMain.CfNomConfig(AncPos)
    WinMain.CfDossSource.Insert NouvPos, WinMain.CfDossSource(AncPos)
    WinMain.CfDossCible.Insert NouvPos, WinMain.CfDossCible(AncPos)
    WinMain.CfModeSync.Insert NouvPos, WinMain.CfModeSync(AncPos)
    WinMain.CfGererDate.Insert NouvPos, WinMain.CfGererDate(AncPos)
    WinMain.CfRemplRecent.Insert NouvPos, WinMain.CfRemplRecent(AncPos)
    WinMain.CfTrashIfBegin.Insert NouvPos, WinMain.CfTrashIfBegin(AncPos)
    WinMain.CfTextTrash.Insert NouvPos, WinMain.CfTextTrash(AncPos)
    WinMain.CfIgnorElt.Insert NouvPos, WinMain.CfIgnorElt(AncPos)
    WinMain.CfPremNiv.Insert NouvPos, WinMain.CfPremNiv(AncPos)
    WinMain.CfIgnorEltList.Insert NouvPos, WinMain.CfIgnorEltList(AncPos)
    WinMain.CfPasIgnor.Insert NouvPos, WinMain.CfPasIgnor(AncPos)
    WinMain.CfPasIgnorList.Insert NouvPos, WinMain.CfPasIgnorList(AncPos)
    WinMain.CfSyncIcones.Insert NouvPos, WinMain.CfSyncIcones(AncPos)
    WinMain.CfBarOutils.Insert NouvPos, WinMain.CfBarOutils(AncPos)
    WinMain.CfSimul.Insert NouvPos, WinMain.CfSimul(AncPos)
    WinMain.CfMargeTps.Insert NouvPos, WinMain.CfMargeTps(AncPos)
    WinMain.CfDossPackFich.Insert NouvPos, WinMain.CfDossPackFich(AncPos)
    WinMain.CfDossExtFich.Insert NouvPos, WinMain.CfDossExtFich(AncPos)
    WinMain.CfUseShell.Insert NouvPos, WinMain.CfUseShell(AncPos)
    WinMain.CfCdeCpShell.Insert NouvPos, WinMain.CfCdeCpShell(AncPos)
    WinMain.CfCopyRBerr.Insert NouvPos, WinMain.CfCopyRBerr(AncPos)
    WinMain.CfBatch.Insert NouvPos, WinMain.CfBatch(AncPos)
End If

If NouvPos > AncPos Then ' Même si on a fait + 1 plus haut il était déjà supérieur
    NouvPos = NouvPos - 1 ' On renlève car on va maintenant supprimer la ligne
Else ' If AncPos > NouvPos Then
    AncPos = AncPos + 1 ' Car on vient d'insérer une ligne avant lui
End If

WinMain.CfNomConfig.Remove AncPos
WinMain.CfDossSource.Remove AncPos
WinMain.CfDossCible.Remove AncPos
WinMain.CfModeSync.Remove AncPos
WinMain.CfGererDate.Remove AncPos
WinMain.CfRemplRecent.Remove AncPos
WinMain.CfTrashIfBegin.Remove AncPos
WinMain.CfTextTrash.Remove AncPos
WinMain.CfIgnorElt.Remove AncPos
WinMain.CfPremNiv.Remove AncPos
WinMain.CfIgnorEltList.Remove AncPos
```

```
  WinMain.CfPasIgnor.Remove AncPos
  WinMain.CfPasIgnorList.Remove AncPos
  WinMain.CfSyncIcones.Remove AncPos
  WinMain.CfBarOutils.Remove AncPos
  WinMain.CfSimul.Remove AncPos
  WinMain.CfMargeTps.Remove AncPos
  WinMain.CfDossPackFich.Remove AncPos
  WinMain.CfDossExtFich.Remove AncPos
  WinMain.CfUseShell.Remove AncPos
  WinMain.CfCdeCpShell.Remove AncPos
  WinMain.CfCopyRBerr.Remove AncPos
  WinMain.CfBatch.Remove AncPos

  ' WinMain.RemplReglage(0) ' On pourrait sauvegarder comme défaut les réglages ac
  tuels
  App.initProg = False
  WinMain.FairePopupConfig
  WinMain.PopupConfig.ListIndex = NouvPos ' Ligne sélectionnée,  initProg  reste à Fa
  lse pour cette instruction car pas besoin de remettre à jour puisque fait via CellClic
  k
  App.initProg = True
  ' WinBatch.FaireListe ' Inutile car la liste affichée a été réordonnée automatiquement

  Return False ' Pour autoriser prochain DragReorder

End Function

Sub CellAction(row As Integer, column As Integer)

  Dim iLigne, FinLigne as Int16 ' Note : CellClick est exécutée avant CellAction, mais v
  oir note plus bas

  If InitWinBatch Then
    ' MsgBox "CellAction ligne " + str(row) +" : " + Cstr(Me.CellCheck(row, 0))

    WinMain.CfBatch(row + 1) = Me.CellCheck(row, 0)

    UpdtBatchEnabTim(True, True)
    PushButtBatchSync.Enabled = DrapBatchEnab
    CheckBoxBatchAuto.Enabled = DrapBatchEnab
    App.EnableMenuBoutons(False) ' Pour FichierBatchSync , pas forcément utile car
     le menu se mettra automatiquement à jour dès qu'on l'affichera, mais je préfèr
    e qu'il soit à jour
```

```
      ' Dans CellClick, la ligne cliquée est automatiquement sélectionnée, sauf si Cell
      Action est exécuté derrière, ce qui est le cas donc je fais par programme
      FinLigne = UBound(WinMain.CfNomConfig) - 1 ' Ou n'importe quel autre, Ligne
      0 correspond à réglage 1 puisque réglage 0 est le réglage Dernière Config
      For iLigne = 0 to FinLigne
         Me.Selected(iLigne) = (row = iLigne)
      Next iLigne

   End If '  InitWinBatch

End Sub

Function CellClick(row as Integer, column as Integer, x as Integer, y as Integer) As Boolea
n

   ' MsgBox "CellClick ligne " + str(row) +" : " + Cstr(Me.CellCheck(row, 0))

   If not App.initProg Then MsgBox "Error ! Please contact the authors." + EndOfLine +
   "ListboxBatch.CellClick, initProg should be True."

   App.SelLBoxBatch = False ' On ne resélectionnera pas la ligne dans ChangeConfig c
   ar fait automatiquement, sauf si on a cliqué une CheckBox (voir note dans CellActio
   n)
   WinMain.PopupConfig.ListIndex = (row + 1) ' Note App.initProg forcément à True
   App.SelLBoxBatch = True

End Function

Function KeyDown(Key As String) As Boolean

   Dim iLigne, NbElt as Int16

   If App.initProg Then
      iLigne = Me.ListIndex
      NbElt = Me.ListCount

      If DrapMenu Then ' De toute façon ListBox désactivée sinon
         ' MsgBox "KeyDown, code ascii = " + str(Asc(Key)) + EndOfLine + "iLigne =
          " + str(iLigne)
         If Key = chr(30) Then ' Flèche haut ' Keyboard.AsyncKeyDown(126)
            If iLigne = -1 Then
               iLigne = 1 ' Si aucune ligne n'était sélectionnée la première (0) le s
               era, soit la deuxième du PopupMenu (1)
               ' Else
```

```
      ' Sinon on ne retranche pas 1 car il y a 1 de décalage entre Popup
      Menu et ListBox
      ' Et si iLigne = 0 (première ligne) on y restera et il ne se passera ri
      en plus bas
    End If
  ElseIf Key = chr(31) Then ' Flèche bas ' Keyboard.AsyncKeyDown(125)
    If iLigne = -1 Then
      iLigne = 1 ' Si aucune ligne n'était sélectionnée la première (0) le s
      era, soit la deuxième du PopupMenu (1)
    Else
      iLigne = iLigne + 2 ' +2 et non +1 car il y a 1 de décalage entre P
      opupMenu et ListBox
      ' Et si on était sur la dernière ligne (NbElt - 1) on y restera et il ne
      se passera rien plus bas
    End If
  Else
    iLigne = -2 ' On ne fera rien car mauvaise touche
  End If
  If (iLigne > 0) and (iLigne <= NbElt) and (not Keyboard.AsyncOSKey) Then'
  Note :  OSKey  =  CommandKey Then
    ' Note : Si liste vide NbElt = 0 donc on n'entre pas dans ce If Then
    ' MsgBox "iLigne = " + str(iLigne)
    App.SelLBoxBatch = False ' On ne resélectionnera pas la ligne dans Ch
    angeConfig sinon ça la décale de 2 car RealBasic fait le décalage à la fi
    n de cet event
    WinMain.PopupConfig.ListIndex = iLigne ' Note App.initProg forcémen
    t à True
    App.SelLBoxBatch = True
    ' Else
    ' Si on est hors champs (hors lignes) on ne fait rien
  End If

Else
  Beep
  MsgBox "Error ! Please contact the authors." + EndOfLine + "ListboxBatch.
  KeyDown, DrapMenu is False"

End If

' Else ' Non pas d'alerte car ça me la faisait parfois, pourtant il n'y a pas de DoE
vent ni ici ni dans les procédures appelées ?!!!
'   Beep
'   MsgBox "Error ! Please contact the authors." + EndOfLine + "ListboxBatch.Ke
yDown, initProg should be True."
```

```
  End If '  initProg

  Return False

End Function

Sub Open()

  Me.ColumnType(0) = ListBox.TypeCheckbox ' Ca marchait sans que je mette ça ava
  nt mais dans ConvertEncoding ça ne marchait pas sans, j'ai dû le mettre !!!
  ' Me.ColumnType(1) = ListBox.TypeEditable ' NON, reste à non éditable
  ' Me.ColumnType(1) = ListBox.TypeDefault ' Same as its column type, comme ici on
   est dans colonne type
  Me.ColumnType(1) = ListBox.TypeNormal

End Sub

Function CellBackgroundPaint(g As Graphics, row As Integer, column As Integer) As Bool
ean

  If (row mod 2) = 0 Then
    g.ForeColor = App.ColBackList ' J'ai mis comme iTunes, avant c'était RGB(232,2
    35,255)
    g.FillRect 0, 0, g.Width, g.Height
    ' Else
    ' Reste à sa valeur (blanc)
  End If

End Function
```

WinBatch Control PushButtBatchSync:

```
Sub Action()

  ActButtBatchSync

End Sub
```

WinBatch Control CheckBoxBatchAuto:

```
Sub Action()

   If InitWinBatch Then
      UpdtBatchEnabTim(False, True)
      App.EnableMenuBoutons(False) ' Pour FichierBatchSync , pas forcément utile car
       le menu se mettra automatiquement à jour dès qu'on l'affichera, mais je préfèr
      e qu'il soit à jour

      ' MsgBox "TimerBatch.Mode = " + str(TimerBatch.Mode) + EndOfLine + "TimerB
      atch.Period = " + str(TimerBatch.Period) + EndOfLine + "App.BatchTempo = "
      + str(App.BatchTempo)
   End If '  InitWinBatch

End Sub
```

WinBatch Control TimerBatch:

```
Sub Action()

   ' MsgBox "BatchLaunch" ' Je passe aussi par le Timer au lancement pour laisser le te
   mps à l'event WinMain.Open de se finir, de mettre DrapMenu à True et d'EnabledMe
   nuBoutons
   ' J'ai réglé TimerBatch à 200, ça doit être largement suffisant pour que WinMain.Ope
   n exécute ses quelques lignes

   If (not(App.SyncEnCours or App.LogSEnCours)) and DrapMenu Then ' Si une synchro
   était en cours, ou si une fenêtre était ouverte (WinInfos), on saute purement et simp
   lement celle-ci
      If WinMain.BevButtAffBatch.Value Then
         ' If Me.Mode = Timer.ModeMultiple Then ' Je ferme car exécution plus rapi
         de et personne ne regarde puisqu'en auto, sauf au lancement de SyncTwoF
         olders
         ' Je ne fais plus comme ci-dessus car maintenant j'ai une CheckBox pour o
         uvrir ou non la fenêtre journal au lancement
         If WinMain.BevButtAffLog.Value and (not App.OuvLogLch) Then WinLog.Clos
         e ' Voir BevButtAffLog si on modifie des trucs ici lors de la fermeture
         ' On ferme WinLog si était affiché et si on ne l'ouvre pas au lancement (si é
         tait ouvert quand on a quitté)
         ActButtBatchSync
      Else
         MsgBox "Error ! Please contact the authors." + EndOfLine + "TimerBatch, B
         evButtAffBatch should be True !"
      End If
```

```
  End If ' Je fais tout le merdier qui suit même si test ci-dessus pas bon, des fois que le gars ait ouvert une fenêtre (DrapMenu à False) dès le lancement de SyncTwoFolders ou qu'il ait
  '  lancé une synchro manuellement aussi sec afin que tout soit paramétré comme il faut de toute manière

  ' MsgBox "TimerBatch" + EndOfLine + "TimerBatch.Mode = " + str(Me.Mode) + EndOfLine + "TimerBatch.Period = " + str(Me.Period) + EndOfLine + "App.BatchTempo = " + str(App.BatchTempo)
  ' Si était à 1 il est remis à 0 dès le début de cet event, sinon reste à 2
  If Me.Mode = Timer.ModeOff Then ' On vient de lancer SyncTwoFolders, sinon c'est qu'il était en Timer.ModeMultiple = 2 et que la période était déjà bonne
     Me.Period = App.BatchTempo * NbMsDsMin ' Car 1 période = 1 millième de sec, et EditFieldTimer est en minute
     Me.Mode = Timer.ModeMultiple ' 2
  Else
     If Me.Period < NbMsDsMin Then
        MsgBox "Error ! Please contact the authors." + EndOfLine + "TimerBatch, TimerBatch.Period = " + Format(Me.Period, F_MBillionSsEsp) + " < " + Format(NbMsDsMin, F_MBillionSsEsp) + " ms = 1 mn !"
        Me.Period = NbMsDsMin ' Sinon, si valeur toute petite, c'est très chiant car se lance en continu
        App.BatchTempo = NbMsDsMin
        EdFieldTimer.Text = Format(NbMsDsMin, F_MBillionSsEsp) ' Minutes
     End If
  End If

  If not((Me.Period = Val(EdFieldTimer.Text) * NbMsDsMin) and (Val(EdFieldTimer.Text) = App.BatchTempo)) Then
     MsgBox "Error ! Please contact the authors." + EndOfLine + "TimerBatch, Timer Batch.Period should be EditFieldTimer and BatchTempo !"
  End If

  ' Note : Si était en mode 1 revient en mode 0 automatiquement (dès le début de cette action), mais si par contre était en mode 2 reste en 2

End Sub
```

WinBatch Control EdFieldTimer:

```
Sub TextChange()
```

```
Dim TampNbre as Int32 ' Des fois qu'un nombre négatif soit entré mais on ne doit p
as pouvoir car mask
Dim TampText as String ' Le mask autorise 6 caractères pour automatiquement écri
re 65535 si on entre un plus grand nombre
Const MinVal = 1 ' Exactement la même chose que EdFieldMargeTps et EdFieldTime
OutCopy sauf pour valeur mini et maxi
Const MaxVal = 34560 ' Timer.Period as Int32 donc max = 2 147 483 647 / 60000
 = 35791,39412  J'arrondi à 24 j x 24 h x 60 mn = 34560 mn

If InitWinBatch Then
   ' MsgBox "Début" + EndOfLine + "TimerBatch.Mode = " + str(TimerBatch.Mode)
    + EndOfLine + "App.BatchTempo = " + str(App.BatchTempo) + " x " + str(NbM
   sDsMin) + " = " + EndOfLine + "TimerBatch.Period = " + str(TimerBatch.Period)
   TampText = Me.Text
   While (Left(TampText, 1) = "0") and (Len(TampText) > 1)
      TampText = Mid(TampText, 2)
   Wend

   TampNbre = Val(TampText)
   ' MsgBox "TampNbre = " + str(TampNbre)
   If TampNbre < MinVal Then
      Beep ' Pour dire que truc pas normal, fais chier de faire une DisplayDialog
      encore !
      App.BatchTempo = MinVal ' Valeur mini
      Me.Text = Format(App.BatchTempo, F_MBillionSsEsp)
   ElseIf TampNbre > MaxVal Then ' Pas même valeur que Mask (999999) car UInt
   16 0 à 65535
      Beep ' Pour dire que truc pas normal, fais chier de faire une DisplayDialog
      encore !
      App.BatchTempo = MaxVal ' Val maxi
      Me.Text = Format(App.BatchTempo, F_MBillionSsEsp)
   ElseIf not(Me.Text = TampText) Then
      ' If App.Sfx Then Bop.Play
      Me.Text = TampText
      App.BatchTempo = TampNbre ' = Val(TampText)
   Else ' L'entrée est bonne
      App.BatchTempo = TampNbre ' = Val(TampText)
   End If
   #If DebugBuild Then
      If not(Format(App.BatchTempo, F_MBillionSsEsp) = Me.Text) Then MsgBox "
      Problème Tom, EdFieldTimer.TextChange, TempoBatch = " + Format(App.
      BatchTempo, F_MBillionSsEsp) + " ≠ EdFieldTimer = " + Me.Text
   #EndIf
```

```
    If not(TimerBatch.Mode = Timer.ModeOff) Then ' Car le Reset ci-dessous le rem
    et à 1 si était à 0 (Resets the Timer and restarts it). Par contre il reste à sa valeu
    r si était à 1 ou à 2
      TimerBatch.Reset ' Je reset car si j'étais par exemple à 10 minutes et que je
       repasse à 1, le Timer se lance immédiatement
    End If
    TimerBatch.Period = App.BatchTempo * NbMsDsMin ' Car 1 période = 1 milliè
    me de sec, et EditFieldTimer est en minute
    ' MsgBox "Fin" + EndOfLine + "TimerBatch.Mode = " + str(TimerBatch.Mode) +
    EndOfLine + "App.BatchTempo = " + str(App.BatchTempo) + " x " + str(NbMsD
    sMin) + " = " + EndOfLine + "TimerBatch.Period = " + str(TimerBatch.Period)

  End If '  InitWinBatch

End Sub

End Class
```

Class WinRegl

Inherits Window

WinRegl.CancelClose:

```
Function CancelClose(appQuitting as Boolean) As Boolean

  Return ((not DrapMenu) and appQuitting) ' Si on retourne Vrai la fenêtre ne se ferme
  ra pas
  ' Voir  WinMain.CancelClose  Note : On n'utilise pas  App.DragDropOuv  car cette fe
  nêtre Log pas ouverte si DragDropOuv
  ' Si on quitte par le Dock on annulerait la fermeture de WinMain mais pas celle-ci q
  ui se fermerait
  '  donc on perdrait les coordonnées de la fenêtre

End Function
```

WinRegl.KeyDown:

```
Function KeyDown(Key As String) As Boolean

  ' Même chose dans chaque  event KeyDown  des autres fenêtres et que dans ActBut
  tSync
```

```
  If App.SyncEnCours or App.LogSEnCours Then
    ' If (Keyboard.AsynckeyDown(&h35) or (Key = Chr(27))) and (not WinMain.Arret
    Urg) Then ' Touche Escape , pour pas faire 2 fois Zoum (procédure Synchro)
    ' Touche  Escape dans les 2 cas, je faisais les 2 tests !
    If (Key = Chr(27)) and (not WinMain.ArretUrg) Then ' Touche Escape , pour pas f
    aire 2 fois Zoum (procédure Synchro)
      WinMain.ArretUrg = True
      If App.Sfx Then
        Zoum.Play ' Pour signifier prise en compte  ArretUrg     On ne l'entend
        s pas si coup de frein juste après, mais des fois
        While Zoum.IsPlaying '                                   que ce soit plu
        s long pour stopper tout
        Wend
      End If
      ' NON, NE PAS faire boucle ci-dessous, sinon cet event (clic sur Bouton Syn
      chro/Arrêter) ne se fait QU'à la fin de la sychro, donc redémarre et arrête a
      ussi sec
      ' While (App.SyncEnCours or App.LogSEnCours) ' Si ça marche !  (NE PAS fai
      re ça car garde la main et le Thread ne se fini pas)
      ' Wend '                                                Même sans DoEvents
      Pause(10, False) ' Pour éviter 2 appuie trop vif
      ' If App.Sfx Then Coup_de_freins.Play ' A été fait dans le Thread de Synchro
    End If
  End If

  Return (not(Key = Chr(9))) ' Car si Tab on passe d'un TextField à l'autre

End Function
```

WinRegl.Moved:

```
Sub Moved()

  If App.InitProg Then RePosFenetr(False, Keyboard.AsyncAltKey) ' Inutile car je ne fais
   plus le DoEvents  and DrapRePosFenetr

End Sub
```

WinRegl.Open:

```
Sub Open()
```

```
  DecPosCtrl(Self) ' Décalage des controls ( Label , TextField , Slider , etc. )
  #If TargetWin32 Then
    PushButtReglDef.TextSize = 12
  #EndIf

  AffMasqHelpTag("", "")

End Sub
```

WinRegl.AffMasqHelpTag:

```
Sub AffMasqHelpTag(CeCtrlNomId as String, CeCtrlHlpTg as String)

  Dim iNbre, Fin_iNbre as Int16 ' Envoyer  RectControl.Name  avec éventuellement son
    Index  (si tableau de RadioButton par exemple) pour ne mettre que lui à jour
  Dim TampNom as String '              Exemple : AffMasqHelpTag(StaticTextNbSpam.Na
  me, "TextDuHelpTag")
  Dim CeCtrl as Control '          ou si index 1 : AffMasqHelpTag(StaticTextNbSpam.Na
  me + "1", "TextDuHelpTag") ' ou  + str(index)
  Dim CeRectCtrl as RectControl
  Static MemRctCtrlNom(-1), MemHlpTgTxt(-1) as String

  Fin_iNbre = ControlCount - 1
  If UBound(MemHlpTgTxt) = -1 Then ' = UBound(MemRctCtrlNom)
    ' MsgBox "On mémorise HelpTag de " + str(Fin_iNbre + 1) + " controls." + End
    OfLine + "CeCtrlNomId = '" + CeCtrlNomId + "'"
    #If DebugBuild Then
      If not(CeCtrlNomId = "") Then MsgBox "Tu t'es gourré Tom, CeCtrlNomId de
      vrait être vide " !" + EndOfLine + "'" + CeCtrlNomId + "'"
    #EndIf
    For iNbre = 0 to Fin_iNbre
      CeCtrl = Self.Control(iNbre)
      ' MsgBox CeCtrl.Name + EndOfLine + str(CeCtrl.Index) + EndOfLine + "Rec
      tControl ? : " + CStr(CeCtrl isa RectControl)
      If CeCtrl isa RectControl Then
        CeRectCtrl = RectControl(CeCtrl) ' Et NON RectControl(iNbre)
        TampNom = CeCtrl.Name
        #If DebugBuild Then
          If (not(CeRectCtrl.Index = CeCtrl.Index) and (CeRectCtrl.Name = T
          ampNom)) Then MsgBox "Bizarre Tom, RectCtrl.Index ≠ Ctrl.Index
           :" + EndOfLine + str(CeRectCtrl.Index) + " ≠ " + str(CeCtrl.Index)
          + EndOfLine + "RectCtrl.Name ≠ Ctrl.Name :" + EndOfLine + str(C
          eRectCtrl.Name) + " ≠ " + str(TampNom)
```

```
        #EndIf
        If CeCtrl.Index > -1 Then TampNom = TampNom + str(CeCtrl.Index) '
        ATTENTION Int32, si pas un tableau de control alors = -2147483648
        ' MsgBox " id = " + str(CeCtrl.Index) + " = " + str(CeRectCtrl.Index) +
        EndOfLine + TampNom
        MemRctCtrlNom.Append TampNom
        MemHlpTgTxt.Append CeRectCtrl.HelpTag ' On mémorise qu'il y en ait
         ou non
        If not App.AffHelpTag Then CeRectCtrl.HelpTag = ""
        ' Si à Vrai alors l'helptag reste à sa valeur
      End If ' RectControl
    Next iNbre

Else ' On les a déjà mémorisés
    ' MsgBox "Nbre de Controls : " + str(Fin_iNbre + 1) + EndOfLine + "Nbre de Rec
    tControls : " + str(UBound(MemHlpTgTxt) + 1) + EndOfLine + "CeCtrlNomId = '
    " + CeCtrlNomId + "'"
    If CeCtrlNomId = "" Then ' On applique à tous
      For iNbre = 0 to Fin_iNbre ' Pas forcément = à  UBound(MemHlpTgTxt)  car
       tous ne sont pas RectControl, mais QUE Control
        CeCtrl = Self.Control(iNbre)
        ' MsgBox CeCtrl.Name + EndOfLine + str(CeCtrl.Index) + EndOfLine +
        "RectControl ? : " + CStr(CeCtrl isa RectControl)
        If CeCtrl isa RectControl Then
          CeRectCtrl = RectControl(CeCtrl) ' Et NON RectControl(iNbre)
          ' MsgBox str(CeRectCtrl.Index) + " = " + EndOfLine + str(CeCtrl.In
          dex) + EndOfLine + CeRectCtrl.Name + EndOfLine + CeRectCtrl.H
          elpTag
          If App.AffHelpTag Then
            CeRectCtrl.HelpTag = MemHlpTgTxt(iNbre) ' S'il n'y en avait p
            as ça reste à ""
          Else
            CeRectCtrl.HelpTag = ""
          End If
        End If ' RectControl
      Next iNbre

    Else ' On applique qu'au control indiqué
      iNbre = MemRctCtrlNom.IndexOf(CeCtrlNomId)
      If iNbre = -1 Then
        MsgBox "Error ! Please contact the authors, AffMasqHelpTag :" + EndO
        fLine + CeCtrlNomId + EndOfLine + "iNbre = -1"
      Else
        CeCtrl = Self.Control(iNbre)
```

```
        ' MsgBox CeCtrlNomId + EndOfLine + "iNbre = " + str(iNbre)
        ' Forcément : If CeCtrl isa RectControl Then
        CeRectCtrl = RectControl(CeCtrl) ' Et NON RectControl(iNbre)
        MemHlpTgTxt(iNbre) = CeCtrlHlpTg ' On met à jour la mémoire du hel
        ptag
        If App.AffHelpTag Then
          CeRectCtrl.HelpTag = CeCtrlHlpTg ' On met à jour le helptag lui-
          même
        Else
          If not(CeRectCtrl.HelpTag = "") Then MsgBox  "Error ! Please contac
          t the authors, AffMasqHelpTag :" + EndOfLine + CeCtrlNomId + "'
          helptag should be empty " :" + EndOfLine + """ + CeRectCtrl.HelpT
          ag + """
        End If
      End If

    End If ' On applique à tous ou au contrôle indiqué  CeCtrlNomId

  End If ' UBound(MemHlpTgTxt) = -1

End Sub
```

WinRegl.RePosFenetr:

```
Sub RePosFenetr(DrapMoveWM as Boolean, DrapDefPos as Boolean)

  Dim FenetrL, FenetrT as Int16 ' En Cocoa les Events se font en Live et ça merde si j'a
  ffecte directement  Self.Left  et  Self.Top
  Dim DrapMemDec, DrapPosFenetr as Boolean ' Pas vraiment utile ces Flags mais j'ai
  me autant ne pas refaire des calculs inutiles, de plus il y a le pb ci-dessus
  Static DecalageL, DecalageT, MemWMw, MemWMh as Int16 ' Static garde en mémoir
  e la valeur de la variable d'un appel de à l'autre de la Method

  ' Inutile car je ne fais plus le DoEvents : DrapRePosFenetr = False ' Voir note plus ba
  s au App.DoEvents
  If DrapMoveWM Then ' On a déplacé ou redimensionné WinMain, on vient donc de W
  inMain.Moved ou WinMain.Resized
    DrapMemDec = False ' On pourrait remémoriser  DecalageL  et  DecalageT  car
    on retrouverait le même nombre mais je n'aime pas
    DrapPosFenetr = True
    FenetrL = WinMain.Left + DecalageL
    FenetrT = WinMain.Top + DecalageT
    If Self.Left > (WinMain.Left + (MemWMw / 2)) Then FenetrL = FenetrL + (WinMai
    n.Width - MemWMw)
```

```
    If Self.Top > (WinMain.Top + (MemWMh / 2)) Then FenetrT = FenetrT + (WinMai
    n.Height - MemWMh)

Else ' On vient de Self.Moved (ou des Prefs)
    DrapMemDec = True
    If DrapDefPos Then
        DrapPosFenetr = True
        FenetrL = WinMain.Left ' Seules ces 2 lignes sont à personnaliser suivant la
         position par défaut de la fenêtre
        FenetrT = WinMain.Top + WinMain.Height + 32 ' Seules ces 2 lignes sont à
         personnaliser suivant la position par défaut de la fenêtre
    Else
        DrapPosFenetr = False
        FenetrL = Self.Left
        FenetrT = Self.Top
    End If

End If

If FenetrL < 0 Then
    DrapMemDec = True
    DrapPosFenetr = True
    FenetrL = 0
ElseIf (FenetrL + Self.Width) > Screen(0).Width Then
    DrapMemDec = True
    DrapPosFenetr = True
    FenetrL = Screen(0).Width - Self.Width
End If
If FenetrT < 38 Then
    DrapMemDec = True
    DrapPosFenetr = True
    FenetrT = 38
ElseIf (FenetrT + Self.Height) > Screen(0).Height Then
    DrapMemDec = True
    DrapPosFenetr = True
    FenetrT = Screen(0).Height - Self.Height
End If

' Non, pas utile : App.DoEvents ' Comme ça l'event Moved de Self se fait alors que
DrapRePosFenetr  est encore False
' Inutile car je ne fais plus le DoEvents : DrapRePosFenetr = True
' MsgBox "On vient de WinMain : " + Cstr(DrapMoveWM) + EndOfLine + "WinMain : "
 + str(WinMain.Left) + ", " + str(WinMain.Top) + EndOfLine + "Self : " + str(FenetrL)
+ ", " + str(FenetrT) + EndOfLine + "Mem décalage : " + Cstr(DrapMemDec) ' + "  "
+ Cstr(DrapRePosFenetr) ' Fin
```

```
  If DrapMemDec Then
    DecalageL = FenetrL - WinMain.Left
    DecalageT = FenetrT - WinMain.Top
    MemWMw = WinMain.Width
    MemWMh = WinMain.Height
  End If
  If DrapPosFenetr Then ' On va immédiatement revenir ici via l'event Self.Moved donc
   on va re-mémoriser  DecalageL  et  DecalageT
    Self.Left = FenetrL
    Self.Top = FenetrT
  End If

End Sub
```

WinRegl.VrifReglDef:

```
Function VrifReglDef(DrapCdeSh as Boolean) As Boolean

  Dim TampDrap as Boolean

  TampDrap = (not(WinMain.MemMargeTps = App.DefMargeTps)) or (CheckBoxDossP
  ack.Value Xor App.DefDossPackFich) or (CheckBoxDossExt.Value Xor App.DefDossE
  xtFich) or (CheckBoxUseShell.Value Xor App.DefUseShell) or (CheckBoxCopyRBerr.Va
  lue Xor App.DefCopyRBerr)
  If DrapCdeSh Then TampDrap = TampDrap or (not(TextFieldCpShell.Text = App.Def
  CdeCpShell))

  Return TampDrap ' Retourne False si les réglages actuels sont ceux par défaut

End Function
```

WinRegl Control EdFieldMargeTps:

```
Sub TextChange()

  Dim TampNbre as Int32 ' Des fois qu'un nombre négatif soit entré mais on ne doit p
  as pouvoir car mask
  Dim TampText as String ' Le mask autorise 6 caractères pour automatiquement écri
  re 65535 si on entre un plus grand nombre
  Const MinVal = 0 ' Exactement la même chose que EdFieldTimer et EdFieldTimeOut
  Copy sauf pour valeur mini et maxi
  Const MaxVal = 65535 ' Max UInt16
```

```
  If App.initProg Then
    TampText = Me.Text
    While (Left(TampText, 1) = "0") and (Len(TampText) > 1)
      TampText = Mid(TampText, 2)
    Wend
    If TampText = "" Then TampText = "0" ' On a entré une lettre

    TampNbre = Val(TampText)
    ' MsgBox "TampNbre = " + str(TampNbre)
    If TampNbre < MinVal Then ' Impossible car mask mais ...
      Beep ' Pour dire que truc pas normal, fais chier de faire une DisplayDialog
      encore !
      Me.Text = Format(WinMain.MemMargeTps, F_MBillionSsEsp) ' Valeur d'avan
      t
    ElseIf TampNbre > MaxVal Then ' Pas même valeur que Mask (999999) car UInt
    16 0 à 65535
      Beep ' Pour dire que truc pas normal, fais chier de faire une DisplayDialog
      encore !
      Me.Text = Format(MaxVal, F_MBillionSsEsp) ' Val maxi
      WinMain.MemMargeTps = MaxVal
    ElseIf not(Me.Text = TampText) Then
      ' If App.Sfx Then Bop.Play
      Me.Text = TampText
      WinMain.MemMargeTps = TampNbre ' = Val(TampText)
    Else ' L'entrée est bonne
      WinMain.MemMargeTps = TampNbre ' = Val(TampText)
    End If
    #If DebugBuild Then
      If not(Format(WinMain.MemMargeTps, F_MBillionSsEsp) = Me.Text) Then Ms
      gBox "Problème Tom, EdFieldMargeTps.TextChange, MemMargeTps = " +
      Format(WinMain.MemMargeTps, F_MBillionSsEsp) + " ≠ EdFieldMargeTps =
       " + Me.Text
    #EndIf

    WinMain.ChangeConfig(-1)

  End If '  initProg

End Sub
```

WinRegl Control CheckBoxDossPack:

```
Sub Action()

   If App.initProg Then
      WinMain.ChangeConfig(-1)
   End If '  initProg

End Sub
```

WinRegl Control CheckBoxDossExt:

```
Sub Action()

   If App.initProg Then
      WinMain.ChangeConfig(-1)
   End If '  initProg

End Sub
```

WinRegl Control CheckBoxUseShell:

```
Sub Action()

   If App.initProg Then
      WinMain.ChangeConfig(-1)
   End If '  initProg

End Sub
```

WinRegl Control PushButtReglDef:

```
Sub Action()

   Dim TampDrap as Boolean

   If App.initProg Then
      App.initProg = False ' Sinon chaque modif ci-dessous va appeller ChangeConfi
      g(-1)
      TampDrap = VrifReglDef(True) ' On vérifie si  WinRegl.TextFieldCpShell.Text =
      App.DefCdeCpShell
      ' Voir  PrefsCharger  et  VrifReglDef
```

```
    ' Si  TampDrap  est True ça ne sert à rien de tout remettre par défaut puisque ç
    a y est, mais je fais quand même des fois que j'ai fais une connerie
    WinMain.MemMargeTps = App.DefMargeTps
    EdFieldMargeTps.Text = Format(WinMain.MemMargeTps, F_MBillionSsEsp)
    ' EdFieldMargeTps.SelStart = Len(EdFieldMargeTps.Text) ' Fait dans ChangeCon
    fig

    ' Voir TraitComDoss
    CheckBoxDossPack.Value = App.DefDossPackFich
    CheckBoxDossExt.Value = App.DefDossExtFich

    CheckBoxUseShell.Value = App.DefUseShell
    TextFieldCpShell.Text = App.DefCdeCpShell

    CheckBoxCopyRBerr.Value = App.DefCopyRBerr

    App.initProg = True ' Pour se retrouver en situation d'origine, mais ChangeConf
    ig le remet à False puis à nouveau à True
    If TampDrap Then WinMain.ChangeConfig(-1)
  End If '  initProg

End Sub
```

WinRegl Control CheckBoxCopyRBerr:

```
Sub Action()

  If App.initProg Then
    WinMain.ChangeConfig(-1)
  End If '  initProg

End Sub

End Class
```

Module Localisation

Const AddConfig_HT = "‡Sauvegarder les réglages actuels"

Const Btn_AffLog_HT = "‡Afficher/Masquer journal"

Const Btn_AffRegl_HT = "‡Réglages supplémentaires. Attention, soyez sûr de ce que vo
us faîtes !"

Const Btn_Annuler = "‡Annuler"

Const Btn_Arreter = "‡Arrêter"

Const Btn_BatchSync = "‡Batch Sync"

Const Btn_Check_HT = "‡Cocher toutes les lignes sélectionnées (⌥ clic pour tout décoc
her), s'applique à toute la liste si aucune ligne n'est sélectionnée"

Const Btn_ChoisirDoss = "‡Choisir dossier…"

Const Btn_Cont = "‡Continuer"

Const Btn_DejDon = "‡J'ai déjà donné !"

Const Btn_Enreg = "‡Enregistrer"

Const Btn_EnregSs = "‡Enregistrer sous…"

Const Btn_FaireDon = "‡Faire un don…"

Const Btn_FAQ = "‡FAQ"

Const Btn_Fermer = "‡Fermer"

Const Btn_Historic = "‡Historique"

Const Btn_LireAbout = "‡Lire À propos"

Const Btn_LogSync = "‡Sync journal"

Const Btn_Ok = "‡Ok"

Const Btn_Recommencer = "‡Recommencer"

Const Btn_ReglDef = "‡Réglages par défaut"

Const Btn_Remplacer = "‡Remplacer"

Const Btn_Simu = "‡Simuler"

Const Btn_SiteWeb = "‡Site Web"

Const Btn_Supp = "‡Supprimer"

Const Btn_Sync = "‡Synchroniser"

Const ChBoxAffHlpTg = "‡Afficher les bulles d'aide"

Const ChBoxAlertReglDef = "‡Alerte si réglages supplémentaires non par défaut"

Const ChBoxAutorisVol = "‡Autoriser la synchronisation des volumes"

Const ChBoxAutoUpdt = "‡Vérifier automatiquement les mises à jour"

Const ChBoxBatchAuto = "‡Lancer toutes les"

Const ChBoxBatchA_HT = "‡Cette fenêtre tiroir doit être ouverte, les réglages cochés s’e xécuteront au lancement de SyncTwoFolders puis toutes les … minutes"

Const ChBoxCopyRBerr = "‡Tenter de copier avec RealBasic si erreur"

Const ChBoxCopyRBerr_HT = "‡AppleScript (Finder) est utilisé en premier. Attention ave c les autorisations des éléments !"

Const ChBoxCoulListLog = "‡Colorier les symboles dans le journal"

Const ChBoxDelTrash = "‡Déplacer les éléments dans la corbeille"

Const ChBoxDelTrash_HT = "‡Plutôt que de les effacer directement"

Const ChBoxDossExt = "‡Avec extension"

Const ChBoxDossExt_HT = "‡Attention aux dossiers dont le nom contient un point '.'"

Const ChBoxDossPack = "‡Paquet"

Const ChBoxGererDate = "‡Gérer dates"

Const ChBoxIgnorElt = "‡Noms ou extensions à ignorer :"

Const ChBoxLierNav = "‡Lier la navigation dans les dossiers"

Const ChBoxOuvLogApS = "‡Ouvrir la fenêtre journal après une synchronisation"

Const ChBoxOuvLogLch = "‡Ouvrir la fenêtre journal au lancement"

Const ChBoxOuvLogLch_HT = "‡Si elle était ouverte en quittant"

Const ChBoxPasIgnor = "‡Noms des éléments spéciaux à gérer :"

Const ChBoxPreNiv = "‡/"

Const ChBoxRelPathLog = "‡Utiliser des chemins relatifs dans le journal"

Const ChBoxRemplRecent = "‡Remplacer plus récent"

Const ChBoxSfx = "‡Effets sonores"

Const ChBoxSimu = "‡Simulation"

Const ChBoxSyncIc = "‡Synchroniser les icônes"

Const ChBoxTrash = "‡Effacer éléments commençant par :"

Const ChBoxUseShell = "‡Utiliser cde shell pour copier :"

Const ChBoxUseShell_HT = "‡Utiliser cette commande shell plutôt que le Finder pour co
pier"

Const DelConfig_HT = "‡Effacer ces réglages du PopupMenu"

Const EditFieldLog_HT = "‡(⌘ et ⌥ clic pour remettre les largeurs de colonnes par défa
ut)"

Const EditFieldPath_Note = "‡^ clic un chemin de la liste pour le copier dans le presse-papiers (+ ⌥ clic pour ShellPath), ⌥ clic pour afficher l'élément dans le Finder"

Const GererDate_HT = "‡Si des fichiers ont des dates de modification différentes, le plus récent remplacera le plus ancien. Si cette option n'est pas cochée, seuls les fichiers non présents dans l'autre dossier seront copiés"

Const GrBoxCible = "‡Cible"

Const GrBoxSource = "‡Source"

Const GrBox_HT = "‡Glisser déposer un dossier ici (⌥ clic sur le bouton pour révéler ce dossier dans le Finder, ⌘ et ⌥ clic pour révéler le dossier utilisé pour la corbeille)"

Const IgnorElt_HT = "‡Lâchez des éléments ici pour les ajouter automatiquement (séparés par “:”), ⌥ pour ajouter l'extension"

Const Infos_StTextCopRight = "‡©2006, Th. Robisson et Ph. Galmel, tous droits réservés."

Const Infos_Texte = "‡Ce programme synchronise deux dossiers. Les fichiers les plus anciens seront remplacés par les plus récents, dans certains cas, des fichiers seront effacés.

Ut...

Const Infos_Text_Don = "‡Un don serait apprécié afin de m'aider à continuer le développement de ce programme, car cela a un coût. Outre le temps passé, je dois payer mes licences.

Const InvDoss_HT = "‡Inverse les dossiers Source et Cible"

Const ListLog_TitreCol = "‡✓ No • Ext Source <—> Cible"

Const Menu_Aide = "‡Aide"

Const Menu_AideAide = "‡Aide SyncTwoFolders"

Const Menu_AppleApropos = "‡À propos de SyncTwoFolders"

Const Menu_AppleVMaJ = "‡Vérifier les mises à jour…"

Const Menu_Edit = "‡Édition"

Const Menu_EditClear = "‡Effacer"

```
Const Menu_EditCopy = "‡Copier"

Const Menu_EditCut = "‡Couper"

Const Menu_EditPaste = "‡Coller"

Const Menu_EditSelectAll = "‡Tout sélectionner"

Const Menu_EditUndo = "‡Annuler"

Const Menu_Fichier = "‡Fichier"

Const Menu_FichierChDossC = "‡Choisir dossier Cible..."

Const Menu_FichierChDossS = "‡Choisir dossier Source..."

Const Menu_FichierCloseWin = "‡Fermer"

Const Menu_Preferences = "‡Préférences..."

Const Menu_Window = "‡Fenêtre"

Const PasIgnor_HT = "‡Tous les éléments invisibles sont ignorés (commençant par un p
oint “.” ou non), mais tous les éléments spéciaux entrés ci-dessous seront gérés (sépare
z les par par “:”)"

Const PopupBarreOutils_HT = "‡Barre d'outils fenêtre:
                                    La ...
Const PopupTtList_HT = "‡Taille du texte"

Const PremNiv_HT = "‡Ignorer ces éléments dans le premier dossier uniquement, les él
éments portant ces noms contenus dans les sous dossiers seront traités. Ceci n'a aucun
 effet sur les extensions, les extensions seront ignorées dans tous les sous dossiers"

Const PtitRond_HT = "‡Indique si les noms des dossiers Source et Cible sont identiques
"


Const RdButtAmovTrash0 = "‡Ce dossier :"

Const RdButtAmovTrash1 = "‡Corbeille utilisateur"

Const RdButtAmovTrash2 = "‡Effacer directement les éléments"

Const RdButtSyncMode0 = "‡Réciproque"
```

Const RdButtSyncMode1 = "‡Source complète Cible"

Const RdButtSyncMode2 = "‡Source remplace Cible"

Const RemplRecent_HT = "‡Si un fichier est plus récent dans le dossier Cible que dans l e dossier Source il sera tout de même remplacé"

Const Simul_HT = "‡Aucun élément ne sera copié ou effacé, toutefois les icônes seront placées si l'option a été choisie"

Const StaticTextCasDiskAmov = "‡Pour les disques amovibles, utiliser :"

Const StaticTextDossPackExt = "‡Traiter ces dossiers comme des fichiers :"

Const StaticTextMargeTps = "‡Deux dates sont considérées comme identiques si leur di fférence est inférieure à … secondes :"

Const StaticTextMn = "‡mn"

Const StaticTextTimeOutCopy = "‡Copie erreur de TimeOut en secondes :"

Const SymbAncien_HT = "‡: Fichier Source plus ancien que fichier Cible (en mode non R éciproque)"

Const SymbCopy_HT = "‡: Élément copié (n'existait pas dans l'autre dossier)"

Const SymbEff_HT = "‡: Élément effacé"

Const SymbOpErrDiv_HT = "‡: Erreur pendant l'opération (disque plein, fichier verrouillé ou invisible, problème d'autorisation, etc.)"

Const SymbOpNon_HT = "‡: Opération non réalisée"

Const SymbOpOui_HT = "‡: Opération réalisée"

Const SymbRempl_HT = "‡: Fichier copié avec remplacement (existait dans l'autre dossi er)"

Const SyncBatch_HT = "‡Synchronise toutes les réglages cochés"

Const Synchro_HT = "‡Démarre ou Arrête la Synchronisation (touche Esc)"

Const SyncIc_HT = "‡Les icônes (position) du dossier Cible seront placées comme celles du dossier Source (même si en mode Simulation)"

Const SyncMode = "‡Mode de Synchronisation"

Const SyncMode0_HT = "‡Les dossiers Source et Cible se complètent réciproquement, s
i un fichier est présent dans les deux dossiers, le plus récent remplacera le plus ancien"

Const SyncMode1_HT = "‡Le dossier Source complète le dossier Cible, les fichiers du do
ssier Source non présents dans le dossier Cible seront copiés, les anciens fichiers du do
ssier Cible seront remplacés par ceux du dossier Source"

Const SyncMode2_HT = "‡Le dossier Source complète le dossier Cible, les fichiers du do
ssier Source non présents dans le dossier Cible seront copiés, les anciens fichiers du do
ssier Cible seront remplacés par ceux du dossier Source, les fichiers du dossier Cible no
n présents dans le dossier Source seront effacés"

Const TrashIfBegin_HT = "‡Les éléments préfixés avec ce texte seront effacés, leurs ho
mologues du deuxième dossier également (même s'ils ne sont pas préfixés avec ce text
e),
lâchez des ...
Const Txt_AdejaTextTrash = "‡Le fichier nommé “¿%” est déjà préfixé avec “¿%”.
mer préfix ... Suppri

Const Txt_AideNonDispo = "‡L'aide n'est pas disponible pour ¿%.
Ve ...
Const Txt_AlertAutorisVol = "‡Ceci est déconseillé."

Const Txt_Arret = "‡Non Terminé"

Const Txt_Attente = "‡..."

Const Txt_Attention = "‡*** Attention ! ***"

Const Txt_CarAccept = "‡Vous ne pouvez entrer que ¿% caractères maximums et seule
ment les caractères suivants : "

Const Txt_ChoisirUnDoss = "‡Choisir un dossier"

Const Txt_DefDossTrash_impo = "‡Impossible de définir le dossier corbeille."

Const Txt_DropQueDoss = "‡Vous ne pouvez glisser déposer qu'un dossier ici."

Const Txt_ErrActiverEff = "‡Vous n'avez pas activé l'effacement des éléments commenç
ant par ..."

```
Const Txt_Fin = "‡Terminé"

Const Txt_LastConfig = "‡Derniers réglages"

Const Txt_NomExiste = "‡Un fichier nommé “¿%” existe déjà."

Const Txt_PasDossTrash = "‡Vous ne pouvez pas synchroniser le dossier corbeille."

Const Txt_PasMemeDoss = "‡C'est deux fois le même dossier. Veuillez recommencer av
ec deux dossiers différents."

Const Txt_PasUnDsAutre = "‡Vous ne pouvez pas sélectionner deux dossiers avec l'un i
nclus dans l'autre."

Const Txt_PasVolume = "‡Vous ne pouvez pas sélectionner un volume."

Const Txt_ReglageExiste = "‡Des réglages nommés “¿%” existent déjà."

Const Txt_Select2Doss = "‡Vous devez d'abord sélectionner deux dossiers."

Const Txt_SelectDossier = "‡Sélectionnez le dossier de sauvegarde :"

Const Txt_SelectQueDoss = "‡Vous pouvez seulement sélectionner un dossier."

Const Txt_SuppReglage = "‡Êtes vous sûr de vouloir supprimer les réglages “¿%” ?"

Const Txt_SyncEnCours = "‡Vous ne pouvez pas quitter car une synchronisation est en
cours, souhaitez-vous l'arrêter ?"

Const Txt_Ver_a_jour = "‡Cette version est à jour."

Const Txt_Ver_MaJ_impo = "‡Impossible de vérifier les mises à jour pour le moment, ve
uillez vérifier votre connexion internet ou réessayer plus tard."

Const Txt_Ver_Nouv = "‡Une nouvelle mise à jour est disponible."

Const Txt_VoirFAQ_HT = "‡Voir FAQ"

Const Txt_VolMemeNom = "‡Plusieurs volumes ont le même nom, il est préférable d'évi
ter afin de ne pas risquer de les intervertir."

Const WinConfig_Title = "‡Nom de ces réglages"

Const WinLog_Title = "‡Journal"
```

```
Const WinRegl_Title = "‡Réglages supplémentaires."
```

Localisation.FracTexte:

```
Function FracTexte(TexteLocalise as String, Partie as Int16) As String

  ' \% J'utilisais ces caractères ci car ceux sont ceux utilisées pour les applications Car
  bon. Mais ça merde avec les Dynamic
  '  constantes alors j'utilise ce code : ¿%
  ' Cette procédure divise TexteLocalise pas ¿% et retourne la Partie indiquée

  Dim TableauTexte(-1) as String

  TableauTexte = Split(TexteLocalise, "¿%")

  ' MsgBox TexteLocalise + EndOfLine + EndOfLine + "###### se divise en  "+str(Ub
  ound(TableauTexte) + 1) + " parties."

  ' On ne fait pas les tests ci-dessous, justement on veut une erreur s'il manque ¿% d
  ans le texte localisé

  ' If (Partie > 0) and (Partie <= (Ubound(TableauTexte) + 1)) Then
  Return TableauTexte(Partie - 1)
  ' Else
  ' Return ""
  ' End If

  ' MsgBox """+ FracTexte("¿%",1)+""" ' A coller là où on veut pour tester cette Méthode

End Function
```

Localisation Note: RdButtNetTrashAv

```
RdButtNetTrashAv

Avant, en 0 j'avais, mais FolderItem.SharedTrashFolder est deprecated :
RdButtNetTrash0
en : Network Trash
fr : Corbeille réseau
es : Papelera de red
```

de : Papierkorb des Netzwerkvolumes
it : Cestino di rete

End Module

Module Fncts

```
Const CarTab = " "

Const F_MBillionAvEsp = "-###\ ###\ ###\ ###\ ###\ ##0"

Const F_MBillionAvEspV = "-###\ ###\ ###\ ###\ ###\ ##0.00"

Const F_MBillionSsEsp = "-################0"

Const F_MBillionSsEspV = "-################0.00"

Const LienSitePerso = "http://throb.pagesperso-orange.fr"

Const Lien_PayPage = "/site/PayPage.html?"

Const Lien_PrgRb = "/site/programmes/xojo/"

Const SepAbsPath = ":"
```

Fncts.AbsPath_f:

```
Function AbsPath_f(Extends CetElt as FolderItem) As String

  Dim TampText as String ' Renvoie le Path comme avant avec RealStudio (avec les ":" comme séparateur sur Mac)
  ' Car  CetElt.NativePath  remplace les "/" par ":" . A noter que  CetElt.ShellPath  n'est pas exactement la même chose que  CetElt.NativePath
  ' Cette fonction plante si  CetElt  est à Nil

  TampText = CetElt.Name
  If CetElt.Directory Then TampText = TampText + SepAbsPath ' Un path de dossier s e termine par ":", même celui d'un Package (à gérer dans le prog si besoin)
  CetElt = CetElt.Parent
```

```
  While not(CetElt = Nil)
    TampText = CetElt.Name + SepAbsPath + TampText
    CetElt = CetElt.Parent
  Wend

  Return TampText

End Function
```

Fncts.AppliAuPplan:

```
Sub AppliAuPplan()

  Dim CetteAppli as FolderItem

  #If TargetMacOS Then
    CetteAppli = App.ExecutableFile.Parent.Parent.Parent ' Car donne l'exécutable
    dans Contents:MacOS:
    If PasNil_Existe_Alias(CetteAppli, False) Then ' Des fois que ... , ce n'est pas un
    alias de toute façon
      ' MsgBox "CetteAppli : '" + CetteAppli.AbsPath_f + "'" + EndOfLine + "Cette
      Appli : '" + GetFolderItem("").Child(App.ExecutableFile.Name).AbsPath_f + "
      .app"
      ' #If DebugBuild Then ' Il y a .debug dans le nom et/ou le path de l'applicat
      ion
      ' Et il y a pas le .app
      TellThisApp(CetteAppli.ShellPath_fAS)
    Else
      MsgBox "Error !"+EndOfLine+"Please contact the authors."+EndOfLine+End
      OfLine+"AppliAuPplan, CetteAppli = Nil"
    End If

  #Else ' Win32 ou autre
    MsgBox "Error ! Please contact the authors." + EndOfLine + "AppliAuPplan is on
    ly for TargetMacOS !"

  #EndIf

  ' TellThisApp(GetFolderItem("").Child(App.ExecutableFile.Name).AbsPath_f + ".app")
  ' Et oui, le .app n'y est pas ?!!!
  ' Car comme on a cliqué Arrêter du la fenêtre Copier du Finder ce dernier est passé
  au 1er plan, donc on rappelle l'application au 1er plan
  '  App.ExecutableFile.AbsPath_f  donne le path de l'app à l'intérieur de Contents
```

```
  ' #Else
  ' #If DebugBuild Then
  ' MsgBox "Problème Tom, tu devrais être en TargetMacOS !"
  ' #EndIf
  ' #EndIf

End Sub
```

Fncts.BeepNbT:

```
Sub BeepNbT(NbreDeBeep as Int16)

  Dim iNbre as Int16 ' Même Int que pour NbreDeBeep ci-dessus

  If NbreDeBeep > 0 Then Beep
  For iNbre = 2 to NbreDeBeep
    Pause(10, False)
    ' MsgBox "On va faire Beep n° " + str(iNbre)
    Beep
  Next iNbre

End Sub
```

Fncts.CountVis_fSH:

```
Function CountVis_fSH(Extends CetElt as FolderItem) As UInt32

  Dim CdeShell as New Shell ' Note : J'utilise  Extends  pour pouvoir écrire  MonElt.Cou
  ntVis_fSH  mais ça fera une Nil Objection si à Nil
  Dim NbEltsVis as UInt32 ' Même que Type de retour
  ' Retourne le nombre d'éléments visibles (ne commençant pas par un '.') contenu da
  ns un dossier, le dossier lui même est compté (un dossier vide compte 1)

  #If DebugBuild Then '  CetElt  doit être un dossier
    If not CetElt.Directory Then MsgBox "Pb Tom, CountVis_fSH ne ne peut s'effectu
    er que sur un dossier." + EndOfLine + CetElt.AbsPath_f
  #EndIf

  ' NE PAS utiliser  """ + MonFolderItem.ShellPath_fAS + """
  ' NON : CdeShell.Execute "/usr/bin/find '" + CetElt.ShellPath_fAS + "/' \! -name '.*' |
   /usr/bin/wc -l;" ' Compte tous les éléments ne commençant pas par '.'
```

```
  ' NON : CdeShell.Execute "/usr/bin/find '" + CetElt.ShellPath_fAS + "/' | /usr/bin/wc
   -l;" ' Compte tous les éléments
  ' On n'utilise pas le Quotedform avec ' ' car s'il y a un ' dans le nom ça merde
  CdeShell.Execute "/usr/bin/find " + CetElt.ShellPath + "/ \! -name '.*' | /usr/bin/wc
   -l;" ' Compte tous les éléments ne commençant pas par '.'
  ' CdeShell.Execute "/usr/bin/find '" + CetElt.ShellPath + "/ | /usr/bin/wc -l;" ' Comp
  te tous les éléments
  ' CetElt.Count donne tous les éléments (invisibles ou non) contenus dans le dossier,
   mais pas les sous-dossiers. Un dossier vide retourne 0
  If CdeShell.ErrorCode = 0 Then
    NbEltsVis = Val(CdeShell.Result)
    ' MsgBox CdeShell.Result
  Else ' Dossier sur lequel on n'a pas les droits ou etc.
    NbEltsVis = 0 ' Car ça compte le dossier lui même, donc s'il est vide ça retourn
    e 1
    ' MsgBox "Error code : " + Str(CdeShell.ErrorCode)
  End If

  Return NbEltsVis

End Function
```

Fncts.DecPosCtrl:

```
Sub DecPosCtrl(CetteFenetr as Window)

  Dim iNbre, Fin_iNbre as Int16 ' Ces RectControls n'ont pas la même position suivant
   compil Carbon ou Cocoa
  Dim CeCtrl as Control
  Dim CeRectCtrl as RectControl
  #If TargetCocoa Then
    Const DecLabel = 1
    Const DecTextField = -1
    ' Const DecTextArea = 0
    Const DecSlider = 2
  #ElseIf TargetCarbon Then ' Voir TypeCompil, je fais autrement pour voir plateform
  e
    Const DecLabel = 0
    Const DecTextField = 0
    ' Const DecTextArea = 0
    Const DecSlider = 2
  #ElseIf TargetWin32 Then
    Const DecLabel = 0
```

```
    Const DecTextField = 0
    ' Const DecTextArea = 0
    Const DecSlider = 2
  #EndIf

  Fin_iNbre = CetteFenetr.ControlCount - 1
  For iNbre = 0 to Fin_iNbre
    CeCtrl = CetteFenetr.Control(iNbre)
    ' MsgBox CeCtrl.Name + EndOfLine + str(CeCtrl.Index) + EndOfLine + "RectCo
    ntrol ? : " + CStr(CeCtrl isa RectControl)
    If CeCtrl isa RectControl Then
      If CeCtrl isa Label Then
        CeRectCtrl = RectControl(CeCtrl) ' Et NON RectControl(iNbre)
        CeRectCtrl.Top = CeRectCtrl.Top + DecLabel
      ElseIf CeCtrl isa TextField Then ' If CeCtrl isa TextArea Then
        CeRectCtrl = RectControl(CeCtrl) ' Et NON RectControl(iNbre)
        CeRectCtrl.Top = CeRectCtrl.Top + DecTextField
        ' ElseIf CeCtrl isa TextArea Then
        '   CeRectCtrl = RectControl(CeCtrl) ' Et NON RectControl(iNbre)
        '   CeRectCtrl.Top = CeRectCtrl.Top + DecTextArea
      ElseIf CeCtrl isa Slider Then
        CeRectCtrl = RectControl(CeCtrl) ' Et NON RectControl(iNbre)
        CeRectCtrl.Top = CeRectCtrl.Top + DecSlider
      End If ' Label , Slider , TextField , TextArea
    End If ' RectControl
  Next iNbre

End Sub
```

Fncts.DossPasPack_f:

```
Function DossPasPack_f(Extends CetElt as FolderItem) As Boolean

  Dim TampDrap as Boolean ' Retourne Vrai si  CetElt  est un directory mais pas un Pa
  ckage

  If CetElt.Directory Then
    TampDrap = (not CetElt.IsPackage_f) ' Si pas TargetMacOS ça rtourne (not False
    ) donc True
  Else
    TampDrap = False
  End If
```

```
  ' Je pourrais faire : TampDrap = CetElt.Directory and (not CetElt.IsPackage_f)  car si
  1er test à False alors on ne fait pas le 2ième mais je n'aime pas

  Return TampDrap

End Function
```

Fncts.Extension_f:

```
Function Extension_f(Extends CetElt as FolderItem) As String

  Dim TampTabNom_f(-1), Elt_ext as String ' Renvoie l'extension de File AVEC le . ou r
  ien si pas d'extension
  Dim NbEltTabNom_f as Int16 ' Cette fonction est inexistante sous RealBasic
  ' Je pourrais vérifier que si  CetElt  est Directory et non Package alors Extension = ""
   mais ce n'est pas toujours le cas

  TampTabNom_f = Split(CetElt.Name, ".")
  NbEltTabNom_f = UBound(TampTabNom_f)

  If (NbEltTabNom_f = 0) or ((NbEltTabNom_f = 1) and (TampTabNom_f(0) = "")) Then '
   (NbEltTabNom_f = 1) and (Left(CetElt.Name, 1) = ".")
    ' Pas de . ou il n'y en a qu'un au début du nom. Je pourrais ne tester que si . au
    début du nom sans me soucier s'il y en a d'autres
    Elt_ext = ""
  Else
    Elt_ext = ("." + TampTabNom_f(NbEltTabNom_f))
  End If

  Return Elt_ext

  ' Solution via un AppleScript :
  ' NomFichExtention = GetFileExtension(CetElt.AbsPath_f) ' Script qui renvoie l'exten
  sion SANS le .
  ' If NomFichExtention <> "" Then NomFichExtention = "." + NomFichExtention

  ' Solution en masquant l'extension (et en la remettant comme avant ensuite) :
  ' CetElt.ExtensionVisible = False
  ' NomFichSsExt = CetElt.DisplayName
  ' Cette méthode posait des Problèmes sous Leopard X 10.5

  ' Autre Solution RealBasic
  ' Dim NomFich, NomFichExtention as String
```

```
  ' Dim nb As Int16

  ' NomFich = CetElt.Name
  ' nb = CountFields(NomFich, ".") ' Découpe le nom
  ' If (nb = 1) or ((nb = 2) and (Left(NomFich, 1) = ".")) Then ' (nb = 2) and (NthField(N
  omFich, ".", 1) = "")
  ' Pas de . ou il n'y en a qu'un au début du nom
  ' NomFichExtention = ""
  ' Else
  ' NomFichExtention = "." + NthField(NomFich, ".", nb)
  ' End If

  ' MsgBox "Fichier : '" + CetElt.AbsPath_f + "'" + EndOfLine + "ext : '" + NomFichExte
  ntion + "'"
  ' Return NomFichExtention

End Function
```

Fncts.FSRef_f:

```
Private Function FSRef_f(Extends CetElt as FolderItem) As MemoryBlock

  ' Cette Fonction doit être Privée car appellée uniquement par  IsPackage_f  pour  Re
  alStudio donc 10.5 (et - , fonctionne sous 10.6)

  ' Ressemble à FolderItemToFSRef de CarbonDeclareLibrary
  Dim Out_f as MemoryBlock ' Voir si on utilise CaronLib ou Carbon

  ' #If TargetCarbon Then ' On ne l'appele que si Carbon
  Out_f = new MemoryBlock(80)

  Soft Declare Function FSGetVolumeInfo Lib "Carbon" (volume as Short, volumeIndex
  as Integer, actualVolume as Ptr, whichInfo as Integer, info as Ptr, volumeName as Ptr,
   rootDirectory as Ptr) as Short
  Soft Declare Function FSMakeFSRefUnicode Lib "Carbon" (parentPtr as Ptr, nameLeng
  th as Integer, name as CString, enc as Integer, outRef as Ptr) as Short

  Const kTextEncodingUnknown = &hFFFF
  Const kFSVolInfoNone = &h0000
  Dim OSErr as Integer

  If CetElt.Parent is Nil Then ' CetElt is the root directory of the volume
     OSErr = FSGetVolumeInfo(CetElt.MacVRefNum, 0, Nil, kFSVolInfoNone, Nil, Nil,
     Out_f)
```

```
  Else
    Dim parentFSRef as memoryBlock = CetElt.Parent.FSRef_f
    Dim itemName as String = ConvertEncoding(CetElt.Name, Encodings.UTF16)

    OSErr = FSMakeFSRefUnicode(parentFSRef, Len(itemName), itemName, kTextEn
    codingUnknown, Out_f)
  End If
  ' #EndIf

  Return Out_f ' as MemoryBlock ' Vide si Fich n'existe pas ou autre problème

End Function
```

Fncts.GetFileValue:

```
Private Function GetFileValue(CetElt as FolderItem, key as String, byref result as ptr) As B
oolean

  ' Cette Fonction doit être Privée car appellée uniquement par  IsPackage_f  Method
  pour savoir si Package avec Xojo Cocoa, donc 10.6 (et +)
  ' Gets an NSURL resource value for the given FolderItem.

  ' @param f The FolderItem to inspect (non-nil).
  ' @param key The NSURL resource value key. See the documentation for NSURL.
  ' @param result Upon return, the resource value or error.
  ' @result Whether or not the function succeeded.

  Declare Function dlopen Lib "System" (path as CString, mode as Integer) as ptr
  Declare Function dlsym Lib "System" (handle as PTr, name as CString) as ptr
  Const RTLD_LAZY = 1
  Const RTLD_GLOBAL = 8

  Dim libPtr as ptr = dlopen("/System/Library/Frameworks/Foundation.framework/Fo
  undation", RTLD_LAZY or RTLD_GLOBAL)
  If libPtr = Nil Then Return False

  Dim symPtr as ptr = dlsym(libPtr, key)
  If symPtr = Nil Then Return False

  ' Now we can create the URL and get the value
  Declare Function NSClassFromString Lib "Foundation" (name as CFStringRef) as ptr
  Declare Function URLWithString Lib "Foundation" selector "URLWithString:" (obj as ptr
  , Str as CFStringRef) as ptr
```

```
  Declare Function getResourceValue Lib "Foundation" selector "getResourceValue:for
  Key:error:" (obj as ptr, ByRef Val as ptr, key as ptr, ByRef error as ptr) as Boolean

  Dim fileURL as ptr = URLWithString(NSClassFromString( "NSURL" ), CetElt.URLPath)
  Dim value, error as ptr
  If getResourceValue(fileURL, value, symPtr.ptr, error) Then
    result = value
    Return True
  Else
    result = error
    Return False
  End If

End Function
```

Fncts.GetFitemAbsPath:

```
Function GetFitemAbsPath(CetAbsPath as String, True_item as Boolean) As FolderItem

  Dim RetMonFitem, EssMonFitem as FolderItem ' Cette procédure fait à peu près la m
  ême chose que GetFolderItem sauf qu'elle
  Dim TampTabPath(-1) as String '  retourne Nil et nom le path de l'appli si on lui don
  ne "" ou ":" et qu'elle ne déconne pas
  Dim AvDerEltTabPath, iVol, iNbre, NbVol as Int16 '  quand on récupère 2 fois de suit
  e un item à partir de son path quand celui-ci contient des accents
  ' Et  GetTrueFolderItem  déconne aussi quand on cherche à obtenir le path de l'alias
   lui-même quand son nom contient des accents
  ' Voir notes plus bas  True_item a Vrai si on veut le vrai Item (l'alias) = GetTrueFolde
  rItem
  ' Par contre, tout comme GetFolderItem(), cette fonction retourne l'alias lui-même si
   la cible n'existe pas

  ' MsgBox "CetAbsPath :" + EndOfLine + CetAbsPath
  TampTabPath = Split(CetAbsPath, SepAbsPath)
  AvDerEltTabPath = UBound(TampTabPath)
  iNbre = 0
  If AvDerEltTabPath < 1 Then ' Il doit y avoir au moins un : car pour un volume on do
  it mettre "Mon HD:" et non "Mon HD" car ça retourne des conneries
    ' Et si  CetAbsPath  est vide on a  AvDerEltTabPath = UBound(TampTabPath) =
    -1 donc  TampTabPath(iNbre = 0)  plante
    RetMonFitem = Nil ' Je pourrais faire Return Nil ici

  ElseIf TampTabPath(iNbre) = "" Then ' Sinon on pourrait démarrer avec le path de l'a
  ppli si un truc du genre ":MonDoss:MonSsDoss:MonFichier.txt" mais non
```

```
  ' De plus s'il n'y avait que ":" GetTrueFolderItem("" + ":") renvoit Nil de toute faç
  on
  ' Et pour GetFolderItem("") qui retourne le path de l'appli elle-même, je ne veux
   pas que ça le fasse (et dans ce cas on serait dans test ci-dessus)
  ' Et je préfère faire 2 tests à la suite que (AvDerEltTabPath < 1) or (TampTabPat
  h(iNbre) = "")
  RetMonFitem = Nil ' Je pourrais faire Return Nil ici

Else '  AvDerEltTabPath ≥ 0 , il y a au moins un élément dans  TampTabPath
  If Right(CetAbsPath, 1) = SepAbsPath Then ' On ignore si on termine par  SepAb
  sPath = ":"
    #If DebugBuild Then
      If not(TampTabPath(AvDerEltTabPath) = "") Then MsgBox "Ca déconne
      Tom, GetFitemAbsPath, CetAbsPath se termine par ':' mais dernier elt
      pas vide"
    #EndIf
    TampTabPath.Remove AvDerEltTabPath ' Là  AvDerEltTabPath  est encore l
    e dernier élément, pas vraiment utile mais bon …
    AvDerEltTabPath = AvDerEltTabPath - 1
    CetAbsPath = Left(CetAbsPath, Len(CetAbsPath) - 1) ' On élimine le dernier
     SepAbsPath = ":"  pour contrôle en fin de procédure
  End If
  ' MsgBox "CetAbsPath : " + CetAbsPath + EndOfLine + "Nb EltTabPath_f = " + s
  tr(AvDerEltTabPath + 1)
  AvDerEltTabPath = AvDerEltTabPath - 1 ' Donc  AvDerEltTabPath ≥ -1  et c'est
  un volume si = -1
  RetMonFitem = Nil ' L'est par défaut mais bon …
  EssMonFitem = Nil ' Idem
  NbVol = VolumeCount - 1
  ' MsgBox "Nb volume : " + str(NbVol + 1) + EndOfLine + "iNbre = " + str(iNbre)
  For iVol = 0 to NbVol
    #If DebugBuild Then
      If not(iNbre = 0) Then MsgBox "Ca déconne Tom, GetFitemAbsPath, tu
      as oublié de remettre iNbre à 0.'"
    #EndIf
    ' MsgBox Volume(iVol).AbsPath_f + " - " + str(Volume(iVol).MacVRefNum)
    + EndOfLine + (TampTabPath(iNbre) + SepAbsPath) + EndOfLine + " = ? "
    + Cstr(Volume(iVol).AbsPath_f = (TampTabPath(iNbre) + SepAbsPath))
    If Volume(iVol).Name = TampTabPath(iNbre) Then
      EssMonFitem = Volume(iVol) ' Si  CetAbsPath  est un volume, donc rie
      n derrière, on ne s'occupe pas de si  True_item  ou non
      ' MsgBox "iNbre = " + str(iNbre) + EndOfLine + (TampTabPath(iNbre)
      + SepAbsPath) + EndOfLine + EssMonFitem.AbsPath_f + "  vol : " + str
      (EssMonFitem.MacVRefNum)
      For iNbre = 1 to AvDerEltTabPath
```

```
      If TampTabPath(iNbre) = "" Then EssMonFitem = Nil ' Je préfère fai
      re le test car ça renvoit des trucs bizarres des fois
      If EssMonFitem = Nil Then ' A pu être à Nil le tour de boucle d'ava
      nt
        Exit ' iNbre
      Else
        EssMonFitem = EssMonFitem.TrueChild(TampTabPath(iNbre))
         ' NON, surtout pas + SepAbsPath)
        ' MsgBox "iNbre = " + str(iNbre) + EndOfLine + TampTabPath
        (iNbre) + EndOfLine + EssMonFitem.AbsPath_f + "  vol : " + st
        r(EssMonFitem.MacVRefNum)
      End If
    Next iNbre
    If (not(EssMonFitem = Nil)) and (AvDerEltTabPath > -1) Then ' pas or (T
    ampTabPath(iNbre) = "") car il faut remettre EssMonFitem à Nil ci-dess
    ous
      iNbre = AvDerEltTabPath + 1
      If TampTabPath(iNbre) = "" Then ' Je préfère faire le test car ça ren
      voit des trucs bizarres des fois
        EssMonFitem = Nil ' On le remet à Nil car a pu prendre une va
        leur le tour d'avant si nom volume correspondait
      Else
        If True_item Then
          EssMonFitem = EssMonFitem.TrueChild(TampTabPath(iN
          bre)) ' NON, surtout pas + SepAbsPath)
        Else
          EssMonFitem = EssMonFitem.Child(TampTabPath(iNbre))
          ' NON, surtout pas + SepAbsPath)
        End If
      End If
    End If
  Else ' Nom du Volume ne correspond pas
    EssMonFitem = Nil ' On le remet à Nil car a pu prendre une valeur le to
    ur d'avant si nom volume correspondait
  End If
  If EssMonFitem = Nil Then
    iNbre = 0 ' On boucle sur Volume suivant en recommençant à 0
  Else
    If EssMonFitem.Exists Then
      ' If EssMonFitem.Directory and CetAbsPath se termine par : Then '
      Je pourrais tester si le path se termine par : donc dossier mais l'in
      struction GetTrueFolderItem ne fait pas la différence
      RetMonFitem = EssMonFitem
      Exit ' On en a trouvé un qui existe, on n'en cherche pas d'autre su
      r les volumes suivants
```

```
        Else
          If RetMonFitem = Nil Then RetMonFitem = EssMonFitem ' Sinon on
           garde le premier trouvé
          iNbre = 0 ' On boucle sur Volume suivant en recommençant à 0
        End If
      End If
    Next iVol

End If

#If DebugBuild Then
    Dim TampText as String

    If RetMonFitem = Nil Then
      ' MsgBox "CetAbsPath suivi de l'élément trouvé :" + EndOfLine + """ + CetA
      bsPath + """ + EndOfLine + "Nil"
    Else
      TampText = RetMonFitem.AbsPath_f
      If Right(TampText, 1) = ":" Then TampText = Left(TampText, Len(TampTex
      t) - 1) ' On enlève le dernier ":" car on l'a enlevé au début, et on a pu trouve
      r un fichier à la place d'un dossier ou vice versa
      ' MsgBox  "CetAbsPath suivi de l'élément trouvé :" + EndOfLine + """ + Cet
      AbsPath + """ + EndOfLine + """ + TampText +"""
      If True_item and (not(CetAbsPath = TampText)) Then ' Si pas  True_item  al
      ors on a trouvé l'Alias et le Path ne correspondra pas
        MsgBox "Ca déconne Tom, CetAbsPath suivi de l'élément trouvé :" + E
        ndOfLine + """ + CetAbsPath + """ + EndOfLine + """ + TampText +"""
      End If
    End If
#EndIf

Return RetMonFitem

' MAIS, il y a 2 problèmes :
' 1- RealBasic réencode le Name différement quand on refait GetFolderItem une 2iè
me fois avec des noms
'    contenant des caractères non ASCII, surtout Asiatiques
'    Même ça ne fonctionne pas : MonTampFitem = GetFolderItem(Join(Split(CetAbsP
ath, SepAbsPath), SepAbsPath))
' 2- Quand RealBasic lit un Path de FolderItem, il le fait en UTF8 et les caractères ac
centués sont
'    combinés (é est en fait e´). Mais quand j'enregistre dans un fichier texte en UTF
8 un path avec des accents
'    puis que je le relis (fichier Prefs), RealBasic transforme les e´ en é etc.. Donc qua
nd je compare avec un autre
```

```
    '    item.AbsolutePath j'ai des différences même quand il s'agit des mêmes fichiers
    ' 3- J'aurais pu abandonner l'utilisation des AbsolutePath et n'utiliser que URLPath
    mais l'AbsolutePath est tout de
    '    même plus simple à manipuler en String, et plus clair pour l'utilisateur (dans les
    Prefs etc.)

End Function
```

Fncts.GetTargetItem_f:

```
Function GetTargetItem_f(Extends CetElt as FolderItem) As FolderItem

    ' Pas trouvé d'autre moyen que de transformer en Path et de reprendre le TrueItem,
    note : Comme GetTrueFolderItem  donne l'alias

    ' 1ère solution
    ' CetElt = GetFitemAbsPath(CetElt.AbsPath_f, False)

    ' 2ème solution
    CetElt = GetFolderItem(CetElt.URLPath, FolderItem.PathTypeURL)

    Return CetElt ' Retourne l'Alias lui même si celui-ci ne pointe nulle part (n'a pas de c
    ible)

End Function
```

Fncts.InStrDG:

```
Function InStrDG(DGStart as Int16, DGSource as String, DGFind as String, DGOccur as Int
16, DepDroite as Boolean) As Int16

    Dim iNbre, Pos, TempPos, Occur, Deb_iNbre as Int16

    ' Cette procédure est InStr : Elle recherche DGFind dans DGSource en démarrant de
    DGStart
    ' Mais améliorée : Elle retourne la DGOccur ième occurence trouvée, et démarre de l
    a droite à rebour si DepDroite à Vrai
    ' Note : On démarre de DGFind pour le 1er caractère cherché. Et on renvoie la positi
    on du 1er caractère de DGFind si on cherche
    ' depuis la gauche, et la position du dernier si on cherche depuis la droite. Exemple
    :
    ' InStrDG(1, "C'est notre ville", "notre", 1, True) = 7     InStrDG(9, "C'est notre ville", "
    notre", 1, True) = 0
```

```
If DGStart < 1 Then DGStart = 1
If (DGStart > Len(DGSource)) or (DGOccur < 1) or (DGSource = "") or (DGFind = "") T
hen
   Pos = 0
Else
   Occur = 0

   If DepDroite Then ' On cherche depuis la droite
      DGSource = Left(DGSource, Len(DGSource) + 1 - DGStart) ' On ne garde qu
      e la partie dans laquelle on cherche
      Pos = Len(DGSource) + 1
      Deb_iNbre = Pos - Len(DGFind)
      For iNbre = Deb_iNbre DownTo 1 ' Pas la peine de chercher après (si Len(D
      GSource) < Len(DSFind) on ne fait pas la boucle)
         TempPos = InStr(iNbre, DGSource, DGFind)
         If (TempPos > 0) and (TempPos < Pos) Then
            Pos = TempPos
            Occur = Occur + 1
            If Occur = DGOccur Then Exit
         End If
         ' MsgBox "Pos = " + Str(Pos) + "    Occur = " + Str(Occur)
      Next iNbre
      If Occur = DGOccur Then
         Pos = Len(DGSource) + DGStart - (Pos + Len(DGFind) - 1)
         '         Len DGSource originale  - Pos (on compte depuis la fin)   on tie
         nt compte de la position du dernier caractère recherché
      Else
         Pos = 0 ' On a fini la boucle sans faire Exit (donc sans avoir trouvé)
      End If

   Else ' On cherche depuis la gauche
      Pos = DGStart - 1
      Do
         Pos = InStr(Pos + 1, DGSource, DGFind)
         Occur = Occur + 1 ' Note : Même si Occur = DGOccur  et qu'on sort d
         e la boucle, si pos = 0 pas de problème
         ' MsgBox "Pos = " + Str(Pos) + "    Occur = " + Str(Occur)
      Loop Until (Occur = DGOccur) or (Pos = 0)

   End If
End If

Return Pos
```

End Function

Fncts.IsPackage_f:

```
Function IsPackage_f(Extends CetElt as FolderItem) As Boolean

  Dim DrapPackage as Boolean ' Retourne Vrai si  CetElt  est un Package (.app .rtfd . p
  kg .mpkg etc.) et Faux sinon. Note : J'utilise  Extends  pour pouvoir écrire  MonElt.Is
  Package_f

  #If DebugBuild Then ' Si  CetElt  n'existe pas ça plantera, doit être testé avant
    If not CetElt.Directory Then MsgBox "Problème Tom, IsPackage_f , tu ne devrais
    (dois ?) y tester que des dossiers !"
  #EndIf

  #If TargetMacOS Then
    #If RBVersion < 2013 Then ' RealStudio donc 10.5 (et +) ' If TargetCarbon Then
      Declare Function LSCopyItemInfoForRef lib "ApplicationServices" (ref as Ptr,
      whichInfo as Integer, info as Ptr) as Integer
      Dim err as Integer
      Dim rec as New MemoryBlock(24)

      err = LSCopyItemInfoForRef(CetElt.FSRef_f, &h4, rec)
      DrapPackage = not(Bitwise.BitAnd(rec.Long(0), 2) = 0)

    #Else ' Xojo donc 10.6 et + ' #If TargetCocoa Then
      Declare Function boolValue Lib "Foundation" selector "boolValue" ( obj As pt
      r ) As Boolean
      Dim isPackage As ptr

      If GetFileValue(CetElt, "NSURLIsPackageKey", isPackage) Then
        DrapPackage = boolValue(isPackage)
      Else
        ' At this point, isPackage isa NSError. Do something with it or just assu
        me that it's not a package.
        DrapPackage = False
      End If

    #EndIf ' Xojo ou RealStudio

  #Else ' Windows , Linux
    DrapPackage = False
```

```
  #EndIf

  ' Méthose Commande Shell
  '#If TargetMacOS Then
  'Dim CdeShell as New Shell
  '' NE PAS utiliser  """ + MonFolderItem.ShellPath_fAS + """
  '' NON : CdeShell.Execute "mdls -name kMDItemContentTypeTree '" + CetElt.ShellPa
  th + """
  '' On n'utilise pas le Quotedform avec ' ' car s'il y a un ' dans le nom ça merde
  '' CdeShell.Execute "mdls " + CetElt.ShellPath
  '' MsgBox CdeShell.Result
  '' Avant : CdeShell.Execute "mdls -name kMDItemContentType -raw " + CetElt.Shell
  Path
  'CdeShell.Execute "mdls -name kMDItemContentTypeTree " + CetElt.ShellPath
  'If CdeShell.ErrorCode = 0 Then
  '' MsgBox CdeShell.Result
  'DrapPackage = not(Instr(1, CdeShell.Result, "com.apple.package") = 0)
  '' Avant : DrapPackage = not((CdeShell.Result = "public.folder") or (CdeShell.Result
  = "public.volume"))
  'Else ' Dossier sur lequel on n'a pas les droits ou etc.
  'DrapPackage = False
  '' MsgBox "Error code : " + Str(CdeShell.ErrorCode)
  'End If
  '
  '' Avant avec AppleScript mais c'est plus lent :
  '' DrapPackage = (IsPackage_s(CetElt.ShellPath_fAS) = "True") ' Méthode AppleScript
  plus lente que méthode avec Carbon
  '
  '#Else
  'DrapPackage = False
  '
  '#EndIf

  Return DrapPackage

End Function
```

Fncts.LireInfoPlist:

```
Function LireInfoPlist(VmimiSyst as String) As Boolean

  Dim f_Elt as FolderItem ' J'utilise cette méthode pour savoir avec quoi je compile (Re
  alStudio ou Xojo) car le système minimum change
```

```
  Dim Stream as TextInputStream
  Dim TampText as String

  f_Elt = App.ExecutableFile.Parent.Parent.Child("Info.plist")
  ' MsgBox f_Elt.AbsPath_f

  If f_Elt.Exists Then ' Ne peut pas être Nil : PasNil_Existe_Alias(f_Elt, True)
    Stream = TextInputStream.Open(f_Elt)
    Stream.Encoding = DefAppEncod
    While not Stream.eof
      TampText = Stream.ReadAll
    Wend
    Stream.Close
  Else
    TampText = "" ' Y est déjà
  End If
  ' MsgBox "Pos '" + VmimiSyst + "' = " + str(InStr(1, TampText, VmimiSyst)) + EndOf
  Line + " " + EndOfLine + TampText

  Return (InStr(1, TampText, VmimiSyst) > 0)

End Function
```

Fncts.MyDialogBox:

```
Function MyDialogBox(Texte1 as String, Texte2 as String, BtnAct as String, BtnAnnul as S
tring, BtnAlt as String, Icone as Int16) As Int16

  Dim MessDiag as New MessageDialog
  Dim MessButt as MessageDialogButton
  Dim BoutonClique as Int16
  Dim DrapMenuTemp as Boolean

  ' Cette Procédure fait exactement la même chose qu'un MsgBox sauf que ses bouto
  ns sont paramétrables
  ' Et qu'elle masque les menus que l'on veut. Une MsgBox ne masque pas les menus
  ayant des sous-menus

  App.MouseCursor = System.Cursors.StandardPointer ' Si on faisait Pomme-... avec l
  e pointeur sur une carte ça resterait le doigt
  '  Ou si une alerteBox apparaît pendant que le curseur est le doigt
  DrapMenuTemp = DrapMenu ' Voir Notes dans WinMain
  DrapMenu = False ' Voir Notes sur CancelClose (on peut quitter sous Windows)
```

```
' @Menu@ EnableMenuItems ' Inutile

MessDiag.icon = Icone ' -1 pas d'icône, 0 Application icône, 1 Caution icône, 2 et 3
 = 0 sous MacOsX
MessDiag.Title = "" ' Windows et Linux seulement
If not(BtnAct = "") Then
   ' MessDiag.ActionButton.Visible = True ' Affiché par défaut (et forcément visibl
   e de toute façon)
   MessDiag.ActionButton.Caption = BtnAct ' "Ok"
Else
   ' MessDiag.ActionButton.Visible = False ' Impossible, le 1er bouton est forcéme
   nt visible
   ' MessDiag.ActionButton.Caption = "Ok" ' Sinon mettrait OK en Anglais
   MsgBox "Error !"+EndOfLine+"Please contact the authors."+EndOfLine+"My Dial
   ogBox, BtnAct = ""
End If
If not(BtnAnnul = "") Then
   MessDiag.CancelButton.Visible = True ' Affiche 2ième bouton (masqué par défa
   ut)
   MessDiag.CancelButton.Caption = BtnAnnul ' Btn_Annuler ' Btn_Non
End If
If not(BtnAlt = "") Then
   MessDiag.AlternateActionButton.Visible = True ' Affiche 3ième bouton (masqué
    par défaut)
   MessDiag.AlternateActionButton.Caption = BtnAlt
End If
If not(Texte1 = "") Then
   MessDiag.Message = Texte1 ' Texte en gras s'il y a du Texte2
Else
   MessDiag.Message = " " ' Sinon affiche ^0
End If
If not(Texte2 = "") Then MessDiag.Explanation = Texte2 ' Texte en petit

MessButt = MessDiag.ShowModal ' Affiche la boîte de dialogue
Select Case MessButt ' Determine which button was pressed.
Case MessDiag.ActionButton
   BoutonClique = 1
Case MessDiag.CancelButton
   BoutonClique = 2
Case MessDiag.AlternateActionButton
   BoutonClique = 3
Else ' Ne peut pas arriver
   BoutonClique = 0
End select
```

```
  DrapMenu = DrapMenuTemp
  ' @Menu@ EnableMenuItems ' Inutile

  App.MouseCursor = Nil ' IMPORTANT, sinon le curseur reste toujours la flèche, quoi
   que l'on fasse dans les fenêtres

  Return BoutonClique ' 1 si ActionBouton, 2 si Annule et 3 si bouton alternatif

End Function
```

Fncts.MyDialogFolder:

```
Function MyDialogFolder(CetElt as FolderItem, DiagTxt as String) As FolderItem

  Dim Dialog as SelectFolderDialog ' Ca c'est pour les dossiers
  Dim f_elt as FolderItem
  Dim DrapMenuTemp as Boolean

  App.MouseCursor = System.Cursors.StandardPointer ' Si on faisait Pomme-... avec l
  e pointeur sur une carte ça resterait le doigt
  '  Ou si une alerteBox apparaît pendant que le curseur est le doigt
  DrapMenuTemp = DrapMenu ' Voir Notes dans WinMain
  DrapMenu = False ' Voir Notes sur CancelClose (on peut quitter sous Windows)
  ' @Menu@ EnableMenuItems ' Inutile

  Dialog = New SelectFolderDialog ' SelectFolder  ne permet pas de choisir dossier pa
  r défaut
  Dialog.Title = DiagTxt
  Dialog.PromptText = "" ' "Sélectionnez le Fichier Programmes :"
  ' Dialog.ActionButtonCaption = "Choisir" ' Même séries de test dans MyDialogFolder
  , MyDialogLoad, MyDialogSave
  If not(CetElt = Nil) Then
    If CetElt.DossPasPack_f Then
      Dialog.InitialDirectory = CetElt
      ' Pas de  SuggestedFileName
    ElseIf not(CetElt.Parent = Nil) Then ' Si ce n'est pas un Directory (dossier ou volu
    me) alors il devrait avoir un parent mais je teste quand même
      Dialog.InitialDirectory = CetElt.Parent
      ' Pas de  SuggestedFileName
    Else
      CetElt = Nil
    End If
```

```
  End If
  If CetElt = Nil Then
    If SpecialFolder.Documents = Nil Then ' Je préfére tester des fois que ça merde
      Dialog.InitialDirectory = GetFolderItem("")
    Else
      Dialog.InitialDirectory = SpecialFolder.Documents
      ' MsgBox SpecialFolder.Documents.AbsPath_f
    End If
    ' Pas de  SuggestedFileName
  End If
  ' Dialog.Filter = FiltrTxt ' Ces types doivent être définis vias Menu Edit / File Types..
  .
  f_elt = Dialog.ShowModal
  ' MsgBox "f_elt = " + f_elt.AbsPath_f

  DrapMenu = DrapMenuTemp
  ' @Menu@ EnableMenuItems ' Inutile
  App.MouseCursor = Nil ' IMPORTANT, sinon le curseur reste toujours la flèche, quoi
   que l'on fasse dans les fenêtres

  Return f_elt ' A  Nil  si cliqué  Annuler

End Function
```

Fncts.MyDialogLoad:

```
Function MyDialogLoad(CetElt as FolderItem, DiagTxt as String, FiltrTxt as String) As Fold
erItem

  Dim Dialog as OpenDialog ' New SelectFolderDialog ' Ca c'est pour les dossiers
  Dim f_elt as FolderItem
  Dim DrapMenuTemp as Boolean

  App.MouseCursor = System.Cursors.StandardPointer ' Si on faisait Pomme-... avec l
  e pointeur sur une carte ça resterait le doigt
  '  Ou si une alerteBox apparaît pendant que le curseur est le doigt
  DrapMenuTemp = DrapMenu ' Voir Notes dans WinMain
  DrapMenu = False ' Voir Notes sur CancelClose (on peut quitter sous Windows)
  ' @Menu@ EnableMenuItems ' Inutile

  Dialog = New OpenDialog ' CetElt = GetOpenFolderItem("list/TEXT") ' Peu flexible, p
  as de prompt etc.
  Dialog.Title = DiagTxt
```

```
  Dialog.PromptText = "" ' "Sélectionnez le Fichier Programmes :"
  ' Dialog.ActionButtonCaption = "Ouvrir" ' Même séries de test dans MyDialogFolder,
   MyDialogLoad, MyDialogSave
  If not(CetElt = Nil) Then
    If CetElt.DossPasPack_f Then
      Dialog.InitialDirectory = CetElt
      ' Pas de  SuggestedFileName
    ElseIf not(CetElt.Parent = Nil) Then ' Si ce n'est pas un Directory (dossier ou volu
    me) alors il devrait avoir un parent mais je teste quand même
      Dialog.InitialDirectory = CetElt.Parent
      ' Pas de  SuggestedFileName
    Else
      CetElt = Nil
    End If
  End If
  If CetElt = Nil Then
    If SpecialFolder.Documents = Nil Then ' Je préfére tester des fois que ça merde
      Dialog.InitialDirectory = GetFolderItem("")
    Else
      Dialog.InitialDirectory = SpecialFolder.Documents
      ' MsgBox SpecialFolder.Documents.AbsPath_f
    End If
    ' Pas de  SuggestedFileName
  End If
  Dialog.Filter = FiltrTxt ' Ces types doivent être définis vias Menu Edit / File Types...
  f_elt = Dialog.ShowModal
  ' MsgBox "f_elt = " + f_elt.AbsPath_f

  DrapMenu = DrapMenuTemp
  ' @Menu@ EnableMenuItems ' Inutile
  App.MouseCursor = Nil ' IMPORTANT, sinon le curseur reste toujours la flèche, quoi
   que l'on fasse dans les fenêtres

  Return f_elt ' A  Nil  si cliqué  Annuler

End Function
```

Fncts.MyDialogSave:

```
Function MyDialogSave(CetElt as FolderItem, DiagTxt as String, FiltrTxt as String, SuggNo
mTxt as String) As FolderItem

  Dim Dialog as SaveAsDialog
```

```
Dim f_elt as FolderItem
Dim DrapMenuTemp as Boolean

App.MouseCursor = System.Cursors.StandardPointer ' Si on faisait Pomme-... avec l
e pointeur sur une carte ça resterait le doigt
'  Ou si une alerteBox apparaît pendant que le curseur est le doigt
DrapMenuTemp = DrapMenu ' Voir Notes dans WinMain
DrapMenu = False ' Voir Notes sur CancelClose (on peut quitter sous Windows)
' @Menu@ EnableMenuItems ' Inutile

Dialog = New SaveAsDialog
Dialog.Title = DiagTxt
Dialog.PromptText = "" ' "Sélectionnez le Fichier Programmes :"
' Dialog.ActionButtonCaption = "Enregistrer" ' Même séries de test dans MyDialogFo
lder, MyDialogLoad, MyDialogSave
If not(CetElt = Nil) Then
   If CetElt.DossPasPack_f Then
      Dialog.InitialDirectory = CetElt
      Dialog.SuggestedFileName = SuggNomTxt
   ElseIf not(CetElt.Parent = Nil) Then ' Si ce n'est pas un Directory (dossier ou volu
   me) alors il devrait avoir un parent mais je teste quand même
      Dialog.InitialDirectory = CetElt.Parent
      Dialog.SuggestedFileName = CetElt.Name
   Else
      CetElt = Nil
      ' SuggestedFileName défini plus bas
   End If
End If
If CetElt = Nil Then
   If SpecialFolder.Documents = Nil Then ' Je préfére tester des fois que ça merde
      Dialog.InitialDirectory = GetFolderItem("")
   Else
      Dialog.InitialDirectory = SpecialFolder.Documents
      ' MsgBox SpecialFolder.Documents.AbsPath_f
   End If
   Dialog.SuggestedFileName = SuggNomTxt
End If
Dialog.Filter = FiltrTxt ' Ces types doivent être définis vias Menu Edit / File Types...
f_elt = Dialog.ShowModal
' MsgBox "f_elt = " + f_elt.AbsPath_f

DrapMenu = DrapMenuTemp
' @Menu@ EnableMenuItems ' Inutile
App.MouseCursor = Nil ' IMPORTANT, sinon le curseur reste toujours la flèche, quoi
 que l'on fasse dans les fenêtres
```

```
  Return f_elt ' A  Nil  si cliqué  Annuler

End Function
```

Fncts.PasNil_Existe_Alias:

```
Function PasNil_Existe_Alias(CetElt as FolderItem, TestAlias as Boolean) As Boolean

  Dim Elt_Existe as Boolean ' Cette procédure renvoie True si  CetElt  Existe SANS PLA
  NTER si il est à Nil
  Dim CetElt_UrlPath as String ' Elle renvoie False si à Nil  Quoiqu'il suffise de teser si
  Nil avant Exists : If not(CetElt = Nil) and CetElt.Exists Then

  ' Si  TestAlias  est vrai elle teste aussi si  CetElt  est un alias et si cet alias est résolu
  (si il ne pointe pas dans le vide)
  ' Note : Un alias non résolu pointe sur lui-même

  If CetElt = Nil Then
    Elt_Existe = False
  Else
    Elt_Existe = CetElt.Exists
    If Elt_Existe and TestAlias and CetElt.Alias Then ' Si l'Alias lui-même n'existe pa
    s, je ne cherche pas à tester si Cible existe ou non
      ' Si  TestAlias à Faux on ne s'occupe pas de si  CetElt  est un alias ou pas et
       si  CetElt  n'est pas un alias on ne fait pas le test ci-dessous
      CetElt_UrlPath = CetElt.URLPath
      ' MsgBox CetElt_UrlPath + EndOfLine + "Alias " + Cstr(CetElt.Alias) + "  Tes
      tAlias " + Cstr(TestAlias) + "  =? " + Cstr(CetElt_UrlPath = GetFolderItem(C
      etElt_UrlPath, FolderItem.PathTypeURL).URLPath) + EndOfLine + GetFolderI
      tem(CetElt_UrlPath, FolderItem.PathTypeURL).URLPath
      Elt_Existe = not(CetElt_UrlPath = GetFolderItem(CetElt_UrlPath, FolderItem.
      PathTypeURL).URLPath) ' A  Faux  si Alias non résolu
      ' Si  CetElt  est un alias pointant dans le vide alors  CetElt.AbsPath_f  retour
      ne le path de l'alias lui-même au lieu
      '  du path de l'élément pointé par l'alias. Je pense qu'il s'agit d'une erreur d
      e RealBasic, il devrait retourner  Nil  ?!
      ' Obligé d'utiliser URLPath car même problème qu'indiqué dans ma procéd
      ure GetFitemAbsPath(), retourne toujours ≠ si caractère accentués ou Asiat
      iques etc. même si =
    End If
  End If
```

```
  ' MsgBox "Existe? " + Cstr(Elt_Existe)

  Return Elt_Existe

  ' Ancienne méthode : Ne fonctionnait pas bien avec caractères accentués, Asiatique
  s, etc.  Voir notes GetFitemAbsPath()
  ' If Elt = Nil then
  ' Elt_Existe = False
  ' Else
  ' If TestAlias and Elt.Alias Then ' Si  TestAlias à Faux on ne s'occupe pas de si  Elt  e
  st un alias ou pas
  '                              ou    Si  Elt  est n'est pas un alias on ne fait pas le test ci-des
  sous
  ' Elt_Existe = not(Elt.AbsPath_f = GetFolderItem(Elt.AbsPath_f).AbsPath_f) ' A  Faux
  si Alias non résolu
  ' Si Elt est un alias pointant dans le vide alors  Elt.AbsPath_f  retourne le path de l'ali
  as lui-même au lieu
  '  du path de l'élément pointé par l'alias. Je pense qu'il s'agit d'une erreur de RealBa
  sic, il devrait retourner  Nil  ?!
  ' Else
  ' Elt_Existe = Elt.exists
  ' End If
  ' End If

End Function
```

Fncts.Pause:

```
Sub Pause(Delai as UInt16, FaireEvents as Boolean)

  Dim TicksStart as Double

  TicksStart = Ticks ' 1 Ticks = 1/60 ième de seconde
  '                    Delai = 60  -> Pause d' 1 seconde

  While Ticks < (TicksStart + Delai)
    If FaireEvents Then App.DoEvents ' On fait les DoEvents si à Vrai
    ' Sinon on boucle sans rien faire d'autre
  Wend

End Sub
```

Fncts.PressPapTxt:

```
Sub PressPapTxt(CeTexte as String)

   Dim PressPap as New Clipboard

   PressPap.Text = CeTexte

   ' Ou
   ' Clipboard.SetText = CeTexte

   PressPap.Close

End Sub
```

Fncts.ShellPath_fAS:

```
Function ShellPath_fAS(Extends CetElt as FolderItem) As String

   Dim TampTextA as String ' Note : J'utilise  Extends  pour pouvoir écrire  MonElt.Shel
   lPath_fAS  mais ça fera une Nil Objection si à Nil
   ' #If DebugBuild Then
   ' Dim TampTextB as String
   ' #EndIf

   TampTextA = ReplaceAll(CetElt.ShellPath, "\\", "::") ' Car impossible d'avoir 2 ':' à la
   suite
   TampTextA = ReplaceAll(TampTextA, "\", "") ' Tous enlevés sauf cas particulier du \
   TampTextA = ReplaceAll(TampTextA, "::", "\")

   '#If DebugBuild Then ' Je n'emploie pas cette méthode car je ne suis pas sûr de rech
   ercher/remplacer tous les caractères. Je ne fais plus le test car il y a  °<>  et pleins
   d'autres
   'TampTextB = ReplaceAll(CetElt.ShellPath, "\ ", " ")
   'TampTextB = ReplaceAll(TampTextB, "\:", ":")
   'TampTextB = ReplaceAll(TampTextB, "\\", "\")
   'TampTextB = ReplaceAll(TampTextB, "\'", "'")
   'TampTextB = ReplaceAll(TampTextB, "\""", """")
   '' TampTextB = ReplaceAll(TampTextB, , ) ' A rajouter s'il en y a d'autres
   'If not(TampTextA = TampTextB) Then
   'PressPapTxt(CetElt.ShellPath + EndOfLine + TampTextA + EndOfLine + TampTextB
   )
   'MsgBox "ShellPath_fAS (ds PressPap) :" + EndOfLine + CetElt.ShellPath + EndOfLine
    + TampTextA + EndOfLine + TampTextB
```

```
  'End If
  '#EndIf

  Return TampTextA ' Avant je retournais simplement  ReplaceAll(CetElt.ShellPath, "\"
  , "")

End Function
```

Fncts.TypeCompil:

```
Function TypeCompil() As String

  Dim TampText as String

  #If TargetWin32 Then ' Est aussi X86 (Intel)
    TampText = "Windows"
  #ElseIf TargetMacOS Then
    #If TargetX86 Then
      #If TargetCarbon Then
        TampText = "Mac INTEL - Carbon"
      #ElseIf TargetCocoa Then
        TampText = "Mac INTEL - Cocoa"
      #Else
        TampText = "Mac INTEL - ???"
      #EndIf
    #ElseIf TargetPowerPC Then
      TampText = "Mac PPC"
    #Else
      TampText = "???"
    #EndIf
  #Else
    TampText = "???"
  #EndIf

  TampText = TampText + " (" + RBVersionString + ")."
  ' #If RBVersion < 2013 Then
  ' TampText = TampText + " (RS)."
  ' #Else
  ' TampText = TampText + " (Xojo)."
  ' #EndIf

  Return TampText
```

```
End Function
```

Fncts.Volume_f:

```
Function Volume_f(Extends CetElt as FolderItem) As FolderItem

  Dim f_elt as FolderItem ' Cette procédure retourne le volume de  CetElt . Si  CetElt  à
   Nil ça plante, si  CetElt  sur volume non monté ça retourne Nil

  #If TargetMacOS Then ' Cette procédure sert aussi à tester que trouve pas 2 volume
  s qui correspondent, ce qui est normalement impossible
    f_elt = CetElt
    While not(f_elt.Parent = Nil)
      f_elt = f_elt.Parent
    Wend

    ' MsgBox CetElt.AbsPath_f + EndOfLine + f_elt.AbsPath_f
    #If DebugBuild Then
      If not(f_elt.AbsPath_f = NthField(CetElt.AbsPath_f, SepAbsPath, 1) + SepAbs
      Path) Then ' Même chose que dans version autre plateforme ci-dessous
        Break ' Crée un BreakPoint car si on est dans un Thread la MsgBox fera
         tout planter
        MsgBox "Problème Tom :" + EndOfLine + f_elt.AbsPath_f + EndOfLine
        + NthField(CetElt.AbsPath_f, SepAbsPath, 1) + SepAbsPath
      End If
    #EndIf

  #Else ' Win32 ou autre
    MsgBox "Error ! Please contact the authors." + EndOfLine + "Volume_f is only fo
    r TargetMacOS !"
    f_elt = GetFolderItem(NthField(CetElt.AbsPath_f, SepAbsPath, 1) + SepAbsPath,
    FolderItem.PathTypeAbsolute) ' Même chose que dans test ci-dessus

  #EndIf

  Return f_elt


  ' Avant j'utilisais  NoVolume qui utilisait MacFSRef mais ça marchait mal car MacFSR
  ef est un id unique par fichier ou dossier, de plus n'aime pas être appelé plein de fo
  is
```

```
  '#If TargetMacOS Then ' Cette procédure sert aussi à tester que trouve pas 2 volum
  es qui correspondent, ce qui est normalement impossible
  'Dim iNoVol as Int16
  '
  'iNoVol = NoVolume(CetElt) ' Si reste à -1 on aura une NilObjectException au Return
   Volume(iNoVol),
  'If iNoVol = -1 Then
  'f_elt = Nil ' Note : Volume(-1) retourne Nil mais je préfére  (f_elt est déjà à Nil mais
   ...)
  '
  'Else
  '#If DebugBuild Then
  'If not(Volume(iNoVol).AbsPath_f = (NthField(CetElt.AbsPath_f, SepAbsPath, 1) + Se
  pAbsPath)) Then MsgBox "Problème Tom :" + EndOfLine + Volume(iNoVol).AbsPath
  _f + EndOfLine + (NthField(CetElt.AbsPath_f, SepAbsPath, 1) + SepAbsPath)
  '#EndIf ' Même chose que dans version autre plateforme ci-dessous
  '
  'f_elt = Volume(iNoVol)
  '
  'End If
  '
  '#Else
  '#If DebugBuild Then
  'Break ' Crée un BreakPoint car si on est dans un Thread la MsgBox fera tout planter
  'MsgBox "Problème Tom, tu devrais être en TargetMacOS !"
  '#EndIf
  'f_elt = GetFolderItem(NthField(CetElt.AbsPath_f, SepAbsPath, 1) + SepAbsPath) ' Mê
  me chose que dans test ci-dessus
  '
  '#EndIf

End Function

DefAppEncod As TextEncoding


DrapMenu As Boolean
```

Fncts Note: @ A lire - External

@ A lire - External

Il faut sauvegarder le module Fncts en faisant un Ctril-clic dessus puis en choisissant "Make External ..."

Ensuite, dans le projet du programme, appuyer sur Alt tout en sélectionnant le menu "File", cela transformera le menu "Import..." en "Import es external..."

Fncts Note: GetFitemAbsPath2

GetFitemAbsPath2

```
' Ne marchait plus sous Leopard (ou avec dernière version de RB) avec les alias dont le path contenait des accents

Dim MonTampFitem, RetMonFitem as FolderItem ' Cette procédure fait à peu près la même chose que GetFolderItem sauf qu'elle
Dim TampTabPath_f(-1) as String '  retourne Nil et nom le path de l'appli si on lui donne "" ou ":" et qu'elle ne déconne pas
Dim DerEltTabPath_f as Int16 '  quand on récupère 2 fois de suite un item à partir de son path quand celui-ci contient des accents
' Voir notes plus bas  True_F a Vrai si on veut le vrai Item (l'alias) = GetTrueFolderItem
' Par contre, tout comme GetFolderItem(), cette fonction retourne l'alias lui-même si la cible n'existe pas

' MsgBox "Abs_Path_f :" + EndOfLine + Abs_Path_f
If (Abs_Path_f = "") or (Abs_Path_f = ":") Then ' GetFolderItem retournerait le path de l'appli elle-même
RetMonFitem = Nil ' Je pourrais faire Return Nil ici
Else
MonTampFitem = GetTrueFolderItem(Abs_Path_f) ' C'est un AbsolutePath qui peut se terminer par un mauvais encodage
' Prendre le TRUEfolderItem car si le nom de l'alias et de la cible différe ça déconnera
If MonTampFitem = Nil Then ' NON or (not MonTampFitem.Exists) même si n'existe pas je retourne l'item pour le créer ou ...
RetMonFitem = Nil ' Je pourrais faire Return Nil ici
Else
If MonTampFitem.Parent = Nil Then ' C'est un volume
' Alors on ne passe PAS par URLPath car ça déconne aléatoirement, 1 fois sur 5 environ on a  MonTampFitem.URLPath = ""    ???
' MsgBox "Abs_Path_f : " + Abs_Path_f + EndOfLine + "MonTampFitem.AbsPath2 : " + MonTampFitem.AbsolutePath + EndOfLine + "MonTampFitem.URLPath : " + MonTampFitem.URLPath
If True_f Then
```

```
RetMonFitem = GetTrueFolderItem(Abs_Path_f) ' = MonTampFitem
Else
RetMonFitem = GetFolderItem(Abs_Path_f)
End If
Else
TampTabPath_f = Split(Abs_Path_f, SepAbsPath)
DerEltTabPath_f = UBound(TampTabPath_f) ' Forcément > 0 puisqu'il y a un Parent
' MsgBox "Abs_Path_f : " + Abs_Path_f + EndOfLine + "MonTampFitem.AbsPath : " + Mo
nTampFitem.AbsolutePath + EndOfLine + "MonTam.Parent.AbsPath : " + MonTampFite
m.Parent.AbsolutePath + EndOfLine + """ + TampTabPath_f(DerEltTabPath_f - 1) + SepA
bsPath + TampTabPath_f(DerEltTabPath_f) + """
If TampTabPath_f(DerEltTabPath_f) = "" Then ' Abs_Path_f se termine par : MAIS il NE fau
t PAS l'ajouter ci-dessous
' MsgBox Abs_Path_f + EndOfLine + "= ? "+ Cstr(Abs_Path_f = MonTampFitem.Parent.A
bsolutePath + TampTabPath_f(DerEltTabPath_f - 1) + SepAbsPath) + EndOfLine + MonT
ampFitem.Parent.AbsolutePath + TampTabPath_f(DerEltTabPath_f - 1) + SepAbsPath
MonTampFitem = MonTampFitem.Parent.TrueChild(TampTabPath_f(DerEltTabPath_f - 1)
) ' + SepAbsPath) ' Il y a forcément au moins 2 élts
' Les 2 solutions, ci-dessus et ci-dessous, fonctionnent                    PAS de + Se
pAbsPath ci-dessus
' MonTampFitem = GetTrueFolderItem(MonTampFitem.Parent.AbsolutePath + TampTab
Path_f(DerEltTabPath_f - 1) + SepAbsPath)
Else
' MsgBox Abs_Path_f + EndOfLine + "= ? "+ Cstr(Abs_Path_f = MonTampFitem.Parent.A
bsolutePath + TampTabPath_f(DerEltTabPath_f)) + EndOfLine + MonTampFitem.Parent.
AbsolutePath + TampTabPath_f(DerEltTabPath_f)
MonTampFitem = MonTampFitem.Parent.TrueChild(TampTabPath_f(DerEltTabPath_f))
' Les 2 solutions, ci-dessus et ci-dessous, fonctionnent
' MonTampFitem = GetTrueFolderItem(MonTampFitem.Parent.AbsolutePath + TampTab
Path_f(DerEltTabPath_f))
End If
' MsgBox "Abs_Path_f : " + Abs_Path_f + EndOfLine + "MonTampFitem.AbsPath2 : " + M
onTampFitem.AbsolutePath + EndOfLine + "MonTampFitem.URLPath : " + MonTampFite
m.URLPath
If True_f Then
RetMonFitem = GetTrueFolderItem(MonTampFitem.URLPath, FolderItem.PathTypeURL)
Else
RetMonFitem = GetFolderItem(MonTampFitem.URLPath, FolderItem.PathTypeURL)
End If
End If
End If
End If

Return RetMonFitem
```

```
' MAIS, il y a 2 problèmes :
' 1- RealBasic réencode le Name différement quand on refait GetFolderItem une 2ième f
ois avec des noms
'    contenant des caractères non ASCII, surtout Asiatiques
'    Même ça ne fonctionne pas : MonTampFitem = GetFolderItem(Join(Split(Abs_Path_f, S
epAbsPath), SepAbsPath))
' 2- Quand RealBasic lit un Path de FolderItem, il le fait en UTF8 et les caractères accent
ués sont
'    combinés (é est en fait e´). Mais quand j'enregistre dans un fichier texte en UTF8 un
path avec des accents
'    puis que je le relis (fichier Prefs), RealBasic transforme les e´ en é etc.. Donc quand j
e compare avec un autre
'    item.AbsolutePath j'ai des différences même quand il s'agit des mêmes fichiers
' 3- J'aurais pu abandonner l'utilisation des AbsolutePath et n'utiliser que URLPath mais l
'AbsolutePath est tout de
'    même plus simple à manipuler en String, et plus clair pour l'utilisateur (dans les Pref
s etc.)
```

Fncts Note: LireInfoPlist

LireInfoPlist

J'utilisais cette fonction pour savoir si compilé en RealStudio ou Xojo,
et je l'appelais avec : If LireInfoPlist("<string>10.5.0</string>") Then ' RealStudio

```
LireInfoPlist(VmimiSyst as String) as Boolean

  Dim f_Elt as FolderItem ' J'utilise cette méthode pour savoir avec quoi je compile (RealSt
udio ou Xojo) car le système minimum change
  Dim Stream as TextInputStream
  Dim TampText as String

  f_Elt = App.ExecutableFile.Parent.Parent.Child("Info.plist")
  ' MsgBox f_Elt.AbsPath_f

  If f_Elt.Exists Then ' Ne peut pas être Nil : PasNil_Existe_Alias(f_Elt, True)
    Stream = TextInputStream.Open(f_Elt)
    Stream.Encoding = DefAppEncod
    While not Stream.eof
```

```
      TampText = Stream.ReadAll
    Wend
    Stream.Close
  Else
    TampText = "" ' Y est déjà
  End If
  ' MsgBox "Pos '" + VmimiSyst + "' = " + str(InStr(1, TampText, VmimiSyst)) + EndOfLin
e + " " + EndOfLine + TampText

  Return (InStr(1, TampText, VmimiSyst) > 0)
```

Fncts Note: MyCount

MyCount

```
  Je ne l'utilise finalement pas

  Dim TampNbre as Int32 ' Permet 2,147,483,647 éléments contenus dans le dossier

  ' Renvoie le nombre d'éléments contenu dans  CetElt  sans merder si  CetElt  n'existe p
as
  ' Et si  CetElt  est Nil retourne une NilObjection comme si on faisait .Count

  ' MsgBox "MyCount = " + str(CetElt.Count)

  If CetElt.Exists Then
    TampNbre = CetElt.Count
    #If DebugBuild Then
      If not CetElt.Directory Then
        If not(CetElt.Count = 0) Then MsgBox "MyCount déconne Tom, CetElt n'est pas un d
ossier et .Count n'est pas 0, CetElt.Count = " + str(TampNbre)
      End If
    #EndIf

  Else
    TampNbre = 0

  End If

  Return TampNbre
```

Fncts Note: NomEtExt

NomEtExt

Merdait avec Leopard

```
' Renvoi un tableau avec le displayed name (sans extension) et l'extension Remplace ma procédure  GetFichExtension
' Cette fonction est inexistante sous RealBasic

Dim NomFichAvExt, NomFichSsExt, NomFichExtention as String

NomFichAvExt = ConvertEncoding(Fich.Name, DefAppEncod) ' Ne marche pas si UTF8 pour les accents
' Ne marche pas non plus si on laisse en l'état, sans ConvertEncoding car Len compte un accent pour 2 sinon

' Compilation conditionnelle
#If TargetMacOS Then ' TargetWin32 Then ' Pour test version Windows
Dim L_NomFichSsExt as Int16

If Fich.ExtensionVisible Then ' Je préfére faire comme ça que comme sous Windows
Fich.ExtensionVisible = False
NomFichSsExt = ConvertEncoding(Fich.DisplayName, DefAppEncod) ' Ne marche pas si UTF8 pour les accents
Fich.ExtensionVisible = True
Else
NomFichSsExt = ConvertEncoding(Fich.DisplayName, DefAppEncod) ' Ne marche pas si UTF8 pour les accents
End If
L_NomFichSsExt = Len(NomFichSsExt)
If NomFichSsExt = Left(NomFichAvExt, L_NomFichSsExt) Then
NomFichExtention = Mid(NomFichAvExt, L_NomFichSsExt+1)
Else ' Nom localisé comme Bureau pour Desktop etc. -> Ancienne méthode avec le Script
NomFichExtention = GetFileExtension(Fich.AbsolutePath) ' Script qui renvoie l'extension SANS le .
If not(NomFichExtention = "") Then NomFichExtention = "." + NomFichExtention
NomFichSsExt = Left(NomFichAvExt, Len(NomFichAvExt)-Len(NomFichExtention))
End If

#Else ' Ca marcherait aussi sous Windows mais la fonction ExtensionVisible est globale sous Windows (s'applique à tous les fichiers)
```

```
#If DebugBuild Then
MsgBox "Problème Tom, tu devrais être en TargetMacOS !"
#EndIf
Dim pos, i As Int16

i = 0
Do
Pos = i
i = Instr(i+1, NomFichAvExt, ".") ' InStr( [start], source, find ) retourne0 si rien de trouvé
Loop Until i = 0
If (Pos > 0) and (Pos < Len(NomFichAvExt)) Then
NomFichSsExt = Left(NomFichAvExt, Pos-1)
NomFichExtention = Mid(NomFichAvExt, Pos)
Else
NomFichSsExt = NomFichAvExt
NomFichExtention = ""
End If

#EndIf

#If DebugBuild Then
If not(NomFichAvExt = (NomFichSsExt + NomFichExtention)) Then MsgBox "NomEtExt : F
ichier : '" + Fich.AbsolutePath + """ + EndOfLine + """ + NomFichAvExt + "' : " + str(Len(N
omFichSsExt)) + EndOfLine + """ + NomFichSsExt + "' '" + NomFichExtention + """
#EndIf

Return Array(NomFichSsExt, NomFichExtention)
```

Fncts Note: NoVolume

NoVolume

NoVolume(Fich as FolderItem) as Int16

Je n'utilise plus cette fonction car MacFSRef n'indique pas l'ID du volume mais l'ID du fi
chier

```
  Dim iNbre, NbElts, F_NoVol as Int16 ' Ne fonctionne que sous Mac car  MacFSRef
  Dim F_RefVol, iRefVol as MemoryBlock

  F_NoVol = -1 ' Reste à -1 si Fich n'est pas à Nil mais pointe sur un volume non monté
```

```
  F_RefVol = Fich.MacFSRef ' .Int64Value(0)
  ' MsgBox "F_RefVol = " + str(F_RefVol.Int64Value(0))
  If not(F_RefVol = Nil) Then ' Si Nil retourne  F_NoVol = -1
    NbElts = VolumeCount - 1
    For iNbre = 0 to NbElts
      iRefVol = Volume(iNbre).MacFSRef ' .Int64Value(0)
      If not(iRefVol = Nil) Then
        ' MsgBox "Vol(" + str(iNbre) + ") : " + Volume(iNbre).AbsolutePath + EndOfLine + "
MacFSRef = " + str(iRefVol.Int64Value(0))
        If iRefVol.Int64Value(0) = F_RefVol.Int64Value(0) Then ' Car l'égalité entre 2  Memor
yBlock  n'est pas réalisée même si égale (comme pour  FolderItem )
          If F_NoVol = -1 Then
            F_NoVol = iNbre
          Else
            MsgBox "Error ! Please contact the authors." + EndOfLine + "Volume_No, More t
han one Volume found." + EndOfLine + "F_NoVol = " + str(F_NoVol) + EndOfLine + "iNb
re = " + str(iNbre)
          End If
        End If
      Else
        MsgBox "Error ! Please contact the authors." + EndOfLine + "Vol(" + str(iNbre) + ") :
 " + Volume(iNbre).AbsolutePath + EndOfLine + "has no MacFSRef !"
      End If
    Next iNbre
  End If

  Return F_NoVol



  ' Avant que  MacVRefNum  soit Deprecated
  'F_NoVol = -1 ' Reste à -1 si Fich n'est pas à Nil mais pointe sur un volume non monté
  'F_RefVol = Fich.MacVRefNum
  '' MsgBox "F_RefVol = " + str(F_RefVol)
  'NbElts = VolumeCount - 1
  'For iNbre = 0 to NbElts
  '' MsgBox "Vol(" + str(iNbre) + ") : " + Volume(iNbre).AbsolutePath + EndOfLine + "Mac
VRefNum = " + str(Volume(iNbre).MacVRefNum)
  'If Volume(iNbre).MacVRefNum = F_RefVol Then
  'If F_NoVol = -1 Then
  'F_NoVol = iNbre
  'Else
  'MsgBox "Error ! Please contact the authors." + EndOfLine + "Volume_No, More than on
e Volume found." + EndOfLine + "F_NoVol = " + str(F_NoVol) + EndOfLine + "iNbre = "
+ str(iNbre)
```

```
'End If
'End If
'Next iNbre
```

Fncts Note: OuExclusifXOR

OuExclusifXOR

Cette fonction existe désormais

Return (DrapA or DrapB) and not(DrapA And DrapB)

Fncts Note: PasNil_Monte

PasNil_Monte

PasNil_Monte(TestElt as FolderItem) as Boolean

```
  Dim Elt_Monte as Boolean ' Cette procédure renvoie True si  TestElt  n'est pas à Nil et e
st monté (car un FolderItem sur volume qu'on vient d'éjecter n'est pas Nil)
  '  et si on fait FolderItem.AbsolutePath d'un élément d'un volume qu'on vient d'éjecter
ça gele l'appli pendant un bon moment
  ' Note : Pas ce problème avec  PasNil_Existe_Alias  car un FolderItem sur un volume qu'
on vient d'éjecter n'existe pas

  #If TargetMacOS Then
    If TestElt = Nil Then
      Elt_Monte = False
    Else
      Elt_Monte = (not(NoVolume(TestElt) = -1))
    End If

  #Else
    #If DebugBuild Then
      Break ' Crée un BreakPoint car si on est dans un Thread la MsgBox fera tout planter
      MsgBox "Problème Tom, tu devrais être en TargetMacOS !"
    #EndIf
```

```
    Elt_Monte = (not(TestElt = Nil))

  #EndIf

  Return Elt_Monte


End Module
```