Utiliser le contrôle ListView en VBA Excel

par SilkyRoad

Date de publication : 7 octobre 2006

Dernière mise à jour : 16 novembre 2007

Ce document décrit l'utilisation du contrôle ListView dans un UserForm, en VBA Excel. Le contrôle ListView fait partie des Common Controls Visual Basic 6.0 (mscomctl.ocx). Toutes les procédures ont été testées à partir d'Excel2002.

I - Description

Le contrôle ListView permet d'afficher des informations sous différentes présentations. Les données visualisées peuvent être issues de sources très diverses: La saisie d'un formulaire, le contenu de fichiers, le résultat de requêtes SQL...
Quatre modes d'affichage sont disponibles:

- Icone (Constante lvwIcon)
- Petit Icone (Constante lvwSmallIcon)
- Liste (Constante lvwList)
- Détails (Constante lvwReport)

Une fois stockées dans la ListView, les données sont facilement manipulables. Il est possible de changer le mode d'affichage, trier les colonnes, supprimer ou ajouter des lignes, appliquer une mise en forme, réexporter le contenu de la ListView... Des images et icônes peuvent être associées aux items d'une ListView.
Les chapitres suivants présentent principalement des exemples en mode "Détails".

II - Exemples d'utilisation

II-A - Alimenter une ListView

Chaque ligne d'une ListView peut être définie en 2 parties:
ListView1.ListItems(x) spécifie la ligne x et la 1ere colonne de cette ligne.
ListView1.ListItems(x).ListSubItems(y) permet de spécifier les colonnes adjacentes. Par exemple ListView1.ListItems(5).ListSubItems(1) indique la 2eme colonne dans la 5eme ligne de la ListView.
La syntaxe pour ajouter une ligne:
ListView1.ListItems.Add [Index], [Key], [Text], [Icon], [SmallIcon]
[Index]: Facultatif.
Indique le numéro de ligne ou doit être ajouté la nouvel élément. Les données sont ajoutées à la suite de la dernière ligne si l'argument n'est pas spécifié.
[Key]: Facultatif.
Attribue une clé unique qui fiabilise l'identification des lignes.
[Text]: Facultatif.
Indique le texte qui va s'afficher dans la première colonne de la ListView.
[Icon]: Facultatif.
Spécifie l'image qui doit s'afficher quand la ListView est en mode lvwIcon.
[SmallIcon]: Facultatif.
Spécifie l'image qui doit s'afficher quand la ListView est en mode lvwSmallIcon, lvwList ou lvwReport.
La syntaxe pour ajouter un sous élément (Les colonnes de droite dans la ligne spécifiée):
ListView1.ListItems(1).ListSubItems.Add [Index], [Key], [Text], [ReportIcon], [TooltipText]
1:
Spécifie le numéro de ligne dans la ListView.
[Index]: Facultatif.
Indique le numéro de colonne pour l'ajout d'une donnée. La valeur 1 correspond à la 2eme colonne d'une ListView.
[Key]: Facultatif.
Attribue une clé unique qui fiabilise l'identification des lignes.
[Text]: Facultatif.
Indique le texte qui va s'afficher dans la ListView.
[ReportIcon]: Facultatif.
Permet d'afficher un icône ou une image dans le sous élément spécifié.
[TooltipText]: Facultatif.
Permet d'ajouter une infobulle dans le sous élément spécifié.
La syntaxe pour définir les colonnes:
Pour ajouter des éléments dans les colonnes, vous devez préalablement définir leur nombre, dimensions et textes d'entête:
ListView1.ColumnHeaders.Add [Index], [Key], [Text], [Width], [Alignment], [Icon]
[Index]: Facultatif
[Key] Facultatif.
Attribue une clé unique qui fiabilise l'identification des entêtes.
[Text]: Facultatif.
Spécifie le texte qui va s'afficher dans l'entête.
[Width]: Facultatif.
Spécifie la largeur de la colonne. La valeur par défaut est de 72 points.
[Alignment]: Facultatif.
Spécifie l'alignement dans la colonne. Les constantes disponibles: lvwColumnLeft(Valeur par défaut), lvwColumnCenter, lvwColumnRight.
[Icon]: Facultatif.
Spécifie l'image qui doit s'afficher dans l'entête.
Ce premier exemple montre le principe de remplissage d'une Listview.

Vba

```
Private Sub UserForm_Initialize()
```

Vba
```
    '----- remplissage ListView-----------------------
    With ListView1
        'Définit le nombre de colonnes et Entêtes
        With .ColumnHeaders
            'Supprime les anciens entêtes
            .Clear
            'Ajoute 3 colonnes en spécifiant le nom de l'entête
            'et la largeur des colonnes
            .Add , , "Nom", 80
            .Add , , "Ville", 50
            .Add , , "Age", 50
        End With

        'Remplissage de la 1ere colonne (création de 3 lignes)
        With .ListItems
           .Add , , "Riri"
           .Add , , "Fifi"
           .Add , , "Loulou"
         End With

        'Remplissage des colonnes 2 et 3 dans la 1ere ligne
        .ListItems(1).ListSubItems.Add , , "Ville01"
        .ListItems(1).ListSubItems.Add , , 30

        'Remplissage des colonnes 2 et 3 dans la 2eme ligne
        .ListItems(2).ListSubItems.Add , , "Ville02"
        .ListItems(2).ListSubItems.Add , , 27

        'Remplissage des colonnes 2 et 3 dans la 3eme ligne
        .ListItems(3).ListSubItems.Add , , "Ville03"
        .ListItems(3).ListSubItems.Add , , 41
    End With
    '-------------------------------------------------

    'Spécifie l'affichage en mode "Détails"
    ListView1.View = lvwReport
End Sub
```

Cette macro est un exemple simplifié et il est bien entendu possible de créer des boucles afin d'optimiser le remplissage.

II-B - Lire et Modifier le contenu des lignes

Une fois affichées dans le contrôle ListView, les données peuvent être lues et modifiées par macro.
Cette procédure boucle sur l'ensemble de la ListView et transfére les informations dans une feuille de calcul.

Vba
```
Private Sub CommandButton2_Click()
    Dim i As Integer, j As Integer

    'Boucle sur toutes les lignes
    For i = 1 To ListView1.ListItems.Count
        Cells(i, 1) = ListView1.ListItems(i).Text

        'Boucle sur les colonnes
        For j = 1 To ListView1.ColumnHeaders.Count - 1
            Cells(i, j + 1) = ListView1.ListItems(i).ListSubItems(j).Text
        Next j
    Next i
End Sub
```

Les informations contenues dans la ListView peuvent facilement être modifiées:
Par exemple, changer le texte dans la 3eme colonne de la première ligne.

Vba
```
ListView1.listItems(1).listSubItems(2).Text = "Test"
```

Un autre exemple pour modifier le texte dans la 1ere colonne de la 4eme ligne.

Vba
```
ListView1.ListItems(4).Text = "Les données"
```

Les données de la 1ere colonne peuvent être modifiées manuellement dans le contrôle ListView.
Vous pouvez empêcher la modification manuelle des données en spécifiant la valeur 1 (lvwManual) dans la propriété LabelEdit.

Vba
```
ListView1.labeledit = 1
```

II-C - Utiliser les clés (Key)

Vous avez vu dans le chapitre II-A qu'il est possible d'attribuer des clés uniques aux éléments d'une ListView (Items, SubItems, Entêtes de colonnes). Quelques soient les déplacements effectués (suite à un tri par exemple), vos données peuvent être retrouvées à partir de cette clé d'identification.
Cet exemple récupère le contenu de l'Item auquel est attribué la clé "K20".
Nota:
La procédure renvoie une erreur si la clé n'existe pas dans la ListView.

Vba
```
MsgBox ListView1.ListItems("K20").Text
```

Et pour récupérer un sous élément spécifique "K21" dans l'élément "K20":

Vba
```
MsgBox ListView1.ListItems("K20").ListSubItems("K21").Text
```

Vous pouvez aussi récupérer la clé d'une ligne.
La procédure renvoie une chaine vide si aucune clé n'est attribuée.

Vba
```
MsgBox ListView1.ListItems(2).Key
```

Cette macro permet d'attribuer une clé au ListItem de la 2eme ligne. Si une clé existait dèja pour cet élément, celle ci sera écrasée. Par contre si vous essayez d'attribuer une clé déja associée à un autre élément, la procédure renvoie un message d'erreur: Ce qui est logique puisque la clé doit être unique.

Vba
```
Private Sub CommandButton2_Click()
    'Lit la clé d'origine
    MsgBox ListView1.ListItems(2).Key

    'Attribue une nouvelle clé au ListItem de la 2eme ligne
    ListView1.ListItems(2).Key = "MaNouvelleClé"
```

```vba
    'Vérification de la nouvelle clé
    MsgBox ListView1.ListItems(2).Key
End Sub
```

II-D - Supprimer des lignes

Vous avez la possibilité de supprimer des lignes précises dans la ListView.

```vba
'Supprime la 3eme ligne dans la ListView
ListView1.ListItems.Remove 3

'Un autre exemple pour supprimer une ligne à partir de sa clé ("K20")
'ListView1.ListItems.Remove "K20"
```

```vba
'Supprime la ligne active
ListView1.ListItems.Remove (ListView1.SelectedItem.Index)
```

Vous pouvez aussi effacer toutes les données d'une ListView.

```vba
ListView1.ListItems.Clear
```

II-E - La mise en forme des données

Il est possible de modifier la mise en forme de la ListView afin de personnaliser la visualisation des informations:
Cet exemple modifie la couleur du texte dans le 2eme sous élément de la 1ere ligne.

```vba
ListView1.listitems(1).ListSubItems(2).ForeColor = RGB(100, 0, 100)
```

Appliquer un format dans une "cellule de la listview

```vba
ListView1.ListItems(2).ListSubItems.Add , , Format(1234567.89, "##,##0.00")
```

Vous pouvez utiliser la propriété FullRowSelect pour surligner la ligne entière lors d'une sélection.

```vba
ListView1.FullRowSelect = True
```

Le propriété Gridlines permet d'afficher un quadrillage dans la ListView.
Cette propriété est très utile pour améliorer la lisibilité des données.

```vba
```

```vba
ListView1.Gridlines = True
```

Une option du contrôle permet d'afficher des cases à cocher dans la colonne de gauche.

```vba

Me.ListView1.CheckBoxes = True
```

Vous pouvez ensuite indiquer le statut par défaut de la CheckBox. Si vous ne spécifiez pas ce paramètre, la case ne sera pas visible tout de suite: Vous devrez cliquer sur le bord gauche de la Ligne pour faire apparaitre la CheckBox.

```vba
Dim i As Integer

For i = 1 To ListView1.ListItems.Count
    ListView1.ListItems(i).Checked = False
Next i
```

Ensuite cet exemple utilise l'évènement ItemCheck pour identifier quand une checkBox est cochée ou décochée. La mise en forme est modifiée (couleur bleue et en gras).

```vba
Private Sub ListView1_ItemCheck(ByVal Item As MSComctlLib.ListItem)
    Dim j As Integer

    If Item.Checked = True Then
            Item.ForeColor = RGB(0, 0, 255) 'Changement couleur
            Item.Bold = True 'Gras

            For j = 1 To Item.ListSubItems.Count
                Item.ListSubItems(j).ForeColor = RGB(0, 0, 255)
                Item.ListSubItems(j).Bold = True
            Next j
        Else
            Item.ForeColor = RGB(1, 0, 0) 'Changement couleur
            Item.Bold = False

            For j = 1 To Item.ListSubItems.Count
                Item.ListSubItems(j).ForeColor = RGB(1, 0, 0)
                Item.ListSubItems(j).Bold = False
            Next j
    End If
End Sub
```

Vous pouvez choisir de masquer les en-têtes de colonnes en utilisant la propriété HideColumnHeaders.

```vba

ListView1.HideColumnHeaders = True
```

La macro suivante spécifie que les données doivent être centrées dans la colonne créée.

```vba

ListView1.ColumnHeaders.Add , , "Ville", 50, lvwColumnCenter
```

La propriété AllowColumnReorder autorise le déplacement des colonnes les unes par rapport aux autres, par un glisser/déposer.

Vba
```
ListView1.AllowColumnReorder = True
```

II-F - Trier les colonnes

Les ListView possèdent une propriété Sorted. Si vous placez cette propriété à True, les Items de la première colonne seront triés par défaut. Vous pouvez spécifier la colonne de tri en indiquant une valeur dans la propriété SortKey. L'index de la première colonne est 0. La propriété SortOrder définit le sens du tri: lvwAscending (Croissant) ou lvwDescending (Décroissant).
Cet exemple utilise l'évènement ColumnClick. Le premier clic effectue un tri décroissant de la colonne cliquée. Le deuxième clic effectue un tri croissant.

Vba
```
' ------ Tri lors de la sélection d'une colonne ----------
Private Sub ListView1_ColumnClick(ByVal ColumnHeader As MSComctlLib.ColumnHeader)
    ListView1.Sorted = False
    ListView1.SortKey = ColumnHeader.Index - 1

    If ListView1.SortOrder = lvwAscending Then
        ListView1.SortOrder = lvwDescending
        Else
        ListView1.SortOrder = lvwAscending
    End If

    ListView1.Sorted = True
End Sub
```

Cet autre exemple permet de trier une colonne contenant des dates.
La procédure est adaptée **d'une solution** donnée par **Jacques Malatier** sur le site developpez.com.
La macro transforme les dates au format décimal, effectue le tri puis repasse au format DD/MM/YYYY.

Vba
```
' ------ Tri d'une colonne  contenant des Dates -------
Private Sub ListView1_ColumnClick(ByVal ColumnHeader As MSComctlLib.ColumnHeader)
   Dim i As Integer, j As Integer

    ListView1.Sorted = False
    ListView1.SortKey = ColumnHeader.Index - 1

    'Boucle sur toutes les lignes
    For i = 1 To ListView1.ListItems.Count

        'Passage des données au format décimal
        ListView1.ListItems(i).ListSubItems(ColumnHeader.Index - 1).Text = _
            CDec(CDate(ListView1.ListItems(i). _
                ListSubItems(ColumnHeader.Index - 1).Text))
    Next i

    ' ------ Application du tri -----------
    If ListView1.SortOrder = lvwAscending Then
        ListView1.SortOrder = lvwDescending
        Else
        ListView1.SortOrder = lvwAscending
    End If

    ListView1.Sorted = True
    '---------------------------------------

    'Boucle sur toutes les lignes
    For i = 1 To ListView1.ListItems.Count
        'Ensuite on revient au format DD/MM/YYYY
```

```vba
        ListView1.ListItems(i).ListSubItems(ColumnHeader.Index - 1).Text = _
            Format(CDate(ListView1.ListItems(i).ListSubItems _
                    (ColumnHeader.Index - 1).Text), "DD/MM/YYYY")
    Next i

End Sub
```

II-G - La multi sélection

Pour autoriser la multi sélection, vous devez tout d'abord passer la propriété Multiselect à True. Ensuite sélectionnez les lignes en gardant enfoncée la touche Ctrl.
Cet exemple permet de boucler sur les lignes sélectionnées.

```vba
Dim i As Integer

For i = 1 To ListView1.ListItems.Count
    'Affiche le contenu de la 1ere colonne pour chaque ligne sélectionnée
    If ListView1.ListItems(i).Selected = True Then _
                    MsgBox ListView1.ListItems(i).Text
Next
```

Remarque:
La première ligne est toujours sélectionnée par défaut lors de l'initialisation. Si vous avez besoin de la déselectionner, utilisez:

```vba
ListView1.ListItems(1).Selected = False
Set ListView1.SelectedItem = Nothing
```

Cet autre exemple permet de désélectionner toutes les lignes.

```vba
Private Sub CommandButton2_Click()
    Dim X As Integer

    For X = 1 To ListView1.ListItems.Count
        ListView1.ListItems(X).Selected = False
    Next

    Set ListView1.SelectedItem = Nothing
End Sub
```

Pour vérifier si au moins une ligne est sélectionnée dans la listview, utilisez:

```vba
Private Sub CommandButton1_Click()
    Dim LstItem As ListItem

    On Error Resume Next
    Set LstItem = ListView1.SelectedItem
    On Error GoTo 0

    If LstItem Is Nothing Then
        MsgBox "Aucune ligne n'est sélectionnée."
        Else
```

```Vba
        MsgBox "Il y a au moins une ligne de sélectionnée."
    End If

End Sub
```

II-H - Affichage dans la zone visible

La méthode EnsureVisible fait apparaître la ligne spécifiée dans la fenêtre de la ListView.
La macro suivante déplace le 50eme item dans la partie visible de la Listview.

```Vba
Listview1.ListItems(50).EnsureVisible
```

Si vous souhaitez afficher le 50eme item de la listView dans la première ligne de la partie visible (l'équivalent de TopIndex dans une ListBox), utilisez.

```Vba
Private Sub CommandButton2_Click()
    Dim i As Integer

    For i = 1 To ListView1.ListItems.Count
        ListView1.ListItems(i).EnsureVisible
        '50 est la ligne que vous souhaitez placer dans tout en haut de la zone visible
        If 50 = ListView1.GetFirstVisible.Index Then Exit For
    Next i
End Sub
```

Un autre exemple pour sélectionner et visualiser la dernière ligne d'une ListView.

```Vba
Private Sub CommandButton2_Click()
    ListView1.ListItems(ListView1.ListItems.Count).EnsureVisible
    ListView1.ListItems(ListView1.ListItems.Count).Selected = True
    ListView1.SetFocus
End Sub
```

II-I - Images et icônes

Des images ou îcones peuvent être associés aux lignes du contrôle. Les arguments Icon et SmallIcon indiquent les images qui doivent être affichées en fonction du mode de présentation défini:
SmallIcon pour les modes lvwSmallIcon, lvwList, lvwReport.
Icon pour le mode lvwIcon.
Les images sont stockées dans une imageList.
Consultez l'exemple dans le tutoriel consacré au contrôle ImageList
Les entêtes de colonne peuvent aussi contenir des îcones.
Pour cet exemple, ajoutez des contrôles ListView et ImageList dans l'UserForm.

```Vba
Private Sub UserForm_Initialize()
    Dim X As Integer

    'Supprime toutes les anciennes images de l'ImageList
    Me.ImageList1.ListImages.Clear
```

Vba

```
    'Définit la dimension des images
    Me.ImageList1.ImageHeight = 16 'Hauteur
    Me.ImageList1.ImageWidth = 16 'Largeur

    'Charge les images dans l'ImageList
    Me.ImageList1.ListImages.Add , "Im2", LoadPicture("C:\fourmiz.JPG")
    Me.ImageList1.ListImages.Add , "Im3", LoadPicture("C:\slcplappl.ico")

    '-------- Associe les images à la ListView ------------
    Set Me.ListView1.ColumnHeaderIcons = Me.ImageList1
    '------------------------------------------------------

    With ListView1
        With .ColumnHeaders
        .Clear

        '-------------------------------------------------------------------------
        'Le dernier argument attribue une image à l'entête, en utilisant les clés
        'de l'ImageList
        .Add , , "Nom", 80, , "Im2"
        .Add , , "Ville", 50, , "Im3"
        .Add , , "Age", 50
        End With
        '-------------------------------------------------------------------------

        'Remplissage 1ere colonne
        With .ListItems
           .Add , , "Riri"
           .Add , , "Fifi"
           .Add , , "Loulou"
         End With
        'Remplissage colonnes 2 et 3
        .ListItems(1).ListSubItems.Add , , "Ville01"
        .ListItems(1).ListSubItems.Add , , "30"
        .ListItems(2).ListSubItems.Add , , "Ville02"
        .ListItems(2).ListSubItems.Add , , "27"
        .ListItems(3).ListSubItems.Add , , "Ville03"
        .ListItems(3).ListSubItems.Add , , "41"
    End With

    'Spécifie l'affichage en mode Icône lors du lancement du UserForm
    ListView1.View = lvwReport
End Sub
```

De la même manière, il est aussi possible d'associer une image aux SubItems.

Vba

```
Private Sub UserForm_Initialize()
    Dim X As Integer

    'Supprime toutes les images de la liste
    Me.ImageList1.ListImages.Clear

    'Définit la dimension des images
    Me.ImageList1.ImageHeight = 16 'Hauteur
    Me.ImageList1.ImageWidth = 16 'Largeur

    'Charge les nouvelles images
    Me.ImageList1.ListImages.Add , "Im2", LoadPicture("C:\fourmiz.JPG")
    Me.ImageList1.ListImages.Add , "Im3", LoadPicture("C:\slcplappl.ico")

    '-----------------------------------------
    'Associe les images à la ListView
 Set Me.ListView1.SmallIcons = Me.ImageList1
    Set Me.ListView1.Icons = Me.ImageList1
    '-----------------------------------------
```

Vba
```
    With ListView1
        'définit le nombre de colonnes et Entêtes (titres et tailles des colonnes)
        With .ColumnHeaders
        .Clear
        .Add , , "Nom", 80
        .Add , , "Ville", 50
        .Add , , "Age", 50
        End With
        'Remplissage 1ere colonne
        With .ListItems
           .Add , , "Riri"
           .Add , , "Fifi"
           .Add , , "Loulou"
         End With
        'Remplissage colonnes 2 et 3

        '-----------------------------------------
        'Le dernier argument attribue une image aux SubItems, en utilisant les clés
        'de l'ImageList
        .ListItems(1).ListSubItems.Add , , "Ville01", "Im2"
        .ListItems(1).ListSubItems.Add , , "30", "Im3"
        .ListItems(2).ListSubItems.Add , , "Ville02", "Im2"
        .ListItems(2).ListSubItems.Add , , "27", "Im3"
        .ListItems(3).ListSubItems.Add , , "Ville03", "Im2"
        .ListItems(3).ListSubItems.Add , , "41", "Im3"
    End With

    'Spécifie l'affichage en mode Icône lors du lancement du UserForm
    ListView1.View = lvwReport
End Sub
```

Ce dernier exemple liste les fichiers d'un répertoire ainsi que certaines de leurs propriétés. L'îcone de l'executable qui ouvre ces fichiers s'affiche aussi, dans le style du volet droit de l'explorateur Windows.

Vba
```
'--------- Procédure à placer dans le module objet du UserForm ----------------

'Ajoutez:
'Un Label (LaBel1)
'Une ImageList (ImageList1)
'Une ListView (ListView1)
'Un commandButton (CommandButton1)
'
'Adaptez le répertoire cible
'
Option Explicit


Private Sub UserForm_Initialize()

    'Définit les entêtes de colonnes
    With ListView1
        With .ColumnHeaders
            .Clear 'Supprime les anciens entêtes

            'Ajout des colonnes
            .Add , , "Nom fichier", 200
            .Add , , "Taille", 40, lvwColumnRight
            .Add , , "Créé le", 60, lvwColumnCenter
            .Add , , "Modifié le", 60, lvwColumnCenter
            .Add , , "Commentaires", 200, lvwColumnLeft
        End With

    .View = lvwReport 'affichage en mode Rapport
    .Gridlines = True 'affichage d'un quadrillage
    .FullRowSelect = True 'Sélection des lignes comlètes
    End With
End Sub
```

Vba

```
Private Sub CommandButton1_Click()

    'Adaptez le répertoire cible
    ElementsRepertoire "C:\Documents and Settings\michel\Repertoire"
End Sub


Private Sub ElementsRepertoire(Chemin As String)
    Dim objShell As Object, strFileName As Object
    Dim objFolder As Object
    Dim i As Integer
    Dim Executable As String

    Set objShell = CreateObject("Shell.Application")
    Set objFolder = objShell.NameSpace(CStr(Chemin))

    Label1 = Chemin
    'Suppression des ancien éléments
    ListView1.ListItems.Clear

    'Supprime les anciens îcones et définit la taille d'affichage
    With ImageList1
        .ListImages.Clear
        .ImageWidth = 16
        .ImageHeight = 16
    End With

    'Boucle sur les fichiers du dossier cible
    '*****************************
    For Each strFileName In objFolder.Items
        'Vérifie s'il s'agit d'un sous dossier (non pris en compte)
        If strFileName.IsFolder = False Then
            i = i + 1
            'Recherche l'executable associé pour ouvrir les fichiers
            Executable = FindExecutable(Chemin & "\" & objFolder.GetDetailsOf(strFileName, 0))

            'Récupère le 1er icone de l'executable dans l'ImageList
            ImageList1.ListImages.Add , "cle" & i, GetIconFromFile(Executable, 0, False)
            'Associe la l'ImageList à la ListView
            ListView1.SmallIcons = ImageList1

                With ListView1
                    'Ajoute une ligne
                    .ListItems.Add , , objFolder.GetDetailsOf(strFileName, 0)

                    'Stocke le chemin complet qui servira à ouvrir le fichier
                    .ListItems(i).Tag = Chemin & "\" & strFileName

                    '---------------

  'Pour plus de détails sur la méthode utilisée pour récupérer les propriétés des fichiers:
                    'http://silkyroad.developpez.com/VBA/ProprietesClasseurs/#LIV-C
                    '---------------

                    'Ajoute une infobulle contenant le type de fichier et le nom de l'auteur
                    .ListItems(i).TooltipText = "Type: " & objFolder.GetDetailsOf(strFileName, 2) & _
                    "  ,  Auteur: " & objFolder.GetDetailsOf(strFileName, 9)

                    'Ajoute les sous éléments (Taille, Créée le, Modifié le, Commentaires)
                    .ListItems(i).ListSubItems.Add , , _
                                objFolder.GetDetailsOf(strFileName, 1)
                    .ListItems(i).ListSubItems.Add , , _
                            Format(objFolder.GetDetailsOf(strFileName, 4), "DD/MM/YYYY")
                    .ListItems(i).ListSubItems.Add , , _
                            Format(objFolder.GetDetailsOf(strFileName, 3), "DD/MM/YYYY")
                    .ListItems(i).ListSubItems.Add , , _
                                    objFolder.GetDetailsOf(strFileName, 14)

                    'Associe l'îcone au type de fichier
```

Vba

```
                    .ListItems(i).SmallIcon = "cle" & i
                End With
            End If

        Next strFileName
End Sub


'Ouvre le fichier par un double clic sur la ligne
Private Sub ListView1_DblClick()
    Dim leFichier As String

    'Le chemin complet est stocké dans le Tag
    leFichier = ListView1.ListItems.Item(ListView1.SelectedItem.Index).Tag
    Unload Me

    If Right(leFichier, 4) = ".xls" Then
        ThisWorkbook.FollowHyperlink leFichier
        Else
        ShellExecute 0, "open", leFichier, "", "", vbNormalFocus
    End If
End Sub
```

Vba

```
'---------- Dans un module Standard ----------------------------------------

    '*************************************************************
    'Nécessite d'activer la référence "Standard OLE Types"
    '*************************************************************

Option Explicit

Declare Function SHGetFileInfo Lib "shell32.dll" Alias "SHGetFileInfoA" _
    (ByVal pszPath As Any, ByVal dwFileAttributes As Long, psfi As SHFILEINFO, _
        ByVal cbFileInfo As Long, ByVal uFlags As Long) As Long

Public Declare Function FindExecutableA Lib "shell32.dll" _
    (ByVal lpFile As String, ByVal lpDirectory As String, ByVal lpResult As String) As Long

Public Const MAX_FILENAME_LEN = 256

Public Declare Function OleCreatePictureIndirect _
    Lib "olepro32.dll" (PicDesc As PicBmp, RefIID As GUID, _
    ByVal fPictureOwnsHandle As Long, IPic As IPicture) As Long

Public Declare Function ShellExecute Lib "shell32.dll" Alias "ShellExecuteA" _
    (ByVal hwnd As Long, ByVal lpOperation As String, ByVal lpFile As String, _
    ByVal lpParameters As String, ByVal lpDirectory As String, ByVal nShowCmd As Long) As Long

Public Type PicBmp
   Size As Long
   tType As Long
   hBmp As Long
   hPal As Long
   Reserved As Long
End Type

Public Type GUID
   Data1 As Long
   Data2 As Integer
   Data3 As Integer
   Data4(7) As Byte
End Type

Public Type SHFILEINFO
    hicon As Long
    iIcon As Long
    dwAttributes As Long
```

Vba
```
    szDisplayName As String * 260
    szTypeName As String * 80
End Type


Public Function GetIconFromFile(FileName As String, IconIndex As Long, _
        UseLargeIcon As Boolean) As IPicture
    '************************************************************
    'Necessite d'activer la reference "Standard OLE Types"
    '************************************************************
    Dim b As SHFILEINFO
    Dim retval As Long
    Dim pic As PicBmp
    Dim IPic As IPicture
    Dim IID_IDispatch As GUID

    retval = SHGetFileInfo(FileName, 0, b, Len(b), &H100)

    With IID_IDispatch
        .Data1 = &H20400
        .Data4(0) = &HC0
        .Data4(7) = &H46
    End With

    With pic
        .Size = Len(b)
        .tType = 3 'vbPicTypeIcon
        .hBmp = b.hicon
    End With

    Call OleCreatePictureIndirect(pic, IID_IDispatch, 1, IPic)
    Set GetIconFromFile = IPic
End Function


Public Function FindExecutable(S As String) As String
    'trouve quel executable ouvre le fichier cible
    Dim i As Integer
    Dim S2 As String

    S2 = String(MAX_FILENAME_LEN, 32) & Chr$(0)
    i = FindExecutableA(S & Chr$(0), vbNullString, S2)
        If i > 32 Then
        FindExecutable = Left$(S2, InStr(S2, Chr$(0)) - 1)
        Else
        FindExecutable = ""
    End If
End Function


Sub LancerUserForm()
    UserForm1.Show
End Sub
```

II-J - Identifier l'utilisation du clic droit

Il n'existe pas d'évènement spécifique **BeforeRightClick**, mais vous pouvez détourner l'action MouseDown pour identifier l'utilisation du clic droit de la souris sur une ListView.

Vba
```
Private Sub ListView1_MouseDown(ByVal Button As Integer, ByVal Shift As Integer, _
    ByVal x As stdole.OLE_XPOS_PIXELS, ByVal y As stdole.OLE_YPOS_PIXELS)

   'Button correspond au clic de la souris
```

Vba

```
        '1= clic gauche
        '2= clic droit

     'Shift: Indique quelle touche du clavier est enfoncée lors de l'évènement:
         '0 = Pas de touche enfoncée
         '1 = Touche Shift
         '2 = Touche Ctrl
         '4 = Touche Alt

    'X renvoie la position horizontale dans la listview
    'Y renvoie la position verticale

    If Button = 2 Then MsgBox "Vous avez effectué un clic droit."

End Sub
```

III - Téléchargement

Téléchargez le classeur démo.