

Gestion de la roulette de la souris dans les formulaires

par Thierry GASPERMENT (arkham46.developpez.com/)

Date de publication : 15/04/06

Dernière mise à jour : 15/04/06

Gestion de la roulette de la souris dans les formulaires.
Ajoutez un événement MouseWheel à vos formulaire et personnalisez l'action de la roulette.
Si possible, envisagez l'utilisation de la dll `MouseWheelIDVPNoReg` pour un déploiement plus aisé.

I - Introduction.....	3
II - La librairie dll fournie par Microsoft.....	4
III - Référencer la librairie MouseWheelDVP?.....	5
III-A - Enregistrement dans Access.....	5
III-B - Enregistrement manuel.....	5
III-C - Enregistrement par le code VBA.....	5
III-D - Enregistrement de la librairie sous Windows 98.....	7
IV - Exploiter le nouvel événement MouseWheel.....	8
V - Exemple : bloquer l'action de la roulette.....	9
VI - Exemple : faire défiler le formulaire au lieu des enregistrements.....	10
VI - Exemple : faire défiler les zones de listes sous le curseur.....	11
VIII - Exemple : faire défiler une zone de texte mémo particulière.....	13
IX - Les téléchargements.....	14
X - Remerciements.....	15

I - Introduction

Utiliser la roulette de la souris dans un formulaire entraîne un changement d'enregistrement qui peut être indésirable. En effet un utilisateur peut s'attendre à faire défiler le formulaire vers le bas, mais l'action sur la roulette de la souris passe à l'enregistrement suivant.

Cela peut être agaçant, voir très problématique car l'enregistrement est validé alors que l'utilisateur ne le souhaitait pas.

Pour pallier ce comportement on peut utiliser une librairie activeX qui va nous fournir un nouvel événement lors de l'action de la roulette



Envisager l'utilisation de la dll `MouseWheelDVPNoReg` pour un déploiement plus aisé.

II - La librairie dll fournie par Microsoft

Microsoft fourni **ICI** une dll qui permet d'annuler l'action de la roulette de la souris.

Malheureusement le code de cette dll n'est pas très fiable et ne fonctionne pas sur Access 2003.

Voici la liste des dysfonctionnements que l'on a pu relever :

- incompatibilité avec Access 2003 (apparemment conflit de nom avec le nouvel événement Mousewheel);
- l'utilisation de la dll dans un sous-formulaire et son parent rend le sous-formulaire inaccessible;
- la fermeture du formulaire lors d'un aperçu avant impression fait crasher l'application;
- l'utilisation de la dll dans plusieurs formulaires est mal gérée, l'événement est renvoyé dans le dernier formulaire uniquement.


--> il n'est donc pas possible de conditionner l'action de la roulette dans les autres formulaires car ils ne reçoivent pas l'événement;

Et il manque à mon goût une information utile dans l'événement :

- dans quel sens a été déplacée la roulette? Cette information est utile si on désire exécuter une action spécifique.
- C'est pour toutes ces raisons que j'ai développé une nouvelle librairie dll, baptisée pour l'occasion MouseWheelDVP.

III - Référencer la librairie MouseWheelDVP?

L'utilisation reste pratiquement identique à celle de la dll originale de Microsoft. C'est une dll activeX, ce qui signifie qu'elle doit être enregistrée et que la référence doit être sélectionnée dans Access.

 **Attention : une fois enregistrée la dll ne doit pas changer d'emplacement.**

 **Attention : Vous devez avoir les autorisations d'administration du poste pour pouvoir enregistrer la dll.**

Si vous désirez changer la dll d'emplacement, désenregistrez la, puis référez la à nouveau depuis le nouvel emplacement.

III-A - Enregistrement dans Access

Pour référencer la dll :

- allez dans l'éditeur VBA : Outils --> Macro --> Visual Basic Editor ; ou ALT+F11; ou Affichage --> Code sous Access 97
- allez dans le menu : Outils --> Références;
- cliquez sur le bouton parcourir et sélectionnez le fichier MouseWheelDVP.dll;
- > la librairie est automatiquement enregistrée dans le registre de Windows en suivant cette procédure.

III-B - Enregistrement manuel

- cliquez sur le menu Démarrer de Windows, puis Exécuter.
 - tapez la commande suivante :
 - pour enregistrer : regsvr32.exe C:\VotreChemin\MouseWheelDVP.dll
 - pour désenregistrer : regsvr32.exe /u C:\VotreChemin\MouseWheelDVP.dll
- Remarque : Il faut également cocher ou décocher la référence dans Access.

III-C - Enregistrement par le code VBA

Il est possible d'enregistrer la librairie avec des fonctions VBA.

Exemple de code à placer dans un module :

Fonctions d'enregistrement de librairies

```
Option Compare Database
Option Explicit

Private Declare Function LoadLibrary Lib "kernel32" Alias "LoadLibraryA" (ByVal lpLibFileName As String) As Long
Private Declare Function FreeLibrary Lib "kernel32" (ByVal hLibModule As Long) As Long
Private Declare Function GetLongPathName Lib "kernel32" Alias "GetLongPathNameA" _
    (ByVal lpszShortPath As String, ByVal lpszLongPath As String, _
    ByVal cchBuffer As Long) As Long

Private Declare Function DVPDllRegisterServer Lib "MouseWheelDVP" Alias "DllRegisterServer" () As Long
Private Declare Function DVPDllUnregisterServer Lib "MouseWheelDVP" Alias "DllUnregisterServer" () As Long
Private Declare Function DVPDllCanUnloadNow Lib "MouseWheelDVP" Alias "DllCanUnloadNow" () As Long

' Enregistre la librairie
Private Sub FnRegLib()
    Dim lLib As Long ' Identifiant de la librairie
    Dim lReg As Long ' Pointeur de la fonction d'enregistrement
    ' Référence la librairie dans Access
```

Fonctions d'enregistrement de librairies

```

On Error Resume Next ' si la librairie est déjà référencée on essaye quand même de l'enregistrer
Application.References.AddFromFile ApplicationPath & "MouseWheelDVP.dll"
On Error GoTo Gestion_Erreurs
' Charge la librairie
lLib = LoadLibrary(ApplicationPath & "MouseWheelDVP.dll")
If lLib = 0 Then
    MsgBox "Impossible de trouver la librairie :" & vbCrLf & ApplicationPath & "MouseWheelDVP.dll"
    Exit Sub
End If

' Enregistre la librairie (en plus du AddFromFile qui parfois n'enregistre pas correctement la librairie)
If DVPDllRegisterServer <> 0 Then
    MsgBox "Erreur lors du référencement de la librairie"
End If
Gestion_Erreurs:
If Err.Number <> 0 Then MsgBox Err.Description
' Libère la librairie
FreeLibrary lLib
End Sub

' Désenregistre la librairie
Private Sub FnUnregLib()
    Dim lLib As Long ' Identifiant de la librairie
    Dim lCanUnload As Long ' Pointeur de la fonction de test d'utilisation de la dll
    Dim lUnReg As Long ' Pointeur de la fonction d'enregistrement
    ' Supprime la référence dans Access

On Error Resume Next ' si la librairie n'est pas référencée on essaye quand même de la désenregistrer
Application.References.Remove Application.References.Item("MouseWheelDVP")
On Error GoTo Gestion_Erreurs
' Charge la librairie
lLib = LoadLibrary(ApplicationPath & "MouseWheelDVP.dll")
If lLib = 0 Then
    MsgBox "Impossible de trouver la librairie :" & vbCrLf & ApplicationPath & "MouseWheelDVP.dll"
    Exit Sub
End If
' Vérifie que la librairie n'est pas en cours d'utilisation
If DVPDllCanUnloadNow <> 0 Then
    MsgBox "Impossible de déréférencer la librairie maintenant" & _
        vbCrLf & "Quittez les formulaires utilisant la librairie"
    GoTo Gestion_Erreurs
End If
' Désenregistre la librairie (elle disparaît alors de la liste des références)
If DVPDllUnregisterServer <> 0 Then
    MsgBox "Erreur lors du référencement de la librairie"
End If
Gestion_Erreurs:
If Err.Number <> 0 Then MsgBox Err.Description
' Libère la librairie
FreeLibrary lLib
End Sub

' Récupère le chemin de l'application (chemin long)
' (pour remplacer CurrentProject.Path dans Access 97)
Public Function ApplicationPath() As String
    Dim lRet As Long
    Dim lShortPathName As String
    Dim lLongPathName As String
    lLongPathName = Space(1024)
    lShortPathName = Left(CurrentDb.Name, Len(CurrentDb.Name) - Len(Dir(CurrentDb.Name)))
    lRet = GetLongPathName(lShortPathName, lLongPathName, Len(lLongPathName))
    ApplicationPath = Left(lLongPathName, lRet)
End Function
    
```

Pour enregistrer la librairie qui se trouve dans le même répertoire que l'application :

Enregistrer la librairie

```
Call FnRegLib
```

Pour désenregistrer la librairie qui se trouve dans le même répertoire que l'application :

Désenregistrer la librairie

```
Call FnUnregLib
```

III-D - Enregistrement de la librairie sous Windows 98

Si vous êtes sous Windows 98 vous allez peut-être rencontrer quelques difficultés pour enregistrer la dll. Celle-ci étant développée en VB6, installez le runtime VB6 pour pouvoir l'enregistrer.

IV - Exploiter le nouvel événement MouseWheel

Pour pouvoir accéder au nouvel événement, il faut créer un objet.

Cet objet est une instance du module de classe inclus dans la dll.

Il déclenchera un événement MouseWheel lorsqu'on utilise la roulette de la souris.

L'événement MouseWheel a trois paramètres :

- Cancel : annule simplement l'événement : la roulette n'a plus aucun effet
- FormScroll : renvoie l'événement au formulaire : ainsi on peut faire défiler un formulaire verticalement
- Delta : déplacement de la roulette par pas de 120 : 120 correspond à un déplacement de un pas vers le haut, -120 vers le bas

Code de base à mettre dans chacun des formulaires utilisant la dll

```
Option Compare Database
Option Explicit

Private WithEvents clMouseWheel As MouseWheelDVP.CMouseWheel

Private Sub clMouseWheel_MouseWheel(Cancel As Integer, FormScroll As Integer, Delta As Long)
' Ici on va spécifier ce que l'on veut faire lors de l'action sur la roulette
End Sub

Private Sub Form_Close()
If Not (clMouseWheel Is Nothing) Then
Set clMouseWheel = Nothing
End If
End Sub

Private Sub Form_Load()
Set clMouseWheel = New MouseWheelDVP.CMouseWheel
Set clMouseWheel.Form = Me
End Sub
```

On notera qu'il ne faut pas oublier de libérer l'objet avant de quitter le formulaire.

V - Exemple : bloquer l'action de la roulette

Très simple il suffit d'affecter la valeur True au paramètre Cancel.

Bloquer l'action de la roulette

```
Option Compare Database
Option Explicit

Private WithEvents clMouseWheel As MouseWheelDVP.CMouseWheel

Private Sub clMouseWheel_MouseWheel(Cancel As Integer, FormScroll As Integer, Delta As Long)
' Annule l'action de la roulette
  Cancel = True
End Sub

Private Sub Form_Close()
  If Not (clMouseWheel Is Nothing) Then
    Set clMouseWheel = Nothing
  End If
End Sub

Private Sub Form_Load()
  Set clMouseWheel = New MouseWheelDVP.CMouseWheel
  Set clMouseWheel.Form = Me
End Sub
```

VI - Exemple : faire défiler le formulaire au lieu des enregistrements

Très simple aussi il suffit d'affecter la valeur True au paramètre FormScroll.

C'est fonctionnalité est très utile si vous avez un formulaire en mode simple qui est plus grand que la taille de l'écran.

Faire défiler le formulaire au lieu des enregistrements

```
Option Compare Database
Option Explicit

Private WithEvents clMouseWheel As MouseWheelDVP.CMouseWheel

Private Sub clMouseWheel_MouseWheel(Cancel As Integer, FormScroll As Integer, Delta As Long)
' Annule l'action de la roulette
  FormScroll = True
End Sub

Private Sub Form_Close()
  If Not (clMouseWheel Is Nothing) Then
    Set clMouseWheel = Nothing
  End If
End Sub

Private Sub Form_Load()
  Set clMouseWheel = New MouseWheelDVP.CMouseWheel
  Set clMouseWheel.Form = Me
End Sub
```

VI - Exemple : faire défiler les zones de listes sous le curseur

Ca se complique un peu plus...

On va :

- annuler le défilement standard avec Cancel = True
- utiliser l'API GetCursorPos pour récupérer la position du curseur sur l'écran.
- utiliser l'API WindowFromPoint pour récupérer l'identifiant de la fenêtre sous le curseur.
- utiliser l'API GetClassName pour lire la classe de cette fenêtre
- vérifier que la fenêtre survolée est une zone de liste (classe oGrid)
- envoyer un événement à la liste pour la faire défiler

Remarque : Dans certaines versions d'Access, si une zone de liste (simple, donc non déroulante) a le focus alors l'événement MouseWheel n'est pas envoyé vers le formulaire, et donc le défilement de la liste ne fonctionne pas. (En tout cas ça fonctionne avec Access 2003 mais pas avec Access 2000)

Faire défiler les zones de listes sous le curseur

```

Option Compare Database
Option Explicit

Private WithEvents clMouseWheel As MouseWheelDVP.cMouseWheel

' Déclaration d'API
Private Declare Function SendMessage Lib "user32" Alias "SendMessageA" (ByVal hWnd As Long, ByVal
    wParam As Long, ByVal lParam As Long, lParam As Any) As Long
Private Declare Function GetCursorPos Lib "user32" (lpPoint As POINTAPI) As Long
Private Declare Function WindowFromPoint Lib "user32" (ByVal xPoint As Long, _
    ByVal yPoint As Long) As Long
Private Declare Function GetClassName Lib "user32" Alias "GetClassNameA" (ByVal hWnd As Long, ByVal
    lpClassName As String, ByVal nMaxCount As Long) As Long
Private Type POINTAPI
    X As Long
    Y As Long
End Type
' Constantes pour le type de scrolling
Private Const WM_VSCROLL = &H115
' Constantes pour les commandes de scrolling
Private Const SB_LINEUP = 0
Private Const SB_LINEDOWN = 1

Private Sub clMouseWheel_MouseWheel(Cancel As Integer, FormScroll As Integer, Delta As Long)
    ' Pour annuler l'événement
    Cancel = True
    ' Position du curseur
    Dim lpt As POINTAPI
    ' Fenêtre sous le curseur
    Dim lhWnd As Long
    ' Code retour de l'API
    Dim lRet As Long
    ' Classe de la fenêtre
    Dim lClassName As String
    ' Récupère la position du curseur
    Call GetCursorPos(lpt)
    ' Récupère l'identifiant de la fenêtre sous le curseur
    lhWnd = WindowFromPoint(lpt.X, lpt.Y)
    ' Recherche la classe de la fenêtre
    lClassName = Space(255)
    lRet = GetClassName(lhWnd, lClassName, 255)
    lClassName = Left(lClassName, lRet)
    ' Si la classe est oGrid alors le curseur est sur une zone de liste
    If lClassName = "oGrid" Then
    If Delta < 0 Then
        ' Déplacement vers le bas
        SendMessage lhWnd, WM_VSCROLL, SB_LINEDOWN, 0&
    Else
        ' Déplacement vers le haut
        SendMessage lhWnd, WM_VSCROLL, SB_LINEUP, 0&
    End If
    End If

```

Faire défiler les zones de listes sous le curseur

```
End If
End Sub

Private Sub Form_Close()
    If Not (clMouseWheel Is Nothing) Then
        Set clMouseWheel = Nothing
    End If
End Sub

Private Sub Form_Load()
    Set clMouseWheel = New MouseWheelDVP.cMouseWheel
    Set clMouseWheel.Form = Me
End Sub
```

VIII - Exemple : faire défiler une zone de texte mémo particulière

On utilise les API pour cet exemple également

On va :

- annuler le défilement standard avec `Cancel = True`
- vérifier que la zone de texte voulue est active (remplacer `NomDeLaZoneDeTexte` par le nom de votre zone de texte).
- utiliser l'API `GetFocus` pour récupérer l'identifiant de la zone de texte.
- envoyer un événement à la zone de texte pour la faire défiler

faire défiler une zone de texte mémo particulière

```

Option Compare Database
Option Explicit

Private WithEvents clMouseWheel As MouseWheelDVP.cMouseWheel

' Déclaration d'API
Private Declare Function SendMessage Lib "user32" Alias "SendMessageA" (ByVal hWnd As Long, ByVal
    wParam As Long, ByVal lParam As Any) As Long
Private Declare Function GetFocus Lib "user32" () As Long
' Constantes pour le type de scrolling
Private Const WM_VSCROLL = &H115
' Constantes pour les commandes de scrolling
Private Const SB_LINEUP = 0
Private Const SB_LINEDOWN = 1

Private Sub clMouseWheel_MouseWheel(Cancel As Integer, FormScroll As Integer, Delta As Long)
    ' Pour annuler l'événement
    Cancel = True
    ' Fenêtre qui a le focus
    Dim lhWnd As Long
    ' Vérifie qu'on est sur le contrôle que l'on veut faire défiler
    On Error Resume Next
    If Screen.ActiveControl.Name <> "NomDeLaZoneDeTexte" Then Exit Sub
    If Err.Number <> 0 Then Exit Sub
    On Error GoTo 0
    ' Récupère l'identifiant de la fenêtre qui a le focus
    lhWnd = GetFocus
    If Delta < 0 Then
        ' Déplacement vers le bas
        SendMessage lhWnd, WM_VSCROLL, SB_LINEDOWN, 0&
    Else
        ' Déplacement vers le haut
        SendMessage lhWnd, WM_VSCROLL, SB_LINEUP, 0&
    End If
End Sub

Private Sub Form_Close()
    If Not (clMouseWheel Is Nothing) Then
        Set clMouseWheel = Nothing
    End If
End Sub

Private Sub Form_Load()
    Set clMouseWheel = New MouseWheelDVP.cMouseWheel
    Set clMouseWheel.Form = Me
End Sub
    
```

IX - Les téléchargements

Télécharger la dll (7Ko) ([HTTP](#))

Télécharger le code source VB6 (11Ko) ([HTTP](#))

Télécharger la base de test (Access 97) (21Ko) ([HTTP](#))

X - Remerciements

Merci à Tofalu, Gaël Donat et FabienN pour leurs tests.