RegFreeX.dll - Appel de composants ActiveX sans enregistrement préalable

Par omen999

Date de publication : 3 novembre 2011

Cette bibliothèque DLL, dont la fonction est de permettre l'instanciation de classes de composants COM sans inscription dans la base de registre, a été écrite spécifiquement pour être appelée au moyen du composant **DynamicWrapperX** dans un langage comme VBScript ou JScript.

Vous pouvez également poser vos questions et laisser vos commentaires et suggestions sur le forum : **Commentez**

1 - Introduction....3
2 - Présentation de RegFreeX....3
2-1 - CreateObjectEx....3
2-2 - CallByNameEx....3
2-3 - CArray....3
3 - Référence....4
3-1 - CreateObjectEx(ClassName, FileName, ErrorName, Obj)....4
3-2 - CallByNameEx(Obj, Name, iCallType, pArgs, ReturnValue)....4
3-3 - CArray(NewArray, iBound)....4
4 - Exemple de script - TestADO.vbs....5
5 - Liens....6
6 - Remerciements....6

Les sources présentées sur cette page sont libres de droits et vous pouvez les utiliser à votre convenance. Par contre, la page de présentation constitue une œuvre intellectuelle protégée par les droits d'auteur. Copyright ® 2011 omen999. Aucune reproduction, même partielle, ne peut être faite de ce site et de l'ensemble de son contenu : textes, documents, images, etc. sans l'autorisation expresse de l'auteur. Sinon vous encourez selon la loi jusqu'à trois ans de prison et jusqu'à 300 000 € de dommages et intérêts.

1 - Introduction

La recherche de la portabilité maximum des scripts écrits en VBScript ou JScript conduit naturellement à s'affranchir des contraintes d'inscription des composants ActiveX dans la base de registre. S'il existe une solution officielle depuis Windows XP SP2 au moyen des *SxS Assemblies* (voir à ce sujet cet **article**), une autre voie a été explorée depuis longtemps par des concepteurs soucieux de rendre portables leurs projets. Cette voie est celle de l'émulation de la fonction *CoCreateInstance* de l'API COM.

Il existe déjà un wrapper DLL dénommé **DirectCOM** mais dont les conventions d'appel de paramètres ne sont pas compatibles avec **DynamicWapperX**. **RegFreeX** a donc été écrit pour y remédier mais également pour repousser certaines limitations constatées dans les *wrappers* existants.

En effet, comme il s'agit d'une DLL standard, VBScript/JScript devront utiliser le composant **DynamicWrapperX** pour appeler ses fonctions. Ce composant et sa documentation sont disponibles **ici**.

2 - Présentation de RegFreeX

Ce *wrapper* se présente sous la forme d'une bibliothèque DLL classique qui exporte trois fonctions.

2-1 - CreateObjectEx

La principale, *CreateObjectEx*, a pour effet d'instancier une variable objet à partir d'une classe exposée par le composant. À la différence d'une fonction *CreateObject* classique, l'objet instancié n'est pas renvoyé par la fonction mais par un paramètre en sortie. La valeur renvoyée renseigne sur le statut de l'opération (succès ou codes d'erreur).

2-2 - CallByNameEx

La deuxième fonction *CallByNameEx* est une particularité de **RegFreeX** et nécessite quelques explications.

Lorsqu'un objet est instancié en liaison différée comme en l'espèce, l'appel d'une méthode - ou d'une propriété ce qui revient au même - conduit le code, chargé de retrouver l'ID de celle-ci, à appeler l'interface *IProvideClassInfo* pour déterminer l'interface par défaut de la classe. Quand *IProvideClassInfo* n'est pas supportée par le composant (elle est facultative), le code jette un œil dans la base de registre pour trouver l'information. Si le composant n'est pas enregistré, l'appel échouera.

Volontiers cachotier, Microsoft n'a pas jugé utile d'attirer l'attention des concepteurs sur cette utilisation particulière de l'interface et il est fréquent que des composants ActiveX non-visuels - notamment ceux écrits en C++ - omettent de l'exposer ce qui les rend inutilisables en mode non-enregistré.

Il existe toutefois un "plan B" qui consiste à coder entièrement l'appel de méthode afin d'éviter tout accès au registre. C'est le rôle de la fonction *CallByNameEx*.

Dans le but d'identifier les composants "incomplets", la fonction *CreateObjectEx* renvoie un code d'instanciation spécifique (voir l'exemple *infra*).

2-3 - CArray

La dernière fonction, *CArray*, est purement utilitaire et pallie l'impossibilité de transmettre directement à **DynamicWrapperX** une variable tableau. Il était donc nécessaire de définir une fonction capable de renvoyer l'adresse de cette variable afin de permettre ce transfert par référence.

Les sources présentées sur cette page sont libres de droits et vous pouvez les utiliser à votre convenance. Par contre, la page de présentation constitue une œuvre intellectuelle protégée par les droits d'auteur. Copyright ® 2011 omen999. Aucune reproduction, même partielle, ne peut être faite de ce site et de l'ensemble de son contenu : textes, documents, images, etc. sans l'autorisation expresse de l'auteur. Sinon vous encourez selon la loi jusqu'à trois ans de prison et jusqu'à 300 000 € de dommages et intérêts.
https://omen999.developpez.com/tutoriels/vbs/RegFreeX/

3 - Référence

Tous les paramètres sont obligatoires et sont mentionnés en gras.

3-1 - CreateObjectEx(ClassName, FileName, ErrorName, Obj)

Instancie une variable objet à partir d'une classe exposée par un composant ActiveX.
ClassName : nom de la classe à instancier ;
FileName : nom du fichier du composant ActiveX ;
ErrorName : chaine décrivant la nature de l'erreur - vide si pas d'erreur ;
Obj : instance de l'objet *IDispatch* renvoyé ;
Valeur retournée :
-2 : objet instancié mais nécessite la fonction *CallByNameEx* (pas de *IProvideClassInfo*) ;
-1 : objet instancié et utilisable directement ;
0 : fichier DLL absent ;
1 : fonction '*DllGetClassObject*' introuvable ;
2 : ressource *typelib* introuvable ;
3 : nom de classe inconnu ;
4 : pas d'interface par défaut.

3-2 - CallByNameEx(Obj, Name, iCallType, pArgs, ReturnValue)

Appel d'une méthode ou gestion d'une propriété d'une instance de classe issue d'un composant dépourvu d'interface *IProvideClassInfo*.
Obj : objet *IDispatch* dont la méthode ou la propriété doit être appelée ;
Name : nom méthode ou propriété ;
iCallType : définit le type d'appel :
1 : méthode ;
2 : lecture valeur propriété ;
3 : écriture valeur propriété ;
4 : écriture valeur propriété objet ;
pArgs : pointeur du tableau variant des paramètres éventuels de la méthode ou de la propriété à modifier ;
ReturnValue : valeur retournée par cet appel - le type dépend de la méthode ou de la propriété ;
Valeur retournée :
-1 : pas d'erreur OK ;
0 : interface *ITypeInfo* non trouvée - l'objet n'a pas été instancié par *CreateObjectEx* ;
1 : méthode ou propriété inconnue ;
2 : mauvais type de paramètre ;
3 : mauvais nombre de paramètres ou variable tableau des paramètres absente.
n : une valeur quelconque d'erreur dépendant de la méthode/propriété appelée (généralement de la forme h800x000x).

3-3 - CArray(NewArray, iBound)

Crée une variable tableau de dimension donnée et renvoie son pointeur
(contourne une limitation de **DynamicWrapperX** qui n'accepte pas les paramètres tableau).
NewArray : variable tableau à créer ;
iBound : nombre d'items du tableau ;
Valeur retournée :
adresse de la variable tableau créée.

Les sources présentées sur cette page sont libres de droits et vous pouvez les utiliser à votre convenance. Par contre, la page de présentation constitue une œuvre intellectuelle protégée par les droits d'auteur. Copyright ® 2011 omen999. Aucune reproduction, même partielle, ne peut être faite de ce site et de l'ensemble de son contenu : textes, documents, images, etc. sans l'autorisation expresse de l'auteur. Sinon vous encourez selon la loi jusqu'à trois ans de prison et jusqu'à 300 000 € de dommages et intérêts.

4 - Exemple de script - TestADO.vbs

Le composant MS ActiveX Data Objects ne peut être géré qu'au moyen de la fonction *CallByNameEx*. L'exemple qui suit ouvre un fichier texte dénommé "rfc6365.txt", y ajoute en dernière ligne la phrase "*Cette ligne a été ajoutée par omen999*" et sauvegarde le fichier sous le nom "rfc6365_.txt".

TestADO.vbs

```
'*************************************************************************
'* outil de test RegFreeX wrapper avec MS ActiveX Data Objects (ADO)
'* msado15.dll doit être désenregistré de la base de registre avant le test...
'* DynamicWrapperX requis - http://www.script-coding.com/dynwrapx_eng.html
'* novembre 2011  omen999 - http://omen999.developpez.com
'*************************************************************************
Option Explicit

Const _
vbMethod=1,_
vbGet=2, _
vbSet=4, _
vbLet=8

Dim oWrap
Dim oShell
Dim sErr
Dim oStream
Dim iRep
Dim vResult
Dim aArgs
Dim pArgs
Dim vSize

'initiation DynamicWrapperX
On Error Resume Next
Set oWrap = CreateObject("DynamicWrapperX")
If oWrap Is Nothing Then
  Set oShell = CreateObject("WScript.Shell")
  oShell.Run "wscript.exe " & Chr(34) & WScript.ScriptFullName & Chr(34)
  WScript.Quit
End If
On Error GoTo 0
oWrap.Register "RegFreeX.dll","CreateObjectEx","i=wwPP","r=l"
oWrap.Register "RegFreeX.dll","CallByNameEx","i=pwupP","r=l"
oWrap.Register "RegFreeX.dll","CArray","i=Pl","r=p"

'instanciation objet Stream
iRep=oWrap.CreateObjectEx("Stream","msado15.dll",sErr,oStream)
If iRep > -1 Then
  MsgBox sErr,16,"Erreur RegFreeX"
ElseIf iRep = -2 Then ' le composant ne supporte pas IProvideClassInfo
  iRep = oWrap.CallByNameEx(oStream,"Open",vbMethod,0,vResult)
  'création de la variable tableau paramètres avec un seul item
  pArgs=oWrap.CArray(aArgs,1)
  'définition du charset souhaité
  aArgs(0)="Windows-1252"
  iRep = oWrap.CallByNameEx(oStream,"Charset",vbSet,pArgs,vResult)
  aArgs(0)="rfc6365.txt"
  iRep = oWrap.CallByNameEx(oStream,"LoadFromFile",vbMethod,pArgs,vResult)
  iRep = oWrap.CallByNameEx(oStream,"Size",vbGet,0,vSize)
  aArgs(0)=vSize
  'le curseur est placé à la fin du stream
  iRep = oWrap.CallByNameEx(oStream,"Position",vbSet,pArgs,vResult)
  aArgs(0)="Cette ligne a été ajoutée par omen999"
  iRep = oWrap.CallByNameEx(oStream,"WriteText",vbMethod,pArgs,vResult)
  aArgs(0)="rfc6365_.txt"
  iRep = oWrap.CallByNameEx(oStream,"SaveToFile",vbMethod,pArgs,vResult)
  If iRep = -1 Then
    MsgBox "Fichier généré avec succès",64,"Test RegFreeX"
  Else
    If iRep < -1 Then iRep = Hex(iRep)
```


Les sources présentées sur cette page sont libres de droits et vous pouvez les utiliser à votre convenance. Par contre, la page de présentation constitue une œuvre intellectuelle protégée par les droits d'auteur. Copyright ® 2011 omen999. Aucune reproduction, même partielle, ne peut être faite de ce site et de l'ensemble de son contenu : textes, documents, images, etc. sans l'autorisation expresse de l'auteur. Sinon vous encourez selon la loi jusqu'à trois ans de prison et jusqu'à 300 000 € de dommages et intérêts.

TestADO.vbs
```
    MsgBox "Erreur code : " & iRep,16,"Erreur RegFreeX"
  End If
  iRep = oWrap.CallByNameEx(oStream,"Close",vbMethod,0,vResult)
End If
```

5 - Liens

Fichier archive contenant les fichiers suivants : RegFreeX.dll, TestADO.vbs et rfc6365.txt.
L'utilisation des ActiveX sans inscription (SxS assemblies) 1ère partie
L'utilisation des ActiveX sans inscription (SxS assemblies) 2ème partie

6 - Remerciements

Je tiens à remercier **bbil** et **Claude LELOUP** pour leur relecture attentive et leurs observations.

Les sources présentées sur cette page sont libres de droits et vous pouvez les utiliser à votre convenance. Par contre, la page de présentation constitue une œuvre intellectuelle protégée par les droits d'auteur. Copyright ® 2011 omen999. Aucune reproduction, même partielle, ne peut être faite de ce site et de l'ensemble de son contenu : textes, documents, images, etc. sans l'autorisation expresse de l'auteur. Sinon vous encourez selon la loi jusqu'à trois ans de prison et jusqu'à 300 000 € de dommages et intérêts.