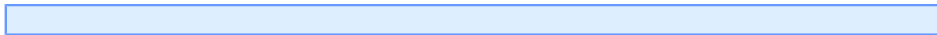


Tutoriel 22 : Protection de nos réalisations sous VB6

par Gilbert Miralles (gilmir.developpez.com)

Date de publication : Lundi 18 mars 2003

Dernière mise à jour : Lundi 4 février 2008



I - Introduction

II - Application

III - Liens

Prochain tutoriel :

I - Introduction

Le soucis de la plupart des programmeurs est de protéger ses réalisations.

Je n'ai pas la prétention de connaître toutes les astuces liées à la protection des programmes et il serait fastidieux de les énumérer tous A

ussi nous allons étudier une protection simple mais qui découragera la plupart des adeptes du tournevis et de la combine

Pour construire notre réalisation, nous avons besoin de données "référence" prise sur le computer du client qui va installer notre réalisation sur sa machine.

Le meilleur moyen est de récupérer le nom de l'utilisateur et le nom de la machine, or j'ai pu constater que certains appareils ne comportaient pas de nom, soit que l'utilisateur ne l'avait pas rentré ou qu'il avait pu être effacé malencontreusement.

Aussi nous allons utiliser le nom de l'appareil en question et pour faire simple nous nous contenterons de programmer 6 caractères tout en sachant qu'une fois le principe maîtrisé, nous pourrons modifier les paramètres utilisés.

II - Application

Après avoir lancé Visual Basic 6.0 et choisi un projet EXE standard et sauvegardé sous le nom de "Protect.vbp", nous écrivons les quelques lignes de code suivantes dans [Général] [Déclaration] d'un module que nous intitule : **general.bas**

```
Option Explicit
'Déclarations pour récupérer le nom de l'utilisateur et de l'ordinateur
Declare Function GetComputerName Lib "kernel32" Alias "GetComputerNameA" (ByVal lpBuffer As String,
nSize As Long) As Long
Declare Function GetUserName Lib "advapi32.dll" Alias "GetUserNameA" (ByVal lpBuffer As String,
nSize As Long) As Long
'Déclaration des variables globales pour les instructions de messageries
Global Title As String
Global msg As String
Global Reponse As Integer
```

Dans ce même module nous utilisons la routine de centrage de feuille :


```
Sub CenterForm(frm As Form)
'Centrage de la feuille
frm.Top = Screen.Height / 2 - frm.Height / 2
frm.Left = Screen.Width / 2 - frm.Width / 2
End Sub
```

Nous allons fabriquer une feuille qui va nous permettre d'effectuer tous les essais nécessaires afin que vous compreniez le système élaboré par votre serviteur.

Nous allons lui laisser le nom de "Form1"

Dans la procédure Form nous écrirons :

- **CenterForm Me** 'Appel de la procédure de centrage de la feuille
- **NomPc** 'Appel de la procédure qui va nous restituer le nom du PC
- **NomUtilisateur** 'Appel de la procédure qui va nous restituer le nom de l'utilisateur

 *Nous n'utiliserons pas "NomUtilisateur" mais, dans la foulée je vous donne l'écriture du code de cette procédure d'autant que même si dans votre application cela ne sert à rien, ça enjolive et je dirais même que cela fait professionnel.*

Pour pouvoir récupérer les données récoltées, nous insérerons deux étiquettes que nous nommerons :

- **PropriétéName** = lblUserCode1 'Etiquette qui va nous restituer le nom de l'ordinateur
- **PropriétéName** = lblUserName1 'Etiquette qui va nous afficher le nom de l'utilisateur

Passons aux choses sérieuses et récupérons le nom du PC de notre utilisateur :

```
Public Function NomPc() As String
Dim lngLongueur As Long
Dim strTampon As String
Dim intR As Integer
```

```

strTampon = Space(255)
lngLongueur = Len(strTampon)
intR = GetComputerName(strTampon, lngLongueur)
NomPc = Left(strTampon, lngLongueur)
Protect!lblUserCode1.Caption = NomPc
End Function

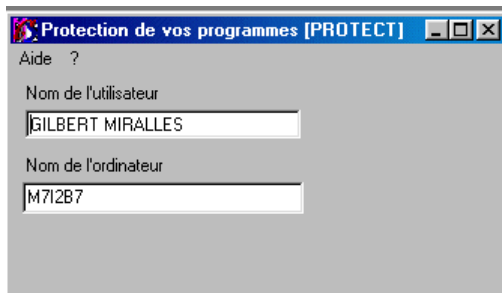
```

Et maintenant le nom de l'utilisateur :

```

Public Function NomUtilisateur() As String
Dim lngLongueur As Long
Dim strTampon As String
Dim intR As Integer
strTampon = Space(256)
lngLongueur = Len(strTampon)
intR = GetUserName(strTampon, lngLongueur)
NomUtilisateur = Left(strTampon, lngLongueur - 1)
Protect!lblUserName1.Caption = UCase$(NomUtilisateur)
End Function

```



Nous constatons que nous avons bien en retour de notre procédure les données recherchées, à savoir :

- le nom de l'ordinateur (M712B7)
- le nom de l'utilisateur.

Notez que si vous essayer ce programme sur votre ordinateur les données que vous allez récupérer seront différentes.

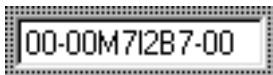
Si vous cliquez sur le point d'interrogation, vous allez ouvrir la fenêtre "A propos de..." et vous constaterez que j'ai fait une réservation dans une étiquette ou va venir se loger le numéro de série que nous allons engendrer avec les nouvelles instructions que je vous propose de saisir.

Sans vouloir faire compliqué, mais en corsant quant même le procédé pour freiner l'enthousiasme des petits futés, nous allons essayer d'embrouiller le client par une suite de caractères qui ne serviront, comme un leurre, qu'à noyer le poisson.


La chaîne de caractères que nous avons eu en retour comprend 6 caractères et, comme cela semble simpliste, nous

allons en rajouter 6 autres, ce qui correspondra à un codage ressemblant à ceci !  soit dans

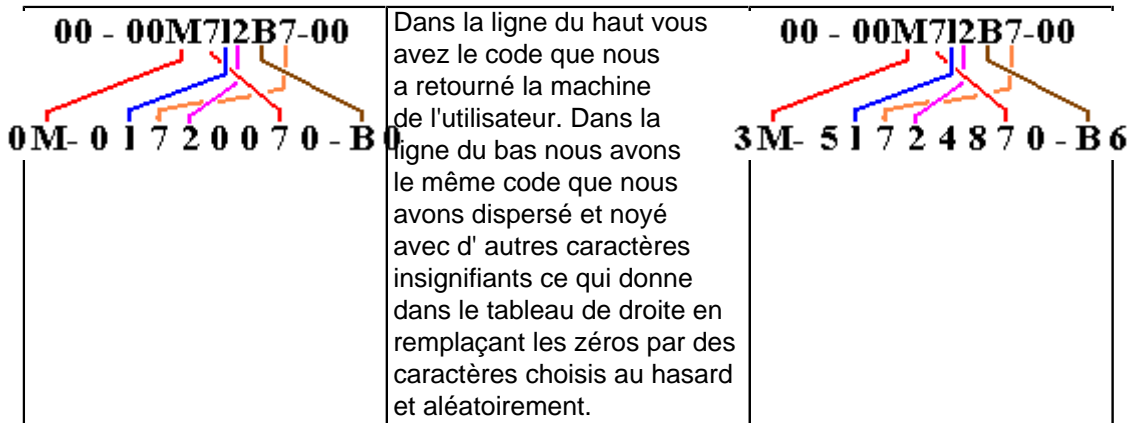
l'ordre :



Mais nous constatons que dans le cas présent, nos caractères zéro ne servent à rien aussi nous allons les modifier

et mettre à la place des caractères plus significatif comme par exemple : , et voila ce que nous recherchons, un numéro de série qui comprend des caractères significatifs inérants à la machine de l'utilisateur.

Avant de passer à l'écriture du code, devant planifier notre construction, essayons de mettre en place la technique que nous allons employer pour dispatcher nos caractères afin de compliquer la découverte du code.



Nous savons maintenant que le code qui doit s'inscrire dans le fichier "A propos de..." et dans l'étiquette prévue à cet effet correspond à : 3M-51724870-B6 (Attention car le caractère après le chiffre 5 est un "L" en minuscule et que cela peut parfois engendrer des erreurs, et il vaut mieux effectuer un copier coller plutôt que d'inscrire le code manuellement.

Ecriture du code pour récupérer nos caractères et les utiliser dans cette réalisation

Recherchons dans l'index de VB quelle est la commande qui permet d'initialiser des nombres aléatoires(pour remplacer les zéros).

Nous trouvons l'instruction **Randomize** avec la syntaxe suivante : Randomize nombre%

Dans la feuille Form nous saisissons la commande Randomize et nous utiliserons la commande suivante pour générer des caractères aléatoires :

```
Protect.ctlLabel1(0).Caption = Int(Rnd * 9)
```

"Label1(0)" ainsi que les 11 autres "Label" servent à stocker individuellement les caractères de notre numéro de série.)

Les "Label" seront disposés côte à côte et numérotés de Label1(0) à Label1(13) ils seront indexés, (vous l' avez compris puisque nous avons des chiffres entre parenthèses)

Les Label devant afficher un chiffre aléatoire sont :

Label1(0) - Label1(3) - Label1(7) - Label1(8) - Label1(10) - Label1(13)

Les "Label1(2)" et "Label1(11)" resteront statiques et afficherons le petit tiret(-)

Les autres "Label" afficheront les caractères retournés par le système.

Voici les codes à utiliser pour les "Label" devant générer des caractères aléatoires :

```
Protect.Label1(0).Caption = Int(Rnd * 9)
Protect.Label1(3).Caption = Int(Rnd * 9)
Protect.Label1(7).Caption = Int(Rnd * 9)
Protect.Label1(8).Caption = Int(Rnd * 9)
Protect.Label1(10).Caption = Int(Rnd * 9)
Protect.Label1(13).Caption = Int(Rnd * 9)
```

Voici les codes à utiliser pour les "Label" devant générer des caractères statiques :

```
Protect.Label1(2).Caption = "-"
Protect.Label1(11).Caption = "-"
```

Voici les codes à utiliser pour les "Label" devant générer les caractères du code machine :

Que recherchons nous dans cette opération?

Tout simplement à récupérer les caractères individuellement qui se trouvent dans la chaîne de caractères qui affiche le nom de la machine.

Interrogeons Visual Basic et recherchons quelle est l'instruction qui permet de récupérer un caractère dans une chaîne.

Nous trouvons la fonction "Mid, ou Mid\$" (Fonction ou Instruction) avec comme descriptif : Renvoie une partie d'une chaîne de caractères ou remplace une partie d'une chaîne de caractères par une autre chaîne de caractères. La syntaxe est, 1 - resultat\$ = Mid\$(chaîne de caractères\$, position% 'longueur%) 2 - Mid\$(chaîne de caractères\$, position%, longueur&) = resultats\$ Remarque : La fonction Mid renvoie une chaîne de caractères Variant, la fonction Mid\$ renvoie une chaîne de caractères au format String. Les instructions Mid et Mid\$ fonctionnent de manière identique. Et nous trouvons les fonctions "Left, Left\$" et Right, Right\$ Left\$ = Retourne un nombre de caractères définis dans la partie gauche d'une chaîne. Right\$ = Retourne un nombre de caractères définis dans la partie droite d'une chaîne. Nous avons maintenant toutes les bases nécessaires à l'écriture du code devant nous permettre de récupérer les données que nous avons besoin.

Saisissons dans la feuille 'Form' l'instruction suivante :

```
'Initialisation des fonctions et instructions usuelles
InitPCNumberSerial          'Lancer l'initialisation de InitPCNumberSérial
```

Et voici la procédure correspondante

```
Sub InitPCNumberSerial()
'Déclaration des variables
Dim X As String
Dim Number1 As String
Dim Number2 As String
'
'RECUPERATION DU NOM DE L'ORDINATEUR.
Dim lngRep As Long, sComputerName As String
sComputerName = Space$(16)
lngRep = GetComputerName(sComputerName, Len(sComputerName))
Form1.ctlLabel1.Caption = sComputerName
'Vous avez ainsi dans sComputerName, le nom de l'ordinateur
'ou le programme est installé.
'GENERATION DU USERNAME
Form1.Label1(0).Caption = Int(Rnd * 9)
Form1.Label1(3).Caption = Int(Rnd * 9)
Form1.Label1(7).Caption = Int(Rnd * 9)
Form1.Label1(8).Caption = Int(Rnd * 9)
Form1.Label1(10).Caption = Int(Rnd * 9)
```

Et voici la procédure correspondante

```
Form1.Label1(13).Caption = Int(Rnd * 9)
'
Form1.Label1(2).Caption = "-"
Form1.Label1(11).Caption = "-"

'Nous savons que le nom du Pc se trouve dans la chaîne : (Form1.lblUserCode1.Text)
'Nous écrivons :
X$ = Form1.lblUserCode1.Text
'
Form1.Label1(1).Caption = Mid$(X$, 1, 1)
Form1.Label1(4).Caption = Mid$(X$, 3, 1)
Form1.Label1(5).Caption = Mid$(X$, 6, 1)
Form1.Label1(6).Caption = Mid$(X$, 4, 1)
Form1.Label1(9).Caption = Mid$(X$, 2, 1)
Form1.Label1(12).Caption = Mid$(X$, 5, 1)

'Définissons notre SerialNumberCode
NumberCode1 = Label1(1) + Label1(4) + Label1(5) + Label1(6) + Label1(9) + Label1(12)

'Il ne nous reste plus qu'à saisir le code qui va concaténer et rassembler le contenu
'des "Label" pour en faire une chaîne de caractère qui sera affichée dans l'étiquette NumberCode1
End Sub
```

Vous constatez que j'ai utilisé parfois le préfixe de "Form1" ou "Protect", je vous précise que celui-ci n'est pas obligatoire si vous saisissez l'instruction dans la feuille dans laquelle elle va être utilisée.

Malgré ce, saisissez le préfixe avec le nom que vous avez donné à votre feuille.

Si celle-ci se nomme "Form1" vous devez écrire :

```
Form1.Label1(1).Caption = Mid$(X$, 1, 1)
```

Si elle se nomme "Protect" vous devez "écrire :

```
Protect!Label1(1).Caption = Mid$(X$, 1, 1)
```

Mais...

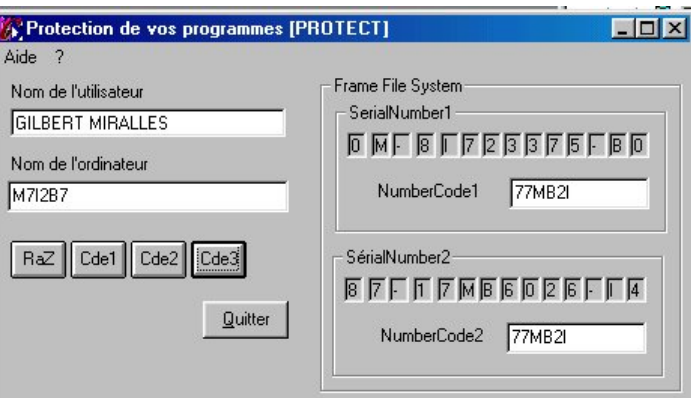
```
Label1(1).Caption = Mid$(X$, 1, 1)
```

fonctionnera également

Le préfixe serait obligatoire si vous aviez mis la procédure dans le module.

Cette petite explication à titre pédagogique, pour vous aider à y voir plus clair!

Une vue de notre prototype dans lequel je dévoile la face cachée qui



affiche le système de codage et décodage de notre application.

Le principe est de générer un code à partir du code initial de l'ordinateur de l'utilisateur, de le mettre en forme en le modifiant (SérialNumber1), ensuite après réception de ce code fourni par l'utilisateur, lui renvoyer un autre code issu du code initial, mais mis en forme également de façon qu'il ne soit pas identique (SérialNumber2) qui va générer un NumberCode identique à celui qui sera stocké sur le computer de l'utilisateur.

Pour savoir si nous devons lui accorder une licence il ne nous reste plus qu'à comparer si les deux codes sont identiques (77MB2i dans notre exemple), et s'il le sont, débrider le logiciel automatiquement.

Je sais que vous pourriez me poser cette question, à savoir, que chaque fois que nous allons initialiser cette application, le système va générer des chiffres aléatoires différents ce qui peut amener le petit malin adepte du tournevis et de la combine à connaître le fonctionnement du système!

Je pourrais vous répondre que si vous voulez connaître la suite vous devrez suivre mes cours sur Internet.

Mais...je suis un temps soit peu philanthrope et ayant la fibre pédagogique qui coule dans mes veines, je vais vous dévoiler comment remédier à ce problème.

Il suffit de créer un fichier à accès direct et de stocker en mémoire le code que le computer de l'utilisateur aura généré, vous voyez c'est très simple à réaliser.

Pour conclure je vous donne quelques explications sur le fonctionnement de ce prototype :

Le bouton de commande RaZ sert à la remise à zéro des caractères affichés dans les champs de saisies.

Le bouton de commande1 sert à afficher les caractères aléatoires dans leurs champs de saisies respectifs

Le bouton de commande2 sert à afficher le code série du computer.

Le bouton de commande3 sert à afficher le code série que vous devez retourner à l'utilisateur, (Votre client)

Le système comparera les données enregistrées, et si elles sont identiques, activera l'intégralité du programme.

Pour éviter de vous chauffer la tête, je vous conseille de réaliser un décodeur qui vous donnera à partir du code initial de la machine du client automatiquement le code à retourner à l'utilisateur.

Le principe de ce codeur décodeur relève intégralement de cet article et vous devez le réaliser sans aucun problème à partir des données que je vous ai transmises.

Bon courage....

III - Liens

Télécharger cet exemple ici

Réflexions sur cette application (mise à jour)

Prochain tutoriel :

 ***Le contrôle documenté "MCI32.ocx"***

