Fondements sur les variables et les constantes

Date de publication : 16/06/2005

Par Maxence HUBICHE (Site Perso)

Que de questions sur les variables ! Et pourquoi je perds ma valeur ... ? Et pourquoi je ne peux pas déclarer ma variable comme ceci ou comme cela ... ? Bref, il y a plusieurs petits détails qu'il est bon de comprendre pour ne pas faire n'importe quoi lorsqu'on déclare, ou utilise des variables. Vous êtes intéressé ? Ce ne sera pas long.

I. Le fonctionnement par défaut
II. Déclaration & Affectation
II-A. Déclaration & Affectation d'une Variable
II-B. Déclaration & Affectation d'une Constante
III. Portée & Durée de vie

I. Le fonctionnement par défaut

Avant de commencer, il convient de rappeler quelques points importants. Ces points ont trait au fonctionnement par défaut des variables lorsqu'on programme en VBA. Voici donc un rappel des plus utiles pour la suite :

- Tout mot inconnu par le langage est automatiquement considéré comme une variable ;
- Toute variable (déclarée ou pas) dont le type n'est pas défini est en variant ;
- Il est impossible sans déclaration, d'obtenir une constante, puisque tout est variable ;
- S'il rencontre une erreur dans un libellé, l'interpréteur VBA pourrait générer une erreur, par exemple si la syntaxe correspond à l'appel d'une fonction, qu'il interprète comme une variable ;
- Il n'y a aucune vérification. Donc, une erreur de saisie d'un nom de variable est possible et conduira à l'interprétation de l'existence de deux variables ;

Par conséquent, ce fonctionnement induit un grand nombre d'erreurs par défaut.

On peut empêcher ce fonctionnement par défaut

Il faut qu'en haut du module (sur les premières lignes, avant les déclarations) il y ait la petite phrase :

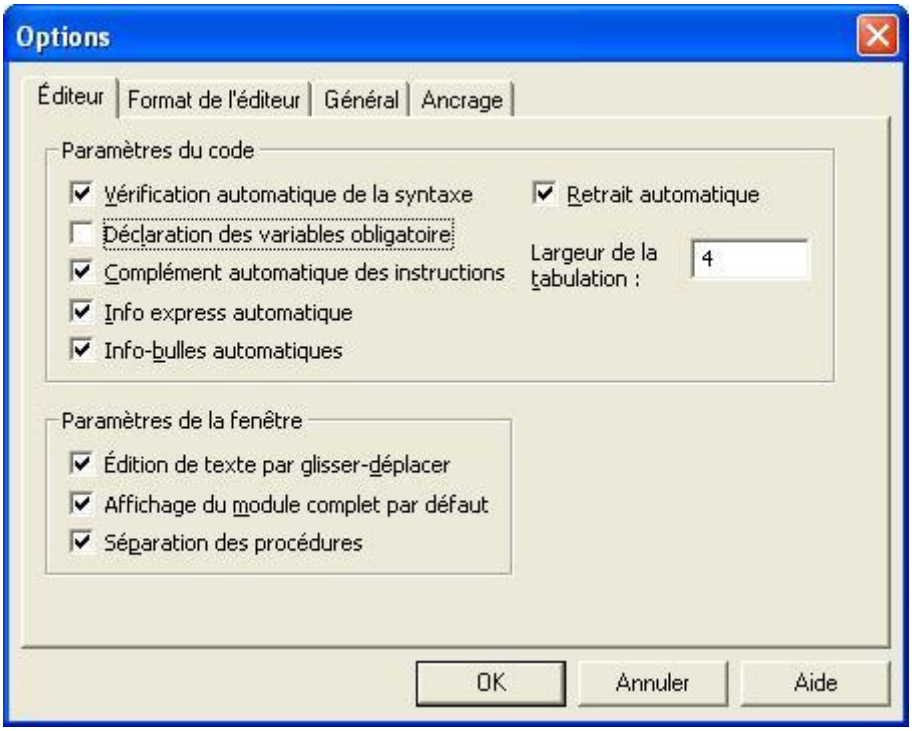
```
Option Explicit
```

Cela indique que **dans ce module toutes** les variables **doivent êtres** déclarées.

Si elles ne le sont pas, on obtient un message d'erreur :

On peut demander que cette ligne soit écrite automatiquement sur chaque **nouveau module.**

Pour cela on fait : Outils/Options … et on coche la seule case qui ne l'est pas :



II. Déclaration & Affectation

Maintenant que nous avons vu le fonctionnement par défaut des variables, examinons les moyens de produire un code propre. Pour cela, il nous faudra comprendre parfaitement ce que nous faisons, que ce soit au moment de la déclaration de notre variable (ou de notre constante), mais aussi au niveau de son affectation, c'est à dire, au moment où nous lui donnons sa valeur (variable de données) ou que nous la faisons pointer sur son objet (variable objet).

II-A. Déclaration & Affectation d'une Variable

Syntaxe de la déclaration :

{Dim (ou Private) | Public | Static} NomVar [([Index[,Index[,…]]])] [AS [New] Type]

Exemples :

```
Dim Toto                        'Déclare une variable Toto de type
Variant
Dim Toto as Long                'Déclare une variable Toto de type
Entier Long
Dim Toto as Recordset  'Déclare une variable objet Toto, de type
Recordset (jeu d'enr.)
Dim Noms(3) as String  'Déclare une variable Noms, de type
Caractère, pouvant contenir 4 valeurs (de 0 à 3)
```

ATTENTION ! Il y a une erreur extrêmement fréquente[1] que l'on retrouve dans le code :

```
Dim i, n, j As Long
```

Ici, seule la variable j est déclarée AS LONG. Les autres variables (i et n) sont en Variant ! Si vous voulez vraiment déclarer plusieurs variables sur une seule et même ligne, vous pouvez soit faire :

```
Dim i as Long, n As Long, j As Long
```

soit faire

```
Dim i as Long: Dim n As Long: Dim j As Long
```

Affectation :

Dans le cas d'une variable de données, la syntaxe est :

```
Dim toto as String              'Déclaration de toto en tant que
variable de type Chaîne de caractères
Toto = "coucou les p'tits loups"      'Affectation de la valeur
dans la variable
```

Dans le cas d'une variable objet, la syntaxe est :

```
Dim toto as TableDef            'Déclaration de la variable toto
comme un objet de type TableDef
Set toto = Currentdb().TableDefs("Commandes") 'toto pointe vers la
TableDef Commandes de la base en cours
```

Vous noterez que, dans ce cas, l'utilisation du mot clef SET est obligatoire pour l'affectation.

II-B. Déclaration & Affectation d'une Constante

Syntaxe de la déclaration :

[Public] Const NomConstante [As Type] = Valeur

Exemple :

```
Const PI as Double = 3.14159
```

[1] Un merci tout particulier à FRED.G qui, lors de la relecture de ce guide m'a rappelé cette erreur fréquente.

III. Portée & Durée de vie

Portée : Accessibilité d'une variable ou constante
Durée de vie : Durée de conservation de la valeur

Pour gérer la portée et la durée de vie, on doit faire attention à l'endroit où on déclare la variable (ou constante) ainsi que le choix de la syntaxe.

Portée - Durée de vie	Dans la procédure	Dans la zone de déclarations
Dim	Procédure - Procédure	Module - Projet
Private	N/A-N/A	Module - Projet
Public	N/A-N/A	Projet - Projet
Static	Procédure - Projet	N/A-N/A

Exemples :

```
Sub Compteur()
    Dim i as Long       'Déclarer i comme une variable de type Entier
Long
    i = i + 1           'Ajouter 1 à la valeur de i
    Msgbox i            'Afficher i
End sub
```

Si je lance cette procédure 4 fois de suite, elle affichera 1 quatre fois!

En effet, la déclaration est faite avec DIM, donc la durée de vie est limitée à la procédure !

```
Sub Compteur()
    Static i as Long    'Déclarer i comme une variable de type Entier
Long
    i = i + 1           'Ajouter 1 à la valeur de i
    Msgbox i            'Afficher i
End sub
```

Si je lance cette procédure 4 fois de suite, elle affichera 1, puis 2, puis 3, puis 4 !

ATTENTION !

L'appui sur le bouton  de la barre d'outils provoque la réinitialisation de toutes les variables.