TUTORIAL : LES VARIABLES SOUS VB

NIVEAU : PREMIERE RENCONTRE AVEC VB / DEBUTANTS / INITIES

Pré requis pour comprendre ce tutorial :

- Savoir lancer Visual Basic et créer un nouveau projet
- Connaître l'environnement de développement de Visual Basic 4.0 ou supérieur

Auteur : Dark sidious
Date de création : 12/01/2003
Version : 1.0

SOMMAIRE

INTRODUCTION

Ce tutorial est destiné à vous présenter les variables de VB, leurs portées, leurs déclarations, leurs types, etc. Enfin, tous ce qu'il faut savoir sur leur utilité tout simplement.
Je me suis efforcé de vous donné quelques petites astuces qui vous simplifieront la programmation, et vous dire des remarques pour que vous évitiez les pièges de VB à cause de son automatisation.
J'espère que la lecture de ce tutorial vous sera plaisante, et que cela vous donnera envie de continuer en Visual Basic qui est un langage merveilleux pour apprendre la programmation.
Pour avoir des informations supplémentaires sur les variables, si vous voyez des erreurs, si vous avez des problèmes ou que vous ne comprenez pas trop bien ce que j'essaie de vous apprendre dans ce tutorial, ou si vous avez des idées pour un nouveau tutorial, vous pouvez m'envoyer un message privé sur le site www.ProgOtoP.net à DARK SIDIOUS.

Bonne lecture, et bonne compréhension.

QU'EST-CE QU'UNE VARIABLE ?

C'est uniquement un endroit de la mémoire qui est réservé au programme et qui stocke des données du programme. En fait, une variable stocke toutes les informations que vous voulez que l'ordinateur se rappelle. Elles sont de plusieurs types : il y a les constantes qui ne sont pas des variables à proprement parlé car elles ne varient pas (et c'est pas un jeu de mot ;-), il y a les variables qui stockent les nombres entiers, les variables qui stockent les nombres à virgules, les variables qui stockent les chaînes de caractères (contrairement à d'autres langages, il est possible de stocker une phrase complète dans une simple variable texte), les variables qui stockent des tableaux, les variables booléenne, les variables monétaires, les variables de dates, les variables d'objets, les variables « variables » (type variant), les variables personnalisées et enfin des variables dites « pointeurs » dont le rôle et de renvoyer vers un emplacement de la mémoire donné sans pour autant occuper beaucoup de place.

Les variables les plus utilisées en VB sont les constantes, les variables de nombres entiers, les variables de nombres à virgules et les variables de caractères. Les variables tableaux sont logiquement utilisées que lorsqu'il faut stocker un grand nombre de donnée étant en liaison, alors que les variables pointeurs sont très souvent laissées de côté par les programmeurs.

LES CONSTANTES

Avant de s'attaquer aux variables à proprement parlé, nous allons voir un type de variable particulier : les constantes.

Comme leurs noms nous le laissent présager, il s'agit de données qui restent toujours à la même valeur.

Il existe deux types de constantes : celles qui sont définies par l'utilisateur, et celles qui sont définies par le système lui-même.

A quoi peuvent servir des données qui ne peuvent pas varier ?

Les constantes sont surtout utilisées pour simplifier la compréhension du code. Par exemple, au lieu d'utiliser des chiffres pour symboliser les couleurs système, on peut attribuer une constante beaucoup plus explicite pour chaque couleur

Pour déclarer une constante personnalisée, il vous suffit de taper le code suivant :

```
Public Const COU_Rouge = 0xFF
```

Leur portée est identique à la portée des variables. (voir plus bas)

Les constantes systèmes sont des constantes attribuées automatiquement par le système pour des données assez variables.

LES CONSTANTES STANDARD IMPORTANTES :

CONSTANTES DE TOUCHES SPECIALES :

Nom de la Constante	Description	Equivalent numérique
VbCrLf	Combinaison Retour chariot/Saut de ligne	Chr$(13) + Chr$(10)
VbCr	Retour chariot	Chr$(13)
VbLf	Saut de ligne	Chr$(10)
VbNewLine	Nouvelle ligne	Chr$(13) + Chr$(10)
VbNullChar	Nul (ASCII zéro)	Chr$(0)
VbNullString	Chaînes de caractères nulles	
VbTab	Tabulation	Chr$(9)
VbBack	Effacement arrière	Chr$(8)
VbFormFeed	Non utilisé sous Windows 95	Chr$(12)
VbVerticalTab	Non utilisé sous Windows 95	Chr$(11)

CONSTANTES DE COULEURS :

Constante de couleur	Couleur	Equivalent hexadécimal
VbBlack	Noir	0x0
VbRed	Rouge	0xFF
VbGreen	Vert	0xFF00
VbYellow	Jaune	0xFFFF
VbBlue	Bleu	0xFF0000
VbMagenta	Magenta	0xFF00FF
VbCyan	Cyan	0xFFFF00
VbWhite	Blanc	0xFFFFFF

CONSTANTES DE CARACTERES DE CONTRÔLES :

Constantes clavier	Commande clavier	Equivalent hexadécimal
VbKeyLButton	Bouton souris gauche	0x1
VbKeyRButton	Bouton souris droite	0x2
VbKeyMButton	Bouton souris milieu	0x4
VbKeyBack	Retour	0x8
VbKayTab	Tab	0x9
VbKeyReturn	Entrée	0xD
VbKeyShift	Majuscule	0x10
VbKeyControl	Ctrl	0x11
VbKeyMenu	Menu	0x12
VbKeyPause	Pause	0x13
VbKeyCapital	Verrouillage majuscule	0x14
VbKeyEscape	Echap	0x1B
VbKeySpace	Espace	0x20
VbKeyPageUp	Page précédente	0x21
VbKeyPageDown	Page suivante	0x22
VbKeyEnd	Fin	0x23
VbKeyHome	Début	0x24
VbKeyLeft	Flèche gauche	0x25
VbKeyUp	Flèche vers le haut	0x26
VbKeyRight	Flèche droite	0x27
VbKeyDown	Flèche vers le bas	0x28
VbKeyPrint	Impression	0x2A
VbKeyKeyInsert	Insertion	0x2D
VbKeyDelete	Suppression	0x2D
VbKeyNumlock	Verrouillage numérique	0x90
VbKey & VbKeyF16	Touches de fonction F1 à F16	0x70 jusqu'à 0x7F

CONSTANTES DE COULEURS SYSTEME :

Constante de couleurs	Elément d'interface	Equivalent hexadécimal
VbScrollBars	Barres de défilement	0x80000000
VbDesktop	Bureau	0x80000001
VbActiveTitleBar	Barre de titre actif	0x80000002
VbInactiveTitleBar	Barre de titre inactif	0x80000003
VbMenuBar	Fond de menu	0x80000004
VbWindowBackground	Fon de fenêtre	0x80000005
VbWindowFrame	Encadrement fenêtre	0x80000006
VbMenuText	Texte de menu	0x80000007
VbWindowText	Texte de fenêtre	0x80000008
VbTitleBarText	Texte de barre de titre	0x80000009
VbActiveBorder	Cadre actif	0x8000000A
VbInactiveBorder	Cadre inactif	0x8000000B
VbApplicationWorkspace	Zone de travail MDI	0x8000000C
VbHighlight	Fond de sélection	0x8000000D
VbHighLighText	Texte de sélection	0x8000000E
VbButtonFace	Bouton	0x8000000F
VbButtonShadow	Ombre du bouton	0x80000010
VbGrayText	Texte inactif	0x80000011
VbButtonText	Texte de bouton	0x80000012
VbInactiveCaptionText	Texte de barre de titre inactive	0x80000013
Vb3Dhighlight	Sélection 3D	0x80000014
Vb3DDKShadow	Ombre 3D	0x80000015
Vb3Dlight	Sélection 3D	0x80000016
VbInfoText	Texte des info bulles	0x80000017
VbInfoBackground	Fond des info bulles	0x80000018

Enfin, il existe deux constantes standards très utilisées : Vrai et Faux ou **True** et **False**. Il s'agit des deux valeurs booléennes prise par une variable de type Boolean. Leur valeur respective est :

- True : -1
- False : 0

LES DIFFERENTS TYPES DE VARIABLES

Voici une énumération ainsi qu'une description des différents types de variables :

OCTET :

- Espace utilisé : 1 octet,
- Plage de valeur : 0 à 255 sans virgule
- Mot clé : Byte
- Utilité : stockage de nombres entier très petit

BOOLEEN :

- Espace utilisé : 2 octets
- Plage de valeur : True ou False (vrai ou faux)
- Mot clé : Boolean
- Utilité : Stockage du renvoie d'une fonction, stockage d'un choix de l'utilisateur.

ENTIER :

- Espace utilisé : 2 octets
- Plage de valeurs : -32 768 à 32 767 sans virgule
- Mot clé : Integer
- Utilité : Souvent utilisé pour stocker des valeurs diverses entières et petites

ENTIER LONG :

- Espace utilisé : 4 octets
- Plage de valeurs : -2 147 483 648 à 2 147 483 647 sans virgule
- Mot clé : Long
- Utilité : Souvent utilisé pour stocker des valeurs diverses entières

NOMBRE A VIRGULE FLOTTANTE SIMPLE PRECISION :

- Espace utilisé : 4 octets
- Plage de valeurs - 3,402 823 E38 à -1,401 298 E-45 pour les valeurs négatives ; 1,401 298 E-45 à 3,402 823 E38 pour les valeurs positives
- Mot clé : Single
- Utilité : Souvent utilisé pour stocker des nombres à virgules

NOMBRE A VIRGULE FLOTTANTE DOUBLE PRECISION :

- Espace utilisé : 8 octets
- Plage de valeurs : -1,797 693 134 862 32 E308 à -4,940 656 458 412 47 E-324 pour les valeurs négatives ; 4,940 656 458 412 47 E-324 à 1,797 693 134 862 32 E308 pour les valeurs positives
- Mot clé : Double
- Utilité : Utilisé pour stocker des nombres avec une précision importante.

ENTIER A DECALAGE :

- Espace utilisé : 8 octets
- Plage de valeurs : -922 337 203 685 477,5808 à 922 337 203 685 477,5807
- Mot clé : Currency
- Utilité : Utilisé pour stocker des nombres à virgules avec un nombre de chiffre après la virgule assez réduit.

DECIMAL :

- Espace utilisé : 14 octets
- Plage de valeurs : +/-79 228 162 514 264 337 593 543 950 335 sans séparateur décimal
  +/-7,9228162514264337593543950335 avec 28 chiffres à droite du séparateur décimal
  Le plus petit nombre différent de zéro est
  +/- 0.0000000000000000000000000001
- Mot clé : Decimal
- Utilité : Surtout utilisé pour stocker des nombres avec une précision très grande, cependant, il demande beaucoup de mémoire.

DATE :

- Espace utilisé : 8 octets
- Plage de valeurs : 1er janvier 100 au 31 décembre 9999
- Mot clé : Date
- Utilité : Surtout pour stocker des dates classiques

CHAINE DE TEXTE A LONGUEUR VARIABLE :

- Espace utilisé : 10 octets + la longueur de la chaîne
- Plage de valeur : de 0 à 2 milliards de caractères
- Mot clé : String
- Utilité : Utilisé pour stocker du texte mais on préfère toutefois les chaînes de texte à longueur fixe, moins gourmandes en mémoire

CHAINE DE TEXTE A LONGUEUR FIXE :

- Espace utilisé : La longueur de la chaîne
- Plage de valeur : de 1 à 65 400 caractères
- Mot clé : String
- Utilité : Très utilisé pour stocker du texte

VARIABLE NOMBRE :

- Espace utilisé : 16 octets
- Plage de valeur : Toute valeur numérique avec la plage de valeur d'une variable Double
- Mot clé : Variant
- Utilité : Surtout utilisé lorsqu'on ne connaît pas la longueur du nombre à stocker.

VARIABLE CARACTERES :

- Espace utilisé : 22 octets + la longueur de la chaîne
- Plage de valeur : identique à la plage de valeur d'une variable de type String longueur variable.
- Mot clé : Variant
- Utilité : Utilisé dans le cas où l'on ne connaît pas le type de données à stocker

TYPE DEFINI PAR L'UTILISATEUR :

- Espace utilisé : dépend des éléments du type
- Plage de valeurs : identique à celles de chacun des éléments du type
- Mot clé : Type
- Utilité : Très utilisé pour lier les variables entres-elles.

LA DECLARATION DES VARIABLES

La déclaration des variables n'est pas indispensable sous VB, mais je vous le conseille grandement pour vous évitez d'avoir des erreurs complètement inattendues, notamment un dépassement de capacité dû au compilateur VB.

Utilisez les mots clés **Option Explicit** pour obliger la déclaration des variables. Ainsi, si vous utilisez une variable non déclarée, VB signalera une erreur.

La déclaration des variables peut se faire de différente façon selon le type de variable, sa portée, ainsi que l'endroit où vous la déclarez. Cependant, la méthode la plus courant pour déclarer une variable est :

```
Dim NOM_DE_LA_VARIABLE As Type
```

Le nom de la variable ne doit pas correspondre à un élément du langage, par exemple, la variable ne doit pas être nommée « Int » ou « Sqr ». Par contre, vous êtes libres de choisir le nom que vous voulez. La limite du nombre de caractère pour le nom de la variable étant de 255.

> *Astuce : Je vous conseille de faire précéder le nom de la variable par 2 ou 3 lettres explicitant le type de la variable, et de donner un nom clair à la variable. Par exemple : INT_Age pour une variable de type Integer qui aurait pour fonction de stocker l'âge de l'utilisateur.*

Il faut distinguer deux sortes de variables : Les variables publiques et les variables privées

VARIABLES PRIVEES :

Les variables privées sont des variables qui ne sont valable que dans la fonction où elles sont déclarées. Par exemple, si vous déclarez une variable dans une fonction avec le mot clé Dim, celle-ci sera privée. Vous ne pourrez accéder à son contenu qu'uniquement dans la fonction où elle a été déclarée.

Remarque : Ce sont les variables les plus utilisées car la mémoire encombrée par une variable privée est automatiquement libérée par le programme dès que la fonction est terminée, contrairement à une variable publique.

L'utilité de telles variables sont surtout le stockage de données temporairement pendant des calculs dans une fonction. Ces variables sont également utilisées pour stocker des informations pour une éventuelle manipulation par la suite

Exemple : Ces variables peuvent stocker une valeur entrée par l'utilisateur pour une manipulation avec des commandes VB : pour convertir un nombre en nombre entier par exemple.

On peut déclarer une variable privée de plusieurs façons :

```
Dim NOM_DE_LA_VARIABLE As Integer

Private NOM_DE_LA_VARIABLE As Integer
```

Cependant, l'utilisation du mot clé Private ne doit se faire que dans la partie déclaration d'une feuille, d'un module, d'un module de classe, etc.

En plus de ces quelques particularités des variables privées, il faut savoir que celles-ci peuvent conserver les données qu'on leurs a attribué même lorsque la fonction est terminée. Pour cela, il faut déclarer la variable de type **Static** :

```
Static NOM_DE_LA_VARIABLE As Integer
```

Lorsqu'une variable est « statique », elle conserve toutes les données, mais cela encombre la mémoire et elles ne sont pas accessibles à partir d'une autre fonction, d'où l'utilité d'utiliser une variable publique.

VARIABLES PUBLIQUES :

Les variables publiques sont des variables qui sont accessibles non pas uniquement dans une fonction, mais dans plusieurs.

Celles-ci sont surtout utilisées pour stocker des informations entre plusieurs fonctions. Cependant, ceci n'est pas très recommandé, car elles occupent souvent inutilement de la mémoire pour rien. De plus, comme elles peuvent être modifiées par plusieurs fonctions, elles mènent souvent à des erreurs inattendues.

> *Remarque : Les variables publiques sont souvent utilisées par les débutants à cause de leur souplesse d'utilisation, cependant, l'utilisation de telles variables encombre le système, et la modification de leurs valeurs doit se faire avec précaution pour ne pas rendre les données inexactes.*

On peut déclarer une variable publique de plusieurs façon. C'est l'endroit de la déclaration qui influence la portée de la variable : Si la variable est déclarée dans un module, elle sera dite « globale », c'est-à-dire que toutes les fonctions du programme pourra accéder aux données enregistrées dans ces variables. Si elle est déclarée dans la partie déclaration d'une feuille, ou d'un module de classe, etc., elle sera publique à la feuille, au module de classe, etc. Ainsi, une fonction extérieure à l'endroit ou la variable publique a été déclarée ne pourra accéder à ses données.

```
Public NOM_DE_LA_VARIABLE As Integer

Dim NOM_DE_LA_VARIABLE as Integer
```

Cependant, l'utilisation du mot clé Public ne doit se faire que dans la partie déclaration d'une feuille, d'un module, d'un module de classe, etc.

LES TABLEAUX

Les tableaux sont une sorte de variable étendue : ils permettent de stocker, dans la même variable, plusieurs données. Par exemple, vous pouvez stocker dans un tableau tous les noms d'utilisateurs du programme.

La déclaration d'une variable de type tableau de change pas beaucoup de celle d'une variable classique, il suffit uniquement d'indiquer en plus le nombre de paramètres du tableau entre parenthèse.

Par exemple, pour une variable de type tableau à 2 lignes et 3 colonnes, il faudrait la déclarer ainsi :

```
Dim NOM_DE_LA_VARIABLE(1,2) As Integer
```

Remarque : Il faut savoir que les paramètres commencent à 0, ainsi, dans l'exemple si dessus, le paramètre 1 indique qu'il s'agit d'une variable tableau à 2 lignes, et le paramètre 2 indique qu'il s'agit d'une variable tableau à 3 colonnes.

Il est possible de définir des tableaux de taille gigantesque, la limite supérieure du nombre de paramètre étant fixée à 60 paramètres ! Attention cependant : la place occupée par une variable tableau peut très vite devenir très importante selon le nombre de paramètres.

Pour faire commencer les index du tableau à 1 au lieu de zéro, il suffit de le déclarer lors de la déclaration de la variable, par exemple :

```
Dim NOM_DE_LA_VARIABLE(1 To 30) As Integer
```

Cette variable de type tableau comportera donc 30 lignes, indexées de 1 à 30.
Il est également possible de déclarer un tableau avec une limite basse négative.

Il est également possible de redimensionner le tableau après l'avoir déclarer. Il suffit pour cela d'utiliser le mot clé **ReDim** :

```
ReDim NOM_DE_LA_VARIABLE(1 To 31) As Integer
```

Cette instruction redimensionne le tableau NOM_DE_LA_VARIABLE en lui affectant 31 lignes indexées de 1 à 31.

Il est possible de définir des tableaux dynamiques dont la taille et le nombre de paramètres peuvent changer lors de l'exécution du programme, et qui n'est donc pas fixé lors de la déclaration de la variable. Pour cela, il suffit de déclarer la variable avec uniquement des parenthèses sans aucun arguments à l'intérieur :

```
DIM NOM_DE_LA_VARIABLE( ) As Integer
```

Ainsi, vous pouvez déclarer un tableau sans en connaître les dimensions lors de l'exécution, puis redimensionner celui-ci selon un paramètre entré par l'utilisateur par exemple.

La portée d'une variable tableau est identique à celle d'une variable classique.

Vous pouvez définir les limites d'un tableau statique (donc non dynamique), grâce aux instruction **UBound** et **LBound** :

```
Dim INT_Limite_Inférieure As Integer

Dim INT_Limite_Supérieure As Integer

INT_Limite_Inférieure = LBound (NOM_DE_LA_VARIABLE_TABLEAU,
NUMERO_PARAMETRE)

INT_Limite_Supérieur = UBound (NOM_DE_LA_VARIABLE_TABLEAU,
NUMERO_PARAMETRE)
```

LES TYPES DE DONNEES

Les types de données personnalisables peuvent être intéressants pour des grosses applications, ou alors, si vous voulez lier les variables entre elles. Sinon, les types de données vous permettent également de créer votre propre type de variable, mais en se basant sur des types déjà définis.

Il faut déclarer ces types dans un module pour qu'ils soient valables dans tout le projet. Après la déclaration du type, il faut déclarer tous les éléments du type. Enfin, vous terminez la déclaration du type par les mots clés **End Type** :

```
Public Type TYP_Personne
        STR_Nom As String
        STR_Prénom As String
        INT_Age As Integer
End Type
```

Ce type peut, par exemple, être défini pour stocker les données relatives à l'utilisateur.

Ensuite, ce type est reconnu comme un type standard de variable, et vous pouvez y accéder avec une variable de type TYP_Personne :

```
Dim TYP_Personnes(20) As TYP_Personne
```

Il est également possible de créer des tableaux avec les types prédéfinis.

Ensuite, pour accéder à un champ du type, il suffit de taper le nom de la variable, suivie d'un « . », suivie du nom du champ. Par exemple :

```
TYP_Personnes(2).STR_Nom
```

Ce code vous renverra la valeur contenue dans le champ STR_Nom du tableau TYP_Personnes à la ligne 3.

Il est possible d'imbriquer les types : si par exemple, vous avez deux types de données, et que le premier type de donnée doit contenir un type de donnée du second, c'est tout à fait réalisable.

```
Type TYP_Nombres

        Dim INT_Nombre As Integer

        Dim INT_Résultat As Long

End Type

Type TYP_Calculs

        Dim INT_Opérande As Integer

        Dim TYP_Nombre As TYP_Nombres

End Type
```

LES VARIABLES POINTEURS

Bien que très peu utilisées, les variables pointeurs existent sous VB. Ces variables ont l'avantage de n'occuper de très peu de place en mémoire, car il s'agit de variable assez particulière : elles ne stockent pas de données brutes, mais un emplacement de la mémoire ou les données sont enregistrées.

Ces types de variables sont disponibles pour la programmation système, mais elles sont très souvent laissées de côté par les programmeurs sous VB. C'est pour cette raison que nous allons uniquement les introduire ici.

Leurs principales utilité viennent des API de windows. Mais cela reste un moyen très puissant pour accéder à des données du système.

Voici les fonctions associées à ces variables :

- **VarPtr** : pointeur sur variable
- **StrPtr** : pointeur sur chaîne de caractère
- **ObjPtr** : pointeur sur objets

En fait, les données stockées dans une variable de type pointer ne stocke que des adresse mémoire, mais aucunement les données que contiennent ces adresses.

L'ATTRIBUTION DE VALEURS AUX VARIABLES

L'attribution de valeurs aux variables se fait de façon très explicite. Il suffit, par exemple, pour attribuer la valeur 10 à une variable de type Integer, de taper :

```
Dim INT_Nombre As Integer

        INT_Nombre = 10
```

Ainsi, l'attribution de valeur se fait de façon très intuitive. De même, pour copier le contenu d'une variable dans une autre, il suffit de taper ceci :

```
Dim INT_Source As Integer

Dim INT_Arrivée As Integer

        INT_Arrivée = INT_Source
```

Vous pouvez également faire des calculs sur les variables et attribué le résultat du calcul dans une autre variable :

```
Dim INT_Nombre As Integer

Dim INT_Résultat As Integer

        INT_Résultat = 2 * INT_Nombre + 4 – INT_Nombre
```

L'attribution de valeur est identique pour les variables de type tableau.

Pour une variable de type String (chaînes de caractères), il est nécessaire de mettre la valeur à attribuer entre parenthèse :

```
Dim STR_Texte As Integer

        STR_Texte = « Salut »
```

Les variables de type Boolean ne peuvent prendre que les valeurs True (-1) ou False (0) :

```
Dim BOL_Réponse As Boolean

        BOL_Réponse = True
```

L'attribution des valeurs à une variable de type personnalisée doit se faire de cette manière :

```
Public Type TYP_Nombre

        INT_Nombre_Entier_Court As Integer

        LNG_Nombre_Entier_Long As Long

End Type

Dim TYP_Nombres As TYP_Nombre

TYP_Nombres.INT_Nombre_Entier_Court = 10
```

CONVERSION DE VALEURS ENTRE DIFFERENTS TYPES DE VARIABLE

Il existe différentes fonctions VB pour convertir les données des variables de différents types.

Voici une énumération de ces mot-clé :

- **Cbool** : Converti n'importe quelle expression en valeur Boolean
- **Cbyte** : Converti n'importe quelle expression en valeur Byte
- **Ccur** : Converti n'importe quelle expression en valeur Currency
- **Cdate** : Converti n'importe quelle expression en valeur Date
- **CDbl** : Converti n'importe quelle expression en valeur Double
- **Cdec** : Converti n'importe quelle expression en valeur Decimal
- **Cint** : Converti n'importe quelle expression en valeur Integer
- **CLng** : Converti n'importe quelle expression en valeur Long
- **CSng** : Converti n'importe quelle expression en valeur Single
- **Cvar** : Converti n'importe quelle expression en valeur Variant
- **CStr** : Converti n'importe quelle expression en valeur String

Cependant, ces fonctions de conversion peuvent donner des résultats assez incohérents pour certains types de données. Par exemple, la conversion d'une chaîne de caractère en type Date. Tout dépend bien entendu de l'expression passée en paramètres.

Exemple : conversion d'une variable Integer en Long :

```
Dim INT_Nombre_Source As Integer

Dim LNG_Nombre_Arrivée As Integer

INT_Nombre_Source = 10

LNG_Nombre_Arrivée = CLng(INT_Nombre_Source)
```

CONCLUSION

Ce tutorial touche à sa fin, et j'espère que vous savez plus sur les variables sous VB. Les variables étant les éléments du langage les plus utilisés, il est important de savoir les manier correctement. C'est la base de la programmation : le maniement des données.
J'espère que la lecture de ce tutorial vous a satisfaite, et que vous continuerez à programmer sous Visual Basic. Et si vous avez des commentaires à me faire, n'hésitez pas à m'envoyer vos messages sur le site www.ProgOtoP.net.