Chapitre 4
Le langage VBA

1. Une programmation impérative

VBA est un langage de programmation impérative, c'est-à-dire que les instructions qui sont codées doivent s'exécuter dans leur ordre d'apparition dans le programme. Une instruction ne peut s'exécuter que si celle qui la précède a elle-même été exécutée.

Un chien suivra les instructions dans leur ordre d'apparition : « Assis », « Debout », « Couché ».

2. Syntaxes possibles

En VBA, les instructions sont séparées par un retour à la ligne ou par le signe deux-points « : ».

Par exemple, considérons le programme suivant :

```
Instruction_1
Instruction_2
Instruction_3
```

L'exécution de ce programme entraînera l'exécution d'`Instruction_1`, puis `Instruction_2` et enfin `Instruction_3`. Le programme aurait pu s'écrire de la manière équivalente suivante :

```
Instruction_1 : Instruction_2 : Instruction_3
```

Si une instruction est trop longue pour tenir sur une seule ligne, ou si vous souhaitez la répartir sur plusieurs lignes pour votre confort de lecture, il est possible de passer à la ligne en utilisant le symbole souligné (underscore) « _ ». Ainsi l'exécution du programme suivant déclenche l'exécution de l'instruction `Instruction_A` :

```
Instruc_
tion_A
```

3. Structure d'un programme

Un programme VBA se décompose en série de procédures et fonctions écrites dans le but de réaliser une ou plusieurs opérations. Les instructions doivent être codées à l'intérieur de ces procédures ou fonctions. Il est question de déclaration de procédures (et fonctions).

Dans un module, le programme est donc constitué d'une série de déclarations de procédures et de fonctions. Une fois ces procédures déclarées, elles seront exécutées lorsqu'elles seront appelées par le programme.

Pour lancer un programme VBA, on exécute une macro, qui est une procédure particulière et qui contient des instructions faisant appel aux autres fonctions et procédures. Un programme VBA doit avoir au minimum une macro déclarée pour fonctionner.

Voici un exemple de programme VBA à l'intérieur d'un module :

```
Sub Procedure_A()
...
End Sub
Sub Procedure_B()
...
End Sub
Function Fonction_C...
...
```

```
End Function
Sub Macro_1()
Procedure_A
...
End Sub
```

Dans cet exemple, on peut constater la déclaration des procédures `Procedure_A` et `Procedure_B`, de la fonction `Fonction_A` et de la macro `Macro_1`. La première instruction de `Macro_1` est un appel de la procédure `Procedure_A`. La première opération exécutée lors du lancement de la macro `Macro_1` sera donc le lancement de la procédure `Procedure_A`.

Remarque

Attention, le nom que vous donnez à vos macros, procédures et fonctions n'a pas d'impact sur leur nature, exemple, vous pouvez avoir :

```
Sub MaFonction()
...
End Sub
```

Mais il est évident que vous ne vous faciliterez pas la vie ainsi.

4. Les variables

Les variables sont des récipients qui permettent de stocker des informations à tout moment de l'exécution d'un programme et de les exploiter à n'importe quel autre moment. En VBA, une variable est définie par deux attributs :

– Son nom, qu'on utilisera pour accéder aux informations qu'elle contient ; pour les conventions, voir Conventions de nommage et typographies du code VBA.

– Son type de donnée stockée.

4.1 La syntaxe de déclaration

Tout comme les fonctions et procédures, pour qu'une variable puisse être appelée et utilisée, il faut la déclarer. La syntaxe de déclaration d'une variable est la suivante :

```
Dim NomDeVariable As TypeVariable
```

Le mot-clé `Dim` sert à déclarer une variable. Il est suivi du nom de la variable. Après le mot-clé `As` se trouve le type de la variable. Il existe plusieurs types de données qu'il est possible de manipuler en VBA.

Lorsqu'on présente son chien, on indique son prénom et sa race (« Snoopy, cocker »), et pour utiliser une variable, on l'appelle par son nom.

4.2 Les types de données

Il existe plusieurs types de données (car on a également des constantes).

Tout comme il existe plusieurs races de chiens.

4.2.1 Les types numériques

Les valeurs entières

Le type `Byte` permet de stocker un entier compris entre 0 et 255 (stocké sur 8 bits, soit 1 octet, Byte en anglais),

```
Dim b As Byte
b = 13
```

Le type `Integer` contient un entier compris entre -32 768 et 32 767 (sur 16 bits).

```
Dim i As Integer
i = 28
```

Le type `Long` contient un entier compris entre -2 147 483 648 et 2 147 483 647 (sur 32 bits).

```
Dim l As Long
l = 52
```

Les valeurs réelles (décimales)

Le type `Single` contient une valeur réelle comprise entre -3.4028823 10^{38} et -1.401298 10^{-45} pour les nombres négatifs et entre 1.401298.10^{-45} et 3.402823 10^{38} pour les nombres positifs.

```
Dim s As Single
s= 2.3
```

Le type `Double` contient une valeur réelle comprise entre -1,797693134 86231.10^{308} et -4,94065645841247.10^{-324} pour les nombres négatifs et entre 4,94065645841247.10^{-324} et 1,79769313486231.10^{308} pour les nombres positifs.

```
Dim d As Double
d = 0.123456
```

Le type `Currency` contient une valeur comprise entre -922 337 203 685 477,5808 et 922 337 203 685 447,5807 (64 bits). Ce type de données est utilisé notamment pour les calculs monétaires ou les calculs à virgule fixe.

```
Dim c As Currency
c = 3.1415
```

Remarque

On notera que le séparateur utilisé en VBA pour séparer la partie entière de la partie décimale est le point « . ».

4.2.2 Les autres types de données

Les valeurs booléennes

Le type `Boolean` peut contenir `True` (vrai) ou `False` (faux).

```
Dim b As Boolean
b = True
```

Les chaînes de caractères

Le type `String` permet de stocker des chaînes de caractères (du texte). Il est possible de déclarer des chaînes de longueur fixe ou de longueur variable. Une chaîne de longueur fixe peut compter jusqu'à 65 536 caractères, alors qu'une chaîne de longueur variable peut en contenir un peu plus de 2 milliards (2^{31} = 2 147 483 648). Les valeurs textuelles seront contenues à l'intérieur de guillemets « " ».

```
Dim str1 As String
Dim str2 As String * 10
str1 = "Hello"
```

Ici, on aura déclaré une chaîne de caractères `str1` de longueur variable, et une chaîne de caractères `str2` d'une longueur de 10 caractères.

Les dates et heures

Le type `Date` permet de stocker des dates et heures, ainsi que des durées. La syntaxe des dates se fait entre dièses « # », au format #MM/JJ/AAAA [hh:mm:ss AM/PM]#, comme dans l'exemple suivant :

```
Dim dt As Date
dt = #7/18/1940 1:32:00 PM#
```

Le type Variant

Le type `Variant` est un type de données qui regroupe les caractéristiques de tous les autres types de données. Il est codé sur 16 bits, mais peut également contenir une chaîne de caractères sur 22 octets. Lorsqu'aucun type de variable n'est défini lors de la déclaration d'une variable, le type par défaut donné par VBA est `Variant`.

```
Dim v as Variant
v = 1.2 : v = "Charles de Gaulle"
```

4.3 Les déclarations multiples de variables

Il est possible, si vous le souhaitez, de déclarer plusieurs variables sur la même ligne. Deux solutions s'offrent à vous. La première consiste à utiliser la syntaxe avec les « : ».

```
Dim a As Integer : Dim b As String : Dim c As Date
```

La seconde consiste à séparer les variables et leur type par des virgules.

```
Dim d As Integer, e As String, f As Date
```

Remarque

NB : attention, la syntaxe suivante déclarera deux variables `g` et `h` de type `Variant`, et une variable `i` de type `String`.

```
Dim g, h, i As String
```

4.4 Affectation d'une valeur à une variable

En VBA, pour affecter une valeur à une variable, la syntaxe est la suivante :

```
Variable = valeur_assignee
```

Comme ceci :

```
Dim a As Integer
a = 2
```

La variable a prend la valeur attribuée 2.