Initiation à la programmation évènementielle avec VBA (Visual Basic pour Applications)

Avant de commencer à étudier le langage VBA, il est intéressant d'en avoir une description au travers d'une définition mais aussi de comprendre les notions de programmation événementielle et celle d'objets.

1. Introduction

Visual Basic for Applications (VBA) est une implémentation de Microsoft Visual Basic qui est intégrée dans toutes les applications de Microsoft Office, dans quelques autres applications Microsoft comme Visio et au moins partiellement dans quelques autres applications comme AutoCAD et WordPerfect. Il remplace et étend les capacités des langages macro spécifiques aux plus anciennes applications comme le langage WordBasic intégré à une ancienne version du logiciel Word, et peut être utilisé pour contrôler la quasi-totalité de l'IHM des *applications hôtes*, **ce qui inclut la possibilité de manipuler les fonctionnalités de l'interface utilisateur comme les menus, les barres d'outils et le fait de pouvoir personnaliser les boîtes de dialogue et les formulaires utilisateurs**. (Extrait de Wikipédia)

La programmation événementielle :

Les composants d'une application événementielle interagissent entre eux et avec l'environnement. Ils communiquent en réponse à des événements. Ces événements peuvent correspondre à une action de l'utilisateur : un click sur un bouton de commande, une saisie dans une zone de texte, un choix dans une case d'option ou une case à cocher, le déplacement d'un objet, ... Ils peuvent aussi être déclenchés par le système : chargement d'une feuille, un top déclenché par l'horloge, ...

Le modèle objet d'Excel :

Il est le cœur de l'utilisation de VBA avec les applications de Microsoft office. Il fournit des commandes pour accéder aux classeurs et aux feuilles de calculs. Il est possible de manipuler des feuilles de calculs à l'aide d'instructions (écrites en Visual Basic) et de les associer à un événement (lors d'un click de souris sur une zone de cette feuille de calcul).

Lorsque vous manipulez Microsoft Excel, vous utilisez implicitement le code VBA. Par exemple, quand vous ouvrez un classeur, le code sous-jacent au menu *Fichier* → *Ouvrir* utilise la même fonctionnalité que la commande « **`Workbooks.Open`** ». L'objet de la feuille de calcul manipulé ici correspond à `Workbooks` et la fonctionnalité associée `Open`.

Un objet se caractérise au travers de ses propriétés. Il se manipule à l'aide de ses méthodes (fonctions, procédures)

Nous étudierons ces objets (Workbook, Worksheet etc.) un peu plus tard.

2. L'environnement de développement d'Excel

- Ouvrir un nouveau document Excel
- Enregistrer le projet (TpVBA1.xls) dans votre répertoire de travail
- Alt+F11 (fonctionnalité réversible)
- Renommer l'objet « Feuil1 » en « gestion » :
  - Selectionnez « Feuil1 » dans l'explorateur d'objet
  - Visualisez les attributs (les caractéristiques) de l'objet « Feuil1 »
  - Modifier le champ « name » afin d'effectuer ce changement de nom
- Renommer le classeur « thisWorkBook » en « classeur »

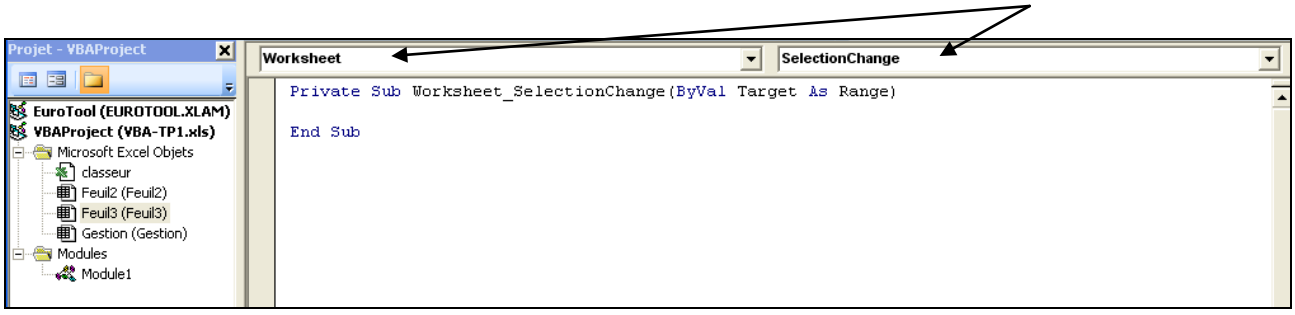
3. Exercice 1

Il s'agit ici de construire une feuille de calcul afin de calculer le montant cumulé des chiffres d'affaires d'une entreprise. Microsoft Excel permet, à l'aide de formules, d'automatiser des calculs. *Qu'en est-il lorsque les traitements (calcul d'une remise, etc.) correspondent à des conditions internes à l'entreprise ?*

- Dans la feuille « gestion » créer le tableau suivant

	A	B	C
1	Utilisation de VBA		
2			
3	CA 2004		
4	CA 2005		
5	CA 2006		
6	TOTAL CA HT	**0,00**	
7	REMISE	**0,00**	
8	TVA	0,00	
9	TOTAL CA TTC	0,00	
10			

- Alt+F11
- Sélection d'un événement associé à la feuille de calcul
  - Sélectionner la feuille « gestion » dans l'explorateur de projet
  - Dans le menu déroulant de la feuille de droite, sélectionner l'objet « `WorkSheet` »
  - Dans le menu déroulant de la feuille de droite, sélectionner la procédure « `SelectionChange` »



Cette fonction est associée à l'événement « SelectionChange ». Ainsi, lorsque l'utilisateur sélectionne une nouvelle zone (cellule) de la feuille « Gestion » les instructions présentes dans cette fonction seront exécutées.

- Copiez les deux lignes suivantes :

```
'Utilisation de la fonction SOMME (sum) d'Excel
gestion.Range("B6") = application.WorksheetFunction.Sum(Range("B3", "B5"))
```

Quel est le rôle joué par les points dans cette notation ?

_____
_____
_____

Que signifie « Range » ?

_____
_____
_____

- Effectuer un test de saisie dans la feuille « gestion » en saisissant des valeurs dans les cellules concernées.
- Alt+F11
- Copier les instructions suivantes (à la suite des lignes précédentes)

```
'Calcul de la remise en fonction du montant total HT
If Range("B6") > 2000 Then
Range("B7") = Range("B6") * 0.05
     ElseIf Range("B6") > 1000 Then
     Range("B7") = Range("B6") * 0.03
          Else
          Range("B7") = 0
End If
```

Expliquez les lignes ci-dessus

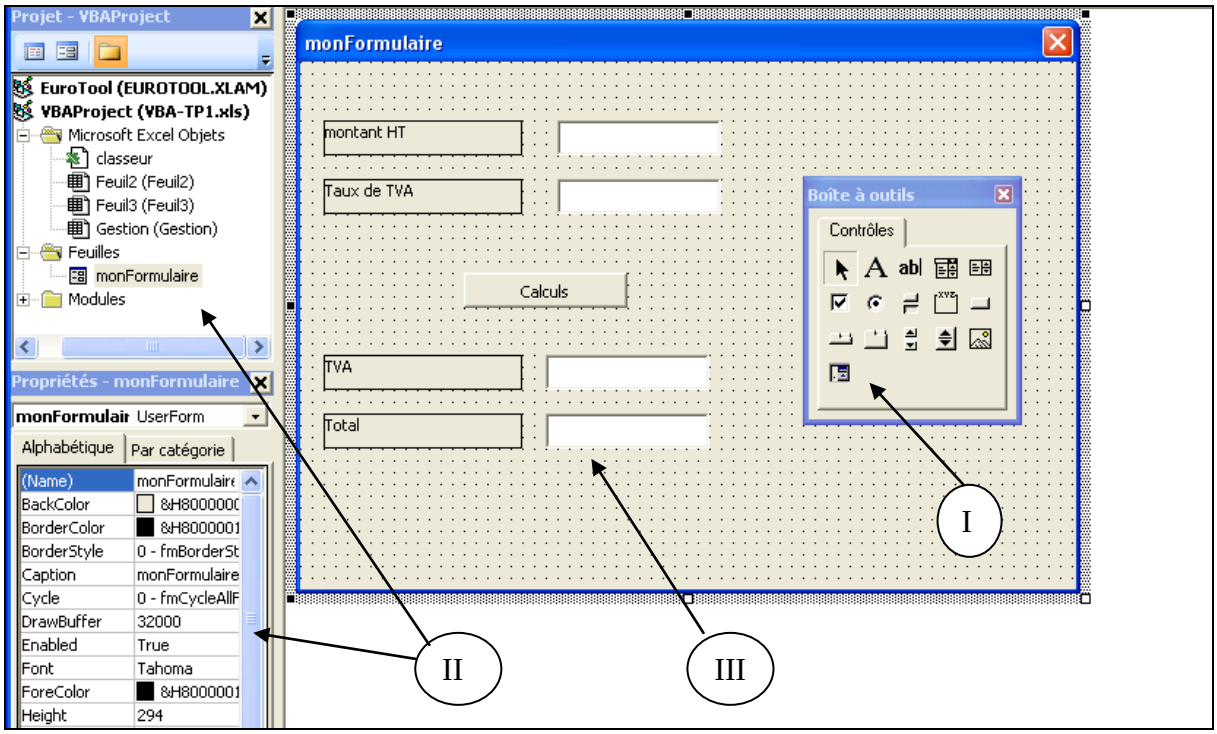
_____
_____
_____
_____
_____
_____

- En vous appuyant sur les éléments utilisés précédemment, complétez la fonction afin de calculer les autres montants.
- Vous ajouterez les instructions qui permettent d'effacer les montants calculés lorsqu'aucun montant ne figure dans les cellules « CA »

4. Exercice 2

Il est possible de créer des formulaires afin de permettre à l'utilisateur de manipuler une interface plutôt que les cellules Excel.

- Click droit sur l'explorateur d'objet : insertion « userForm »
- Renommer le nom (« name ») du formulaire « userForm1 » en « monFormulaire »
- A l'aide de la boite à outils, créer les objets suivants
  - 4 zones de textes
  - 4 intitulés
  - 1 bouton de commande
- Chaque élément sera renommé. Associez des nouvelles valeurs aux attributs de vos objets



I Boite à outils → Sélectionner l'objet souhaité et faites un glisser-déposer
II Explorateur de projet et propriété de l'objet sélectionné (ici le formulaire)
III Objet à renommer

- Double cliquer sur l'objet « bouton de commande »
- Compléter la fonction associée à l'événement click qui a été générée.
  - La fonction vérifie si la valeur de l'objet « montant » est différente de 0 ou n'est pas vide
  - La fonction vérifie si la valeur de l'objet « tauxTva » est différente de 0 ou n'est pas vide
  - La fonction calcul le montant de TVA et le montant TTC

Proposition de corrigé

Exercice 1 Découverte d'un évenement et manipulation des cellules Excel

```
Private Sub Worksheet_SelectionChange(ByVal Target As Range)

'Utilisation de la fonction somme "summ" d'Excel
gestion.Range("B6")    =    Application.WorksheetFunction.Sum(Range("B3",
"B5"))


'Calcul de la remise en fonction du montant total HT
If Range("B6") > 2000 Then
Range("B7") = Range("B6") * 0.05
    ElseIf Range("B6") > 1000 Then
    Range("B7") = Range("B6") * 0.03
    Else
    Range("B7") = 0
End If


'Calcul de la TVA
gestion.Range("B8")  =  (gestion.Range("B6")  -  gestion.Range("B7"))  *
0.196

'calcul du montant TTC
gestion.Range("B9")  =  gestion.Range("B6")  -  gestion.Range("B7")  +
gestion.Range("B8")

'réinitialisation des valeurs calculées
If Range("B3") = 0 And Range("B4") = 0 And Range("B5") = 0 Then
Range("B6:B9") = ""
End If
```

Exercice 1 Découverte et manipulation d'un objet formulaire

```
Private Sub CommandButton1_Click()

'verification que le champ montant est renseigné
If monFormulaire.mt.Value = 0 Or monFormulaire.mt.Value = "" Then
MsgBox "Vous devez saisir un montant"
ElseIf monFormulaire.tauxtva.Value = 0 Or monFormulaire.tauxtva.Value =
"" Then
    MsgBox "la Tva n'est pas renseignée"
     Else
     monFormulaire.tva.Value      =      (monFormulaire.mt.Value      *
     monFormulaire.tauxtva.Value) / 100
     monFormulaire.total.Value   =   monFormulaire.mt.Value   +   0   +
     monFormulaire.tva.Value

End If


End Sub
```