Le Tableur EXCEL
La Programmation en VBA

U.F.R. d'informatique

Juliette Dibie

PLAN

I. Introduction à excel VBA ____ *1*

II. L'enregistreur de macro ____ *2*

II.1. Enregistrer une macro ____ **2**

II.2. Exécuter une macro ____ **4**

II.3. Exercice ____ **4**

III. L'environnement Visual Basic Editor ____ *5*

III.1. L'explorateur de projets ____ **6**

III.2. Les modules de code ____ **7**

III.3. Les procédures ____ **8**

III.4. Modifier une macro ____ **9**

III.5. Exercice ____ **10**

IV. Les divers moyens pour exécuter une macro ____ *11*

IV.1. Exécuter une macro à partir d'un raccourci clavier ____ **11**

IV.2. Exécuter une macro à partir d'un bouton d'une barre d'outils ____ **12**

IV.3. Exécuter une macro à partir d'un menu ____ **14**

V. Le modèle objet d'Excel ____ *16*

V.1. Les objets ____ **16**

V.2. Les collections ____ **16**

V.3. L'accès aux objets ____ **17**

V.4. Exercice ____ **17**

V.5. Les propriétés des objets ____ **18**

V.6. Exercice ____ **18**

V.7. Les méthodes des objets ____ **19**

V.8. Exercice ____ **19**

V.9. Les événements ____ **20**

V.10. Exercice ____ **22**

V.11. L'explorateur d'objets ____ **23**

V.12. Comment obtenir de l'aide : quelques astuces ____ **25**

VI. Les fonctions définies par l'utilisateur ____ *26*

VI.1. Créer une Fonction personnalisée ____ **27**

VI.2. Exercice ____ **28**

VI.3. Utiliser les fonctions intégrées d'Excel ____ **29**

VI.4. Exercice ____ **30**

VII. Le langage VBA 31

VII.1. Les variables et les constantes 31
La déclaration des variables 31
Le type des variables 32
La portée des variables 34
La déclaration des constantes 34
L'instruction With...End With 36
Exercice 36

VII.2. Exécuter une procédure sans argument 37

VII.3. Les entrées et sorties standards 38
La fonction MsgBox 38
La fonction InputBox 40

VII.4. Récapitulatif sur les parenthèses et les listes d'arguments 41

VII.5. Exercice 42

VII.6. Les énoncés conditionnels 43
L'instruction If écrite sur une seule ligne 43
Le bloc If 43

VII.7. Les énoncés itératifs 45
La boucle Do...Loop 45
Exercice 45
La boucle For...Next 46
La boucle For Each...Next 47
L'instruction Exit 47
Exercice 47

VII.8. Les tableaux 48
Exercice 48

VII.9. Le débogage 49
Les erreurs de compilation et d'exécution 49
Les erreurs Logiques 49

VIII. Les objets UserForm 53

VIII.1. Créer un objet UserForm 55

VIII.2. Afficher et fermer un objet UserForm 57
Afficher un objet UserForm 57
Fermer ou masquer un objet `UserForm` 57

VIII.3. Associer du code à un objet UserForm 58
Associer du code à un bouton de commande 59
Exercice 59
Initialiser un objet UserForm 60
Accéder aux contrôles d'un objet UserForm 62

VIII.4. Afficher un objet UserForm à partir d'un bouton d'une feuille de calcul 64

IX. Bibliographie 65

I. INTRODUCTION A EXCEL VBA

EXCEL VBA (Visual Basic pour **Application)** est un langage de programmation permettant d'utiliser du code **V**isual **B**asic pour exécuter les nombreuses fonctionnalités de l'**A**pplication EXCEL.

Un programme écrit en VBA est souvent appelé une **macro**.

Les macros permettent notamment d'automatiser des tâches répétitives réalisées sous EXCEL. Elles peuvent aussi être utilisées pour créer des boîtes de dialogue afin de rendre une application développée sous EXCEL plus conviviale.

Une macro peut être créée en utilisant l'**enregistreur de macros**, qui ne nécessite aucune connaissance du langage VBA.

Cependant une macro ainsi créée ne s'exécutera que sur un ensemble de cellules données et le code produit ne sera pas toujours très efficace. Pour pouvoir créer des macros propres à ses besoins, efficaces et interactives, il faut apprendre à programmer en VBA.

II. L'ENREGISTREUR DE MACRO

L'enregistreur de macro permet d'écrire du code VBA à partir d'opérations effectuées manuellement sous EXCEL.

II.1. ENREGISTRER UNE MACRO

Créer une macro MACRO1 qui écrit dans la cellule A1 le texte "AgroParisTech" en caractères gras et dans la cellule G1 la date du jour en caractères italiques.

Avant de commencer l'enregistrement, il faut se poser plusieurs questions :

- penser à la manière d'effectuer les opérations manuelles ;
- se demander quand commencer l'enregistrement. Dans notre cas, la sélection de la cellule A1 fait partie de l'enregistrement puisqu'on veut que le texte "AgroParisTech" se trouve dans cette cellule. Si l'enregistrement ne commence pas par la sélection de la cellule A1, le texte tapé est écrit dans la cellule active.
- se demander quand arrêter l'enregistrement.

  Question : que faire pour que la cellule active après l'exécution de la macro soit la cellule A1 ?

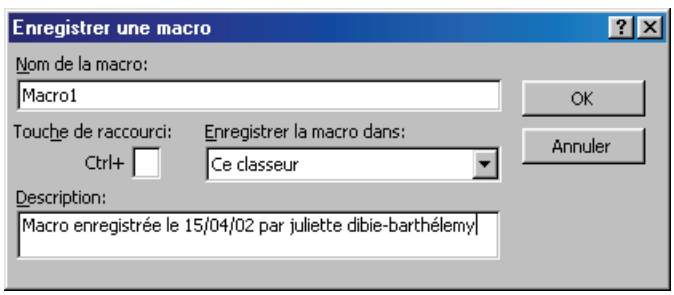
1) Ouvrir un nouveau classeur et l'enregistrer sous TEST-MACRO.XLS.

2) Les cellules dans une macro sont identifiées à l'aide d'une lettre (colonne) suivie d'un chiffre (ligne), comme par exemple la cellule A1. Afin de faciliter la lecture du code VBA généré, il est préférable de choisir la même identification des cellules dans le classeur dans lequel est enregistré la macro. Si tel n'est pas le cas, activer la commande OUTILS OPTIONS, cliquer sur l'onglet GENERAL et décocher la case STYLE DE REFERENCE L1C1.

3) Positionner le curseur sur une autre cellule que la cellule A1 de la feuille de calcul FEUIL1.

4) Activer la commande OUTILS MACRO NOUVELLE MACRO.

5) Taper MACRO1 dans la zone NOM DE LA MACRO.



6) Dans la zone ENREGISTRER LA MACRO DANS, sélectionner l'option CE CLASSEUR ; la macro MACRO1 n'est alors disponible que dans le classeur TEST-MACRO.XLS.

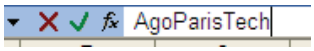
Remarque : La zone ENREGISTRER LA MACRO DANS permet de déterminer l'endroit où la macro MACRO1 sera enregistrée. Il existe trois options :

- Avec l'option CLASSEUR DE MACROS PERSONNELLES, la macro est enregistrée dans un classeur spécial appelé PERSO.XLS. Ce classeur est ouvert automatiquement lors du lancement d'EXCEL et ses macros sont donc disponibles dans tous les classeurs ouverts. La commande FENETRE AFFICHER permet d'afficher ce classeur et la commande FENETRE MASQUER de le masquer.
- L'option NOUVEAU CLASSEUR permet d'enregistrer la macro dans un nouveau classeur.
- L'option CE CLASSEUR permet d'enregistrer la macro dans le classeur actif.

7) Cliquer sur le bouton OK.

8) La barre d'outils ARRET DE L'ENREGISTREMENT apparaît, ce qui marque le début de l'enregistrement. Par défaut, l'enregistrement d'une macro s'effectue en utilisant des **références absolues aux cellules**.

Remarque : Il est possible d'enregistrer une macro en utilisant des références relatives, en cliquant sur le bouton REFERENCE RELATIVE de la barre d'outils ARRET DE L'ENREGISTREMENT.

9) Positionner le curseur sur la cellule A1, taper le texte "AgroParisTech", valider avec le bouton √ 

et sélectionner le style gras (**G**).

10) Positionner le curseur sur la cellule G1, taper la formule =AUJOURDHUI(), valider avec le bouton √ et sélectionner le style italique (***I***).

11) Arrêter l'enregistrement en cliquant sur le carré bleu de la barre d'outils ARRET DE L'ENREGISTREMENT.

NE JAMAIS OUBLIER D'ARRETER L'ENREGISTREMENT D'UNE MACRO.

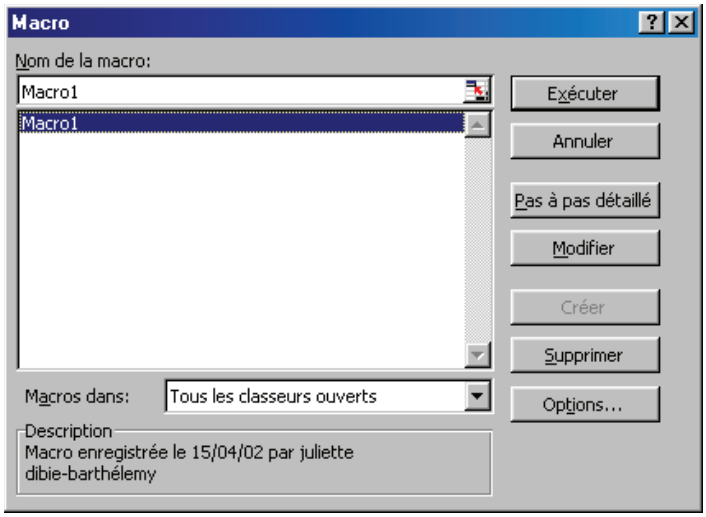
II.2. EXECUTER UNE MACRO

1) Effacer le contenu des cellules A1 et G1.

2) Positionner le curseur sur une autre cellule que la cellule A1 de la feuille de calcul FEUIL1.

3) Activer la commande OUTILS MACRO MACROS.

4) Sélectionner la macro MACRO1 et cliquer sur le bouton EXECUTER.



POUR SUPPRIMER UNE MACRO, ACTIVER LA COMMANDE OUTILS MACRO MACROS, SELECTIONNER LA MACRO A SUPPRIMER ET CLIQUER SUR LE BOUTON SUPPRIMER.

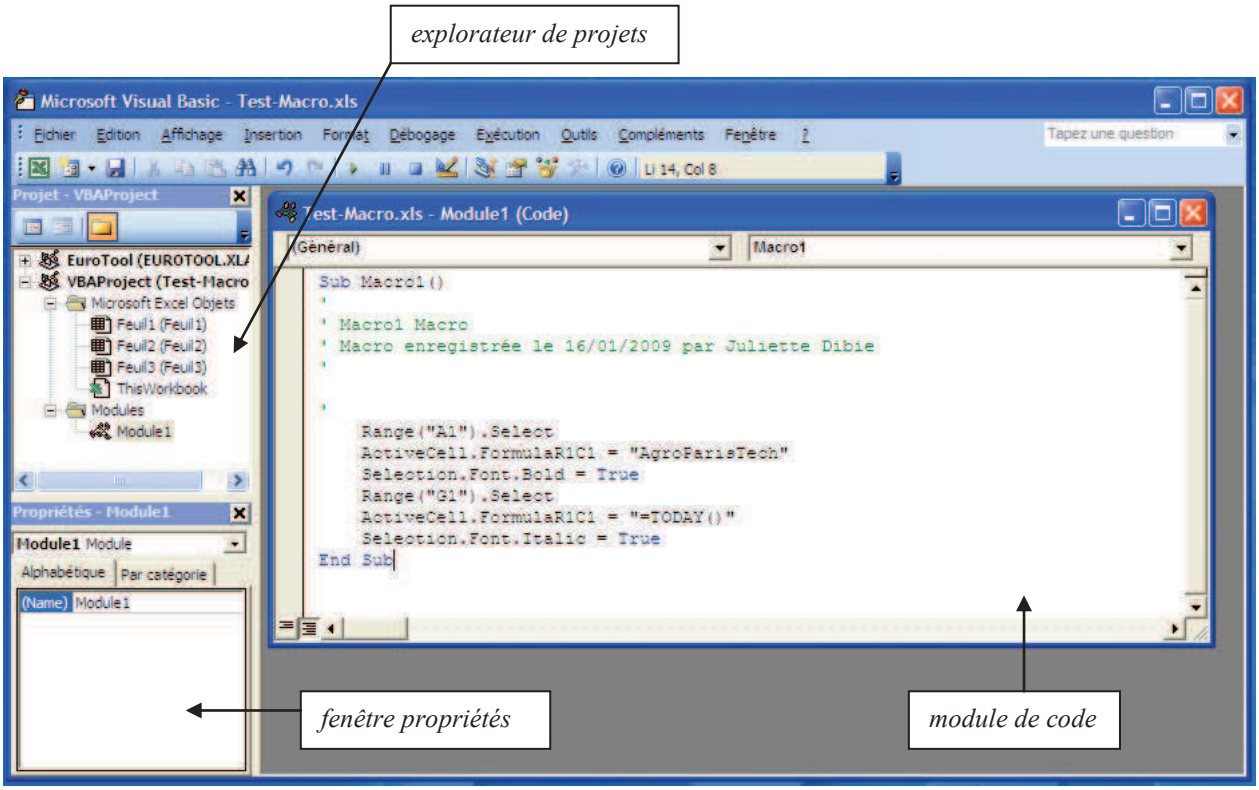
II.3. EXERCICE

Créer une macro MACROMOIS qui écrit les noms de mois Janvier, Février et Mars en colonne à partir de la cellule active. Utiliser l'enregistreur de macro avec référence relative aux cellules.

III. L'ENVIRONNEMENT VISUAL BASIC EDITOR

Nous avons vu comment créer une macro à l'aide de l'enregistreur de macro. Nous allons maintenant examiner le code VBA produit. Pour ce faire, il faut utiliser l'éditeur de Visual Basic, **VISUAL BASIC EDITOR (VBE)**, qui s'exécute dans une fenêtre différente de celle d'EXCEL.

Ouvrir VBE en activant la commande AFFICHAGE BARRE D'OUTILS VISUAL BASIC et en cliquant sur l'objet VISUAL BASIC EDITOR.



L'EXPLORATEUR DE PROJETS

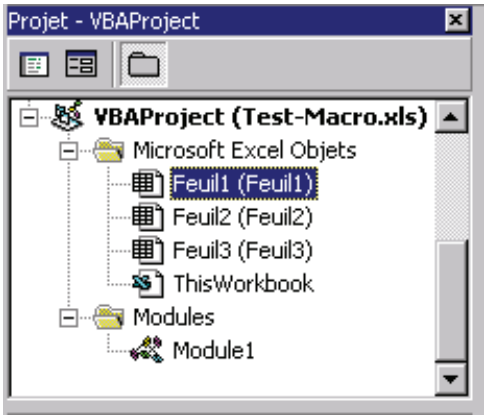
A chaque classeur EXCEL ouvert est associé un **projet VBA**.

L'**explorateur de projets** affiche une liste hiérarchisée des différents projets VBA associés aux classeurs EXCEL ouverts.

Un projet VBA associé à un classeur regroupe les éléments du classeur, comme ses feuilles de calcul ou des boîtes de dialogue, et les procédures et les fonctions associées au classeur et stockées dans un ou plusieurs modules de code.

Le projet VBA associé au classeur TEST-MACRO.XLS est composé de deux dossiers :

- le dossier MICROSOFT EXCEL OBJETS qui contient les éléments attachés au projet : le classeur TEST-MACRO.XLS (THISWORKBOOK) et ses feuilles de calcul FEUIL1, FEUIL2 et FEUIL3 ;
- le dossier MODULES qui contient les modules de code du projet : le module MODULE1 qui contient la macro MACRO1.



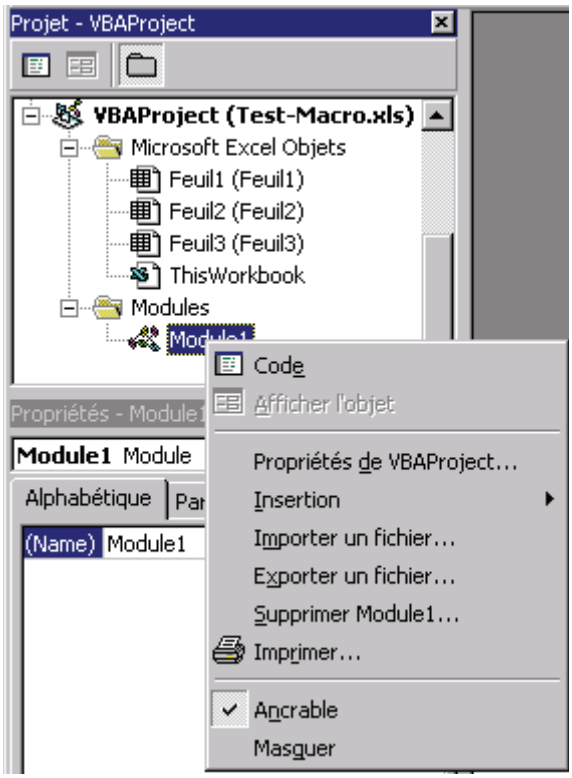
SAUVEGARDER REGULIEREMENT SON TRAVAIL A L'AIDE DE LA COMMANDE FICHIER ENREGISTRER "NOMCLASSEUR.XLS".

LA COMMANDE FICHIER FERMER ET RETOURNER A MICROSOFT EXCEL PERMET DE FERMER VBE ET DE RETOURNER DANS EXCEL.

III.2. LES MODULES DE CODE

L'explorateur de projets permet, à l'aide du clic droit de la souris, d'ouvrir un module de code (option CODE), d'insérer un nouveau module de code (option INSERTION) ou d'en supprimer un (option SUPPRIMER <NOMMODULE>).

La commande INSERTION MODULE permet également d'insérer un nouveau module de code.



Toutes les macros sont enregistrées dans un module de code. L'enregistreur de macro a inséré le module de code MODULE1 qui contient la macro MACRO1. Un module de code peut contenir plusieurs macros. On peut insérer autant de modules de code qu'on le désire.

III.3. LES PROCEDURES

Une macro est appelée en VBA une **procédure**, c'est une suite d'instructions qui ne retourne pas de valeur.
VBA permet également d'écrire des **fonctions**. Une fonction est une suite d'instructions qui retourne une valeur. Nous les étudierons plus loin.

L'enregistreur de macro ne génère que des procédures. Une procédure commence par le mot clé Sub suivi du nom de la procédure et d'une liste d'arguments entre parenthèses, qui peut être vide. Elle se termine par le mot clé End Sub. Une procédure a la syntaxe suivante :

```
Sub NomProcédure([argument_1,..., argument_n])
   Instructions
   ...
End Sub
```

Remarque : Les crochets **[ ]** signifient que les arguments sont facultatifs.

Une **instruction** exécute une tâche précise. Elle est généralement écrite sur une seule ligne.
Exemple :

```
Range("A1").Select
```