Editions ENI

VBA Excel 2016

Programmer sous Excel : Macros et langage VBA

Collection
Ressources Informatiques

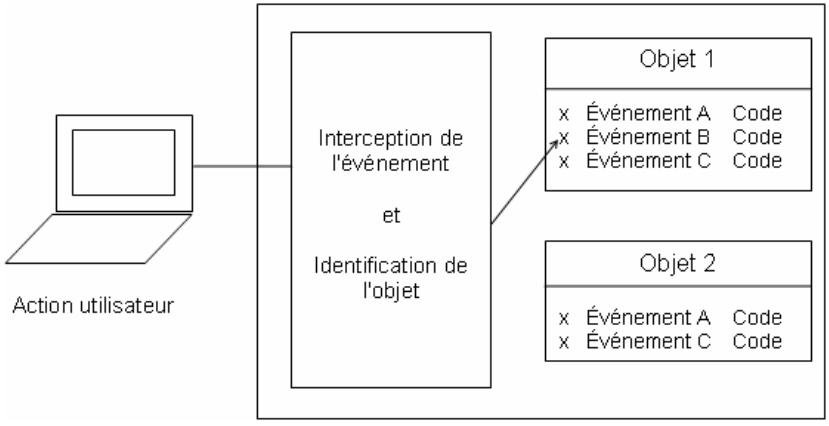
Extrait

Chapitre 9
Gestion des événements

1. Présentation

Un événement est une **action** utilisateur ou système reconnue par un objet Microsoft Excel. Il déclenche la procédure événementielle associée à l'événement de l'objet activé.

Les procédures événementielles vous permettent d'associer un code personnalisé en réponse à un événement qui se produit sur un objet Excel (classeur, feuille, formulaire, graphique...).



2. Écriture des événements

2.1 Événements de classeur, de feuille ou de formulaire

Vous pouvez accéder aux procédures événementielles associées à un objet de la façon suivante :

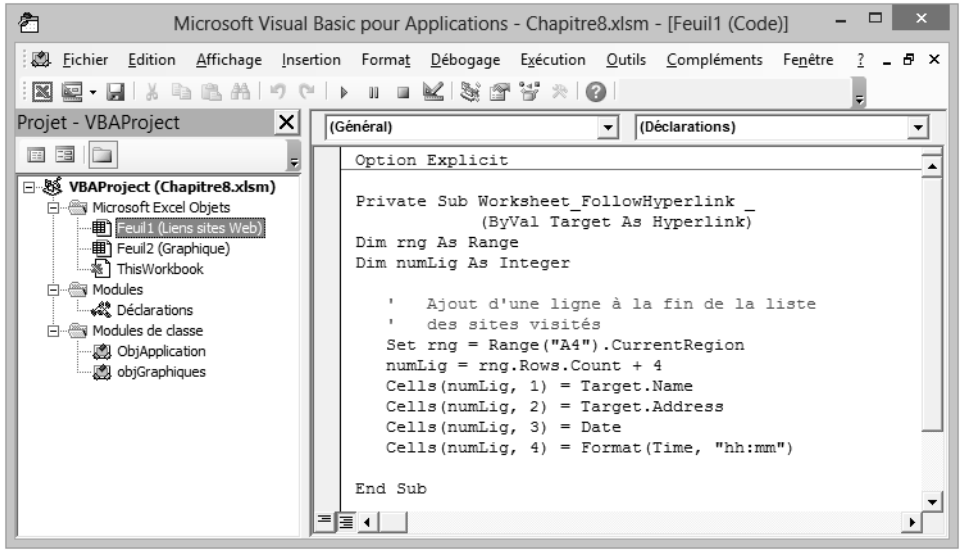
- Dans la fenêtre **Explorateur de projet**, double cliquez sur l'objet souhaité (classeur, feuille, ou formulaire) afin de faire apparaître la fenêtre de code correspondante.
- Ouvrez la liste déroulante de gauche de la fenêtre de code et sélectionnez **Workbook**, **Worksheet** ou **UserForm** en fonction de l'objet sélectionné.
- Vous pouvez alors sélectionner l'un des événements liés à l'objet sélectionné dans la liste déroulante de droite afin de lui associer un code personnalisé.

Remarque

*Vous pouvez à tout moment désactiver l'exécution des procédures événementielles en affectant **False** à la propriété **EnableEvents** de l'objet **Application**.*

Exemple

Cet exemple montre comment obtenir l'historique de tous les liens hypertextes qui ont été visités à partir de la feuille de calcul active.



Résultat dans Excel :

	A	B	C	D	E	F	G	H	I
1									
2		**LISTE DES LIENS VISITÉS**				**LISTE DES LIENS DISPONIBLES**			
3									
4	**NOM**	**ADRESSE**	**DATE**	**HEURE**		Microsoft			
5	Microsoft	http://www.microsoft.com/	08/02/2013	12:09		Microsoft France			
6	SNCF	http://www.sncf.fr/	08/02/2013	16:20		Editions ENI			
7	Air France	http://www.airfrance.fr/	09/02/2013	18:24		Boursorama			
8	Boursorama	http://www.boursorama.com/	10/12/2013	09:32		Météo			
9	Météo	http://www.meteo.fr/	10/02/2013	16:45		SNCF			
10	Boursorama	http://www.boursorama.com/	11/02/2013	13:43		Air France			
11	Boursorama	http://www.boursorama.com/	12/02/2013	16:10					
12	Microsoft France	http://www.microsoft.fr/	13/02/2013	13:44					
13									

Liens sites Web | Graphique

2.2 Événements de l'objet Application

Trois étapes sont nécessaires à l'écriture et à l'exécution des événements de l'objet **Application**.

Étape 1

- Insérez un module de classe :
  Insertion - **Module de classe**
  ou ouvrez la liste  et cliquez sur **Module de classe**.
- Une fois le module inséré, nommez-le.

Exemple

Nommez le module de classe ObjApplication.

Étape 2

- Dans le module de classe, créez un objet **Application** par le code suivant :

```
Public WithEvents NomObjet As Application
```

Exemple

Création de l'objet appelé **oMonAppli** en tant qu'application.

```
Public WithEvents oMonAppli As Application
```

L'objet ainsi créé devient disponible dans la liste de gauche du module.

- Sélectionnez l'objet ainsi créé dans la liste de gauche du module puis sélectionnez l'événement attendu dans la liste de droite. Écrivez le code des procédures à générer.

Exemple

Création de deux procédures événementielles : la première concerne l'insertion d'une nouvelle feuille, la seconde la création d'un nouveau classeur.

```
Public WithEvents oMonAppli As Excel.Application
_____________________________________________________________
Private Sub oMonAppli_WorkbookNewSheet _
           (ByVal Wb As Workbook, ByVal Sh As Object)
   Dim oNomFeuille As String
   '  A chaque ajout de feuille, on demande à l'utilisateur
   '  de saisir un nom qui sera ensuite affecté à la feuille insérée
   '  après les feuilles existantes
   oNomFeuille = InputBox("Entrez le nom de la feuille")
   ActiveSheet.Name = oNomFeuille
   ActiveSheet.Move After:=Sheets(Sheets.Count)
End Sub
_____________________________________________________________
Private Sub oMonAppli_NewWorkbook(ByVal Wb As Workbook)
   Dim iNbFeuilles As Integer
   Dim iNbActuel As Integer
   Dim iDifférence As Integer
   '  Pour chaque nouveau classeur,
   '  on demande à l'utilisateur le nombre de feuilles
   '  Suivant les cas, on ajoute ou on supprime des feuilles
   Do
      iNbFeuilles = Application.InputBox(Title:="", _
                  Prompt:="Nombre de feuilles ?", Default:="3", Type:=1)
   Loop While iNbFeuilles = False
   iNbActuel = Sheets.Count
   iDifférence = iNbActuel - iNbFeuilles
   '   Suppression des feuilles en trop
   '   Suppression des messages d'alerte afin de ne pas
   '   avoir de message lors de la suppression de feuilles
   Do While iDifférence > 0
      Application.DisplayAlerts = False
      Sheets.Item(iDifférence).Select
      ActiveWindow.SelectedSheets.Delete
      iDifférence = iDifférence - 1
   Loop

   '   Ajout de feuilles si nécessaire
   '   Les événements sont désactivés afin de ne pas avoir
   '   à saisir le nom des nouvelles feuilles
   Do While iDifférence < 0
      Application.EnableEvents = False
      Sheets.Add
```

```
        iDifférence = iDifférence + 1
    Loop
    '   Réactivation des événements et alertes
    Application.EnableEvents = True
    Application.DisplayAlerts = True
End Sub
```

Étape 3

- Activez un module quelconque et connectez l'objet déclaré dans le module de classe avec l'objet **Application** par les instructions suivantes :

```
Dim oNomVariable As New NomModuleDeClasse

Sub NomProcédure ()
Set oNomVariable.NomObjet = Application
End Sub
```

Exemple

Ajoutez le code suivant dans le module Déclarations.

```
Option Explicit
Dim oApp As New ObjApplication
_____________________________________________________________
Sub InitializeMonAppli()
   Set oApp.oMonAppli = Application
End Sub
```

Enfin appelez la procédure InitializeMonAppli lors de l'ouverture du classeur (module de classe ThisWorkbook).

```
Private Sub Workbook_Open()
   InitializeMonAppli
End Sub
```

Lorsque ce classeur sera ouvert, les procédures événementielles créées au cours de l'étape 2 s'exécuteront automatiquement lors de l'ajout de classeurs ou de feuilles. Ces procédures seront désactivées à la fermeture du classeur.

Editions ENI

VBA Excel 2016

Créez des applications professionnelles

Exercices et corrigés

Collection
Les TP Informatiques

Extrait

Chapitre 4
Structures de contrôle

Durée : 1 heure 55

Mots-clés

condition, choix, test, alternative, branchement conditionnel, compteur, itération, incrémenter, décrémenter, sortie

Objectifs

Maîtriser les structures de décision afin de tester des conditions puis effectuer des actions différentes selon le résultat obtenu. Maîtriser les instructions d'itération qui, associées aux instructions conditionnelles, permettent d'écrire du code Visual Basic pour la prise de décision et la répétition des actions. Vous retrouverez ces structures dans la suite du livre. Dans ce chapitre, nous nous limitons à l'emploi des boîtes de dialogue déjà rencontrées.

Prérequis

Pour valider les prérequis nécessaires, avant d'aborder le TP, répondez aux questions ci-après (certaines questions peuvent nécessiter plusieurs réponses) :

1. Les structures suivantes sont des structures de décision :
   - **a.** `If ... Then ... Else ... End If`
   - **b.** `Do ... Loop`
   - **c.** `Select Case ... Case ... End Select`
2. Résultat = `IIf(7 / 2 > 3, IIf(2.8 * 3.3 < 11, "X", "Y"), "Z")`
   La variable `Résultat` contient la valeur :
   - **a.** `X`
   - **b.** `Y`
   - **c.** `Z`

3. Les instructions suivantes appartenant chacune à une structure de contrôle différente sont correctes :

a. `Case If N1 > N2`

b. `Case A, B, C`

c. `Case 1 to 10`

d. `Case Nombre, Is > 50`

4. Le mot-clé `ElseIf` :

a. peut apparaître de suite après une clause `Else`.

b. est facultatif.

c. peut être utilisé plusieurs fois dans un bloc `If`.

5. Répétition d'instructions tant qu'une condition a la valeur `True` :

a. `For Each ... Next`

b. `For ... Next`

c. `Do ... Loop`

d. `While ... Wend`

e. `With ... End With`

6. Utilisation d'un compteur pour exécuter des instructions un certain nombre de fois :

a. `For Each ... Next`

b. `For ... Next`

c. `Do ... Loop`

d. `While ... Wend`

e. `With ... End With`

7. Répéter un groupe d'instructions pour chaque élément d'un tableau ou d'une collection :

a. `For Each ... Next`

b. `For ... Next`

c. `Do ... Loop`

d. `While ... Wend`

e. `With ... End With`

8. Répéter un groupe d'instructions le nombre de fois indiqué :

a. `For Each ... Next`

b. `For ... Next`

c. `Do ... Loop`

d. `While ... Wend`

e. `With ... End With`

9. Exécuter une série d'instructions appliquées à un seul objet ou à un type défini par l'utilisateur :

a. `For Each ... Next`

b. `For ... Next`

c. `Do ... Loop`

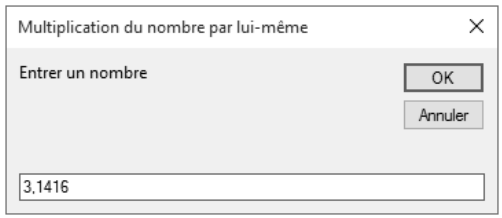
d. `While ... Wend`

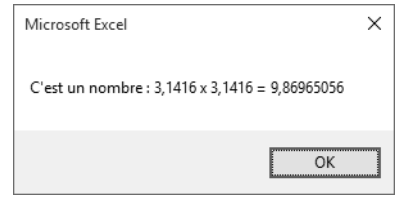
e. `With ... End With`

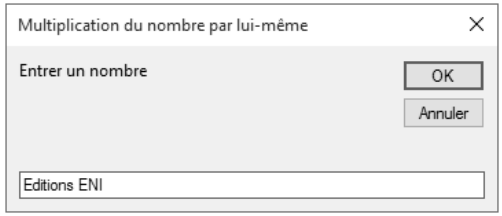
Corrigé p. 249

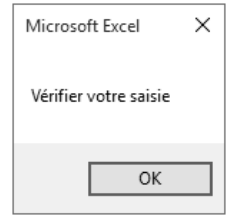
Énoncé 4.1 Vérifier que la saisie est numérique

Exercice 1
Durée estimative : 10 minutes
Créez la procédure **ContrôleSaisie** qui propose une boîte de saisie et vérifiez qu'il s'agit d'un nombre. Si c'est le cas, effectuez l'opération qui consiste à multiplier le nombre par lui-même, sinon affichez un message. Exemple :





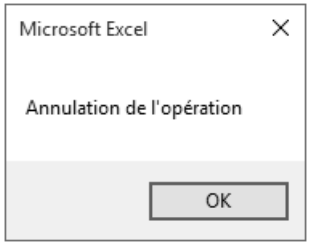




Indice

Utilisez la fonction `IsNumeric` *pour tester la saisie.*

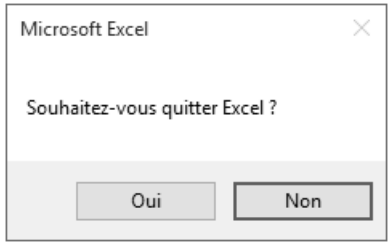
Exercice 2
Durée estimative : 5 minutes
Complétez la procédure précédente qui vérifie que l'utilisateur a cliqué sur le bouton **OK**. Dans le cas contraire, affichez un message. Exemple :

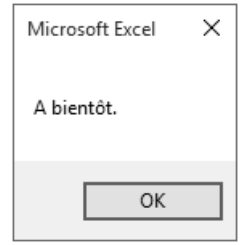


Corrigé p. 249

Énoncé 4.2 Dire au revoir

Durée estimative : 5 minutes
Complétez la procédure **AuRevoir**. Celle-ci propose une boîte de dialogue demandant si l'on souhaite quitter Excel. Elle affiche le message "A bientôt" si l'utilisateur clique sur le bouton **Oui**, et le message "Poursuivons" sinon. Exemple :





```
Sub AuRevoir()
  Dim strMessage As String
  Dim intStyle As Integer
  Dim intChoix As Integer
  strMessage = "Souhaitez-vous quitter Excel ?"
  intStyle = vbYesNo + vbDefaultButton2
  ...  = MsgBox(strMessage, intStyle)
  If ... = vbYes Then
    MsgBox "A bientôt."
  ...
    MsgBox "Poursuivons."
  ...
End Sub
```

Indice

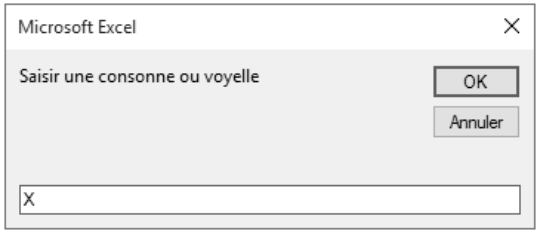
Voici un extrait de l'aide VBA en ligne :

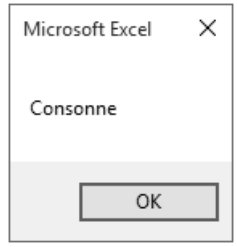
Constante	Valeur	Description
vbOKOnly	0	Bouton **OK** uniquement (par défaut)
vbOKCancel	1	Boutons **OK** et **Annuler**
vbAbortRetryIgnore	2	Bouton **Abandonner**, **Réessayer** et **Ignorer**
vbYesNoCancel	3	Boutons **Oui**, **Non** et **Annuler**.
vbYesNo	4	Boutons **Oui** et **Non**

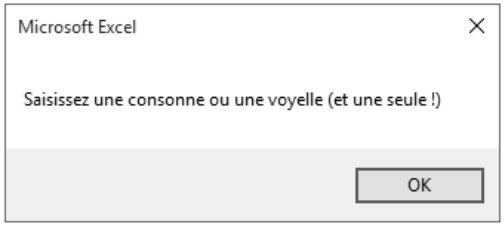
Corrigé p. 250

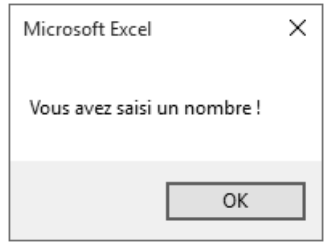
Énoncé 4.3 Contrôler la saisie d'une consonne ou voyelle

Durée estimative : 10 minutes
Créez la procédure **ConsonneVoyelle** qui demande de saisir une voyelle ou une consonne. Affichez un message selon la saisie : "Voyelle", "Consonne" ou un message approprié en cas d'erreur. Exemple :









Indice

Pour tester la lettre en majuscule, on effectuera la transformation suivante :

```
varChoix = VBA.UCase(varChoix)
```

Corrigé p. 250

Énoncé 4.4 Afficher un message selon l'âge et le genre

Durée estimative : 15 minutes
Créez la procédure **VotreAge** qui demande le sexe et l'âge de l'utilisateur. Les messages seront différents selon qu'il s'agisse d'un homme ou d'une femme. Prenez également en considération l'abandon de la saisie et la saisie de lettres pour l'âge.
0 - 17 ans : Gamin
18 - 30 ans : Vous êtes jeune
31 - 50 ans : Vous êtes encore jeune
> 50 ans : Vous commencez à vieillir