Excel 2000 et VBA

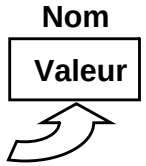
VB et VBA pour Excel :

Conclusion, révisions et extensions

Philippe Pasquier

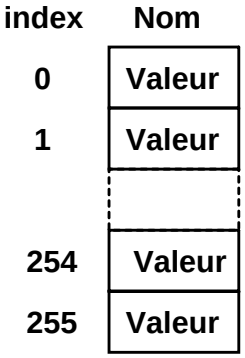
Rappels d'algorithmique

- **Nous avons vu :**
  - **Définition : Un algorithme est une suite finie d'instructions qui, étant donné des paramètres typés, calcule une solution d'un problème donné**
  - **Variables**
  - **Constantes**
  - **Opérateurs, fonctions**
  - **Expressions**
  - **Une instruction : l'affectation**
  - **Une méthode de programmation utilisant tout ça : la programmation événementielle**
  - **Un environnement de programmation (Visual Studio 6)**



Rappels d'algorithmique

- **Notion de tableau :**
  - **Collection de valeurs d'un même type**
  - **Accessibles par indexation**
  - **1 ou plusieurs dimensions**
  - **De taille statique (fixe) ou dynamique**
  - **Généralement parcourus ou traités à l'aide de boucles**



Les tableaux et VBA

- **Exemples :**

```
Sub tablo1()
    'déclaration d'un tableau d'entiers
    Dim mytab(1 To 5) As Integer

    'initialisation du tableau
    mytab(1) = 1
    mytab(2) = 2
    mytab(3) = 3
    mytab(4) = 4
    mytab(5) = 5

    'corps de la procédure
    Worksheets(1).Range("A1").Value = "nombre :"
    Worksheets("feuil1").Range("B1") =
    Application.WorksheetFunction.Count(mytab)
End Sub
```

Les tableaux et VBA

```
Sub tablo2()
    'déclaration des variables
    Dim mytab(1 To 5) As Integer
    Dim i As Integer

    'initialisation du tableau
    For i = 1 To 5
        mytab(i) = i * i
    Next i

    'corps de la procédure
    Worksheets(1).Range("A2").Value = "Moyenne : "
    Worksheets("feuil1").Range("B2") =
      Application.WorksheetFunction.Average(mytab)
End Sub
```


Objets Range comme tableaux de cellules

```
Sub Range1()
    'déclaration des variables locales
    Dim plagelocale As Range
    'initialisation
    Set plagelocale = Worksheets(2).Range("A6:A16")
  'accès aux éléments du Range comme à ceux d'un tableau
    plagelocale(1).Value = 3
    plagelocale(2).Value = 4
    plagelocale(3) = 8          'déconseillé
    plagelocale(4) = 5
  'indique le nombre d'éléments de l'objet Range manipulé
    plagelocale(5) = plag.Count

End Sub
```


Notion de structure de contrôle

- **Branchement conditionnel :**

```
If condition Then
     [instructions]
Else
     [instructions]
End If
```

```
Select Case expression
   Case ListeValeurs1
      [Instructions]
   Case ListeValeurs2
      [Instructions]
   Case Else
      [Instructions]
End Select
```


Notion de structure de contrôle

- **Itérations et itérations conditionnelles :**

```
For Compteur = Début To Fin [Step Incrément]
   [Instructions]
Next [Compteur]
```

```
Do While Condition
   Instructions
Loop
```

```
Do Until Condition
   Instructions
Loop
```

```
Do
   Instructions
Loop While Condition
```

```
Do
     Instructions
Loop Until Condition
```

Notion de structure de contrôle

Exemples d'utilisation de l'itération :

- **Sommes :** $\sum_{i=exp1}^{i=exp2} (exp(i))$

```
résulat = 0 'élément neutre pour l'addition
For i=exp1 To exp2
 résultat=résultat + exp(i)
Next i
```

- **Produits :** $\prod_{i=exp1}^{i=exp2} (exp(i))$

```
résulat = 1 'élément neutre pour la multiplication
For i=exp1 To exp2
 résultat = résultat * exp(i)
Next i
```


Procédures et fonctions

- **Procédures :**
  - **Une procédure est un ensemble d'instructions qui participent à une même tâche.**

```
[Public/Private][Static] Sub nom([liste_arguments])
    déclarations
    instructions
End Sub
```

- **Fonctions :**

```
[Public/Private]Function Nom([liste_arguments])As type
    déclarations
    instructions
    Nom = exp_du_bon_type
End Sub
```


Passage de paramètres / arguments

```
Private Sub Affiche(Message As String)
      MsgBox(Message)
  End Sub

Private Sub cmdAllez_Click()
   Dim news As String
   News=« cool ! »
   Affiche(«Ift-20403»)        ' affiche ift-20403
   Affiche News                ' affiche cool !
   Affiche (News)              ' affiche cool !
   Call Affiche(«news»)        ' affiche news
   Call Affiche News           'erreur compilation

End Sub
```

Passage par référence

Passage par valeur

Notion de portée / durée de vie

Nous avons vu différents niveaux de visibilité :

module

```
Option Explicit                         ' force la déclaration
Public x As Integer                     ' variable publique
Dim y As Integer                        ' variable globale

Private Sub CmdLancer_Click()           ' procédure évènementielle
    Dim x As Integer                            ' variable locale
    x = 3                               ' initialisation
    y = 3                                       ' initialisation
    Call calcul(x, (y))                 ' appel de procédure
    MsgBox ("x=" & x & " y=" & y)       ' affichage
    Call calcul(x, (y))                 ' appel de procédure
    MsgBox ("x=" & x & " y=" & y)       ' affichage
End Sub                                 ' destruction de x

Private Sub calcul(w As Integer, z As Integer)
    Static count As Integer             ' variable static initialisée à 0
    count = count + 1                           ' incrémentation
    MsgBox ("appel de calcul numero:" & count) ' affichage
    w = w + 1                                   ' incrémentation
    z = z + 1                                   ' incrémentation
End Sub                                 ' destruction des identifiants z/w
```

procédure

procédure

Les variables locales / globales

```
'déclaration des variables globales
Dim plageGlobale As Range
```

Visible dans tout le module

```
Sub initialise()

  Set plageGlobale = Application.InputBox("Entrer la
  plage des cellules à compléter :", , , , , , , 8)

End Sub


Sub efface()

    'efface le contenu sans toucher à la mise en forme
    plageGlobale.ClearContents

End Sub
```


Récursivité

- Pas récursif : appel récursif
- Condition d'arrêt : bien définir le cas de base qui terminera la séquence d'appels récursifs
- Exemple de la factorielle :

```
Private Function Factorial(Byval y As Double) As Double
      If y <= 1 Then
         Factorial = 1                    'Cas de base
      Else
         Factorial = y * Factorial(y-1)   'Pas récursif
      End if
End function
```


Gestion des erreurs

- Outils pour la mise au point de programme :
  - Faciliter l'écriture du code
  - Tester le code (pas à pas, variables espionnes,…)
- Traitement des erreurs à l'exécution :
  - Certaines erreurs ne surviennent qu'à l'exécution, pas à la compilation :
    - Fichier absent
    - Division par zéro
    - Entrée de l'utilisateur erronée

Traitement des erreurs à l'exécution

- Routine de gestion d'erreur, syntaxe :

```
Private Sub nom(arguments)
   <déclarations>
  On error Goto <étiquette>
   <Traitement normal>
  Exit Sub
  étiquette :
         <Traitement erreur>
         resume/resume next/Exit Sub 'mode de retour
End Sub
```

Quitte la procédure

Ré-exécute l'instruction suivant la fautive

Ré-exécute l'instruction fautive

- Si une erreur se produit pendant traitement normal, l'exécution est routée vers l'`étiquette`.
- On peut consulter `Err.number` pour connaître la nature de l'erreur

Gestion de fichiers

- **Fichiers à accès séquentiel : fichier de données alphanumériques structuré comme un texte**
- **Structure d'enregistrement : type utilisateur constitué de champs**
- **Fichiers à accès direct / aléatoire : ensemble d'enregistrement dont on connaît la structure et donc la taille à l'avance ce qui permet l'accès direct**

VBA - Programmation orientée objet

- **Classe : type abstrait encapsulant données et méthodes**
- **Objet : instance d'une classe, comprenant,**
  - **Propriétés : données internes à l'objet**
  - **Méthodes : procédure et fonctions que l'on peut lui appliquer**
  - **Événements : des procédures événementielles peuvent être définies pour les traiter**
- **Collections : ensemble d'objets d'une même classe**
- **VBE : sorte de Visual Studio sous Excel**
- **Macros : procédures VB écrites automatiquement et que l'on peut exécuter de différentes manières**

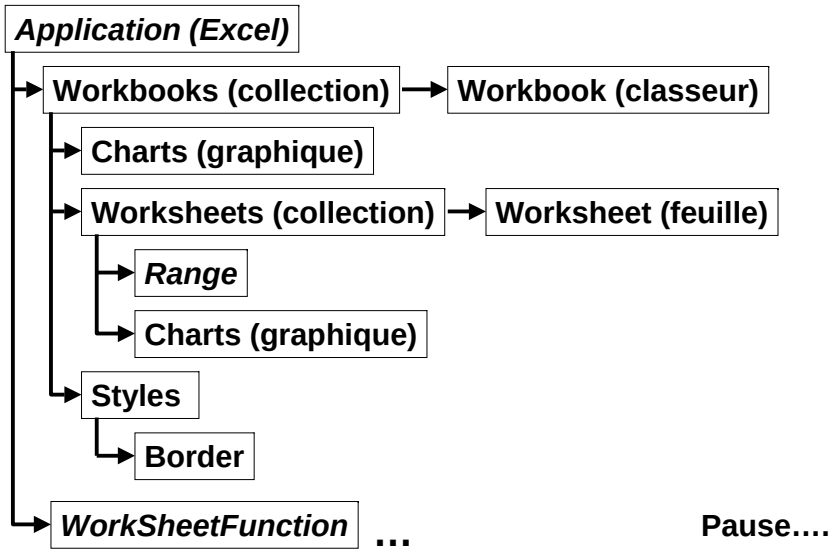
Utiliser les procédures et fonctions VBA

- **Les fonctions VBA s'ajoutent aux fonctions Excel pré-définies :**
  - **Exemple 1 : Fonction carré**
  - **Exemple 2 : Fonction MaPuiss**
  - **Exemple 3 : Fonction PrimeDeBase**
- **Les procédures doivent être appelées de manière spécifique :**
  - **Procédure = macro (on a vu pas mal de manières d'appeler une macro au dernier cours : menu, bouton de commande, bouton ▶, zone sensible d'élément graphique, …)**
  - **Exemple : exercice actuariel de Base**
- **Procédures et fonctions peuvent être appelées à partir du code VBA, comme en VB**

Utiliser les procédures et fonctions VBA

- **Il y a différentes manières de programmer une procédure ou fonction :**
  - **Lorsque rien ne le contre-indique, on peut utiliser les fonctions prédéfinies Excel :**
    - `WorkSheetFunction`
    - `.formula, .formulalocal, .formlaL1C1, …`
    - `Application.Evaluate`
    - **Directement dans la barre de formule**
  - **En écrivant le code détaillé du calcul lorsque l'on vous indique de ne pas utiliser la classe `WorkSheetFunction`**
- **Exemples : `sommeavecWorksheetFunction()`, `sommeAvecFormula()`, `SommeCommeFonction`, `SommeSansFormulaSansWorksheet()`**

Modèle Objet d'Excel - Extrait



La programmation événementielle VBA

- Cela fonctionne comme en VB classique, sauf que l'on peut utiliser les évènements liés aux objets Excel
- Un événement Excel déclenche la procédure événementielle qui lui est associée, s'il y en a une.
- Il faut sélectionner la fenêtre de code de l'élément auquel on souhaite associer le traitement (`Worksheet, Workbook`, ...) puis déterminer l'événement à traiter

Exemple

```
' déclaration d'une variable globale publique
Public NbModifs As Integer

Private Sub Worksheet_Change(ByVal Target As Range)

    'À chaque modification de cellule, le compteur est
      'incrémenté
    NbModifs = NbModifs + 1

End Sub

Private Sub Worksheet_Deactivate()

    MsgBox ("Vous avez réalisé : " & NbModifs &
  "modifications sur la feuille " & Worksheets(2).Name)
    NbModifs = 0 'réinitialise le compteur

End Sub
```


Événements de l'objet Application

- Ils sont plus complexes à faire fonctionner :
- `NewWorkBook` : un nouveau classeur est créé
- `SheetActivate` : une feuille est activée
- `WorkbookBeforePrint` : survient avant une impression
- `WorkbookBeforeSave` : survient avant l'enregistrement d'un classeur
- `WorkbookNewSheet` : une feuille est créée
- `WorkbookOpen` : un classeur est ouvert
- ...

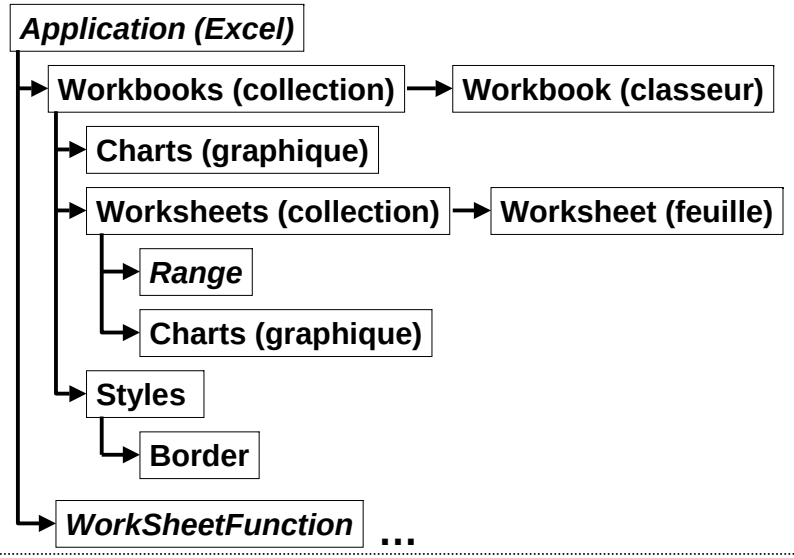
Événements de l'objet Workbook

- **`Activate` : survient quand le classeur est activé**
- **`Deactivate` : survient lorsque le classeur est désactivé**
- **`BeforeClose` : avant la fermeture**
- **`BeforePrint` : avant l'impression**
- **`BeforeSave` : avant la sauvegarde**
- **`NewSheet` : création d'une nouvelle feuille**
- **`Open` : ouverture du classeur**
- **`SheetSelectionChange` : changement de sélection sur une des feuilles**
- **…**

Événements de l'objet WorkSheet

- **`Activate` : survient lorsque la feuille est activée**
- **`Deactivate` : survient lorsque la feuille est désactivée**
- **`BeforeDoubleClic` : survient lorsque l'utilisateur double clic sur la feuille**
- **`BeforeRightClic` : survient lorsque l'utilisateur fait un clic droit dans la feuille**
- **`Calculate` : survient après le recalcul de la feuille**
- **`Change` : survient lors de la modification d'une cellule de la feuille**
- **`SelectionChange` : changement de sélection sur la feuille**
- **…**

Modèle Objet d'Excel - Extrait



Les formulaires sous VBA

- **Module de formulaire (correspond au .frm dans VB classique), interface graphique, boite de dialogue personnalisée, feuille utilisateur ou UserForm) : bien étudié pour la partie VB (Contrôles ActiveX + procédures évènementielles)**
- **Insertion dans VBE :**
  - **Dans VBE, `Insertion>UserForm`**
  - **Bouton pour l'affichage des propriétés**

Exécution et fermeture d'un formulaire

- **Méthode Show, syntaxe :**
  - **`objetUserForm.Show`**
  - **Affiche l'objet formulaire spécifié**
- **Instruction Load, syntaxe :**
  - **`Load objetUserForm`**
  - **Charge le formulaire sans l'afficher**
- **Méthode Hide, syntaxe :**
  - **`objetUserForm.Hide`**
  - **Masque le formulaire sans le décharger**
- **Instruction Unload, syntaxe :**
  - **`Unload ObjetUserForm`**
  - **Supprime le formulaire de la mémoire**

Exécution et fermeture d'un formulaire

- **Exemple :**

```
Private Sub CmdAjouter_Click()

    Static i As Integer 'COMPTEUR STATIQUE initialisé à 0

    i = i + 1 'incrémentation du compteur statique

    Worksheets(3).Columns(1).Cells(i + 1).Value =
      TxtNum.Value
    Worksheets(3).Columns(2).Cells(i + 1).Value =
      txtVal.Value

End Sub

Private Sub CmdQuitter_Click()
    End 'ferme le formulaire
End Sub
```


Extensions

- **Ce que nous ne verrons pas, mais dont il ne faut pas ignorer l'existence**
- **VB et VBA sont des outils propriétaires mais très populaire : actuariat, informatique de gestion, …**
- **S'ils sont versatiles, il ne peuvent tout faire dans leur forme originale : de nombreuses extensions existent.**
- **Nous allons en survoler quelques-unes**

Extensions

- **Contrôles ActiveX : de nombreux contrôles que nous n'avons pas vus**
- **Il est possible de développer ses propres contrôles**
- **Ressources pour le multi-média : sons, images et vidéo**
- **Gestion de Bases de Données :**
  - **Microsoft Access**
  - **SQL[Structured Query Language]**

Extensions

- **Liens entre les applications :**
  - **La technologie Automation ou OLE[Object Link and Embedding] permet de manipuler les objets d'une application à partir d'une autre**
  - **Piloter Word, Access, Outlook avec VBA ou piloter Word depuis Excel**
  - **Le protocole DDE[Dynamic Data Exchange] permet l'échange dynamique de données entre applications Windows en mode Client Serveur**

Extensions - Internet

- **Objet HyperLink permet d'insérrer des liens HTML dans vos feuilles, …**
- **Requêtes sur Internet :**
  - **Insérer un tableau provenant d'un site Internet**
  - **Plages de données externes (bases de données Access ou SQL, requêtes internet) : Objet QueryTable**
- **Export de fichiers internet :**
  - **XML[Extended MarkUp Language] : langage de description et de représentation des données**
  - **HTML[Hyper Text MarkUp Language] : langage de présentation des pages Web.**
  - **Méthode saveAs**
- **Publication de pages Web : il est possible de créer et de publier (en statique ou dynamique) une page Web à partir d'un classeur, d'une feuille, d'une plage de cellule, …**

Extensions

- **Programmation Windows :**
  - **Même si VB et VBA sont des langages très complets (car très utilisés) qui intègrent un grand nombres de fonctions systèmes**
  - **Travailler avec les API [Application Programming Interface] disponibles sous Windows. Les APIs sont des série de fonctions contenues dans des DLL[Dynamic Link Librarie].**
  - **Exemples : l'API Windows**
    - **Kerne32.dll : fonctions système de bas niveau**
    - **User32.dll : fonctions de gestion windows**
    - **GDI32.dll : fonctions de gestion des périphériques**

.Net le nouveau frameWork

- **.Net est le nouvel environnement de développement de Microsoft.**
- **C'est un environnement intrusif, très complet qui permet de développer des applications mêlant différents langages : C, C++, C#, VB.Net, …**
- **C'est plus puissant, de nombreuses nouvelles possibilités sont offertes, mais c'est plus complexe d'approche**

⇨ **VB6 et VBA seront encore utiles quelques années car le transfert n'est pas toujours possible ni souhaitable**

Questions