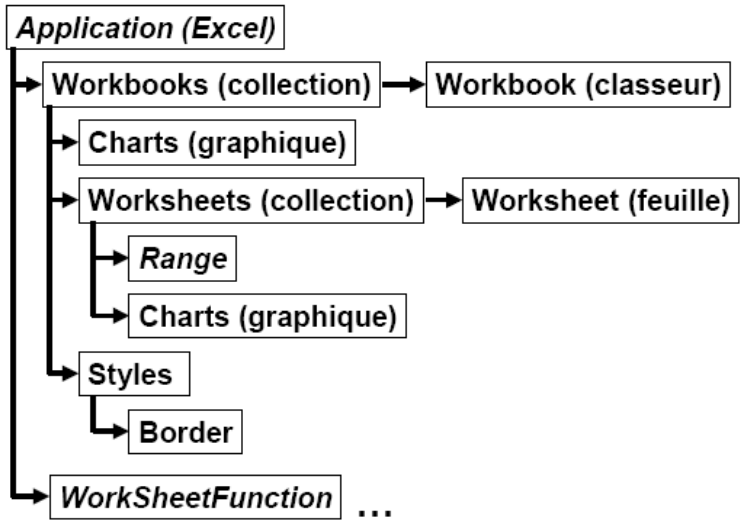
Manipulation des Objets Excel sous VBA

Définitions

- Projet : à chaque classeur Excel est associé un projet VBA regroupant tous les modules de code VB
- Module : on a vu de quoi il s'agit en VB, ils peuvent être exportés en fichiers indépendants afin d'être importés dans d'autres classeurs.

La hiérarchie des objets Excel

- VB et VBA sont des langages de POO : on peut définir des modules de classe.
- Excel a été programmé en POO : Excel est constitué d'objets.
- Extrait :



Propriétés des objets Excel

- Les objets Excel sont dotés d'attributs (ou propriétés).
- On y fait référence par : Objet.propriété
- Exemples :
  - `CmdQuitter.Enabled=True 'Rend le bouton actif`
  - `boolEtat=CmdQuitter.Enabled 'récupère l'état`
  - `Application.Cursor=xlWait 'sablier`
  - `Application.Cursor=xlDefault 'curseur standard`
  - `MsgBox Application.Version 'affiche la version de l'application Excel active`

Méthodes des objets Excel

- sont les procédures et fonctions attachées aux objets.
- On les appelles par :

```
objet.méthode argument1,argument2,…
```

- Exemples :
  - `Range(«A1:C12»).Select ‘sélectionne la plage`
  - `Selection.Clear ‘efface le contenue des cellules sélectionnées`
  - `ActiveWorkbook.SaveAs «C:\devis\devis-3.xls» ‘Enregistre le classeur actif dans un fichier.`
  - `ActiveCell.Name = «Total» ‘nomme la cellule active`
  - `Range(«B2:B45»).Name = «Total» ‘nomme la plage`

Gestion des événements

- Les objets Excel répondent à des évènements déclenchés par l'utilisateur :
  - Ouverture d'un classeur
  - Ajout d'une feuille
  - Sélection de cellules
  - Clic sur un bouton de commande
  - Changement de cellule active
  - Entrée d'une donnée, …
- On utilise les procédures évènementielles pour les traiter (lorsque nécessaire).

- Pour alléger le code :

```
With <objet>
        <code utilisant méthodes et propriétés>
End With
```

- Exemple :

```
With ActiveWorkbook
    'creé une feuille après la dernière feuille du classeur actif
    .Worksheets.Add , .WorkSheets(Worksheets.Count)
    'modifie la nouvelle feuille et renseigne A1
    With .ActiveSheet
      .Name=« synthèse »
      .Range(« A1 ») = Récapitulation des devis
    End With
End With
```

- L'instruction If TypeOf :

```
If TypeOf <objet> Is <TypeObj> Then
        <code utilisant <objet>>
End If
```

- Exemple :

```
If TypeOf obj.Parent Is Worksheet Then …
```

- L'instruction Set :

```
Set <Objet> = [New]<expression objet>
```

  - Sert à référencer des objets déjà existants ou à créer de nouveaux objets (avec New)
  - Exemple :

```
Dim Classeur As Workbook
Dim i As Integer
'création d'un nouveau classeur
Set Classeur = Application.Workbooks.Add
' affectation des noms
With Classeur
   .workSheets(1).Name=« ventes 2000 »
   .workSheets(2).Name=« ventes 2001 »
End With
```

Les objets d'Excel - Application

- L'objet application regroupe :
- Propriétés relatives à l'environnement Excel :
  - Options du menu Outils, imprimante active, …
  - Présentation de la fenêtre application : `Height` (double, hauteur de la fenêtre), `Left` (double, espace entre le bord gauche de l'écran et la fenêtre Excel), `Top` (double), …
  - Propriétés système : `MemoryFree (Long),`
  - `OperatingSystem (String)`
- Des propriétés renvoyant les objets et collections de premier niveau du modèle objet
- Des propriétés spécifiques faisant directement référence à des objets : `ActiveCell, ActiveSheet, ActiveWindow, ActiveWorkBook, ThisWorkbook, Selection`…
- Comme c'est l'objet racine, c'est l'objet par défaut et on peut l'omettre dans les notations.

Les objets d'Excel - Workbook

- Les objets de la classe Workbook représentent des classeurs Excel, ils sont membres de la collection Workbooks
- Cet objet est renvoyé par les propriétés suivantes de l'objet application :
  - `Workbooks(« nom ») ou Workbooks(index)`
  - `ActiveWorkbook`
  - `ThisWorkbook`
- Exemple :

```
MsgBox(ActiveWorkbook.Name)
```

- Méthodes importantes
  - `Activate` : rend le `Workbook` actif
  - `Add` : ajout d'un nouveau classeur
  - `SaveAs «chemin/nom»` : sauvegarde
  - `Save, Close` : sauvegarde ou ferme le classeur actif
  - `PrintOut` : imprime le classeur spécifié

    ```
    Workbooks(«classeur1»).PrinOut 'adresser un objet d'une collection.
    ```
  - `Select` : sélectionne le classeur indiqué

- Exemple :

```
Sub fermeTousSaufMoi()
   Dim classeur As Workbook
   For Each classeur In Workbooks
         IF classeur.Name<>ThisWorkbook.Name Then
               classeur.Close True
         End if
   Next classeur
End Sub
```

Les objets d'Excel - WorkSheet

- Les objets de la classe `Worksheet` représentent des feuilles de calcul, ils sont membres de la collection `Worksheets` de l'objet `Workbook`
- Propriétés qui renvoient un objet WorkSheet :
  - WorkSheets(« nom ») ou WorkSheets(index) ou Sheets(…)
  - ActiveSheet : désigne la feuille active du classeur adressé

- Propriétés :
  - Name : nom de la feuille
    - ThisWorkbook.WorkSheets(1).Name=« exemple »
  - Visible : booléen qui indique si la feuille est visible ou non
  - Protect : booléen qui indique si la feuille référée est protégée ou non
  - Tab.Color : Couleur de l'onglet de la feuille
- Exemple :

```
ActiveWorkbook.Worksheet(« Tarifs »).Tab.Color = vbRed
```

- Méthodes :
  - Activate : active la feuille désignée (cela équivaut à cliquer sur l'onglet de la feuille)
  - Calculate : provoque le calcul des cellules de la feuille de calcul spécifiée
  - Delete : supprime la feuille de calcul désignée
  - PrintOut : imprime la feuille désignée
  - Protect et Unprotect : active et désactive la protection de la feuille

Les objets d'Excel - objet Range

- L'objet Range représente l'union des cellules ou plages de cellules indiquées.
- Syntaxe :

  Range(«cel1[,cel2,…]») ou Range(«plage1[,plage2,…]»)
- Inclue comme propriété de différentes classe :
  - Aplication.Range() :désigne l'objet Range indiqué de la feuille active
  - Worksheet.Range() : désigne l'objet Range indiqué de la feuille indiquée
  - Range.Range() : désigne le Range indiqué du Range indiqué

- Propriétés des objets Range :
  - Count : nombre de cellule désigné par l'objet Range en question

    MsgBox (Range("A1:N23").Count) 'affiche 322
  - Row : renvoie le numéro de la première ligne de la plage
  - RowHeight : hauteur des lignes de la plage
  - Column : renvoie le numéro de la première colonne de la plage
  - ColumnWidth : largeur des colonnes de la plage
  - Font.Bold : propriété booléenne indiquant si les fontes des cellules sont en gras ou pas

- Propriétés des objets Range :
  - AddressLocal : référence de la plage dans le format utilisateur

    MsgBox(Selection.AddressLocal) ‘ affiche $B$4:$B$9
  - Address : référence de la plage

    MsgBox(Selection.Address) ‘ affiche $B$4:$B$9
  - Name : permet de nommer les cellules ou plage concernées

    Columns(5).Name = « total »
  - Value : indique la valeur d’une cellule (ne fonctionne que si le Range en question ne désigne qu’une cellule.

    Range(« B6 »).Value = 34000

    Range(« B6:B8 »).Value = 12 ‘ interdit : mauvaise pratique
  - Formula : indique la formule attachée à une cellule

    ActiveCell.Formula=« =Moyenne(B1:B8) »

- Méthodes des objets Range (renvoyant des objets):
  - Find (information) : renvoie un objet Range qui représente la première cellule où cette information apparaît
  - FindNext : cellule suivante
  - FindPrevious : cellule précédente

- Méthodes des objets Range (ne renvoyant pas d'objets) :
  - Activate : active la première cellule du Range concerné
  - AddComment : ajoute un commentaire aux plages correspondants au Range désigné
  - ClearComments : efface les commentaires associés aux cellules désignées
  - Clear : efface tout le contenu des cellules

    Exemple : Selection.Clear
  - ClearContents : efface le contenu des cellules
  - ClearFormats : efface le format des cellules

- Méthodes (ne renvoyant pas d'objet) :
  - Copy, Cut, Paste : copie, coupe et colle les contenus des cellules désignées
  - Justify : aligne le contenu des cellules
  - Select : sélectionne le Range considéré

    Exemple : Range(« A1:N34 »).Select

Exemple :

```
Sub CreationTablo()
        Dim i As Integer ‘déclaration d’un compteur
        With Application.ActiveSheet
                .Range("B1").Value = "Résultats annuel"
                ‘indique les mois en colonne
                For i = 1 To 12
                        .Range("A" & i + 3).Value = "mois " & i
                Next I
                ‘assigne le total
                .Range("A16").Value = "Total"
                .Range("B16").Formula = "=SUM(B3:B15)"
        End With
End Sub
```

- Propriétés renvoyant un objet Range :
  - Propriété Cells :

    Application.Cells(nl,nc) : adresse une cellule de la feuille active.
  - Range(«B1:G19»).Cells(2,1) : adresse B2
  - Exemple :

    Range(«B1:G19»).Cells(2,1) = « février »
  - <Worksheet>.Cells : adresse une cellule de la feuille considérée
  - Exemple:

    Activesheet.Cells(3,2) = « Mars » ‘modifie le contenu de la cellule B3

- méthodes renvoyant un objet Range :
  - Application.Intersect : fait l'intersection de Range
  - Application.Union : renvoie l'union des Ranges passés en arguments
  - Exemple :

```
Dim Zonetout As Range
Set Zonetout = Union(Range(«B2:B8»),Range(«B9:F12»),Range(«C4:C22»)
```

  - Range.Areas(index) : permet de désigner une zone d'une union ou intersection de plages

```
‘ met en gras la plage B9:F12
Zonetot.Areas(2).Font.Bold = True
```

- Autres propriétés renvoyant des Ranges :
  - Columns et Rows :
    - syntaxe : Columns(index) ou columns(«lettre»)
    - Retourne l'objet Range correspondant à la colonne/ligne sélectionnée
    - Existe comme propriété de trois classes :
      - Application.Columns()
      - Range.Columns()
      - Worksheet.Columns()
    - Pareil pour Rows : Application.Rows, Range.Rows, Worksheet.Rows

- Autres propriétés renvoyant des Ranges :
  - Selection : désigne l'objet Range correspondant à la sélection courante
  - Exemple : Selection.InsertIndent
  - ActiveCell : désigne l'objet Range correspondant à la cellule active de la fenêtre active ou spécifiée

Les objets d'Excel -WorkSheetFunction

- Objet contenant toutes les fonctions pré-définies d'Excel (toutes celles chargées à l'ouverture de l'application)

  - Syntaxe :
    Variable = WorksheetFunction.NomFonction(argument1,argument2,…)
    'attention au type de Variable

  - Exemples :
    - Moy=Application.WorkSheetFunction.Average(Selection)
    - MsgBox(Application.WorksheetFunction.Log10(123))
    - Range(«B1»).value= Application.WorkSheetFunction.Average(Selection)
    - Range("C16").Value = Application.WorksheetFunction.Asin(0.987)
      'les noms français ne marchent pas et le . est le séparateur de décimale.
    - Range("C16").Value = Application.WorksheetFunction.Moyenne(Selection)

Fonctions utiles

- SUM : calcule la somme des éléments
- AVERAGE : calcule la moyenne des éléments
- MDETERM : calcule le déterminant d'une matrice
- MIN : retourne le minimum des éléments
- MAX : retourne le maximum des éléments
- MEDIAN : retourne la médiane des éléments
- STDEV : calcule l'écart-type des éléments (en supposant échantillon)
- STDEVP : calcule l'écart-type des éléments (en supposant population)

- CORREL : calcule le coefficient de corrélation entre deux séries de données
- SKEW : calcule le coefficient d'asymétrie (skewness) des éléments
- KURT : calcule le coefficient d'aplatissement (kurtosis) des éléments
- Toutes les distributions (NORMDIST,
- POISSON, TDIST, GAMMADIST, ...)
- Fonctions financières
- Fonctions logiques
- etc.

Les collections

- Lorsque plusieurs objets d'une même classe cohabitent, on parle de collection.
- Référence à un objet d'une collection:
  - NomCollection!NomObjet
  - NomCollection![NomObjet]
  - NomCollection(« NomObjet »)
  - NomCollection(var), où var est une expression de type String correspondant au nom de l'objet
  - NomCollection(index) où index est le numéro d'index de l'objet dans la collection.

- Exemples :
  - Workbooks![Devis.xls].Worksheets![Feuil1].Activate
  - Workbooks!(«Devis.xls»).Worksheets!Feuil1.Activate
  - Workbooks![Devis.xls].Worksheets!(«Feuil1»).Activate
  - Parcours d'une collection :

```
Dim i as Integer
For i=1 To ActiveWorkBook.Worksheets.Count
        ActiveWorkbook.worksheets(i).Name=«DevisN» & i
Next i
```

  - Ou encore

```
Dim Feuille As Worksheet
For Each Feuille In activeWorkbook.Worksheets
        Feuille.Name=«DevisN» & Feuille.Index
Next Feuillle
```

That’s all folks ☺

Exercice

- Téléchargez le fichier Document_brut.xls à l'adresse http://www.labri.fr/perso/zemmari/Ens/M2IRE
- Ce fichier contient des informations sur les différentes commandes de produits des différents clients.
- Le but de l'exercice est d'éclater la feuille contenant l'ensemble de toutes les informations en :
  - Une feuille Clients : contenant les informations sur les clients
  - Une feuille Villes : contenant les codes postaux et les villes
  - Une feuille Commandes : avec les détails des différentes commandes
  - Une feuille Produits : les informations sur les stocks des produits.
- Écrire une procédure optimiser() permettant de réaliser ce découpage.