Les cellules sont des objets Range

Range("nom_plage") ..... Plage nommée "nom_plage"
Range("A1:B2") ..... Plage dont l'adresse est A1:B2
[A1:B2] ..... Plage dont l'adresse est A1:B2
Cells(3,5) ..... Cellule à l'intersection de la 3ème ligne et la 5ème colonne : E3
Range("A1").Offset(3,5) ..... Plage située 3 lignes en dessous et 5 colonnes à droite
Range(Range("A1"), Range("B2")) ..... Plage contiguë depuis A1 jusque B2
ActiveCell ..... Cellule active
Selection ..... Cellules sélectionnées

<objet>.Cells ..... L'ensemble des cellules de l'objet (feuille ou plage)
<feuille>.UsedRange ..... L'ensemble des cellules occupées d'une feuille
<cellule>.CurrentRegion ..... Plage continue dont fait partie la cellule
<plage>.Columns ..... L'ensemble des colonnes de la plage
<plage>.Rows ..... L'ensemble des lignes de la plage
<plage>.SpecialCells(*<type>*) ..... Plages spéciales d'une plage
xlCellTypeBlanks ..... cellules vides
xlCellTypeConstants ..... cellules contenant des constantes
xlCellTypeFormulas ..... cellules contenant des formules
xlCellTypeLastCell ..... dernière cellule dans la plage utilisée
xlCellTypeVisible ..... toutes les cellules visibles
...
<plage>.End(*<type>*) ..... Cellule située à la fin de la zone qui contient la plage
xlDown ..... vers le bas
xlRight ..... vers la droite
...
<cellule>.Value ..... Valeur de la cellule
<cellule>.Text ..... Valeur affichée de la cellule au format texte

Les feuilles de calcul sont des objets Worksheet

Worksheets ..... Collection des feuilles de calcul d'un classeur
Sheets ..... Collection de l'ensembles des feuilles d'un classeur

Worksheets("nom_feuille") ..... Feuille de calcul nommée "nom_plage"
Sheets("nom_feuille") ..... Feuille nommée "nom_plage"
<CodeName> ..... Feuille dont le nom de code est "CodeName"
Ce dernier doit être nommé dans les propriétés de la feuille dans l'éditeur.
ActiveSheet ..... Feuille active

Worksheets.Add ..... Pour ajouter une feuille de calcul
Worksheets.Count ..... Pour compter les feuilles de calcul
Sheets.Count ..... Pour compter les feuilles

<feuille>.Move ..... Déplace une feuille
<feuille>.Copy ..... Copie une feuille
<feuille>.Calculate ..... Calcule une feuille
<feuille>.Name ..... Texte de l'onglet d'une feuille
<feuille>.Visible = (*<type>*) ..... Visibilité d'une feuille
xlSheetVisible ..... la feuille est visible
xlSheetHidden ..... la feuille est masquée
xlSheetVeryHidden ..... la feuille est très masquée

Les classeurs sont des objets Workbook

Workbooks ..... Collection des classeurs ouverts

Worksheets("nom_classeur") ..... Classeur "nom_classeur"
ThisWorkbook ..... Classeur contenant le code
ActiveWorkbook ..... Classeur actif

Workbooks.Add ..... Pour créer un nouveau classeur
Workbooks.Count ..... Pour compter les classeur
Workbooks.Open *<chemin\nom_classeur>* ..... Pour ouvrir un classeur

<classeur>.Save ..... Enregistre un classeur
<classeur>.SaveAs ..... Enregistre-sous un classeur
<classeur>.Close [*True*|*False*], [*<chemin\nom_classeur>*] ..... Ferme un classeur, éventuellement en le sauvegardant ou non sous un chemin et un nom définit.
<classeur>.Activate ..... Sélection d'un classeur
<classeur>.Names ..... Renvoie la liste des noms d'un classeur
<classeur>.Name ..... Nom d'un classeur
<classeur>.FullName ..... Chemin et nom d'un classeur
<classeur>.Path ..... Chemin du dossier contenant un classeur
<classeur>.Windows ..... La collection des fenêtres d'un classeur, il y en a au moins 1
<classeur>.Windows(1).Visible = *True*|*False* ..... Visibilité d'un classeur

Excel est représenté par l'objet Application

Application.GetOpenFilenameAffiche la boîte de dialogue ***Ouvrir*** et récupère le nom du fichier sélectionné sans l'ouvrir.

Application.Quit ..... Quitter
Application.Caption ..... Lit ou écrit le texte affiché dans la barre de Titre d'Excel
Application.Dialogs(*<constante_XlBuiltInDialog>*) ..... Ouvre une boite de dialogue prédéfinies d'Excel (selon la constante indiquée)

Application.DisplayAlerts = *True*|*False* ..... True masque les messages d'alerte
Application.ScreenUpdating = *True*|*False* ..... False ne rafraîchit plus l'écran
Application.WorksheetFunction. *<fonction_Excel>*Permet d'utiliser la plupart des fonctions d'Excel
Application.Evaluate .. Convertit une chaîne en sa valeur ou en l'objet auquel elle fait référence.

Application.CutCopyMode=False ..... Sort du mode couper ou copier

Application.Intersect(*<plage>*,*<plage>*,...) ..... Renvoie une plage située à l'intersection des plages indiquées.
Application.Union(*<plage>*,*<plage>*,...) ..... Renvoie une plage réunissant les plages indiquées.

VBA - Excel

Principales instructions
Collections — Objets
Conseils

L'informatique à votre portée
L'informatique prêt-à-porter

Formation ~ Développement
Bureautique & Web

Liste thématique des principales instructions

© Cathy Astuce

Déclarations

```
Dim | Private | Public | Static <Variable> As <Type_de_Données>

[Private | Public] Sub <Nom_Procedure>
    <Instructions>
End Sub
[Private | Public] Function <Nom_Fonction>
    <Instructions>
End Function

Dim | Private | Public <Collection> As [New] Collection
Dim | Private | Public  Const <Constante> As <Type_de_Données> = <valeur>

Type <nom_type_perso>
    <element1> As <Type_de_Données>
    <element2> As <Type_de_Données>
    …
End Type
```

Tableau

```
Dim | Private | Public | Static <Tableau> As Variant
Dim | Private | Public | Static <Tableau>(<n>) As <Type_de_Données>
Dim | Private | Public | Static <Tableau>(<n1>,<n2>,...) As <Type_de_Données>

Redim  <Tableau>(<n>)
Redim  Preserve <Tableau>(<n>)

LBound(<Tableau>,<n> )
UBound(<Tableau>,<n> )

Join(<Tableau>[, separateur])

<Tableau> = Split( <texte>, separateur])
```

Définit ou redéfinit en cours de procédure , le tableau se vide entièrement lors de cet instruction.

Preserve conserve les valeurs existantes.

Déterminer la taille d'une dimension dans un tableau

Concatène les valeurs texte des éléments du tableau en insérant un séparateur

Découpe le texte au séparateur et met le résultat dans un tableau

Affectations

```
<Variable> = <Valeur>

<Variable1> = <Variable2>

Set <Variable_objet> = <objet>

Set <Variable_objet> = [New] <type_objet>

Set <Variable_objet> = Nothing

<Tableau> = Array( <valeur1>, <valeur2>, ...)

<Tableau>(n) = <valeur>
```

Pour vider la variable et libérer la mémoire

Si … Alors … Sinon …

```
If <Instructions> Then <Une_Instruction>

If <condition> Then
    <Instructions>
End If

If <condition> Then
    <Instructions>
Else
    <Instructions>
End If

If <condition> Then
    <Instructions>
ElseIf <condition> Then
    <Instructions>
ElseIf <condition> Then
    <Instructions>
...
```

Si une condition est vraie alors on exécute les instructions

Si une condition est vraie alors on exécute des instructions, sinon d'autres exécutions

Si … Alors … Sinon Si … Sinon …
Équivalent à plusieurs Si imbriqués

Selon la valeur

```
Select Case <variable>
    Case <valeur>
    <Instructions>
    Case <valeur1>, <valeur2>, ...
        <Instructions>
    Case <valeur1> To <valeur2>
        <Instructions>
    Case Is < <valeur>
        <Instructions>
    Case Is <= <valeur>
        <Instructions>
    Case Is > <valeur>
        <Instructions>
    Case Is >= <valeur>
        <Instructions>
    Case Else
        <Instructions>
```

Chaque **Case** peut s'appliquer à

- Une valeur
- Une liste de valeurs
- Une plage de valeurs
- Une position par rapport à une valeur
- Une position par rapport à une variable

Conditions possibles

= *égal* < *inférieur* <= *inférieur ou égal* > *supérieur*
>=*supérieur ou égal* <> *différent*

IsNumeric(<variable>), IsDate(<variable>), IsObject(<variable>), IsArray(<variable>), IsNull(<variable>), IsEmpty(<variable>), IsMissing(<variable>)

Bloc Objet

```
With <Objet>
    .<propriété | méthode>
    ...
    <instructions>
End With

With <Variable_Objet>
    .<propriété | méthode>
    ...
    <instructions>
End With
```

Faire et Boucler

```
Do
    <Instructions>
Loop Until <condition>
Do Until <condition>
    <Instructions>
Loop

Do
    <Instructions>
Loop While <condition>
Do While <condition>
    <Instructions>
Loop
```

Jusqu'à ce que la condition soit vrai

La boucle est exécutée au minimum 1 fois

La boucle peut ne pas être exécutée

Tant que la condition est vrai

La boucle est exécutée au minimum 1 fois

La boucle peut ne pas être exécutée

Pour … Suivant

```
For <compteur> = <valeur_depart> To <valeur_fin> [Step <pas>]
    <Instructions>
Next [<compteur>]

For Each <objet> In <Collection_Objet>
    <Instructions>
Next [<objet>]
```

La boucle est effectuée n fois où
n =(<valeur_fin> - <valeur_depart>) / <pas>

La boucle est effectuée pour chaque objet de la Collection

Sortir d'une boucle ou d'un bloc

```
Exit Do
Exit For
Exit Sub
Exit Function
```

Relations avec l'utilisateur

MsgBox [<*Message*>],[<*Bouton*>]+[<*Icone*>]+[<*Bouton_defaut*>],[<*Titre*>]

- <*Bouton*> ..... vbOKOnly ..... Ok seulement
- ..... vbOKCancel ..... Ok et Annuler
- ..... vbAbortRetryIgnore ..... Abandonner, Réessayer, Annuler
- ..... vbYesNoCancel ..... Oui, Non et Annuler
- ..... vbYesNo ..... Oui et Non
- ..... vbRetryCancel ..... Réessayer et Annuler
- <*Icone*> ..... vbCritical ..... Critique (croix rouge)
- ..... vbExclamation ..... Stop avec Point d'exclamation
- ..... vbInformation ..... i bleu
- ..... vbQuestion ..... Point d'interrogation
- <*Bouton_defaut*>.. vbDefaultButton1 ..... 1er bouton déjà sélectionné
- ..... vbDefaultButton2 ..... 2ème bouton déjà sélectionné
- ..... vbDefaultButton3 ..... 3ème bouton déjà sélectionné

<*Valeur*> = MsgBox([<*Message*>],[<*Bouton_Icone*>]+[<*Icone*>]+[<*Bouton_defaut*>], [<*Titre*>])

- <*Valeur*>..... vbOK ..... Clic sur Ok
- ..... vbCancel ..... Clic sur Annuler
- ..... vbAbort ..... Clic sur Abandonner
- ..... vbRetry ..... Clic sur Réessayer
- ..... vbIgnore ..... Clic sur Ignorer
- ..... vbYes ..... Clic sur Oui
- ..... vbNo ..... Clic sur Non

<*Valeur*> = InputBox [<*Message*>],[<*Titre*>],[<*Valeur_par_défaut*>]

- <*Valeur*>..... "" ..... Clic sur Annuler
- ..... "<*Un_texte*>" ..... Clic sur Ok

Erreurs

On Error Goto <*Etiquette*>
On Error Resume Next
On Error GoTo 0

En cas d'erreur :
le code suivant l'étiquette Etiquette est exécuté,
le code continu sur la ligne suivante,
désactive le traitement des erreurs

Err.Number
Err.Description

N° et Description de l'erreur générée

Procédures évènementielles

Private Sub Workbook_Open()
Private Sub Workbook_BeforeClose(Cancel As Boolean)
Private Sub Workbook_BeforePrint(Cancel As Boolean)

Cancel = True
permet d'annuler le traitement habituel de l'évènement.

Private Sub Worksheet_Activate()
Private Sub Worksheet_SelectionChange(ByVal Target As Range)
Private Sub Worksheet_Change(ByVal Target As Range)
Private Sub Worksheet_BeforeDoubleClick(ByVal Target As Range, Cancel As Boolean)
Private Sub Worksheet_BeforeRightClick(ByVal Target As Range, Cancel As Boolean)

Target représente la plage ayant provoqué l'évènement

Conseils

Poser le problème

Toujours commencer par une analyse de ce que l'on veut faire. Mieux vous connaîtrez les possibilités d'Excel, plus votre code sera concis.

Les variables

Déclarez toujours vos variables avant de les utiliser et donnez leur un type le plus précis possible.
Préciser le type après chaque variable.
Minimisez toujours la portée de vos variables, afin qu'elles n'existent que là où elles sont utiles.
Libérer la mémoire dès que possible.
Mettez dans des variables les propriétés dont vous avez besoin dans les boucles.

Les fonctions

En général, passez des paramètres et n'utilisez pas des variables publiques.
N'oubliez pas qu'une fonction (ou une procédure peut modifier) les paramètres passés ByRef (par défaut) et donc retourner des valeurs par ses arguments. Si vous ne voulez pas risquer de les modifier, utilisez ByVal.
Demandez-vous toujours ce que vous avez besoin de récupérer.

Les objets

Utilisez des blocs With lorsque vous avez besoin d'accéder à plusieurs propriétés ou méthodes d'un objet.
Mettez dans des variables les objets que vous appelez fréquemment.
Certaines propriétés ou méthodes d'un objet renvoient un objet.

L'application

Pensez à désactiver le mode de calcul et la mise à jour d'écran et éventuellement les alertes.
Remettez toujours l'objet Application dans l'état où il était.
Désactivez les évènements quand vous n'en avez pas besoin.

Sélection

Sauf lorsque c'est indispensable, n'utilisez pas les méthodes Select et Activate.
Lorsque vous avez plusieurs objets feuilles ou classeurs, utilisez des variables ou leur nom de code pour les utiliser. Nommez vos plages.

Utilisation d'un modèle

Pour simplifier la programmation, on utilise en général un modèle situé sur une feuille cachée (voir très cachée). Il est souvent inutile de programmer toute la mise en forme d'un document, alors que l'on peut appeler un modèle contenant déjà :

* La mise en page
* Les formules de calculs
* Les graphes
* Les plages nommées

Lors de la création, nommez des plages spécifiques afin de s'y référer sans aucun problème et permettre à l'utilisateur de les déplacer, sauf dans le cas où vous ne connaissez pas leur taille.
Utilisez des noms clairs et significatifs.

L'enregistreur de macros

Une bonne méthode pour découvrir un nouvel objet, une propriété ou une méthode.
N'enregistrez qu'un petit nombre d'action (5 maxi) afin de bien les identifier.
Pensez à alléger ce code. Certaines lignes générées sont totalement inutiles.

Portée

La portée indique à quel endroit du projet un élément (une constante, une variable, une fonction ou une procédure) est utilisable ; ce peut être à des niveaux plus ou moins restreints :

* au niveau général du projet (dans tous les modules)
* au niveau d'un unique module
* au niveau d'une seule procédure ou fonction.

La portée se définit avec les mots-clefs Public ou Private ou omis.
En l'absence d'indication, la portée d'une constante ou d'une variable est équivalente à Private (limitée au niveau de la déclaration : un module , une procédure ou une fonction), celle d'une procédure ou fonction est équivalente à Public.
Ces mots-clefs se place soit :

* avant une déclaration ou une liste de déclarations de constante, en haut d'un module devant Const
* au début de la déclaration ou d'une liste de déclarations de variable, en haut d'un module à la place de Dim
* avant la déclaration d'une procédure ou d'une fonction,

Private
Indique une portée au niveau du module
Public
Indique une portée au niveau du classeur.
Une procédure ou une fonction Public peut être appelée depuis le code d'un autre classeur à l'aide de la commande
Application.Run 'nom du classeur"!Nom_du_module.Nom_procedure

Bien configurer l'éditeur de macros

Dans le menu Outils/Options de l'éditeur, il est conseillé :

* De cocher Déclarations de variables obligatoires dans l'onglet Éditeur
* De modifier les couleurs des textes suivants dans l'onglet Format de l'éditeur :
  - Texte du mot clef
  - Texte de l'identificateur

Identifier les erreurs

Utilisez la lecture de votre code en mode Pas à Pas : touche F8.
Ajoutez des points d'arrêt à l'intérieur de votre code : placez votre curseur dans votre code, puis touche F9.
Ajoutez des arrêts dans votre code avec l'instruction Stop. Touche F8 ou F5 pour poursuivre.
Ajoutez des espions sur les variables, objets ou portion de code : sélectionnez, puis clic-droit et ajouter un espion
Utilisez l'objet Debug qui affiche ce que vous lui demandez dans la fenêtre Exécution.
Debug.Print <*variable*>|<*expression*>

Liste des principaux objets et collections

© Cathy Astuce

Type de données

Collection	Objet	Définition
Borders	Border	Bordures d'une cellule
Charts	Chart	Feuilles graphique d'un classeur
ChartObjects	ChartObject	Graphiques incorporées dans une feuille de calcul
	Characters	Les caractères d'un texte d'un objet (cellule,…)
Comments	Comment	Commentaires de cellules
CustomViews	CustomViews	Affichages personnalisées
Dialogs	Dialog	Boîte de dialogue d'Excel
Filters	Filter	Les filtres d'un plage filtrée automatiquement
	Font	Police d'un objet (cellule,…)
FormatConditions	FormatCondition	L'ensemble des mises en forme conditionnelles d'une plage
HyperLinks	HyperLink	Les liens hypertexte d'une plage ou d'une feuille de calcul
	Interior	Intérieur ou Fond d'un objet (cellule,…)
Names	Name	Les noms définies d'une feuille de calcul ou d'un classeur
	Outline	Plan dans une feuille de calcul
	PageSetup	Description de la mise en page d'une feuille
PivotTables	PivotTable	Les Tableau Croisé Dynamique d'une feuille de calcul
PivotFields	PivotField	Les champs d'un TCD
PivotItem	PivotItem	Les éléments d'un champ d'un TCD
PivotItemList	PivotItem	Les éléments d'un TCD
PivotFormulas	PivotFormula	Les formules d'un TCD
QueryTables	QueryTable	Tableaux établi à partir de données externe
Range	Range	Une plage (1 ou plusieurs cellules) La propriété cells renvoie une collection Range.
Shapes	Shape	Tous les objets (dessin, image, contrôle de formulaire, …)
Sheets	Worksheet ou Graph	Les feuilles d'un classeur, quelque soit son type
	Validation	La validation de données d'une plage
Windows	Window	Les fenêtres ouvertes de l'application ou d'un classeur
Workbooks	Workbook	Les classeurs ouverts
Worksheets	Worksheet	Les feuilles de calcul d'un classeur.

Catégorie	Type	Taille (octet)	Limites	Explication	Préfixe
Numérique	Byte	1	0 ⇒ 255		bt
	Integer	2	-32 768 ⇒ 32 767	Entier	i
	Long	4	-2 147 483 648 ⇒ 2 147 483 647		l
	Single	4	-3,402823 E38 ⇒ -1,401298 E-45 et 1,401298 E-45 ⇒ 3,402823 E38	Virgule flottante à simple précision	sng
	Double	8	1,7976931348623 E308 ⇒ -4,94065645841247 E-324 et 4,94065645841247 E-324 ⇒ 1,79769313486231 E308	Virgule flottante à double précision	d
	Currency	8	-922 337 203 685 477,5808 ⇒ 922 337 203 685 477,5807	Virgule fixe avec 4 décimales	c
	Decimal	12	+/-79 228 162 514 264 337 593 543 950 335 si entier +/-7,922816251426433759354395033 5 si décimales (28)	On ne peut déclarer un type Decimal, il faudra la déclarer en Variant puis la convertir en décimal avec CDec. Utile pour les calculs exigeant un grand nombre de chiffres et ne pouvant tolérer d'erreurs d'arrondi.	b
Caractères	String	1 par caractère	2^{31} caractères maximum	La taille est variable	s
	String *x	x	2^{16} caractères maximum	La taille est fixe	
Booléen	Boolean	2	True ou False		b
Variant	Variant	16 si nombre 22 + longueur du texte		La variable n'est pas définit lors de sa déclaration mais lors de son affectation. Une même variable peut donc passer d'un type à un autre. Elle peut également contenir les valeurs Empty , Error, Nothing et Null	v
Objet	Object				o (généraliste) ou autre

Thème	Collection, Objet
Affichage personnalisé	CustomViews, CustomView
Bordures	Borders, Border
Cellule	Range, Cells, ActiveCell, Target (proc. évènementielles)
Classeur	Workbooks, Workbook, ThisWorkbook, ActiveWorkbook
Commentaires	Comments, Comment
Données externes	QueryTables, QueryTable
Fenêtre	Windows, Window, ActiveWindow
Feuille	Sheets, Worksheets, Worksheet, ActiveSheet
Filtre automatique	Filters, Filter
Fond d'une cellule	Interior

Thème	Collection, Objet
Format conditionnel	FormatConditions, FormatCondition
Graphique	Charts, Chart, ChartObjects, ChartObject, ActiveChart
Impression	PageSetup
Intérieur d'une cellule	Interior
Lien hypertexte	HyperLinks, HyperLink
Nom défini	Names, Name
Plan	Outline
Police	Font
TCD	PivotTables, PivotFields, PivotItems, PivotItemList PivotTable, PivotField, PivotItem
Validation de données	Validation