UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Introduction à VBA

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

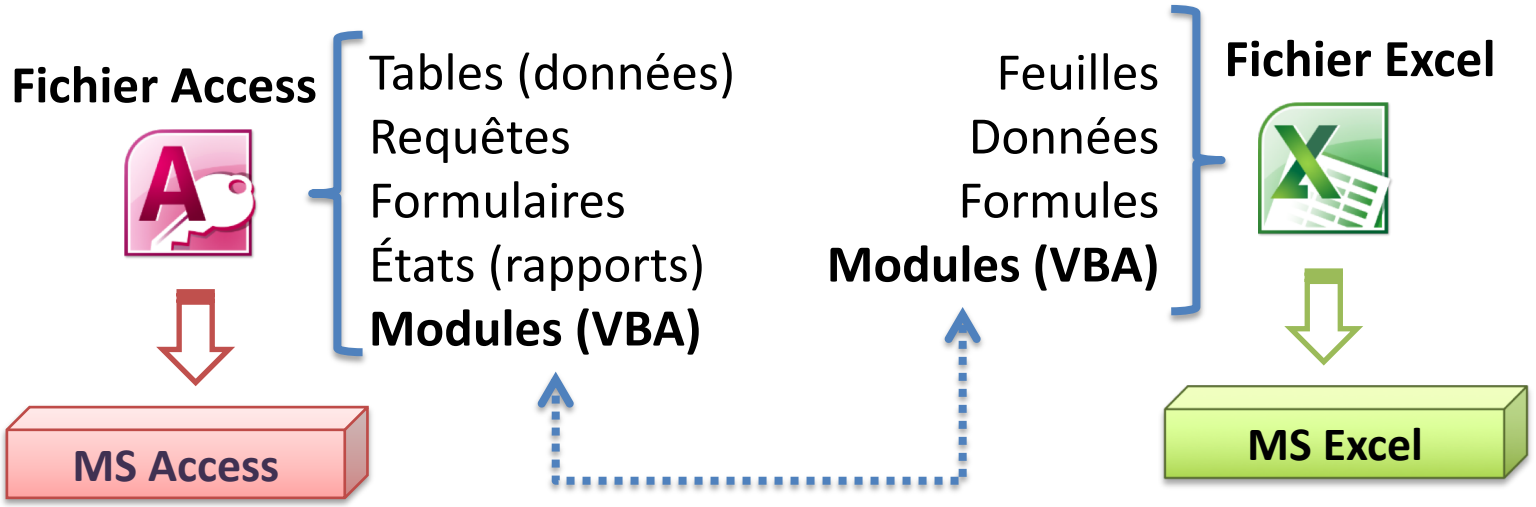
VBA : quoi & pourquoi ?

- VBA : quoi ?
  - **Langage** et **environnement** de programmation Orienté Objets
  - **Attaché aux documents** MS Office
- VBA : pourquoi ?
  - Associer **un comportement actif** à des documents Office
    - Calculs, vérifications, nettoyage des données, etc.

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : documents MS Office

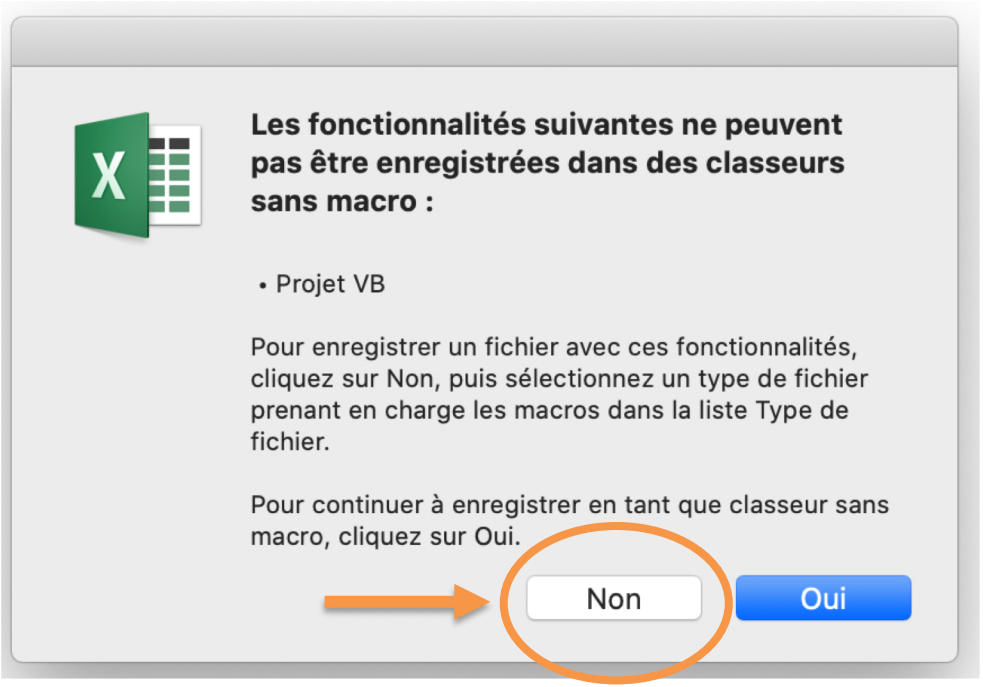
- Un document MS Office est composé des plusieurs éléments…



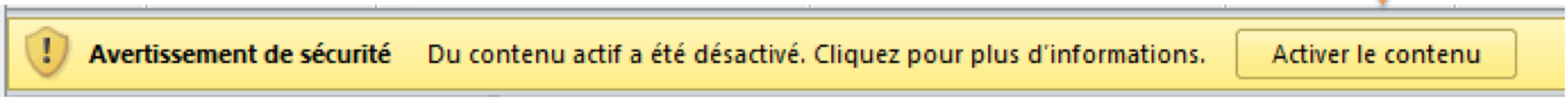
UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : documents MS Office

- Format de **fichier actif**
  - Les fichiers sont enregistrés dans un format spécial
  - **.xlsm**, **.accdb** ou encore **.docm**
  - A l'ouverture, on doit **autoriser le contenu**

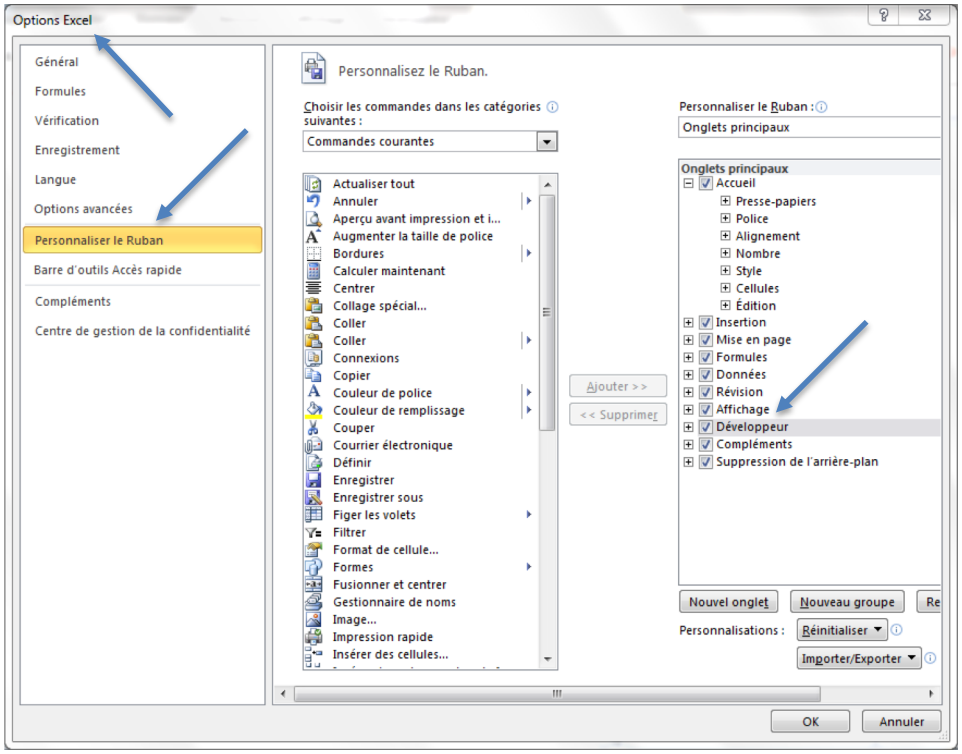


Attention aux risques de sécurité !
Ne ***jamais activer le contenu*** d'un document dont on n'est pas sûr de la provenance.



UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

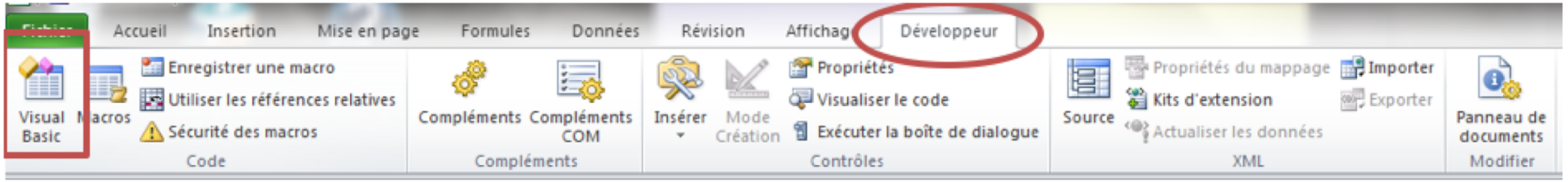
VBA : menu développeur



Pour travailler sur **VBA** et enregistrer des **macros**, il faudra utiliser le **ruban « développeur »**

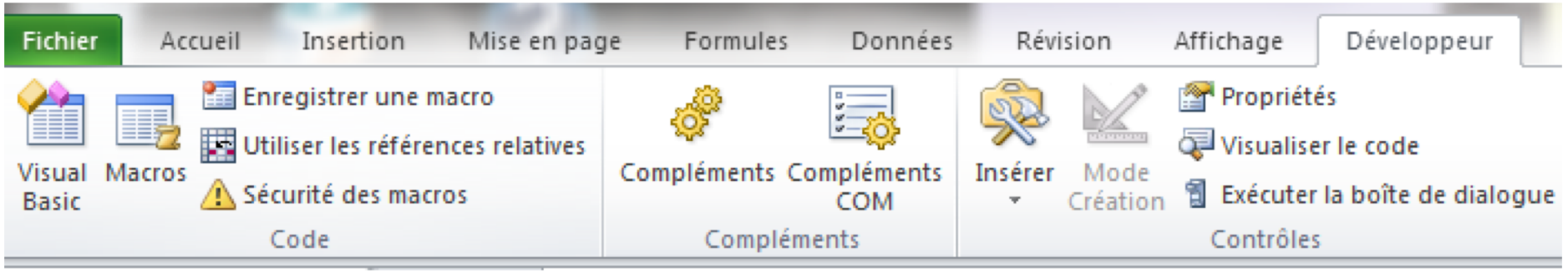
Pour activer le ruban « développeur »
« Options » → « Personnaliser le Ruban » et cocher la case « Développeur »

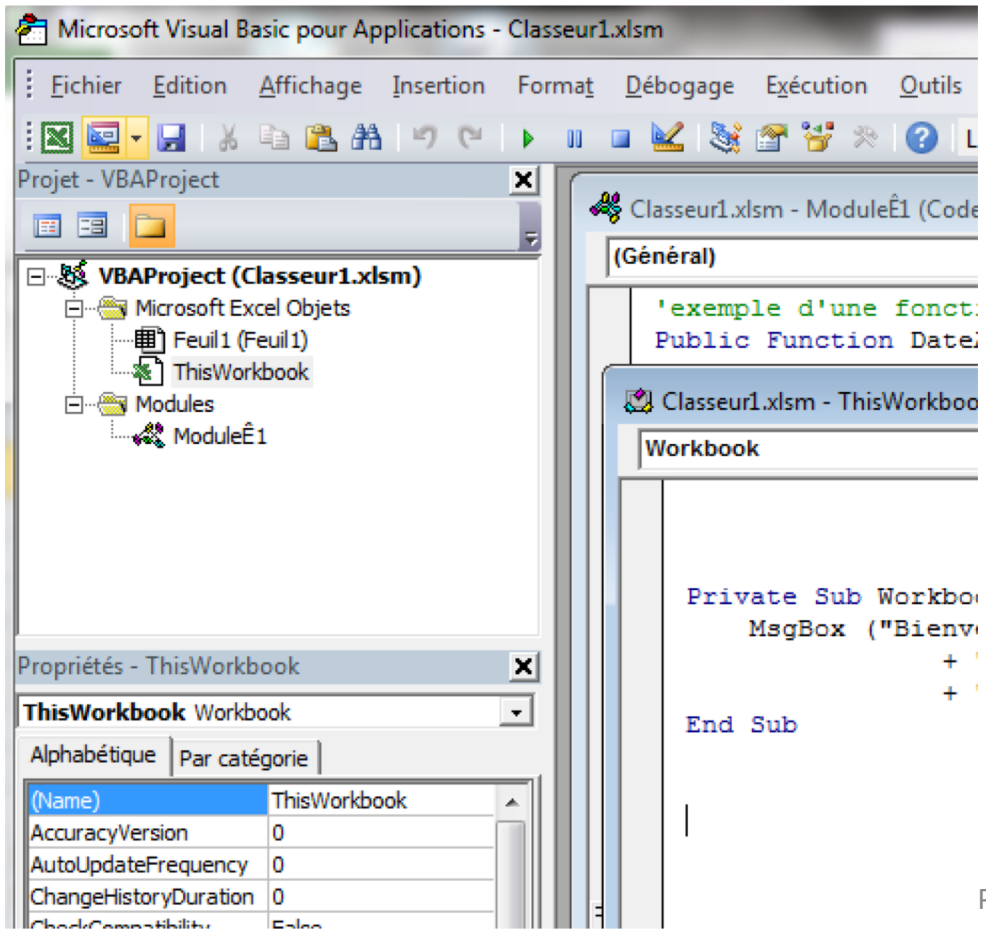
À partir du ruban « développeur »
On a accès à l'environnement VBA (VBA environnement → VBE)



UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT

VBA : environnement





- **Environnement de programmation (VBE)**
  - Accès aux codes VBA associés au fichier
  - Les codes produits par les macros s'y trouveront aussi

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

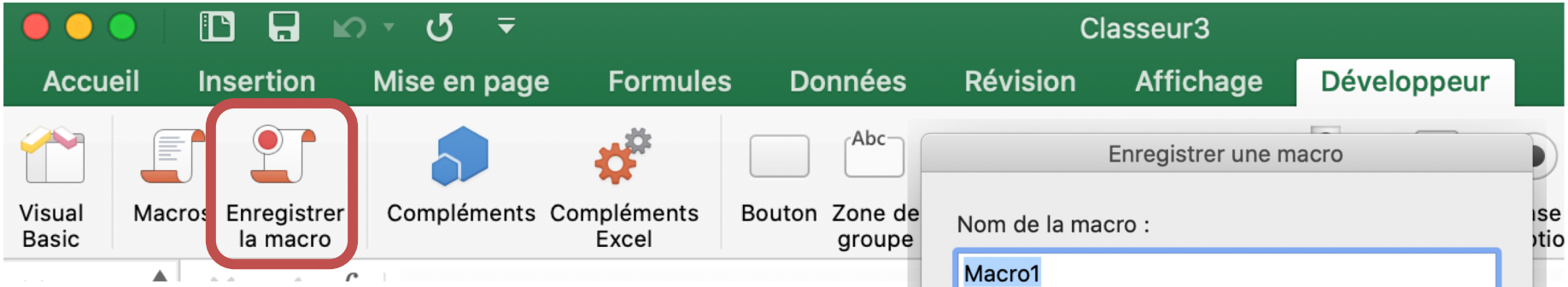
Macros

- **Macros** :
  - suite d'opérations réalisées de manière séquentielle
  - permettent de reproduire à identique une ensemble d'actions réalisées
- **Fonctionnement :**
  - **Enregistrement**
    - Office observe et enregistre chaque action réalisée
  - **Exécution**
    - Office reproduit chaque action réalisée
- **Attention** :
  - La reproduction étant à l'identique, on est souvent amené à modifier / adapter le code

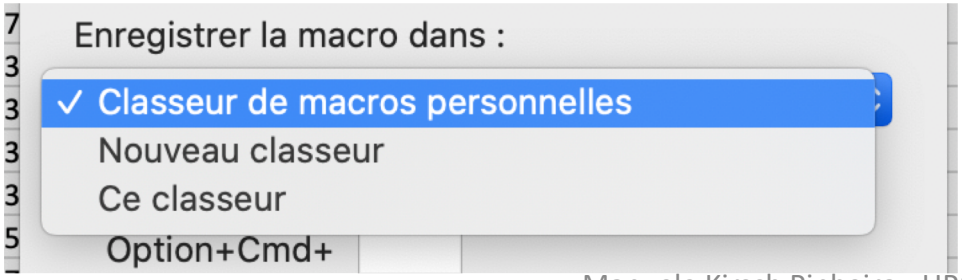
UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

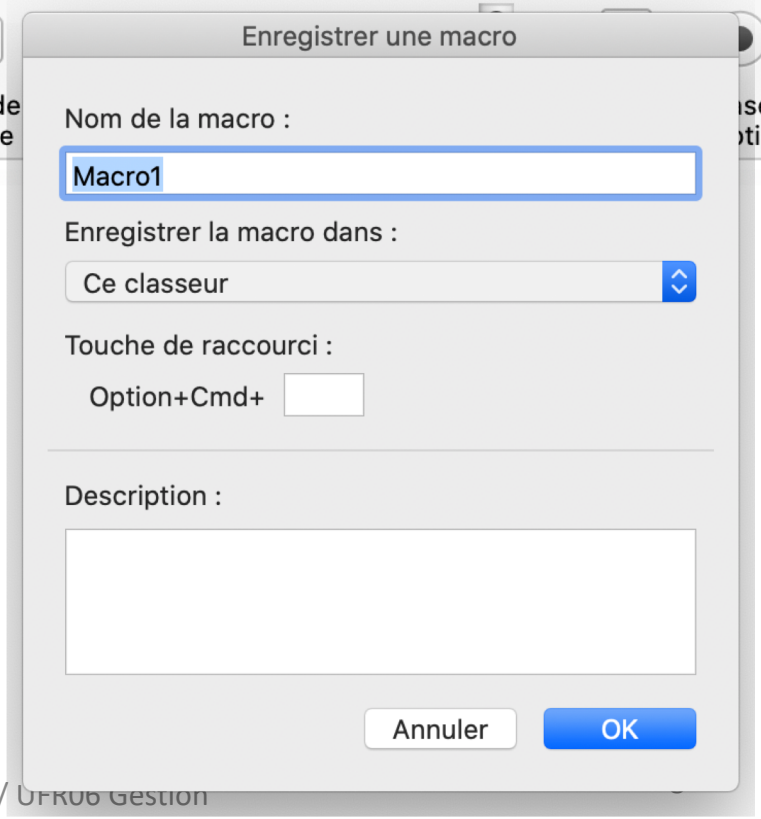
Macros

- **Enregistrement**
  - À partir du ruban « développeur »



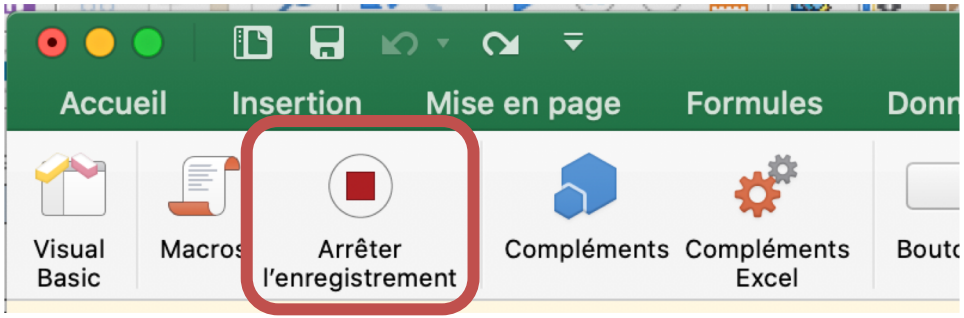
On peut enregistrer dans …
- Le classeur lui-même
- Le classeur de macros personnelles

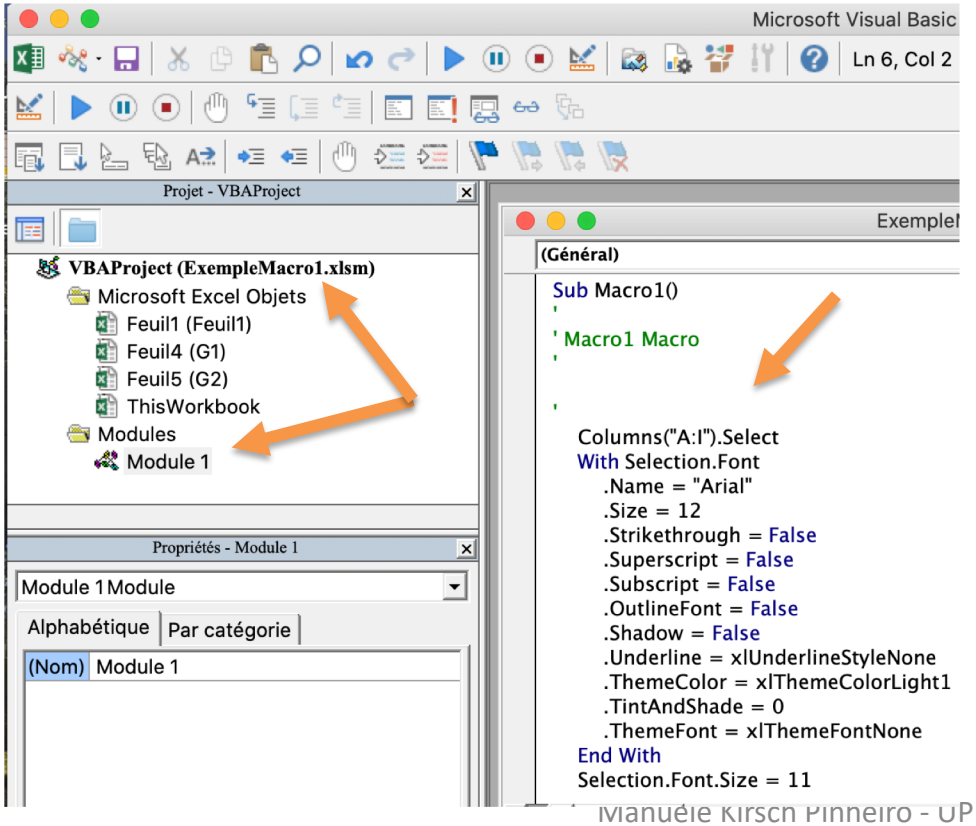




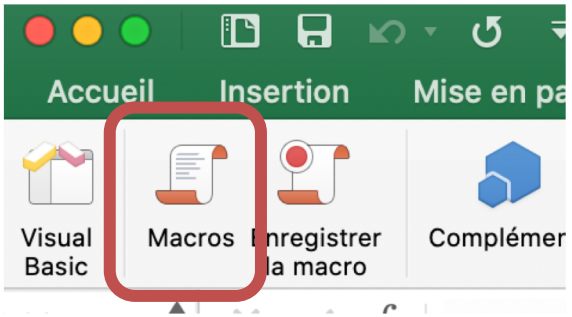
UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Macros





Toutes les **actions sont consignées** dans un code VBA, ce qui permet une reproduction **à l'identique**.



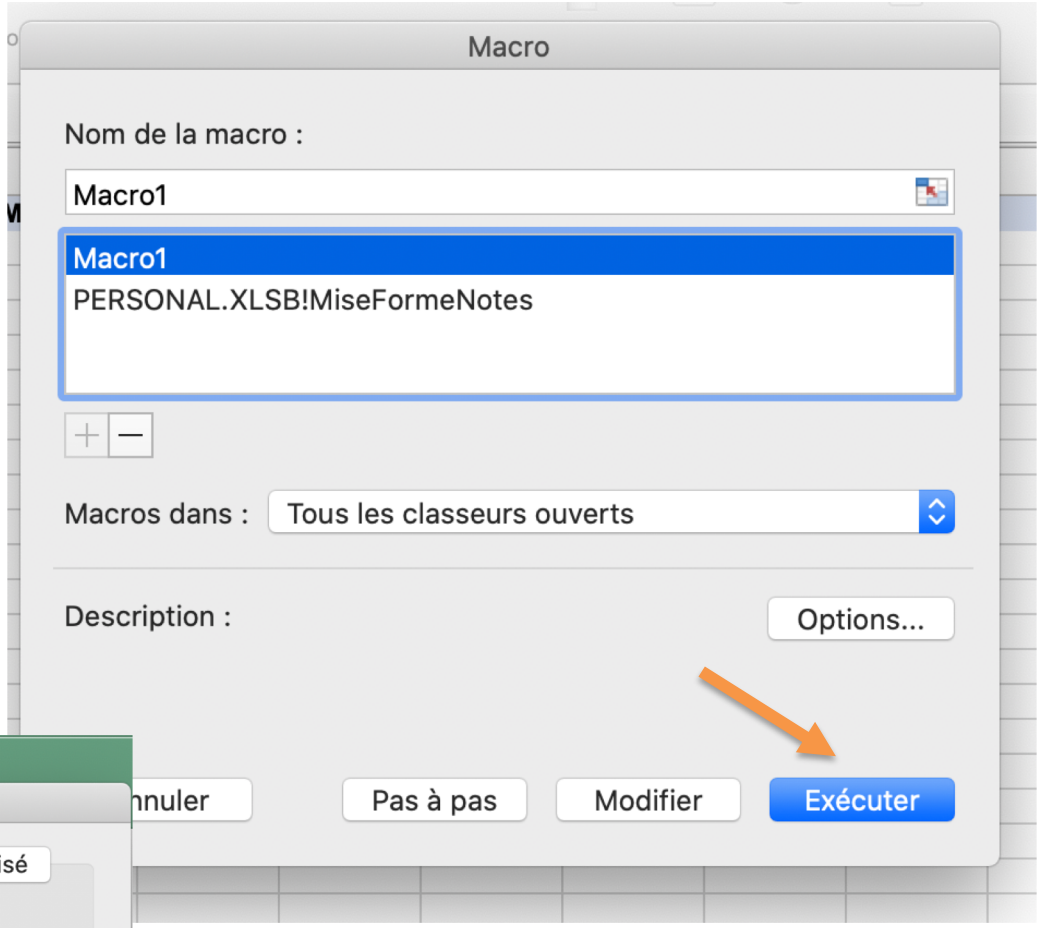
UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

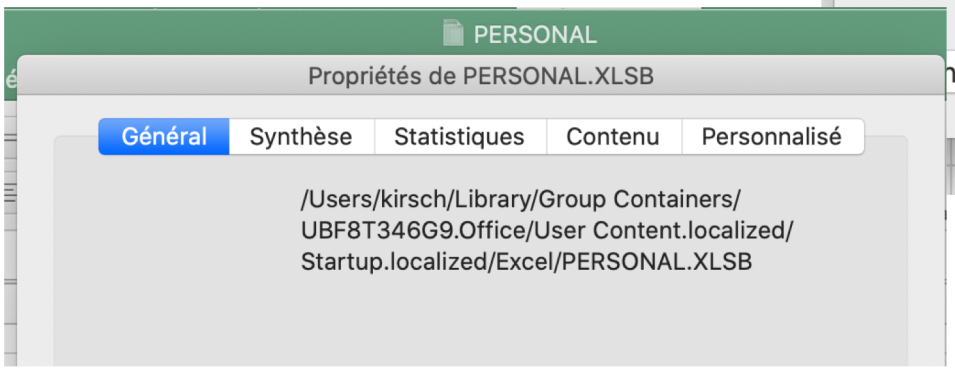
Macros

On peut exécuter les macros localisées sur les classeurs ouverts

PERSONAL.XLSB est le **classeur de macros personnelles**

Fichier caché ouvert de manière automatique lorsqu'**Excel s'ouvre**.





Sur **Windows**, **PERSONAL.XLSB** se trouve sur C:\Utilisateurs\nomUser**AppData\Local\Microsoft\Excel\XLStart**

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Macros

- **Exécution macro**
  - Souvent on doit modifier le code de la macro pour pouvoir reproduire correctement son comportement
  - **Exemple :** Macro pour classer les notes par la moyenne finales

ExempleMacro1.xlsm - Module 2 (Code)

(Général) | TrierParNote

Code généré par la macro :

```
Sub TrierParNote()
'
' TrierParNote Macro
'

'
    ActiveWorkbook.Worksheets("G1").Sort.SortFields.Clear
    ActiveWorkbook.Worksheets("G1").Sort.SortFields.Add2 Key:=Range("D2:D10"), _
        SortOn:=xlSortOnValues, Order:=xlAscending, DataOption:=xlSortNormal
    ActiveWorkbook.Worksheets("G1").Sort.SortFields.Add2 Key:=Range("B2:B10"), _
        SortOn:=xlSortOnValues, Order:=xlAscending, DataOption:=xlSortNormal
    ActiveWorkbook.Worksheets("G1").Sort.SortFields.Add2 Key:=Range("C2:C10"), _
        SortOn:=xlSortOnValues, Order:=xlAscending, DataOption:=xlSortNormal
    With ActiveWorkbook.Worksheets("G1").Sort
        .SetRange Range("A1:I10")
        .Header = xlYes
        .MatchCase = False
        .Orientation = xlTopToBottom
        .SortMethod = xlPinYin
        .Apply
    End With
End Sub
```

Le code mentionne *ActiveWorkbook.**Worksheets***("**G1**")
Autrement dit, ça ne fonctionne que dans cette feuille…

Pour que ça fonctionne ailleurs
(sur la feuille active, p.ex.),
il faut modifier le code
ActiveSheet
Rows.Count

Même chose les lignes, on va jusqu'à la ligne 10…
… .***Range***("**D2:D10**")

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Macros

lignes = **Rows.Count** → ligne max ou
lignes = **Cells**(**Rows.Count, 1**).**End**(**xlUp**).**Row** → dernière ligne utilisée

(Général) TrierParNote

```
'
  ' lignes = Cells(Rows.Count, 1).End(xlUp).Row
  lignes = Rows.Count
   ActiveWorkbook.ActiveSheet.Sort.SortFields.Clear
   ActiveWorkbook.ActiveSheet.Sort.SortFields.Add2 Key:=Range("D2:D" & lignes), _
      SortOn:=xlSortOnValues, Order:=xlAscending, DataOption:=xlSortNormal
   ActiveWorkbook.ActiveSheet.Sort.SortFields.Add2 Key:=Range("B2:B" & lignes), _
      SortOn:=xlSortOnValues, Order:=xlAscending, DataOption:=xlSortNormal
   ActiveWorkbook.ActiveSheet.Sort.SortFields.Add2 Key:=Range("C2:C" & lignes), _
      SortOn:=xlSortOnValues, Order:=xlAscending, DataOption:=xlSortNormal
   With ActiveWorkbook.ActiveSheet.Sort
      .SetRange Range("A1:I" & lignes)
      .Header = xlYes
      .MatchCase = False
      .Orientation = xlTopToBottom
      .SortMethod = xlPinYin
      .Apply
   End With
End Sub
```

Range("D2:D" **& lignes**) → concaténation

ActiveWorkbook.**ActiveSheet**.Sort … → feuille active

C'est donc important de connaître un minimum le **langage VBA**..

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Exercices

- A partir du fichier « **FichierNotes.xlsx** » :
  - Créer une macro pour mettre en forme les données des notes sur la feuille « G1 »
    - Police Arial 11pt
    - Entête en Arial 12pt, fond bleu
    - Notes en format « Nombre »
  - Analyser le code généré par la macro
  - Exécuter la macro sur la feuille « G2 »

A1 | Groupe

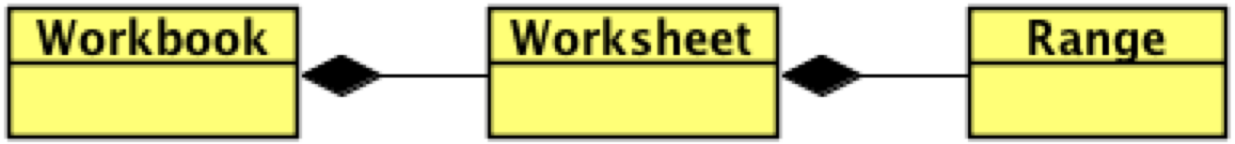
	A	B	C	D	E	F	G	H	I	J
1	Groupe	Nom	Prénom	Moy Calc	Examen	Moy CC ca	Note1	Note2	Note3	
2	G1	AAAA	aaaaa	13,08	12,00	14,17	17,75	8,00	16,75	
3	G1	BBBB	bbbb	14,29	14,00	14,58	19,00	8,00	16,75	
4	G1	CCCC	cccc	14,67	16,75	12,58	17,00	6,00	14,75	
5	G1	DDDD	ddddd	13,29	16,75	9,83	16,50	0,00	13,00	
6	G1	EEEE	eeee	11,29	14,75	7,83	10,00	0,00	13,50	
7	G1	FFFF	fffff	11,13	8,00	14,25	18,00	8,00	16,75	
8	G1	GGGG	ggggg	13,83	13,75	13,92	19,00	6,00	16,75	
9	G1	HHHHH	hhhh	8,00	6,00	10,00	12,00	8,00		
10	G1	KKKKKK	kkkkkk	10,67	8,00	13,33	18,00	8,00	14,00	
11										

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

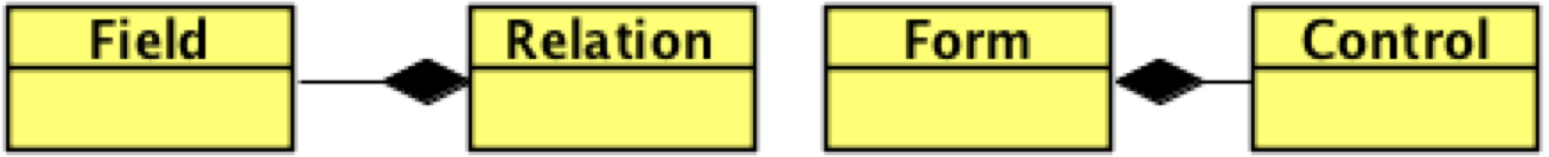
VBA : Modèle OO

- **Modèle Objets de VBA**
  - Les éléments manipulés par VBA sont des **objets**
    - **Excel : Workbook** (classeur), **Worksheet** (feuille de calcul), **Range** (cellules)…
    - **Access** : **Form** (formulaire), **Report** (état), **RecordSet** (requête)…
  - Chaque objet propose des **propriétés** et des **actions**

Modèle OO Excel



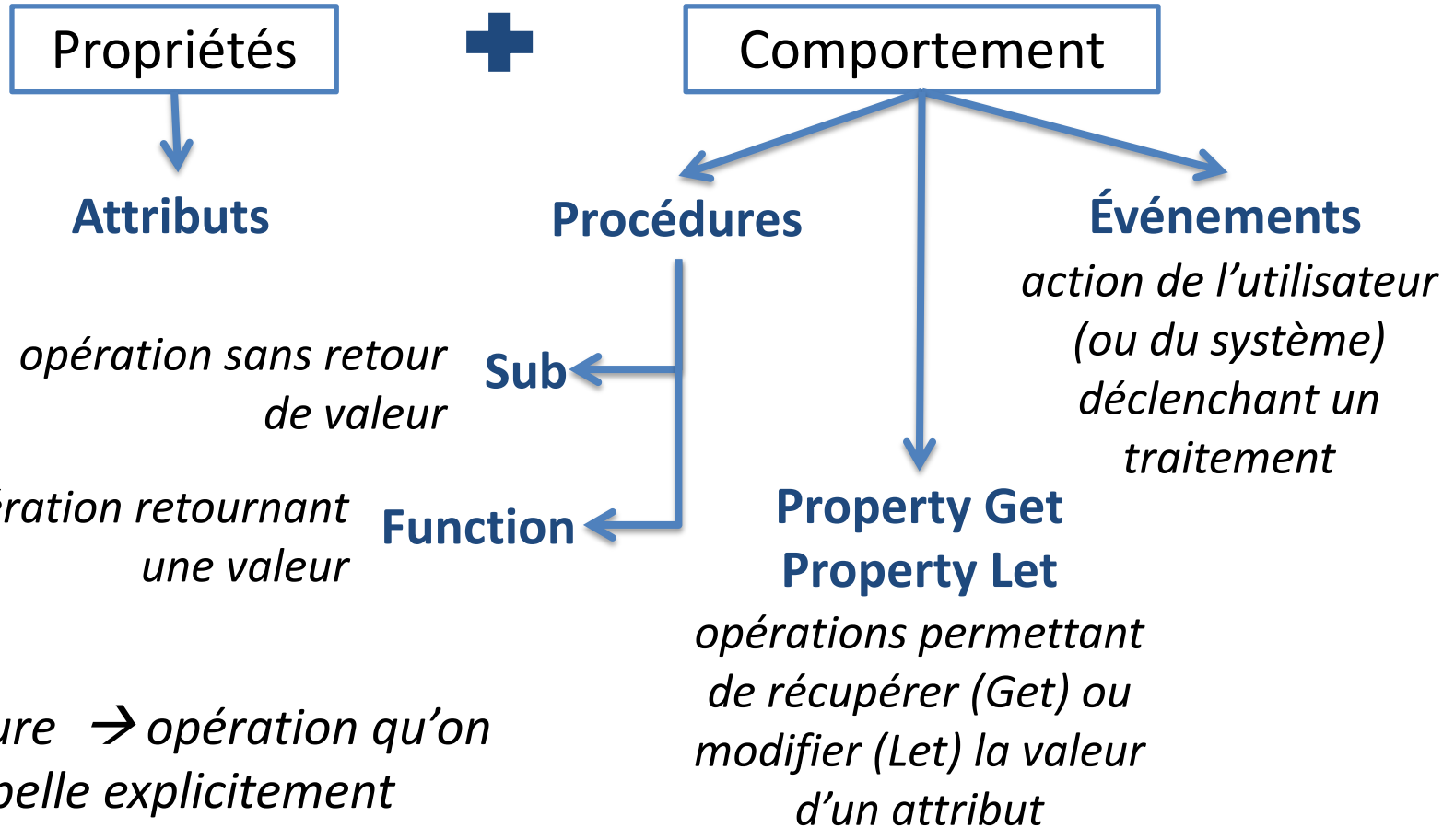
Modèle OO Access



UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : Modèle OO

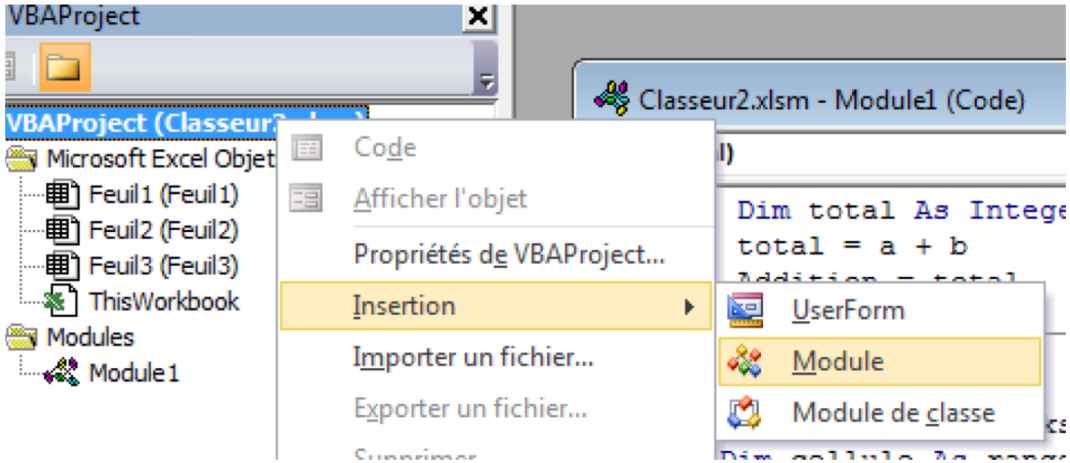
- Un objet VBA possède :



UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : Modules

- Modules contiennent le code VBA pour un document
- Plusieurs types de modules
  - Microsoft Objets
    - Evènements (*Open*, *Close*…)
  - Modules standards
    - Procédures utilitaires (crées par l'utilisateur)
  - Modules de classe
    - Classes crées par l'utilisateur

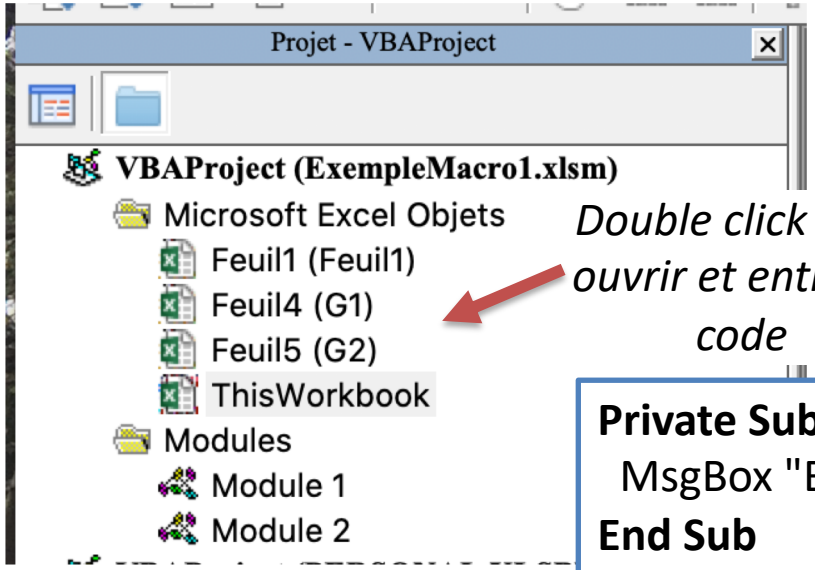


Modules
→ macros et codes de l'utilisateur
MS Excel Objects
→ codes pour le traitement des événements

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : Exemple

- **Exercice :**
  - Afficher un message de bienvenue à l'ouverture du classeur « **FichierNotes.xlsx** »
    - Ouverture d'un classeur → événement « **Open** »
    - Événements sur le classeur → module « **ThisWorkook** »
    - Nom du classeur actif → propriété « **ActiveWorkbook.Name** »



Double click pour ouvrir et entrer le code

Workbook_Open() → événement **Open** de la classe **Workbook**

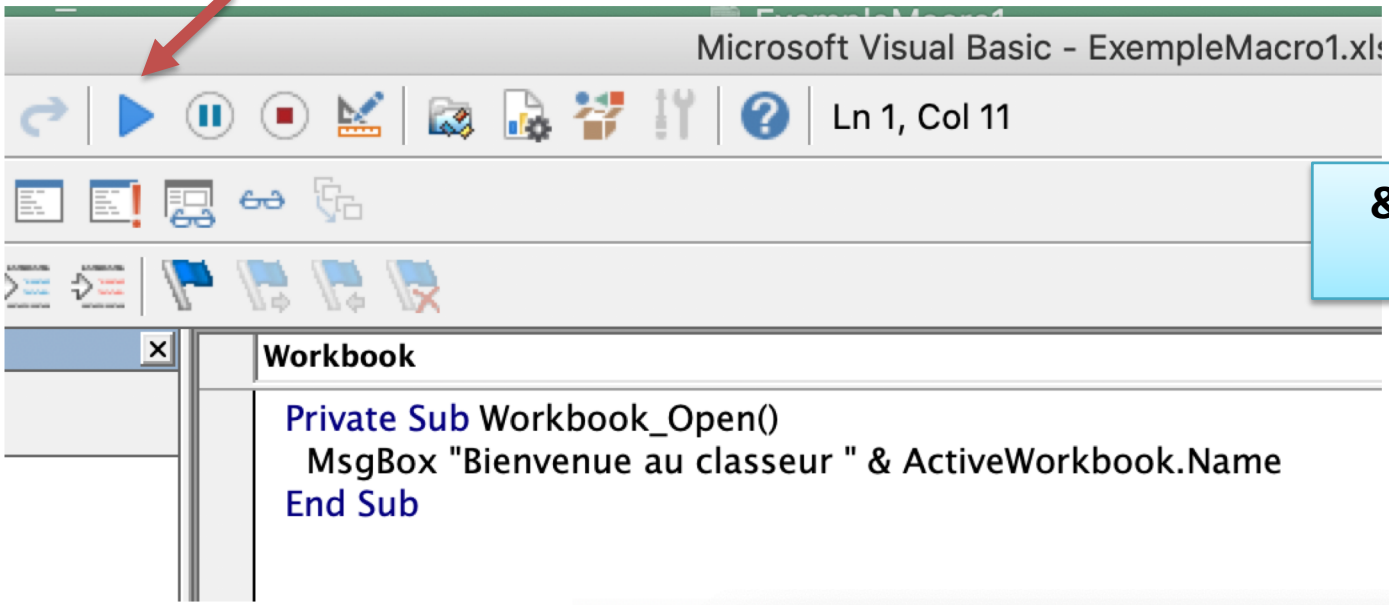
ActiveWorkbook.Name → propriété **Name** de l'objet **ActiveWorkbook**

```
Private Sub Workbook_Open()
  MsgBox "Bienvenue au classeur " & ActiveWorkbook.Name
End Sub
```

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

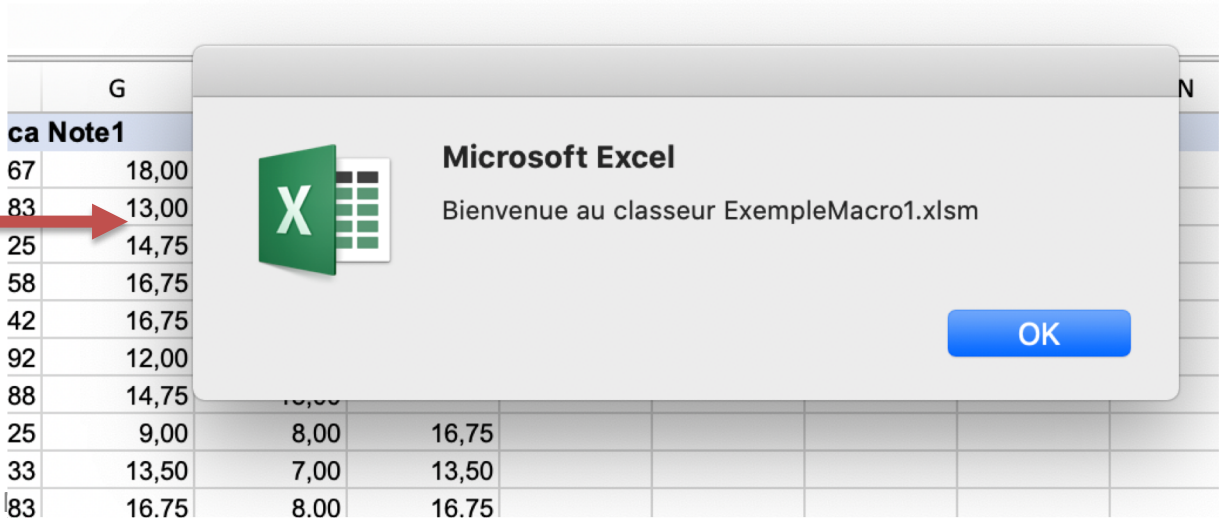
VBA : Exemple

On peut exécuter directement la Sub



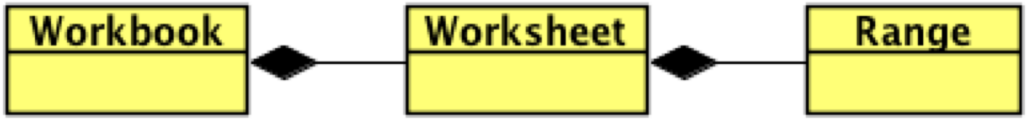
& : opérateur de concaténation

Lors de l'ouverture du fichier



UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : accès aux données EXCEL



- Les données dans un document Excel (objet **ThisWorkbook**) s'organisent en feuilles de calcul (objets **Worksheet**) et en cellules (objets **Range**)

- Quelques objets importants
  - **ThisWorkbook** : document courant
  - **ActiveWorkbook** : document en 1er plan (actif)
  - **ActiveSheet** : feuille de calcul active

- Les **collections** vont permettre l'accès aux différents objets de la classe correspondante
  - **Worksheets**("Feuil1") → accès à la feuille « Feuil1 »

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : Collections

- **Collections**
  - Ensemble d'objets d'un même type
    - **Worksheets** (toutes feuilles de calcul), **Range** (ensemble de cellules), **Forms** (tous les formulaires), **Controls** (tous champs d'un formulaire) …

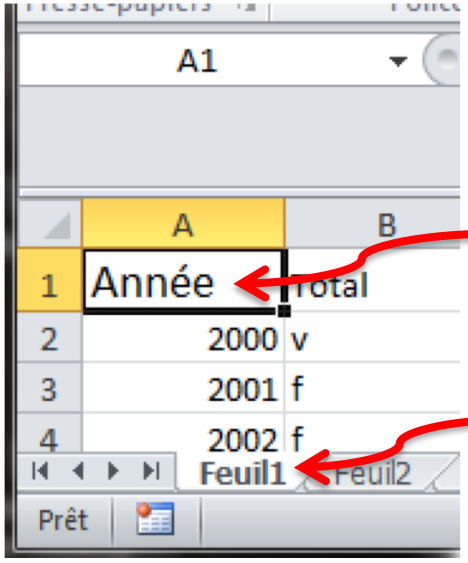
Collection(*"NomObjet"*)
Collection(*Numéro*)

Worksheets(*"Feuil1"*)
Worksheets(*1*)

Formulaire « Employe »

Worksheets(*1*).***Range***(*"A1"*)

Worksheets("Feuil1").Cells(1,1)

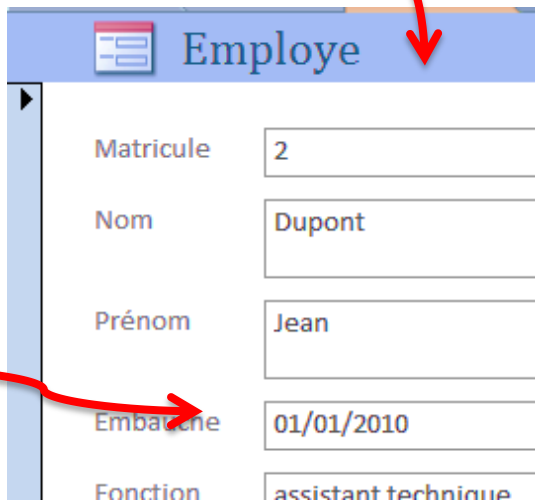


Cellule (1,1)

***Cells*(*ligne* , *colonne*)**

Feuille « Feuil1 »

Contrôle « Embauche »



UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : Exemple

- **Exercice** :
  - Améliorer le code d'ouverture du classeur « **FichierNotes.xlsx** »
    - Afficher le **nombre de feuilles** dans le classeur
    - Collection « **Worksheets** » → ensemble de feuilles
    - Propriété « **Count** » → nombre d'éléments

Worksheets.Count → nombre de feuilles dans le classeur

```
Private Sub Workbook_Open()
  MsgBox "Bienvenue au classeur " & ActiveWorkbook.Name _
     & Chr(10) _
     & "Nombre de feuilles " & Worksheets.Count
End Sub
```

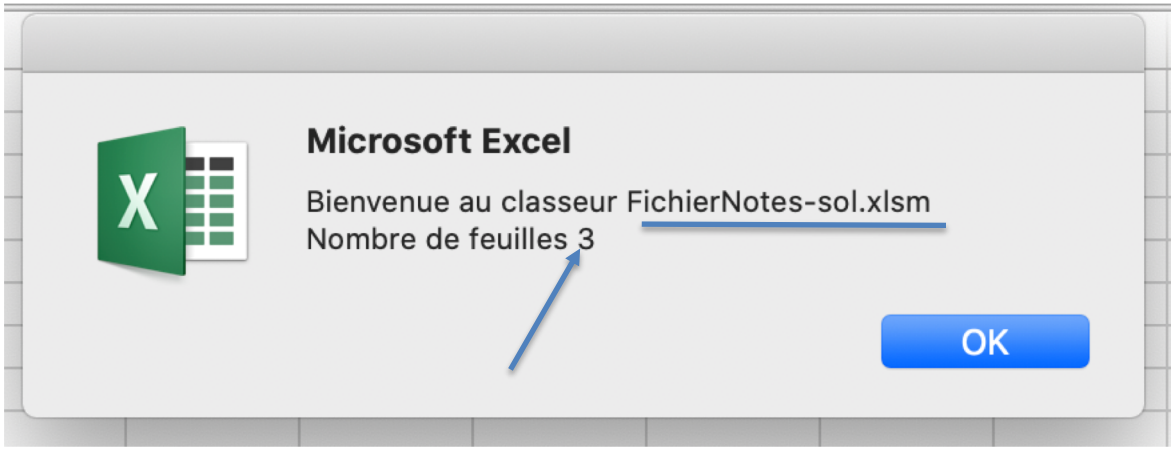
_ : saut de ligne
Chr(10) : caractère de nouvelle ligne

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : Exercice

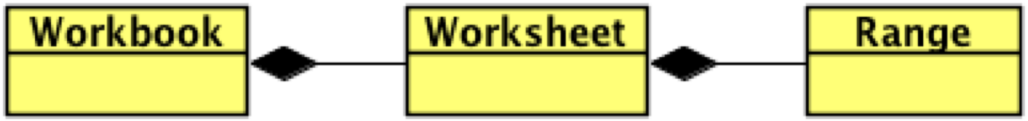
Workbook | Open

```
Private Sub Workbook_Open()
  MsgBox "Bienvenue au classeur " & ActiveWorkbook.Name _
    & Chr(10) _
    & "Nombre de feuilles " & Worksheets.Count
End Sub
```



UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : accès aux données EXCEL



- **Range**
  - Un objet Range est une région dans une feuille contenant une ou plusieurs cellules
  - Les opérations de la classe Range vont nous permettre d'interagir ou de modifier les cellules

```
ActiveSheet.Range("A1").Value
ActiveSheet.Cells(1, 2).Value
```

L'opération **Value** récupère la **valeur de la cellule**

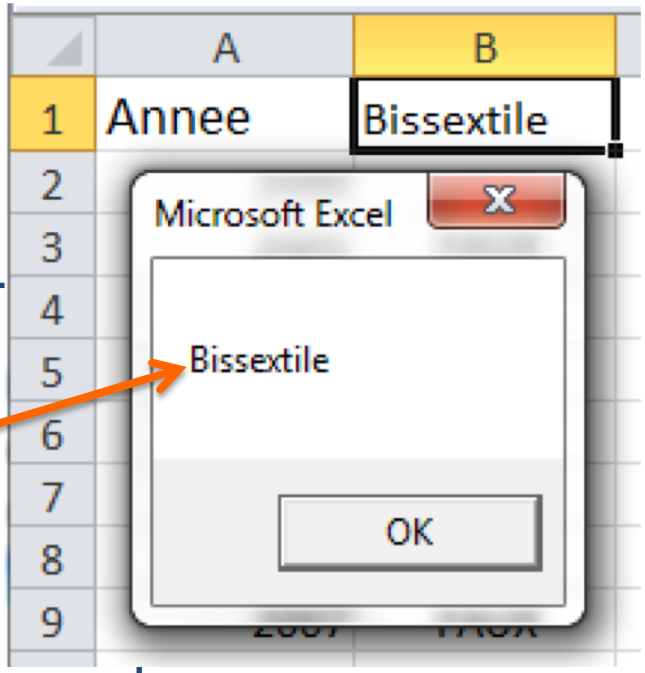
Attention : les collections démarrent en **1 (et *pas 0*)**

La collection **Range** permet d'indiquer un ensemble de cellules **Range("A1:B1")**

VBA : accès aux données EXCEL

Attention
Cells(ligne , colonne)

```
Sub manipulationRange()

    MsgBox ActiveSheet.Range("A1").Value
    MsgBox ActiveSheet.Cells(1, 2).Value

    Worksheets("Feuil1").Cells(1, 1).Font.Size = 14
    Worksheets("Feuil1").Range("A1:B1").Font.Bold = True

End Sub
```



changement de la taille et gras de la police

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Exercice

- **Exercice** :
  - Toujours dans le classeur « **FichierNotes.xlsx** »,
  - Modifier l'événement « Open » afin qu'il puisse placer la colonne « Moy Calc » en gras
    - Récupérer la feuille avec la collection **Worksheets**
    - Récupérer la colonne avec la collection **Range**
    - Modifier la police avec les propriétés **Font.Bold**

Worksheets("G1") → Récupérer la feuille nommé G1
Worksheets(2) → Récupérer la 2ème feuille du classeur

Worksheets("G1").Range("D:D") → range D:D de la feuille G1
Worksheets(2). Range("D:D") → range D:D de la 2ème feuille

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Exercice

Workbook | Open

```
Private Sub Workbook_Open()
  MsgBox "Bienvenue au classeur " & ActiveWorkbook.Name _
    & Chr(10) _
    & "Nombre de feuilles " & Worksheets.Count

  Worksheets("G1").Range("D:D").Font.Bold = True
  Worksheets(3).Range("D:D").Font.Bold = True

End Sub
```

Deux possibilités, avec
***Worksheets*("G1") et G2**
ou
***Worksheets*(2) et 3**

	A	B	C	D	E	F	G
1	Groupe	Nom	Prénom	Moy Calc	Examen	Moy CC ca	Note1
2	G1	HHHHH	hhhh	**8,00**	6,00	10,00	12,00
3	G1	KKKKKK	kkkkkk	**10,67**	8,00	13,33	18,00
4	G1	FFFF	fffff	**11,13**	8,00	14,25	18,00
5	G1	EEEE	eeee	**11,29**	14,75	7,83	10,00
6	G1	AAAA	aaaaa	**13,08**	12,00	14,17	17,75
7	G1	DDDD	ddddd	**13,29**	16,75	9,83	16,50
8	G1	GGGG	ggggg	**13,83**	13,75	13,92	19,00
9	G1	BBBB	bbbb	**14.29**	14.00	14.58	19.00

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : les bases

- Pour être plus à l'aise sur VBA, mieux vaut maitriser certains concepts
  - Variables & tableaux
  - Evénements
  - Procédures Sub et Function
  - Instructions de contrôle
    - With … End With
    - If … Then … ElseIf … Then … Else … End If
    - Do While … Loop
    - For .. To … Step … Next
    - For Each … In … Next

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : variables

- **Variables**
  - Une variable est un **conteneur**, on y garde **une valeur**
  - Déclaration est fortement recommandée, mais **pas obligatoire**
    - **Dim** *variable* **AS** *Type*

Dim** auj **As Date

auj ***= Date*** *'date actuelle*

auj

19 / 4 / 2021

Exemples types des données :

AS **Integer** → entier
AS **Single** → numérique (*float*)
As **Double** → numérique précision double
AS **Boolean** → booléen (*True* / *False*)
AS **String** → chaîne de caractères
As **Currency** → valeur monétaire
AS **Variant** →
n'importe quel type de données
AS **Object** →
objet de n'importe quelle classe

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : variables

- **Variables**
  - Pour affecter la valeur à un objet, on utilise **Set**

```
Dim a As Integer
Dim b As Integer

a = 2
b = a * a
```

1) déclaration

2) affectation

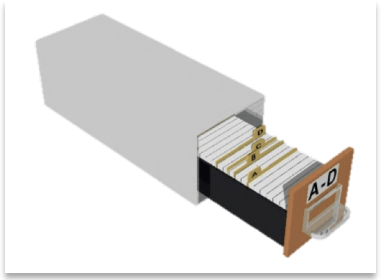
```
Dim cellule As Range

Set  cellule = Worksheets("Feuil1").Cells(1,1)

MsgBox cellule.Valeur
```

3) usage

Propriété **Valeur** de la classe **Range**

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : tableaux



- **Tableaux**
  - Variables capables de garder **plusieurs valeurs**
  - On indique la première et la dernière position e
    - Première position est ***0, sauf indication*** contraire

Dernière position → Ubound (untab)

*Dim **untab (3)** As Currency*
***untab(1)** = 1.5*
***untab(3)** = 3*

untab	0	**1.5**	0	**3**
	0	**1**	2	3

Première position → Lbound (autreTab)

*Dim **autreTab (1 To 3)** As Currency*
***autreTab(1)** = 1.5*

autreTab	**1.5**	0	0
	1	2	3

*Dim **tabBiDim(1, 2)** As Currency*
***tabBiDim(0, 2)** = 2.5*
***tabBiDim(1, 0)** = 1.5*

tabBiDim	**0**	1	**2**
0	0	0	**2.5**
1	**1.5**	0	0

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Exercice

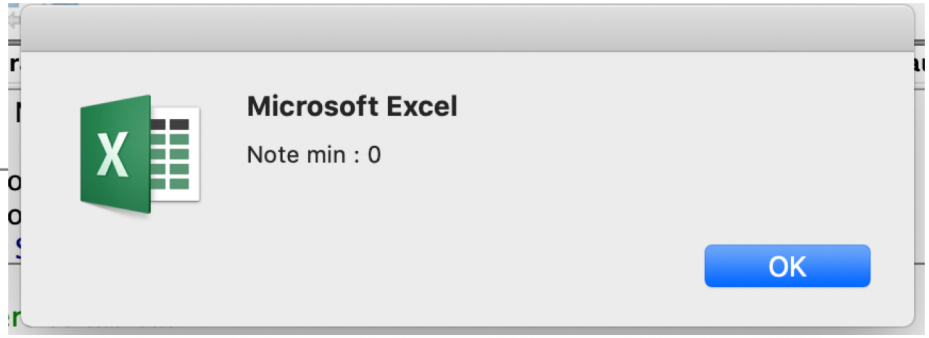
- Exercice
  - Dans le fichier « **FichierNotes.xlsx** », créer un nouveau **module**
  - Créer une Sub « **ExerciceTableau**() » qui :
    - Récupère les 3 notes sur la ligne 2 de la feuille G1 et les place dans un tableau « **notes** »
    - Utilise la fonction « **Min** » de l'objet « **WorksheetFunction** » pour trouver la note la moins élevée
    - Place cette valeur dans une variable « **min** »
    - Affiche avec un **MsgBox** le contenu de cette variable

Bien penser à la **déclaration des variables** :
Dim *notes*(1 To 3) **As Double**
Dim min **As Double**

Application de l'objet **WorksheetFunction** :
min = **Application**.**WorksheetFunction**.**min** (*notes*)

Exercice

```
Sub ExerciceTableau()

Dim notes(1 To 3) As Double
Dim min As Double

notes(1) = Worksheets("G1").Range("G2").Value
notes(2) = Worksheets("G1").Range("H2").Value   'on peut aussi faire avec cells
notes(3) = Worksheets("G1").Cells(2, 9).Value

min = Application.WorksheetFunction.min(notes)

MsgBox "Note min : " & min

End Sub
```



	A	B	C	D	E	F	G	H	I
1	Groupe	Nom	Prénom	Moy Calc	Examen	Moy CC ca	Note1	Note2	Note3
2	G1	HHHHH	hhhh	**8,00**	6,00	10,00	12,00	8,00	
3	G1	KKKKKK	kkkkkk	**10,67**	8,00	13,33	18,00	8,00	14,00

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : événements

- **Événements**
  - **Actions** réalisées par l'utilisateur (ou le système) sur un **élément** (classeur, feuille…)
    - Exemples : ouverture d'un document, clique sur un bouton, fermeture de l'application, mise à jour des données…
  - Le type d'événement varie en fonction de l'élément qui reçoit l'action

Contrôle
- **Click** : lorsqu'on lui clique dessus

Sur un classeur (Workbook_) :
- **Open** : à l'ouverture
- **Activate** : à l'activation
- **BeforeClose**(Cancel As Boolean) : avant la fermeture
- **BeforePrint**(Cancel As Boolean) : avant l'impression

Sur une feuille (Worksheet_) :
- **Activate**: à l'activation
- **Change**(ByVal Target As Range): lors d'une modification
- **PivotTableUpdate**(ByVal Target As PivotTable) : lors de la mise à jour d'un TCD

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : événements

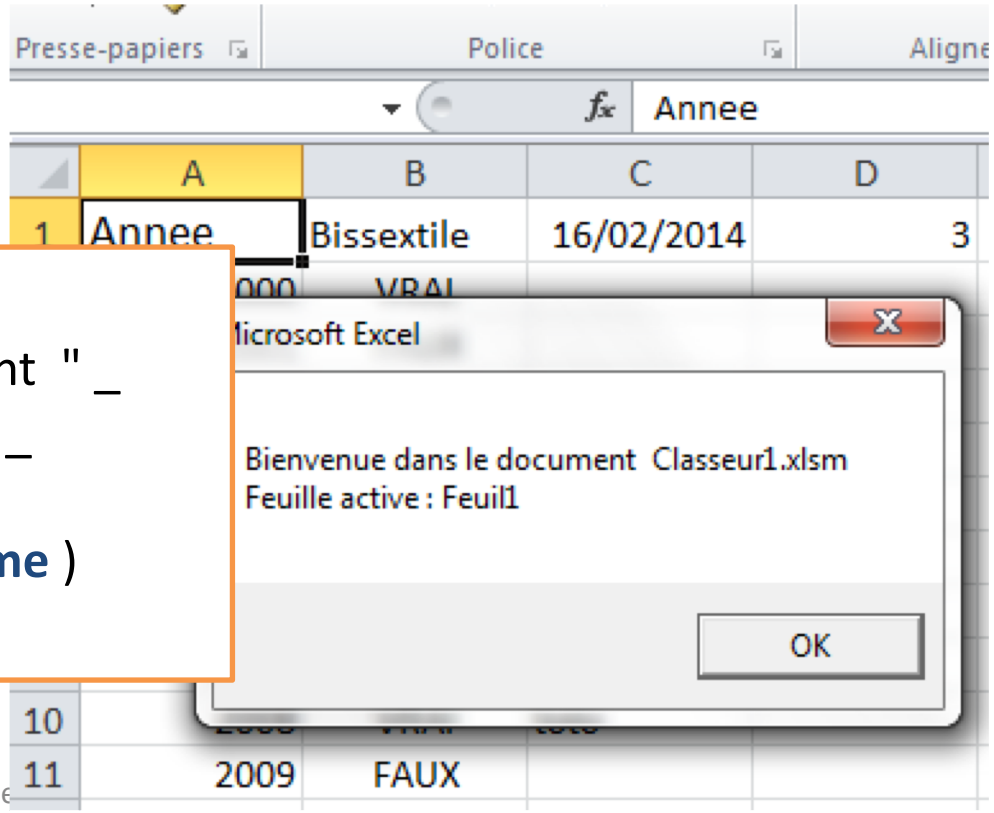
- Evénements
  - Exemple : affichage d'un message à l'ouverture d'un document Excel

& : opérateur de concaténation
_ : saut de ligne
Chr(10) : caractère de nouvelle ligne

élément qui reçoit l'action (« **Workbook** »)

quelle type d'action (« **Open** »)

```
Private Sub Workbook_Open()
  MsgBox ("Bienvenue dans le document  " _
      & ThisWorkbook.Name & Chr(10) _
      & "Feuille active : " _
      & ThisWorkbook.ActiveSheet.Name )
End Sub
```



VBA : fonctions & sub

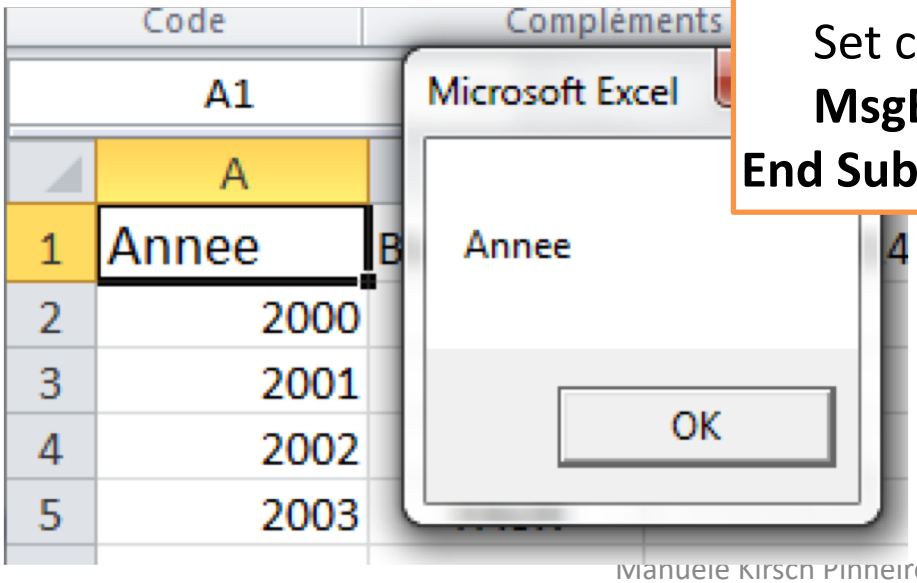
- **Procédures de type Sub**
  - Exécution d'une opération qui ne retourne aucune valeur

Private | Public Sub *NomProcédure* **(** *paramètre AS type* … **)**

. . .

End Sub

```
Sub ValeurA1()
    Dim cellule As Range
    Set cellule = Worksheets("Feuil1").Cells(1, 1)
    MsgBox (cellule.Value)
End Sub
```



UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

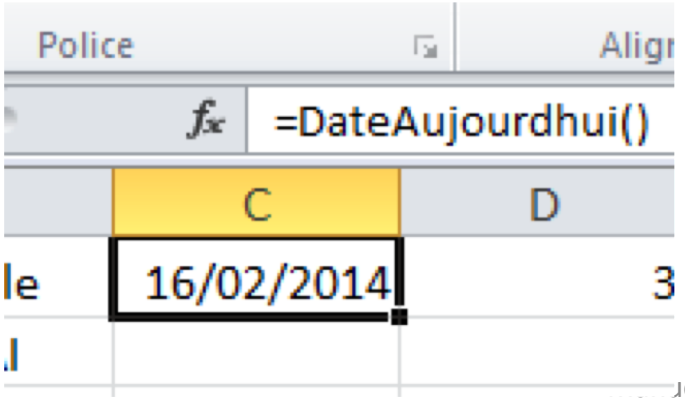
VBA : fonctions & sub

- **Fonctions (Function)**
  - Opération qui retourne une valeur
  - Variable correspondant au **nom de la fonction**
  - On peut les utiliser dans les **feuilles de calcul**

Private | Public Function *MaFonction* **(** *paramètres…* **) AS Type**

MaFonction = valeur_à_retourner

End Sub




```
Public Function DateAujourdhui() As Date
    Dim auj As Date
    auj = Date
    DateAujourdhui = auj
End Function
```

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Exercice

- **Exercice**
  - Toujours dans le nouveau module créé sur le fichier « **FichierNotes.xlsx** »
    - Créer une Fonction « **JourApres** » qui récupère une date en paramètre et l'avance d'un jour
  - Utilisez cette fonction avec plusieurs valeurs qui vous ajouteriez dans la feuille « **Feuil1** »

Nom fonction *Paramètre* *Type retour*

```
Function JourApres(unJour As Date) As Date
    JourApres = unJour + 1
End Function
```

*Valeur de retour variable **JourApres***

Application de la fonction :
=**JourApres**(D1)

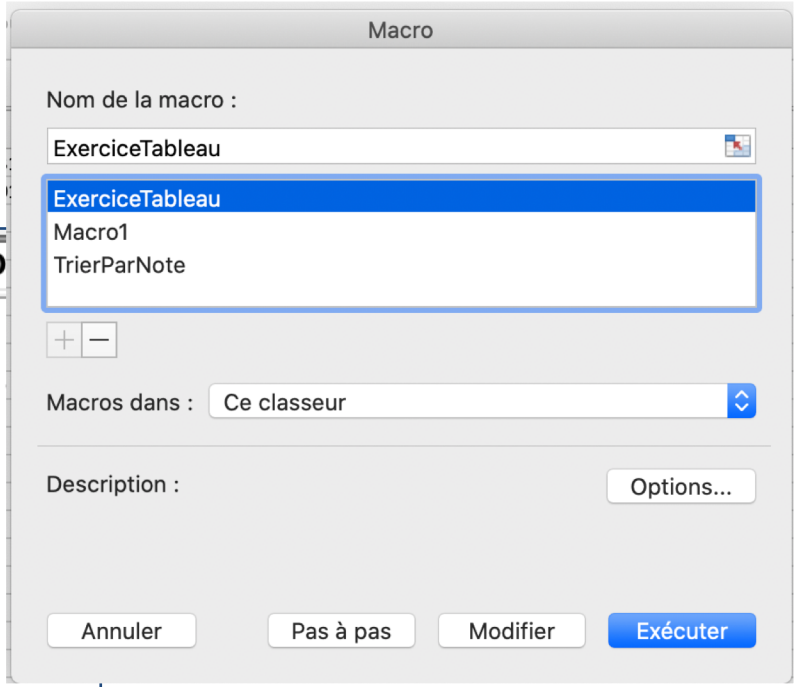
fx =JourApres(D1)

	C	D	E	F
	Date :	*18/04/2021*	03/02/2020	31/05/2021
	Jour après :	19/04/2021	04/02/2020	01/06/2021

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : évènements & procédures

- **Evènements & Procédures**
  - Les **Function** et **Sub** peuvent être invoquées dans un événement ou dans d'autres procédures **Sub** / **Function**
  - Les **Sub** sont perçues comme des **macro**




```
Workbook
Private Sub Workbook_Open()
  MsgBox "Bienvenue au classeur " & ActiveWorkbook.Name
   & Chr(10) _
   & "Nombre de feuilles " & Worksheets.Count

  Worksheets("G1").Range("D:D").Font.Bold = True
  Worksheets(3).Range("D:D").Font.Bold = True

  Worksheets(1).Range("D3").Value = JourApres(Date)

End Sub
```

```
'exercice function
 Function JourApres(unJour As Date) As Date
    JourApres = unJour + 1
End Function
```

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : Instructions de contrôle

- Instructions de contrôle : **With … End With**
  - Permet de réaliser plusieurs opérations avec un même objet

Range("D1").Formula = "=TODAY()"
*On attribut une **formule** à une cellule.*
*Attention nom de la formule en **anglais.***

```
With Worksheets("Feuil1")
    .Range("D1").Formula = "=TODAY()"
    .Range("D1").Font.Italic = True
End With
```

Équivaut à dire

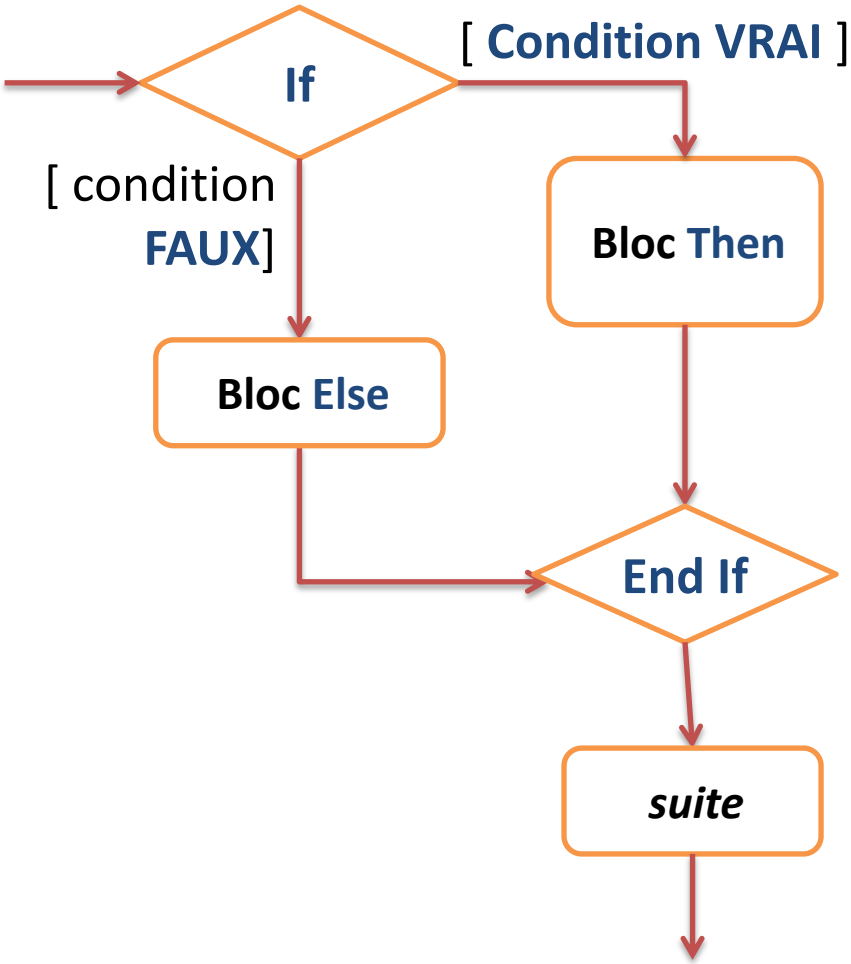
```
Worksheets("Feuil1").Range("D1").Formula = "=TODAY()"
Worksheets("Feuil1").Range("D1").Font.Italic = True
```

VBA : Instructions de contrôle

- Instructions conditionnelles : **if … else … end if**

```
If condition Then
    'si condition vraie …
Else
    'si condition faux
End If
```

optionnel




```
If mont >= 10 Then
    MsgBox "Validé" , vbOKOnly
Else
    MsgBox "Ajourné", vbCritical
End If
```

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : Instructions de contrôle

- Instructions conditionnelles : **if…elseIf…else… end if**

```
If condition Then
    'si condition vraie …
ElseIf condition2 Then
    'si condition2 vrai
Else
    'si aucune des
          ' précédentes
End If
```

```
If mont > 6 And mont < 10 Then
      MsgBox "Ajourné", vbExclamation
ElseIf mont <= 6 Then
      MsgBox "Rattrapage",   vbCritical
Else
      MsgBox "Validé", vbOKOnly
End If
```

Exercice

- **Exercice**
  - Toujours sur le fichier « **FichierNotes.xlsx** » :
  - Créer une fonction « Situation » qui retourne la situation (« Ajourné », « Validé », « Rattrapage ») d'un étudiant en fonction de la moyenne calculé
    - **Function Situation (note as Range) as String**
  - Appliquer cette fonction sur la feuille G1 en utilisant la moyenne calculée
    - =**Situation**(D2)

Exercice

```
Function Situation(note As Range) As String
    Dim mont As Single

    mont = note.Value

    If mont > 6 And mont < 10 Then
      Situation = "Ajourné"
   ElseIf mont <= 6 Then
      Situation = "Rattrapage"
   Else
      Situation = "Validé"
   End If

End Function
```

Ne pas oublier d'affecter une **valeur** à la variable **Situation** (**variable de sortie**)

J2 | =Situation(D2)

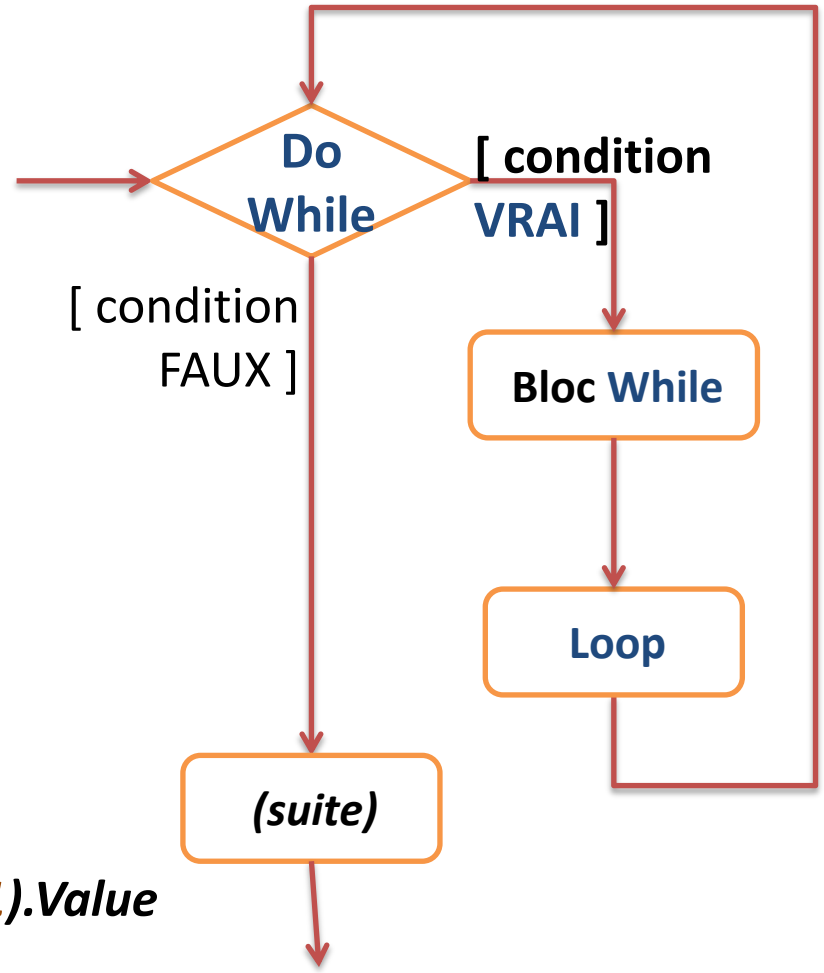
	D	E	F	G	H	I	J
1	**Moy Calc**	**Examen**	**Moy CC ca**	**Note1**	**Note2**	**Note3**	
2	**8,00**	6,00	10,00	12,00	8,00		Ajourné
3	**10,67**	8,00	13,33	18,00	8,00	14,00	Validé
4	**5,17**	6,00	4,33	5,00	4,00	4,00	Rattrapage
5	**11,29**	14,75	7,83	10,00	0,00	13,50	Validé
6	**13,08**	12,00	14,17	17,75	8,00	16,75	Validé

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : Instructions de contrôle

- Boucles : **Do While… Loop**

```
Do While condition
    ' exécute tant que la
    ' condition est vraie
Loop
```




```
Set cel = ActiveSheet.Range("D2:D10")
nbLig = ActiveSheet.Range("D2:D10").Count
compteur = 1

Do While compteur <= nbLig
    somme = somme +  cel.Cells(compteur, 1).Value
    compteur = compteur + 1
Loop
```

Première ligne / colonne est toujours 1

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Exercice

- **Exercice**
  - Toujours sur le fichier « **FichierNotes.xlsx** » :
  - Créer une fonction « **MoyenneGroupe** » qui calcule la moyenne d'un Range
    - **Function MoyenneGroupe (note as Range) as Single**
  - Ajouter sur la feuille « feuil1 » le calcule de la moyenne finale des groupes G1 et G2
    - =**MoyenneGroupe**('G1'!D2:D10)

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Exercice

```
Function MoyenneGroupe(groupe As Range) As Single

    Dim compteur As Integer
    Dim somme As Single

    somme = 0
    compteur = 1

    Do While compteur <= groupe.Count
        somme = somme + groupe.Cells(compteur, 1).Value
        compteur = compteur + 1
    Loop

    If groupe.Count > 0 Then
        MoyenneGroupe = somme / groupe.Count
    Else
        MoyenneGroupe = 0
    End If

End Function
```

C	D	E	F
Prénom	**Moy Calc**	**Examen**	**Moy CC ca N**
hhhh	**8,00**	6,00	10,00
kkkkkk	**10,67**	8,00	13,33
fffff	**5,17**	6,00	4,33
eeee	**11,29**	14,75	7,83
aaaaa	**13,08**	12,00	14,17

fx =MoyenneGroupe('G1'!D2:D10)

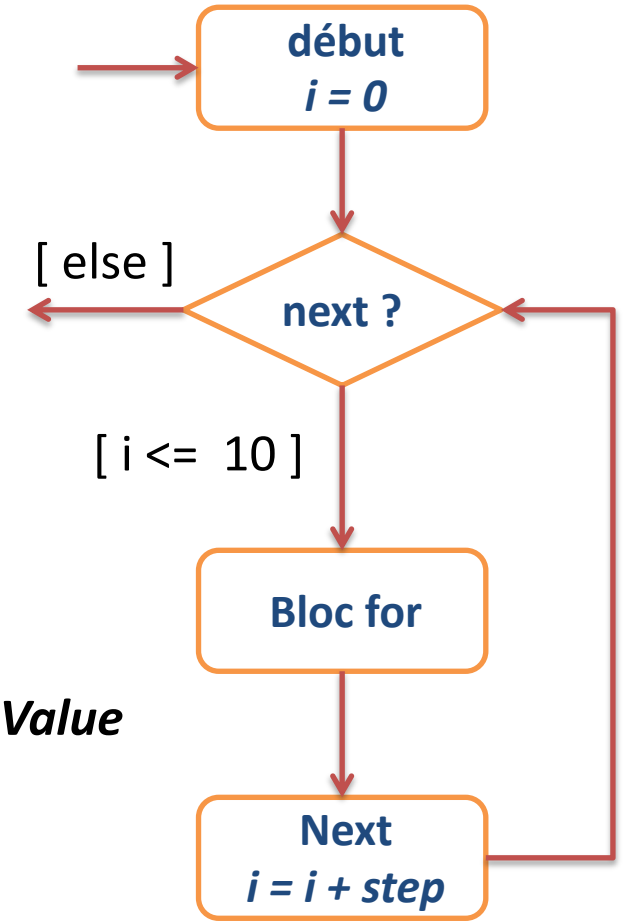
C	D	E	F
Date :	*19/04/2021*	03/02/2020	31/05/2021
Jour après :	20/04/2021	04/02/2020	01/06/2021
Demain :	20/04/2021		
Moyenne G1	11,5879631		
Moyenne G2	14,0746536		

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : Instructions de contrôle

- Boucles : **For…Next**

```
For début To fin Step n
    ' exécute de début
    ' jusqu'à fin
    ' de n en n (step)
Next
```




```
For lin = 1 To 10 Step 1
    somme = somme + Worksheets("G1").Cells( lin, 4 ).Value
Next
```

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Exercice

- **Exercice**
  - Toujours sur le fichier « **FichierNotes.xlsx** » :
  - Créer une nouvelle fonction « **MoyenneGroupe2** » qui calcule la moyenne d'un Range en utilisant une boucle **For**
    - **Function MoyenneGroupe2 (note as Range) as Single**
  - Ajouter sur la feuille « feuil1 » le calcule de la moyenne d'examen des groupes G1 et G2
    - =**MoyenneGroupe2**('G1'!E2:E10)

Exercice

```
Function MoyenneGroupe2(groupe As Range) As Single
    Dim somme As Single
    somme = 0

    For lin = 1 To groupe.Count Step 1
        somme = somme + groupe.Cells(lin, 1).Value
    Next

    If groupe.Count > 0 Then
        MoyenneGroupe2 = somme / groupe.Count
    Else
        MoyenneGroupe2 = 0
    End If
End Function
```

C	D	E	F
Prénom	**Moy Calc**	**Examen**	**Moy CC ca**
hhhh	**8,00**	6,00	10,00
kkkkkk	**10,67**	8,00	13,33
fffff	**5,17**	6,00	4,33
eeee	**11,29**	14,75	7,83
aaaaa	**13,08**	12,00	14,17

fx =MoyenneGroupe2('G1'!E2:E10)

C	D	E	F
Date :	*19/04/2021*	03/02/2020	31/05/2021
Jour après :	20/04/2021	04/02/2020	01/06/2021
Demain :	20/04/2021		
Moyenne G1	11,5879631	12	
Moyenne G2	14,0746536	15,229167	

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : Instructions de contrôle

- Boucles : **For each … Next**
  - On peut aussi parcourir toute une collection

```
For Each var IN collection
    ' pour chaque élément var
    ' dans la collection (ou tableau)
Next
```

```
'on affiche la valeur de chaque cellule du range
For Each cellule In Worksheets("G1").Range("D2:D10")
    MsgBox cellule.Value
Next
```

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Exercice

- **Exercice**
  - Toujours sur le fichier « **FichierNotes.xlsx** » :
  - Créer une nouvelle fonction « **MoyenneGroupe3** » qui calcule la moyenne d'un Range en utilisant une boucle **For Each**
    - **Function MoyenneGroupe3 (note as Range) as Single**
  - Ajouter sur la feuille « feuil1 » le calcule de la moyenne de CC des groupes G1 et G2
    - =**MoyenneGroupe3**('G1'!F2:F10)

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Exercice

```
Function MoyenneGroupe3(groupe As Range) As Single
    Dim cellule As Range

    Dim somme As Single
    somme = 0

    For Each cellule In groupe
        somme = somme + cellule.Value
    Next

    If groupe.Count > 0 Then
        MoyenneGroupe3 = somme / groupe.Count
    Else
        MoyenneGroupe3 = 0
    End If

End Function
```

C	D	E	F
Prénom	**Moy Calc**	**Examen**	**Moy CC ca**
hhhh	**8,00**	6,00	10,00
kkkkkk	**10,67**	8,00	13,33
fffff	**5,17**	6,00	4,33
eeee	**11,29**	14,75	7,83
aaaaa	**13,08**	12,00	14,17

fx =MoyenneGroupe3('G1'!G2:G10)

C	D	E	F
Date :	*19/04/2021*	03/02/2020	31/05/2021
Jour après :	20/04/2021	04/02/2020	01/06/2021
Demain :	20/04/2021		
Moyenne G1	11,5879631	12	14,916667
Moyenne G2	14,0746536	15,229167	15

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : Instructions de contrôle

- Boucles :
  - On peut imbriquer plusieurs boucles…

```
For i = Lbound(tabBiDim,1) To Ubound(tabBiDim,1)
    For j = LBound(tabBiDim, 2) To UBound(tabBiDim, 2)
        somme = somme + tabBiDim ( i , j )
    Next
Next
```

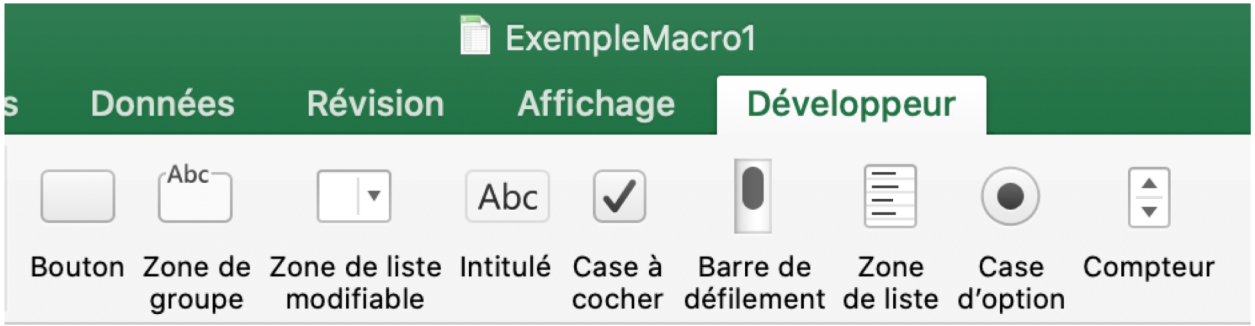
```
For début To fin Step n
    ' exécute de début
    ' jusqu'à fin
    ' de n en n (step)
Next
```

Pour chaque valeur de ligne ***i***,
on fait ***j*** parcourir toutes les colonnes,
de la ***première position*** (**Lbound**) à la
dernière position (**Ubound**)

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

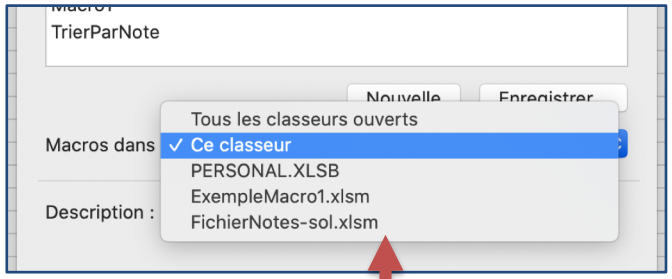
VBA : boutons et autres contrôles

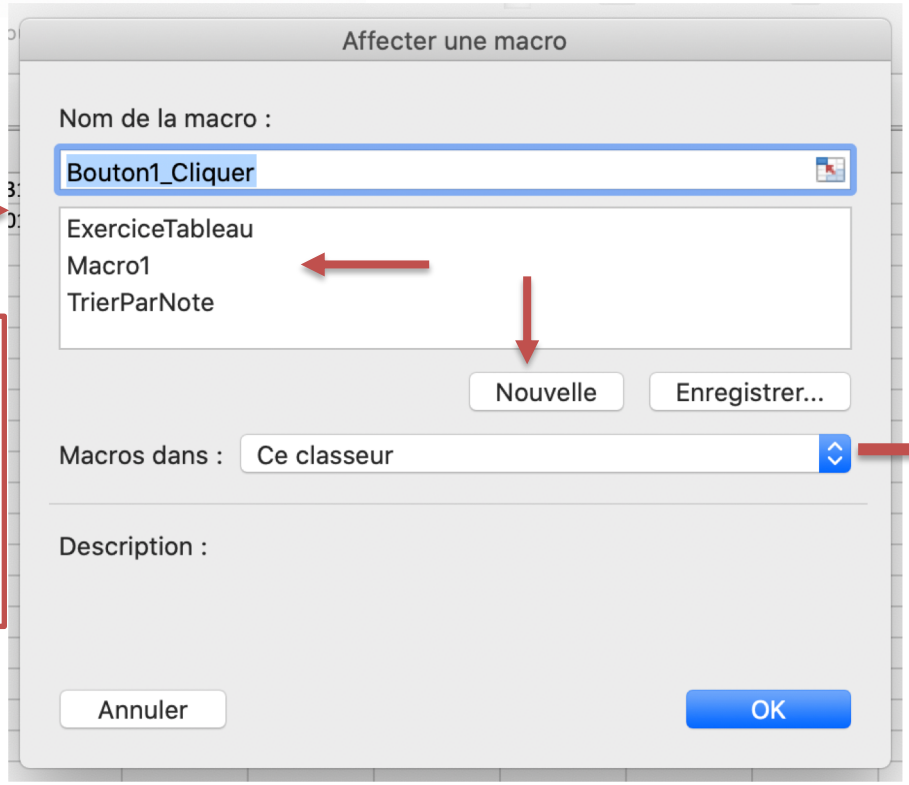
- Des contrôles (boutons, cases à cocher, listes…) sont disponibles sur Excel
- On associe un comportement (code VBA) à ces contrôles



UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : boutons et autres contrôles





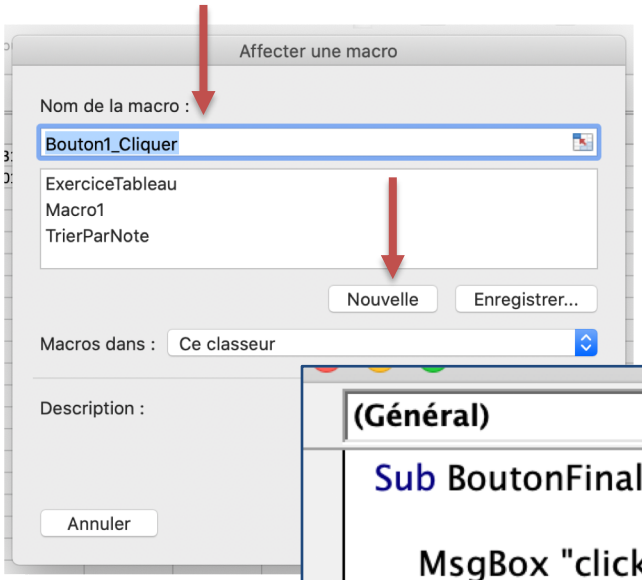
On associe un **comportement** au bouton, en choisissant une **macro existante** ou en créant une **nouvelle**.

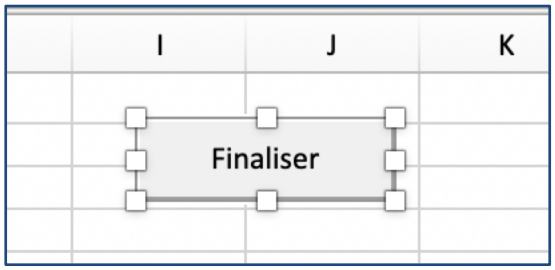
On peut utiliser une **macro** issue d'un **autre fichier** ouvert (ou du **PERSONAL.XLSB**).

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

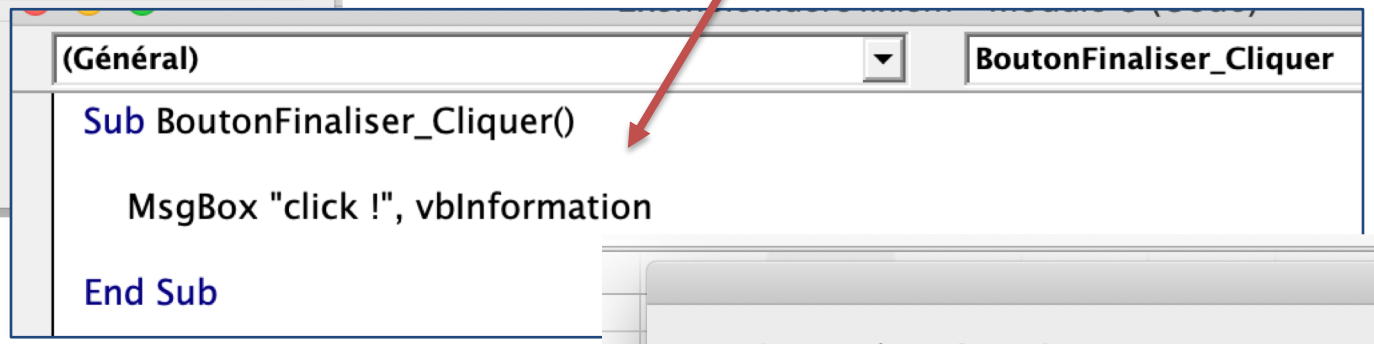
VBA : boutons et autres contrôles

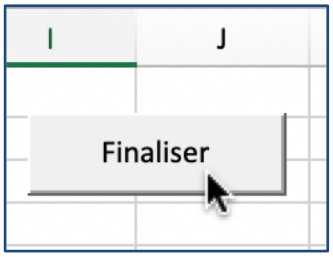
1) On insère le bouton
(penser à lui donner un nom logique)



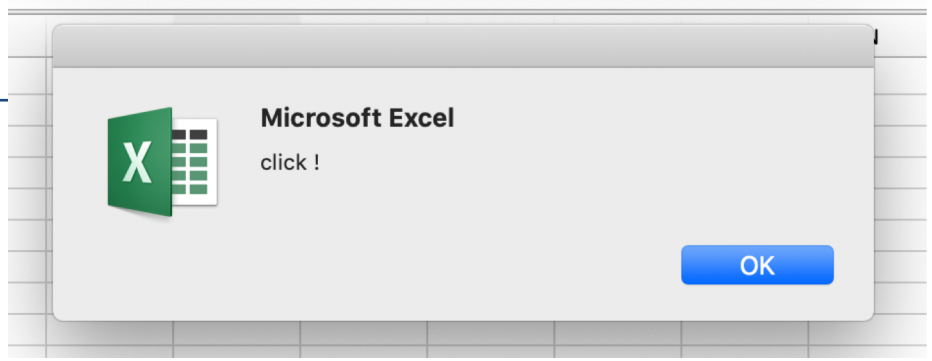


2) On programme son comportement
*(**événement** clique)*





*3) Lorsqu'on clique sur le bouton, le code associée est **exécuté***



UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Exercices

- **Exercice** (fichier « **FichierNotes.xlsx** ») :
  - Ajouter une nouvelle feuille « **NotesFinales** »
  - Ajouter à cette feuille trois colonnes « **Nom** », « **Prénom** » et « **Note** »
  - Ajouter à la feuille 1 un bouton « **Finaliser** »
  - Programme ce bouton pour qu'il puisse copier les informations des colonnes B, C et D des feuilles « G1 » et « G2 » à la nouvelle feuille

- **Attention aux points suivants :**
  - On doit trouver combien étudiants on a dans chaque feuille (NBVAL) :

```
lignesG1  = Application.WorksheetFunction.CountA(Worksheets("G1").Range("A:A"))
```

  - Pour faire de Copy-Paste on peut utiliser

```
Worksheets("G1").Range("B2:D" & lignesG1).Copy
Worksheets("NotesFinales"). Range("A" & ligneDst).PasteSpecial xlPasteValues
```

  - Penser à « décaler » le début du paste pour les informations du groupe « G2 » n'écrasent pas celles du groupe « G1 »

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

On fait ensemble…

1ère étape : on ajoute le bouton

On ajoute le bouton et la macro (vide pour l'instant)

FichierNotes-sol.xlsm - Module 4 (Code)

(Général) | BoutonFinaliser_Cliquer

```
Sub BoutonFinaliser_Cliquer()
    Dim lignesG1, lignesG2, ligneDst As Integer


End Sub
```

2ème étape : Déclaration des variables

On aura besoin de connaitre :

- La dernière ligne feuille G1
- La dernière ligne feuille G2
- La ligne où on va recopier les données

3ème étape : On trouve les valeurs à copier

Pour connaitre la dernière ligne d'une feuille, on utilise **NBVAL**, qui corresponde à la fonction « **COUNTA** » de l'objet « **WorksheetFunction** »

FichierNotes-sol.xlsm - Module 4 (Code)

(Général) | BoutonFinaliser_Cliquer

```
Sub BoutonFinaliser_Cliquer()
    Dim lignesG1, lignesG2, ligneDst As Integer

    lignesG1 = Application.WorksheetFunction.CountA(Worksheets("G1").Range("A:A"))
    lignesG2 = Application.WorksheetFunction.CountA(Worksheets("G2").Range("A:A"))

    MsgBox lignesG1 & " " & lignesG2
```

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

On fait ensemble…

4ème étape : On copie les données de G1
Pour faire une copie, il faut indiquer le **Range** des données **à copier.** L'opération « **Copy** » de la collection « **Range** » nous permet de le faire.

Worksheets("G1").**Range**("B2:D" **& lignesG1**).**Copy**

*La concaténation « **& lignesG1** » permet d'ajouter l'indication de la dernière ligne qu'on a calculé avant.*

Worksheets("NotesFinales**").Range("**A2**").PasteSpecial** ***xlPasteValuesAndNumberFormats***

5ème étape : On colle les données sur la feuille
Pour coller les données qu'on vient de copier, on utiliser l'opération « **PasteSpecial** » de la collection « **Range** » nous permet de coller les valeurs, formats, formules (***xlPasteAll***, ***xlPasteValues***, ***xlPasteValues***…)

```
Sub bouton_finaliser_Cliquer()
    Dim lignesG1, lignesG2, ligneDst As Integer

    lignesG1 = Application.WorksheetFunction.CountA(Worksheets("G1").Range("A:A"))
    lignesG2 = Application.WorksheetFunction.CountA(Worksheets("G2").Range("A:A"))

    'MsgBox lignesG1 & " " & lignesG2

    Worksheets("G1").Range("B2:D" & lignesG1).Copy
    Worksheets("NotesFinales").Range("A2").PasteSpecial xlPasteValuesAndNumberFormats
```

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

On fait ensemble…

*Si on exécute notre code, à ce moment, on a ça dans la feuille « **NotesFinales** »*

	A	B	C
1	**Nom**	**Prénom**	**Notes**
2	HHHHH	hhhh	8,00
3	KKKKKK	kkkkkk	10,67
4	FFFF	fffff	5,17
5	EEEE	eeee	11,29
6	AAAA	aaaaa	13,08
7	DDDD	ddddd	13,29
8	GGGG	ggggg	13,83
9	BBBB	bbbb	14,29
10	CCCC	cccc	14,67
11			

6ème étape : On copie les données de G2

On va refaire la même opération « **Copy** » de la collection « **Range** », maintenant pour « G2 ».

```
Worksheets("G2").Range("B2:D" & lignesG2).Copy
```

7ème étape : On colle les données sur la feuille

On va utiliser à nouveau l'opération « **PasteSpecial** » de « **Range** ».
Le problème ici est qu'il faut **coller les données après** celles qu'on a déjà.
On va donc se servir de la variable « **lignesG1** », puisqu'elle indique la **dernière ligne** qu'on vient de copier. On démarre le coller à **ligne d'après**.

```
ligneDst = lignesG1 + 1
Worksheets("NotesFinales").Range("A" & ligneDst).PasteSpecial _
                                        xlPasteValuesAndNumberFormats
```

FichierNotes-sol.xlsm - Module 4 (Code)

(Général) | BoutonFinaliser_Cliquer

```
Sub BoutonFinaliser_Cliquer()
    Dim lignesG1, lignesG2, ligneDst As Integer

    lignesG1 = Application.WorksheetFunction.CountA(Worksheets("G1").Range("A:A"))
    lignesG2 = Application.WorksheetFunction.CountA(Worksheets("G2").Range("A:A"))

    'MsgBox lignesG1 & " " & lignesG2

    Worksheets("G1").Range("B2:D" & lignesG1).Copy
    Worksheets("NotesFinales").Range("A2").PasteSpecial xlPasteValuesAndNumberFormats

    ligneDst = lignesG1 + 1
    Worksheets("G2").Range("B2:D" & lignesG2).Copy
    Worksheets("NotesFinales").Range("A" & ligneDst).PasteSpecial xlPasteValuesAndNumberFormats

End Sub
```

	A	B	C
1	**Nom**	**Prénom**	**Notes**
2	HHHHH	hhhh	8,00
3	KKKKKK	kkkkkk	10,67
4	FFFF	fffff	5,17
5	EEEE	eeee	11,29
6	AAAA	aaaaa	13,08
7	DDDD	ddddd	13,29
8	GGGG	ggggg	13,83
9	BBBB	bbbb	14,29
10	CCCC	cccc	14,67
11	MMMM	ddddd	12,33
12	SSSSS	sssss	12,67
13	RRRRR	kkkkkk	12,88
14	QQQQ	hhhh	13,04
15	TTTTT	tttt	13,21
16	LLLLLL	cccc	13,33
17	UUUUUU	uuuuu	13,69
18	HIHHI	bbbb	14,88
19	NNNNN	eeee	15,17
20	OOOOO	fffff	15,29
21	PPPPP	ggggg	16,04
22	JJJJJ	aaaaa	16,38
23			

On fait ensemble…

On arrive donc au code ci-dessus, et lorsqu'on l'exécute, on voit apparaître les nouvelles données dans la feuille « NotesFinales ».