UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Introduction rapide à VBA

Manuele Kirsch Pinheiro
Manuele.Kirsch-Pinheiro@univ-paris1.fr

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

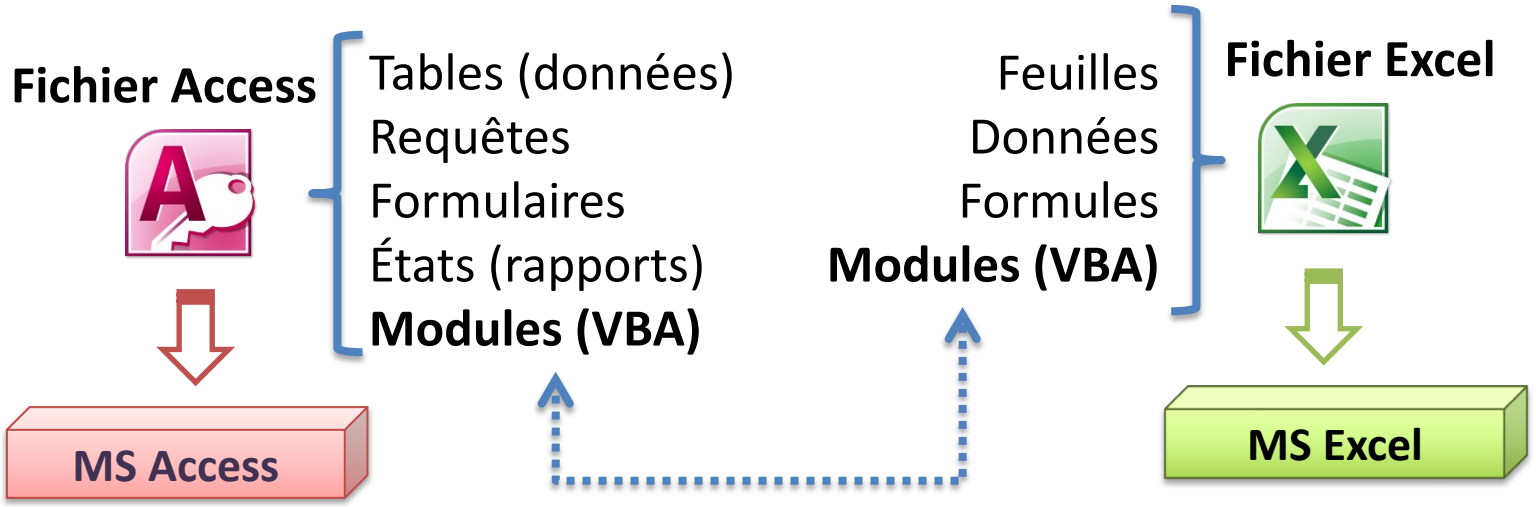
VBA : quoi & pourquoi ?

- 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 et contrôle des données, etc.

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : documents MS Office

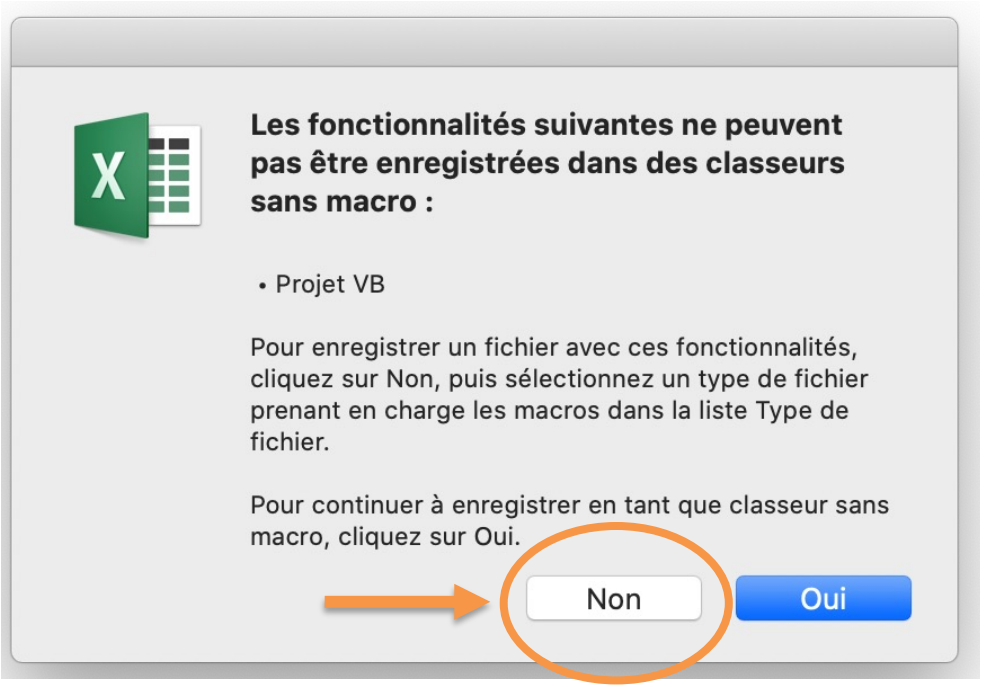
- Un document MS Office est composé des plusieurs éléments…



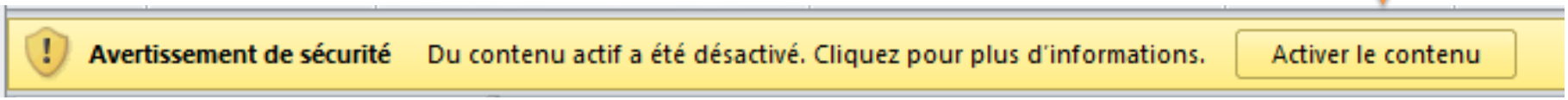
UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

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**

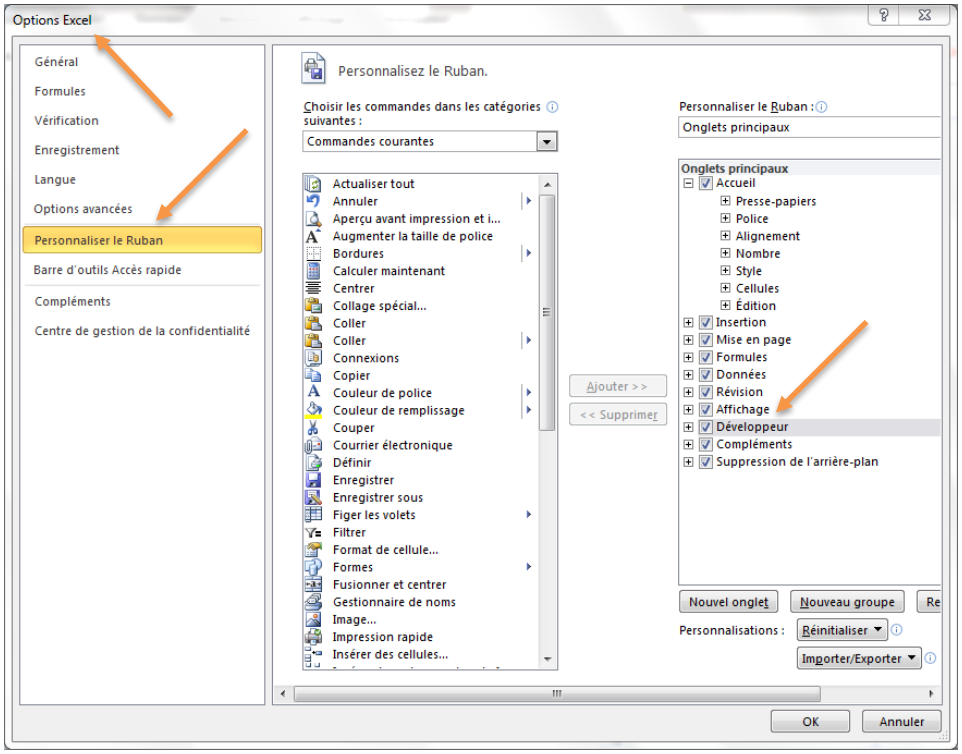


Attention aux risques de sécurité !
Ne ***jamais activer le contenu*** d'un document dont on n'est pas sûr de la provenance.



UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

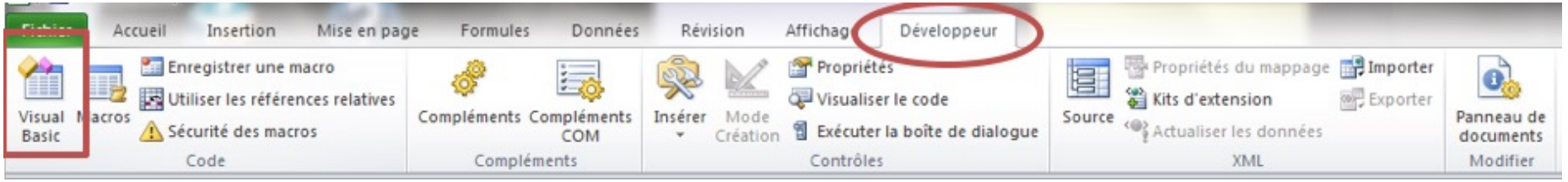
VBA : menu développeur



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

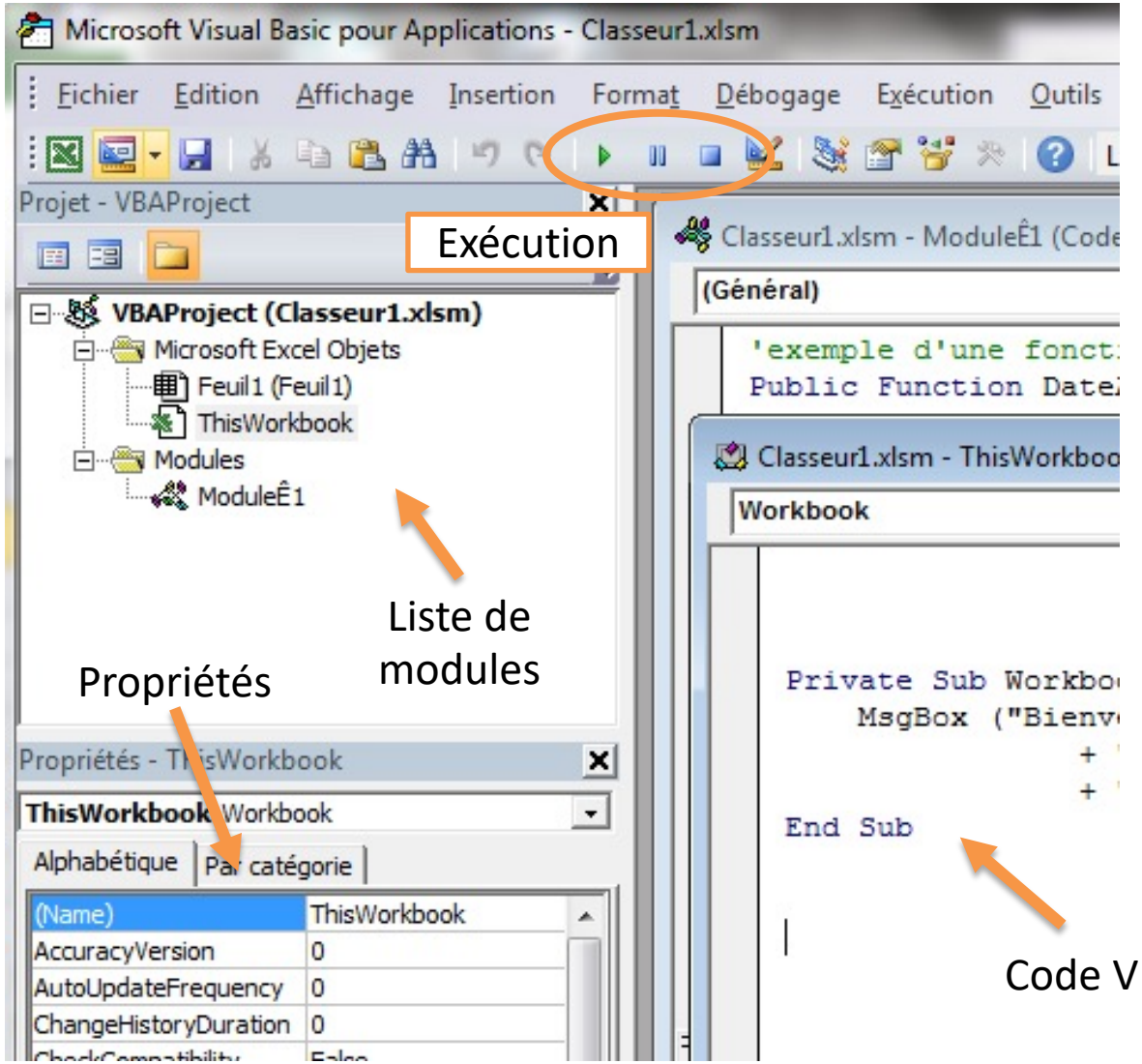
Pour activer le ruban « développeur »
« Options » → « **Personnaliser le Ruban** » et cocher la case « **Développeur** »

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



UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

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

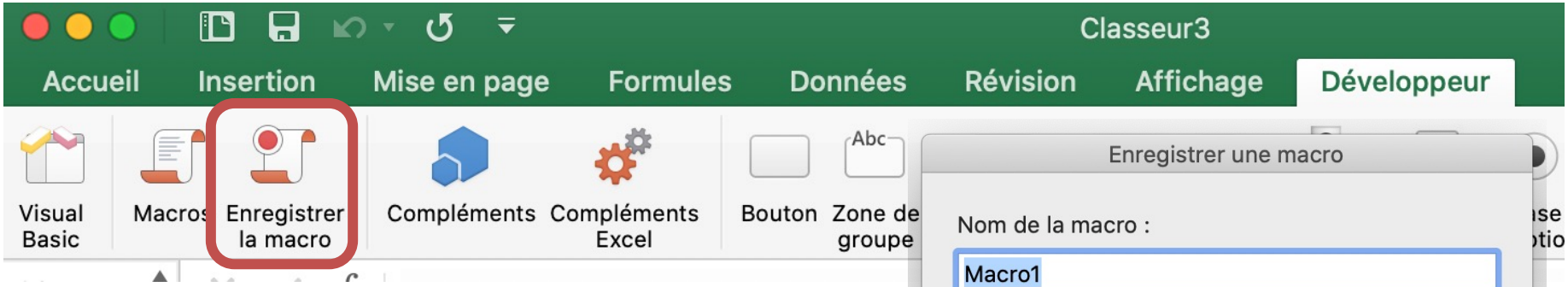
Macros

- **Macros** :
  - suite d'opérations réalisées de manière séquentielle
  - permettent de reproduire ***à l'identique*** un 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

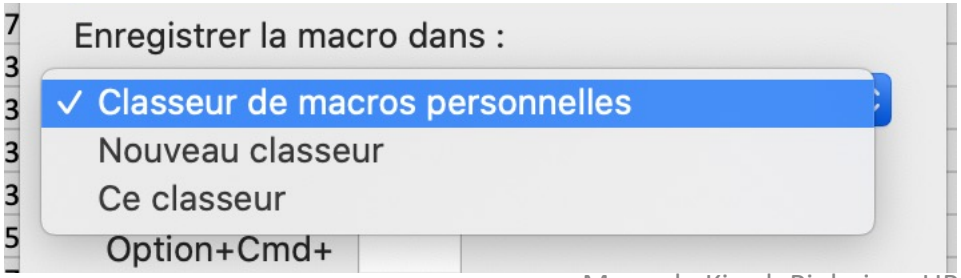
UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

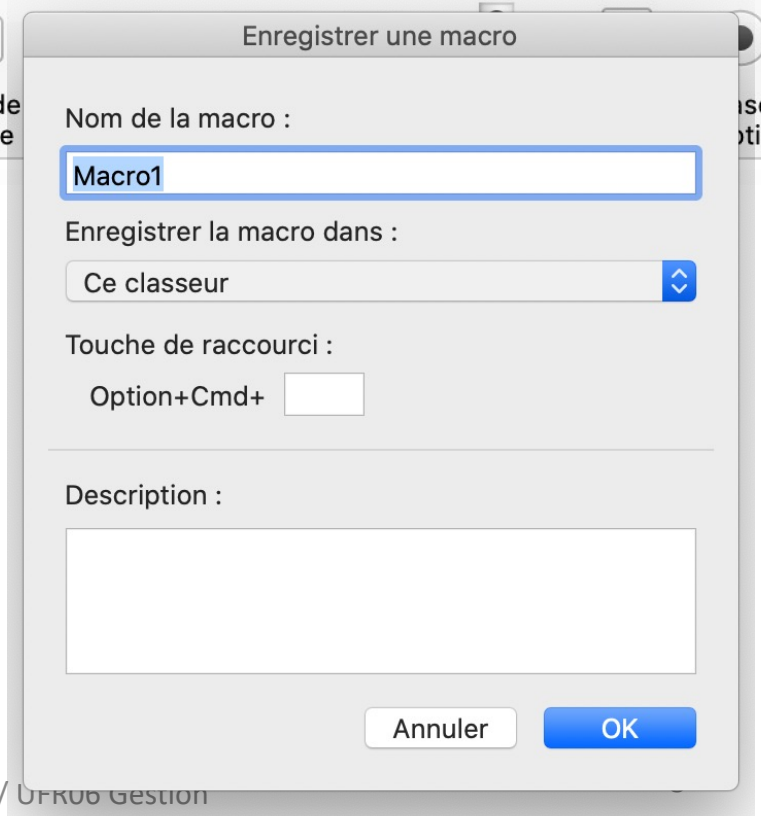
Macros

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



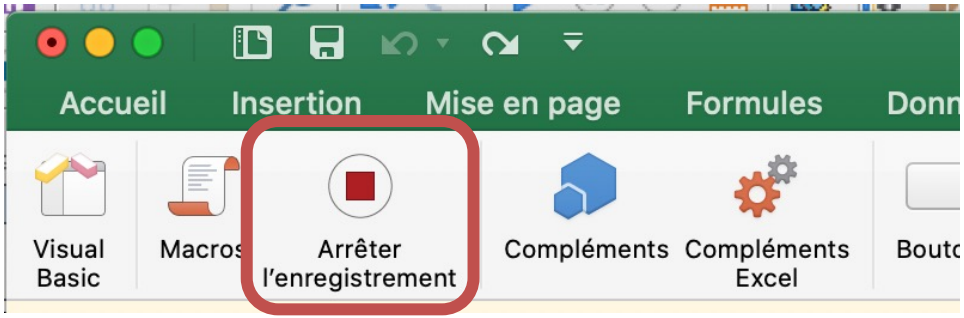
On peut enregistrer dans …
- Le classeur lui-même
- Le classeur de macros personnelles



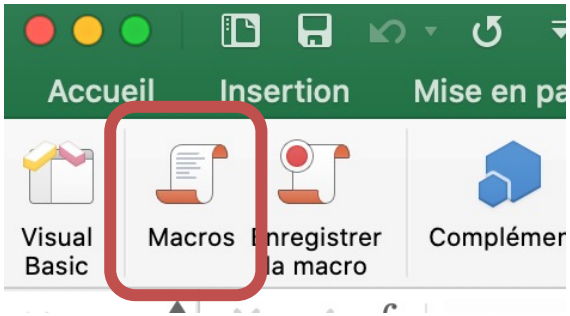


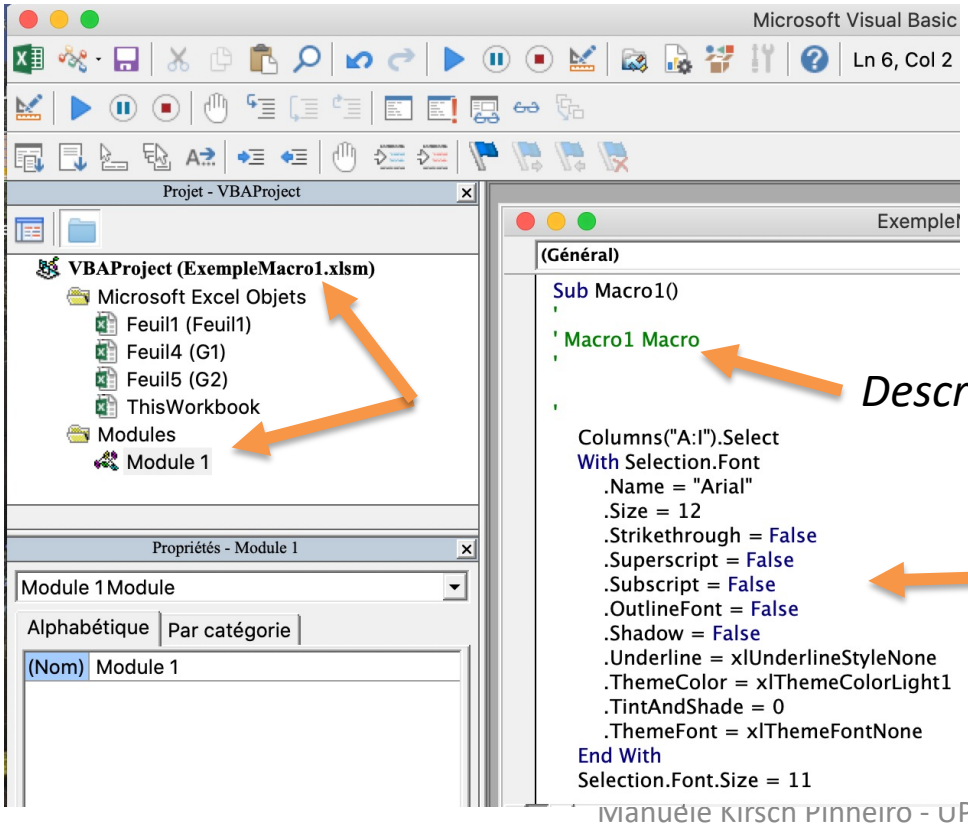
UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Macros



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





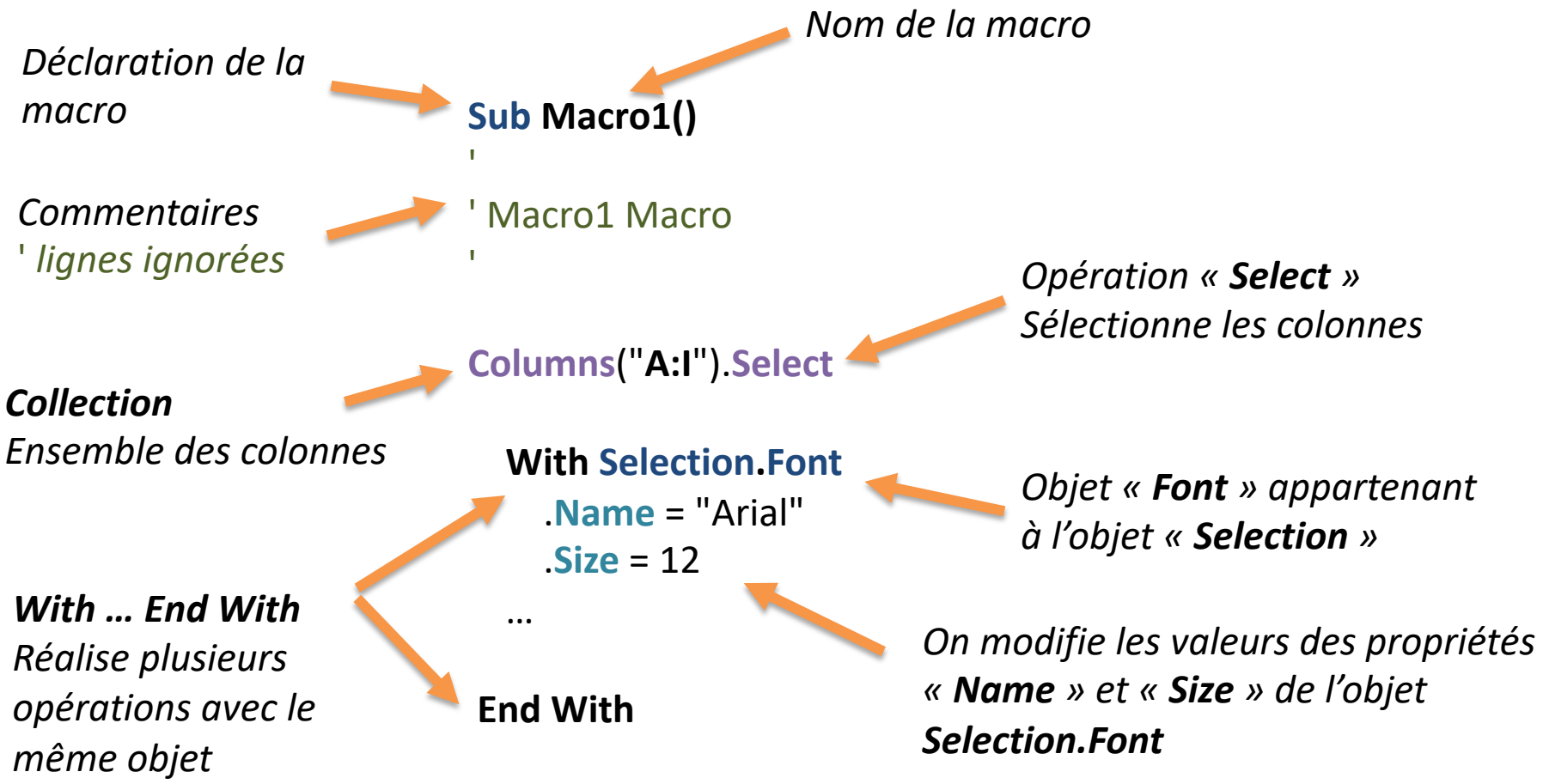
Description

Exemple : macro pour que les données soient en police Ariel taille 12.

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Macros

- On décortique un peu le code…



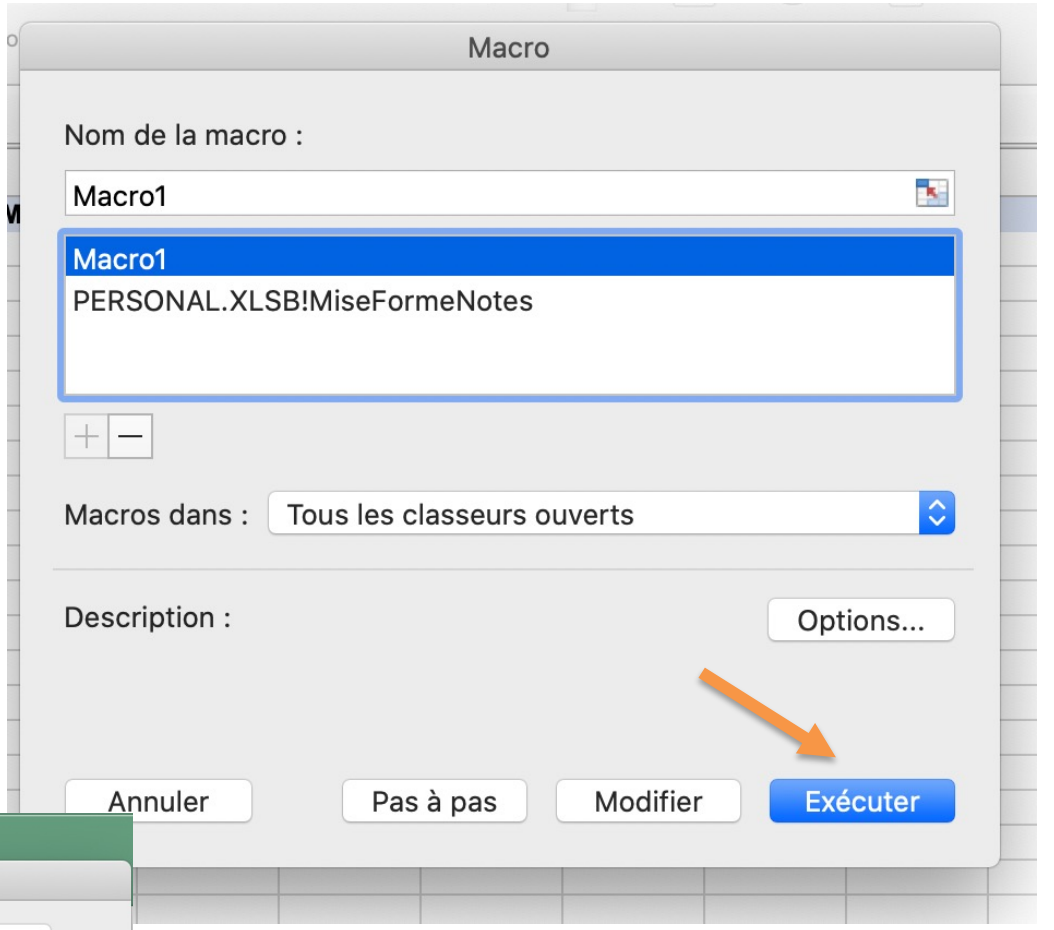
UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

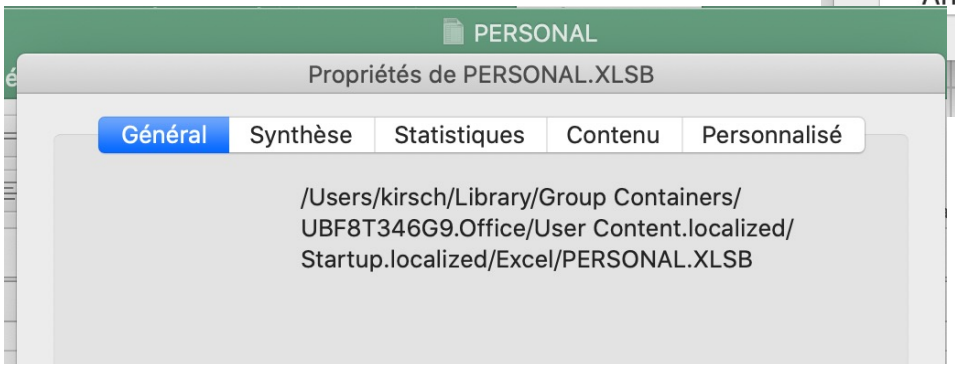
Macros

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

Macros

- **Exécution macro**
  - Par défaut, le code généré utilise des **références absolues** → référence à des cellules et des feuilles précises
  - Pas ou peu reproductible sur une autre feuille ou avec + de données
  - **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**")
Problème si des nouvelles lignes arrivent…

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

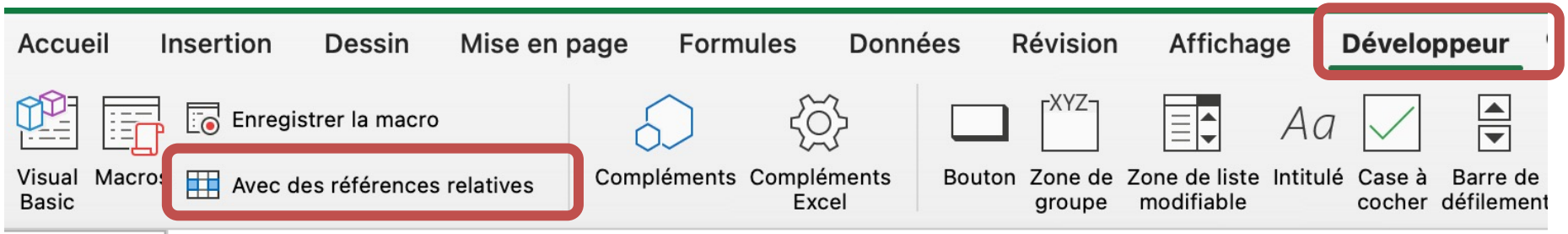
Souvent on **modifie le code** de la macro
- Éliminer les lignes inutiles
- Mieux reproduire la macro ailleurs

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Macros

- Usage des **références relatives**

Sur le ruban « développeur », choisir **« Avec des références relatives »**




```
Sub TrierParNote2()
'
' TrierParNote2 Macro
'
'
    ActiveCell.Offset(-1, -4).Range("A1:I10").Select
    ActiveWorkbook.Worksheets("G1").Sort.SortFields.Clear
    ActiveWorkbook.Worksheets("G1").Sort.SortFields.Add2 Key:=ActiveCell.Offset(0 _
        , 3).Range("A1:A9"), SortOn:=xlSortOnValues, Order:=xlDescending, _
        DataOption:=xlSortNormal
    With ActiveWorkbook.Worksheets("G1").Sort
        .SetRange ActiveCell.Offset(-1, 0).Range("A1:I10")
        .Header = xlYes
        .MatchCase = False
        .Orientation = xlTopToBottom
        .SortMethod = xlPinYin
        .Apply
    End With
End Sub
```

Toujours des références à la feuille G1, mais aussi des « Offset » (décalage) qui peuvent poser problème à la reproduction…

Mais le résultat n'est pas forcément celui qu'on imagine…

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** »

D	E	F	G	H	I
Moy Calc	**Examen**	**Moy CC ca Note1**		**Note2**	**Note3**
14,88	18,50	11,25	9,00	8,00	16,75
16,38	18,50	14,25	18,00	8,00	16,75
13,33	15,75	10,92	12,00	6,00	14,75
12,33	13,00	11,67	18,00	4,00	13,00
15,17	19,00	11,33	13,50	7,00	13,50
15,29	16,75	13,83	16,75	8,00	16,75
16,04	17,25	14,83	16,75	11,00	16,75
13,04	13,50	12,58	16,75	10,00	11,00
12,88	13,50	12,25	14,75	12,00	10,00
12,6666667	11,5	13,8333333	13	16,5	12
13,2083333	12	14,4166667	16,75	10	16,5
13,6875	13,5	13,875	14,75	13	

A1 | *fx* | Groupe

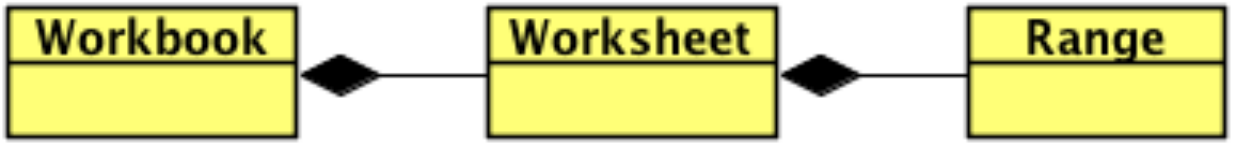
	A	B	C	D	E	F	G	H	I	J
1	**Groupe**	**Nom**	**Prénom**	**Moy Calc**	**Examen**	**Moy CC ca Note1**		**Note2**	**Note3**	
2	G1	AAAA	aaaaa	13,08	12,00	14,17	17,75	8,00	16,75	
3	G1	BBBB	bbbb	14,29	14,00	14,58	19,00	8,00	16,75	
4	G1	CCCC	cccc	14,67	16,75	12,58	17,00	6,00	14,75	
5	G1	DDDD	ddddd	13,29	16,75	9,83	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

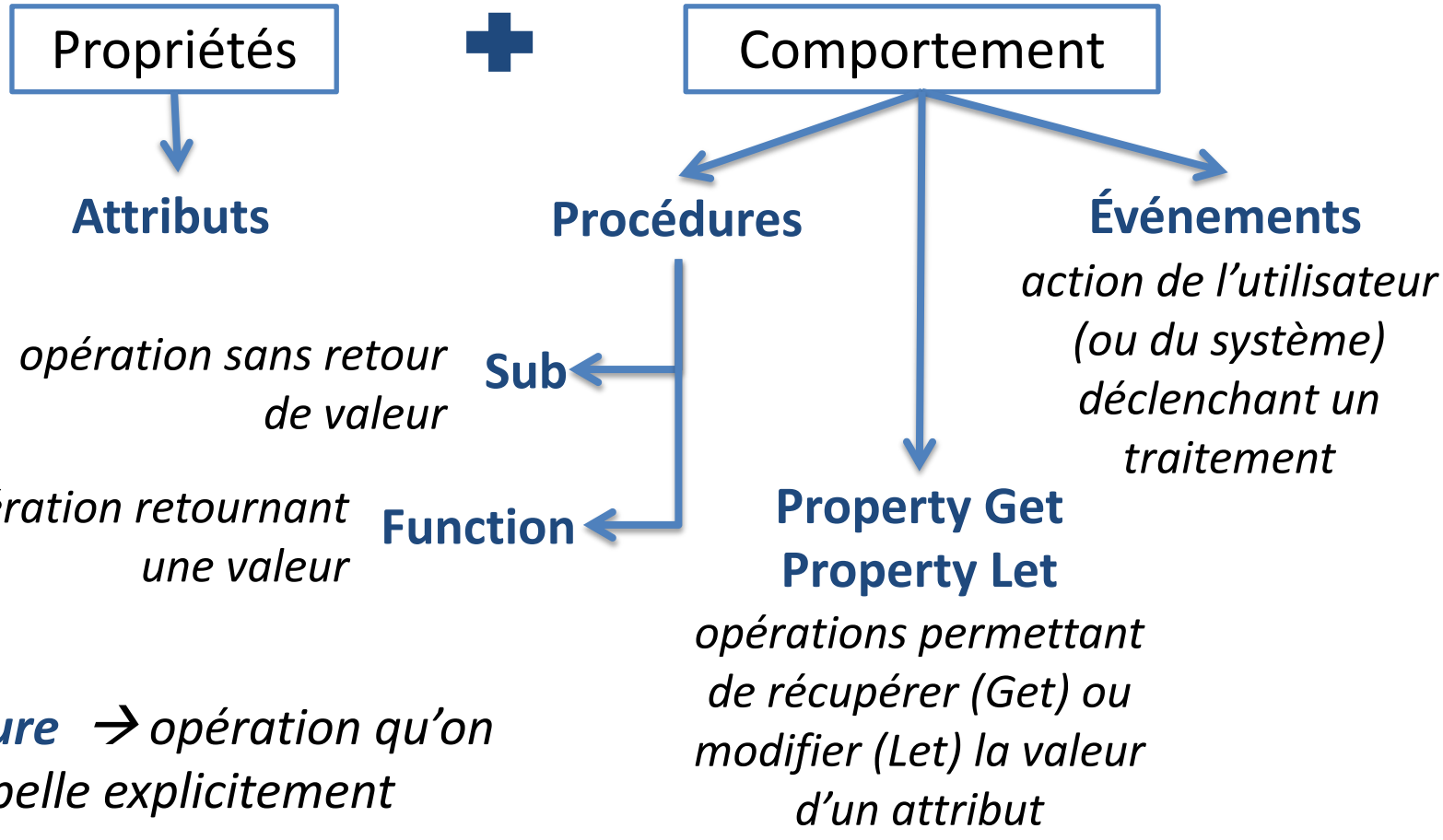


- Les données dans un document Excel (objet **Workbook**) s'organisent en feuilles de calcul (objets **Worksheet**) et en cellules (objets **Range**)

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : Modèle OO

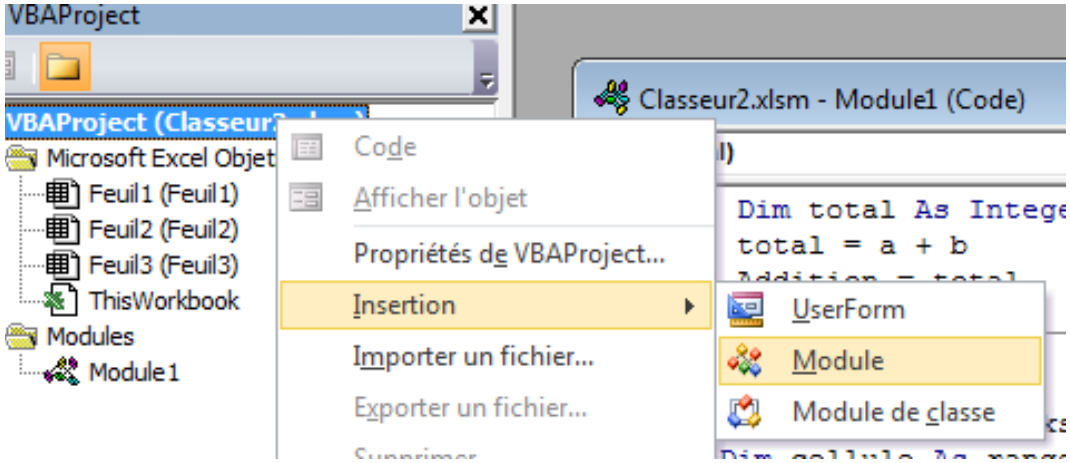
- Un objet VBA possède :



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
  - MS Excel Objets
    - **Evènements** (*Open*, *Close*…)
  - Modules (standards)
    - Procédures et **fonctions** 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 : fonctions & sub

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

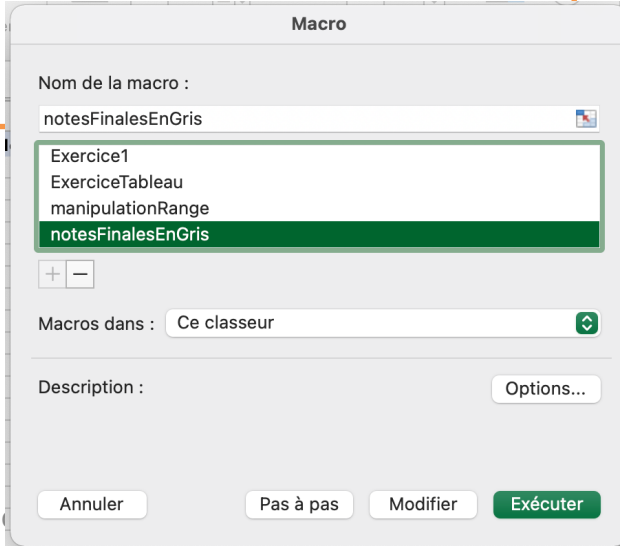
Private | Public Sub *NomProcédure* **(** *paramètre AS type* … **)**

. . .

End Sub

```
Sub notesFinalesEnGris()

    ActiveSheet.Range("D:D").Font.Color = RGB(25, 25, 250)
    ActiveSheet.Range("D:D").Font.Bold = True

End Sub
```

	A	B	C			
	Groupe	Nom	Prénom	Moy Calc	Examen	Moy
	G1	AAAA	aaaaa	13,08	12,00	
	G1	BBBB	bbbb	14,29	14,00	
	G1	CCCC	cccc	14,67	16,75	
	G1	DDDD	ddddd	13,29	16,75	
	G1	EEEE	eeee	11,29	14,75	
	G1	FFFF	fffff	6,17	8,00	
	G1	GGGG	ggggg	13,83	13,75	
	G1	HHHHH	hhhh	8,00	6,00	



UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : événements

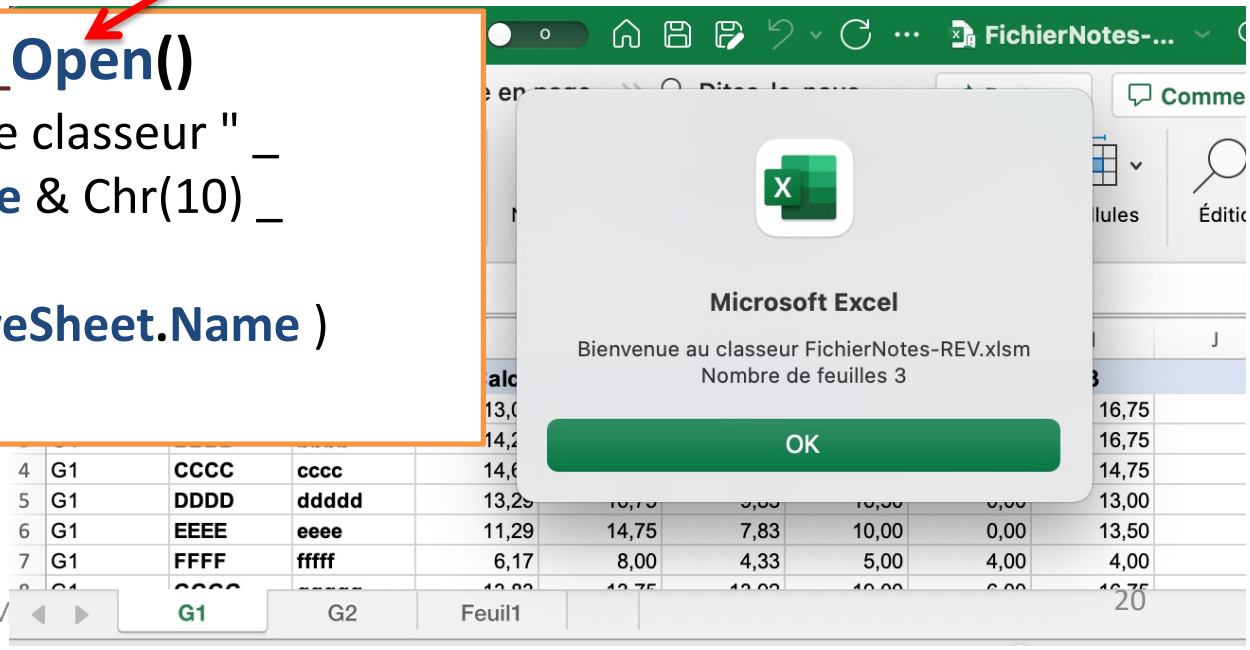
- **Evénements**
  - Exemple : affichage d'un message à l'ouverture d'un document Excel

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

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

quel type d'action
(« **Open** »)

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



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

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

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

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 : Exercices

- Exercice sur le fichier « **Commandes.xlsx** »
  - Enregistrer une macro pour actualiser les données de feuille « **catalogue** » (ruban « **données** ») et les TCDs de la feuille « **TCD** ».
  - Vérifier le code de la macro
  - Sur l'environnement VBE
    - Ouvrir « **ThisWorkbook** »
    - Programmer l'événement « **Workbook_Open** » pour appeler la macro

```
Sub ToutMettreAJour()

    Sheets("catalogue").Select
    ActiveWorkbook.RefreshAll
    Sheets("TCD").Select
    ActiveSheet.PivotTables("Tableau croisé dynamique2").PivotCache.Refresh

End Sub
```

```
Private Sub Workbook_Open()
    ToutMettreAJour
End Sub
```

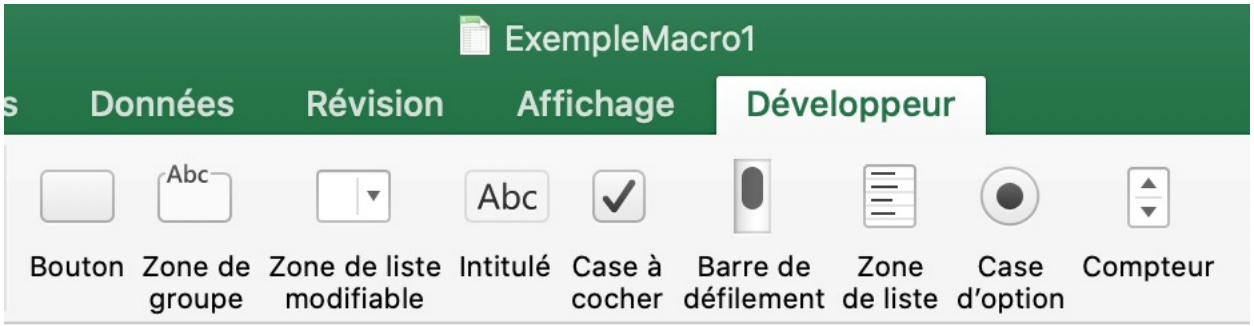
nom de la macro créée

  - Fermer le document, modifier un produit sur « **catalogue.csv** » et réouvrir le fichier

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

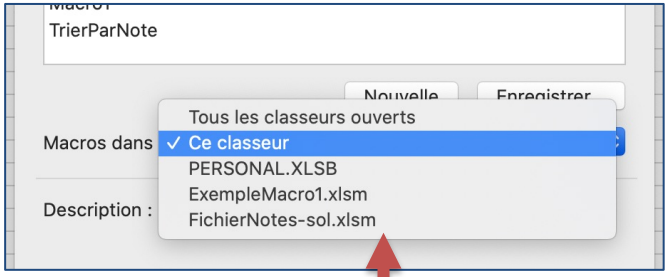
VBA : boutons et autres contrôles

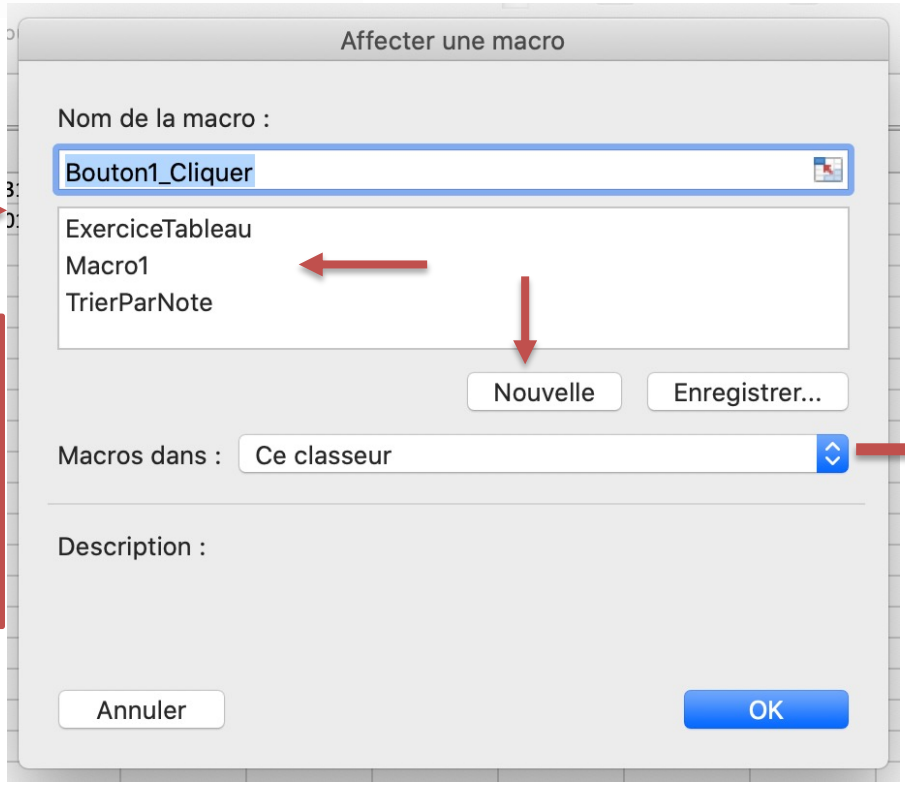
- Des contrôles (boutons, cases à cocher, listes…) sont disponibles sur Excel
- On associe un comportement (code VBA) à ces contrôles



UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : boutons et autres contrôles





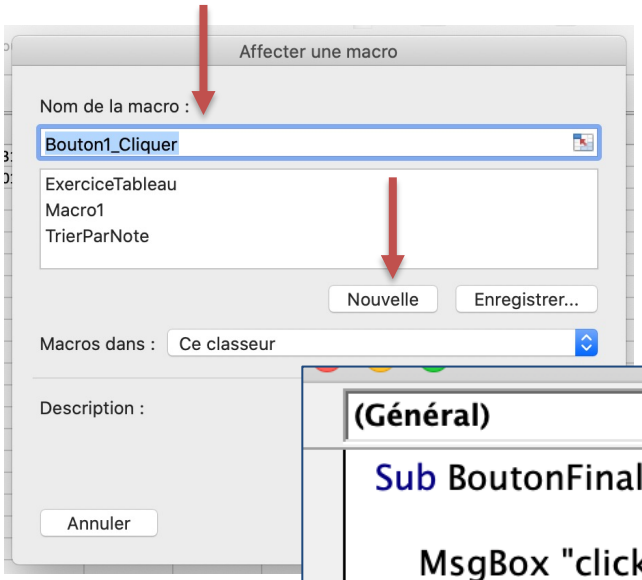
On associe un **comportement** au bouton, en choisissant une **macro existante** ou en créant une **nouvelle**.

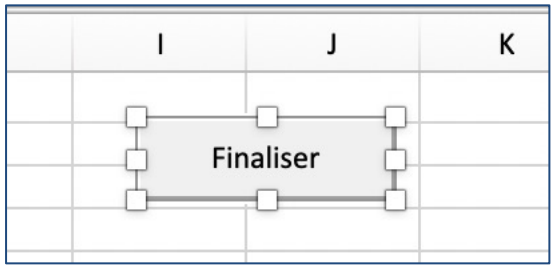
On peut utiliser une **macro** issue d'un **autre fichier** ouvert (ou du **PERSONAL.XLSB**).

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

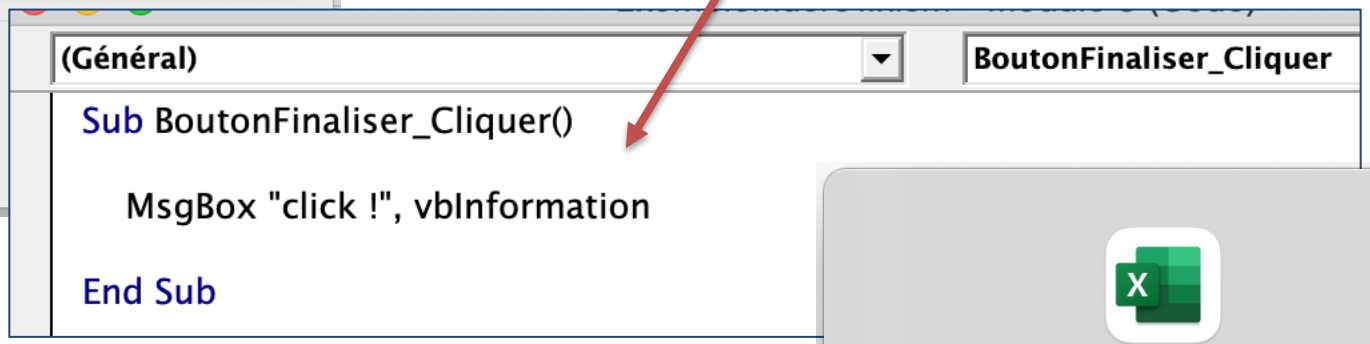
VBA : boutons et autres contrôles

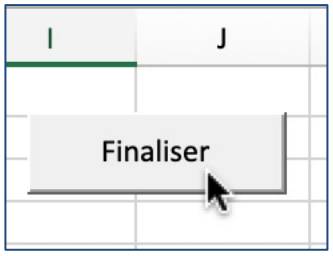
1) On insère le bouton
(penser à lui donner un nom logique)



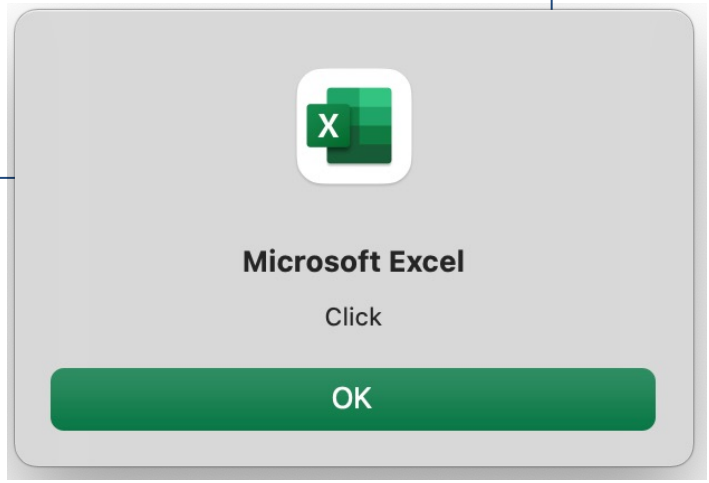


2) On programme son comportement
*(**événement** clique)*





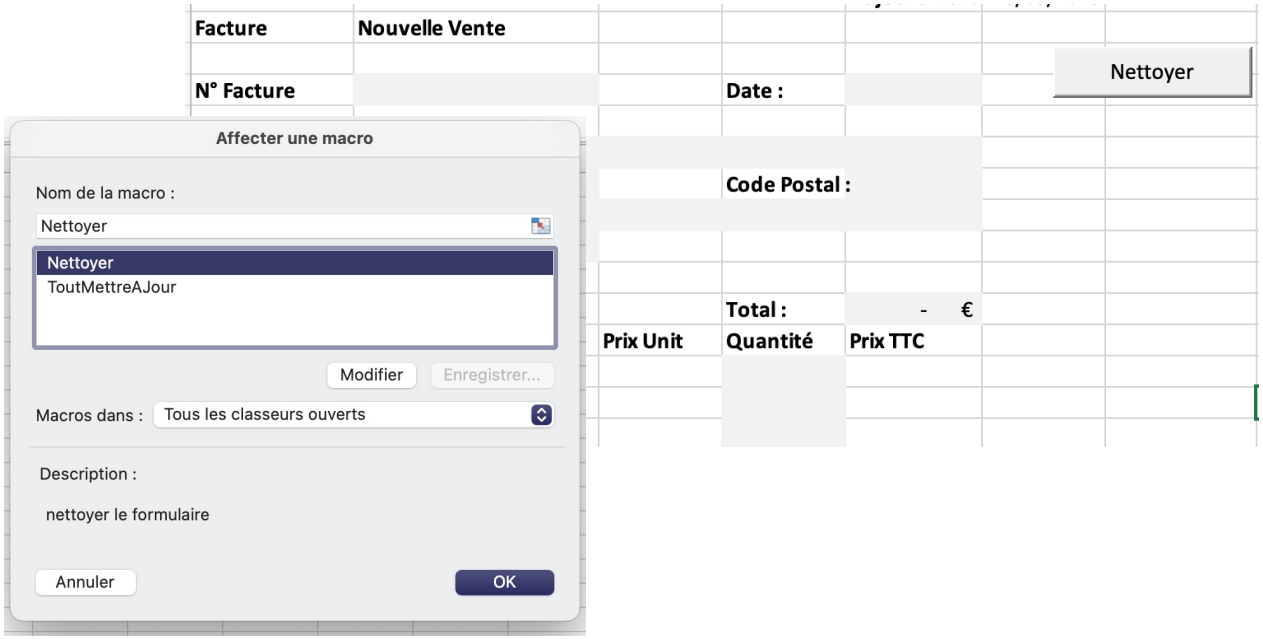
*3) Lorsqu'on clique sur le bouton, le code associé est **exécuté***



Exercice

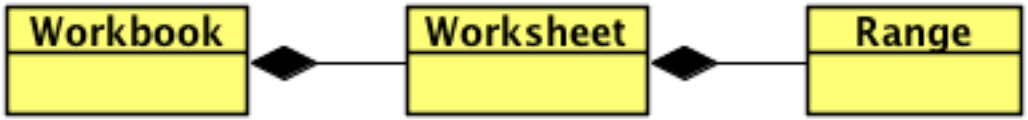
- Sur le fichier « **Commandes.xlsx** » :
  - Enregistrer une macro qui supprime l'ensemble des valeurs du formulaire de la feuille « **formulaire** »
  - Vérifier le code de la **macro**
  - Ajouter dans cette même feuille un bouton « **nettoyer** » et l'associer à cette **macro**

```
(Général)
Sub Nettoyer()

    Range("B4").Select
    Selection.ClearContents
    Range("E4").Select
    Selection.ClearContents
    Range("B6").Select
    Selection.ClearContents
    Range("B7").Select
    Selection.ClearContents
    Range("B8").Select
    Selection.ClearContents
    Range("B9").Select
    Selection.ClearContents
    Range("E7").Select
    Selection.ClearContents
   Selection.ClearContents
    Range("A13:A25").Select
    Selection.ClearContents
    Range("D13:D25").Select
    Selection.ClearContents
    Range("B4").Select
End Sub
```



UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : accès aux données EXCEL



- Quelques objets importants
  - **Application** : application (Excel) active
  - **WorkSheetFunction** : ensemble de fonctions Excel
    - Attention : nom des fonctions en anglais
  - **Workbooks** : collection de classeurs ouverts
  - **WorkSheets** : collection de feuilles de calcul d'un classeur

  } ***Collections***

  - **ThisWorkbook** : document courant
  - **ActiveWorkbook** : document en 1er plan (actif)
  - **ActiveSheet** : feuille de calcul active
  - **ActiveCell** : cellule (Range) active

  } *Propriétés Application et/ou* ***Workbook***

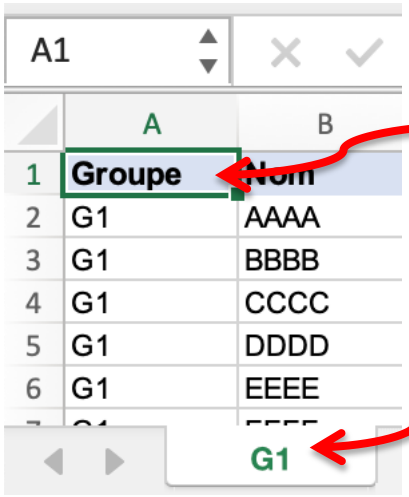
UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : Collections

- **Collections**
  - Ensemble d'objets d'un même type
    - **Worksheets** (toutes feuilles de calcul), **Range** (ensemble de cellules)…
  - Les **collections** vont permettre l'accès aux différents objets de la classe correspondante
    - **Worksheets**("Feuil1") → accès à la feuille « Feuil1 »

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

Worksheets(*"G1"*)
Worksheets(*1*)



Cellule
(1,1)

Feuille
« G1 »

Worksheets("G1").Cells(1,1)

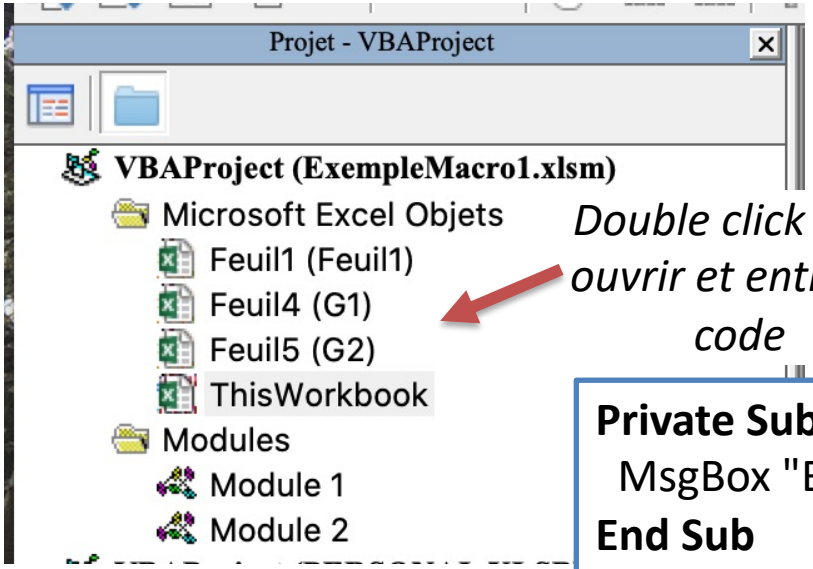
Worksheets(1).Range("A1")

Cells(ligne , colonne)

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**

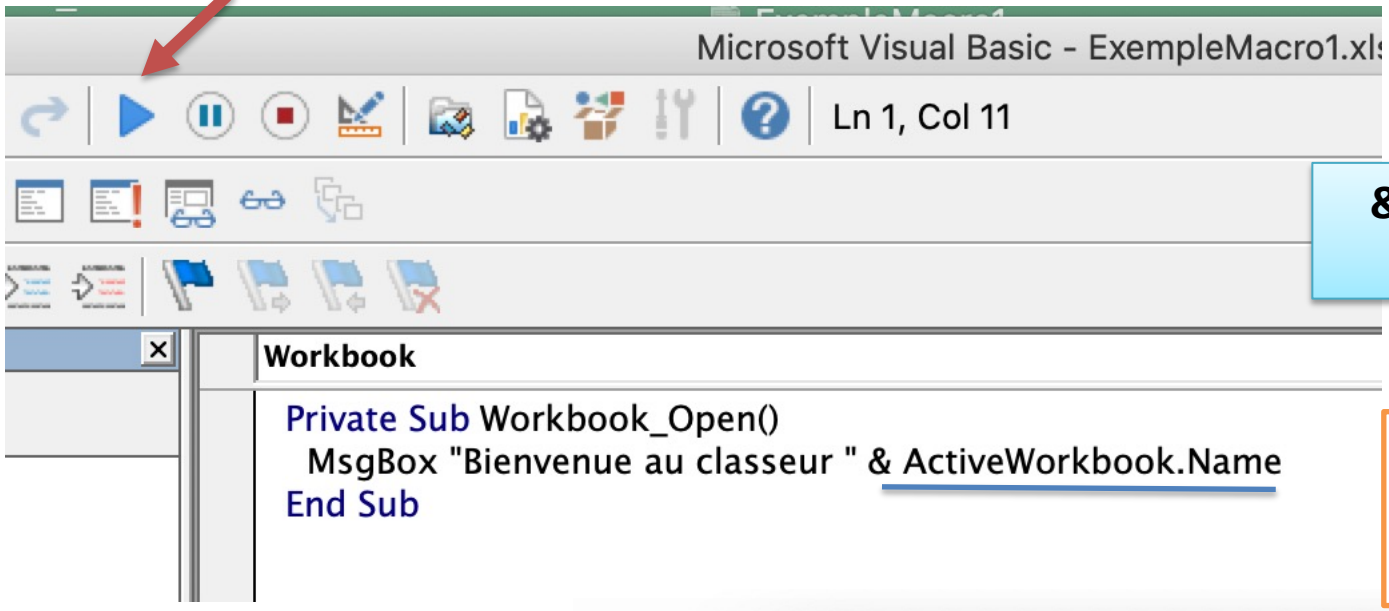
ActiveWorkbook.Name → propriété **Name** de l'objet **ActiveWorkbook**

```
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

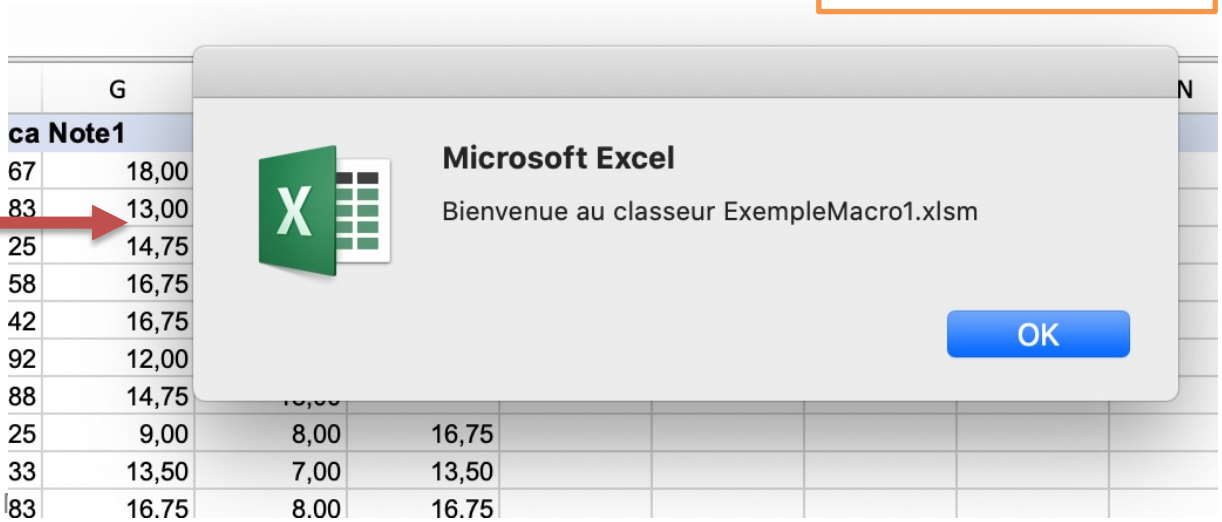
On peut exécuter directement la Sub



& : opérateur de concaténation

Notation :
Objet.Propriété
Objet.Opération

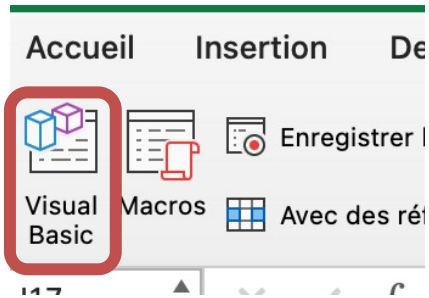
Lors de l'ouverture du fichier

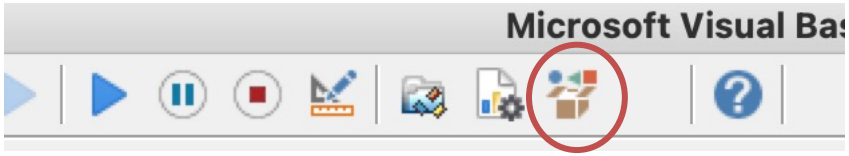


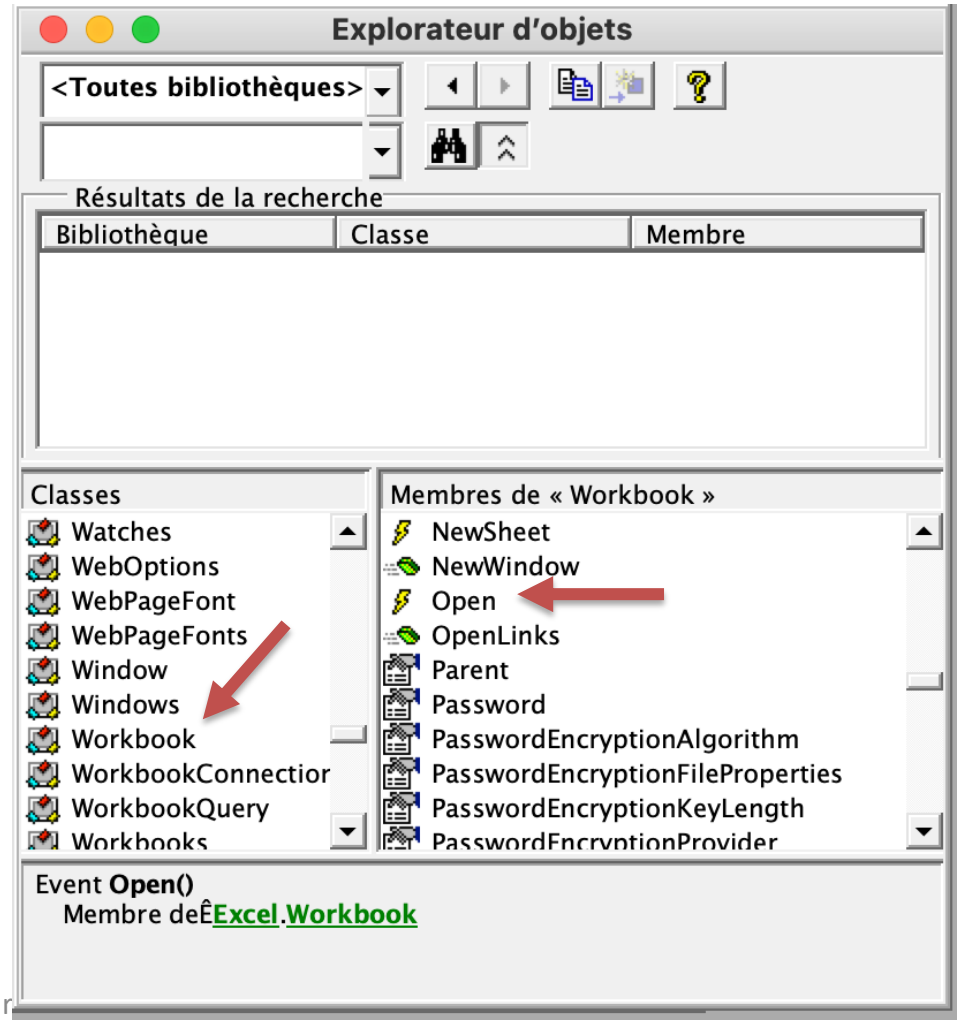
UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : comment trouver les informations ?

- Comment trouver les informations sur une classe ou un objet ?
  - **Explorateur d'objets**







VBA : Exemple

- **Exercice** :
  - Améliorer le code d'ouverture du classeur « **FichierNotes.xlsx** »
    - Afficher le **nombre de feuilles** dans le classeur
    - **MsgBox** → affiche fenêtre avec un message
    - 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

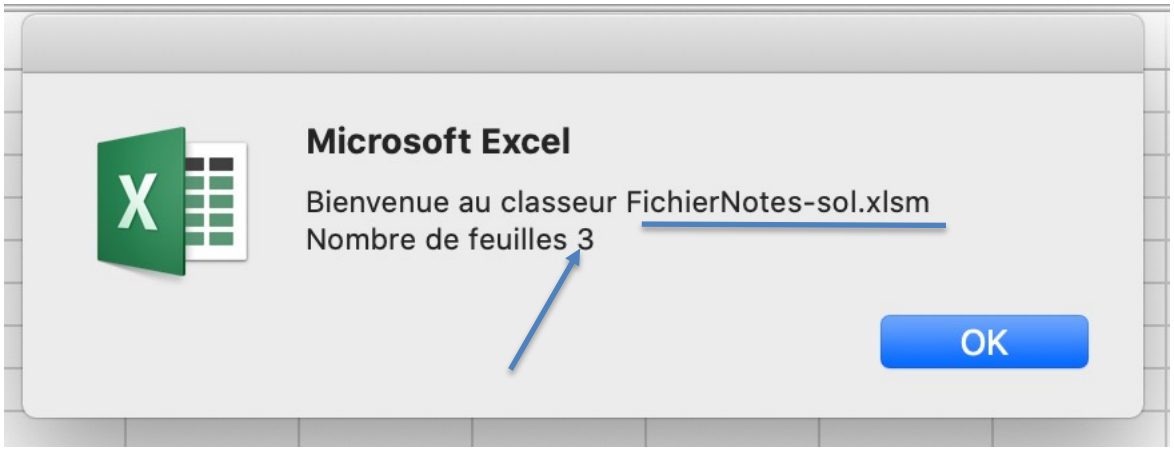
VBA : Exercice

Workbook | Open

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

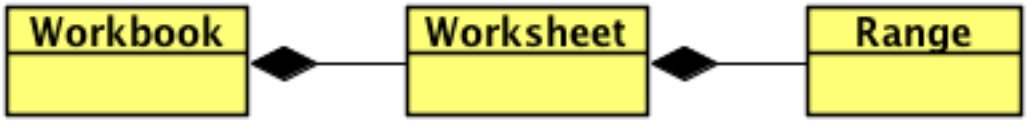
Objet **ActiveWorkbook**
Propriété **Name**

Collection **Worksheets**
Propriété **Count**



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
```

La propriété **Value** récupère la **valeur de la cellule**

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

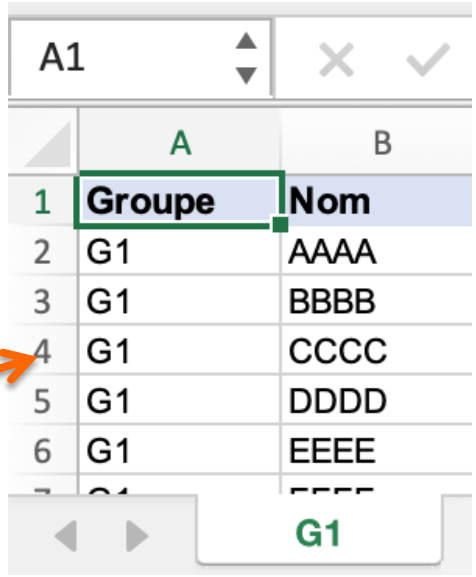
La collection **Range** permet d'indiquer un ensemble de cellules **Range("A1:B1")**

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

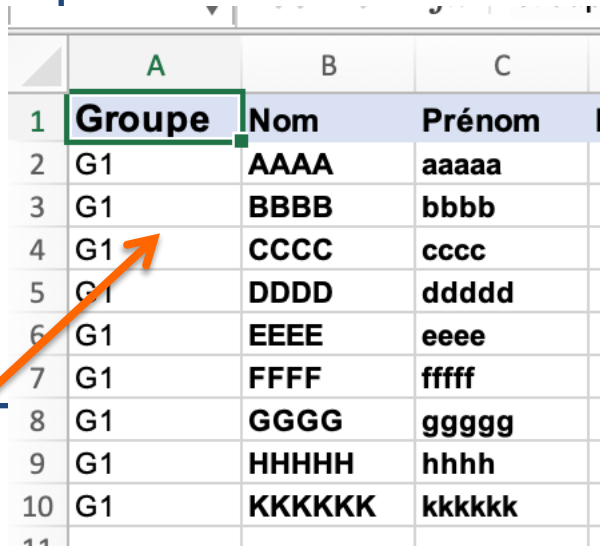
VBA : accès aux données EXCEL

Attention
Cells(ligne , colonne)

```
Sub manipulationRange()

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

    Worksheets("G1").Cells(1, 1).Font.Size = 14
    Worksheets("G1").Range("B:C").Font.Bold = True

End Sub
```



changement de la taille et gras de la police



Exercice

- **Exercice** :
  - Toujours dans le classeur « **FichierNotes.xlsx** »,
  - Modifier l'événement « **Open** » afin qu'il puisse placer les colonnes « **Nom** » et « **Prénom** » 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(1) → Récupérer la 1ème feuille du classeur

Worksheets("G1").Range("B:C") → range B:C de la feuille G1
Worksheets(1). Range("B:C") → range B:C de la 1ème feuille

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Exercice

(Général) (Déclarations)

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

    Worksheets("G1").Range("B:C").Font.Bold = True
    Worksheets(2).Range("B:C").Font.Bold = True

End Sub
```

Deux possibilités, avec
***Worksheets*("G1") et G2**
ou ***Worksheets*(1) et 2**

Attention au changement d'ordre des feuilles

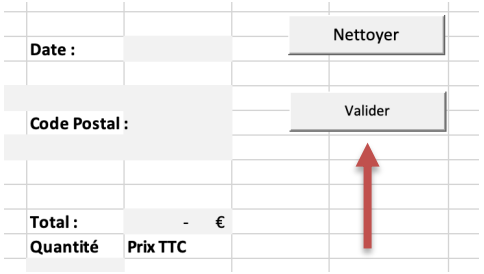
B2 | fx | AAAA

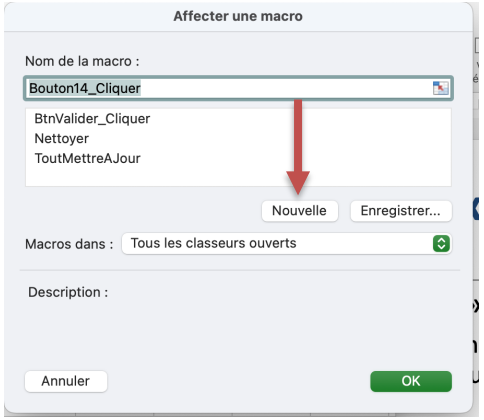
	A	B	C	D	E	F	G
1	Groupe	Nom	Prénom	Moy Calc	Examen	Moy CC ca	Note1
2	G1	AAAA	aaaaa	13,08	12,00	14,17	17,
3	G1	BBBB	bbbb	14,29	14,00	14,58	19,
4	G1	CCCC	cccc	14,67	16,75	12,58	17,
5	G1	DDDD	ddddd	13,29	16,75	9,83	16,
6	G1	EEEE	eeee	11,29	14,75	7,83	10,
7	G1	FFFF	fffff	6,17	8,00	4,33	5,
8	G1	GGGG	ggggg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

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Exercice

- Sur le fichier « **Commandes.xlsx** » :
  - Sur la feuille « **formulaire** », on souhaite copier les informations sur la nouvelle facture vers la feuille « **facture** »
  - Ajouter un bouton « **valider** » à cette feuille
  - Programmer une **nouvelle macro** pour ce bouton





*2° copier les informations de la feuille « **formulaire** » à la feuille « **facture** »*

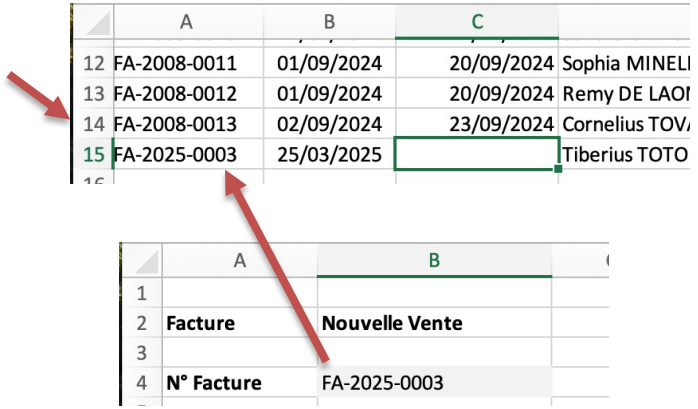
	A	B	C	D	E
1					**Aujourd'hui :**
2	**Facture**	**Nouvelle Vente**			
3					
4	**N° Facture**	FA-2025-0003		**Date :**	25/03/2025
5					
6	**Client :**	Tiberius TOTO			
7	**Ville :**	Paris		**Code Postal**	75005
8	**Adresse :**	2 rue cujas			
9	**Téléphone :**	01 23 45 56 78			

1° trouver où ajouter (après la dernière facture)

	A	B	C	D	E	F	G	H	I
12	FA-2008-0011	01/09/2024	20/09/2024	Sophia MINELLI	MINELLI	59700	Marq-en-Bare	9 rue de Venise	09 40 38 62 48
13	FA-2008-0012	01/09/2024	20/09/2024	Remy DE LAON	DE LAON	51100	Reims	2 rue Simon	06 66 44 77 11
14	FA-2008-0013	02/09/2024	23/09/2024	Cornelius TOVALSKI	TOVALSKI	33025	Bordeaux	32 rue de rivière	07 82 92 21 13
15	FA-2025-0003	25/03/2025		Tiberius TOTO		75005	Paris	2 rue cujas	01 23 45 56 78

Exercice

- Sur le fichier « **Commandes.xlsx** » :
  - Trouver la ligne de la dernière facture



	A	B	C	
12	FA-2008-0011	01/09/2024	20/09/2024	Sophia MINELI
13	FA-2008-0012	01/09/2024	20/09/2024	Remy DE LAOI
14	FA-2008-0013	02/09/2024	23/09/2024	Cornelius TOV/
15	FA-2025-0003	25/03/2025		Tiberius TOTO

	A	B
1		
2	**Facture**	**Nouvelle Vente**
3		
4	**N° Facture**	FA-2025-0003

```
Set Factures = Worksheets("Factures").Range("A:A")
dernièreFacture = WorksheetFunction.CountA(Factures)
```

  - Copier les données vers cette ligne

```
prochaineFacture = dernièreFacture + 1

Worksheets("Factures").Range("A" & prochaineFacture).Value =  _
                                    Worksheets("Formulaire").Range("B4").Value

Worksheets("Factures").Range("B" & prochaineFacture).Value = _
                                    Worksheets("Formulaire").Range("E4").Value
```

(faire la même chose pour les autres informations sur la nouvelle facture, hors ces produits)

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Exercice

- Sur le fichier « **Commandes.xlsx** » :
  - Modifier la macro pour qu'elle puisse **copier les formules** contenues dans les autres colonnes de la feuille « **facture** » à partir de la dernière facture

Worksheet().Range().Copy Destination

```
Worksheets("Factures").Range("C" & dernièreFacture).Copy _
                                    Worksheets("Factures").Range("C" & prochaineFacture)

Worksheets("Factures").Range("E" & dernièreFacture).Copy _
                                    Worksheets("Factures").Range("E" & prochaineFacture)

Worksheets("Factures").Range("J" & dernièreFacture & ":N" & dernièreFacture).Copy _
            Worksheets("Factures").Range("J" & prochaineFacture & ":N" & prochaineFacture)
```

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
10	FA-2008-0009	23/08/2024	13/09/2024	François BASCHET	BASCHET	04210	Brunet	12 rue de Tinqueux	06 39 29 45 40	1 125,14 €	56,26 €	1 181,40 €	4	Colissimo
11	FA-2008-0010	22/07/2024	12/08/2024	Sandra CEREJA	CEREJA	99 999	Santa Maria, E	Tv Domingos Trevisan	00 55 4 68 79 8(	241,05 €		241,05 €	99	A DEFINIR
12	FA-2008-0011	01/09/2024	20/09/2024	Sophia MINELLI	MINELLI	59700	Marq-en-Bare	9 rue de Venise	09 40 38 62 48	443,63 €		443,63 €	59	Toto Transpor
13	FA-2008-0012	01/09/2024	20/09/2024	Remy DE LAON	DE LAON	51100	Reims	2 rue Simon	06 66 44 77 11	138,32 €		138,32 €	51	Camion propr
14	FA-2008-0013	02/09/2024	23/09/2024	Cornelius TOVALSKI	TOVALSKI	33025	Bordeaux	32 rue de rivière	07 82 92 21 13	1 040,54 €		1 040,54 €	33	Colissimo
15	FA-2025-0003	25/03/2025	15/04/2025	Tiberius TOTO	TOTO	75005	Paris	2 rue cujas	01 23 45 56 78	- €		- €	75	Toto Transpor

Exercice

- Code complet

```
Sub BtnValider_Cliquer()
  ' on retrouve la ligne de la dernière facture
  Set Factures = Worksheets("Factures").Range("A:A")
  derniereFacture = WorksheetFunction.CountA(Factures)

  'on rajoute une nouvelle ligne avec les informations issues du formulaire
  prochaineFacture = dernieFacture + 1
  Worksheets("Factures").Range("A" & prochaineFacture).Value = Worksheets("Formulaire").Range("B4").Value 'n° facture
  Worksheets("Factures").Range("B" & prochaineFacture).Value = Worksheets("Formulaire").Range("E4").Value  'date
  Worksheets("Factures").Range("D" & prochaineFacture).Value = Worksheets("Formulaire").Range("B6").Value 'client
  Worksheets("Factures").Range("F" & prochaineFacture).Value = Worksheets("Formulaire").Range("E7").Value  'code postal
  Worksheets("Factures").Range("G" & prochaineFacture).Value = Worksheets("Formulaire").Range("B7").Value 'ville
  Worksheets("Factures").Range("H" & prochaineFacture).Value = Worksheets("Formulaire").Range("B8").Value 'adresse
  Worksheets("Factures").Range("I" & prochaineFacture).Value = Worksheets("Formulaire").Range("B9").Value 'telephone

  'puis on recopie les formules à partir de la dernière facture pour les autres colonnes
  'format : feuille.Range(range).Copy destination
  Worksheets("Factures").Range("C" & dernieFacture).Copy Worksheets("Factures").Range("C" & prochaineFacture)
  Worksheets("Factures").Range("E" & dernieFacture).Copy Worksheets("Factures").Range("E" & prochaineFacture)
  Worksheets("Factures").Range("J" & dernieFacture & ":N" & dernieFacture).Copy _
                          Worksheets("Factures").Range("J" & prochaineFacture & ":N" & prochaineFacture)
```

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

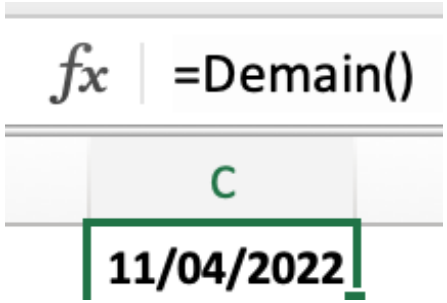
VBA : fonctions & sub

- **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 Demain() As Date
    Dim auj As Date
    auj = Date
    Demain = auj + 1
End Function
```

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

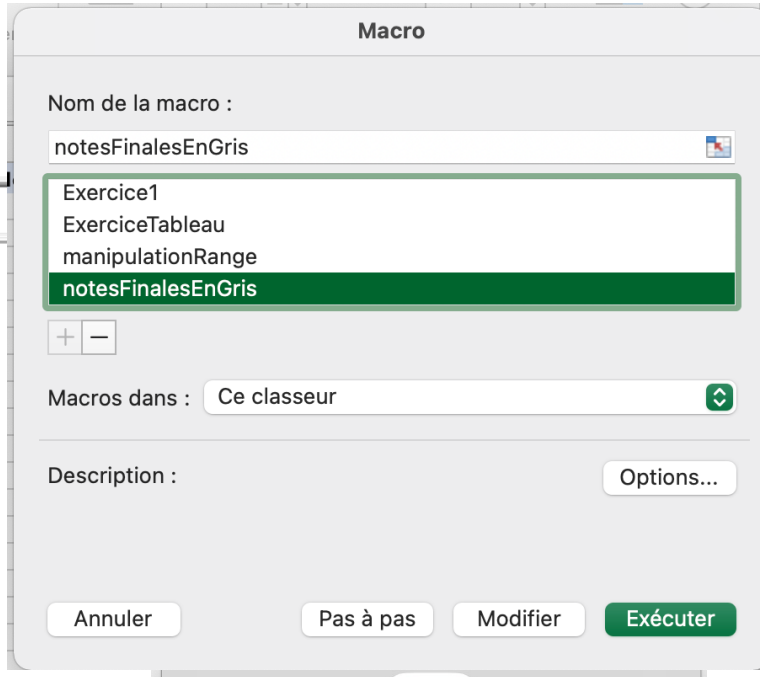
VBA : évènements & procédures

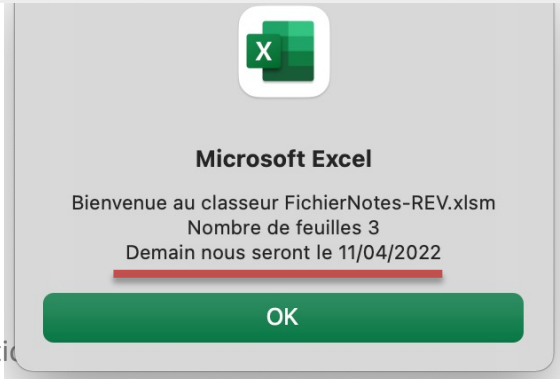
- **Evènements & Procédures**
  - Les **Function** et **Sub** peuvent être invoquées dans un événement ou dans d'autres procédures **Sub** / **Function**
  - Les **Sub** sont perçues comme des **macro**

Workbook | Open

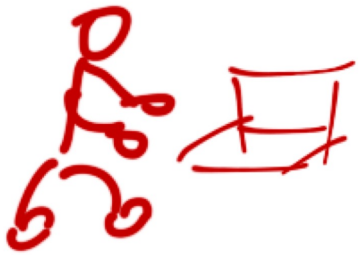
```
Private Sub Workbook_Open()
    MsgBox "Bienvenue au classeur " & ActiveWorkbook.Name _
     & Chr(10) _
     & "Nombre de feuilles " & Worksheets.Count _
     & Chr(10) _
     & "Demain nous seront le " & JourApres(Date)

    Worksheets("G1").Range("B:C").Font.Bold = True
    Worksheets(2).Range("B:C").Font.Bold = True

End Sub
```

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





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 : 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

10 / 4 / 2022

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 auj As Date

a = 2
auj = Date
```

1) déclaration

2) affectation

```
Dim cellule As Range

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

cellule.Valeur = auj + a
```

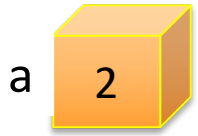
3) usage

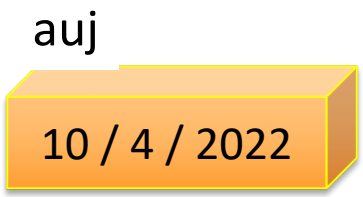
Propriété **Valeur** de la classe **Range**

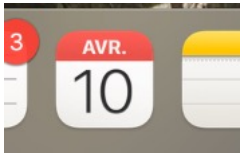
UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : variables

```
Sub ExempleVariable()
   Dim auj As Date
   Dim a As Integer
   Dim cellule As Range

   a = 2
   auj = Date   ' date actuelle

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

   cellule.Value = auj + a

End Sub
```



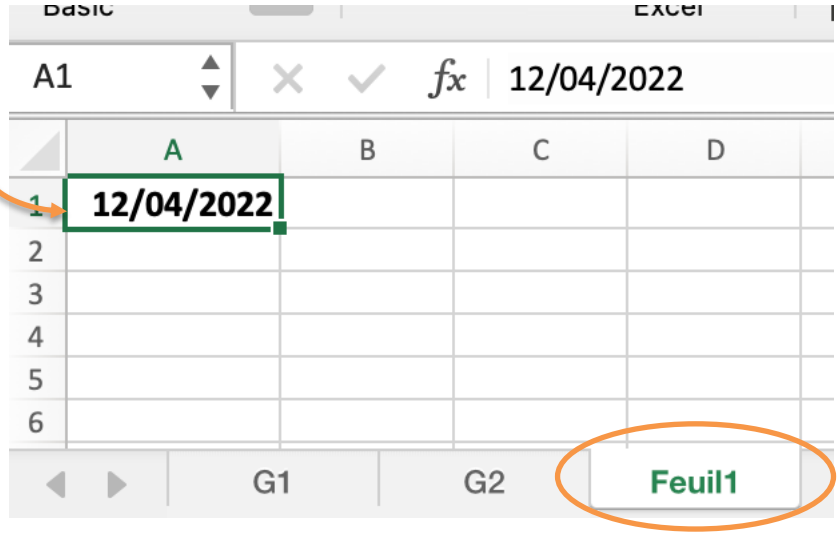




Aujourd'hui
10 / 04 / 2022

Objet « cellule »
Correspond à la cellule A1

On modifie sa valeur
(propriété **Value**)



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

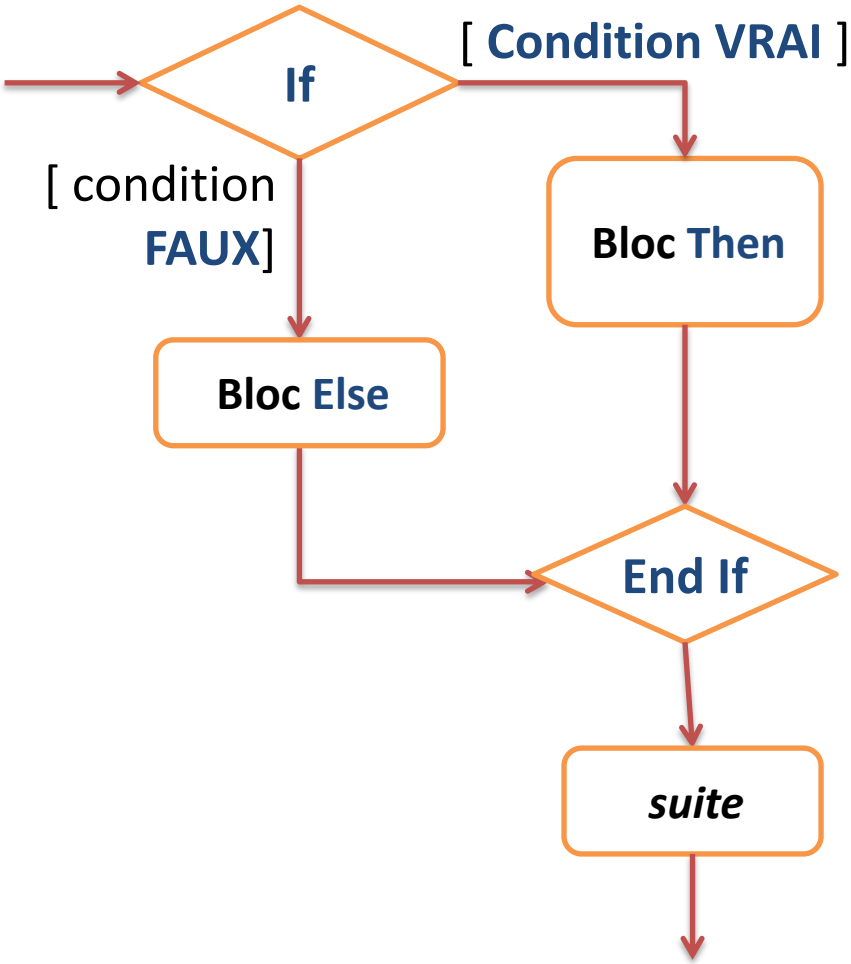
```
Worksheets("Feuil1").Range("D1").Formula = "=TODAY()"
Worksheets("Feuil1").Range("D1").Font.Italic = True
```

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
```

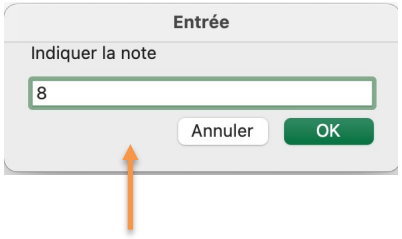
UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : Instructions de contrôle

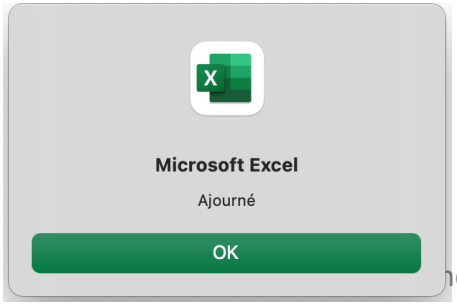
- Instructions conditionnelles : **if…elseIf…else… end if**

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

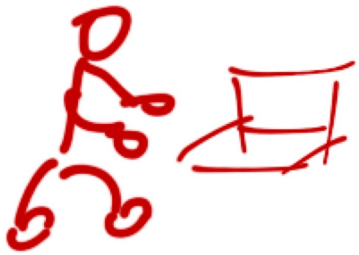
```
Dim mont As Double
mont = Application.InputBox("Indiquer la note")

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



InputBox

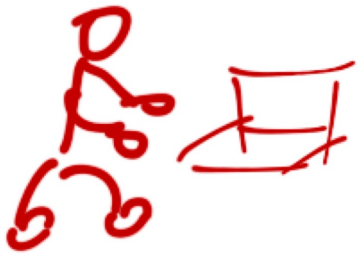


Exercice



- **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)

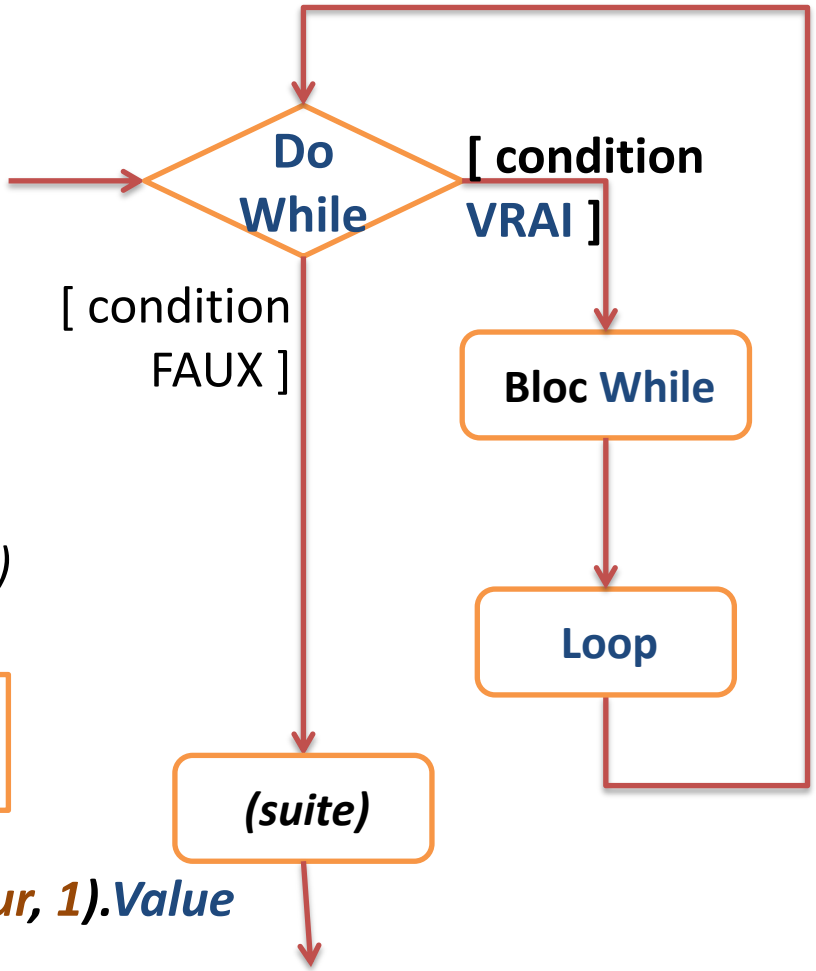
	D	E	F	G	H	I	J
1	Moy Calc	Examen	Moy CC ca	Note1	Note2	Note3	
2	**8,00**	6,00	10,00	12,00	8,00		Ajourné
3	**10,67**	8,00	13,33	18,00	8,00	14,00	Validé
4	**5,17**	6,00	4,33	5,00	4,00	4,00	Rattrapage
5	**11,29**	14,75	7,83	10,00	0,00	13,50	Validé
6	**13,08**	12,00	14,17	17,75	8,00	16,75	Validé

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
```

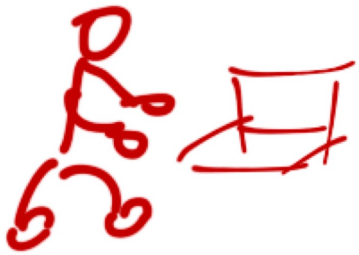



```
Set cellules = ActiveSheet.Range("D2:D1000")
nbLig = WorksheetFunction.CountA(cellules)
compteur = 1

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

Fonction NBVAL

***Cells** : première ligne / colonne est toujours 1*

Exercice

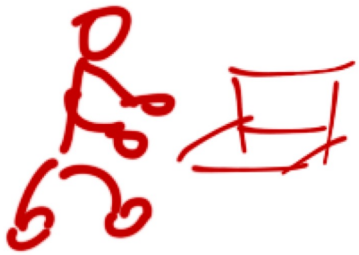


- **Exercice**
  - Toujours sur le fichier « **FichierNotes.xlsx** » :
  - Créer une fonction « **MoyenneGroupe** » qui calcule la moyenne d'un Range
    - **Function MoyenneGroupe (groupe as Range) as Single**
  - Ajouter sur la feuille « feuil1 » le calcule de la moyenne finale des groupes G1 et G2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

groupe.Count → permet de savoir combien des lignes on a dans le **Range**

groupe.Cells(ligne, colonne).Value → permet de récupérer la valeur contenue dans la **cellule** position (**ligne**,**colonne**) contenue dans le **Range**

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
```

Single *= numérique de précision réduite*

	A	B	C	D
1	**Groupe**	**Nom**	**Prénom**	**Moy Calc**
2	G1	AAAA	aaaaa	13,08
3	G1	BBBB	bbbb	14,29
4	G1	CCCC	cccc	14,67
5	G1	DDDD	ddddd	13,29
6	G1	EEEE	eeee	11,29
7	G1	FFFF	fffff	6,17
8	G1	GGGG	ggggg	13,83
9	G1	HHHHH	hhhh	8,00
10	G1	KKKKKK	kkkkkk	10,67

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

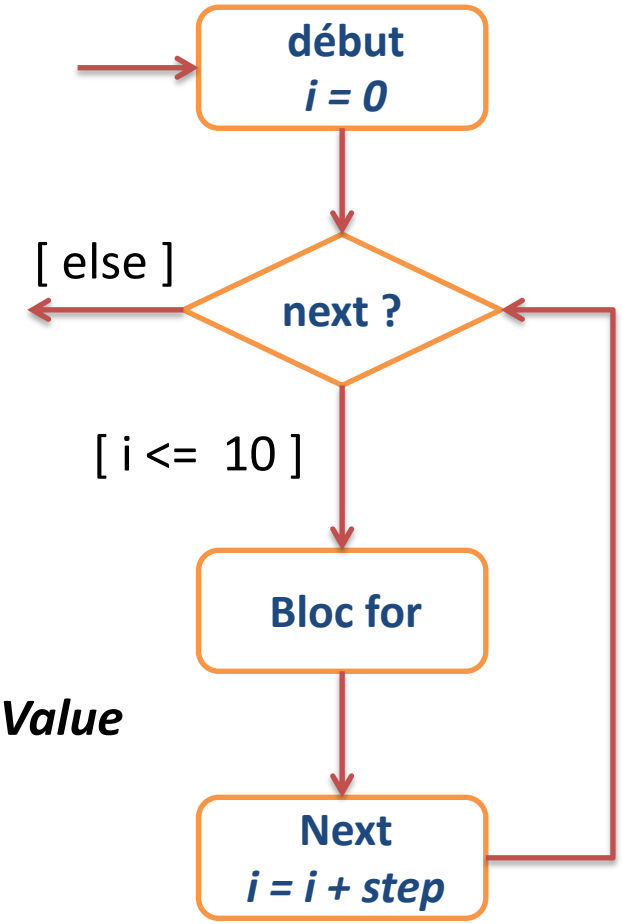
	C	D	E
	Moyenne G1	*11,699074*	
	Moyenne G2	*14,074651*	

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : Instructions de contrôle

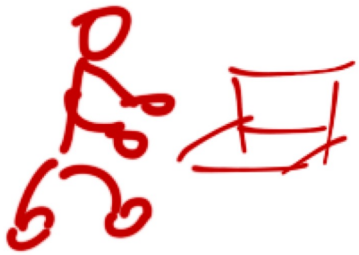
- Boucles : **For…Next**

```
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
```

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)

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

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
```

*La variable **lin** joue le rôle de **compteur***

Prénom	Moy Calc	Examen
aaaaa	13,08	12,00
bbbb	14,29	14,00
cccc	14,67	16,75
ddddd	13,29	16,75
eeee	11,29	14,75
fffff	6,17	8,00
ggggg	13,83	13,75
hhhh	8,00	6,00
kkkkkk	10,67	8,00

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

	C	D	E
	Moyenne G1	*11,699074*	12,2222223
	Moyenne G2	*14,074651*	15,229167

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : Instructions de contrôle

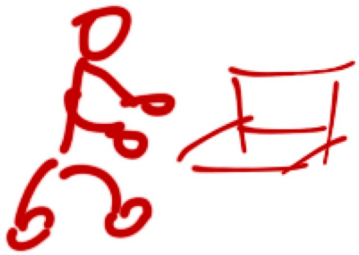
- Boucles : **For each … Next**
  - On peut aussi parcourir toute une collection

```
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

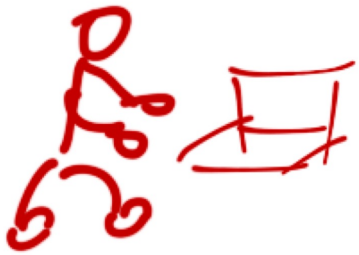
Exercice



- **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
	aaaaa	13,08	12,00	14,17
	bbbb	14,29	14,00	14,58
	cccc	14,67	16,75	12,58
	ddddd	13,29	16,75	9,83
	eeee	11,29	14,75	7,83
	fffff	6,17	8,00	4,33
	ggggg	13,83	13,75	13,92
	hhhh	8,00	6,00	10,00
	kkkkkk	10,67	8,00	13,33

=MoyenneGroupe3('G1'!F2:F10)

	C	D	E	F
	Moyenne G1	11,699074	12,2222223	11,1759253
	Moyenne G2	14,074651	15,229167	12,9201393

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

VBA : Tableaux

- Les tableaux ne sont pas des ranges comme les autres…
- Collection → **ListObjects**
- Un tableau → **ListObject**

ListObjects("TabVentes") → *tableau*

	A	B	C	D	E	F	G	H	I
5	N° Facture	Date	Produit	Description	Prix Unit	Quantité	Prix HT	Prix TTC	Prix Prod
6	FA-2008-0001	02/07/2024	P9287	Table	160,00 €	1	160,00 €	191,84 €	160,00 €
7	FA-2008-0002	08/07/2024	P6953	Ensemble jardin	280,00 €	1	280,00 €	335,72 €	280,00 €
8	FA-2008-0002	08/07/2024	P3026	Porte tablette	22,00 €	3	66,00 €	79,13 €	22,00 €
9	FA-2008-0003	21/06/2024	P5875	Table	88,00 €	2	174,78 €	209,56 €	87,39 €
10	FA-2008-0004	22/08/2024	P1176	Dessous verre	12,00 €	2	24,00 €	28,78 €	12,00 €
11	FA-2008-0004	22/08/2024	P7910	Ensemble table	300,00 €	1	300,00 €	359,70 €	300,00 €

ListRows
(toutes les lignes de données du tableau)

ListRows.Add
(ajoute une nouvelle ligne à la fin du tableau)

ListRow
(une ligne du tableau)

ListRow.Range(5)
(une cellule du tableau)

Exercice

- Sur le fichier « **Commandes.xlsx** » :
  - Modifier notre macro associée au bouton « **valider** »
  - Pour chaque produit indiqué au formulaire, on ajoute une ligne au tableau « **TabVentes** »

On trouve la dernière ligne (c.a.d. le dernier produit)

```
dernierProd = Worksheets("Formulaire").Range("A:A").Cells(Rows.Count, 1).End(xlUp).Row

Dim nouvelleLigne As ListRow

For lin = 13 To dernierProd Step 1

   Set nouvelleLigne = Worksheets("Ventes").ListObjects("TabVentes").ListRows.Add

   With nouvelleLigne
            .Range(1) = Worksheets("Formulaire").Range("B4").Value      'n° facture
            .Range(3) = Worksheets("Formulaire").Range("A" & lin).Value 'prod
            .Range(4) = Worksheets("Formulaire").Range("B" & lin).Value ' desc
            .Range(5) = Worksheets("Formulaire").Range("C" & lin).Value ' prix
            .Range(6) = Worksheets("Formulaire").Range("D" & lin).Value ' qte
   End With

Next
```

De la ligne 13 (1er produit) à la dernière ligne du formulaire

On ajoute une nouvelle ligne

On copie les informations sur le produit commandé

Exercice

- Code à ajouter à l'ancien

```
Dim nouvelleLigne As ListRow

'on récupère la dernière ligne contenant des produits dans le formulaire
dernierProd = Worksheets("Formulaire").Range("A:A").Cells(Rows.Count, 1).End(xlUp).Row

'on sait que le premier produit est dans la ligne 13 du formulaire
'donc pour chaque ligne de la ligne 13 jusqu'à la dernière ligne,
'on ajoute une ligne au tableau ventes avec le produit concerné

For lin = 13 To dernierProd Step 1
   Set nouvelleLigne = Worksheets("Ventes").ListObjects("TabVentes").ListRows.Add 'on ajoute une nouvelle ligne vide

   With nouvelleLigne
     'on remplit la ligne avec les valeurs, les formules sont déjà recopiées avec le Add
      .Range(1) = Worksheets("Formulaire").Range("B4").Value  'n° facture
      .Range(3) = Worksheets("Formulaire").Range("A" & lin).Value 'prod
      .Range(4) = Worksheets("Formulaire").Range("B" & lin).Value ' desc
      .Range(5) = Worksheets("Formulaire").Range("C" & lin).Value ' prix
      .Range(6) = Worksheets("Formulaire").Range("D" & lin).Value ' qte
   End With

Next

MsgBox "Nouvelle facture " & Worksheets("Formulaire").Range("B4").Value & " ajoutée avec succès", vbInformation
```

VBA : Instructions de contrôle

- Boucles :
  - On peut imbriquer plusieurs boucles…

Lbound (tableau, **dimension**)
Ubound (tableau, **dimension**)

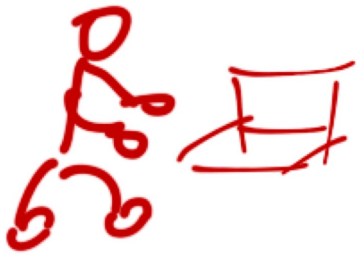
```
For i = Lbound(tabBiDim,1) To Ubound(tabBiDim,1)
    For j = LBound(tabBiDim, 2) To UBound(tabBiDim, 2)
        somme = somme + tabBiDim ( i , j )
    Next
Next
```

tabBiDim

i \ j	1	2	3
1	0	0	**2.5**
2	**1.5**	**4.5**	0

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

Exemple

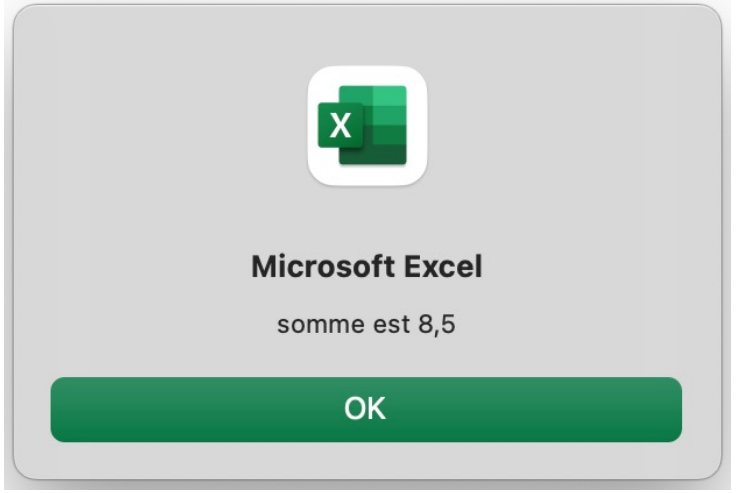


(Général) | ExempleBoucleImb

```
Sub ExempleBoucleImb()
    Dim somme As Single
    Dim tabBiDim(1 To 2, 1 To 3) As Currency

    tabBiDim(1, 3) = 2.5
    tabBiDim(2, 1) = 1.5
    tabBiDim(2, 2) = 4.5

    somme = 0

    For i = LBound(tabBiDim, 1) To UBound(tabBiDim, 1)
        For j = LBound(tabBiDim, 2) To UBound(tabBiDim, 2)
            somme = somme + tabBiDim(i, j)

            MsgBox tabBiDim(i, j) & " :  " & somme
    Next
Next

MsgBox "somme est " & somme

End Sub
```

tabBiDim

i \ j	1	2	3
1	0	0	**2.5**
2	**1.5**	**4.5**	0



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** »
  - Programmer ce bouton pour qu'il puisse **copier** les informations des **colonnes B, C et D** des feuilles « **G1** » et « **G2** » sur 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 »

Exercices

D'abord, on copie les données de la feuille G1 vers la nouvelle feuille

	A	B	C	D
1	**Nom**	**Prénom**	**Note**	
2	AAAA	aaaaa	13,0833333	
3	BBBB	bbbb	14,2916667	
4	CCCC	cccc	14,6666667	
5	DDDD	ddddd	13,2916667	
6	EEEE	eeee	11,2916667	
7	FFFF	fffff	6,16666667	
8	GGGG	ggggg	13,8333333	
9	HHHHH	hhhh	8	
10	KKKKKK	kkkkkk	10,6666667	
11	HIHHI	bbbb	14,875	
12	JJJJJ	aaaaa	16,375	
13	LLLLLL	cccc	13,3333333	
14	MMMM	ddddd	12,3333333	
15	NNNNN	eeee	15,1666667	
16	OOOOO	fffff	15,2916667	
17	PPPPP	ggggg	16,0416667	
18	QQQQ	hhhh	13,0416667	
19	RRRRR	kkkkkk	12,875	
20	SSSSS	sssss	12,6666667	
21	TTTTT	tttt	13,2083333	
22	UUUUUU	uuuuu	13,6875	
23				
24				

Données G1

On passe alors à la prochaine ligne

Données G2

Puis, on copie les données de la feuille G2 vers la nouvelle feuille

B2 : AAAA

	A	B	C	D
1	**Groupe**	**Nom**	**Prénom**	**Moy Calc**
2	G1	AAAA	aaaaa	13,08
3	G1	BBBB	bbbb	14,29
4	G1	CCCC	cccc	14,67
5	G1	DDDD	ddddd	13,29
6	G1	EEEE	eeee	11,29
7	G1	FFFF	fffff	6,17
8	G1	GGGG	ggggg	13,83
9	G1	HHHHH	hhhh	8,00
10	G1	KKKKKK	kkkkkk	10,67

G1 | G2 | Feuil1

Combien de lignes copier ?

Où coller les données ?

B2 : HIHHI

	A	B	C	D
1	**Groupe**	**Nom**	**Prénom**	**Moy Calc**
2	G2	HIHHI	bbbb	14,88
		JJJJJ	aaaaa	16,38
		LLLLLL	cccc	13,33
		MMMM	ddddd	12,33
		NNNNN	eeee	15,17
7	G2	OOOOO	fffff	15,29
8	G2	PPPPP	ggggg	16,04
9	G2	QQQQ	hhhh	13,04
10	G2	RRRRR	kkkkkk	12,88
11	G2	SSSSS	sssss	12,6666667
12	G2	TTTTT	tttt	13,2083333
13	G2	UUUUUU	uuuuu	13,6875
14				

G1 | G2 | Feuil1

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** »

BoutonFinaliser_Cliquer

```
ser_Cliquer()
Dim lignesG1, lignesG2, ligneDst As Integer

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

MsgBox "G1 : " & lignesG1 & " G2 : " & lignesG2

End Sub
```

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 trouvé avant.*

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***, ***xlPasteValuesAndNumberFormats***…)

```
Worksheets("NotesFinales").Range("A2").PasteSpecial  xlPasteValues
```

```
lignesG1 = WorksheetFunction.CountA(Worksheets("G1").Range("B:B"))
lignesG2 = WorksheetFunction.CountA(Worksheets("G2").Range("B:B"))

'MsgBox "G1 : " & lignesG1 & " G2 : " & lignesG2

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

	A	B	C
1	Nom	Prénom	Note
2	AAAA	aaaaa	13,0833333
3	BBBB	bbbb	14,2916667
4	CCCC	cccc	14,6666667
5	DDDD	ddddd	13,2916667
6	EEEE	eeee	11,2916667
7	FFFF	fffff	6,16666667
8	GGGG	ggggg	13,8333333
9	HHHHH	hhhh	8
10	KKKKKK	kkkkkk	10,6666667
11			

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** »*

Données de G1

	A	B	C
1	**Nom**	**Prénom**	**Note**
2	AAAA	aaaaa	13,0833333
3	BBBB	bbbb	14,2916667
4	CCCC	cccc	14,6666667
5	DDDD	ddddd	13,2916667
6	EEEE	eeee	11,2916667
7	FFFF	fffff	6,16666667
8	GGGG	ggggg	13,8333333
9	HHHHH	hhhh	8
10	KKKKKK	kkkkkk	10,6666667
11			

6ème étape : On copie les données de G2

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

*Il faut qu'on colle les données de G2 à partir de la ligne **11**…*

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

HIHHI	bbbb	14,88
JJJJJ	aaaaa	16,38
LLLLLL	cccc	13,33
MMMM	ddddd	12,33
NNNNN	eeee	15,17
OOOOO	fffff	15,29
PPPPP	ggggg	16,04
QQQQ	hhhh	13,04
RRRRR	kkkkkk	12,88
SSSSS	sssss	12,6666667
TTTTT	tttt	13,2083333
UUUUUU	uuuuu	13,6875

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 xlPasteValues
```

(Général) | BoutonFinaliser_Cliquer

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

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

  'MsgBox "G1 : " & lignesG1 & " G2 : " & lignesG2

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

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

  ligneDst = lignesG1 + 1
  Worksheets("NotesFinales").Range("A" & ligneDst).PasteSpecial xlPasteValues

End Sub
```

	A	B	C
1	**Nom**	**Prénom**	**Note**
2	AAAA	aaaaa	13,0833333
3	BBBB	bbbb	14,2916667
4	CCCC	cccc	14,6666667
5	DDDD	ddddd	13,2916667
6	EEEE	eeee	11,2916667
7	FFFF	fffff	6,16666667
8	GGGG	ggggg	13,8333333
9	HHHHH	hhhh	8
10	KKKKKK	kkkkkk	10,6666667
11	HIHHI	bbbb	14,875
12	JJJJJ	aaaaa	16,375
13	LLLLLL	cccc	13,3333333
14	MMMM	ddddd	12,3333333
15	NNNNN	eeee	15,1666667
16	OOOOO	fffff	15,2916667
17	PPPPP	ggggg	16,0416667
18	QQQQ	hhhh	13,0416667
19	RRRRR	kkkkkk	12,875
20	SSSSS	sssss	12,6666667
21	TTTTT	tttt	13,2083333
22	UUUUUU	uuuuu	13,6875
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 ».