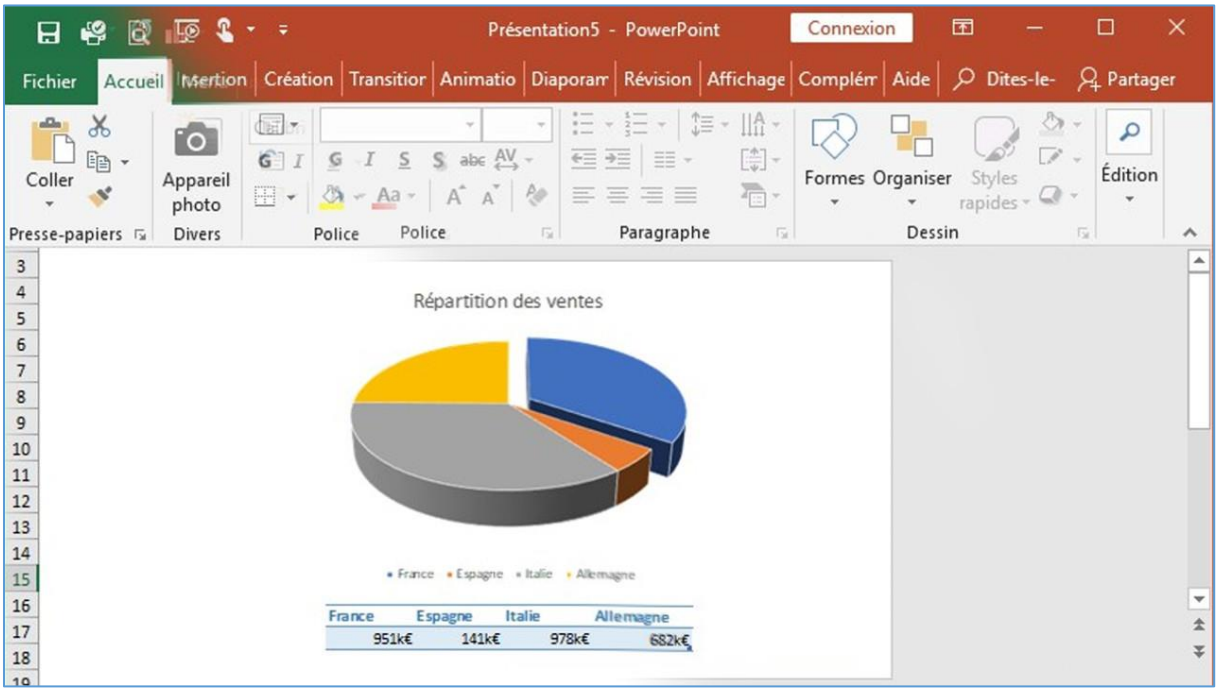
# Comment transformer vos fichiers Excel en présentation PowerPoint en 1 clic ?

*Je vous propose, aujourd'hui, un article invité d'Alexandre d'Excel Formation.*




Dans cet article, nous allons voir comment, **d'un seul clic**, il est possible de **transformer une feuille de calcul Excel** en une **présentation PowerPoint**, de manière **instantanée**.

Pour ce faire, nous allons **décomposer** notre feuille de calcul en **plusieurs slides**, en utilisant les **zones d'impressions**. Mais nous aurions également pu utiliser d'autres outils (comme par exemple transformer chaque feuille de calculs en autant de slides).

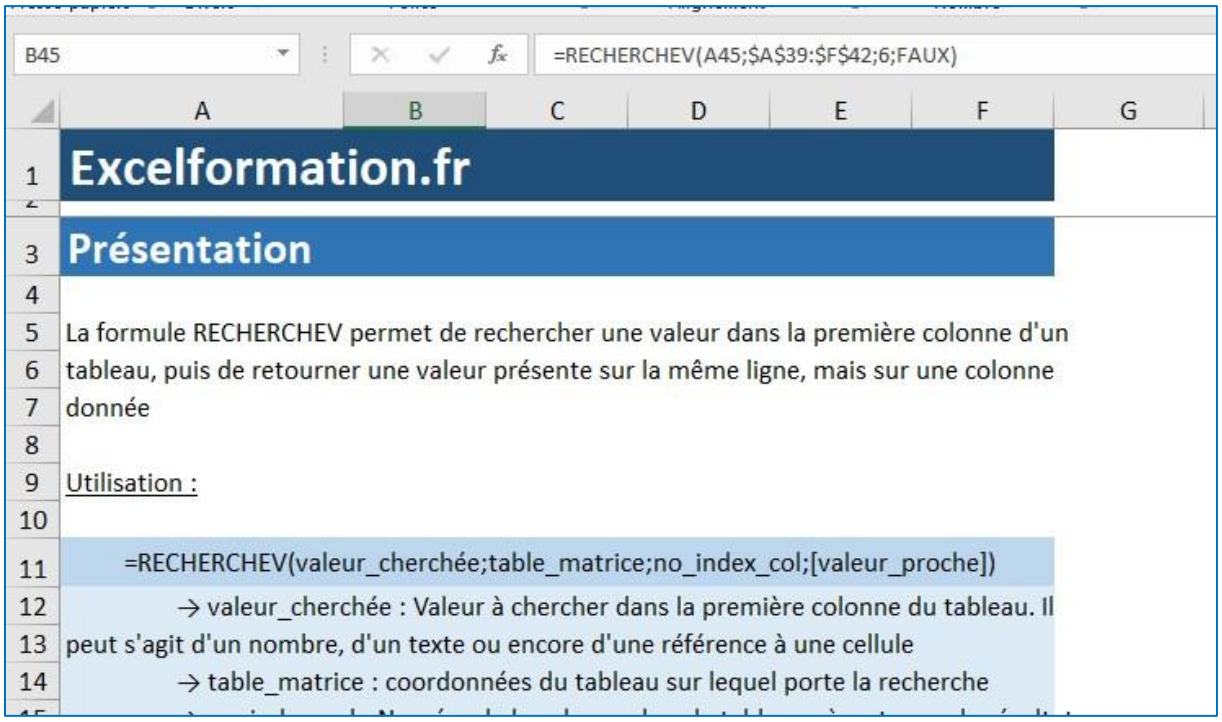
Cet article va être découpée en **2 parties**, plus une 3e partie bonus que je vous laisse découvrir en fin de texte :

1. Dans la première partie, nous allons voir comment **extraire les coordonnées de chacune des zones d'impression** de la feuille ;
2. Puis, dans la seconde partie, nous verrons comment **exporter le contenu de chacune de ces zones** dans une présentation PowerPoint.

Le lien pour télécharger le fichier de macros complémentaires contenant l'ensemble de la procédure est disponible à la fin du tutoriel.

**Décomposons notre feuille de calculs en slides :**

Pour illustrer cet article, nous allons utiliser un classeur d'exemple que nous avions mis en place pour un article précédent ayant pour thème la formule RECHERCHEV(). Celui-ci se prête bien à l'exercice, car il est constitué de nombreuses lignes et sera donc scindé en plusieurs slides. Mais, bien entendu, libre à vous d'utiliser votre propre classeur Excel.

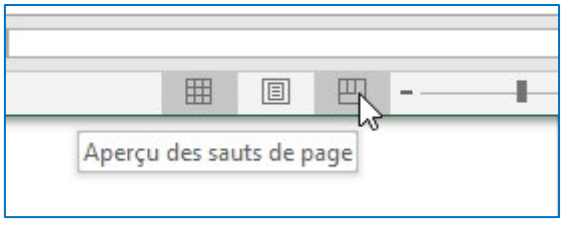



Voyons à présent comment récupérer les coordonnées de nos zones d'impression.

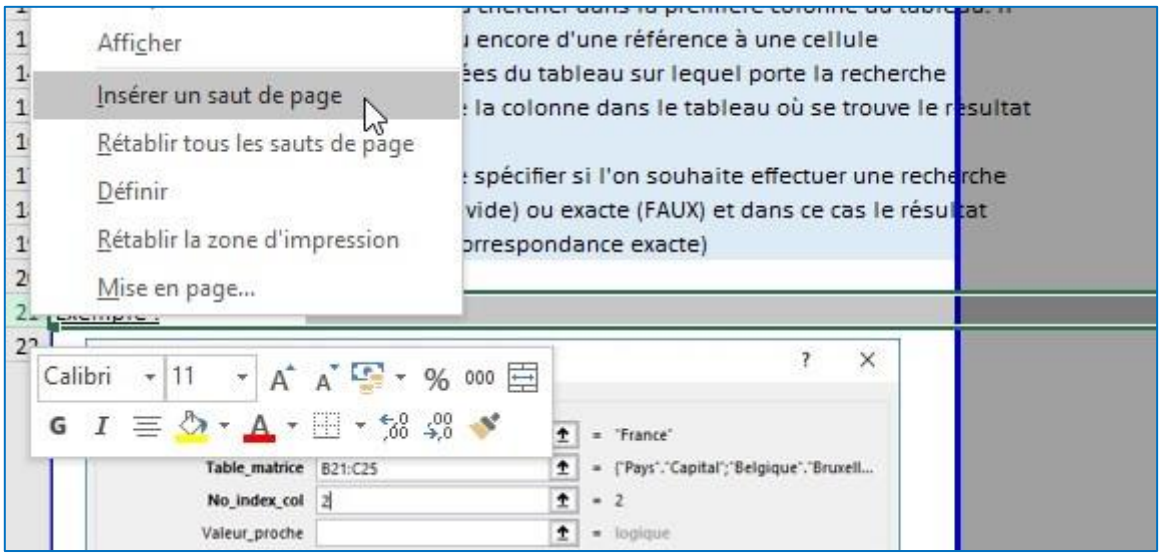
Si l'on effectue un **aperçu avant impression** de la feuille de calculs, on constate que toutes les pages vont être imprimées, mais qu'aucune règle n'est clairement définie. Les passages aux pages suivantes se font automatiquement en bas de feuille.

Nous allons commercer par **insérer des sauts de page** pour forcer l'édition à passer sur une nouvelle page lorsque nous le choisissons :

- Pour cela, nous allons passer l'affichage en mode *Afficher les sauts de page*, en cliquant sur le bouton correspondant en bas de la fenêtre :



- Puis, nous insérons **manuellement tous les sauts de page**, en gardant à l'esprit que ces derniers vont nous permettre de **délimiter chacun de nos slides** :
  - Nous commençons par **sélectionner la ligne** sur laquelle le saut de page devra être inséré ;
  - Puis, nous effectuons un clic droit ;
  - Enfin, nous choisissons **Insérer un saut de page** ;
  - Cette opération est à répéter pour tous les sauts de page :

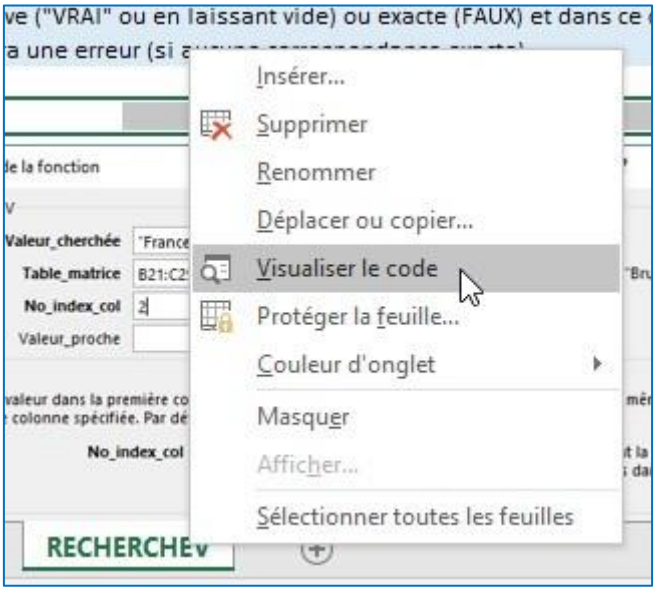



Cela étant fait, nous pouvons repasser en mode *Normal* et vérifier le résultat avec un nouvel *Aperçu avant impression*.

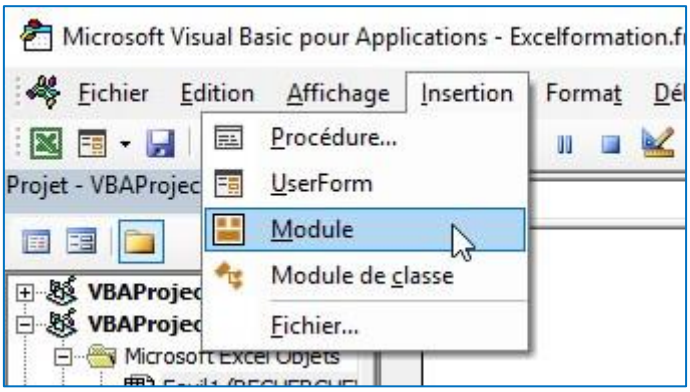
À présent, nous allons coder une nouvelle **macro-commande en VBA** qui va nous permettre de récupérer les coordonnées de chacune des zones d'impression. Pour cela :

- Effectuons un clic droit sur l'onglet de la feuille de calcul ;

- Pour choisir **Visualiser le code** :




- Nous arrivons alors dans **l'outil de développement d'Excel** (VBA) ;
- Nous allons insérer un **nouveau module** (menu **Insertion**, puis *Module*) :




- Dans ce module, nous allons créer une **nouvelle procédure** que nous allons appeler *sub exporterVersPowerpoint*, qui va nous permettre de générer notre export :

```
Sub exporterVersPowerpoint()
    'Partie 1 : Récupérer l'adresse des pages d'impressions
    'Partie 2 : Exporter chacune de ces zones vers une présentation PowerPoint
End Sub
```

Comme vous pouvez le constater, notre code sera décomposé en 2 étapes :

- Dans la première partie, nous allons identifier et stocker dans une variable les coordonnées de chacune des zones d'impression que nous avons définie précédemment ;

- Dans une seconde partie, nous allons "exporter" ces plages de cellules dans des diapos PowerPoint.

Pour stocker les adresses des zones d'impressions, nous allons instancier une nouvelle variable de type *String* (c'est-à-dire une **chaîne de caractères**). Chaque adresse sera séparée au sein de cette chaîne par un tiret ("-") :

```
'Partie 1 : Récupérer l'adresse des pages d'impressions
Dim plages as String: plages = ""
```

Cette variable "plages" est pour le moment vide. Ensuite, pour récupérer les adresses des zones d'impressions, nous allons utiliser l'objet *HPageBreaks* qui reprend chacun des **sauts de pages** que nous avons définis au tout début de cette partie. Pour ne pas avoir à le ressaisir à chaque fois, nous allons utiliser l'opérateur *with* :

```
With ActiveSheet.HPageBreaks
End With
```

De cette manière, dès lors qu'une propriété située entre ces 2 balises va commencer par un point ("."), alors il s'agira d'une **sous-propriété** de *HPageBreaks*.

Les 2 expressions sont ainsi équivalentes :

```
'Expression 1 : Avec l'opérateur with
With ActiveSheet.HPageBreaks
        nombreSautDePage = .Count
End With
'Expression 2 : Sans l'opérateur with
nombreSautDePage = ActiveSheet.HPageBreaks.Count
```

Pour commencer, nous allons vouloir savoir **combien de sauts de page** ont été insérés. De cette manière, si **aucun saut de page** n'est identifié, alors cela signifie que toute **notre feuille devra être insérée dans un seul slide**. Pour cela, nous allons utiliser l'instruction *Count* de *HPageBreaks* (en commençant par un point).

```
If .Count = 0 Then
Else
End If
```

Ainsi, si *.Count* est égal à zéro, alors notre variable "plages", instanciée juste avant, est égale aux **coordonnées de l'ensemble des cellules** utilisées dans notre feuille, que nous allons récupérer avec l'instruction *UsedRange* :

```
If .Count = 0 Then
    plages = ActiveSheet.UsedRange.Address
```

Par contre, si **au moins un saut de page** est identifié par Excel, alors nous allons vouloir les **coordonnées de chacun d'entre eux**. L'objet *HPageBreaks* est en fait constitué d'autant de sous-objets qu'il y a de sauts de page dans la feuille de calculs. Chacun de ses sous-objets porte le nom d'*Item*.

Nous allons les passer en revue l'un après l'autre, en utilisant une boucle for :

```
Else
    For i = 1 To .Count
    Next
```

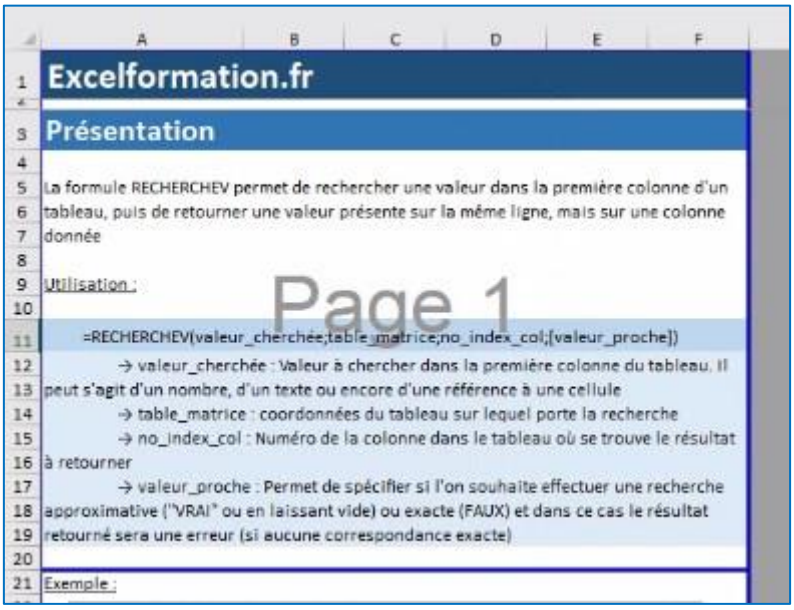
La variable *i* que nous utilisons pour notre boucle va ainsi prendre une valeur qui va **s'incrémenter lors de chaque nouvelle boucle** (1, puis 2, puis 3 et ainsi de suite). Nous pourrons donc récupérer le numéro de la ligne sur la laquelle est placé le saut de page numéro *i* grâce à la propriété *.Item(i).Location.Row*

```
For i = 1 To .Count
    ligneSaut = .Item(i).Location.Row
    MsgBox ligneSaut
Next
```

Pour tester qu'Excel nous retourne bien le bon numéro de ligne, nous utilisons la fonction **msgbox** qui permet d'afficher un message.

De retour dans notre feuille de calcul, si **nous exécutons la commande** (en appuyant sur les touches *Alt* et *F8* en même temps), le premier résultat retourné par Excel est la ligne *21* :


Le premier slide aura donc pour coordonnées les lignes *1* à *20* (et non *21*, car la ligne 21 est déjà sur le second slide) :




Pour connaître les **coordonnées des colonnes**, nous allons utiliser à nouveau une des propriétés de *UsedRange* que nous avons vue précédemment, afin de compter le nombre de colonnes :

*Dim derniereColonne As String*

*dernieColonne = ActiveSheet.UsedRange.Columns.Count*

Nous pouvons maintenant déterminer l'adresse des cellules qui vont constituer le premier slide :

```
For i = 1 To .Count
        ligneSaut = .Item(i).Location.Row
        Dim derniereColonne As Integer
        dernieреColonne = ActiveSheet.UsedRange.Columns.Count
        plages = plages & Range(ActiveSheet.Cells(1, 1), ActiveSheet.Cells(ligneSaut - 1, dernieреColonne)).Address & "-"
        MsgBox plages
Next
```

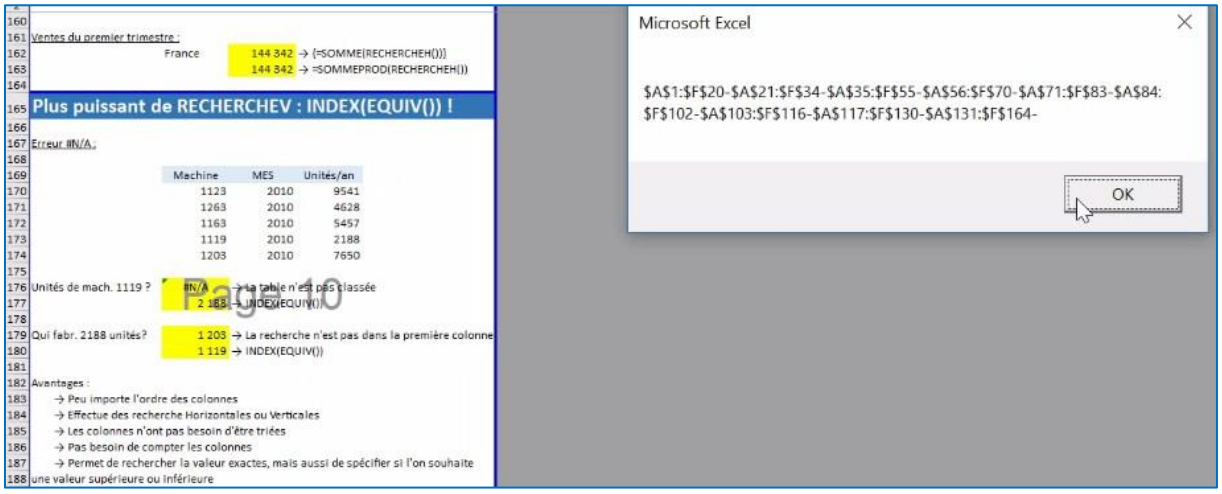
Vous noterez la présence du **tiret** en fin de chaîne pour séparer chacune des adresses. Si vous testez la macro-commande à ce stade de son écriture, vous noterez que celle-ci **commence toujours sur la première ligne de notre feuille** ($A$1) :


Pour régler ce problème, nous allons introduire une nouvelle variable que nous allons appeler *debut* avec pour valeur *1* (pour la première ligne), puis à la fin de chaque cycle de la boucle, celle-ci prendra la valeur de *ligneSaut*.

Enfin, nous utiliserons cette variable dans l'adresse de notre plage :

```
Else
        Dim debut As Integer
        debut = 1
        For i = 1 To .Count
            [...]
            plages = plages & Range(ActiveSheet.Cells(debut, 1), ActiveSheet.Cells(ligneSaut - 1, derniereColonne)).Address & "-"
            debut = ligneSaut
        Next
End If
```

**Attention :** si vous regardez attentivement les coordonnées retournées par la variable plages, vous constaterez que celle-ci **s'arrête au dernier saut de page** (ligne *164* dans notre exemple), et donc la dernière partie de la feuille de calcul n'est pas prise en compte (lignes *165* à *188*) !




Il faut donc ajouter une dernière ligne à la suite de la boucle :

```
ligneFin = ActiveSheet.UsedRange.Rows.Count
plages = plages & Range(ActiveSheet.Cells(debut, 1), ActiveSheet.Cells(ligneFin, dernieColonne)).Address & "-"
```

La dernière opération de cette première partie va consister à **exclure le dernier caractère** de la variable "plages" (qui sera toujours un tiret).

Pour cela, nous allons utiliser l'instruction *Left()*, en spécifiant que nous désirons récupérer la partie gauche de la variable "plages" pour le nombre de caractères contenus dans celle-ci, **moins un caractère** :

```
'Partie 1 : Récupérer l'adresse des pages d'impressions
Dim plages As String: plages = ""
With ActiveSheet.HPageBreaks
    If .Count = 0 Then
        plages = ActiveSheet.UsedRange.Address
    Else
        Dim debut As Integer
        debut = 1
        For i = 1 To .Count
            ligneSaut = .Item(i).Location.Row
            Dim derniereColonne As Integer
            derniereColonne = ActiveSheet.UsedRange.Columns.Count
            plages = plages & Range(ActiveSheet.Cells(debut, 1),
        ActiveSheet.Cells(ligneSaut - 1, derniereColonne)).Address & "-"
            debut = ligneSaut
        Next
        ligneFin = ActiveSheet.UsedRange.Rows.Count
        plages = plages & Range(ActiveSheet.Cells(debut, 1),
    ActiveSheet.Cells(ligneFin, derniereColonne)).Address & "-"
        plages = Left(plages, Len(plages) - 1)
    End If
End With
```

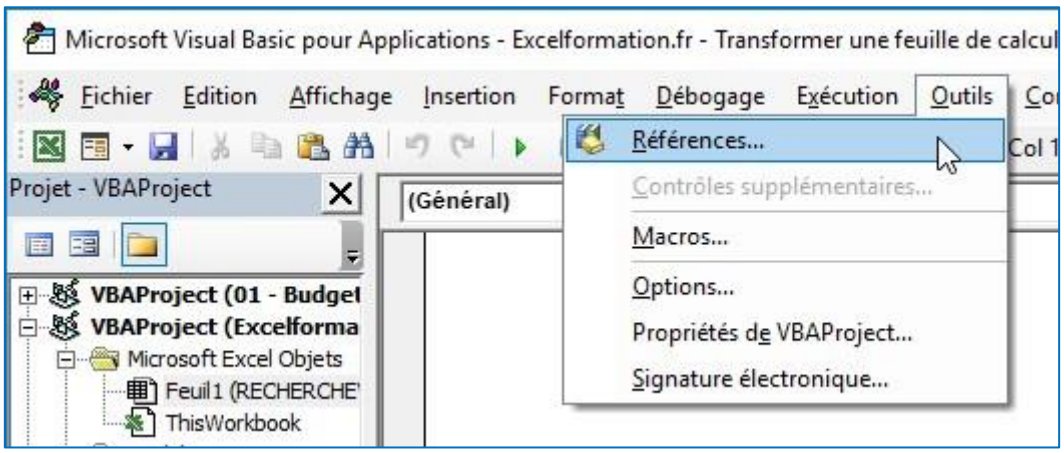
Nous disposons à présent d'une chaîne de caractères reprenant les adresses de chacune des zones d'impressions, qui sont séparées par un tiret ("-") :


Nous allons maintenant pouvoir les exporter dans PowerPoint.

## Exportons nos zones d'impressions dans PowerPoint :

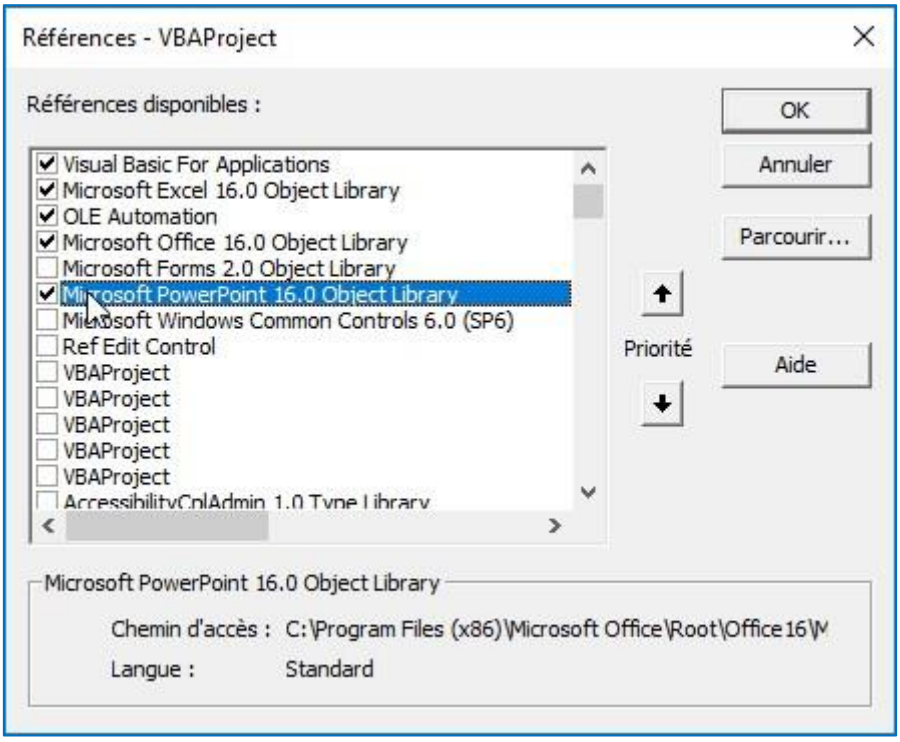
Pour utiliser une **application externe**, telle que **PowerPoint**, depuis VBA, nous allons tout d'abord devoir **activer cette dernière**.

Dans le menu **Outils**, choisissez **Références** :




Dans la fenêtre qui s'affiche, nous allons rechercher **Microsoft PowerPoint Object Library** et cocher la case correspondante. Si vous avez activé cette référence dernièrement, celle-ci devrait se trouver en **haut de la liste** (comme sur l'image ci-dessous). Dans le cas contraire, vous la trouverez beaucoup plus bas, en suivant les références **classées dans l'ordre alphabétique**.

Enfin, nous validons avec le bouton **Ok** :




À présent, nous allons pouvoir **instancier un nouvel objet** qui va nous permettre de **prendre le contrôle de PowerPoint** :

*'Partie 2 : Exporter chacune de ces zones vers une présentation powerpoint*

*Dim oPowerPoint As Object*

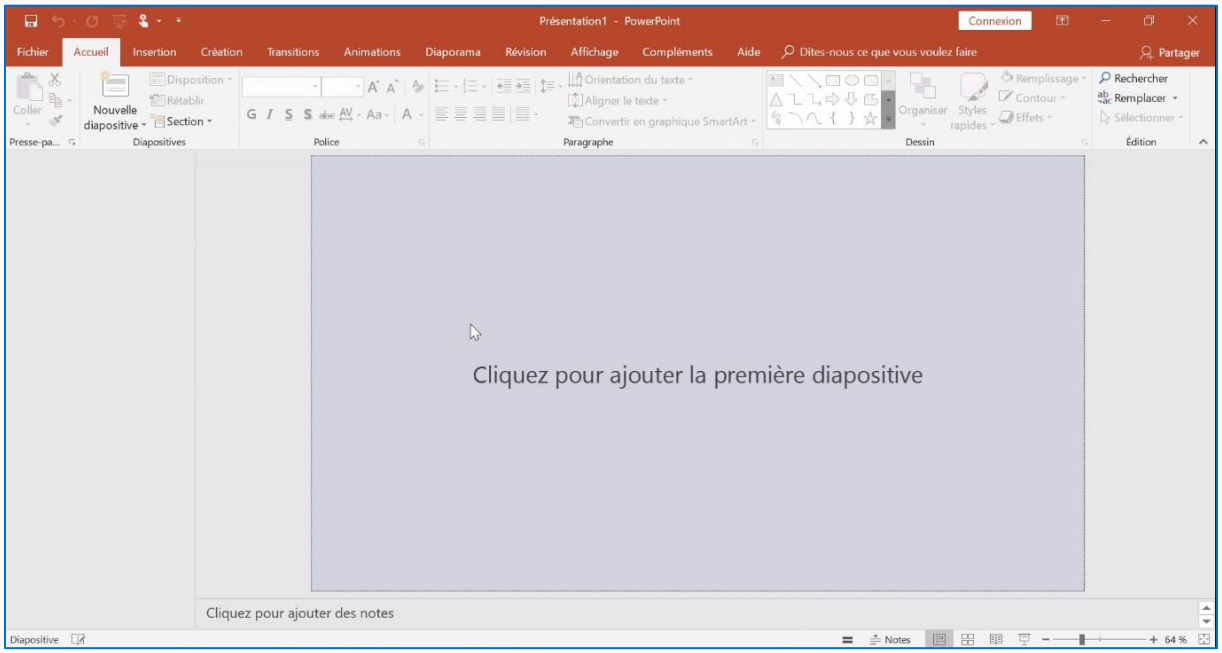
*Set oPowerPoint = CreateObject("Powerpoint.application")*

Grâce à cet objet que nous venons de créer, nous allons **pouvoir générer une nouvelle présentation** PowerPoint, qui sera un sous-objet de *oPowerPoint* :

*Dim oDiaporama As Object*

*Set oDiaporama = oPowerPoint.Presentations.Add*

Si vous testez le code à ce moment de son développement, vous constaterez qu'**Excel va demander à PowerPoint de se lancer**, puis de créer une nouvelle présentation. Évidemment, celle-ci **sera vide** :




Nous allons maintenant **ajouter les slides** qui correspondent aux cellules de chaque zone d'impressions que nous avons déterminées dans la première partie. Pour commencer, nous allons instancier une nouvelle variable qui va **reprendre l'identifiant de chaque diapositive**. Nous allons initialiser cette variable avec la valeur *1*, pour le premier slide

*idDiapo= 1*

Puis, nous allons **parcourir les différentes plages de cellules** en utilisant une boucle *for each*, combinée avec l'instruction *split()*. Cette dernière va nous permettre de **séparer les plages** en fonction de chaque tiret

*For Each plage In Split(plages, "-")*

*Next*

Lors de **chaque passage dans la boucle**, nous allons créer une **nouvelle diapositive** dans notre présentation en créant un nouvel objet. Cette diapositive aura pour index la variable *idDiapo* et sera de type *ppLayoutBlank*, c'est-à-dire une diapositive vide :

```
For Each plage In Split(plages, "-")
    Dim diapositive As Object
    Set diapositive = oDiaporama.Slides.Add(Index:=idDiapo, Layout:=ppLayoutBlank)
Next
```

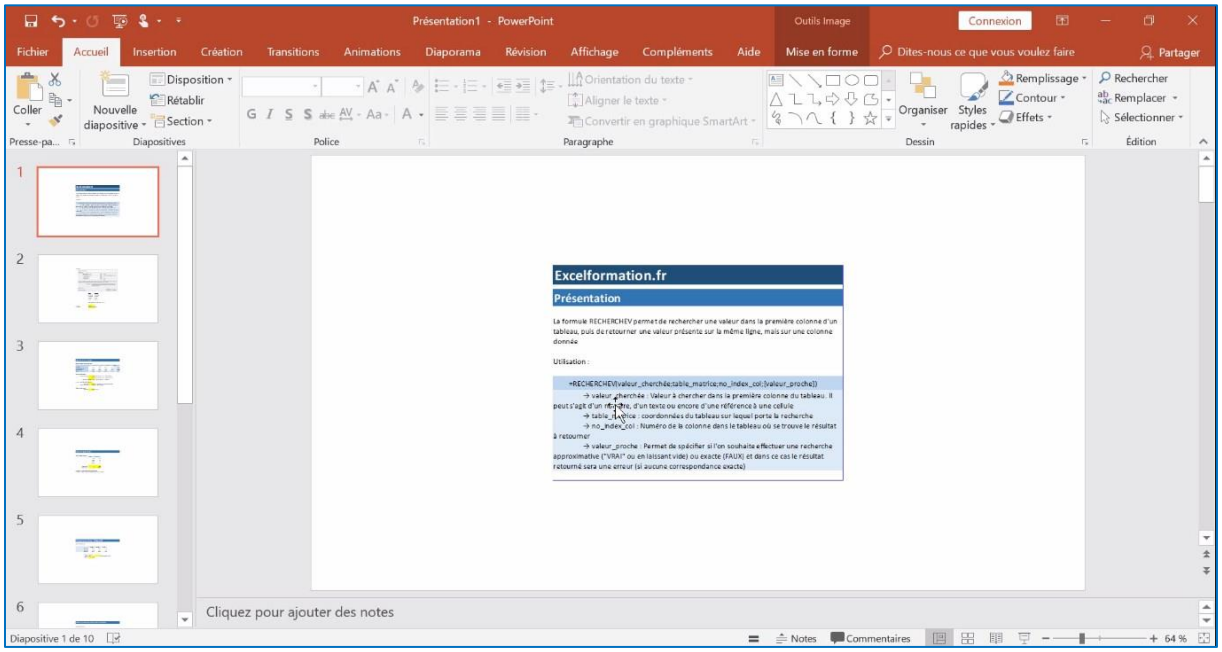
Pour exporter les données dans la présentation, nous allons effectuer un copier-coller un peu particulier. Comprenez par-là que nous allons effectuer un **collage spécial de type *ppPasteEnhancedMetafile***, c'est-à-dire sous la forme d'une image :

```
    ActiveSheet.Range(plage).Copy
    oDiaporama.Slides(idDiapo).Shapes.PasteSpecial
DataType:=ppPasteEnhancedMetafile
```

Enfin, nous pouvons passer à la diapo suivante :

```
idDiapo = idDiapo + 1
```

À présent, nous pouvons tester la macro-commande et si tout se passe bien, vous devriez vous retrouver avec une présentation PowerPoint de l'ensemble de vos zones d'impressions :



Cette présentation est pour le moment **assez rudimentaire**, mais vous pourrez l'améliorer en quelques clics seulement, par exemple en utilisant les **thèmes intégrés à PowerPoint**. Je vous invite également à suivre les **excellents tutos disponibles sur votreassistante.net** qui vont vous permettre d'améliorer grandement votre présentation.

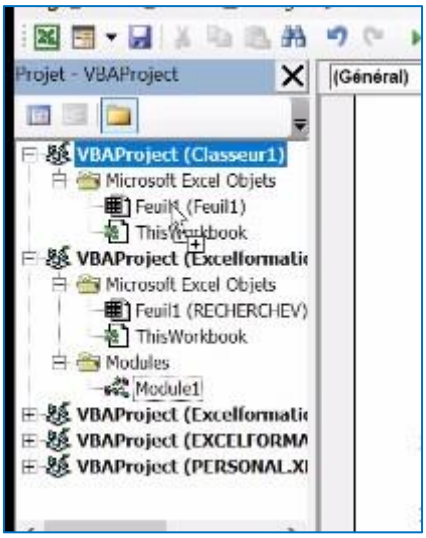
**En bonus**, pour les personnes qui ont suivi ce tuto jusqu'au bout, nous allons maintenant voir **comment créer un bouton dans la *Barre d'outils Accès rapide***, afin de pouvoir transformer n'importe quelle feuille de calcul en présentation.

## BONUS : Comment accéder à notre macro en 1 clic depuis n'importe quelle feuille de calcul ?

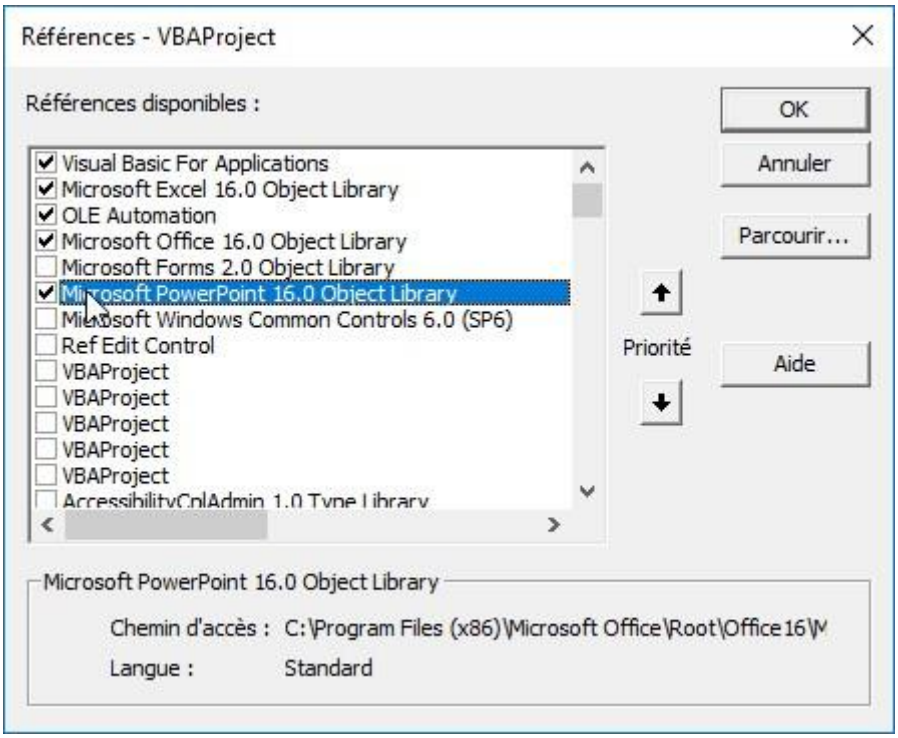
En effet, la macro-commande que nous venons de créer est **directement liée à la feuille de calcul** de test. Par conséquent, **lorsque celle-ci sera fermée, la macro-commande ne sera plus accessible**.

Pour commencer, nous allons **créer un nouveau classeur** (en appuyant sur les touches *Ctrl + N*).

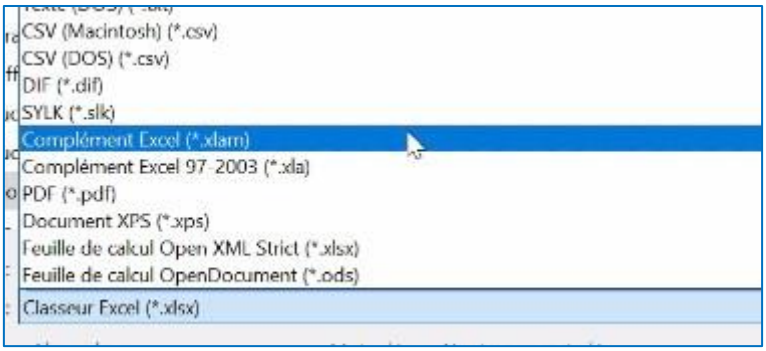
Puis, depuis l'outil de développement VBA, nous **allons effectuer un glisser-déposer** du module que nous venons créer, **afin de le dupliquer**. Pour cela nous cliquons dessus, et nous faisons glisser la souris jusqu'à notre nouveau classeur :



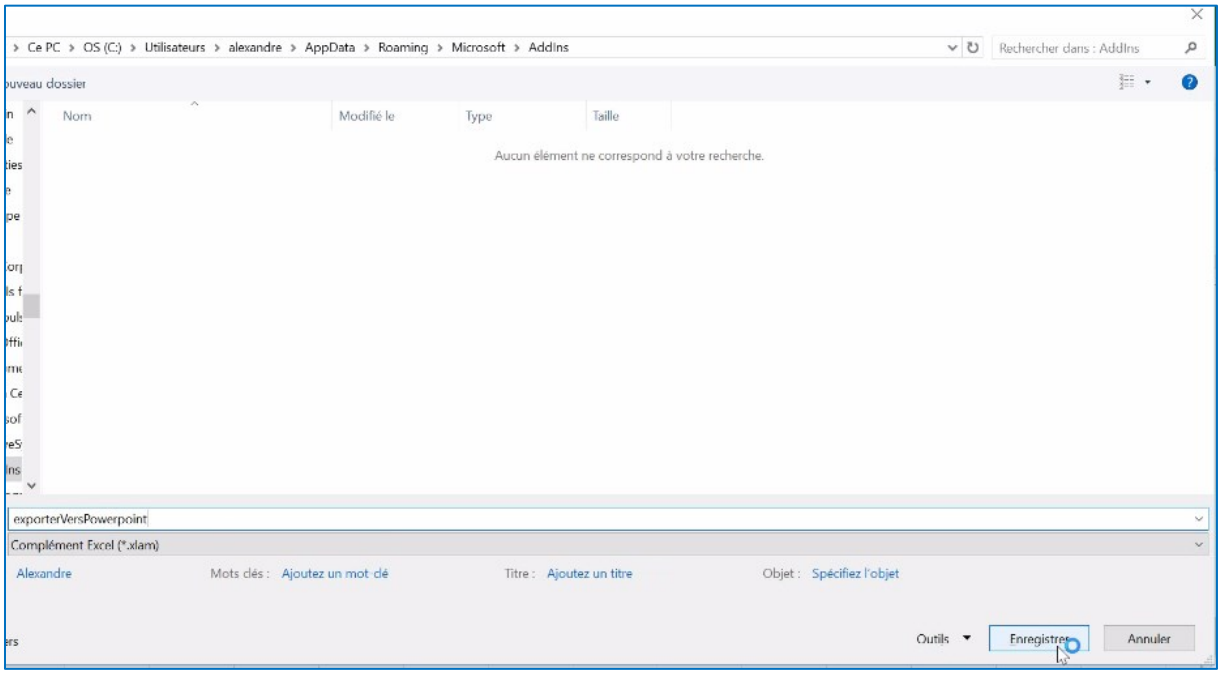
Pour que la macro fonctionne, il faut **activer la référence** *Microsoft PowerPoint Object Library* sur ce nouveau classeur comme nous l'avons vu juste au-dessus :




Ensuite, nous revenons sur le classeur que nous allons enregistrer en effectuant un **Enregistrer sous** (touche *F12* du clavier). Dans le type de classeur, nous allons choisir **Complément Excel (*.xlam)** :



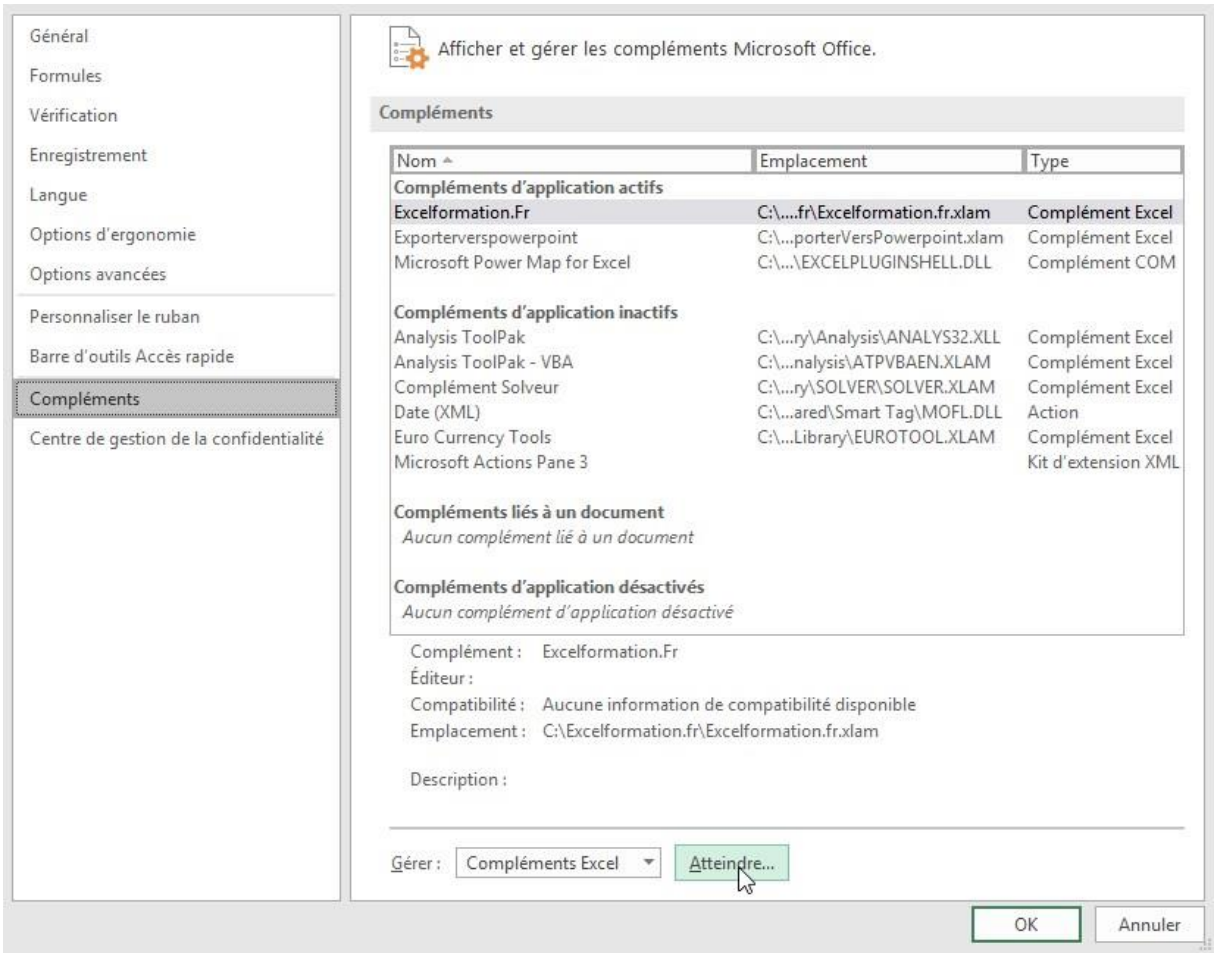
Automatiquement, Excel ouvre le **répertoire d'enregistrement des macros complémentaires** (*AddIns*), nous allons alors donner le nom *exporterVersPowerpoint.xlam* à ce classeur et appuyer sur la touche **Enregistrer** pour valider :




En réalité, Excel enregistre une **copie du fichier**, l'original restant toujours intact.

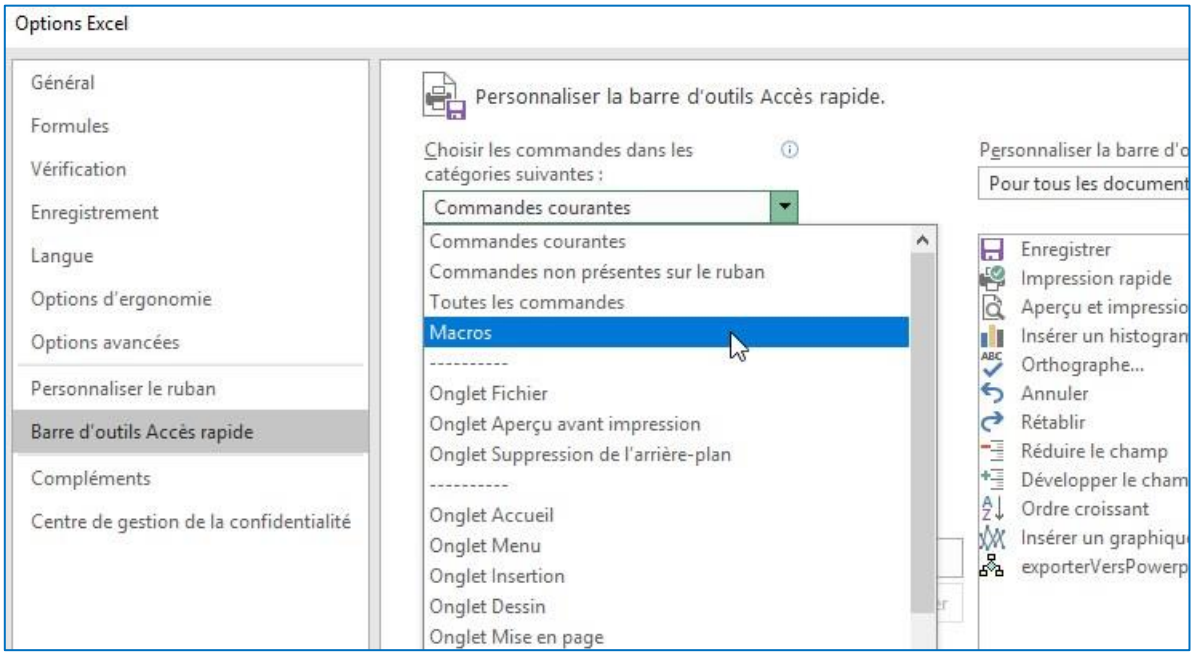
Nous pouvons à présent **quitter les 2 classeurs Excel**, sans enregistrer, puis **relancer Excel**.
Pour **activer le complément**, rendez-vous dans **Fichier** > **Options** > *Compléments*.

Dans le menu **Gérer** en bas de la fenêtre, choisissez **Compléments Excel**, puis *Atteindre* :



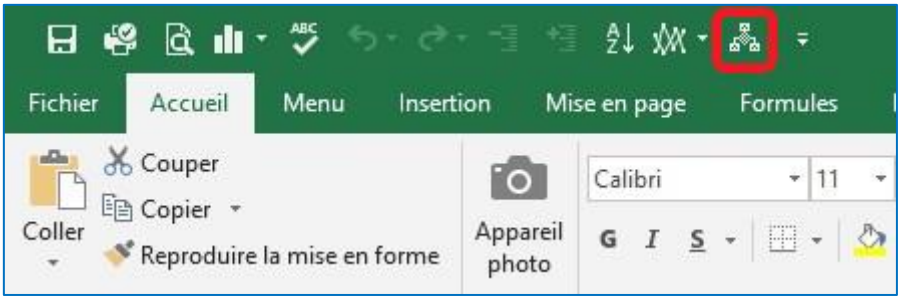

Dans la fenêtre qui s'affiche, sélectionnez l'option **exporterVersPowerpoint** et validez avec **Ok**.

Enfin, pour créer un bouton accessible **depuis n'importe quel classeur**, rendez-vous dans **Fichier** > **Options** > *Barre d'outils Accès rapide*, puis, dans le menu, choisissez les commandes, puis les **Macros** :




Il suffit ensuite de repérer la macro *exporterVersPowerpoint*, de **double-cliquer sur celle-ci** et enfin de valider avec **Ok**.

Ce tutoriel est maintenant terminé, si vous l'avez suivi à la lettre, vous devriez à présent avoir un **nouveau bouton dans la barre d'accès rapide** qui se trouve par défaut en haut à gauche (au-dessus du ruban) :




Le fichier complet est disponible à cette adresse. Vous trouverez également un tutoriel indiquant la marche à suivre pour installer ce dernier.

***Tutoriel réalisé avec Excel 2016***

**Voir la version vidéo de cet article**