

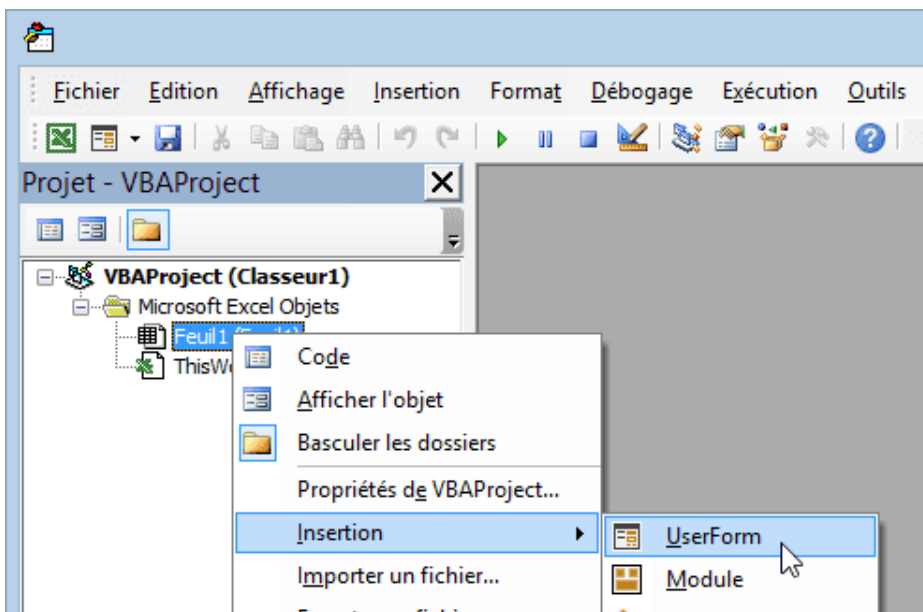


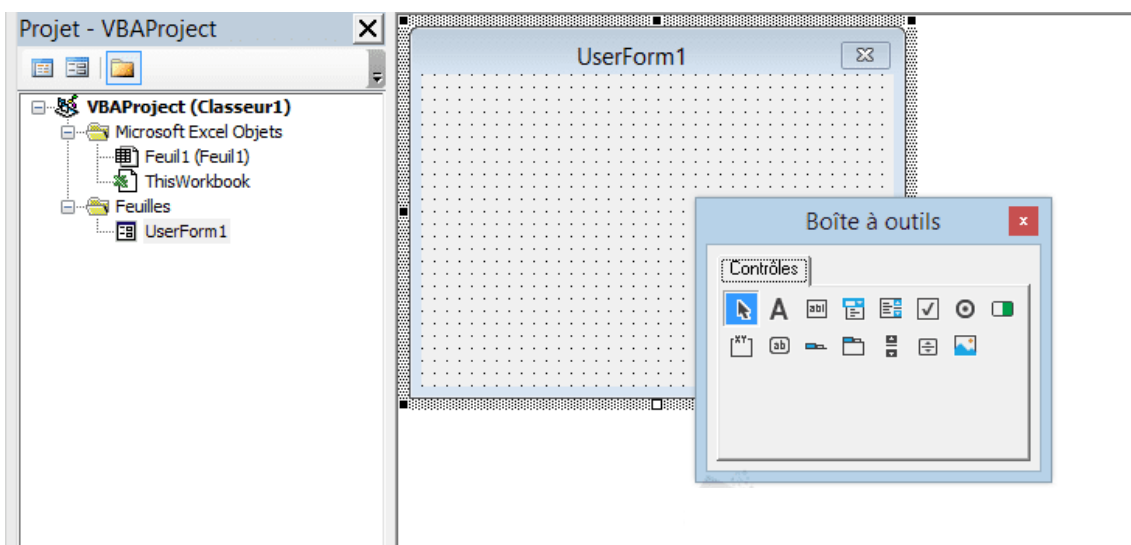
A1	BUREAUTIQUE	
	TP19 Les formulaires et contrôles	

Objectif : Utiliser les formulaires et contrôles en VBA

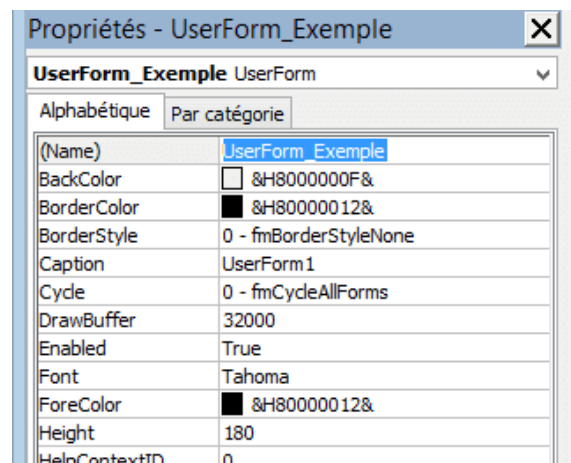
Pour ajouter un UserForm, procédez de la même manière que pour un nouveau module :



La fenêtre de l'UserForm ainsi que celle de la Boîte à outils apparaissent :

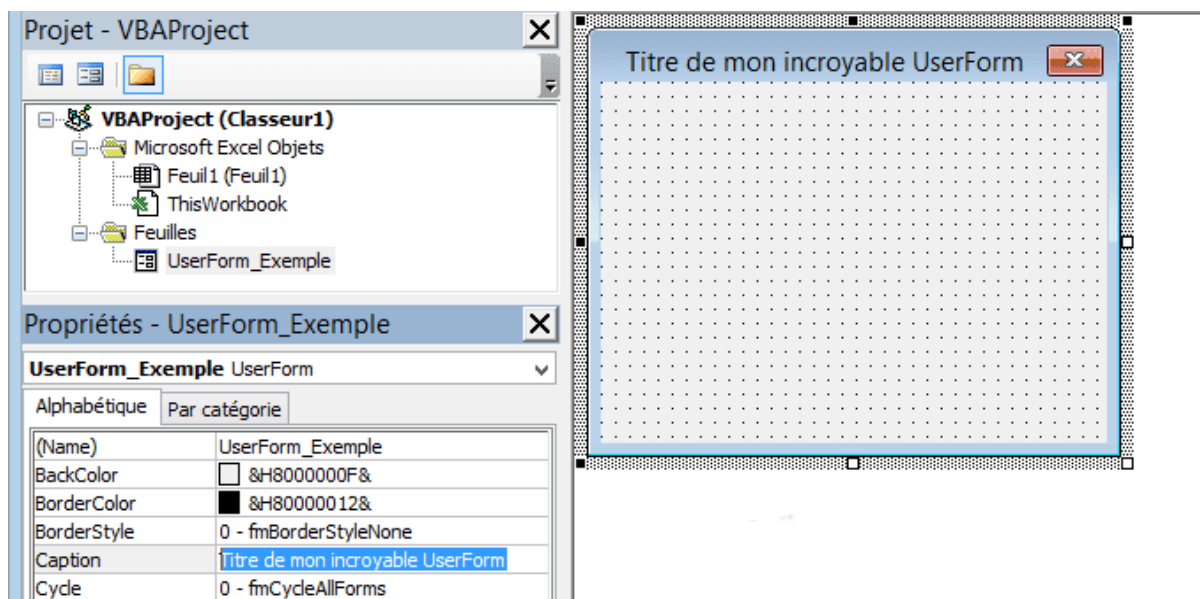


Si la fenêtre des propriétés n'est pas présente, affichez-la (F4) et commencez par modifier le nom de l'UserForm (pour mieux s'y retrouver par la suite) :



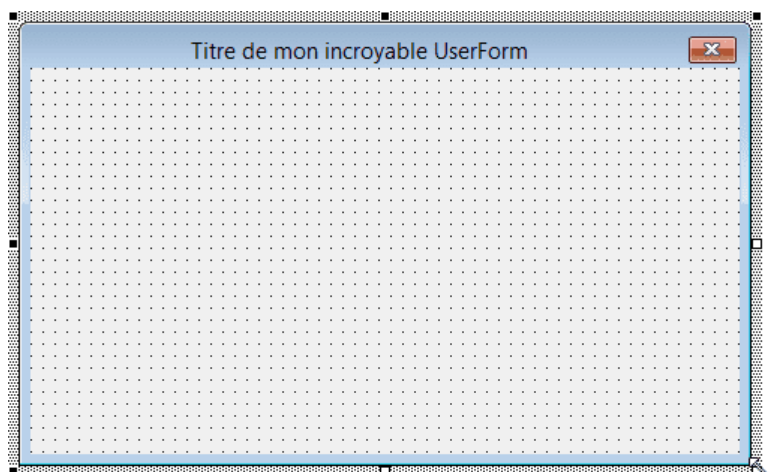
TITRE DE L'USERFORM

Pour modifier le titre de l'UserForm, modifiez sa propriété Caption :



DIMENSIONS DE L'USERFORM

Pour modifier les dimensions de l'UserForm, modifiez ses propriétés Width et Height ou redimensionnez l'UserForm à la main :

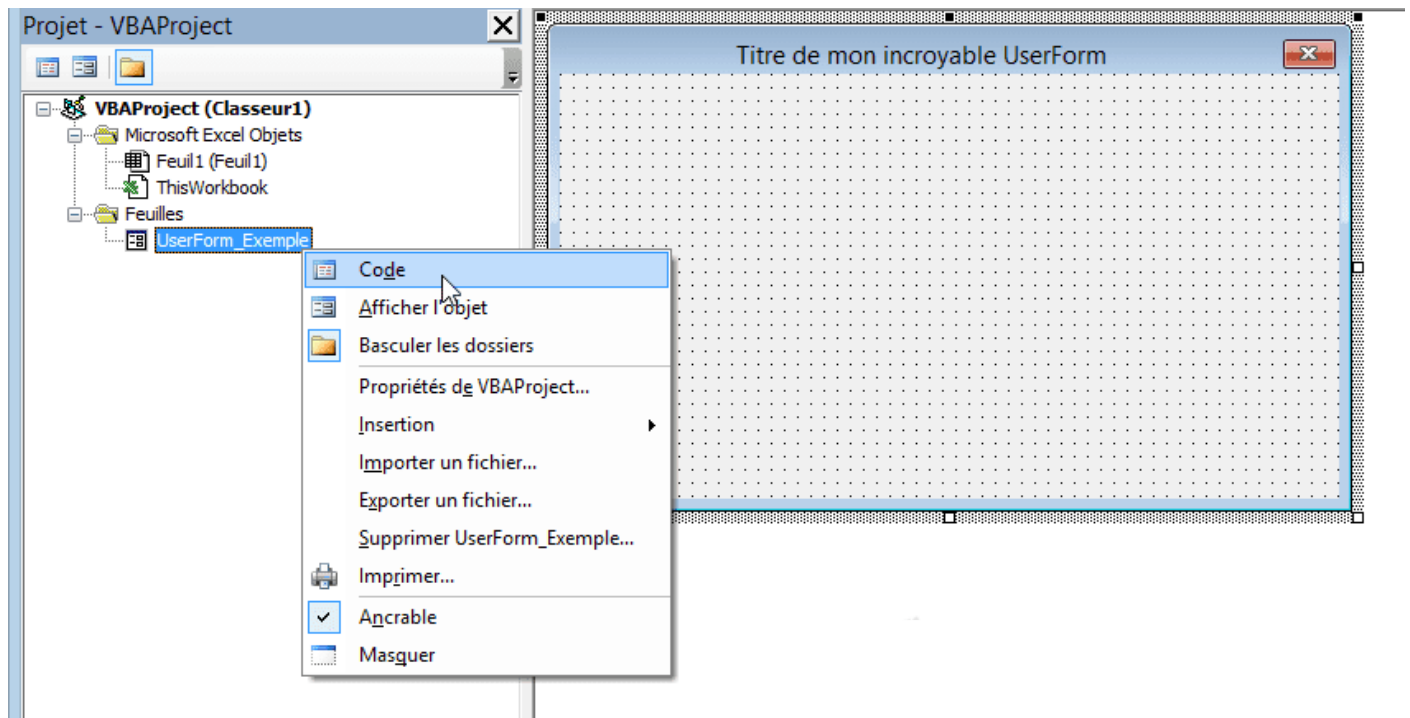


Les propriétés d'un UserForm peuvent également être modifiées à partir d'un code VBA.

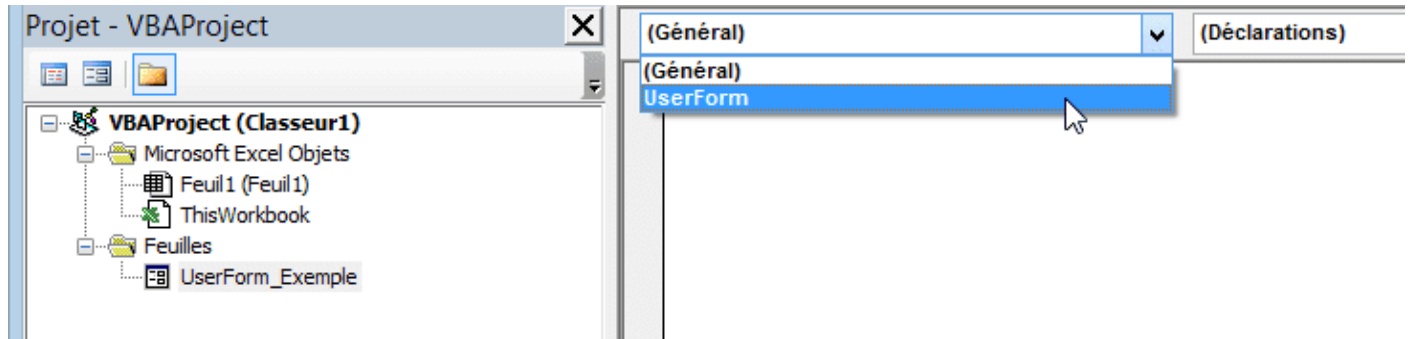
LES ÉVÉNEMENTS DE L'USERFORM

Tout comme le classeur ou ses feuilles, l'UserForm a ses propres événements.

Commencez par afficher le code de l'UserForm :



Cliquez ensuite sur UserForm :

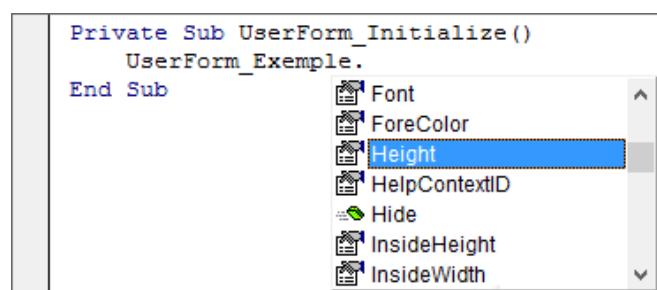


Et sélectionnez l'événement UserForm_Initialize qui se déclenche au lancement de l'UserForm :

```
Private Sub UserForm_Initialize()  
  
End Sub
```

Pour prendre un exemple, nous allons créer deux événements. Le premier pour définir les dimensions initiales de l'UserForm et le second pour augmenter ses dimensions de 50 par clic.

Entrez le nom de l'UserForm suivi d'un . :



La propriété Height est la hauteur et Width la largeur :

```
Private Sub UserForm_Initialize()  
  
    UserForm_Exemple.Height = 250  
    UserForm_Exemple.Width = 250  
  
End Sub
```

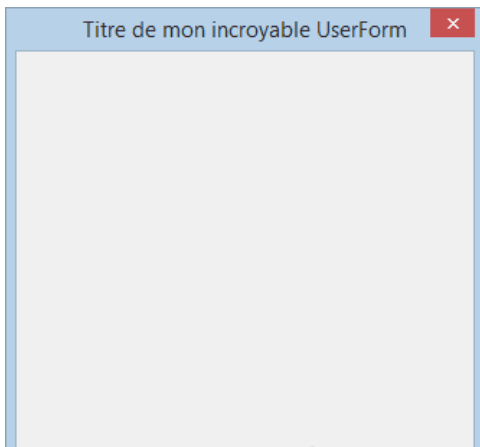
Pour simplifier le code, nous pouvons remplacer le nom de l'UserForm par Me (puisque ce code est placé dans l'UserForm sur lequel on souhaite agir) :

```
Private Sub UserForm_Initialize()  
  
    Me.Height = 250  
    Me.Width = 250  
  
End Sub
```

Le second événement est déclenché au clic sur l'UserForm :

```
Private Sub UserForm_Initialize()  
  
    Me.Height = 250  
    Me.Width = 250  
  
End Sub  
  
Private Sub UserForm_Click()  
  
    Me.Height = Me.Height + 50  
    Me.Width = Me.Width + 50  
  
End Sub
```

Aperçu de l'UserForm (F5) :



LANCER UN USERFORM

Pour lancer un UserForm à partir d'une procédure, utilisez Show :

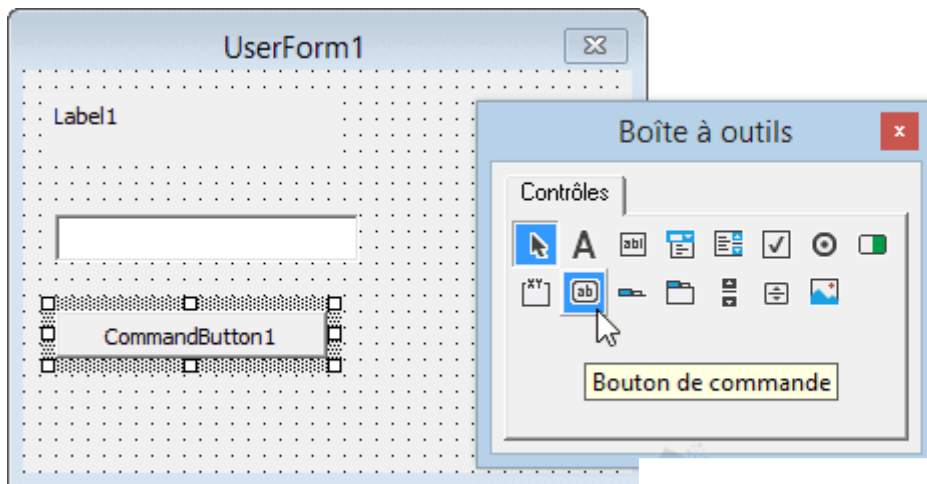
```
Sub lancerUserform()  
  
    UserForm_Exemple.Show  
  
End Sub
```

Les contrôles

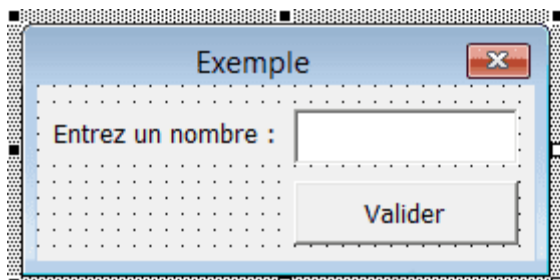
Les contrôles sont les éléments (boutons, intitulés, zone de texte, cases à cocher, etc.) qui peuvent être insérés sur un UserForm (ou sur une feuille Excel).

Les contrôles ont également toute une panoplie de propriétés et d'événements qui diffèrent d'un contrôle à l'autre.

Pour commencer, ajoutez un UserForm et insérez les 3 contrôles suivants : un intitulé Label, une zone de texte TextBox et un bouton CommandButton :



Modifiez les propriétés de l'UserForm et des contrôles (dont les propriétés (Name) pour le nom, Caption pour le texte et Font pour la taille du texte) pour obtenir ceci :

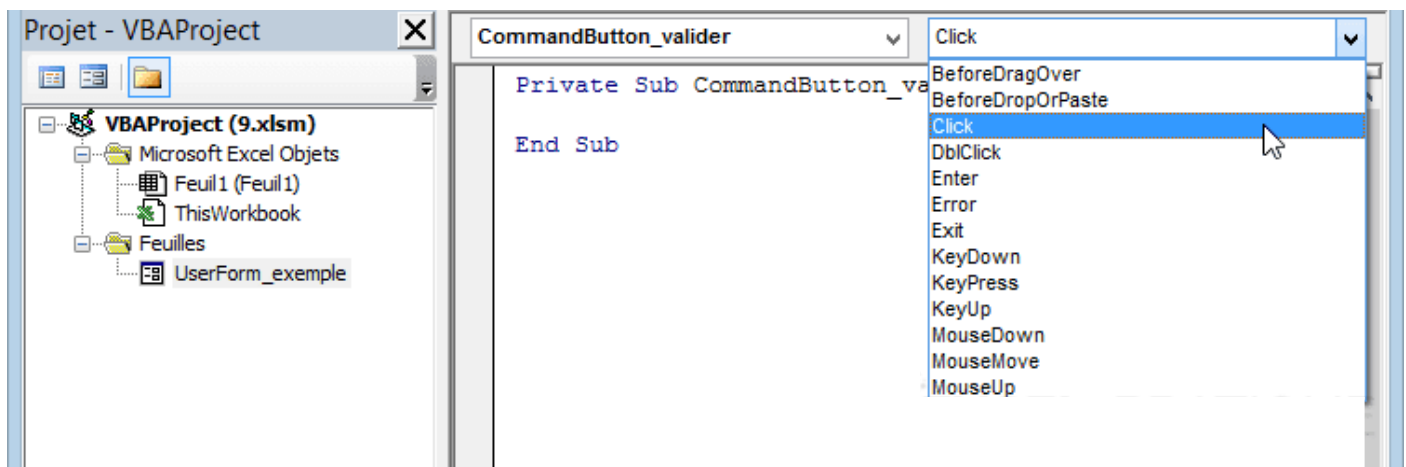


Pour positionner les contrôles de manière plus précise qu'avec un déplacement manuel, modifiez les propriétés Left et Top du contrôle. De même, pour redimensionner les contrôles de manière plus précise qu'avec un redimensionnement manuel, modifiez les propriétés Width et Height.

Pour le moment, lorsque l'on entre un nombre et que l'on clique sur le bouton, il ne se passe rien.

Pour y remédier, nous allons commencer par ajouter un événement pour entrer la valeur de la zone de texte dans la cellule A1 et fermer l'UserForm.

En double-cliquant sur le bouton, un événement par défaut est ajouté dans le code de l'UserForm. Dans ce cas, il s'agit de l'événement souhaité, mais en cas de besoin, vous pouvez sélectionner un autre événement dans la liste :



L'événement Click est déclenché au clic sur le bouton :

```
Private Sub CommandButton_valider_Click()

    'La cellule A1 (de la feuille active) obtient la valeur de la zone de texte nommée
    "TextBox_nombre"
    Range("A1") = TextBox_nombre.Value

    'Fermeture (Unload) de l'UserForm (Me)
    Unload Me

End Sub
```

La valeur est alors entrée dans la cellule A1 avant de fermer l'UserForm.

Nous allons maintenant ajouter un événement qui s'active au changement de valeur de la zone de texte et qui va modifier la couleur de fond si la valeur n'est pas numérique :

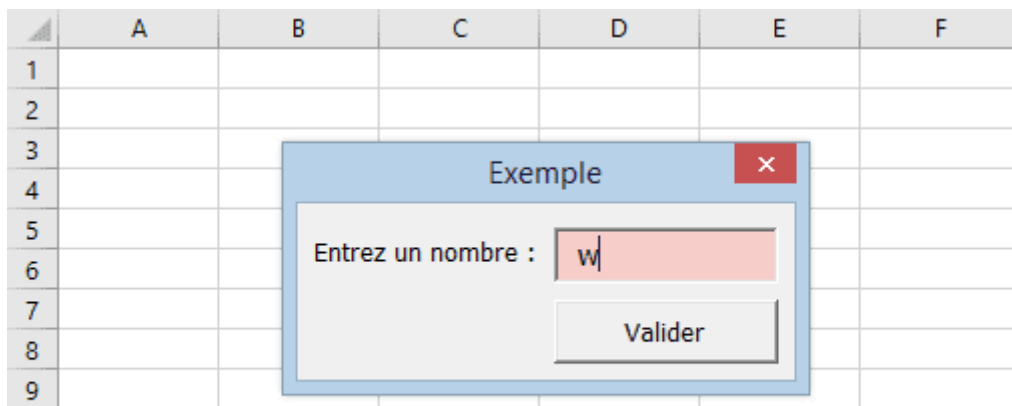
```
Private Sub TextBox_nombre_Change()

    If IsNumeric(TextBox_nombre.Value) Then 'Si valeur numérique
        TextBox_nombre.BackColor = RGB(255, 255, 255) 'Blanc
    Else 'Sinon
        TextBox_nombre.BackColor = RGB(247, 205, 201) 'Rouge clair
    End If

End Sub
```

L'événement est déclenché à chaque entrée ou suppression de caractère dans la zone de texte.

Aperçu :



Il nous reste encore à empêcher la validation du formulaire si la valeur n'est pas numérique en ajoutant une instruction If :

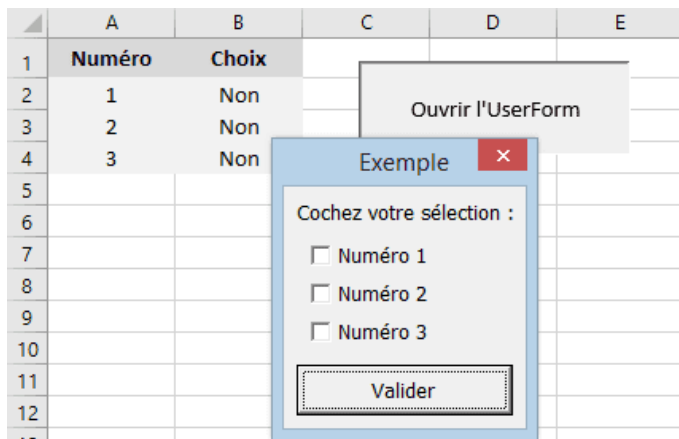
```
Private Sub CommandButton_valider_Click()

    'Si valeur numérique
    If IsNumeric(TextBox_nombre.Value) Then
        Range("A1") = TextBox_nombre.Value
        Unload Me
    End If

End Sub
```

LES CASES À COCHER (CHECKBOX)

Voici un exemple d'utilisation de cases à cocher dont l'objectif est de modifier les valeurs en colonne B en fonction des cases cochées dans l'UserForm :



L'événement Click du bouton enregistre ici les choix de l'utilisateur et ferme ensuite l'UserForm :

```
Private Sub CommandButton_valider_Click()

    'Numéro 1
    If CheckBox1.Value = True Then 'Si coché
        Range("B2") = "Oui"
    Else 'Si décoché
        Range("B2") = "Non"
    End If

    'Numéro 2
    If CheckBox2.Value = True Then 'Si coché
        Range("B3") = "Oui"
    Else 'Si décoché
        Range("B3") = "Non"
    End If

    'Numéro 3
    If CheckBox3.Value = True Then 'Si coché
        Range("B4") = "Oui"
    Else 'Si décoché
        Range("B4") = "Non"
    End If

    'Fermeture
    Unload Me

End Sub
```

Rappelez-vous qu'une condition cherche toujours à savoir si le résultat du test est True :

```
If CheckBox1.Value = True Then 'Si coché
```

Il n'est donc pas utile d'ajouter = True :

```
If CheckBox1.Value Then 'Si coché
```

Pour simplifier encore davantage l'écriture, la propriété Value est la propriété par défaut de la plupart des contrôles, il n'est donc pas nécessaire de l'ajouter (comme nous l'avons vu précédemment avec la propriété Value des cellules) :

```
If CheckBox1.Value Then 'Si coché
```

L'ajout de .Value est donc facultatif :

```
If CheckBox1 Then 'Si coché
```

Suite à ces simplifications, le code ressemble maintenant à ceci :

```
Private Sub CommandButton_valider_Click()  
  
    'Numéro 1  
    If CheckBox1 Then 'Si coché  
        Range("B2") = "Oui"  
    Else 'Si décoché  
        Range("B2") = "Non"  
    End If  
  
    'Numéro 2  
    If CheckBox2 Then 'Si coché  
        Range("B3") = "Oui"  
    Else 'Si décoché  
        Range("B3") = "Non"  
    End If  
  
    'Numéro 3  
    If CheckBox3 Then 'Si coché  
        Range("B4") = "Oui"  
    Else 'Si décoché  
        Range("B4") = "Non"  
    End If  
  
    'Fermeture  
    Unload Me  
  
End Sub
```

Maintenant, imaginez que vous n'avez pas 3 mais 30 cases à cocher ...

Dans ce cas, l'utilisation d'une boucle est plus que bienvenue :

```
Private Sub CommandButton_valider_Click()  
  
    Dim i As Integer  
  
    'Boucle des cases à cocher  
    For i = 1 To 3  
        If Controls("CheckBox" & i) Then 'Si coché  
            Range("B" & i + 1) = "Oui"  
        Else 'Si décoché  
            Range("B" & i + 1) = "Non"  
        End If  
    Next  
  
    'Fermeture  
    Unload Me  
  
End Sub
```

Controls("CheckBox1") est l'équivalent du contrôle CheckBox1 et permet d'accéder à un contrôle en fonction de son nom, ce qui peut être très pratique notamment dans une boucle.

Dans cet exemple, les cases sont toutes décochées à l'ouverture de l'UserForm.

Pour cocher les cases dont la valeur de la cellule correspondante est Oui au lancement de l'UserForm, ajoutez l'événement UserForm_Initialize et les tests suivants :

```
Private Sub UserForm_Initialize()  
  
    If Range("A2") = "Oui" Then  
        CheckBox1 = True  
    End If  
  
    If Range("B2") = "Oui" Then  
        CheckBox2 = True  
    End If  
  
    If Range("C2") = "Oui" Then  
        CheckBox3 = True  
    End If  
  
End Sub
```

Pour simplifier ce code, vous avez la possibilité d'écrire l'instruction If sur une seule ligne et sans End If lorsqu'il n'y a qu'une seule action à effectuer :

```
Private Sub UserForm_Initialize()  
  
    If Range("B2") = "Oui" Then CheckBox1 = True  
  
    If Range("B3") = "Oui" Then CheckBox2 = True  
  
    If Range("B4") = "Oui" Then CheckBox3 = True  
  
End Sub
```

L'utilisation d'une boucle est également possible :

```
Private Sub UserForm_Initialize()  
  
    Dim i As Integer  
  
    For i = 1 To 3  
        If Range("B" & i + 1) = "Oui" Then Controls("CheckBox" & i) = True  
    Next  
  
End Sub
```

	A	B	C	D	E
1	Numéro	Choix			
2	1	Oui			
3	2	Non			
4	3	Oui			
5					
6					
7					
8					
9					
10					
11					
12					
13					

Ouvrir l'UserForm

Exemple

Cochez votre sélection :

☒ Numéro 1

☐ Numéro 2

☒ Numéro 3

Valider

LES BOUTONS D'OPTION (OPTIONBUTTON)

Contrairement aux cases à cocher, l'utilisateur ne peut choisir qu'un seul bouton d'option par groupe.

Il faudra séparer ici les boutons d'option en 2 groupes et enregistrer les résultats dans 2 cellules :

Sondage

Que pensez-vous de ce cours VBA ?

- ☐ Passionnant
- ☐ Intéressant
- ☐ Bof
- ☐ Une horreur

Prévoyez-vous de développer un premier projet en VBA à la fin de ce cours ?

- ☐ Oui
- ☐ Non
- ☐ Je ne sais pas
- ☐ Chaque chose en son temps

Enregistrer

La première étape consiste à créer les groupes de boutons (car pour le moment vous ne pouvez sélectionner qu'une seule réponse parmi les 8 réponses).

Pour faire cela, sélectionnez les 4 premiers contrôles et entrez une valeur dans la propriété GroupName :

Projet - VBAProject

- VBAProject (userform3b.xlsm)
 - Microsoft Excel Objects
 - Feuilles
 - UserForm_exemple
 - Modules

Propriétés - UserForm_exemple

Alphabétique Par catégorie

Accelerator	
Alignment	1 - fmAlignmentRight
AutoSize	False
BackColor	&H8000000F&
BackStyle	1 - fmBackStyleOpaque
Caption	
ControlSource	
ControlTipText	
Enabled	True
Font	
ForeColor	&H80000012&
GroupName	1
Height	18
HelpContextID	0

Sondage

Que pensez-vous de ce cours VBA ?

- ☐ Passionnant
- ☐ Intéressant
- ☐ Bof
- ☐ Une horreur

Prévoyez-vous de développer un premier projet en VBA à la fin de ce cours ?

- ☐ Oui
- ☐ Non
- ☐ Je ne sais pas
- ☐ Chaque chose en son temps

Enregistrer

Répétez ensuite l'opération pour les 4 autres contrôles (en entrant une valeur différente).

Vous pouvez à présent sélectionner une réponse par groupe.

Pour enregistrer les réponses dans les cellules de la feuille, nous allons tout d'abord ajouter l'événement Click du bouton Enregistrer.

Il faut ensuite ajouter une boucle pour chaque groupe de boutons d'option et enregistrer l'information lorsque la valeur du contrôle est True :

```
Private Sub CommandButton_valider_Click()  
  
    Dim i As Integer  
  
    'Question 1  
    For i = 1 To 4  
        If Controls("OptionButton_a_" & i) Then Range("A2") =  
Controls("OptionButton_a_" & i).Caption  
    Next  
  
    'Question 2  
    For i = 1 To 4  
        If Controls("OptionButton_b_" & i) Then Range("B2") =  
Controls("OptionButton_b_" & i).Caption  
    Next  
  
    'Fermeture  
    Unload Me  
  
End Sub
```

	A	B	C	D	E	F
1	Question 1	Question 2		Afficher le sondage		
2	Intéressant	Oui				
3						

Mais plutôt que d'enregistrer le choix au format texte, nous allons plutôt enregistrer son numéro (de 1 à 4) :

```
Private Sub CommandButton_valider_Click()  
  
    Dim i As Integer  
  
    'Question 1  
    For i = 1 To 4  
        If Controls("OptionButton_a_" & i) Then Range("A2") = i  
    Next  
  
    'Question 2  
    For i = 1 To 4  
        If Controls("OptionButton_b_" & i) Then Range("B2") = i  
    Next  
  
    'Fermeture  
    Unload Me  
  
End Sub
```

	A	B	C	D	E	F
1	Question 1	Question 2		Afficher le sondage		
2	2	1				
3						

Si l'on souhaite que le formulaire ne puisse être validé que lorsque l'utilisateur a répondu aux 2 questions, une solution consiste à enregistrer le choix de chaque groupe dans une variable, vérifier ensuite s'il y a un choix pour chacune des 2 variables et enregistrer les choix dans les cellules :

```

Private Sub CommandButton_valider_Click()

    Dim i As Integer, choix1 As Integer, choix2 As Integer

    'Question 1
    For i = 1 To 4
        If Controls("OptionButton_a_" & i) Then choix1 = i
    Next

    'Question 2
    For i = 1 To 4
        If Controls("OptionButton_b_" & i) Then choix2 = i
    Next

    'Si 2 réponses
    If choix1 > 0 And choix2 > 0 Then

        'Enregistrement
        Range("A2") = choix1
        Range("B2") = choix2

        'Fermeture
        Unload Me

    'Si une ou plusieurs réponses manquantes
    Else

        'Message d'erreur
        MsgBox "Vous devez répondre à toutes les questions avant de valider le formulaire.", 48, "Erreur"

    End If

End Sub

```

LA LISTE DÉROULANTE (COMBOBOX) ET LA ZONE DE LISTE (LISTBOX)

Le point de départ de ce nouvel exemple est le fichier : UserForm

	A	B	C	D	E	F	G	H
1	Suisse	France	Belgique	Canada		Ouvrir l'UserForm		
2	Zürich	Paris	Bruxelles	Montréal				
3	Genève	Lyon	Anvers	Québec				
4	Lausanne	Toulouse	Namur					
5	Berne	Nantes	Bruges					
6	Sion	Lille						
7		Bordeaux						
8		Rennes						
9								
10								
11								
12								
13								
14								
15								

Exemple

Pays :

Villes :

Valider

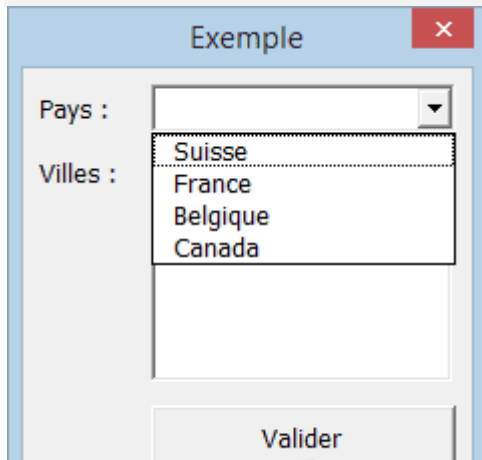
Au lancement de l'UserForm, nous voulons que les 4 pays soient chargés dans la liste déroulante (à l'aide de la méthode AddItem) :

```
Private Sub UserForm_Initialize()

    Dim i As Integer

    'Boucle pour ajouter les 4 pays à la liste déroulante
    For i = 1 To 4
        ComboBox_pays.AddItem Cells(1, i)
    Next

End Sub
```



Au changement de sélection dans la liste déroulante, la liste des villes correspondant au pays choisi doit ensuite être affichée dans la zone de liste.

Pour faire cela, nous avons besoin de connaître le numéro de colonne ainsi que le nombre de villes de cette colonne.

La propriété `ListIndex` de la liste déroulante correspond au numéro de la sélection dans la liste (contrairement à la propriété `Value` qui correspond à la valeur au format texte).

Sachant que `ListIndex` commence à 0 (comme les tableaux), le numéro de colonne est donc :

```
colonne = ComboBox_Pays.ListIndex + 1
```

Pour obtenir le nombre de lignes de la colonne du pays choisi, nous pouvons rechercher le numéro de ligne de la dernière cellule d'un bloc de cellules non vides, comme ceci :

```
nbLignes = Cells(1, colonne).End(xlDown).Row
```

Grâce à ces informations, il est désormais possible de créer l'événement `Change` de la liste déroulante :

```
Private Sub ComboBox_Pays_Change()

    Dim colonne As Integer, nbLignes As Integer

    'Zone de liste vidée (sinon les villes sont ajoutées à la suite)
    ListBox_villes.Clear

    'Numéro de la sélection
    colonne = ComboBox_pays.ListIndex + 1

    'Si le numéro de colonne = 0 (donc si aucun pays sélectionné) la procédure est
    quittée
    If colonne = 0 Then Exit Sub

    'Nombre de lignes de la colonne du pays choisi
    nbLignes = Cells(1, colonne).End(xlDown).Row

    'Boucle pour ajouter les villes dans la zone de liste
```

```

For i = 2 To nbLignes
    ListBox_villes.AddItem Cells(i, colonne)
Next
End Sub

```

Il ne reste ensuite plus qu'à ajouter un événement au clic sur le bouton Valider pour traiter cette information. Dans ce cas, un simple affichage de la sélection dans une boîte de dialogue :

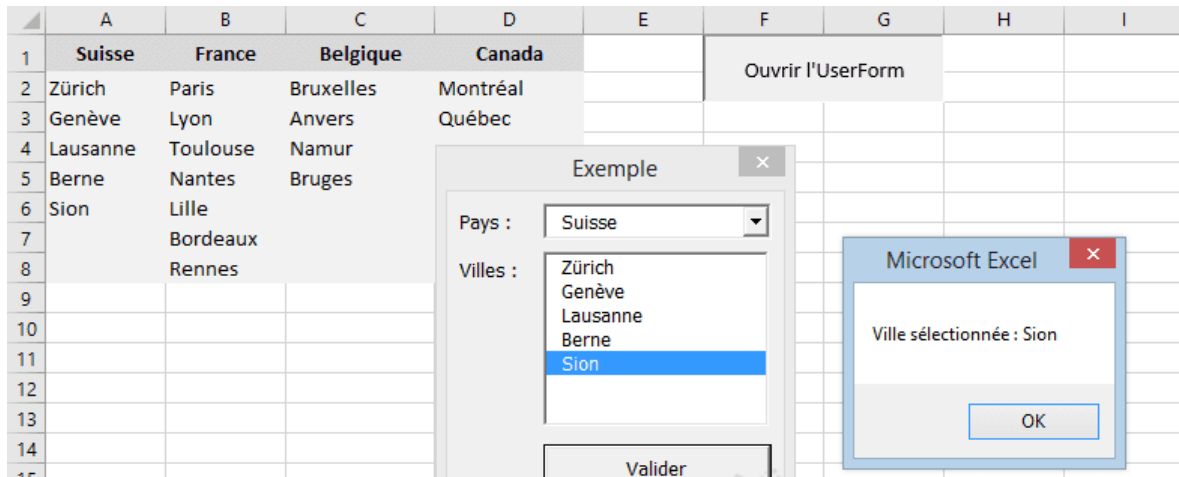
```

Private Sub CommandButton_valider_Click()

    MsgBox "Ville sélectionnée : " & ListBox_villes '(propriété Value de
ListBox_villes)

End Sub

```

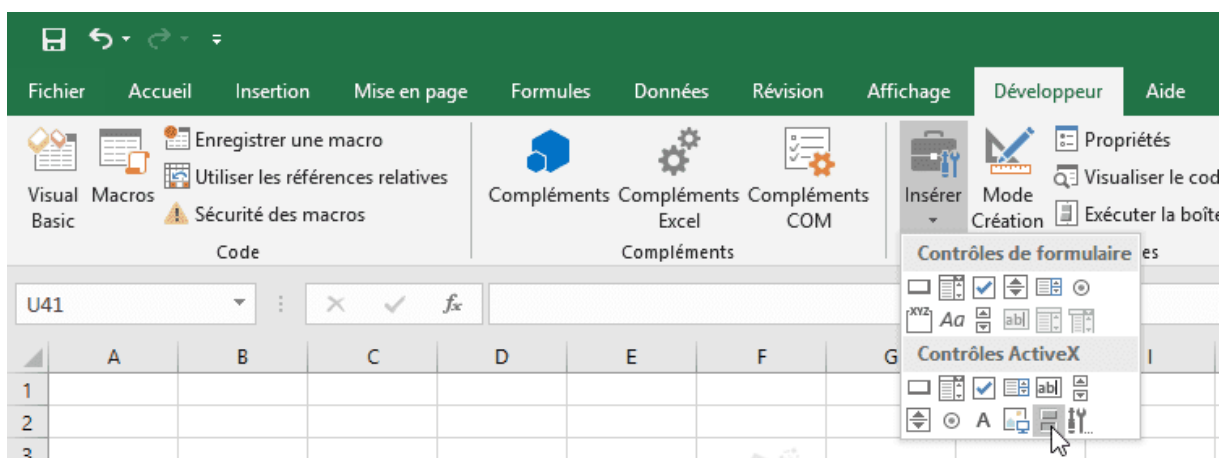


INSERTION SUR UNE FEUILLE

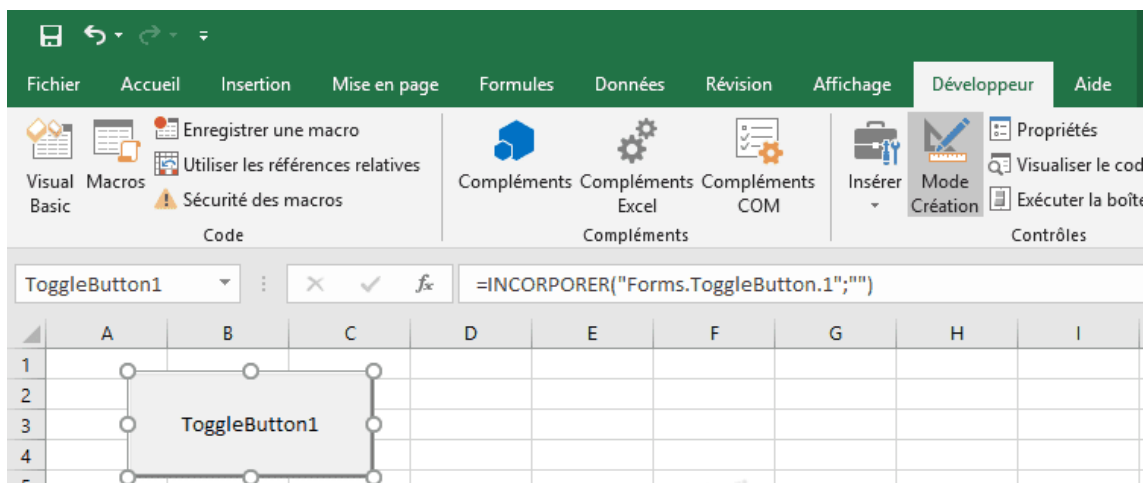
Les contrôles peuvent être utilisés également sur une feuille Excel. Pour cet exemple, nous ajouterons donc le contrôle directement sur la feuille.

LE BOUTON BASCULE (TOGGLEBUTTON)

Insérez pour commencer un bouton bascule (contrôle ActiveX) à partir de l'onglet Développeur :



Notez que pour manipuler un contrôle ActiveX sur une feuille, le Mode Création doit être activé :

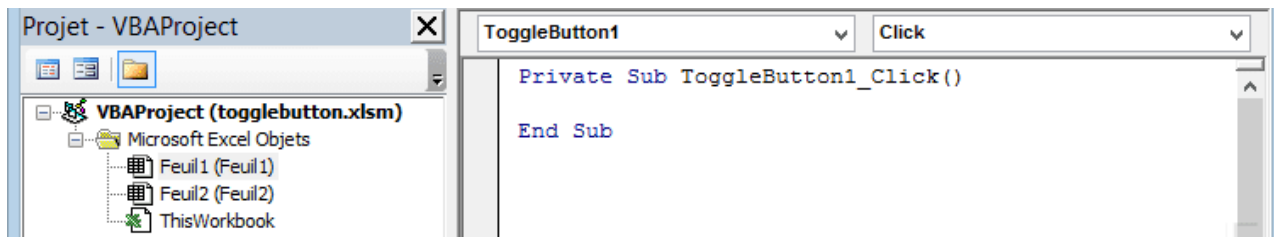


Double-cliquez maintenant sur le bouton et modifiez ses propriétés pour obtenir :

L'objectif ici est de masquer la feuille 2 lorsque le bouton est pressé ou de l'afficher dans le cas contraire.

	A	B	C	D	E
1					
2		La feuille 2 est affichée			
3					

Le précédent double-clic a également ajouté l'événement Click du bouton dans la feuille où se trouve le bouton :



Il ne reste plus qu'à entrer les instructions à exécuter au clic sur le bouton :

```
Private Sub ToggleButton1_Click()

    'Si le bouton est pressé
    If ToggleButton1 Then

        'Masquer la feuille et modifier le texte du bouton
        Sheets("Feuil2").Visible = 2
        ToggleButton1.Caption = "La feuille 2 est masquée"

    'Sinon
    Else

        'Afficher la feuille et modifier le texte du bouton
        Sheets("Feuil2").Visible = -1
        ToggleButton1.Caption = "La feuille 2 est affichée"

    End If

End Sub
```

	A	B	C	D	E
1					
2		La feuille 2 est masquée			
3					