Vk

Zeste de savoir

L'objet UserForm

samedi 16 novembre 2024

Table des matières

Introduction

Certains programmes nécessitent de l'interaction avec l'utilisateur, en lui demandant de saisir des informations ou de sélectionner des options par exemple. D'autres doivent lui présenter des éléments d'une certaine façon qui ne serait pas possible dans l'application même (Excel, Word, …).

Recourir à des boîtes de dialogue basiques telles que des InputBox ou encore ajouter des contrôles de formulaire peut alors parfois suffire. Toutefois, ils montrent vite leurs limites.

Grâce aux boîtes de dialogue personnalisées (UserForm), nous pouvons construire des fenêtres simples ou complexes afin de répondre aux besoins. Il est ainsi possible d'y ajouter toutes sortes de contrôles, de mettre en forme l'affichage ou encore de gérer les événements.

Au cours de cette série de billets, nous allons apprendre à créer nos propres fenêtres pas à pas. Ce billet portant sur l'objet UserForm en est le premier. Nous nous placerons dans l'environnement Microsoft Office, et plus particulièrement dans Excel pour cela.

Une boîte de dialogue personnalisée est représentée par un objet UserForm.

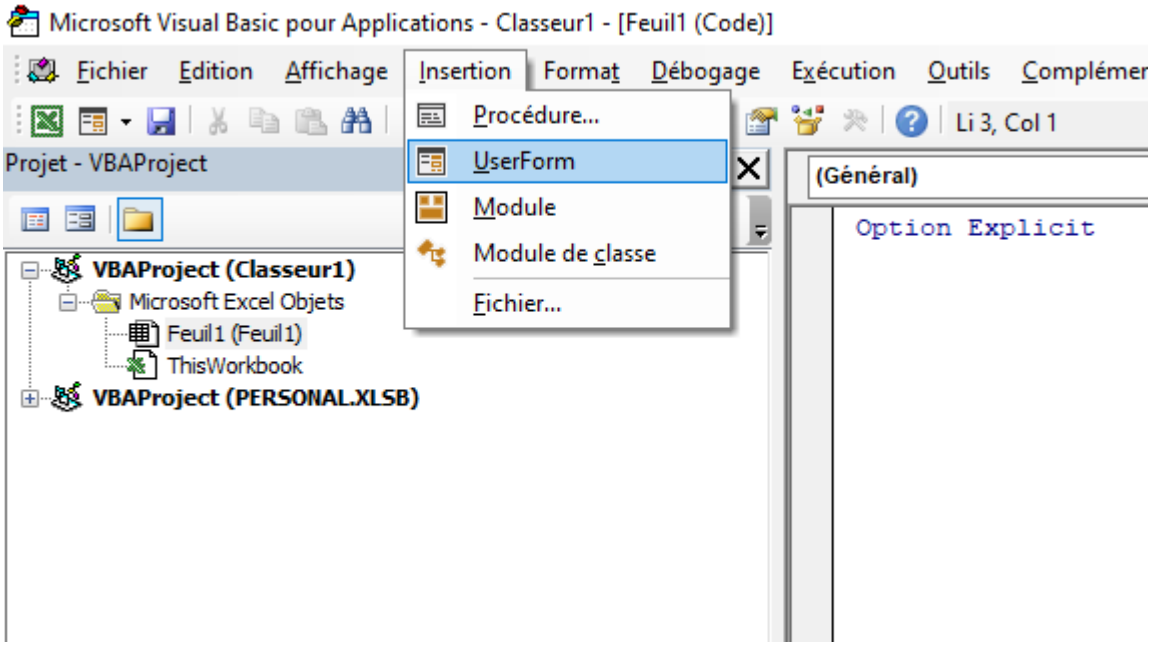
Il est possible de gérer celle-ci aisément à travers le VBE, que ce soit pour la créer, la paramétrer ou encore l'exécuter. Nous pouvons également réaliser certaines opérations en VBA si besoin.

1. Création

L'éditeur fourni avec Microsoft Office nous permet de créer des fenêtres facilement.

1.1. Insertion

Pour insérer un UserForm, nous pouvons passer par le menu "Insertion" ou encore par le menu contextuel au niveau de l'explorateur de projets.



Le résultat se présente alors ainsi :

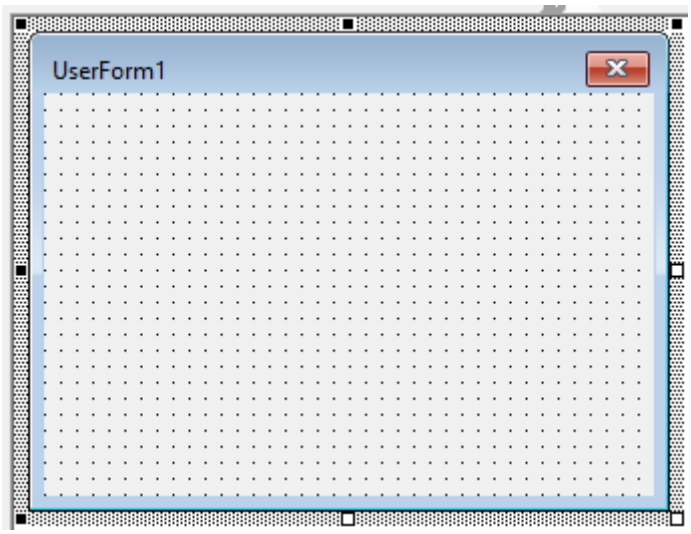


FIGURE 1.1. – Vue objet de la boîte de dialogue personnalisée.

1.2. Bascule entre le code et l'objet

Nous avons actuellement en visuel l'objet. Nous verrons dans un autre billet qu'un UserForm peut aussi avoir du code, notamment pour les événements.

Pour basculer de la vue objet à la vue code il y a plusieurs possibilités comme illustré dans l'image ci-dessous. La plus simple est le raccourci F7 pour obtenir la vue code et Maj + F7 pour ouvrir la vue objet.

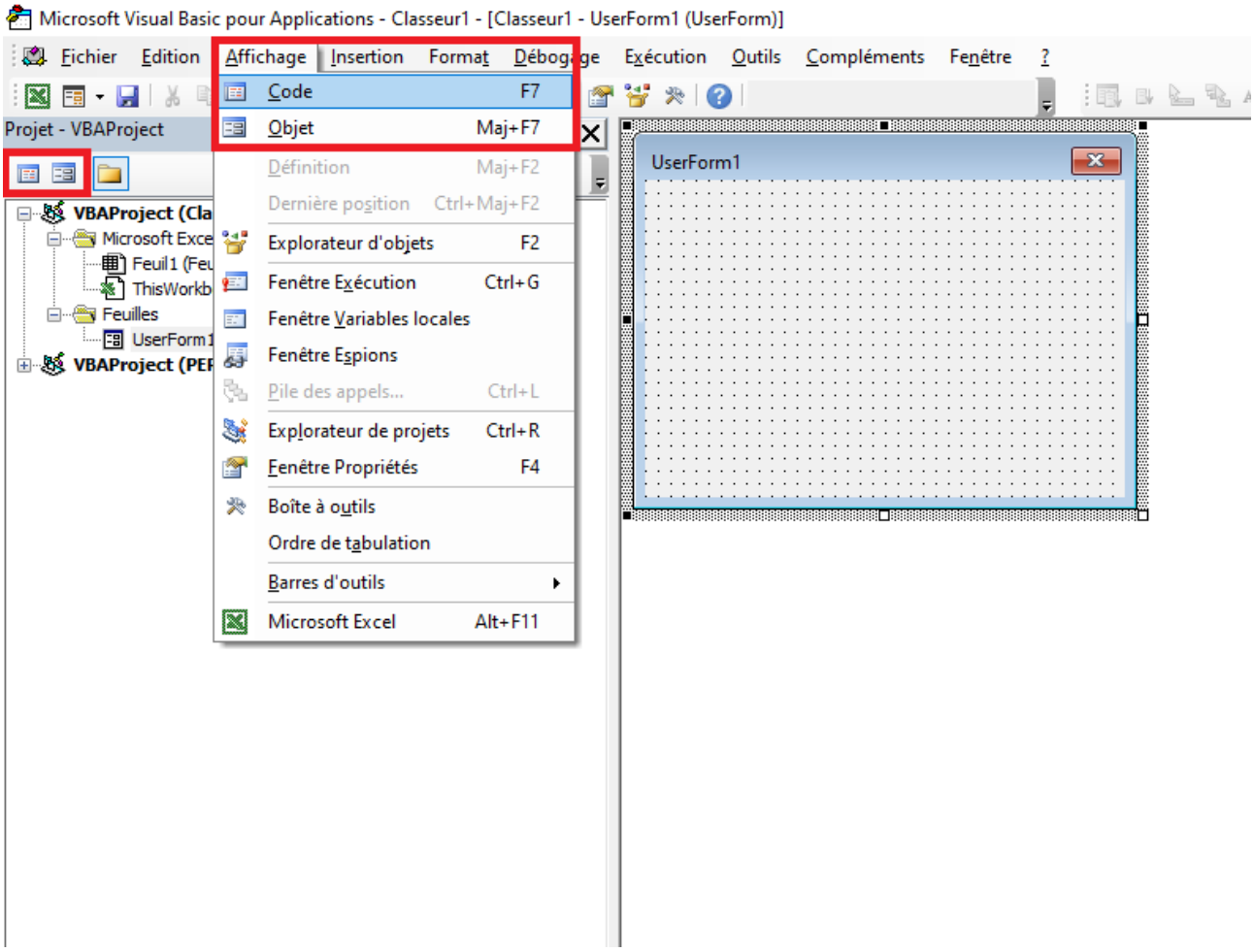


FIGURE 1.2. – Bascule entre le code et l'objet pour un UserForm.

1.3. Explorateur d'objets

L'explorateur d'objets (raccourci F2) est un bon outil pour en apprendre davantage sur l'interface de programmation d'une classe (propriétés et méthodes) ainsi que pour accéder à l'aide en ligne.

Cela pourra nous servir pour savoir à quoi correspondent certaines propriétés en VBA par exemple.

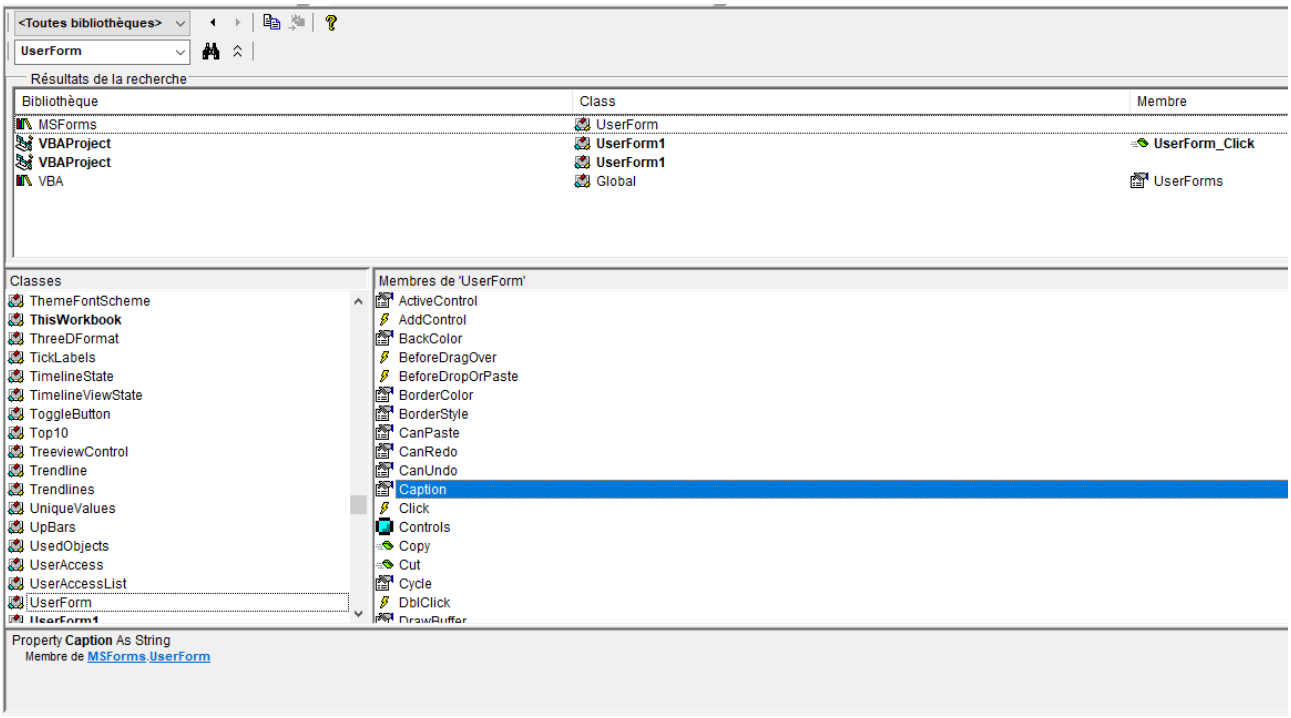


FIGURE 1.3. – L'explorateur d'objets pour la classe UserForm.

Au fil de cette première section, nous avons vu comment ajouter des UserForm dans nos projets.

2. Paramétrage

Nous allons maintenant voir comment paramétrer notre fenêtre grâce aux propriétés.

Pour vous les présenter, le plus simple est de passer par la fenêtre adéquate. Si celle-ci n'est pas visible dans l'éditeur, nous pouvons l'ajouter de diverses manières, par le menu "Affichage", par le menu contextuel de la vue objet de l'UserForm ou encore via le raccourci F4.

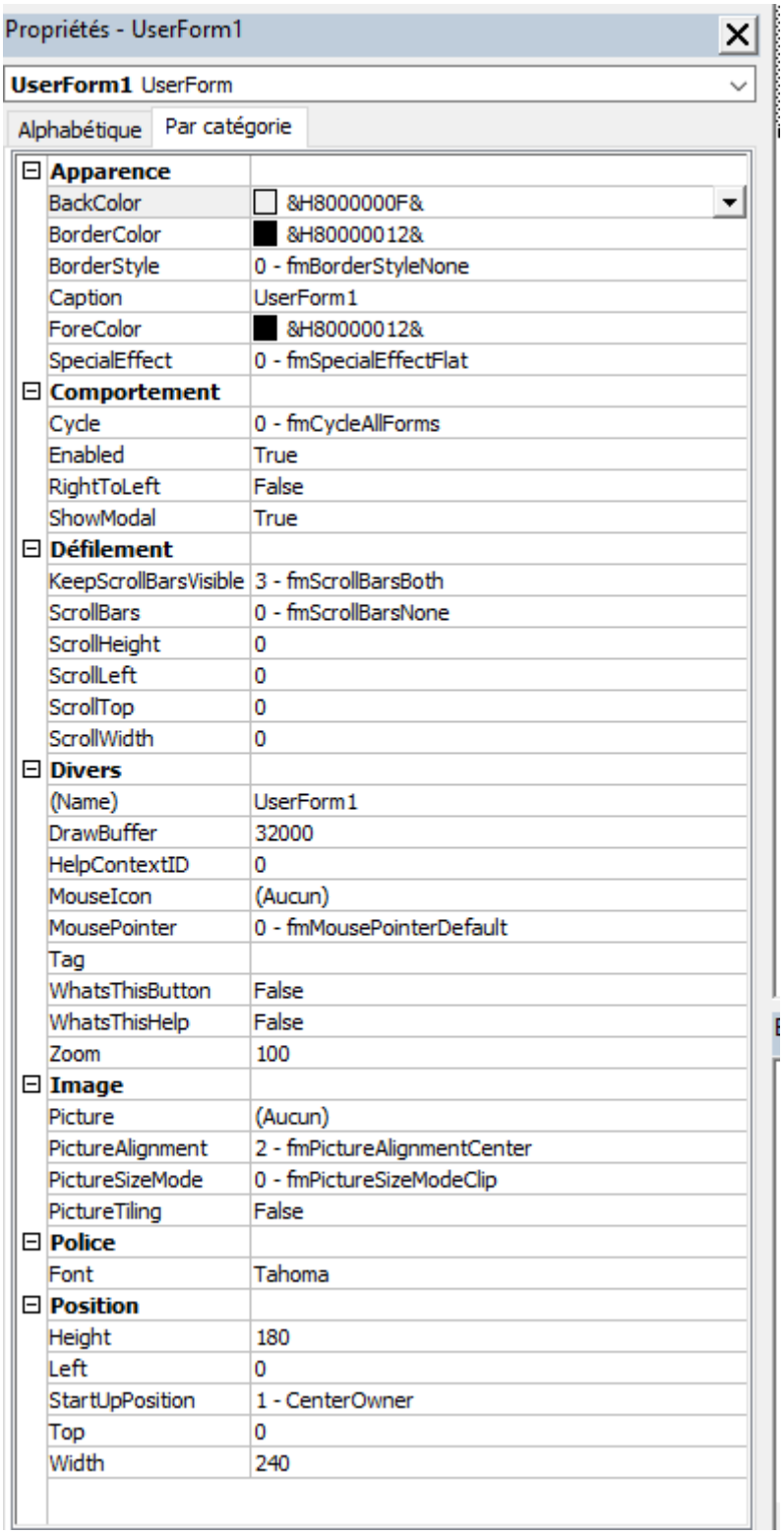


FIGURE 2.4. – Fenêtre des propriétés de l'UserForm.

Il est bienvenu de trier les propriétés par catégorie afin de s'y retrouver plus facilement.

2.1. Divers

Une première propriété à renseigner est `Name` dans la catégorie "Divers". Celle-ci permet de donner un nom parlant à notre objet lorsque nous voulons l'appeler. Nous pouvons mettre `UserForm_Exemple1` et constater que le nom a bien changé dans l'arborescence du projet.

2.2. Position

Depuis la catégorie "Position", nous sommes en mesure de renseigner la position de départ de la fenêtre ainsi que sa taille.

2.2.1. Position de départ

La choix de positionnement de départ est fait au moyen de `StartUpPosition` qui peut prendre une des valeurs suivantes :

Option liste	Valeur	Description
Manual	0	L'objet est placé selon les propriétés `Left` et `Top`
CenterOwner	1	L'objet est centré par rapport à l'élément auquel il appartient
CenterScreen	2	L'objet est centré par rapport à l'écran
Windows Default	3	L'objet est placé dans le coin supérieur gauche de l'écran

Il n'y a pas de constantes ou d'énumération à utiliser en VBA pour ces valeurs.

Dans le cas d'un choix manuel, la position de départ est celle des coordonnées du coin supérieur gauche (propriétés `Left` et `Top`).

2.2.2. Taille

On fait varier la largeur ainsi que la hauteur au moyen des propriétés respectives `Width` et `Height`.

En plus de ces propriétés, la vue objet permet d'agrandir et de réduire facilement la largeur et/ou la hauteur de l'UserForm à partir d'un clique glissant sur les poignées de redimensionnement (c'est-à-dire les carrés blancs).

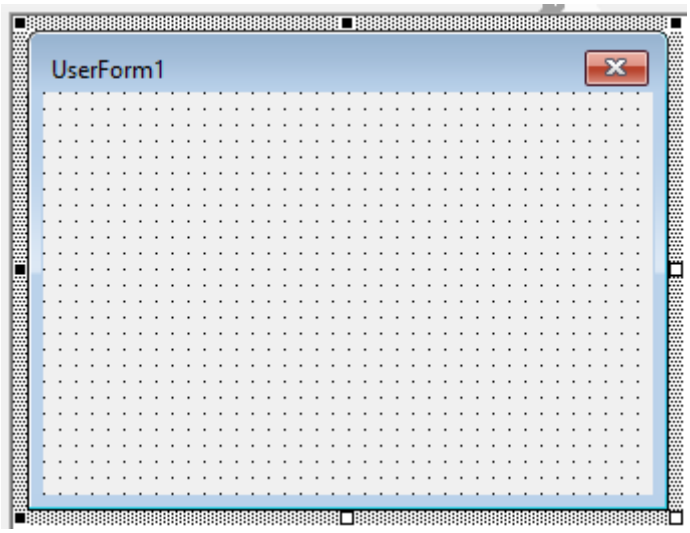


FIGURE 2.5. – Vue objet de la boîte de dialogue personnalisée.

2.3. Apparence et police

Les catégories ”Apparence” et ”Police” nous offre un moyen de mettre en forme l'ensemble.

> *i* Certaines propriétés (je pense à `Font` et à `ForeColor`) sont héritées par les contrôles créés par la suite. Il est donc intéressant de définir les valeurs voulues au niveau de l'objet UserForm pour gagner du temps et pour uniformiser le style. Notons qu'il est aussi possible de copier-coller des contrôles ce qui a pour effet de conserver leurs propriétés de style.

2.3.1. Couleur de fond

La couleur de fond de la fenêtre est contenue dans `BackColor`.

Pour les couleurs, soit nous écrivons la valeur voulue directement, soit nous passons par le choix des couleurs qui apparaît en cliquant sur la petite flèche.

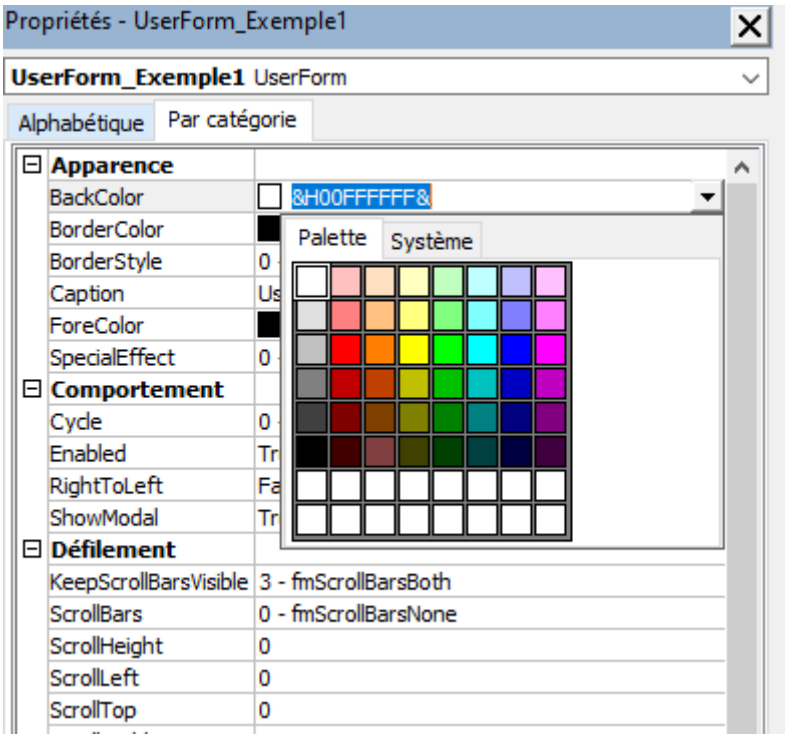


FIGURE 2.6. – Couleur palette.

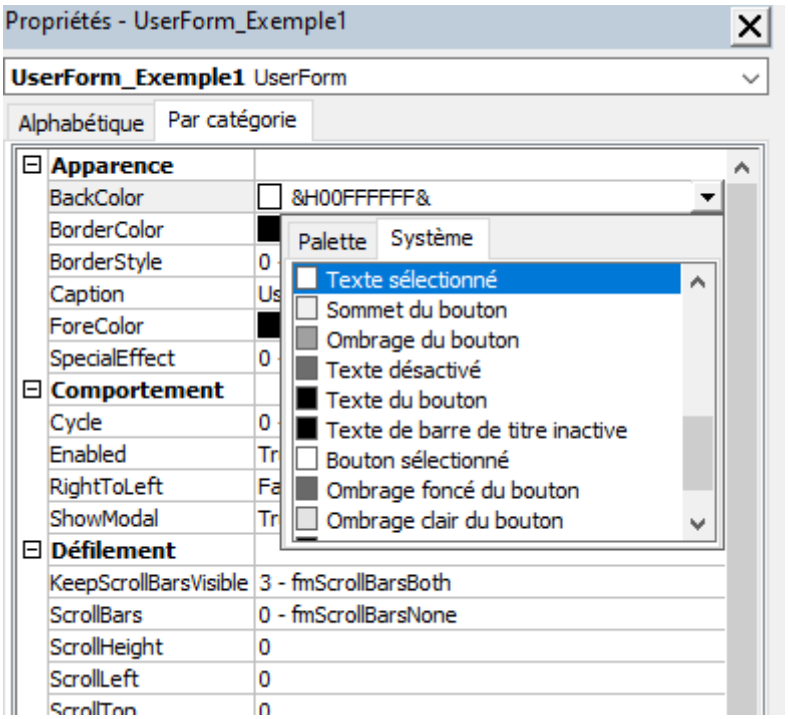


FIGURE 2.7. – Couleur système.

Pour saisir une couleur personnalisée directement, l'ensemble est délimité par `&`. Les deux premiers caractères sont `H0` (code pour indiquer que c'est une couleur personnalisée) et sont suivis des six caractères hexadécimales de la couleur dans l'ordre bleu, vert puis rouge. L'ordre est donc inversé. Par exemple, pour une couleur `#A3C589`, il faudra saisir `&H089C5A3&`. Tout ceci est expliqué dans cet article .

Par ailleurs, nous verrons dans un autre billet l'usage de la fonction RGB de VBA afin de personnaliser facilement les couleurs depuis les macros.

2.3.2. Police

En double-cliquant sur le nom de police ou en cliquant sur les trois points au niveau de la valeur, nous ouvrons la fenêtre de police qui contient divers choix : police, style, taille, effets ou encore script.

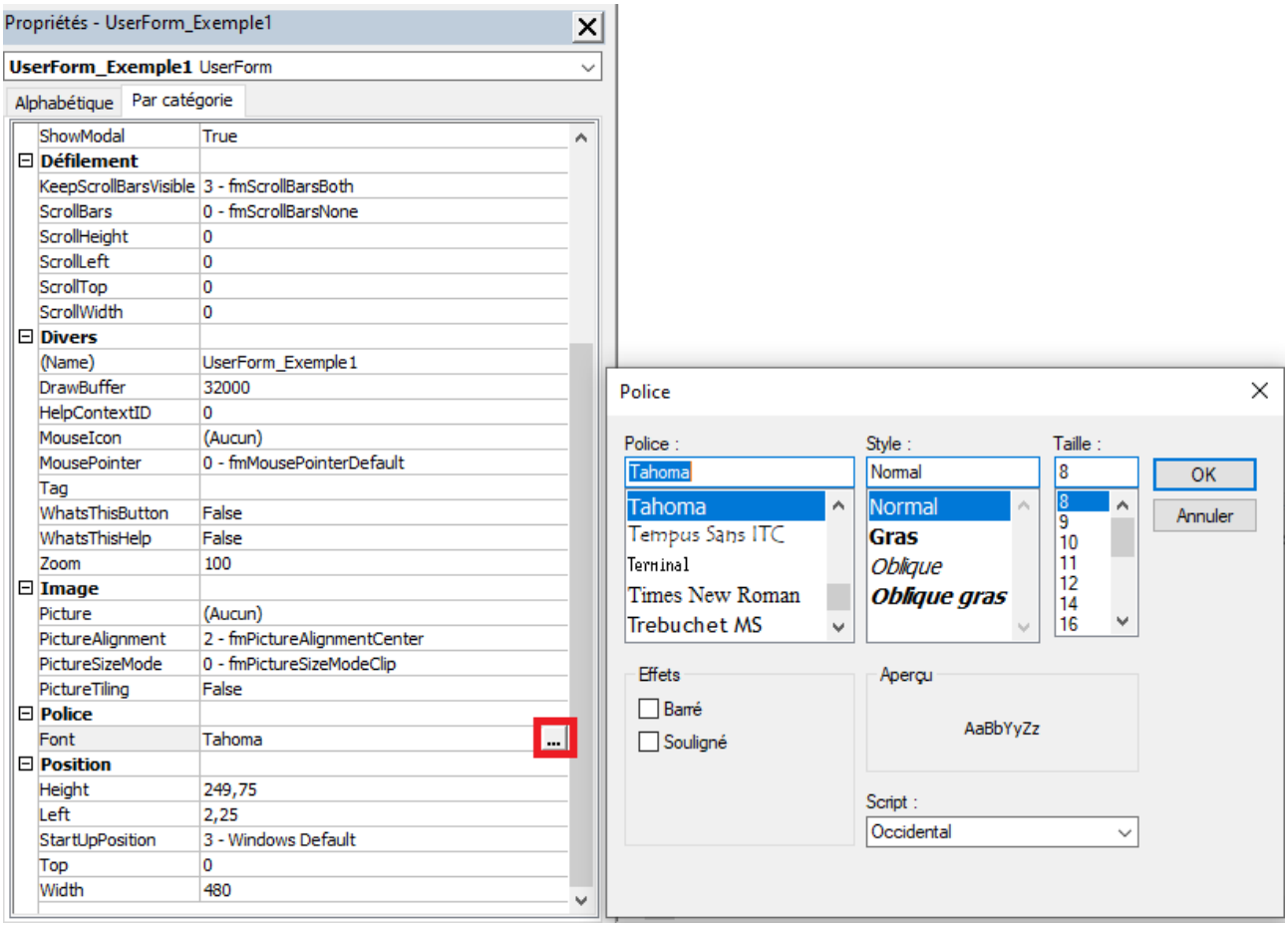


FIGURE 2.8. – Fenêtre de police.

Il y a même une zone d'aperçu !

La couleur de texte est à part, dans la propriété `ForeColor`.

2.3.3. Bordure

Par défaut, la fenêtre n'a pas de bordure puisque sa propriété `BorderStyle` contient la valeur `fmBorderStyleNone`. Si nous souhaitons en mettre, il faut passer cette valeur à `fmBorderStyleSingle` et éventuellement choisir une couleur dans `BorderColor`.

2.3.4. Titre

`Caption` sert à renseigner le titre de la fenêtre.

2.4. Comportement

2.4.1. Affichage modal ou non

La propriété `ShowModal` contient un booléen pour dire si la fenêtre doit être modale ou non. Si la fenêtre est modale, les interactions sont limitées à celle-ci (l'utilisateur ne peut pas interagir avec l'application ou un autre UserForm par exemple).

2.4.2. Activation et désactivation

Mettre à faux la propriété `Enabled` a pour effet de rendre impossible les interactions avec l'objet.

> *i* Certaines propriétés peuvent également être lues ou écrites en VBA. Par exemple, pendant l'événement d'initialisation (`Initialize`) que nous étudierons plus tard.

Il y a encore d'autres propriétés que nous ne présenterons pas ici, mais qui peuvent être utiles suivant les cas (image de fond, défilement, aide, …).

Cette section a présenté le paramétrage d'une boîte de dialogue personnalisée. Nous y avons vu les principales propriétés à renseigner.

3. Exécution

Il est temps de donner vie à notre boîte de dialogue personnalisée en l'exécutant.

3.1. Via l'éditeur

Lorsque nous nous trouvons sur un UserForm, nous pouvons l'exécuter et l'arrêter au moyen des boutons de gestion d'exécution du VBE :

FIGURE 3.9. – Gestion exécution via les boutons.

3.2. Via du code

Par ailleurs – et heureusement – il existe différentes instructions et méthodes pour gérer le lancement d'un UserForm. Cela nous permet de déclencher une boîte de dialogue personnalisée en cliquant sur un bouton, depuis un autre UserForm ou encore depuis la barre d'accès rapide par exemple.

3.2.1. Charger/Décharger

Certaines instructions permettent de mieux gérer la vie d'un UserForm, mais ne sont pas forcément nécessaires.

L'instruction `Load` permet de charger un objet donné en mémoire. Son événement d'initialisation (`Initialize`) est alors déclenché. Il n'est pas rendu visible pour autant.

L'instruction `Unload`, à l'opposé, sert à supprimer un objet donné de la mémoire et ainsi la libérer : `Unload UserForm1`. Son événement de destruction (`Terminate`) est déclenché. Par conséquent, il disparait. C'est ce qu'il se passe lors du clique sur la croix pour fermer la fenêtre.

3.2.2. Afficher/Masquer

Nous pouvons afficher une boîte de dialogue personnalisée grâce à sa méthode `Show`. Si la fenêtre n'a pas été chargée via `Load`, elle l'est alors implicitement. Il est possible de passer un paramètre optionnel afin de choisir si la fenêtre doit être modale ou non (respectivement `vbModal` ou `vbModeless`). Si renseigné, ce paramètre prendra le dessus sur la propriété `ShowModal` étudiée auparavant.

À l'inverse, nous pouvons la masquer grâce à sa méthode `Hide`.

Un UserForm masqué n'est pas déchargé de la mémoire. Autrement dit, il reste accessible par programmation (récupération de valeur, changement de propriété, …).

Mettons cela en application en ajoutant un bouton de commande ActiveX dans une feuille qui nous permet d'afficher la boîte de dialogue lorsque nous cliquons dessus :

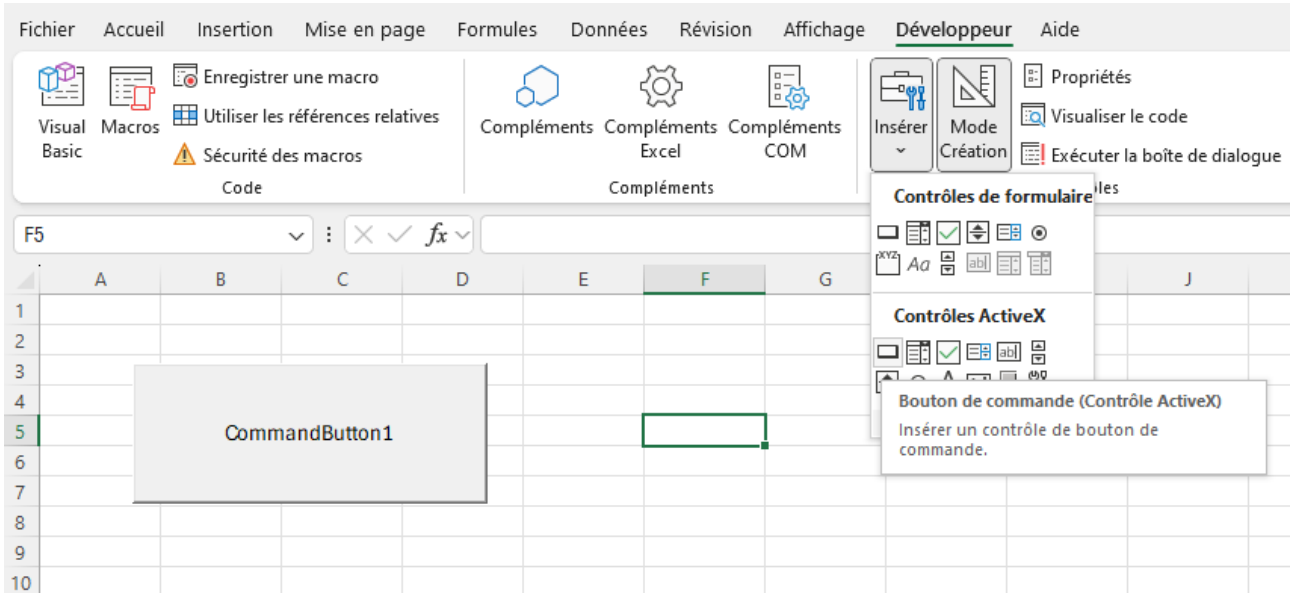


FIGURE 3.10. – Ajout bouton de commande ActiveX.

```vba
Private Sub CommandButton1_Click()
    UserForm_Exemple1.Show
End Sub
```

Pour définir `CommandButton1_Click`, le plus simple est de double-cliquer sur le bouton en mode création afin d'accéder directement à l'événement.

Durant cette section, nous avons appris à exécuter nos boîtes de dialogue personnalisées.

Conclusion

Au cours de ce premier billet, nous avons vu comment ajouter, paramétrer et exécuter nos premières boîtes de dialogue personnalisées.

À bientôt !