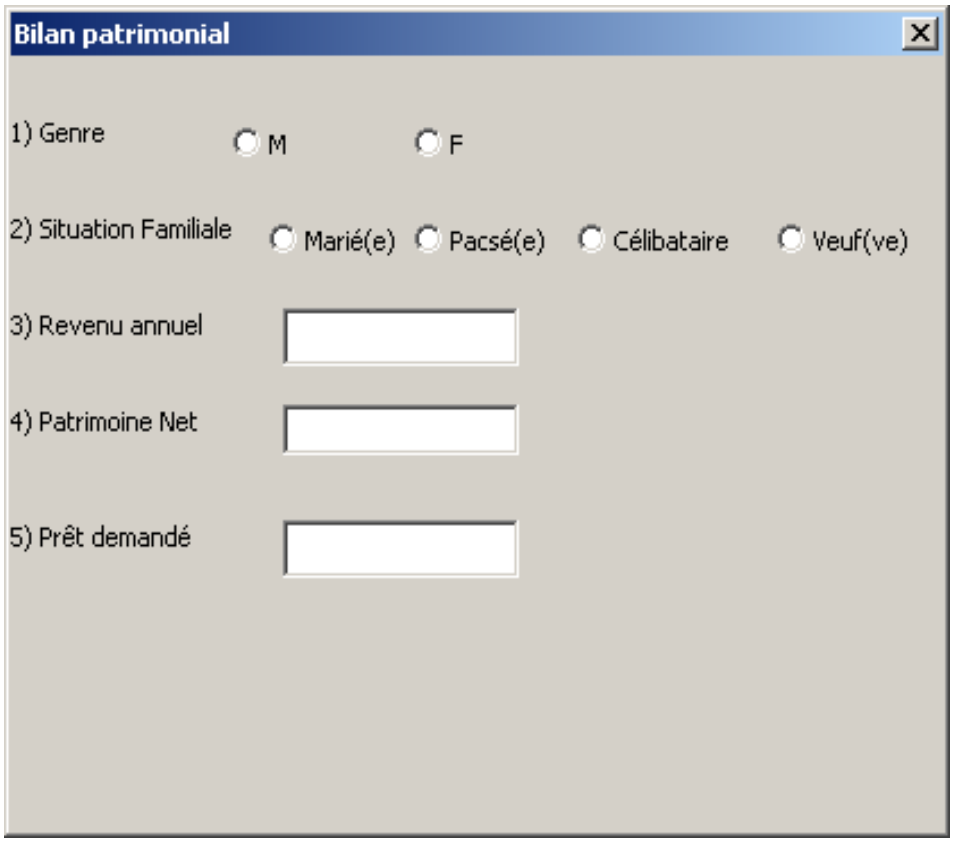
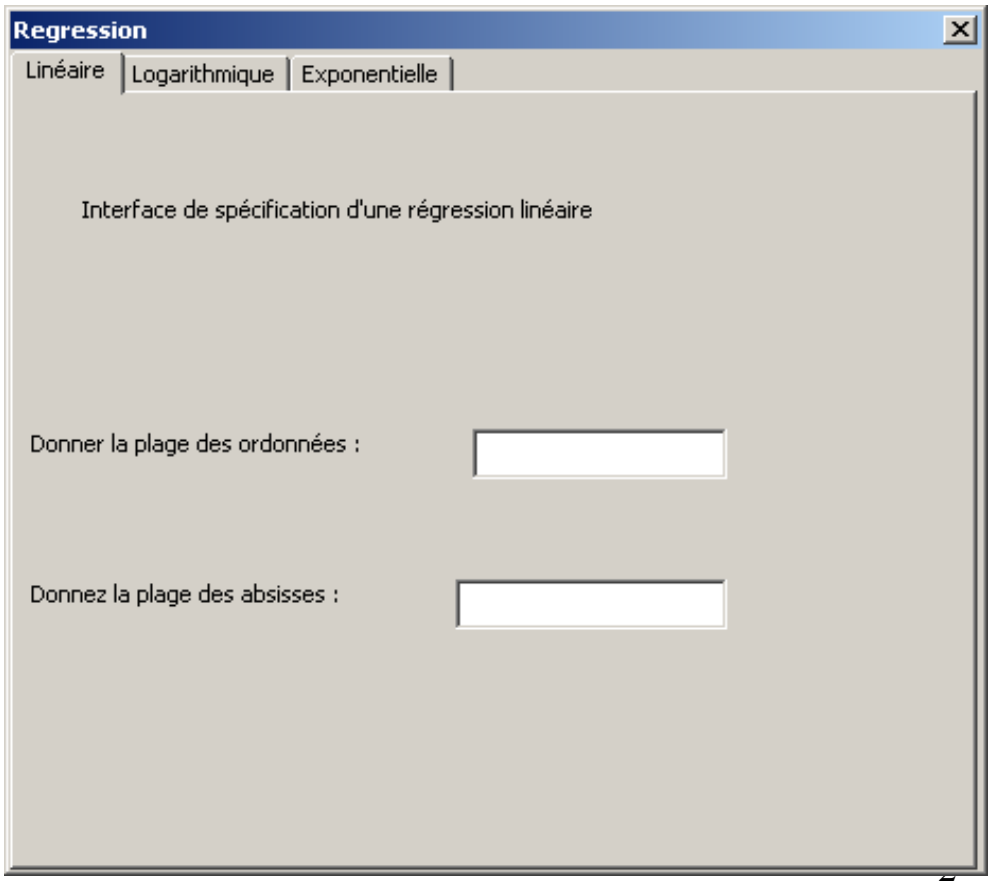
Formation VBA 3 – Interagir

MCours.com

Utilisation des UserForms

- Les UserForms sont des interfaces largement configurables, il convient de ne pas les limiter à tel ou tel usage qui pourrait être présenté à titre d'exemple.





Utilisation des UserForms (2)

- Programmation évenementielle
- Possibilité de contrôles des données
- Possibilités de modification dynamique des UserForms.
- Possibilité de manipulation de l'ensemble de l'interface via les UserForms

=> Possibilités considérables.

Contenu

- **1) Création et dessins des UserForms**

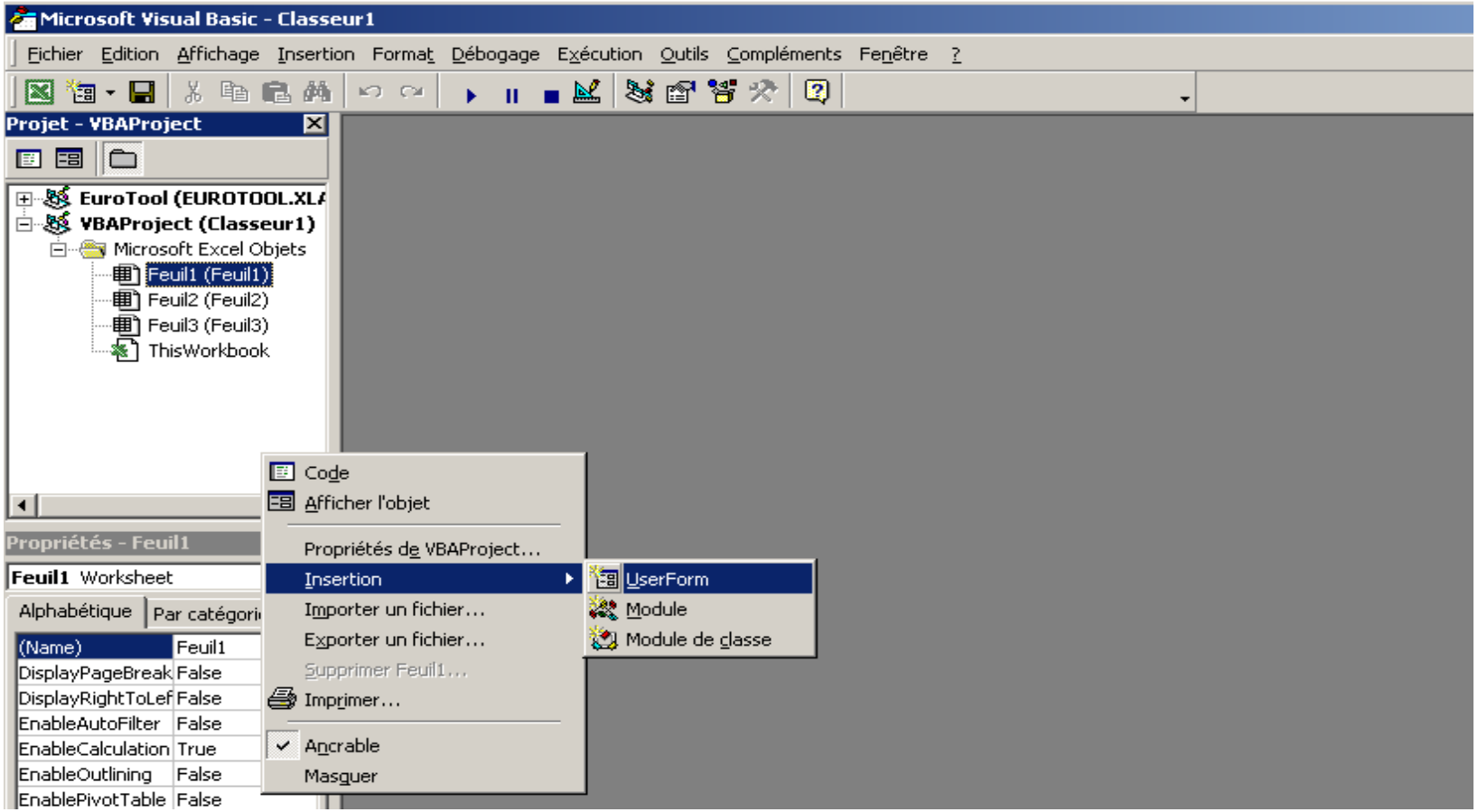
- 2) Comportement des UserForms
  - Affichage des UserForms
  - Comportement des UserForms : écriture du code associé
  - Intégration dans la hiérarchie et manipulation des widgets

- 3) Boîte de dialogue : d'autres widgets mais pré-implémentés

- 4) Programmation évenementielle
  - Aspects théoriques
  - Exemples d'implémentation
  - Développement de quelques éléments de la hiérarchie

Création et dessins des UserForms

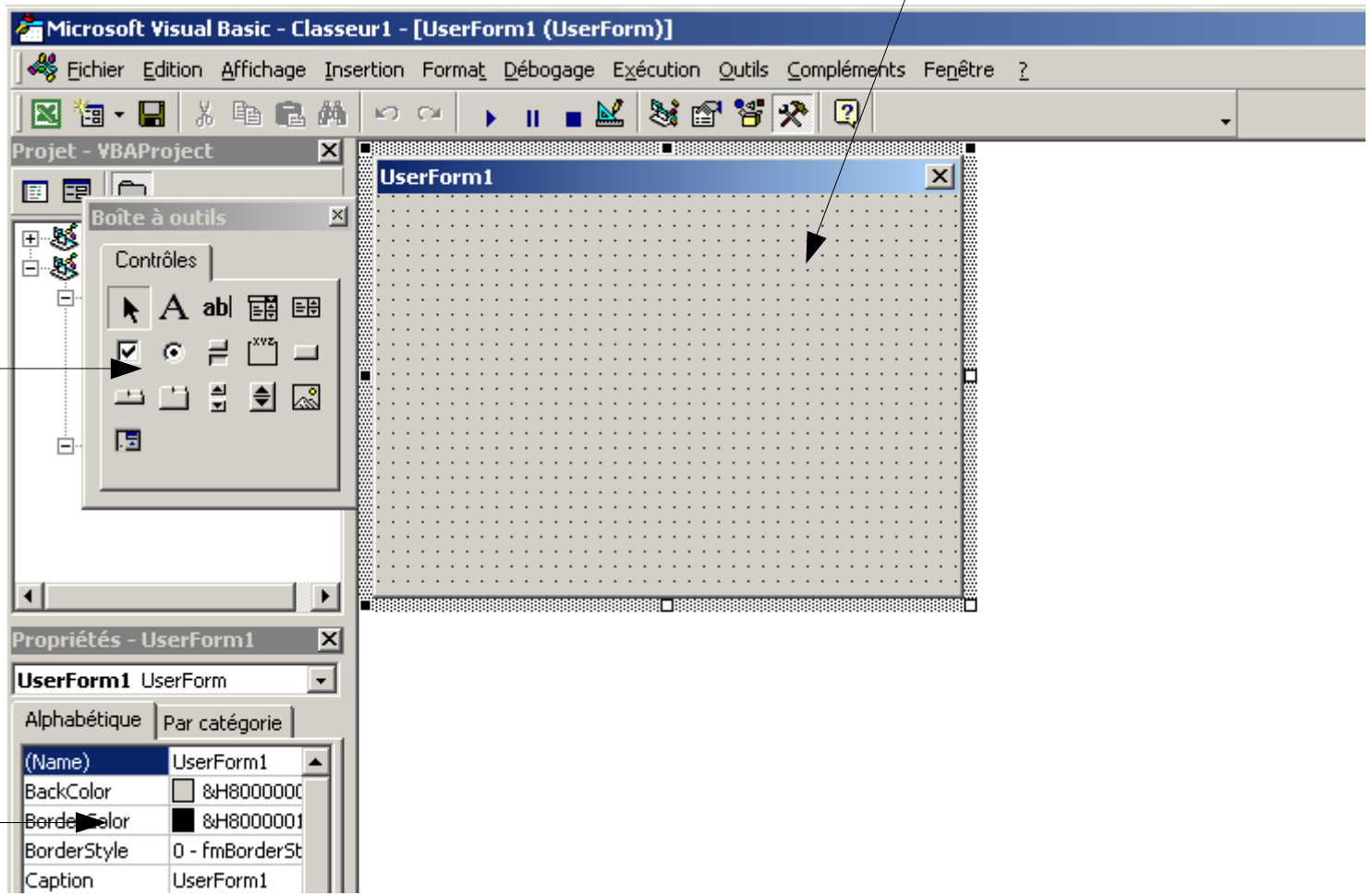
- Les userForms sont des interfaces qu'il est possible de construire à partir d'une boîte d'outils de base.
- La création d'un UserForm, clic droit dans l'explorateur de projet :



Dessins sur un UserForm

- Les userForms sont des interfaces qu'il est possible de construire à partir d'une boîte d'outils de base.

Espace de dessin



Elements de construction de l'interface / widget

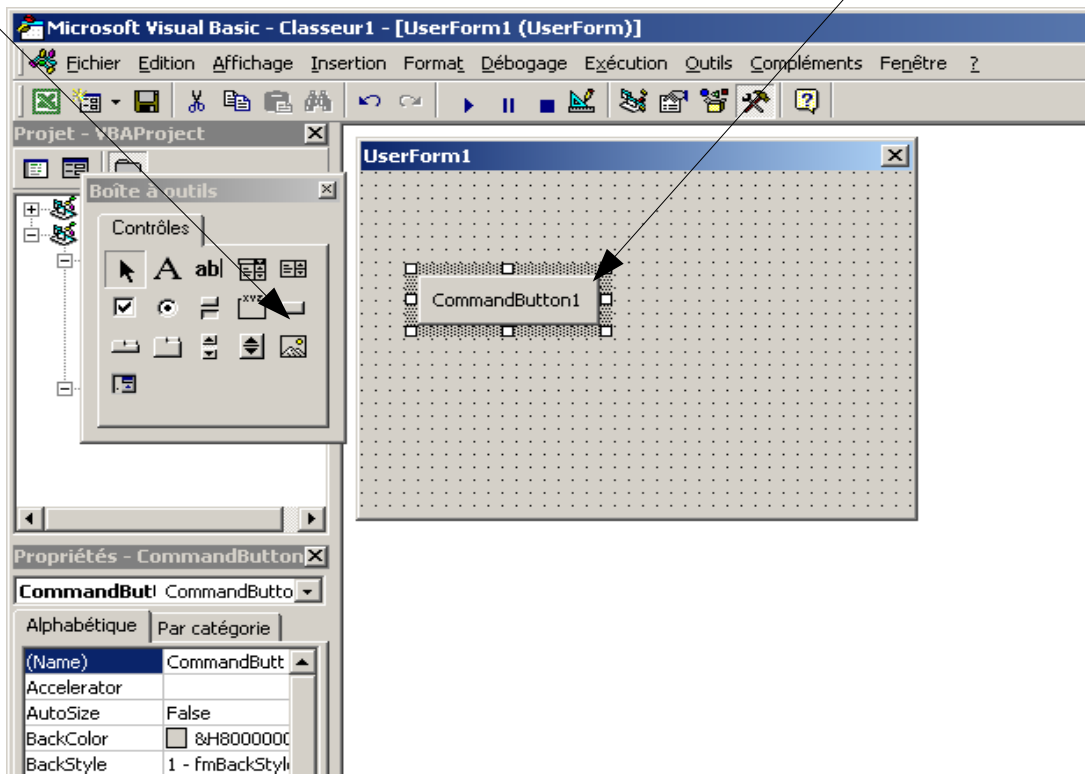
Propriété de l'objet UserForm

Dessins sur un UserForm - bouton

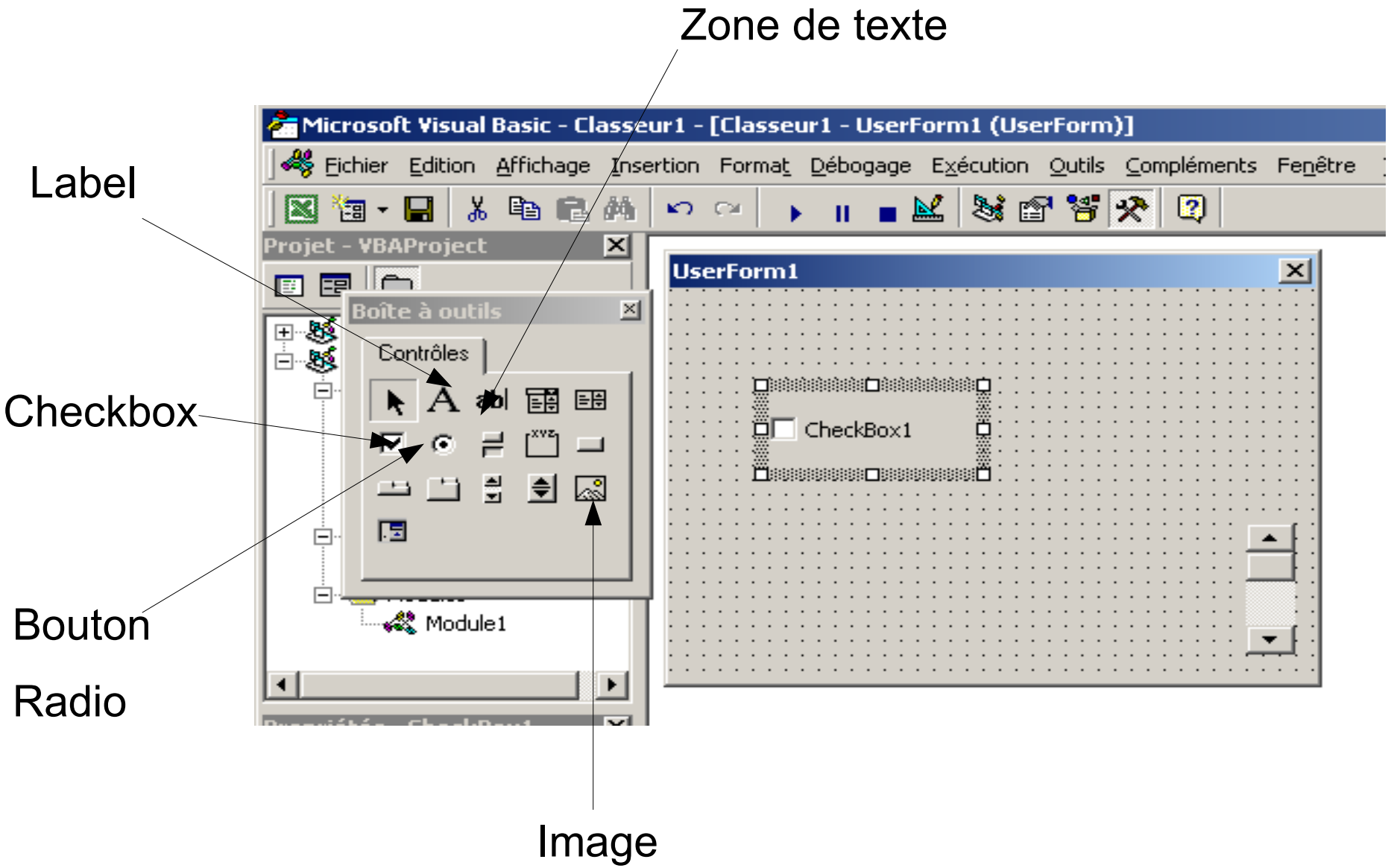
- Bouton :

1- Selection du widget-bouton

2- Dessin du widget bouton



Différents Widgets (ou controles)



Contenu

- 1) Création et dessins des UserForms

- **2) Comportement des UserForms**
  - **Affichage des UserForms**
  - **Comportement des UserForms : écriture du code associé**
  - **Intégration dans la hiérarchie et manipulation des widgets**

- 3) Boîte de dialogue : d'autres widgets mais pré-implémentés

- 4) Programmation évenementielle
  - Aspects théoriques
  - Exemples d'implémentation
  - Développement de quelques éléments de la hiérarchie

Affichage des UserForms

- La création d'un UserForm ajoute un objet dans la hiérarchie.
- L'affichage est une procédure associée. En première approche, pour afficher le UserForm, on fait une procédure d'un module, d'un classeur ou d'une feuille de calcul. Soit par exemple le rajout d'un UserForm "UserForm1", son affichage :

```
Sub main()

UserForm1.show

End Sub
```

ou :

```
Sub main()

Call UserForm1.show()

End Sub
```

- Dans ce qui suit, on verra une manière plus élegante d'afficher les UserForms (voir la partie sur la programmation évenementielle)

Comportement des UserForm : écriture du code associé

- A partir de la définition de l'UserForm dans VBE, on accède à une interface de définition du code via *clic droit sur le UserForm - code*
- OU BIEN : en cliquant deux fois sur l'un des widgets qui a été défini : dans ce cas, apparaît une procédure possible pour le widget.
- La forme des procédures sur la page associée au UserForm : *nomObjet_evenementSurObjet()*. Quelques exemples :

```
Private Sub Frame1_Click()

Private Sub CheckBox1_Click()

Private Sub Label1_Click()
```

- Ces procédures sont déclarées private. Elles sont activées lors des actions de l'utilisateur sur l'interface.
- => PROGRAMMATION EVENEMENTIELLE

Comportement des UserForm : premiers tests

- On définit un UserForm avec un label qui sera automatiquement nommé Label1 du UserForm et une zone de texte qui sera automatiquement définie comme textBox1_change. En double cliquant les objets, on fait apparaître des procédures liées :

```
Private Sub Label1_Click()
MsgBox "Label1"
End Sub

Private Sub textBox1_change()
MsgBox "textBox1"
End Sub
```

- Il y a réaction et activation du code dès que l'utilisateur rentre une valeur dans le champ texte ou dès qu'il clique sur le label.

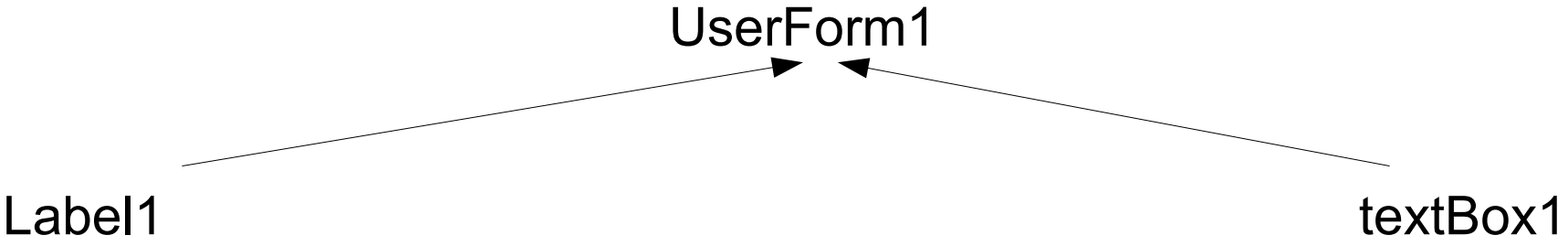
Comportement des UserForm :

- Pour chaque élément rajouté au widget : il dispose de propriétés et de procédures / fonctions associées. Les voir en cours de frappe pour Label1 dans l'exemple précédent par exemple (aide à la saisie de VBE).
- Pour chaque clic sur le label, on peut provoquer un changement de couleur en utilisant la propriété BackColor de Label1:

```
Private Sub Label1_Click()

Randomize

Label1.BackColor = RGB(255 * Rnd, 255 * Rnd, 255 * Rnd)

End Sub
```

Le userForm est une hiérarchie d'objets

- En créant un UserForm, on crée un nouvel arbre d'objets :



- Notamment, le chemin d'accès complet des objets-widgets du UserForm :
  - UserForm1.Label1
  - UserForm1.textBox1

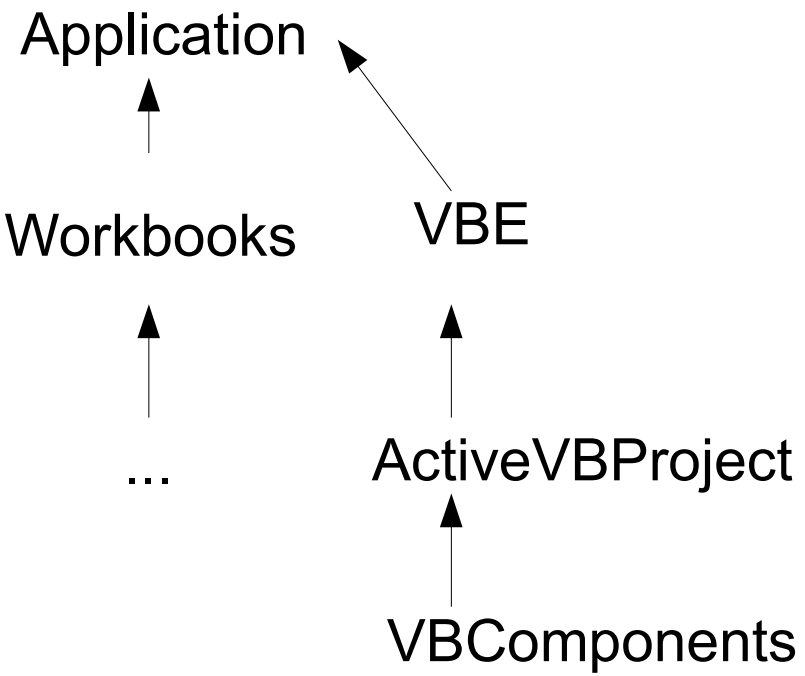
Le userForm est une hiérarchie d'objets (2)

- Conséquences pour l'accès aux variables et aux procédures/fonctions
- Un exemple :

```
Sub modifUF()
UserForm1.Label1.BackColor = RGB(0, 0, 255)
UserForm1.Show
End Sub
```

- Cette instruction peut être spécifiée sur la page de code associée au userForm ou sur un module autre

Le UserForm est lui même un objet d'une hiérarchie



- **Créer un UserForm c'est rajouter une hiérarchie d'arbre dans un arbre. Cf la définition de l'arbre et le maintien des propriétés.**

Manipulation des UserForms via le code (1)

- L'exécution des codes suivants nécessite l'activation de plusieurs librairies : *Outils-References* et activation de :
  - Microsoft Visual Basic Édition Applications,
  - Microsoft Forms 2.0

  (le numéro ou le nom des bibliothèques peut être légérement différent en fonction de la version)

- Ajout automatique de UserForm :

```
Sub test()
Application.VBE.ActiveVBProject.VBComponents.Add (vbext_ct_MSForm)
End Sub
```

Manipulation des UserForms via le code (2)

```
Sub Add_Form2()

    ' Declare a variable to hold the UserForm.
    Dim mynewform As Object

    ' Create a new UserForm. You can now use this new VBComponent object
    ' to manipulate the User Form.
    Set mynewform = Application.VBE.ActiveVBProject.VBComponents.Add(vbext_ct_MSForm)

    With mynewform
       .Properties("Height") = 246
       .Properties("Width") = 616
       .Name = "HelloWord"
       .Properties("Caption") = "This is a test"
    End With

  End Sub
```

Manipulation des UserForms via le code (3)

- Accès au projet :

```
Application.VBE.ActiveVBProject.VBComponents.Add(vbext_ct_MSForm)
```

- Accès et lancement de l'interface :

```
Sub lancementUSF()
VBA.UserForms.Add("UserForm1").Show
End Sub
```

- La différence est importante mais pas forcément à faire immédiatement. En première approche, on se contente de manipuler les UserForms via l'interface VBE.

Propriétés des UserForms

- Sur un UserForm :

BackColor	Spécifie la couleur de fond
BorderColor	Spécifie la couleur de bordure
Caption	Spécifie le texte affiché dans la barre de titre
Forecolor	Définit la couleur de la police
Height	Définit la dimension verticale
Left	Définit la position par rapport au bord gauche de l'application
MouseIcon	Affecte un icône personnalisé
MousePointer	Spécifie le type de pointeur
Picture	Spécifie l'image de fond dans l'UserForm
ScrollBars	Indiquee si les barres de défilement verticales et horizontales doivent être affichées
StartUpPosition	Indique la position du UserForm lors de sa première apparition
Tag	Permet de stocker des informations supplémentaires
Top	Définit la position par rapport au bord supérieur de l'application
Width	Définit la dimension horizontale
Visible	Spécifie si l'objet est masqué ou affiché.

- **Différence entre la modification de l'interface dans Excel et la modification du projet.**

Evolution des UserForms

- Le fait de saisir F5 lorsque le UserForm est selectionné ou de lancer une procédure du type :

```
Private Sub CommandButton1_Click()

    UserForm1.Show

End Sub
```

conduisent à l'affichage du UserForm comme interface, bloquant l'accès aux menus standard ou empechant de saisir ou modifier du code dans l'interface VBE. L'userForm est dit **modal**.

Evolution des UserForms (2)

- Cacher un UserForm en conservant les modification, les valeurs pour les différents Widget, les modifications de couleurs sur l'interface etc...

```
UserForm1.Hide
```

- Pour fermer l'UserForm en libérant les ressources associées et en perdant les valeurs courantes des champs textes etc... :

```
unload UserForm1
```

Les évenements gérés par les UserForms

- La gestion des évenemens se fait Par la déclaration d'une procédure :
  - private
  - dans la feuille de code associée au userForm.
  - de la forme *NomObjet_Evement*
- Le code de la procédure sera lancé au moment de l'évenement.
- Par exemple :

```
Private Sub UserForm_Initialize()
    UserForm2.BackColor = RGB(0, 0, 255)
End Sub
```

Page de code / UserForm

- Accès en double-cliquant sur le userForm
- Accès avec *clic-droit sur le userForm et code*
- Les différents Widgets sont accessibles entre eux.
- Accessibilité de l'ensemble du modèle objet non déclaré Private.

Les différents widgets : checkboxes

- Les attributs/propriétés :

BackColor	Couleur de fond derrière le texte
Caption	Texte à côté
Font	Forme du texte à côté (police, taille…)
Height	Hauteur en pixels
Locked	Le fait que la valeur est modifiable
Value	La valeur : {True (le widget est coché) /false}
Visible	Le widget apparaît

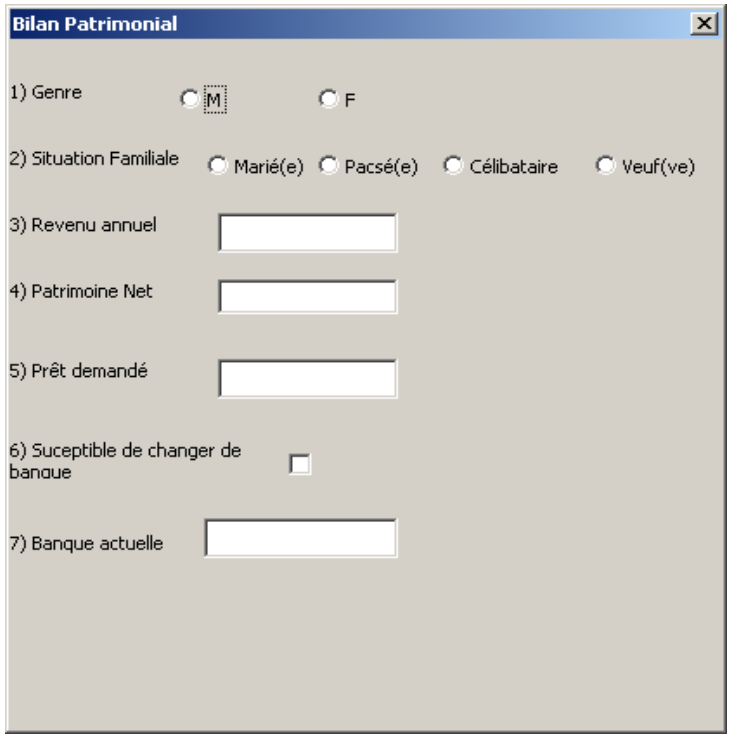
- Les méthodes / propriétés :

Move	On fait bouger le widget au sein du UserForm
setFocus	Le widget est selectionné au sein du userForm

- Comportement / évenements :

```
Private Sub CheckBox1_Click()
```

Checkbox : un exemple (fichier usForms.xls)



L'appel depuis un module :

```
Sub userFormQuestionnaire()
UserForm2.Label7.Visible = False
UserForm2.TextBox4.Visible = False
UserForm2.Show
End Sub
```

Le code associé à CheckBox1 :

```
Private Sub CheckBox1_Click()
    If CheckaBox1.Value = True Then
    Label7.Visible = True
    TextBox4.Visible = True
    Else
    Label7.Visible = False
    TextBox4.Visible = False
    End If
End Sub
```

Pourquoi le Else dans l'instruction conditionnelle ?

Paramètres des fonctions Move ou setFocus ?

Les différents widgets : bouton radio

- Les attributs/propriétés :

BackColor	Couleur de fond derrière le texte
Caption	Texte à côté
Cancel	Le bouton peut être déselectionné
Font	Forme du texte à côté (police, taille…)
Height	Hauteur en pixels
Locked	Le fait que la valeur est modifiable
Value	La valeur : {True (le widget est coché) /false}
Visible	Le widget apparaît

- Les méthodes / propriétés :

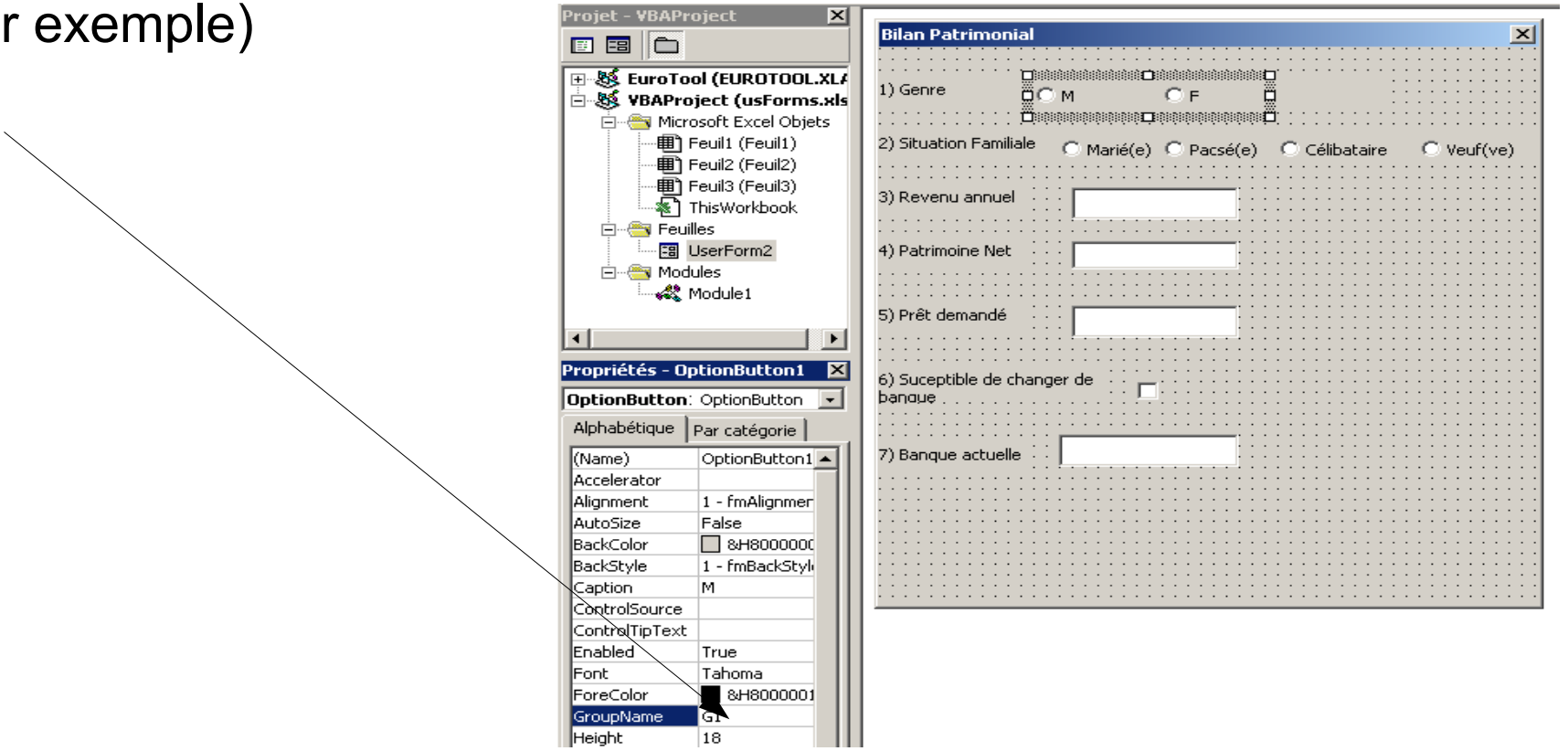
Move	On fait bouger le widget au sein du UserForm
setFocus	Le widget est selectionné au sein du userForm

- Comportement / évenements :

```
Private Sub OptionButton1_Click()
```

Exemple d'utilisation

- Spécifié tel quel le fichier usForms.xls présente une limite considérable : le fait de cliquer sur un bouton radio exclue de pouvoir cliquer sur un autre.
- Pour régler ce problème, on rattache les boutons radios par groupes : (G1 pour le genre et G2 pour la situation familiale par exemple)



- Modification directe des propriétés !
- La machine fait directement un contrôle : pourrait-on penser le type de contrôle qu'elle effectue.

Les différents widgets : textBox

- Les attributs/propriétés :

BackColor	Couleur de fond derrière le texte
Caption	Texte à côté
Cancel	Le bouton peut être déselectionné
Font	Forme du texte à côté (police, taille...)
GroupName	Un attribut de type chaîne qui rattache le bouton à un groupe (voir l'exemple)
Height	Hauteur en pixels
MaxLength	Le nombre de caractère maximal admis
Locked	Le fait que la valeur est modifiable
Value	La valeur : {True (le widget est coché) /false}
Visible	Le widget apparaît

- Les méthodes / propriétés :

Copy	Copie le contenu dans le presse papier
Cut	Elimine le contenu et le copie dans le presse papier
Move	On fait bouger le widget au sein du UserForm
Paste	Colle le contenu du Presse Papier
setFocus	Le widget est selectionné au sein du userForm

- Comportement / évenements :

Private Sub TextBox1_Change()

textBox : forcer un contenu numérique

- Comment forcer que l'utilisateur rentre un contenu numérique ?
- Dans usForms.xls, on veut forcer un contenu numérique pour le champ recevant le revenu par exemple :

```
Private Sub TextBox1_Change()
    If Not IsNumeric(TextBox1.Value) Then
    TextBox1.Value = Mid(TextBox1.Value, 1, Len(TextBox1.Value) - 1)
    End If
End Sub
```

Contenu

- 1) Création et dessins des UserForms

- 2) Comportement des UserForms
  - Affichage des UserForms
  - Comportement des UserForms : écriture du code associé
  - Intégration dans la hiérarchie et manipulation des widgets

- **3) Boîte de dialogue : d'autres widgets mais pré-implémentés**

- 4) Programmation évenementielle
  - Aspects théoriques
  - Exemples d'implémentation
  - Développement de quelques éléments de la hiérarchie

Les widgets pré-implémentés

- Dans ce qui précède, on a constaté que les fonctions ou les procédures VBA permettent de manipuler les MsgBoxes et les boutons de commandes de manière automatique.
- L'affichage MsgBox tel que pratiqué jusqu'à maintenant, correspond à une fonction. L'instruction habituellement utilisée :

```
MsgBox "Valeur de x " & x
```

- La création de bouton passe par une fonction qui renvoie une référence sur le bouton crée :

```
ActiveSheet.OLEObjects.Add(ClassType:="Forms.CommandButton.1",_
    Link:=False,  DisplayAsIcon:=False,  Left:=201.75,  Top:=172.5,  Width:=77.25,_
    Height :=29.25)
```

Les widgets pré-implémentés : inputBox

- Note : à l'instar de la MsgBox, ou des UserForms, lorsque une InputBox est active, elle bloque l'accès aux menus et la saisie de code dans l'interface VBE.
- Les InputBoxes permettent une interaction plus riche que ne le permettent les MsgBoxes.
- Alors que l'appel à une MsgBox est l'appel d'une procédure, l'appel à une InputBox est un appel de fonction

La forme générale de la création d'une Inputbox

InputBox(Message,[Title],[Default],[xPos],[xPos]) As String

- Message est la chaine affichée dans la boîte de dialogue
- Title est le titre éventuel
- Default est la valeur par défaut dans le champ de saisie
- xPos : position de l'affichage au sein de la fenêtre
- yPos : position de l'affichage au sein de la fenêtre

inputBox : Quelques exemples d'utilisation

- Dans le cas des InputBox, on attend que l'utilisateur renvoie une réponse :

```
Sub age()
Dim str As String
str = InputBox("Donnez votre age : ", "Age", , 10, 10)
   While (Not IsNumeric(str))
   str = InputBox("Donnez une valeur numérique pour votre age : ", "Age", , 10, 10)
   Wend
MsgBox "Age saisi : " & str
End Sub
```

Les widgets pré-implémentés : MsgBox et forme plus générale

- La forme générale de l'utilisation des MsgBox :

MsgBox (Message, [typeBouton + typeIcone], [titre]) As vbMsgBoxResult

TypeBouton renvoie à des possibilités d'organisation des boutons de réponse

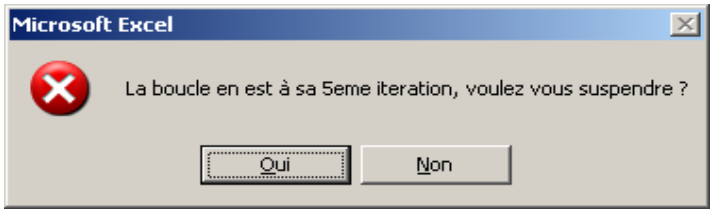
TypeIcone renvoie au type d'icone qui illustre la boîte

Titre est le titre donné à la boîte.

Les widgets pré-implémentés : Quelques exemples

```
Sub testMsg1()
Dim str As String
str = MsgBox("Voulez vous sauvegarder avant de quitter l'application ?", vbYesNo, "Sauvegarde ?")
End Sub
```


```
Sub testMsg2()
Dim retour As Integer, i As Integer
    For i = 0 To 100
    retour = MsgBox("La boucle en est à sa " & i & "eme iteration, voulez vous suspendre ?", _
    vbYesNo + vbCritical)
        If retour = 6 Then
        i = 10001
        End If
    Next
End Sub
```



Les valeurs possibles

- Les types d'ensemble de boutons possibles :

La référence du type de bouton	La valeur associée
vbOkOnly	OK
vbOKCancel	Ok Annuler
vbAbortRetryIgnore	Abandonner Recommencer Ignorer
vbYesNoCancel	Oui Non Annuler
vbYesNo	Oui Non
vbRetryCancel	Répeter Annuler
vbOKCancel	Affichage bouton aide

- Les valeurs de retour et leurs équivalents en entier :

Nom du bouton	Valeur renvoyée lorsque cliqué
vbOK	1
vbCancel	2
vbAbor	3
vbRetry	4
vbIgnore	5
vbYes	6
vbNo	7

Contenu

- 1) Création et dessins des UserForms
- 2) Comportement des UserForms
  - Affichage des UserForms
  - Comportement des UserForms : écriture du code associé
  - Intégration dans la hiérarchie et manipulation des widgets
- 3) Boîte de dialogue : d'autres widgets mais pré-implémentés
- **4) Programmation évenementielle**
  - **Aspects théoriques**
  - **Exemples d'implémentation**
  - **Développement de quelques éléments de la hiérarchie**

4- Programmation évenementielle - Démarche

- Pour un certain nombre des éléments de la hiérarchie objet, il existe des comportements programmables et on peut en faire des écouteurs d'évements à l'instar de ce qui est fait pour les widgets des UserForms ou sur les UserForms.
  - un classeur peut être manipulé : il peut être ouvert, fermé, dit autrement, l'ouverture et la fermeture sont des évenements pour le classeur

  ...
- Le même schéma fonctionne que celui qui existe pour les widgets : un évenement est défini par le nom d'un objet, un underscore et le nom d'un évenmement, la syntaxe est la suivante : Workbook_open. Les conditions doivent être les mêmes :
  - La procédure doit être déclarée private
  - La procédure doit être saisie dans la feuille associée, par exemple, sur la feuille Workbook du projet pour que le comportement évenementiel soit pris en compte pour le classeur.
- Le comportement sera recherché automatiquement lors d'une opération.

4- Programmation évenementielle – Un premier exemple

Un exemple simple d'exécution à l'ouverture :

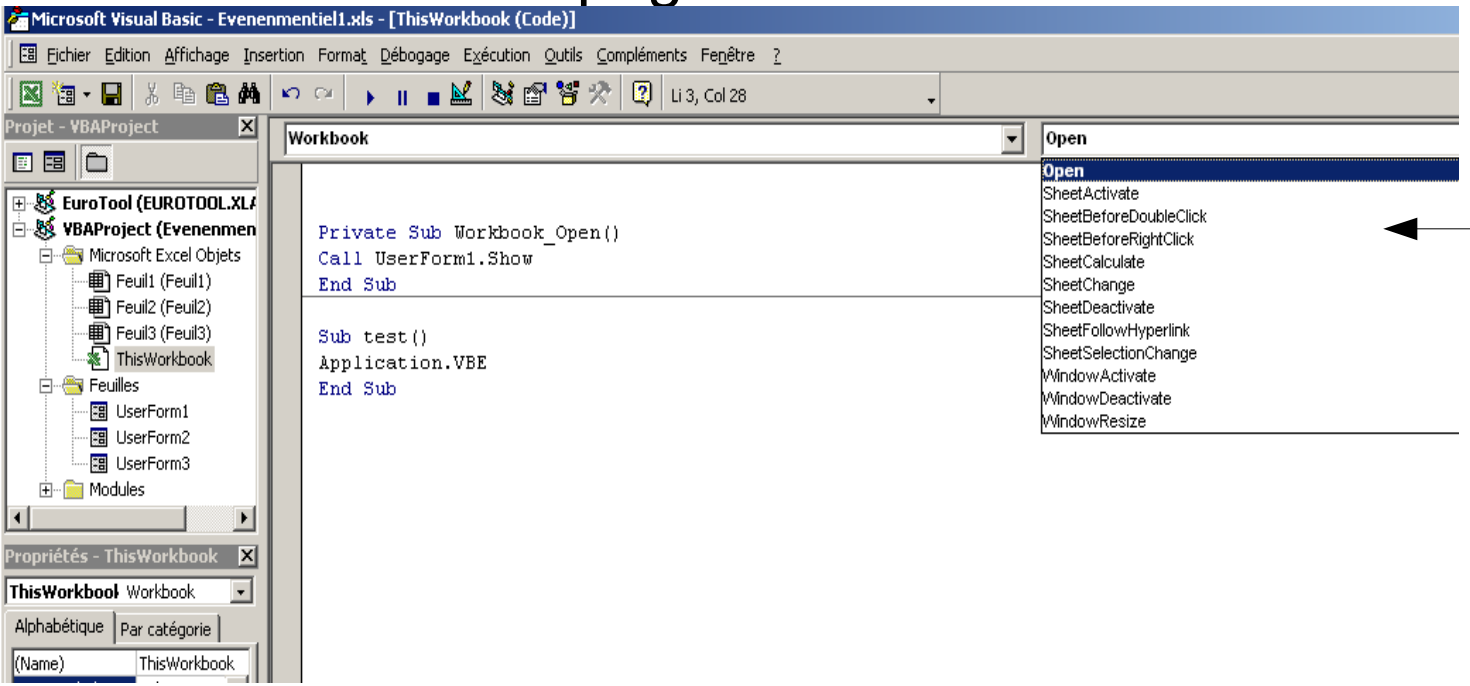
```
Private Sub Workbook_Open()
    MsgBox "Bonjour " & Environ("UserName")
End Sub
```

Activation d'un UserForm dès l'ouverture d'un classeur :

```
Private Sub Workbook_Open()
        userform1.show()
End Sub
```

4- Programmation évenementielle – VBE

- En allant sur la page associée au classeur – Workbook :



Liste des comportements

- Pour un objet donné, on ne peut pas définir plusieurs comportement pour le même évenement. Par contre, il n'y a pas de limite au nombre de procédures que peut appeler la fonction de réaction à un évenement :

```
Sub Workbook_open()

Call proc1()

Call proc2()

End Sub
```

4- Programmation évenementielle - Workbook

- Quelques évenements pour Workbook :
  - BeforeClose : effectué avant la fermeture du classeur
  - BeforeSave : effectué avant la sauvegarde : par exemple, vérification de la cohérence des données, interdiction de sauvegarde si certaines données ne sont pas cohérentes.
  - Open : comportement à l'ouverture du classeur.
  - NewSheet : appel de ce comportement au moment du rajout d'une nouvelle feuille.

- Les fonctions qu'il est possible de décrire :
  - Private Sub Workbook_BeforeClose
  - Private Sub Workbook_BeforeSave
  - Private Sub Workbook_Open
  - Private Sub Workbook_NewSheet

4- Programmation évenementielle - Worksheet

- Quelques évenements pour Worksheet :
  - Activate : effectuée lorsque la Worksheet est activée
  - Desactivate : effectuée lorssque la Worksheet est désactivée

CONCLUSION

- Pour chaque objet de la hiérarchie :
  - **il a des propriétés**
  - **il a des méthodes**
  - **il est éventuellement possible de définir un comportement de réaction aux évenement.**
- Construction d'interfaces évenementielles.
- Intégrer des userForms c'est développer l'arbre de la hiérarchie des objets et personnaliser l'interface et les possibiltés de la programmation évenementielle.
- Ensemble de possibilité considérables à très faible coût
- Par rapport aux autres modèles évenementiels ?