Aperçu général de
Visual Basic

Introduction intuitive à travers trois exemples (cf. TPs)

- Le programme bonjour
- Minimum de deux nombres
- Résolution d'une «équation du premier degré ».

Principes de bases

- Programmation événementielle
- Objets VB
- Architectures des applications événementielles
- Méthodes de développement d'applications VB

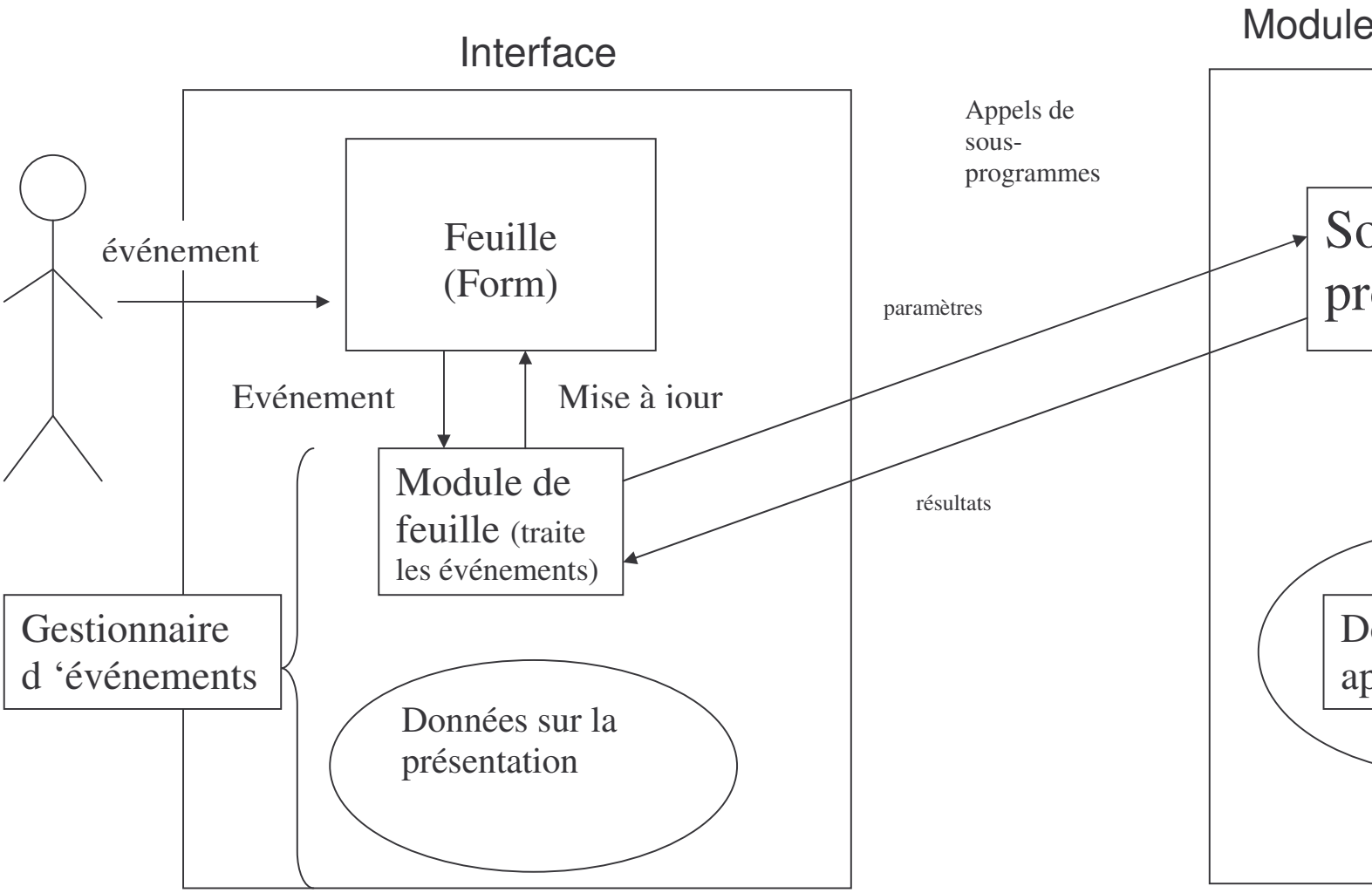
Le langage Visual Basic

- Structure de données
- Instructions
- Structure de contrôle

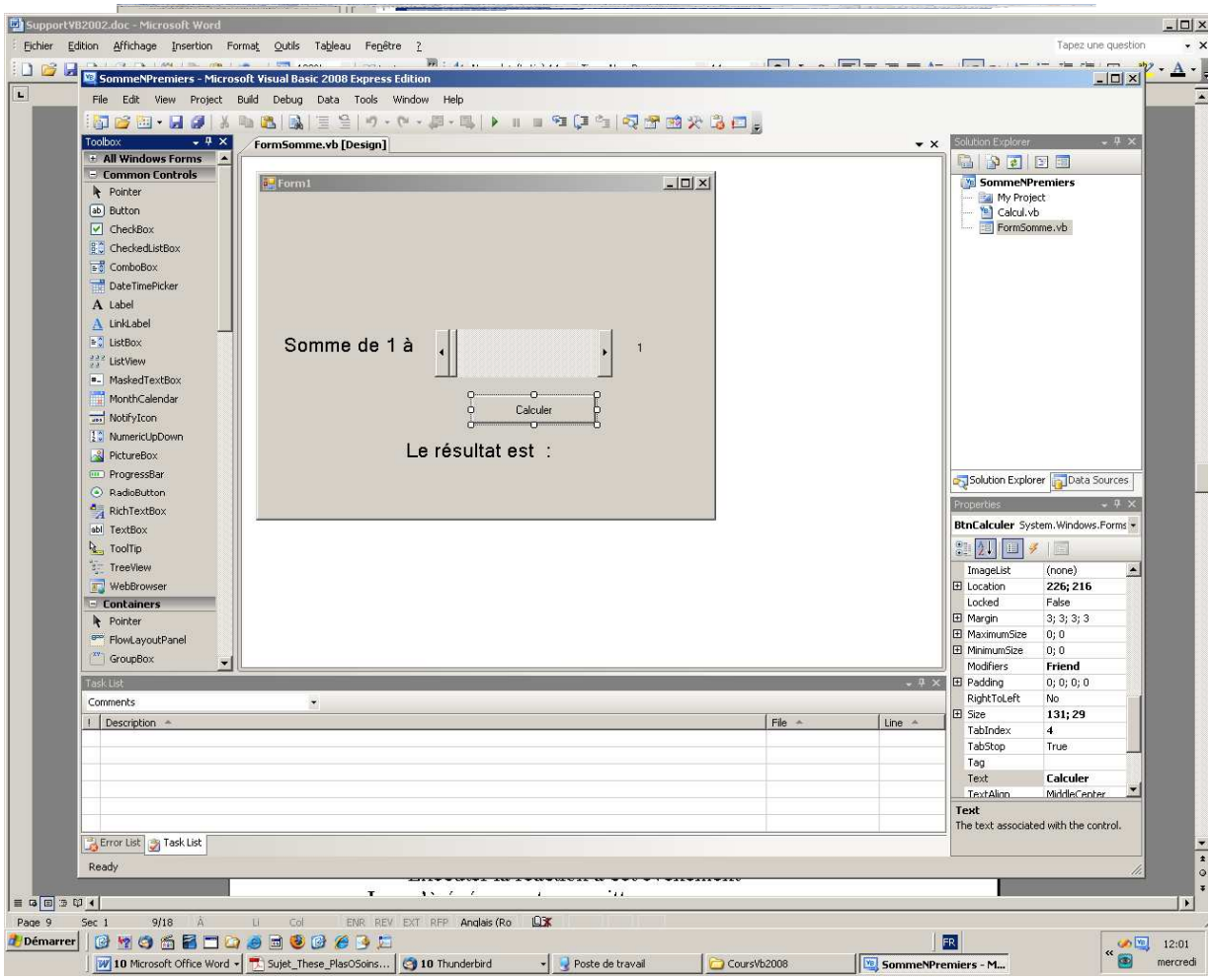
Feuilles, contrôles et menus (polycopié séparé)

- Introduction aux objets que vous pouvez assembler pour créer une application.

L'environnement VBNet (cf. TPs)



Architecture d'une application visuelle et événementielle « à la VB »


SupportVB2002.doc - Microsoft Word
SommeNPremiers - Microsoft Visual Basic 2008 Express Edition
FormSomme.vb [Design]
Form1
Somme de 1 à
1
Calculer
Le résultat est :
SommeNPremiers
My Project
Calcul.vb
FormSomme.vb
Calculer
Ready

Les contrôles de l'application

Nom du contrôle (name)	Type	Propriétés
LblSomme	Label	Text : Somme de 1 à
LblN	Label	Text : 1
ScrN	HScrollBar	Value : 1
TxtValeur2	TextBox	Text :
BtnCalculer	Button	Text : Calculer
LblResultat	Label	Text : le resultat est :

Réaction aux événements sur la forme

```
Public Class FormSomme

    Private Sub HSCN_Scroll(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.ScrollEventArgs) Handles HSCN.Scroll
        LblV.Text = Str(HSCN.Value)
    End Sub



      Private Sub BtnCalculer_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BtnCalculer.Click
        LblResultat.Text = "le résultat est : " & Str(somme(HSCN.Value))
    End Sub
End Class
```

Sous-Programme dans un Module

```
Module Calcul
    Public Function somme(ByVal n As Integer) As Long
        somme = (n * (n + 1)) / 2

    End Function
End Module
```

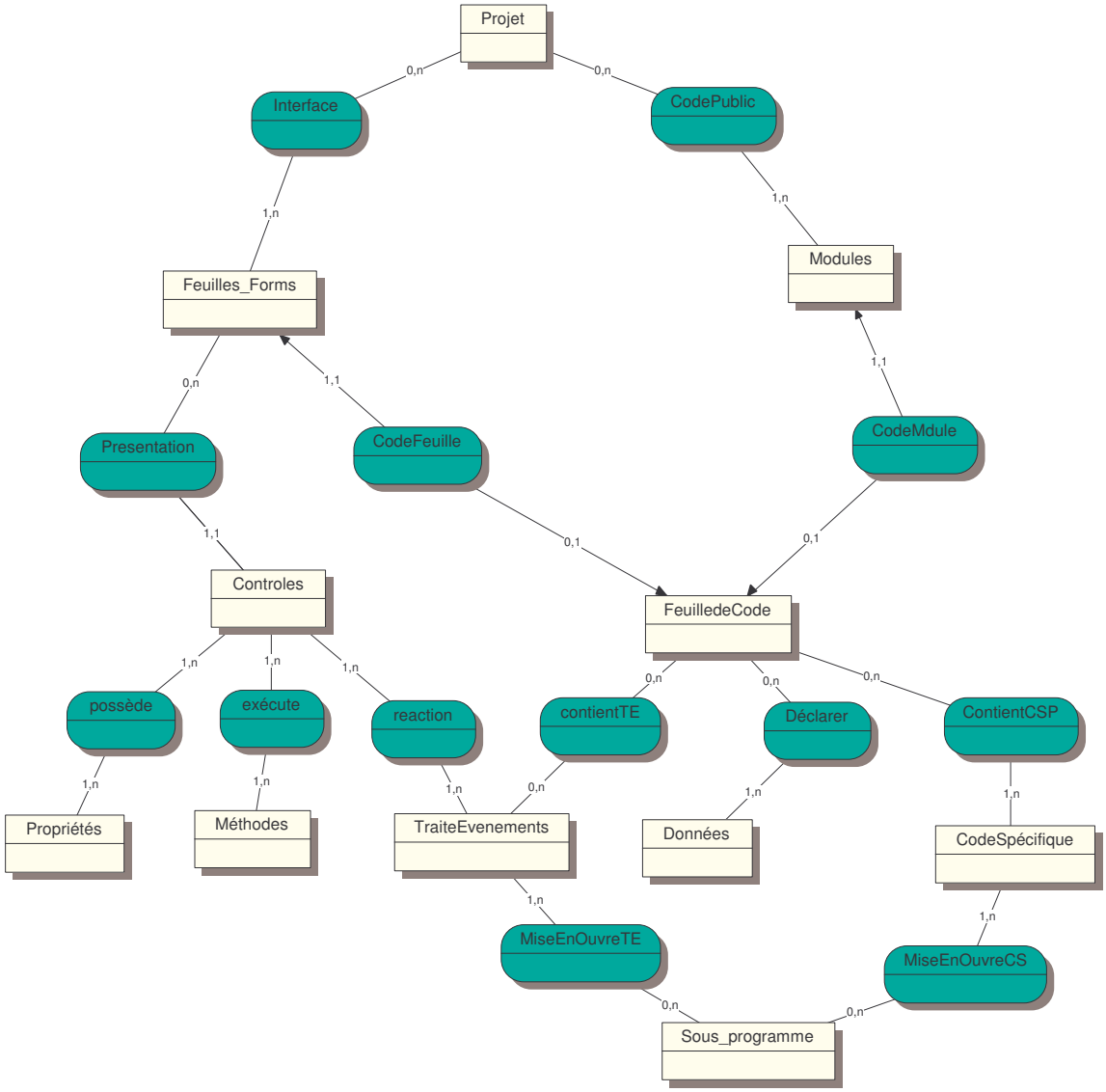
Programmation visuelle et événementielle

- Windev, Delphi, Power Objects et Visual Basic permettent de développer des applications avec interfaces graphiques.

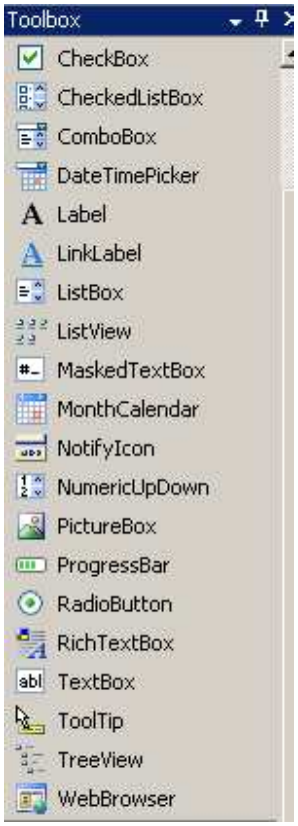
- Dans les applications traditionnelles, c'est l'application elle-même, et non un événement, qui contrôle les parties du code qui sont exécutées, ainsi que leur ordre d'exécution. Celle-ci commence à la première ligne de code et suit un chemin défini dans l'application, appelant les procédures au fur et à mesure des besoins.
- Dans une application événementielle, le code ne suit pas un chemin prédéterminé. Différentes sections du code sont exécutées en réaction aux événements. Ceux-ci peuvent être déclenchés par des actions de l'utilisateur, par des messages provenant du système ou d'autres applications, voire même par l'application proprement dite. L'ordre de ces événements détermine l'ordre d'exécution du code. Le chemin parcouru dans le code de l'application est donc différent à chaque exécution du programme.

Les objets sous Visual Basic

- Exemple : contrôles, feuilles, bases de données.
- Chaque objet est défini par une classe. Exemple : classe des CommandButtons.
- Chaque objet est une occurrence d'une classe.
- La classe définit l'ensemble des données et des traitements communs à tous les objets de la classe.
- Chaque objet est créé sous la forme d'une copie identique de sa classe et contient :
  - **Des propriétés** : données qui personnalisent l'objet (nom, titre, couleur, taille, visible) désignées par <*nom objet*>.*propriété*
  - **Des Traitements** : méthodes (fonctions ou procédures) pré-définies adaptées à l'objet (setfocus, refresh, hide, show, move, …) et désignées par <nom objet>.<nom de la méthode> [<paramètres>].
  - **Des événements** (actions) auxquels l'objet est sensible (glisser poser, click, …). Le code d'un événement est écrit par le concepteur. Désigné par <*nom objet*>_<*nom du type de l'événement*>. Si ce code n'est pas écrit, l'objet ne réagira pas à cet événement.


Projet
0,n
0,n
Interface
CodePublic
1,n
1,n
Feuilles_Forms
Modules
0,n
1,1
1,1
CodeFeuille
CodeMdule
Presentation
1,1
0,1
0,1
Controles
FeuilledeCode
1,n
1,n
1,n
0,n
0,n
0,n
possède
exécute
reaction
contientTE
Déclarer
ContientCSP
1,n
1,n
1,n
0,n
1,n
1,n
Propriétés
Méthodes
TraiteEvenements
Données
CodeSpécifique
1,n
1,n
MiseEnOuvreTE
MiseEnOuvreCS
0,n
0,n
Sous_programme

Les contrôles importants



Exécution d'une application

2 modes d'exécution pour une application :

-**compilé** : traduction en langage machine, génération d'une fichier .exe, exécutable depuis windows.

-**interprété** : dans l'environnement Vbwin, offre des possibilités de mise au point, attention de sauvegarder avant.

L'exécution d'une application :

Chargement de la feuille principale
L'événement load est exécuté
Répéter
    Attente d'un événement (sélection d'un menu, événement sur un contrôle, événement temporel)
    Exécuter la réaction à cet événement
Jusqu'à événement = « quitter »

Les types prédéfinis

Type	Taille (octet)	Domaine de valeurs
Boolean	*	True ou False (Vrai ou Faux)
Byte	1	0 à 255
SByte	1	-128 à 127
Short	2	-32 768 à 32 767
Integer	4	-2 147 483 648 à 2 147 483 647
Long	8	-9 223 372 036 854 775 808 à 9 223 372 036 854 775 807
Single	4	Valeurs négatives : -3,4028235E+38 à -1,401298E-45 Valeurs positives : 1,401298E-45 à 3,4028235E+38
Double	8	Valeurs négatives : -1,79769313486231570E+308 à – 4,94065645841246544E-324 Valeurs positives : 4,94065645841246544E-324 à 1,79769313486231570E+308
Decimal	16	Un nombre (positif ou négatif) à 28 chiffres ou un nombre à 28 chiffres après la virgule : De 0 à +/- 79 228 162 514 264 337 593 543 950 335 et de 0 à +/- 7,9228162514264337593543950335 (le plus petit nombre = +/- 0,0000000000000000000000000001)
Char	2	Un seul caractère unicode (65535 caractères différents)
String(x)	*	0 à approximativement 2 milliards de caractères (x est la taille demandée)
Date	8	00:00:00 (minuit) 1er janvier 0001 jusqu’à 11:59:59 PM 31 décembre 9999
Object	4 (processeur 32 bits) 8 sinon	Référence à une instance de type Object)

Déclarations et Portée des variables

Exemple de déclarations :

Const tva = 18.6
Dim prix as currency
Public nom as string

Option explicit (déclaration générale) : rend obligatoire la déclaration des variables.

	Locale à un sous-programme	Accessibles à tous les sous-programmes du module de feuille ou du module où la variable est déclarée.	Globale (accessible partout)
Dans le corps du sous-programme	**Dim**		
Dans la partie général déclaration (module ou module de feuille)		**Dim, Private**	
Dans la partie général/déclaration d'un module			**Public**

- DIM, PRIVATE : Les variables déclarées à l'aide de l'instruction Dim ou Private au niveau général/déclaration (module ou module de feuille) sont disponibles pour toutes les procédures du module ou du module de feuille concerné. Au niveau procédure, les variables ne sont disponibles qu'au sein de la procédure.
- PUBLIC : Instruction utilisée au niveau général/déclaration du module pour déclarer des variables publiques c'est à dire accessible à tous les modules et au module de feuille.

Structures de contrôle en Visual Basic

- **alternative**

```
If <condition> then
    <Instructions>
Else
    <Instructions>
Endif
```

- **Itération**

Tant que :

```
While <condition de poursuite>
    Instructions
Wend
```

Répéter…Jusqu'à :

```
Do
    Instructions
Loop Until <condition d'arrêt>
```

Pour…Fin Pour

```
For <variable> = <départ> to <Fin> [ step <pas>]
    <instructions>
Next <variable>
```

Procédures et Fonctions

Procédures

[Private | Public] Sub <nom> [<paramètres formels>]
    <déclarations locales>
    <instructions>
End Sub

Paramètres formels :
[ByVal|ByRef] varname[( )] As type , [ByVal|ByRef] varname[( )] As type], …

Private/Public : un sous programme peut être public et donc visible et utilisable partout ou privé et alors utilisable uniquement dans le module où il est déclaré.

Fonctions

[Public | Private] Function <nom> [(<paramètres formels>)] As <type>
<déclarations locales>
<instructions>

<nom> = <expression>
End Function

Quelques instructions et fonctions de base

Affectation <variable>=<expression>
Appel de procédure [Call] <nom procédure> (parametre1, paramètre 2, …)
Appel d'une fonction <variable>=nomfonction(paramètre, ….)

Opérateurs et comparateurs :
Opérateur Booléens : not, and, or, xor
Comparateur : =,>,<,<=,>=,<>, is
Opérateurs Arithmétique : ^(exposant), -(unaire),*,/, DIV,MOD,+,-
Opérateurs sur les chaînes de caractères : & (concaténation)

Procédures et fonctions prédéfinies

Numériques

Abs	Retourne la valeur absolue d'un nombre
Cos, sin, tan, atn	Fonctions trigonométriques, l'angle étant exprimé en radians
Exp, log	Fonctions exponentiel et log népérien
Sqr	Racine carré
int	Retourne la partie entière d'un nombre réel
Randomize	Procédure qui initialise le générateur de nombres aléatoires
Rnd	Retourne un single entre 0 et 1
Val	Conversion d'une chaîne en une valeur numérique
Str	Conversion d'un nombre en une chaîne de caractère

Procédures et fonctions prédéfinies

Sur les chaînes de caractères

Chr	Retourne le caractère dont le code est donné en paramètre. chr(1 3) insère un saut de ligne
Ltri,rtrim,trim	Retourne la chaîne passée en paramètre privée de ses caractères espaces initiaux, terminaux les deux.
Left(ch,i), right(ch,i)	Retourne les i premiers, (derniers) caractères de la chaîne ch
mid(ch,i,lg)	Retourne lg caractères de ch à partir du ième élément
Len(ch)	Retourne le nombre de caractères de la chaîne
Isnumeric (ch)	Retourne vraie si la chaîne peut être interprétée comme un nombre , sinon faux est renvoyée.
Val	Convertit une chaîne de caractères en un nombre
Str	Convertit un nombre en une chaîne de caractères

Date et heure

Time	Retourne l'heure actuelle dans une chaîne de 8 caractères hh.mm.ss
Date	Retourne la date dans une chaîne de 10 caractères jj.mm.aaaa
Now	Retourne un réel double qui code la date et l'heure actuelle
Dateadd, datediff, datepart	Calcul sur les dates (ajout, différence et extraction d'une partie de la date)

Interaction avec l'utilisateur

Emet un signal sonore	beep
Fonction qui retourne la chaîne de caractère saisie par l'utilisateur	Inputbox(message, titre, valdefaut, posx,posy)
Affichage d'une valeur dans une boîte de dialogue	Msgbox(message, type de la boite, titre)

Type de la boîte peut prendre l'une ou la somme des valeurs suivantes : vbOkCancel,vbYesNo, vbCritical, vbQuestion, vbExclamation, vbInformation
Utilisée comme une fonction, la valeur retournée dépend du bouton que l'utilisateur a cliqué : vbOk, vbCancel, vbYes, vbNo.