Visual Basic

A. Belaïd
Université de Nancy 2

Introduction

- Pourquoi Visual Basic ?
  - L'un des langages de programmation les plus performants et les plus simples à utiliser
  - Créé par John G. Kemeny et Thoams E. Kurtz en 1963 et devient vite un langage populaire
  - Adapté sur PC par Bill Gates, au milieu des années 70
  - Depuis, plusieurs versions améliorées pour PC ont vu le jour :
    - Microsoft QuickBasis et MS-DOS Qbasic
  - La simplicité du langage explique son choix pour le développement d'interfaces

Introduction

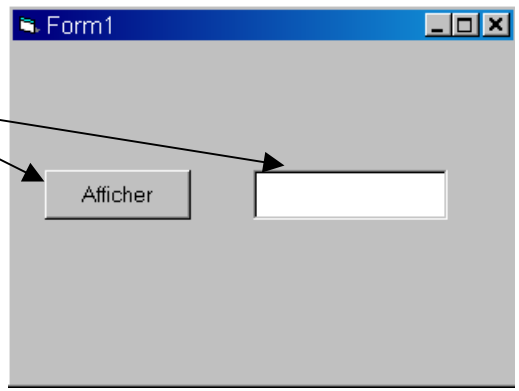
- Développement d'un programme sous VB
  - Trois étapes :
    - Création de l'interface utilisateur à l'aide des contrôles VB
    - Définition des caractéristiques ou propriétés des éléments qui composent l'interface
    - Ecriture du code de programmation pour un ou plusieurs éléments de l'interface en fonction des besoins

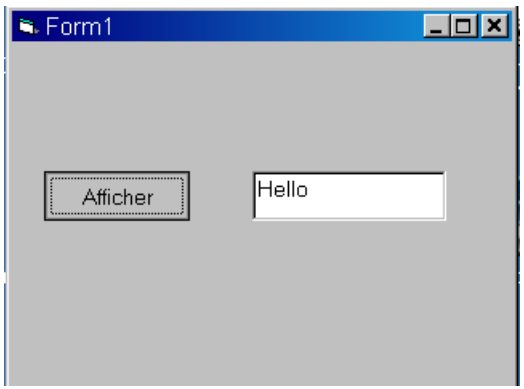
Introduction

- Exemple
  - Ecrire un programme VB qui affiche "Bonjour"
  - Solution : exo1-demmarrage

Interface

Objets





Avant le click

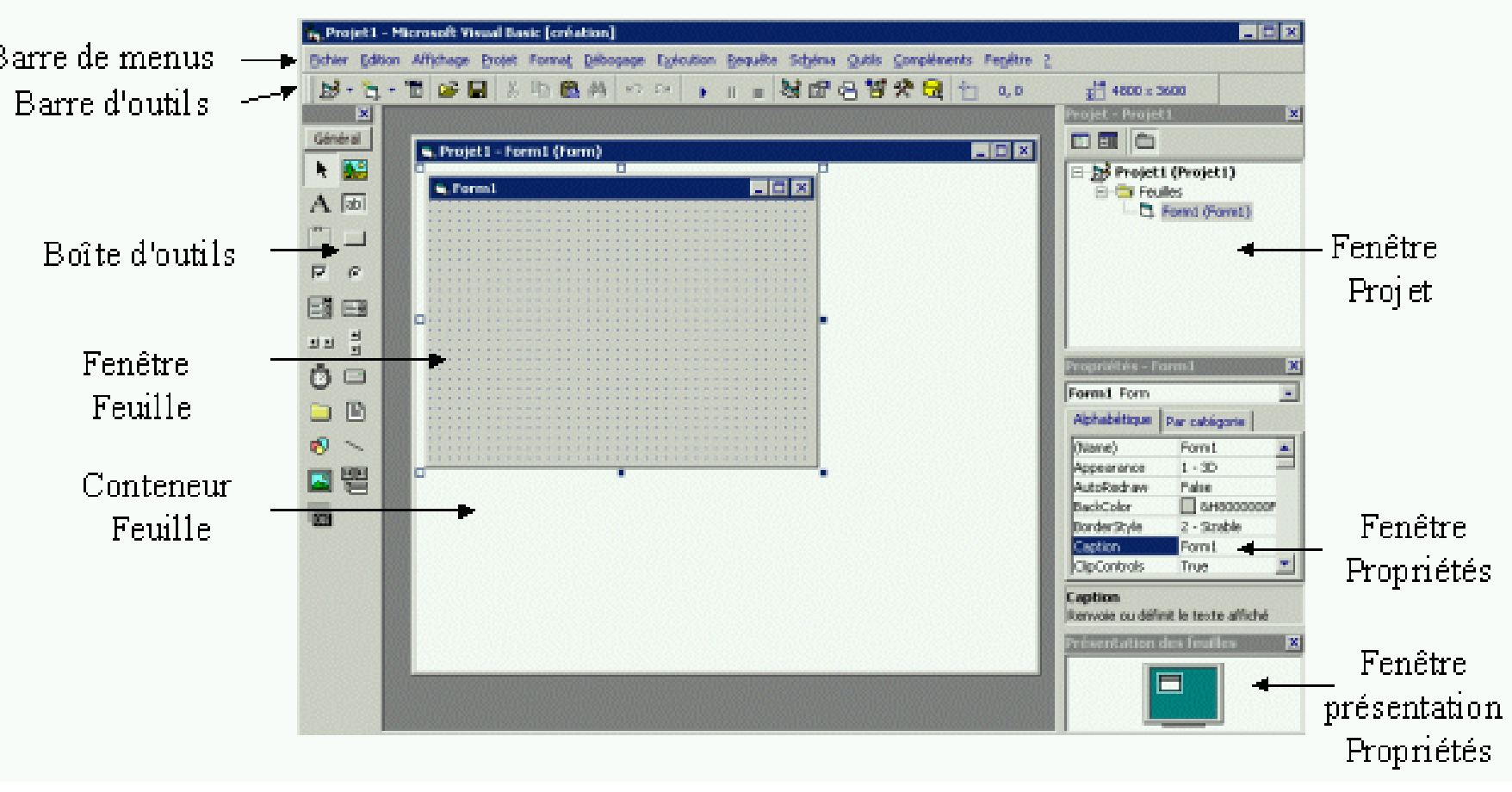
Après le click

Prise de connaissance

- Démarrage de VB
  - Cliquer sur « Démarrer », sur « Programmes », puis sur le dossier « Microsoft Basic 6.0 »
  - Cliquer sur l'icône du programme VB
    - La boîte de dialogue « Nouveau projet » s'affiche et un choix se prose pour un type de projet à créer
  - Cliquer sur « ouvrir » pour accepter la proposition par défaut
    - Un nouveau projet s'ouvre accompagné de fenêtres et d'outils
      - La taille et la forme exacte de ces fenêtres dépendent de la configuration du système

Prise de connaissance

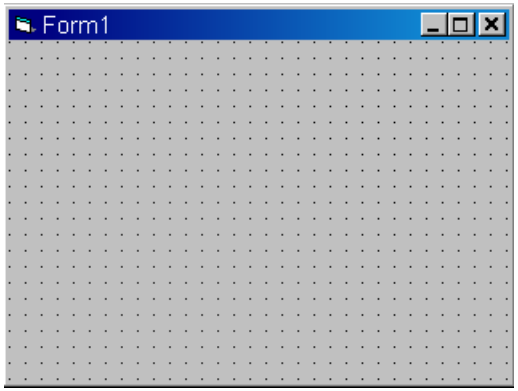
Interface



Prise de connaissance

Interface

- Feuille Interface Utilisateur
  - C'est la feuille par défaut (Fenêtre feuille)
    - appelée **Form1**, s'affiche au démarrage
  - Grille standard avec des points servant à aligner les éléments créés et composant l'interface Utilisateur



  - On peut ajuster la taille de l'interface à l'aide de la souris
  - On peut ajouter des feuilles supplémentaires à l'aide de la commande : « Ajouter une feuille » du Menu « Projet »

Prise de connaissance

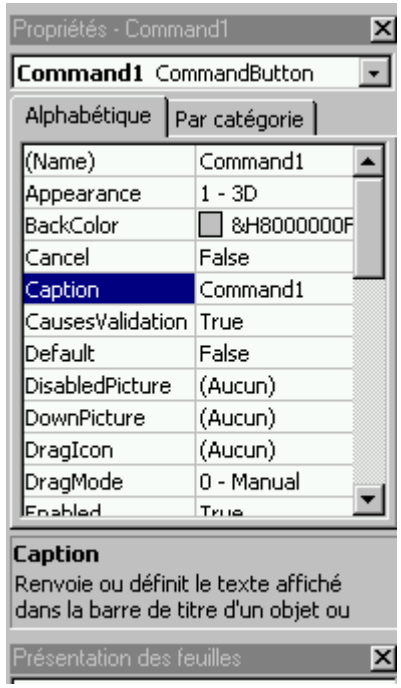
Interface

- Boîte à outils
  - Contient des **outils** et **contrôles** permettant d'ajouter des éléments à l'interface
  - Chaque contrôle ajouté à l'interface devient un **objet**, ou élément programmable de l'interface
  - A l'exécution du programme, ces objets agiront comme tous les objets standards d'une application Windows

Prise de connaissance

Interface

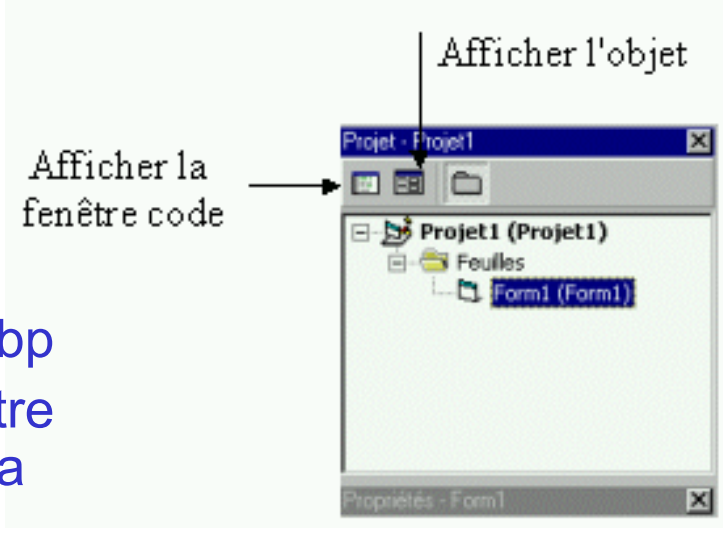
- Fenêtre Propriétés
  - Répertorie les propriétés possibles des éléments de l'interface et offre la possibilité de les changer
  - On peut affecter directement des propriétés aux objets sélectionnés
  - Ces propriétés peuvent être ensuite changées par programme (en agissant sur le code)
  - Si la fenêtre n'apparaît pas, cliquer sur le bouton correspondant de la barre d'outils



Prise de connaissance

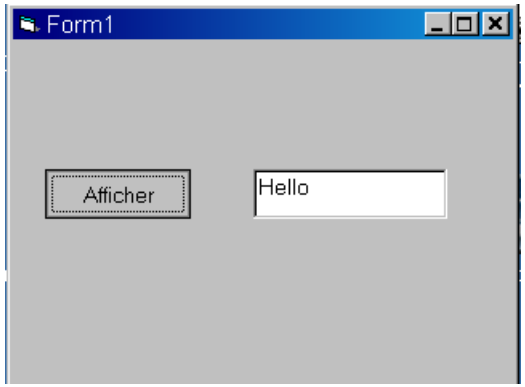
Interface

- Fenêtre Projet
  - Répertorie les fichiers crées dans le projet
  - On y accède par 2 boutons
    - Code / Afficher l'objet
  - Fichier projet est suffixé par .vbp
  - Sous le nom du projet, la fenêtre affiche les composants sous la forme d'une arborescence
    - Cliquer sur le bouton « Explorateur de projet » pour l'afficher



Ecriture d'un programme

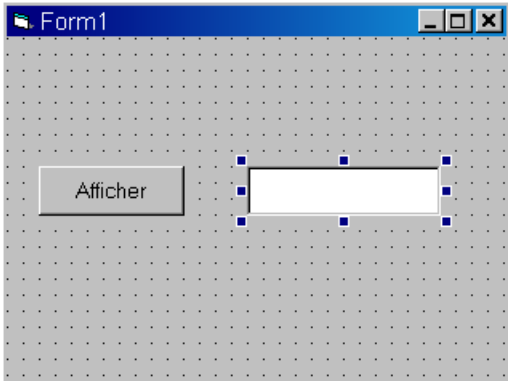
- Exemple 1 : exo1-demmarrage
  - Réaliser le programme d'affichage du message "Hello"
  - Règle :
    - En cliquant sur "Afficher"
      - ➢ Le message s'affiche dans la zone de texte



Utilisation des contrôles

- Exemple 1 : actions
  - Insérer un bouton Commande
  - Utiliser Caption pour donner le nom "Afficher"
  - Insérer une zone de texte par : **TextBox** (bouton ab)
  - Cliquer sur la zone Text1, annuler la valeur de la propriété Text (de manière à avoir la zone de texte vide au départ) ;
  - Double-Cliquer sur la zone de commande « ok » pour rentrer l'instruction dans la fenêtre de code de Command1_Click()

    Text1.text = "Hello "

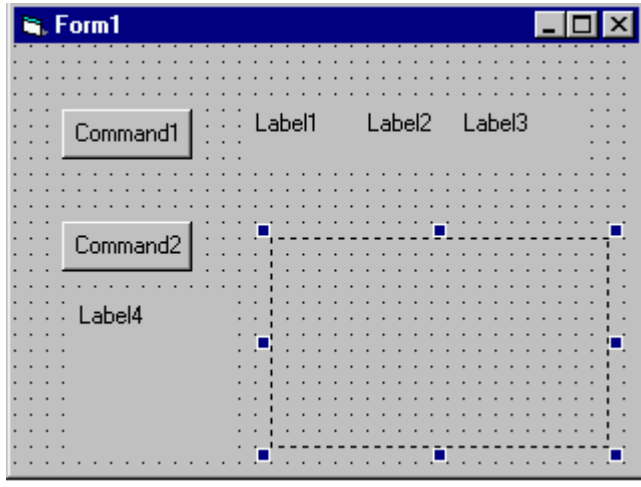


Ecriture d'un programme

- Exemple 2 : programme de jeu
  - Il s'agit de réaliser une interface d'un programme de jeu simulant une machine à sous : Lucky Seven
  - Règle :
    - En cliquant sur "Jouez"
      - Des chiffres apparaissent, si un 7 apparaît parmi eux, alors l'image des sous est affiché
    - Pour arrêter, appuyer sur le bouton : Arrêtez

Ecriture d'un programme

- Exemple 2 : Création de l'interface
  - Ouvrir un nouveau projet
  - Créer les deux boutons de commande : bouton « commande » rester appuyé dessus, le déplacer sur la feuille et le positionner à l'endroit voulu
    - Le re-dimensionner avec la souris (pointer sur le coin inférieur et tirer avec la souris)
  - Pour les chiffres, utiliser le bouton « label »
  - Prévoir une zone plus importante pour le label 4 accueillant le texte Lucky Seven
    - Introduire une zone « Image » pour y insérer l'image des sous



Ecriture d'un programme

- Donner des propriétés aux commandes
  - Cliquer sur le bouton Command1
    - Double-cliquer sur la fenêtre propriétés
    - Double-cliquer sur Caption
      - Saisir « Jouer »
    - Faire la même chose avec Command2 en saisissant « Arrêter »
      - Pour retrouver les commandes, il suffit d'aller dans la zone de liste déroulante, Objet situé en haut de la fenêtre Propriétés

Ecriture d'un programme

- Définition des propriétés des étiquettes de chiffres
  1. Sélectionner les trois étiquettes de chiffres en cliquant d'abord sur la 1ère puis sur les deux autres en maintenant le bouton MAJ appuyé
     - Un rectangle de sélection encadre chacune des étiquettes
       - *Comme plusieurs objets ont été sélectionnés, seules les propriétés susceptibles d'être changées collectivement sont affichées dans la fenêtre Propriétés.*
  2. Propriétés à définir :
     - Alignement : choisir 2-center ;
     - BorderStyle : choisir 1-Fixed dans le menu ;
     - Font : Times New Roman, style Gras, taille : 24 points ;
  3. Supprimer les trois libellés afin que les cases soient vides au démarrage du programme :
     - Sélectionner individuellement chacune des trois étiquettes ;
     - Double-cliquer sur la propriété Caption et appuyer sur SUPPR. Le libellé de l'objet Label1 est supprimé. Répéter l'opération pour les deux autres

Ecriture d'un programme

- Définition de l'étiquette descriptive
  1. Cliquer sur l'objet étiquette ;
  2. Changer la propriété Caption en Lucky Seven ;
  3. Changer la fonte, la taille… comme précédemment ;
  4. Changer la couleur en agissant sur ForeColor. Cliquer sur l'onglet "Palette" puis sur la case violet foncé. La couleur est traduite en Hexadécimal dans la fenêtre

Ecriture d'un programme

- Définition des propriétés de la zone Image
  - Cette zone est sensée contenir le graphique des pièces. Ce graphique apparaît lorsque l'utilisateur remporte le jackpot (au moins une fois le chiffre 7)
    1. Cliquer sur l'objet zone d'image
    2. Mettre la propriété Stretch à True
    3. Double cliquer sur la propriété Picture dans la fenêtre Propriétés. La boîte de dialogue "Charger une image" apparaît, puis aller chercher l'image dans la partition Microsoft sous Clipart. Le fichier s'appelle "Pieces.wmf". En l'ouvrant, le métafichier Windows est chargé dans la zone d'image de la feuille
    4. Mettre la propriété Visible sur False de manière à masquer les pièces au démarrage du programme. (Vous le ferez apparaître ultérieurement dans le programme)

Ecriture d'un programme

- Écriture du code du programme
  - Il s'agit du code chargé de :
    - calculer les chiffres aléatoires, de les afficher dans les cases correspondantes et de détecter un éventuel jackpot.
  - Comme le programme est géré par l'activation des boutons "Jouer" et "Arrêter", il suffit d'associer le code approprié à ces boutons
  - La fenêtre *Code* est une fenêtre spéciale de l'environnement de programmation permettant d'entrer et d'éditer les instructions de programmation

Ecriture d'un programme

- Écriture du code du programme : exo-Lucky
  - Doubler cliquer sur la fenêtre "Arrêter" sur la feuille. La fenêtre Code apparaît
  - Rentrer l'instruction : End
  - Doubler cliquer sur la fenêtre "Jouer" sur la feuille. La fenêtre Code apparaît
  - Rentrer le code suivant :

```
Image1.Visible = False
Label1.Caption = Int(Rnd * 10)
Label2.Caption = Int(Rnd * 10)
Label3.Caption = Int(Rnd * 10)
If (Label1.Caption = 7) Or (Label2.Caption = 7) Or _
(Label3.Caption = 7) Then
Image1.Visible = True
Beep
End If
```

Utilisation des contrôles

- Application 3 : réalisation d'un browser
  - Il s'agit de permettre, lors de l'exécution, la sélection d'un fichier image et d'en afficher le contenu
  - La sélection concerne :
    - La partition : a, c, d… : une fenêtre spéciale sera ouverte à cet effet
    - Le répertoire : une fenêtre spéciale sera ouverte à cet effet avec un menu déroulant. Le contenu sera mis à jour à partir de la sélection de la partition
    - Le fichier : une fenêtre spéciale sera ouverte à cet effet avec un menu déroulant. Le contenu sera mis à jour à partir de la sélection du répertoire

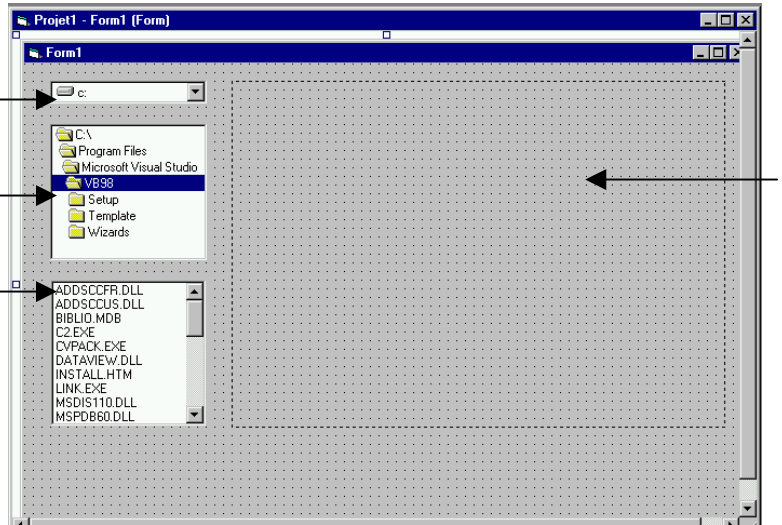
Utilisation des contrôles

- Application 3 (suite)
  - Les commandes :
    - Création de la zone du lecteur : contrôle DriveListBox
    - Création de la zone du répertoire : contrôle DirListBox
    - Création de la zone du fichier : contrôle FileListBox
    - Création de la zone image : contrôle Image

zone du lecteur

zone du répertoire

zone du fichier



zone de l'image

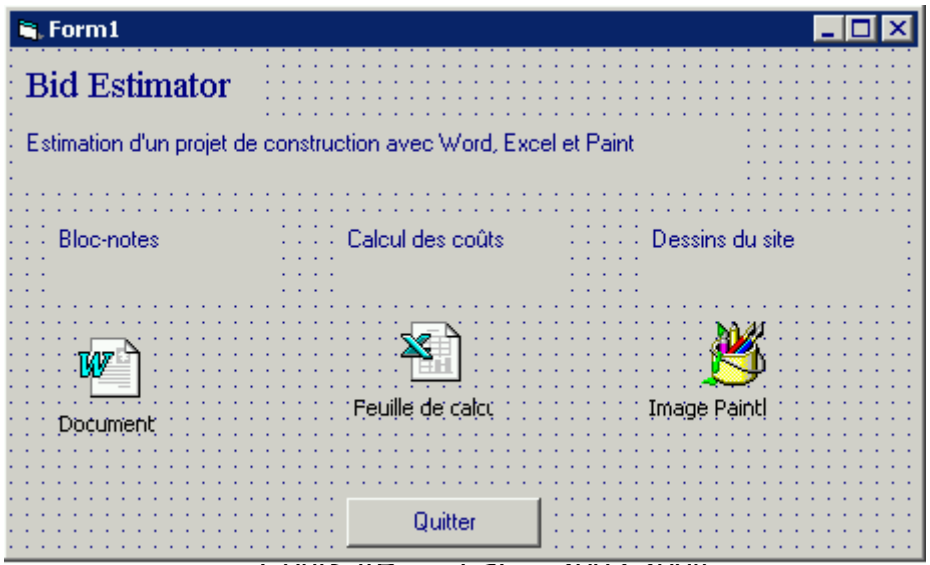
Utilisation des contrôles

- Application 3 (suite)
  - Mise à jour des propriétés
    - Le fichier image doit être d'un des types suivants :
      - *.bmp, *.mf, *.ico ;
    - Image1 :
      - Stretch= true, BorderStyle = 1-Fixed Single
  - Code :

```
Private Sub Dir1_Change()
    File1.Path = Dir1.Path
End Sub
Private Sub Drive1_Change()
    Dir1.Path = Drive1.Drive
End Sub
Private Sub File1_Click()
    SelectedFile = File1.Path & "\" & File1.FileName
    Image1.Picture = LoadPicture(SelectedFile)
End Sub
```

Utilisation des contrôles

- Application 4 : ***Utilisation d'un Objet OLE***
  - Il s'agit de créer une interface de boutons d'applications à l'aide du contrôle OLE. Comme le montre la figure suivante, il s'agit de créer 3 boutons correspondant aux applications Microsoft : Word, Excel et Paint



Utilisation des contrôles

- Application 4 : ***Utilisation d'un Objet OLE***
  - Commandes :
    - Créer les labels tels que mentionné dans la figure
    - Utiliser le contrôle OLE pour créer trois rectangles en dessous des labels :
      - bloc-notes, calcul des coûts et Dessins du site ;
    - Une fenêtre d'objets s'affiche. Faire défiler les objets et choisir l'application souhaitée
    - Changer les propriétés des objets :
      - Pour Label 1 : Font : Times…
      - Pour Ole1, Ole2 et Ole3 : BorderStyle (0-None), Appearence (0-Flat) et BlackColor (Gris Clair)
    - Mettre le code End dans la procédure événementielle Command1_Click

Programmation en VB

- Anatomie d'une instruction VB
  - C'est un ordre donné à la machine pour réaliser un travail donné
  - Cet ordre peut être de différentes natures :
    - lecture d'une donnée, affichage d'un résultat, ouverture d'un fichier, calcul d'un résultat, action système, etc.
  - L'instruction a une syntaxe qui dénote une phrase dans le langage d'expression des ordres
  - Exemples

    Beep // un seul mot clé dont l'effet est d'émettre un son

    Label1.Caption = Time // syntaxe plus développée conduisant à affecter par le signe = la valeur de la fonction de VB Time à la propriété Caption de l'objet Label1

Programmation en VB

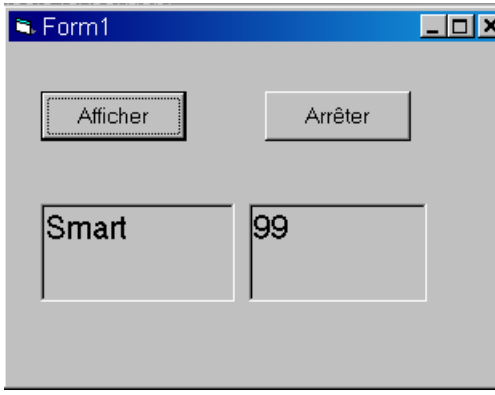
- Notion de variable
  - C'est un conteneur de valeur
  - Déclaration

    DIM Nom
    - Réserve un espace mémoire
    - En l'absence de déclaration de type, Nom est déclaré de type variant et peut contenir aussi bien des nombres que du texte
  - Déclaration implicite sans DIM

Nom = "Toto" Valeur1 = 24 Valeur2 = 38.5

Programmation en VB

- Notion de variable
  - **Exercice :** écrire une interface VB affichant le contenu d'une variable Nom, contenant successivement le texte "smart" et le nombre 99
    1. Créer deux boutons : Afficher et Arrêter
    2. Changer Caption de Command1 en Afficher
    3. Changer Caption de Command2 en Arrêter
    4. Créer deux labels : Label1 et Label2 associés au bouton Afficher
    5. Changer les propriétés des labels : Caption =vide, BorderSTyle = 1-Fixed Single, Font = MS sans sérif, normal, 12



Programmation en VB

- Notion de variable
  - **Exercice (suite)**
    6. Introduire comme code dans la procédure événementielle Command1_Click associé à Label1 et Label2:

```
Dim Nom
Nom = "Smart"
Label1.Caption = Nom
Nom = 99
Label2.Caption = Nom
```

    7. Introduire comme code dans la procédure événementielle Command2_Click :

```
End
```

Programmation en VB

- Utiliser une variable pour l'entrée : InputBox
  - InputBox est une fonction de lecture VB qui renvoie une valeur. Cette valeur est affectée à une variable
    1. Créer deux boutons commandes, l'un appelé InputBox et l'autre Quitter (agir sur leur Caption pour indiquer ces noms)
    2. Créer un bouton label, le creuser en agissant sur BorderStyle
    3. Mettre End dans le code de la commande Quitter
    4. Inscrire le code suivant pour la commande InputBox

```
Dim Invite, NomComplet
Invite = "Saisissez votre prénom et votre nom"
NomComplet = InputBox$(Invite)
Label1.Caption = NomComplet
```

Programmation en VB

- Utiliser une variable pour la sortie : MsgBox
  - Il est possible d'afficher le contenu d'une variable en l'affectant à une propriété (par ex.. caption d'un objet étiquette) ou en le passant comme argument à une fonction de boîte de dialogue
- La syntaxe de MsgBox est :
  - BoutonCliqué = MsgBox(Message, NumeroDeBouton, Titre)
  - Message : texte affiché à l'écran
  - BoutonCliqué : est un numéro de style de bouton (de 1 à 5)
  - Titre : texte affiché dans la barre de titre de la boîte de dialogue
  - La variable BoutonCliqué est affectée du résultat livré par la fonction MsgBox, c'est le bouton sur lequel l'utilisateur a cliqué dans la boîte de dialogue

Programmation en VB

- Utiliser une variable pour la sortie : MsgBox
  - **Pratique :**
    - Réutiliser le projet précédent
    - Ajouter dans le code de la procédure Command1_Click l'instruction suivante avant

```
Label1.Caption = NomComplet : MsgBox (NomComplet), ,
    "Résultat de l'entrée"
```

    - Exécuter : une boîte de dialogue apparaît pour confirmer l'entrée du label avant de l'afficher dans label1, comme montré ci-après :

Programmation en VB

Principaux types

Type de données	Taille	Plage	Exemple
Entier(Integer)	2 octets	-32768 à 32768	Dim Oiseaux% Oiseaux%=37
Entier long (Long Integer)	4 octets	-2147483648 à +2147483648	Dim Salaire& Salaire& = 350000
Virgule flottante simple précision	4 octets	-3,402823 E38 à 3,402823 E38	Dim Prix! Prix = 899,90
Virgule flottante double précision	8 octets		Dim Pi# Pi# = 3,1415926535
Monétaire (Currency)	8 octets		Dim Debit@ Debit@ = 7600300,50
Chaîne (String)	1 octet par caractère	De 0 à 65535 caractères	Dim Chien$ Chien$ = "pointer"
Booléen (Boolean)	2 octets	Vrai ou Faux	Dim Flag as Boolean Flag = True
Date	8 octets	1 janvier 100 au 31 décembre 9999	Dim Anniv as Date Anniv = #1/3/63#
Variant	16 octets (pour les nombres), 22 octets + 1 octet par caractère	Toutes les plages des autres types	Dim Total Total = 289,13

Programmer en VB

- Type de données personnalisées
  - VB permet de créer des types de données personnalisées grâce à l'instruction **Type**
  - Exemple :

```
Type Personnel
    Nom As String
    DateDeNaissance As Date
    DateEmbauche As Date
End Type
```

  - Après la création du type, on peut l'utiliser dans le programme

```
Dim ChefProduit As Personnel
ChefProduit.Nom = "Eric Cody"
```

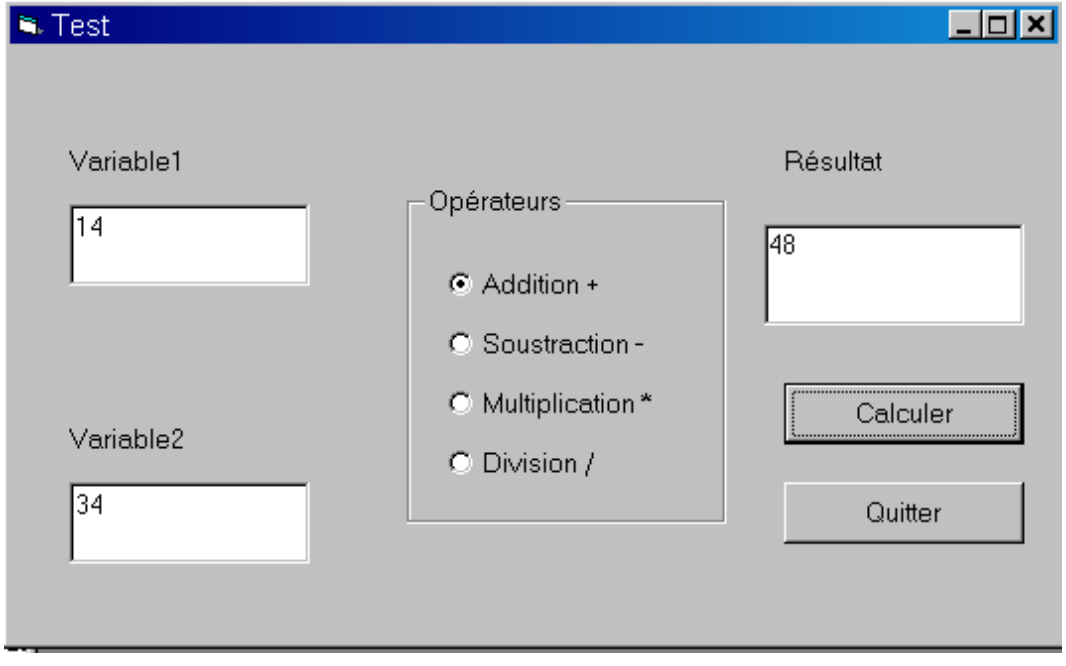
Programmer en VB

- Notion d'opérateur

Opérateur	Opération mathématique
+	addition
-	soustraction
*	multiplication
/	division
\	Division entière
Mod	Modulo (reste de la division entière)
^	Puissance
&	Concaténation de chaînes

Programmer en VB

- Notion d'opérateur : exercice
  - Créer l'interface suivante permettant de réaliser les opérations mathématiques binaires, à la manière de l'interface suivante



Programmer en VB

- Notion d'opérateur : exercice : exo9-operateurs
  1. Deux contrôles label, Label1 et Label2 portant le texte Variable 1 et Variable 2 dans Caption
  2. Deux contrôles TextBox, Text1 et Text2. Effacer leur Caption
  3. Un contrôle Frame pour créer la zone des opérateurs, changer sa Caption
  4. Utiliser la commande OptionButton avec comme Caption le texte de l'opérateur pour introduire les opérateurs
  5. Un contrôle Label pour Label3 portant le texte Résultat dans la propriété Caption
  6. Un contrôle Label, Label4, dont la propriété Caption est vide et Borderstyle à 1-Fixed Single
  7. Un contrôle CommandButton portant le texte Calculer dans la propriété Caption et un autre portant le texte Quitter dans Caption

Programmer en VB

- Notion d'opérateur : exercice (suite 1)
  8. Mettre End comme Code à la procédure de Quitter ;
  9. Mettre le code suivant dans la procédure de Calculer.

```
Dim Premier, Second 'Déclaration de variables
Premier = Val(Text1.Text) 'Conversion en nombre
Second = Val(Text2.Text) 'Conversion en nombre

'Si le premier cercle est activé
If Option1.Value = True Then
Label4.Caption = Premier + Second
End If
'Si le deuxième cercle est activé
If Option2.Value = True Then
Label4.Caption = Premier - Second
End If
```

Programmer en VB

- Notion d'opérateur : exercice (suite 2)

```
'Si le troisième cercle est activé
If Option3.Value = True Then
Label4.Caption = Premier * Second
End If
'Si le quatrième cercle est activé
If Option4.Value = True Then
Label4.Caption = Premier / Second
End If
```

Programmer en VB

- Les expressions conditionnelles
  - Opérateurs de comparaison
  - Exemples :

    10 <> 20 'donne vrai

    Score < 20 'donne vrai si la valeur de score est inférieure à 20

    Score = Label1.Caption 'donne vrai …

    Text1.text = "Jean" 'donne vrai si …
  - Structures de décision If…Then
    - Forme 1:

      If Condition Then Instruction
    - Exemple :

      If Score>=20 Then Label1.Caption = "Vous avez gagné !"

Programmer en VB

- Les expressions conditionnelles
  - Forme 2 :

```
If Condition1 Then Instructions1
    ElseIf Condition2 Then  Instruction2 …
    Else
        Instructions-finales
End If
```

Programmer en VB

- Pratique
  - Ecrire une interface qui valide à la connexion, pour une machine, le nom de son utilisateur

Programmer en VB

- Solution : code à écrire pour la commande d'ouverture

```
NomUtilisateur = InputBox("Saisissez votre nom")
If NomUtilisateur = "Laura" Then
MsgBox ("Bonjour, Laura ! En forme pour attaquer le travail ? ")
Form1.Picture = _
LoadPicture("C:\Program Files\Microsoft Office\Clipart\Popular\Approuv.wmf")
ElseIf NomUtilisateur = "Marc" Then
MsgBox ("Bonjour, Marc ! Prêt à attaquer le travail ? ")
Form1.Picture = _
LoadPicture("C:\Program Files\Microsoft Office\Clipart\Popular\Approuve.wmf")
Else
MsgBox ("Désolé, Je ne vous connais pas !")
   End 'Abandon du programme
End If
```

Programmer en VB

- Solution (suite)

  Une connexion négative donnera ce résultat :

Programmer en VB

- Opérateurs logiques
  - On peut écrire des expressions conditionnelles complexes en utilisant les opérateurs logiques suivants : and, or, not, xor
  - Pratique :
    - compléter l'exercice précédent par le contrôle du mot de passe
    - Pour cela : il suffit d'ajouter après l'entrée du nom :

```
Pass = InputBox ("Saisissez votre mot de passe")
If NomUtilisateur = "Laura" And Pass ="May17" Then …
```

Programmer en VB

- Opérateurs logiques (suite)
  - Select Case
    - C'est une sélection à choix multiple
  - Forme :

```
Select  Case variable
Case Value 1
        Instruction1
Case value2
        Instruction2
…
End Select
```

Programmer en VB

- Opérateurs logiques (suite)
  - Exemple

```
Select Case Pourcent
    Case Is >= 90
        Lettre = "A"
    Case 60 to 89
        Lettre = "B"
    Case Else
        Lettre = "F"
    End Select
```

  - Notez qu'il y a plusieurs façons d'exprimer la condition du Case:
    - avec les signes < et > il faut utiliser le IS
    - on peut spécifier un range: 60 TO 89
    - on peut spécifier des valeurs: 44, 46, 55, 62

Programmer en VB

- Les boucles
  - Boucle For … Next
    - Permet d'exécuter un groupe d'instructions un certain nombre de fois
  - Syntaxe :

```
For variable = start To end
        Instructions à exécuter répétitivement
Next Variable
```

  - Exemple :

```
For i = 1 To 4  {step j}
    Beep
Next i
```

Programmer en VB

- Utilisation du Print

Form1.Print " Coucou "

  - Va écrire en haut à gauche de la Form1 " Coucou "
  - La prochaine instruction Print provoquera une écriture juste en dessous… sauf si nous avions terminé l'instruction précédente par un point-virgule. Auquel cas, la prochaine écriture s'effectuera à la suite

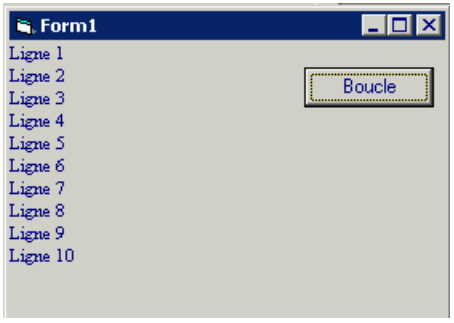
Programmer en VB

- Utilisation du Print : Remarque
  - A priori, si on redimensionne Form... le texte disparaît. Cela peut être gênant
  - Il est facile d'y remédier :
    - Fixer auparavant la propriété AutoRedraw de la Form à True
    - Attention, cela ralentit toutefois l'application
  - Autres solutions possibles :
    - Gérer soi-même l'événement Paint de la Form, qui correspond à un redimensionnement…

Programmer en VB

- Application 1 : affichage du compteur de la boucle par Print
  - Sélectionner Form1
  - Mettre la propriété AutoRedraw à True
  - Double-cliquer sur la commande Boucle et mettre le code

```
For i =1 To 10
    Print "Ligne"; i
Next i
```



Programmer en VB

- Application 2 : affichage d'images
  1. Préparation d'imagettes
     - Ouvrir la fenêtre Poste de travail et passer en mode d'affichage Grandes icônes par le menu Affichage
     - Appuyer sur les touches ALT+IMPR ECRAN : cette commande a pour effet de copier la fenêtre active dans le Presse-papiers de Windows
     - Charger Paint
     - Appeler la commande Coller du menu Edition
     - Activer l'outil Sélection et tracer un cadre de sélection autour de la première icône

Programmer en VB

1. Préparation d'imagettes (suite)
   - Appeler Copier du menu Edition pour placer une copie de cette icône dans le Presse-papiers
   - Appeler la commande "Copier vers" du menu Edition. Dans la boîte de dialogues, localiser le répertoire Exercices et ranger l'image sous le nom Image1.bmp

– Refaire la même chose pour les 3 autres images

Programmer en VB

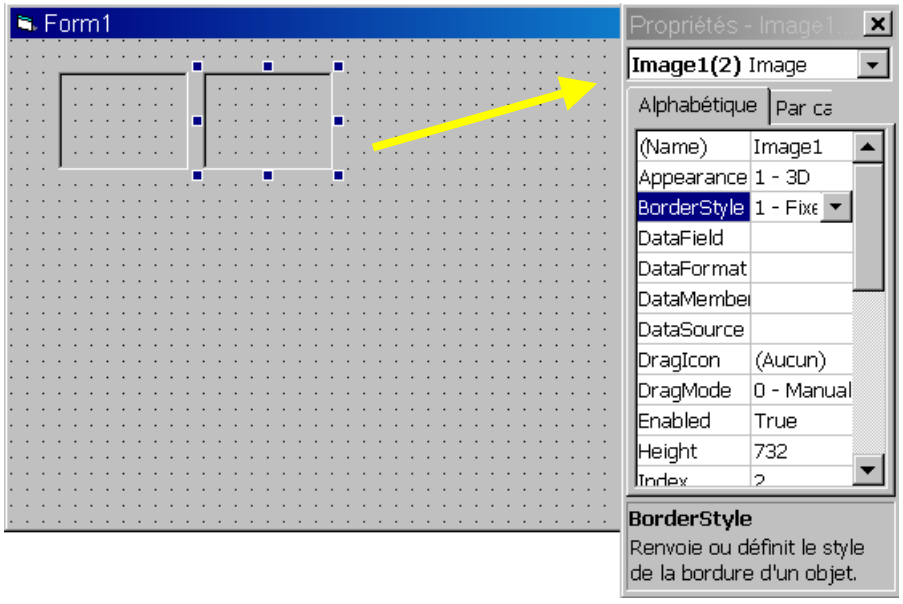
2. Création de l'application
   - Utiliser le contrôle Image et créer un petit contrôle image en haut de la feuille
   - Dans le menu Edition, Cliquer sur la commande Copier, une copie du contrôle Image est placée dans le presse Papiers de Windows. On va s'en servir pour créer trois nouveaux contrôles Image sur la feuille
   - Appeler la commande Coller du menu Edition, répondre oui à la création d'un groupe de contrôle.
     - Un deuxième contrôle Image apparaît, le glisser pour le superposer au rectangle de l'image

Programmer en VB

2. Création de l'application (suite)
   - Appeler Coller du menu Edition … et faire cela pour toutes les images
   - Créer un bouton de commande, appelée "Afficher les images" et lui affecter le code mentionné dans la figure
   - Enregistrer la feuille et le projet sous les MonGroupeContrôles, puis exécuter

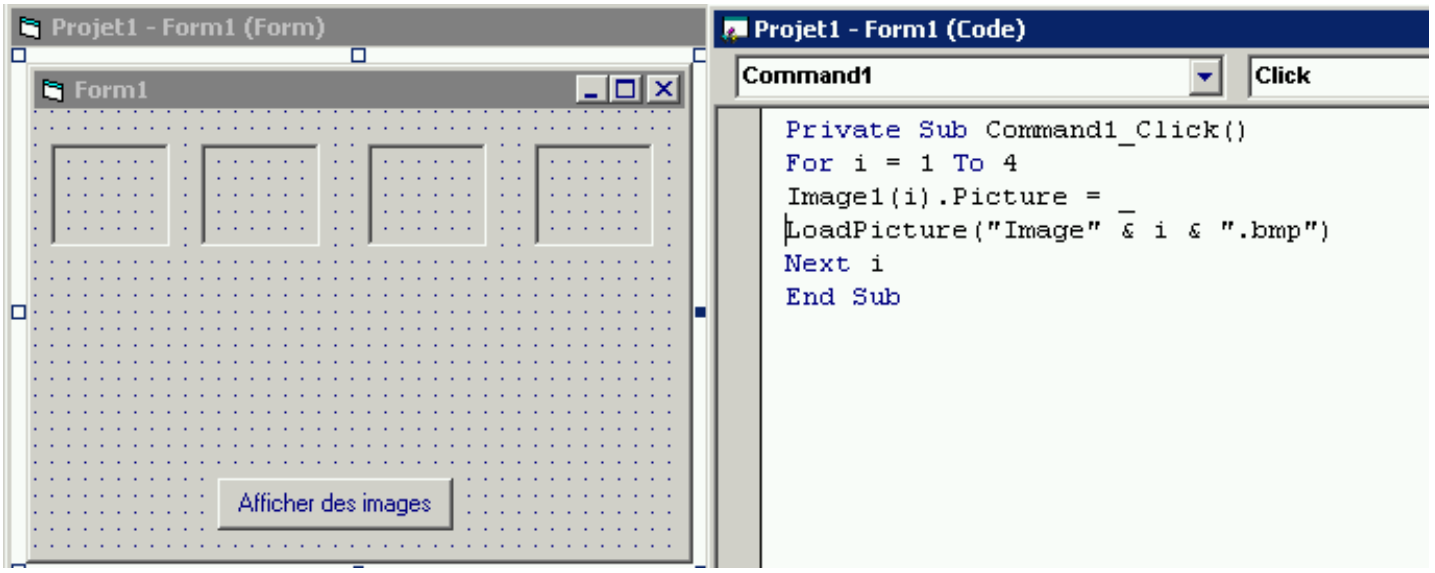
Programmer en VB

Vérifier bien en recopiant les zones d'images que leur numéro va de Image1(1) à Image1(4), sinon ça ne marchera pas



Programmer en VB

- Application 2 : interface



Programmer en VB

- Instruction EXIT FOR
  - Cette instruction permet d'interrompre la boucle et de sortir :
  - Exemple :

```
For i = 1 To 10
    Nom = InputBox("Saisissez votre nom ou tapez Fini pour quitter")
    If Nom ="Fini" Then Exit For
    Print Nom
Next i
```

Programmer en VB

- Boucle Do…Loop
  - Elle offre une alternative aux boucles For…Next. Elle exécute le bloc jusqu'à ce qu'une condition définie devienne True
  - **Syntaxe :**

    Do While condition

    Instructions à exécuter répétitivement

    Loop
  - **Exemple :**

```
Do While Nom <> "Fini"
    Nom = InputBox(("Saisissez votre nom ou tapez Fini pour quitter")
    If Nom <> "Fini" Then Print Nom
Loop
```

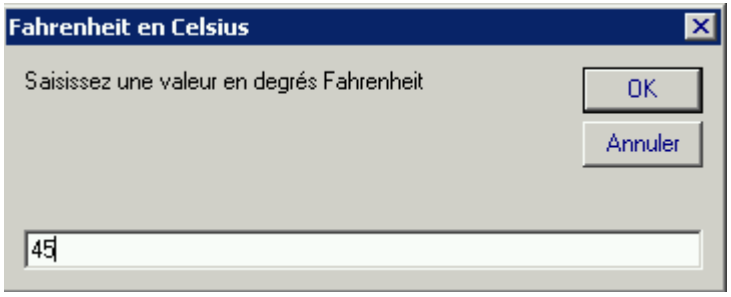
Programmer en VB

- Application
  - Ouvrir la fenêtre Propriétés de la feuille, mettre Visible à False (exécution en arrière plan)
  - Double-cliquer sur la feuille et entrer le code suivant :

```
Prompt = "Saisissez une valeur en degrés Fahrenheit"
Do
    Ftemp=InputBox(Prompt, "Fahrenheit en Celsius")
    If  Ftemp <> "" Then
        Celsius = Int(Ftemp + 40) * 5/9-40)
        MsgBox (Celsius),  , "Température en degrés Celsius"
    End If
Loop While Ftemp <> ""
End
```

Programmer en VB

- Résultat



Programmer en VB

- Until
  - Exemple :

```
Do Nom = InputBox(("Saisissez votre nom ou tapez Fini
    pour quitter")
    If Nom <> "Fini" Then Print Nom
Loop Until Nom = "Fini"
```

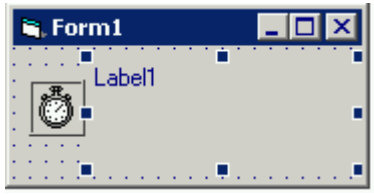
Programmer en VB

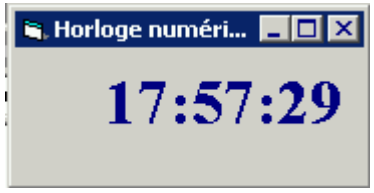
- Objet Timer
  - VB permet d'exécuter un groupe d'instructions pendant un Laps de temps déterminé en utilisant un objet Timer
  - C'est un objet horloge invisible permettant d'accéder à l'horloge système à partir d'un programme

Programmer en VB

- **Application 1 :** création d'une horloge numérique
  - Créer un nouveau projet, redimensionner la feuille
  - Cliquer sur le contrôle Label ;
  - Donner les propriétés suivantes :
    - Label 1 : Caption (vide), Font (Times, gras, 24), Alignement (2-Center)
  - Timer1 : Interval (1000), Enabled (True);
  - Form1 : Caption (Horloge numérique) ;
  - Double-Cliquer sur Timer et rentrer l'instruction suivante

    Label1.Caption = Time





Programmer en VB

- **Application 2 :** contrôler le délai de saisie du mot de passe
  - Créer une zone de texte, une commande, un label et un Timer
  - Text1 : Text (vide), PasswordChar (*)
  - Form1 : Caption (Mot de passe)
  - Label1 : Caption (Saisissez…)
  - Command1 : Caption (Test…)
  - Timer1 : Interval (15000), Enabled (True)
  - Mettre le code indiqué dans la figure et exécuter.

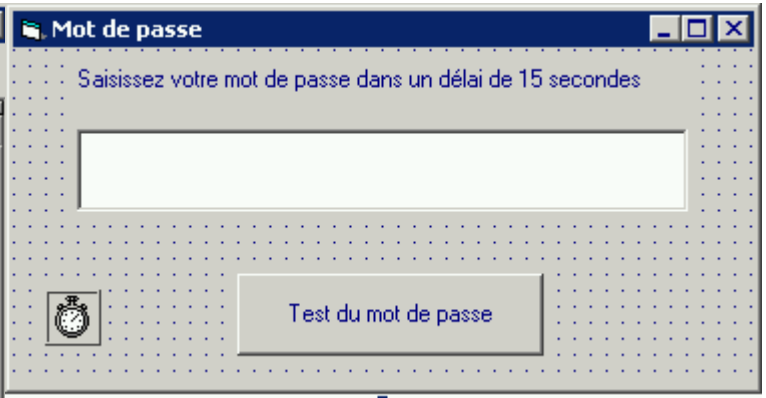
Programmer en VB

- **Application 2**

Projet1 - Form1 (Code)

Command1 | Click

```
Private Sub Command1_Click()
If Text1.Text = "Secret" Then
Timer1.Enabled = False
MsgBox ("Bienvenue sur le système !")
End
Else
MsgBox ("Désolé, je ne vous connais pas !")
End If
End Sub

Private Sub Timer1_Timer()
MsgBox ("Désolé, le délai est dépassé")
End Sub
```



Programmer en VB

- Notions de modules et de procédures
  - Si on écrit des programmes complexes
    - On va avoir besoin de multiplier les feuilles (Form) et les procédures événementielles qui utilisent les mêmes variables
  - Or
    - Les variables sont locales aux procédures événementielles et ne peuvent pas donc partagées (ni lues ni modifiées) par d'autres procédures
  - De même
    - Les procédures événementielles sont locales par rapport à la feuille dans laquelle elles ont été créées
  - Pour partager des variables et des procédures
    - Il faut les déclarer dans un ou plusieurs modules standards

Programmer en VB

- Module standard
  - C'est un fichier spécial
    - Suffixé par .bas
    - Contenant des variables et des procédures utilisables n'importe où dans le programme
    - L'enregistrer en faisant Enregistrer Module1
  - Contrairement aux feuilles
    - Les modules ne contiennent pas d'objets ni de propriétés
    - Ils ne sont formés que de code affichable et éditable dans la fenêtre code (du module)

Programmer en VB

- Module standard : exemple
  - Créer un nouveau projet
  - Cliquer sur la commande "Ajouter un module" dans le menu Projet et cliquer sur "Ouvrir"
    - VB ajoute au projet un module standard appelé Module1
    - Observer dans toutes les fenêtres l'apparition de module1
  - L'enregistrer comme Form1 et Projet1
  - Double-cliquer sur la fenêtre propriété, seule la propriété Name apparaît permettant de spécifier le nom d'objet du module
    - Ce nom permettra de distinguer les modules si on crée plusieurs
  - Renommer la propriété Name en modVariables et appuyer sur la touche ENTREE

Programmer en VB

- Déclarer une variable Public
  - Pour pouvoir partager une variable par toutes les procédures, il suffit de la déclarer Public dans un module standard
  - Ex :
    - Public TotalCourant
  - Par défaut
  - Les variables publiques sont déclarées de type Variant dans les modules, mais on peut spécifier un type
  - Ex :
    - Public Nom As String

Programmer en VB

- Application
  - Réouvrir le projet Lucky.vpb
  - Enregistrer le projet sous les noms Gains.frm pour la feuille et Gains.vbp pour le projet
  - Ajouter un nouveau contrôle Label à la feuille
  - Donner les propriétés suivantes à Label5 :
    - Alignment(2-center), Captions (Gains : 0), Font (Arial, Gras Italique, 12 points), ForeColor (Vert), Name (lblGains)
  - Donner la propriété suivante à Form1
    - Caption Lucky Seven

Programmer en VB

- Application (suite)
  - Ajouter un module standard
  - Ecrire dans le module :

    Public Gains
  - Enregistrer Module sous le nom : Gains.bas
  - Double-cliquer sur la commande "Jouer" de Feuille1 et ajouter les instructions suivantes après Beep :

    Gains = Gains+1

    lblGains.Caption = "Gains : " & Gains
  - Jouer et apprécier

Programmer en VB

- Créer une procédure à caractère général
  - En plus des variables publiques, un module standard peut contenir des procédures à caractère général
  - Cette procédure :
    - Peut être appelée de partout dans le programme
    - N'a rien avoir avec les procédures événementielles qui sont associées à des objets
  - Il existe 3 types de procédures
    1. Procédures de fonctions
       - Appelables par leur nom, peuvent recevoir des arguments et retournent une valeur associée à leur nom
    2. Procédures Sub
       - Idem que les procédures de fonctions sauf qu'elles ne retournent pas de valeurs associées à leur nom

Programmer en VB

- Créer une procédure à caractère général
  3. Procédures Property
     - Procédures utilisées pour créer et manipuler des propriétés personnalisées dans un programme
     - Elles sont bien pratiques pour personnaliser les contrôles et d'étendre le langage en créant de nouveaux objets, nouvelles propriétés ou nouvelles méthodes

Programmer en VB

- les procédures de fonction
  - Elles ont un nom, comprennent des arguments et retournent un résultat
  - Syntaxe :

```
Function NomFunction ([arguments]) [As Type]
    Instructions de la fonction
End Function
```

  - Les arguments sont séparés par des virgules. La fonction retourne toujours une valeur portée par son nom

Programmer en VB

- Application
  - Ajouter une fonction au programme Lucky Seven pour calculer le taux de réussite.
  - Pour cela mettre en place une fonction taux et une variable publique appelée Jeux dans le module standard
  - Cette fonction sera appelée chaque fois que le bouton "jouer" est activé
  - Le résultat doit apparaître dans un nouveau label à placer sur la feuille

Programmer en VB

- Application
  1. Enregistrer le projet précédent sous le nom Reussite
  2. Créer une nouvelle étiquette Label sous Gains avec comme propriétés pour Label5 : Alignment (2-center), Caption (0,0%), Font (Arial, Gras Italique, 12 points), ForeColor (Rouge), Name(lblTaux)
  3. Dans la fenêtre Projet, double-cliquer sur Reussite.bas pour l'ouvrir dans la fenêtre code, saisir:

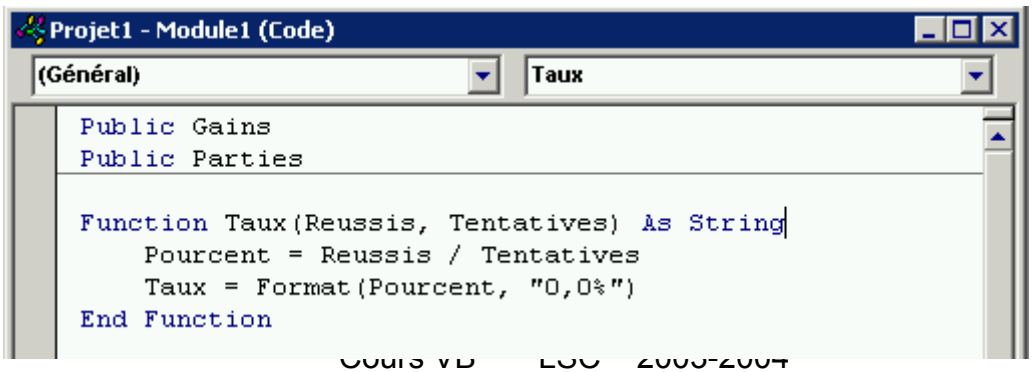
     Public Parties
  - Saisir la déclaration de fonction suivante dans le module standard

Programmer en VB

- Application
  - Ajouter les 2 instructions suivantes :
    - Celle là après la dernière instruction comprenant Rnd :

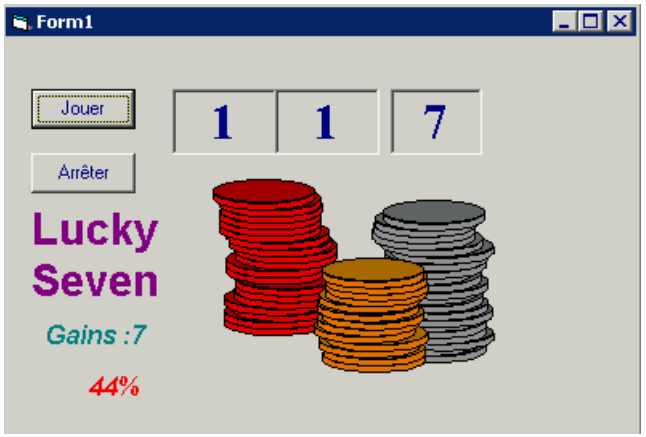
      Parties = Parties + 1
    - Celle là entre End If et End Sub :

      lbltaux.Caption = Taux(Gains, Parties)



Programmer en VB

- Résultat



Programmer en VB

- Les procédures Sub
  - Syntaxe :

```
Sub NomProcedure ([Argument])
    Instructions
End Sub
```

  - Les noms et nombre d'arguments doivent correspondre aux noms et nombre à l'appel

Programmer en VB

- Exemple :

```
Sub AjouteNomAliste(personne$)
If personne$ <> "" Then
    Form1.List1.AddItem personne$
    Msg$ = personne$ & "ajouté à la liste"
Else
    Msg$ = "Nom indéfini"
End If
MsgBox (Msg$), ,  "Ajoutde nom"
End Sub
```

Programmer en VB

- Appel
  - Il suffit d'écrire le nom de la procédure et de répertorier ses arguments :

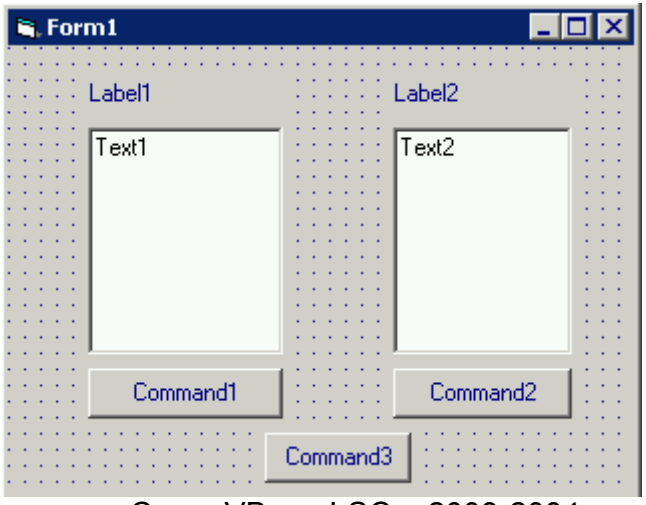
    AjouteNomAliste "Mariane"

    Ou

    AjouteNomAliste NouveauNom$

Programmer en VB

- Application
  - Créer l'interface dessinée ci-après. Les zones de texte contiendront les noms de salariés dans deux départements de l'entreprise :

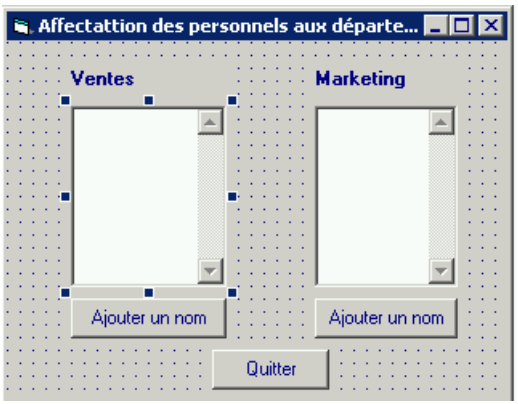


Programmer en VB

- Opérations
  - Text1 et Text2 : Text(vide), Multiline (True), Scrollbars (2-vertical), Tabstop (False), Locked (True), Name (txtVentes pour Text1 et txtMarketing pour Text2)
  - Label1 : caption(Ventes, Font (Gras), Name (lblVentes)
  - Label2 : caption(Marketing, Font (Gras), Name (lblMarketing)
  - Command1 : Caption(Ajouter un nom), Name(cmdVentes)
  - Command2 : Caption(Ajouter un nom), Name(cmdMarketing)
  - Command3 : Caption(Quitter), Name(cmdQuit)
  - Form1 : Caption (Affectation des personnels aux départements)

Programmer en VB

- Interface créée



Programmer en VB

- Ajouter un module standard, et saisir le code suivant

```
Sub AjouterNom(Team$, ChaineRetour$)
Prompt$ = "Saisissez un salarié de " & Team$
Nm$ = InputBox(Prompt$, "Boîte de saisie")
WrapCharacter$ = Chr(13) + Chr(10)
ChaineRetour$ = Nm$ & WrapCharacter$
End Sub
```

Projet1 - Form1 (Code)

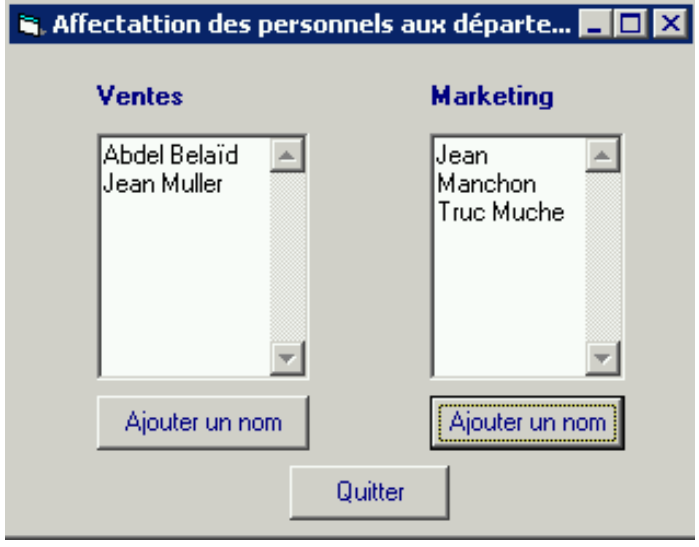
cmdQuit | Click

```
Private Sub cmdMarketing_Click()
AjouterNom "Marketing, MarketingPosition$"
txtMarketing.Text = txtMarketing.Text & _
MarketingPosition$
End Sub

Private Sub cmdQuit_Click()
End
End Sub

Private Sub cmdVentes_Click()
AjouterNom "Ventes", VentesPosition$
txtVentes.Text = txtVentes.txt & VentesPosition$
End Sub

Private Sub txtMarketing_Change()

End Sub
```

ants :

Programmer en VB

- Enregistrer le projet sous le nom : Départements et Exécuter



Programmer en VB

- Transmission des arguments
  - Par référence :
    - Tout changement effectué dans la procédure sur la variable est retourné à la procédure appelante. On peut accompagner une variable par le mot-clé ByVal pour forcer le passage par valeur.

      Sub CoutPlusInteret(ByVal Cout, Total)
  - Par valeur : il suffit de placer la variable entre parenthèses

      CoutPlusInteret(Cout), Total)

Programmer en VB

- Les tableaux
  - Déclaration d'un tableau statique

    Public NomTableau(Dim1, Dim2, …) As TypeDeDonnées
  - Exemple :

    Public Employés (9) As String
  - Lors de la création, VB réserve de l'espace en mémoire
  - La référence aux éléments se fait de 0 à 8
  - Cependant, si l'on veut que la référence se fasse à partir de 1, il suffit d'ajouter l'instruction suivante dans un module standard :

    Option Base 1

Programmer en VB

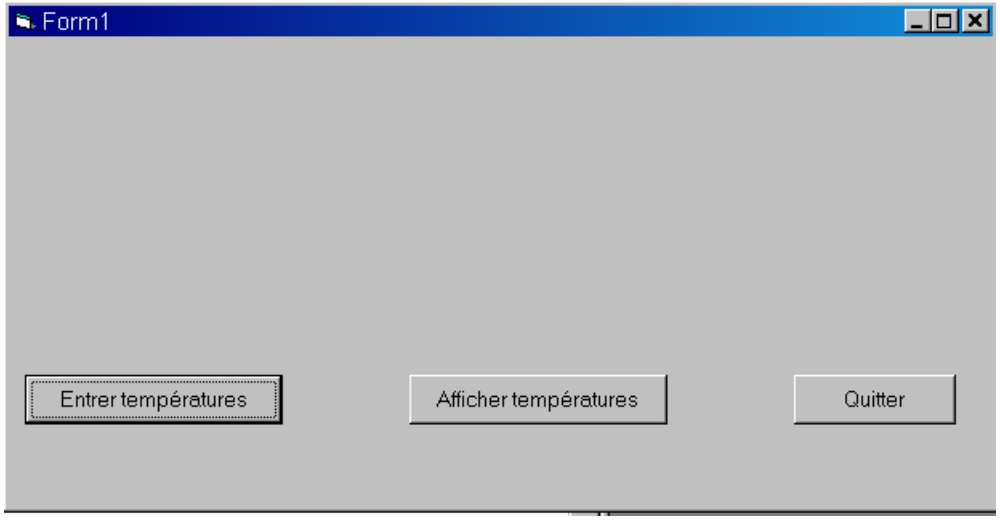
- Les tableaux
  - **Référence au tableau :**
    - Employés(5) = "Leslie"
  - Déclaration d'un tableau dynamique
    - La déclaration en statique empêche l'extension du tableau en cas d'enregistrements de valeurs supplémentaires. Pour résoudre ce problème, on déclare le tableau en dynamique, et on le redimensionne une fois connu le nombre d'éléments

```
Public Températures () As Variant
…
Days = InputBox("Combien de jours ,", "Créer le tableau")
….
Redim Températures(Days)
```

Programmer en VB

- Application
  - Créer un tableau de températures
    1. Créer trois boutons de commande, les placer en bas de la feuille, avec les propriétés suivantes :
       - Command1 : Caption(Entrer températures), Name (cmdEntrerTemps)
       - Command2 : Caption(Afficher Températures), Name (cmdAfficherTemps)
       - Command3 : Caption (Quitter), Name(cmdQuitter), Form1 : Caption (Températures), AutoRead(True) ;

Programmer en VB



Programmer en VB

2. Dans le menu Projet, cliquer sur la commande Ajouter un module, puis sur Ouvrir pour créer un module standard destiné à la déclaration du tableau

   Un module standard apparaît dans la fenêtre Code

3. Dans le module standard, saisir les instructions :

   Option Base 1

   Public Températures(7) As Variant

   - Public permet de rendre le Tableau global et donc disponible dans tout le programme

Programmer en VB

4. Entrer le code suivant pour la commande cmdEntrerTemps_Clik (procédure événementielle rattachée au bouton "Entrer Températures"

```
Cls 'efface l'écran
Prompt$ = "Entrer la température la plus haute"
For i% = 1 To 7
Title$ = "Jour" & i%
Températures(i%) = InputBox(Prompt$, Title$)
Next i%
```

Programmer en VB

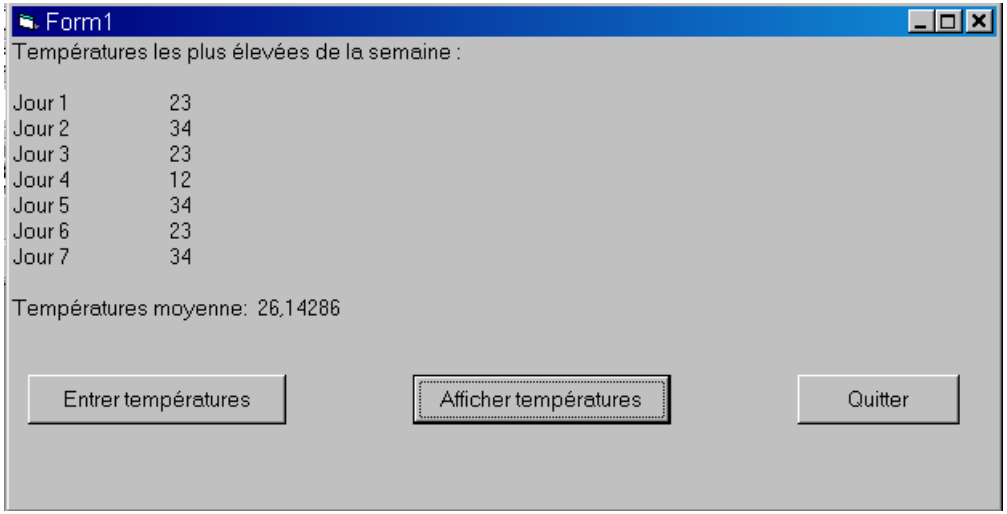
5. Entrer le code suivant pour la commande cmdAfficherTemps_Clik (procédure événementielle rattachée au bouton "Entrer Températures"

```
Print "Températures les plus élevées de la semaine :"
Print
For i% = 1 To 7
Print "Jour"; i%, Températures(i%)
Total! = Total! + Températures(i%)
Next i%
Print
Print "Températures moyenne: "; Total! / 7
```

6. Mettre End dans la procédure Quitter

Programmer en VB

- Résultat



Programmer en VB

- Affichage du tableau dans une zone de texte
  - Déclarer une zone de texte avec la commande TextBox
  - Soit Text1 le nom de cette zone
  - Soit Employes le nom du tableau
  - On écrit :

```
Text1.Text = "" //initialisation à blanc
For i% = Tete To Queue
Text1.Text = Text1.Text & " " & Employes(i)
Next i%
```

Programmer en VB

- Quelques remarques
  - Déclaration de variables publiques
    - Ouvrir un Module pour le faire
  - Initialisation de ces variables
    - Se mettre dans Form1
    - Cliquer sur le coin en haut à droite
    - Cliquer sur Initialize
      - Création d'une subroutine Private Sub Form_Initialize() dans laquelle on fait l'initialisation