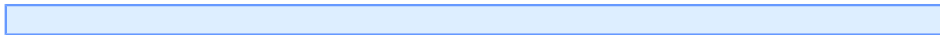


# Tutoriel 3 : L'environnement de programmation de Visual Basic

par Gilbert Miralles ([gilmir.developpez.com](http://gilmir.developpez.com))

Date de publication : Lundi 18 mars 2003

Dernière mise à jour : Lundi 28 janvier 2008



- I - Faisons connaissance avec le langage de programmation objet
  - I-A - Généralités
  - I-B - Ecriture du code BASIC
  - I-C - Les procédures
- II - La fenêtre principale ( Interface graphique de conception)
  - II-A - 3 barres horizontales
  - II-B - La boîte à outils
  - II-C - La fenêtre de projet
  - II-D - La fenêtre de propriétés
  - II-E - Procédures
- III - Avançons dans l'application
  - III-A - Un tout petit peu de Théorie
    - III-A-1 - Méthodes
    - III-A-2 - Programmation événementielle
    - III-A-3 - Domaine de validité des variables et des procédures
    - III-A-4 - Compilation
    - III-A-5 - Les types de données
    - III-A-6 - Quelques précisions
    - III-A-7 - Qu'est ce qu'une variable ?
    - III-A-8 - Déclarations de variables
    - III-A-9 - Propriétés d'un objet
  - III-C - Première lignes de code

Prochain tutoriel :

## I - Faisons connaissance avec le langage de programmation objet

### I-A - Généralités

#### **Visual Basic, est en parfaite harmonie avec Windows. Le développement d'une application passe par les étapes suivantes :**

- Dessin de l'interface d'utilisation, c'est à dire les fenêtres et leur constituants, à l'aide d'un outil interactif de dessin, "**l'environnement Visual Basic**".
- Valorisation initiale des propriétés qui sont des attributs ou caractéristiques de chaque élément de l'interface.

### I-B - Ecriture du code BASIC

La programmation des applications Visual Basic est dite événementielle, par opposition à la programmation linéaire traditionnelle.

Ainsi, une application Visual Basic est constituée d'un ensemble de procédures indépendantes les unes des autres.

### I-C - Les procédures

Une procédure comprend des instructions écrites à l'aide du langage BASIC. Elle est associée à un objet, c'est à dire à un des éléments d'une feuille, la feuille elle même, ou bien un bouton, une liste, un champs de saisie etc...

La procédure est appelée par Visual Basic lorsqu'il se produit un événement pour l'objet correspondant.

Si vous n'avez pas écrit de code dans une procédure chargée de traiter un type d'événement pour un objet donné, il ne se passe rien de particulier lorsque l'événement est généré.

Pour écrire le code d'une application, il convient donc de déterminer les événements auxquels on souhaite réagir, et pour quels objets.

Cela détermine les procédures dans lesquelles le code est écrit.

## II - La fenêtre principale ( Interface graphique de conception)

### II-A - 3 barres horizontales

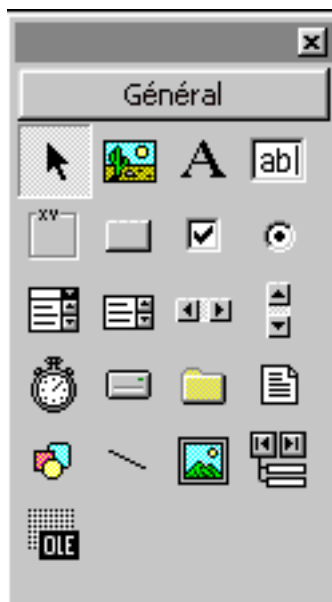
**La fenêtre principale comprend trois barres horizontales :**

- **la barre de titre**, comme toute application Windows.
- **la barre de menu**, permettant la saisie des commandes, et
- **la barre d'outils** donnant un accès rapide aux principales commandes



### II-B - La boîte à outils

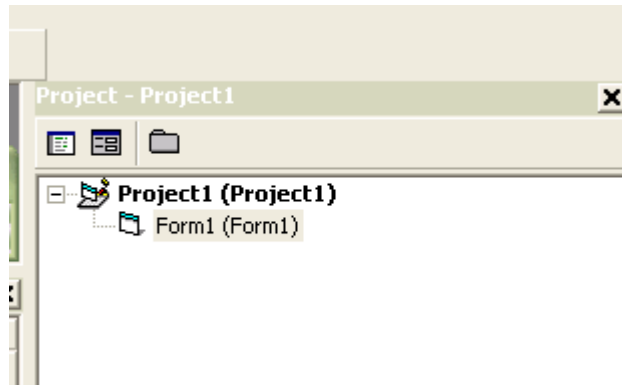
La boîte à outils, est une fenêtre qui comprend initialement tous les icônes visualisant des contrôles personnalisés (application complémentaire dont certains contrôles sont en option selon la version que vous possédez) pour la version standard 20 icônes.



Elle n'est accessible qu'en phase de conception (Mode Arrêt)

### II-C - La fenêtre de projet

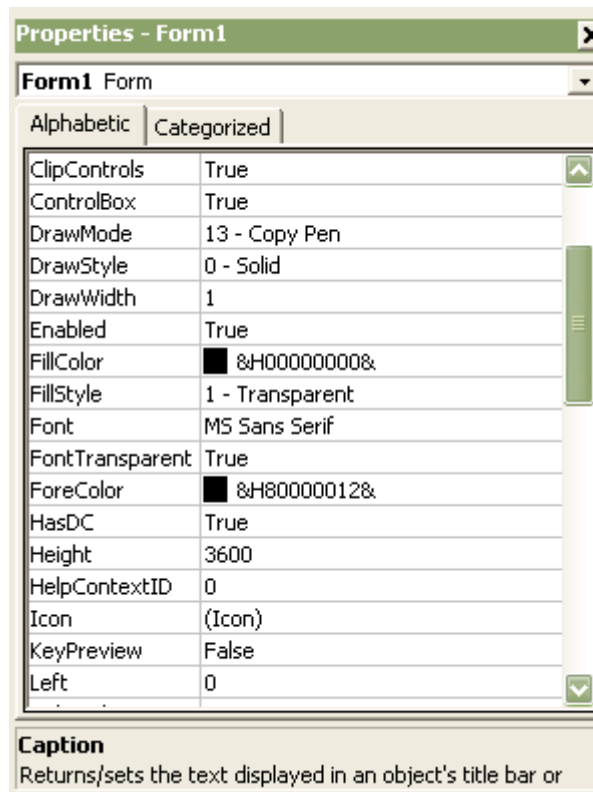
La fenêtre de projet comprend la liste des divers fichiers constituant une application.



*(Pour l'instant nous n'avons qu'un seule fichier "Form1")*

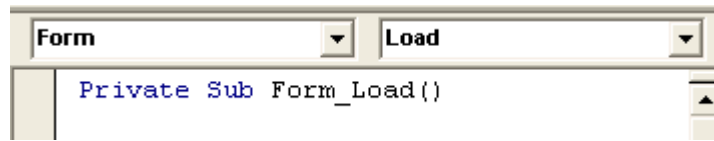
## II-D - La fenêtre de propriétés

La fenêtre de propriétés comprend la liste des propriétés de l'objet sélectionné dans la feuille courante (feuille ou contrôle) ainsi que leurs valeurs.



## II-E - Procédures

En haut la liste des objets, présent définis dans le fichier sélectionné dans la fenêtre projet et à droite la procédure appelée par Visual Basic lorsqu'il se produit un événement pour l'objet correspondant



```
Form Load
Private Sub Form_Load()
```

### III - Avançons dans l'application

A ce stade de l'application, nous n'avons créé qu'une seule fenêtre, c'est donc la fenêtre principale.

La fenêtre principale que j'appelle l'interface d'accueil est la feuille que l'utilisateur voit en premier lorsque le programme est lancé.

Nous pouvons construire autant de feuilles que l'on veut et déterminer par la suite qu'elle est la feuille qui deviendra l'interface d'accueil.

Cette possibilité s'obtient par le paramétrage de l'environnement de travail de Visual Basic.

Cliquez dans la barre de menus sur l'étiquette "Projet", puis sur "Propriétés de projet1", sélectionnez "Objet de démarrage".

Si le champs de saisie comprend la valeur "Sub main", modifiez et écrivez "Form1" qui est le nom de la feuille que nous avons créée et sur laquelle nous sommes en train de travailler.

Par la suite nous modifierons la valeur de l'étiquette pour saisir le nom de la feuille que nous souhaitons lancer au démarrage du programme.

Avant de nous lancer dans la super production qui va nous tenir en haleine pendant quelques heures, je me dois de vous instruire de quelques notions qui me paraissent indispensable pour la bonne compréhension de la suite du programme.

#### III-A - Un tout petit peu de Théorie

##### III-A-1 - Méthodes

Sous Visual Basic, certaines fonctions ne sont accessibles que combinées avec des objets. Ces instructions sont dénommées méthodes. Par exemple la méthode **Show** charge et affiche une fenêtre définie par l'utilisateur (vous en l'occurrence) (se référer aux sources pour les autres méthodes)

##### III-A-2 - Programmation événementielle

Les programmes Visual Basic sont commandés par les événements générés par les objets de l'interface utilisateur.

Par exemple, le chargement d'une feuille est un événement auquel certaines instructions peuvent être rattachées., comme par exemple l'effacement d'une feuille dans la mémoire.

L'événement sera causé par un clic de souris, dans le cas d'un bouton, ou par une modification du contenu d'une zone de texte.

##### III-A-3 - Domaine de validité des variables et des procédures

Comme tout langage moderne, Visual Basic connaît les variables globales, locales et statiques.

Les variables globales sont accessibles depuis le programme entier mais les variables locales ne sont connues que dans leur procédure. Les variables statiques conservent leur valeur à la sortie de la procédure, ce qui les distingue des variables locales.

### III-A-4 - Compilation

Quand un projet a été compilé, son extension devient EXE, et il n'est plus possible de le "démonter" : tous les fichiers qui le composent sont désormais intimement associés.

Les images installées dans un contrôle sont elles aussi intégrées à l'exécutable. Seuls les fichiers OCX ou TXT restent autonomes, et devront être livrés à l'utilisateur.

Les fichiers DLL devront également être fournis avec l'application.

### III-A-5 - Les types de données

Suffixe	Type de données	Taille	Limites d'utilisation
%	Integer	2 oct	de -32.768 à 32767
&	Long	4 oct	de -2.147.483.648 à 2.147.483.647
!	Single (à virgule flottante en simple précision)	4 oct	entre -3,402823E38 et -1,401298E-45 pour les nombres négatifs  et entre 1,401298E-45 et 3,402823E38 pour les positifs
#	Double (à virgule flottante en double précision)	8 oct	entre -1,79769313486232E308 et -4,94065645841247E-324 pour les nombres négatifs  et entre 4,94065645841247E-324 et 1,79769313486232E308 pour les positifs
@	Currency (Virgule fixe 15 chiffres à gauche du séparateur décimal et 4 chiffres à droite )	8 oct	entre -922 337 203 685 477,5808  et 922 337 203 685 477,5807
\$	String	1 oct	par caractère
rien	Variant	nombre requis	

### III-A-6 - Quelques précisions

De ce tableau on retient que si l'on doit déclarer une variable alphanumérique (caractères textes) on choisira une variable de type **String**

Pour une opération avec des chiffres ou des opérations de comptage simples, on choisira une variable numérique de type **Integer**, ou **long** pour des opérations importantes (voir limites).

Pour les calculs monétaires, on choisira la variable de type **Currency**.

Le type par défaut d'une variable est **VARIANT**. Ce type indique à Visual Basic que la donnée est susceptible de contenir divers types de données.

Je déconseille d'utiliser la variable de type **Variant**, sauf cas de force majeure.

### III-A-7 - Qu'est ce qu'une variable ?

Les instructions sont constituées de verbes qui agissent sur des variables.

Une variable peut être considérée comme une case mémoire. Elle possède un nom et une valeur.

Le nom est constitué de lettres, de chiffres et de caractères `<_>` (le souligné), le premier caractère est obligatoirement une lettre et il ne peut y avoir plus de 40 caractères dans un nom de variable.

### III-A-8 - Déclarations de variables

La déclaration d'une variable se fait de plusieurs façons , à savoir :

**Dim** <nom de la variable> As <type de la variable>

Ex : pour déclarer une variable de type String (alphanumérique), que vous avez nommée <Bazile>, vous écrirez

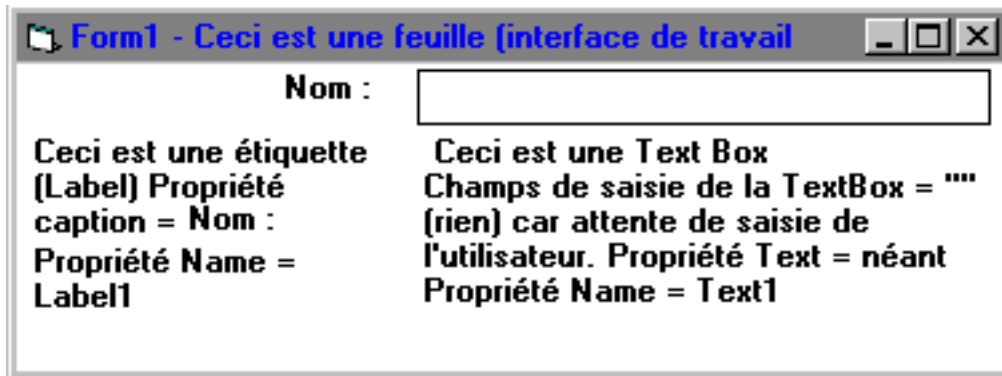
**Dim** Bazile As String représentée dans le programme par <Bazile\$>

Si la même variable est de type Integer (numérique), vous écrirez :

**Dim** Bazile As Integer représentée dans le programme par <Bazile%>

### III-A-9 - Propriétés d'un objet

Exemple d'un champs de saisie comportant une étiquette ayant comme **propriété Caption** : Nom, et **comme propriété Name** : Label1.



Ne pas confondre la propriété Name qui est le nom par défaut de l'objet, et la propriété Caption qui peut avoir n'importe quel nom et qui dans le cas présent est le Nom de l'utilisateur à saisir. Vous mettrez un certain temps à vous y faire, mais, vous comprendrez avec un peu d'expérience.

La propriété Name est le nom que vous donnez à l'objet dans la phase de développement. Cette propriété n'est pas visible par l'utilisateur. Par contre la propriété Caption est le nom que vous donnez à la valeur de l'étiquette, ici dans l'exemple c'est Nom : , mais cela aurait pu être <Adresse> ou <Ville> etc...et cette propriété est visible par l'utilisateur, mais au contraire de la TextBox celle-ci n'est pas modifiable.

### III-C - Première lignes de code

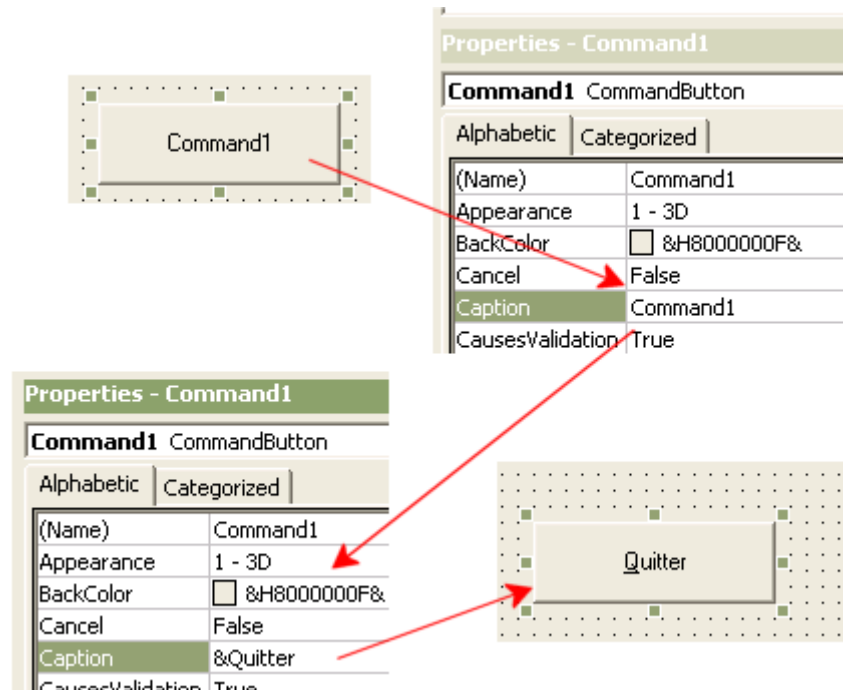
Nous allons essayer de mettre en pratique les formules précédemment acquises.

Lorsque vous travaillez sur l'interface de travail vous vous trouvez en mode développement.

Vous pouvez contrôler le bon déroulement des instructions ou fonctions écrites dans les pages de code en passant en mode exécution.

Nous allons écrire notre première ligne de code et visualiser le fonctionnement en mode exécution.

Ecrire dans la propriété Caption de bouton de commande "Command1" &Quitter.



Nous allons nous servir à titre de démonstration de ce bouton pour quitter notre application.

En effectuant un "DoubleClick" sur le bouton, nous ouvrons une feuille qui est la feuille de code de l'objet qui vient d'être sélectionné.



*Vu de la feuille de code correspondante au bouton de commande "Command1"*

Cette feuille se présente avec une "ComboBox"(Command1) qui vous permet en cliquant sur le petit bouton de droite de dérouler une liste qui comprend tous les noms d'objets utilisés dans cette feuille(Form1).

La "ComboBox de droite(Click) liste toutes les propriétés disponibles pour l'objet sélectionné.

Nous devons générer un événement lorsque l'utilisateur appuiera sur le bouton, et lorsqu'il générera un clic avec le curseur de la souris.

Nous sélectionnerons donc l'événement "Click".

Ecriture du code dans la feuille de code du bouton de commande "Command1" avec l'événement "Click"

Nous écrivons notre première procédure en saisissant:

```
Private Sub Command1_Click ( )  
'Sortie du programme  
End  
End Sub
```

Vous avez vu que j'ai mis une apostrophe devant la ligne de texte "'Sortie du programme"

Ces lignes commençant par une apostrophe sont nommées "lignes de commentaires. Il est conseillé en effet d'indiquer sous forme de texte l'explication de l'événement que l'on a voulu effectuer.

Cela sera très utile surtout lorsque notre programme sera composé de plusieurs dizaines de procédures.

Ne pas s'en priver d'autant que le compilateur ignore complètement les lignes de commentaires.

Si vous êtes un peu curieux (il faut l'être en programmation) et que vous avez cliqué sur "Général" dans la feuille de code vous constatez que l'interface comprend une instruction "Option Explicite".

Cette instruction force le développeur à déclarer toutes les variables à utiliser dans son programme.

Si vous oubliez de déclarer une variable en phase de conception, le programme vous rappellera à l'ordre et vous indiquera une erreur.


Puisque nous venons de créer notre première procédure, essayons de la lancer pour visualiser ce qu'il se passe.

Dans la barre d'outils, cliquez sur le bouton exécuter



formé d'un petit triangle.

La feuille doit s'afficher, et lorsque vous cliquez sur le bouton "Quitter", l'application doit se fermer.

 **Visual Basic écrit sous la couleur bleue tous les mots dits "Réservés", vous ne pouvez pas les utiliser en dehors de la fonction établie par l'environnement de programmation.**

La prochaine leçon sera réservée à la création de tableaux.

Prochain tutoriel :

 ***Les tableaux***

