

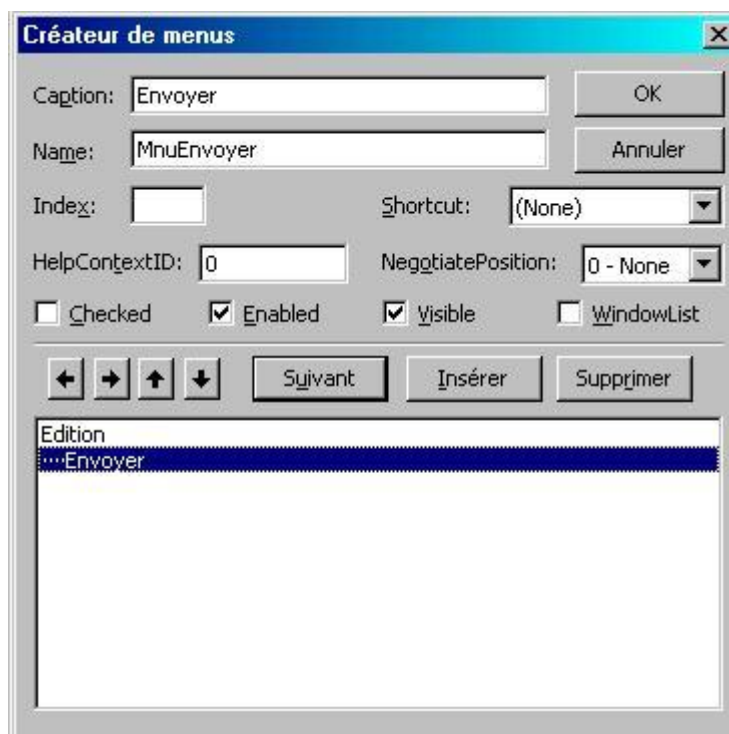
## FICHE TECHNIQUE N° III (Suite – 2/2)

- Chapitre : ELEMENTS DU LANGAGE -

### LES MENUS

Visual Basic intègre un éditeur de menu. Celui ci permet de créer des menus contextuels personnalisés pour vos programmes Visual Basic. Pour le lancer, cliquez sur "Créateur de menu" dans le menu contextuel "Outils" du logiciel "Visual Basic".

Dans l'exemple ci-dessous, on crée un menu "édition" et le sous-menu "envoyer" du menu "édition".



La propriété "caption" permet de renseigner l'étiquette du menu (ex : fichier, édition, aide ...). La propriété "name" permet de renseigner le nom du contrôle "Menu" permettant de l'identifier. Le seul événement disponible pour le contrôle "Menu" est "Click". Les instructions seront donc écrites dans la procédure NomContrôleMenu\_Click. Les instructions seront exécutées lorsque l'utilisateur cliquera sur le menu ou le sous-menu correspondant.

Il est également possible de créer des sous-menus. L'opération de création est la même. Il faut juste spécifier que le menu est un sous-menu en cliquant sur le bouton flèche. Trois petits points apparaissent devant l'étiquette du menu dans la liste du créateur de menu. Cela signifie



que c'est un sous menu du menu placé immédiatement au dessus dans la liste du créateur de menu.

Sur l'exemple ci-dessus, "Envoyer" est un sous-menu du menu "Edition".

Astuce : L'utilisation du caractère "&" devant une lettre, dans la propriété "caption" du menu, permet de créer un raccourci clavier sur la lettre pour actionner le menu. Le menu est actionnable en mode exécution en utilisant la combinaison de touche Alt + "lettre".



## LES MODULES

Un module est un sous-programme pouvant être utilisé dans plusieurs programmes Visual Basic. Un module contient des déclarations de variables, des structures de données et des instructions au même titre qu'un programme. L'utilisation des modules dans les programmes permet d'éviter la redondance de codes. Les modules peuvent être paramétrés au même titre que les fonctions et les procédures. Un module est en fait une fonction ou une procédure (selon qu'il retourne ou non un résultat) indépendant du programme principal et pouvant être utilisé dans d'autres programmes. Puisqu'il est extérieur au programme principal, un module est sauvegardé sous la forme d'un fichier.

Il existe deux types de module :

- Les modules standards (utilisés les plus fréquemment),
- Les modules de classe (orientés objet).

Pour utiliser un module dans un programme (encore appelé projet), il faut l'ajouter à ce projet. Pour cela, il faut cliquer sur "Ajouter un module" ou "Ajouter un module de classe" (selon le type de module choisit) dans le menu contextuel "Projet". Là on a le choix entre créer un nouveau module ou ouvrir un module existant. Une fois le nouveau module sauvegardé ou le module existant ouvert, ce module est ajouté au projet et est utilisable dans tous le programme.



## LES FONCTIONS INTEGREES

Comme dans tous langage de programmation, Visual Basic intègre un grand nombre de fonctions intégrées. Une fonction intégrée est écrite dans le compilateur et est donc utilisables directement sans être déclarées. Ces fonctions sont par exemple des fonctions mathématiques, des fonctions de chaînes, des fonctions systèmes ou autres. La plupart du temps elles nécessitent des paramètres d'entrées que le programmeur doit renseigner.

Pour connaître ces différentes fonctions, il faut ce documenter. Généralement leurs syntaxes en Visual Basic sont assez proches de celle des autres langages.

Voici quelques exemples :

Date	Renvoie la date système
Time	Renvoie l'heure système
Avg	Moyenne arithmétique d'une série de valeur
Len	Nombre de caractère d'une chaîne
Round	Renvoie l'arrondi d'une valeur fractionnaire
Rnd	Renvoie un entier aléatoire
Format	Formatage d'une expression
Lcase	Convertit une chaîne en minuscules
Ucase	Convertit une chaîne en majuscule
VbCrLf	Génère un retour chariot dans une chaîne de caractères
StrComp	Compare une chaîne à une autre

## LES FONCTIONS GRAPHIQUES

Il existe deux types de fonctions graphiques :

- MsgBox : Permet d'afficher un message dans une boîte de dialogue,
- InputBox : Permet de saisir une valeur dans une boîte de dialogue.

### La MsgBox

Syntaxe : MsgBox "Message" , Type du bouton, " Libellé de la boîte de message"

Différents types de boutons sont disponibles, les plus courants sont VBOkOnly (bouton "OK") et VBYesNo (bouton "Yes" et bouton "No").

Remarque : si on utilise l'option VBYesNo, on peut faire exécuter telle ou telle instruction en fonction du bouton cliqué par l'utilisateur.

Exemple : Dans le code suivant : "message" est le nom de la boîte de message, si l'utilisateur clique sur "Ok" alors il y a un bip.

```
message = MsgBox("Voulez vous quitter l'application ?", vbYesNo, "VB generator")  
If message = vbYes Then Beep
```

Donne la boîte de dialogue suivante :



### L' InputBox

Syntaxe : NomVariable = InputBox("Message de l'InputBox", "Titre de l'InputBox")

Le texte qui sera saisi par l'utilisateur dans l'InputBox sera placé dans la variable "NomVariable".

Exemple : Dans le code suivant le texte "VB Generator" saisi par l'utilisateur est placé dans la variable "var" et sera donc utilisable pour un traitement par le programme.

```
var = InputBox("Entrer votre nom", "VB Generator")
```

Donne la boîte de dialogue suivante :



## LES CARACTERES SPECIAUX

En VB, certains caractères ont une signification particulière :

' : permet d'écrire des commentaires, tout ce qui est écrit derrière l'apostrophe et jusque la fin de la ligne est considéré comme commentaire et est donc ignoré par le compilateur. Toute la ligne de commentaire s'affiche alors en vert.

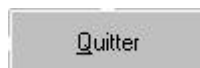
Exemple : `'Ceci est une ligne de commentaire`

\_ : permet d'écrire une instruction sur plusieurs lignes. En VB, une instruction est écrite sur une ligne, lorsque l'on passe sur la ligne suivante le compilateur attend une nouvelle instruction. Cependant, lorsque l'instruction est longue, pour la clarté du programme, il est intéressant de l'écrire sur deux lignes grâce au caractère "\_". Le caractère se place à l'endroit où l'on veut couper l'instruction et doit être précédé d'un espace. La seconde ligne sera interprétée par le compilateur comme la suite de l'instruction de la ligne précédente.

Exemple : `MsgBox "Bonjour les internautes" _  
 , vbOKOnly, "VB Generator"`

Attention, on ne peut pas couper une ligne au milieu d'une chaîne de caractères ou d'un mot clé.

& : utilisé dans la propriété "caption" des boutons de commande ou du menu, il permet de créer un raccourci clavier pour actionner le bouton de commande ou le menu. Cette fonctionnalité est utilisable en mode "exécution" et en mode "conception". Exemple : on crée un bouton "quitte" dont la propriété "Caption" est "&Quitte". Le bouton s'affiche de cette manière :



De cette manière, le bouton "Quitte" est actionnable en utilisant la combinaison de touche Alt + Q.

& + *code hexadécimale* : permet d'annoncer que le code qui va suivre est de l'hexadécimal, il est très utilisé pour la spécification des couleurs dans le code.

Exemple : Le code suivant permet de changer la couleur de fond de la Form en bleu.  
`Form1.BackColor = &H8000000D.`

& : est aussi l'opérateur de concaténation.

"" : Chaîne vide = aucun caractère.

< ; > : Inférieur, supérieur.

= Affectation, égalité.



( ) : Parenthèses arithmétiques et alphanumériques (utilisées par exemple pour spécifier les paramètres d'une fonction, d'une procédure ou d'un module).

1, 0 : Signifie Vrai, Faux dans une instruction booléen.

Exemple : Check est une case à cocher, l'expression suivante :

Check value = 1 signifie que la case est cochée, car 1 = "True" = "Vrai"

+ ; - ; \* ; / : Opérateurs arithmétiques. Dans l'ordre : plus, moins, multiplier et diviser.

<> : Signifie "différent de".

![ ] : Utilisé pour encadrer les champs en base de données (voir dans la section accès aux données).

Exemple : "RstChargement" est le nom du jeu d'enregistrement,

"REUN\_LIB" est le nom du champs.

L'accès au champs s'effectue grâce à la syntaxe suivante : *RstChargement![REUN\_LIB]* où

bien par

*RstChargement("REUN\_LIB")*

VbCrLf : fonction qui génère un retour chariot, c'est à dire un retour à la ligne dans une chaîne de caractères.