



FICHE TECHNIQUE N° VI (Suite – 2/3)

LA FENETRE D'EDITION DE CODE

Cette fenêtre permet, comme son nom l'indique, d'éditer le code Visual Basic que l'on place derrière les objets de l'interface. A chaque feuille objet (encore appelé Form) correspond une feuille de code. A tous moments du développement du programme, il est possible de basculer entre le mode graphique (affichage de l'interface) et le mode édition de code (affichage du code) en cliquant sur "Objet" ou sur "Code" dans le menu contextuel "Affichage".

Dans cette fenêtre, le code est structuré en plusieurs parties. En effet, elle contient :

- Un module général de déclaration,
- Une ou plusieurs procédures événementielles,
- Eventuellement des procédures et des fonctions.

Le module général de déclaration :

C'est dans le module général de déclaration que l'on déclare les variables et les constantes publiques de la feuille, c'est à dire les variables et les constantes que l'on pourra utiliser dans toute la feuille. C'est aussi dans ce module général de déclaration que l'on peut inscrire l'instruction "Option Explicit" permettant de forcer la déclaration des variables dans toute la feuille (pour plus d'informations sur l'option "Explicit", reportez vous à la section "Les variables". Le module général de déclaration est accessible en choisissant " Général" dans la ComboBox en haut à gauche et en choisissant "Déclaration" dans la ComboBox en haut à droite.

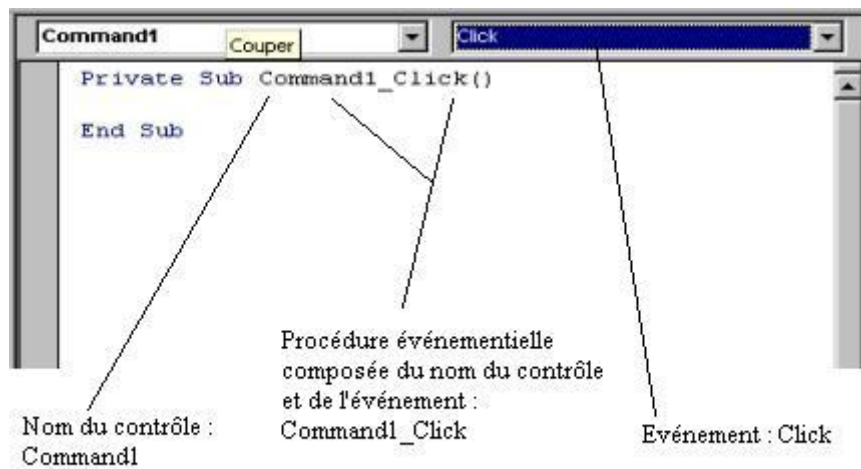
Remarque : ce module est placé tous en haut de la feuille de code, il ne peut rien y avoir avant.

Les procédures événementielles :

Les instructions déclenchées par un événement (sur un objet graphique) sont donc enregistrées dans des procédures événementielles dont la structure est la suivante :

```
Private Sub NomDuContrôle_NomDeL'Événement ()  
    (suite d'instructions)  
End Sub
```

Exemple : on crée la procédure événementielle Command1_Click ()



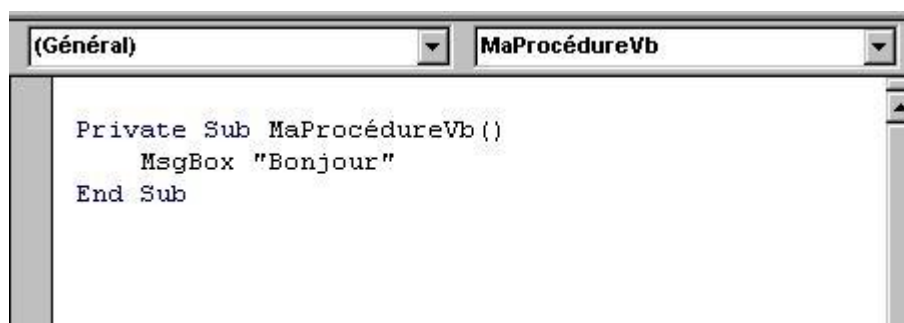
Pour créer ou accéder à une de ces procédures, il faut sélectionner le contrôle souhaité dans la ComboBox en haut à gauche et l'événement souhaité dans la ComboBox en haut à droite. Si la procédure existe déjà, elle s'affichera à l'écran, sinon elle sera créée et il ne vous restera plus qu'à écrire les instructions à l'intérieur.

Remarque : Toutes les procédures événementielles d'une feuille sont privées à cette feuille, et commencent donc tous par les mots réservés "Private Sub" et finissent par le mot clé "End Sub".

Les procédures et fonctions éventuelles :

Un feuille peut contenir aussi des procédures classiques et des fonctions écrites par les utilisateurs. Pour plus d'informations sur la création des fonctions et des procédures, reportez vous aux chapitres du même nom. Une fois créé, ces fonctions et procédures seront accessibles en choisissant " Général" dans la ComboBox en haut à gauche et en choisissant le nom de votre procédure ou de votre fonction dans la ComboBox en haut à droite.

Exemple :





LA FENETRE D'EXECUTION

La fenêtre d'exécution permet d'afficher des informations relatives au programme pendant son exécution. Par exemple, on peut afficher la valeur que reçoit une variable pendant l'exécution du programme.

Pour afficher cette fenêtre, il faut sélectionner "Fenêtre d'exécution" dans le menu contextuel "Affichage". Elle apparaît en bas de l'écran et bien sûr reste affichée en mode exécutable mais n'est utilisable que dans l'environnement de développement Visual Basic, c'est à dire que l'on ne peut pas l'utiliser dans un programme exécutable compilé.

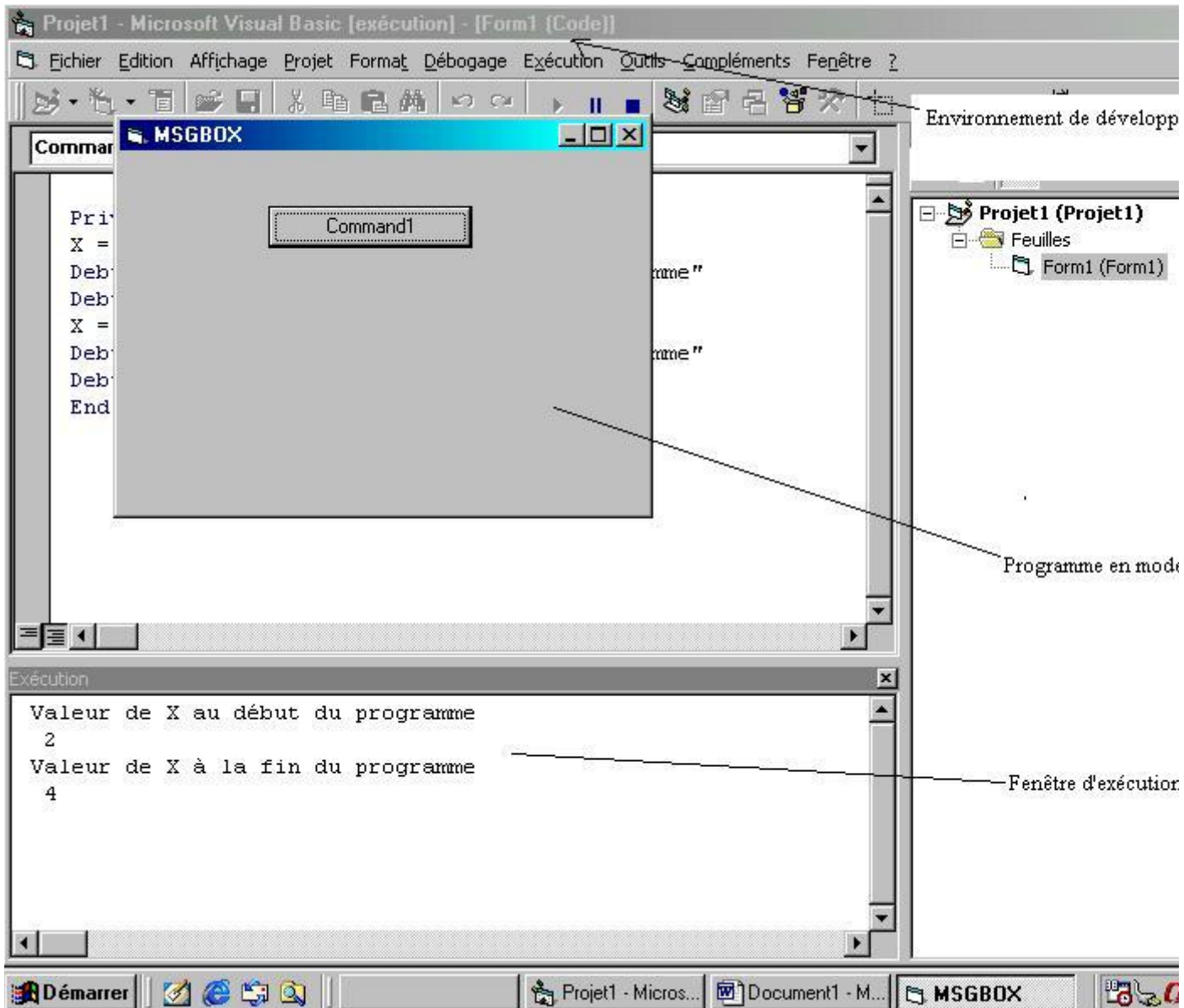
Pour afficher une valeur lors de l'exécution d'un programme, il faut envoyer cette valeur dans la fenêtre d'exécution. Pour cela, on utilise la propriété "Print" de l'objet "Debug".

Exemple : Debug.Print "Bonjour" 'Affiche Bonjour dans la fenêtre d'exécution

Exercice : on a une variable X de type entier dont la valeur initiale est 2, puis on ajoute 2 à cette variable. On demande l'affichage de la variable avant et après modification dans la fenêtre d'exécution.

```
X = 2                                     'Valeur initiale
Debug.Print "Valeur de X au début du programme"   'Afficher intitulé
Debug.Print X                                 'Afficher la valeur de X à ce stade
X = X + 2                                     'Valeur finale
Debug.Print "Valeur de X à la fin du programme"   'Afficher intitulé
Debug.Print X                                 'Afficher la valeur finale de X
```

Ce qui donne l'écran suivant :





L'OUTIL DE DEBOGAGE

Visual Basic intègre un certain nombre d'outils de débogage. Les outils de débogage permettent au programmeur de contrôler le bon fonctionnement du programme en mode exécution. Tous les outils de débogage sont disponibles dans le menu contextuel "Débogage".

L'outil de débogage le plus utilisé est le point d'arrêt. Le point d'arrêt se place devant une ligne exécutable dans la fenêtre d'édition de code. Il permet de stopper l'exécution du programme à la ligne devant laquelle il est placé. Tous comme la fenêtre exécution, l'utilisation du point d'arrêt n'est possible que dans l'environnement de travail, un programme compilé ne prend pas en compte les points d'arrêts.

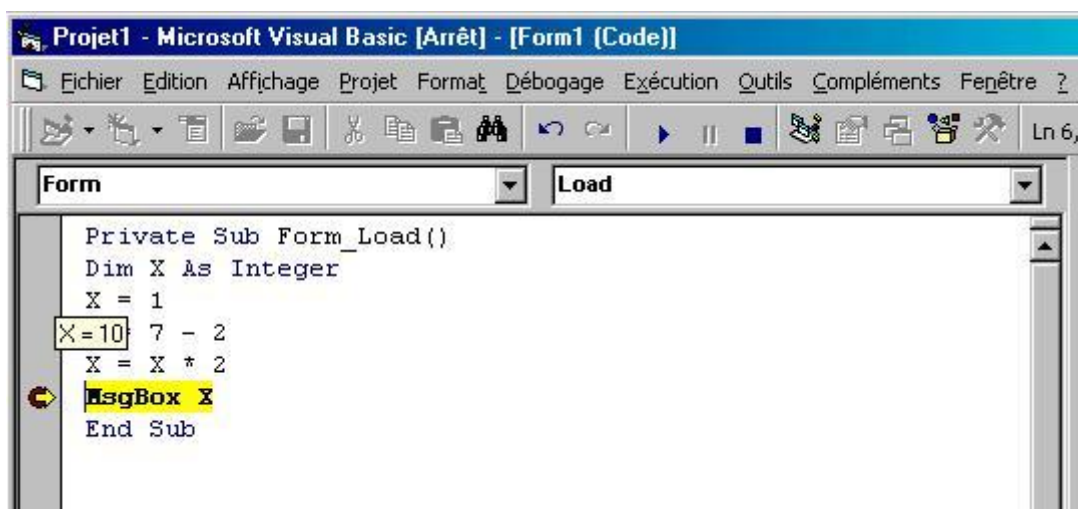
Pour placer un point d'arrêt, il faut simplement cliquer dans la marge devant la ligne où l'on souhaite le placer en mode édition dans la fenêtre d'édition de code. Un point rouge apparaît, c'est le point d'arrêt.

Remarque : on ne peut le placer que devant une ligne exécutable, on ne pourra donc pas le placer devant une déclaration de variable. Pour enlever un point d'arrêt, il faut simplement cliquer dessus.

Fonctionnement :

Lorsque l'on passe en mode exécution, le compilateur s'arrête au point d'arrêt, affiche la fenêtre de code et met la ligne correspondante en surbrillance couleur jaune. A partir de ce moment, le programmeur peut par exemple consulter la valeur d'une variable dans la partie du programme déjà exécutée (code écrit au-dessus de la ligne en surbrillance) en positionnant le curseur dessus (la valeur de la variable apparaît alors en info bulle).

Aperçu :





Remarques :

- le point d'arrêt est le point rouge dans la marge,
- Le compilateur s'est arrêté à la ligne où il y a le point d'arrêt,
- La valeur de X au moment où le compilateur s'est arrêté est 10.

Pour reprendre l'exécution du programme, il y a plusieurs possibilités :

- soit on reprend l'exécution jusqu'à la fin du programme ou jusqu'au prochain point d'arrêt si il y en a un,
- soit on utilise la commande "pas à pas détaillé" dont le raccourci clavier est "F8", dans ce cas le compilateur va s'arrêter après chaque instruction jusqu'à la fin du programme. Pour passer à l'instruction suivante, tapez sur "F8" à chaque fois.