Débuts en VBA sous Access

par

Date de publication : 01/04/2007

Dernière mise à jour :

Comprendre les fonctionnement des procédures et fonctions sous Access en VBA.

I - Mes débuts en VBA

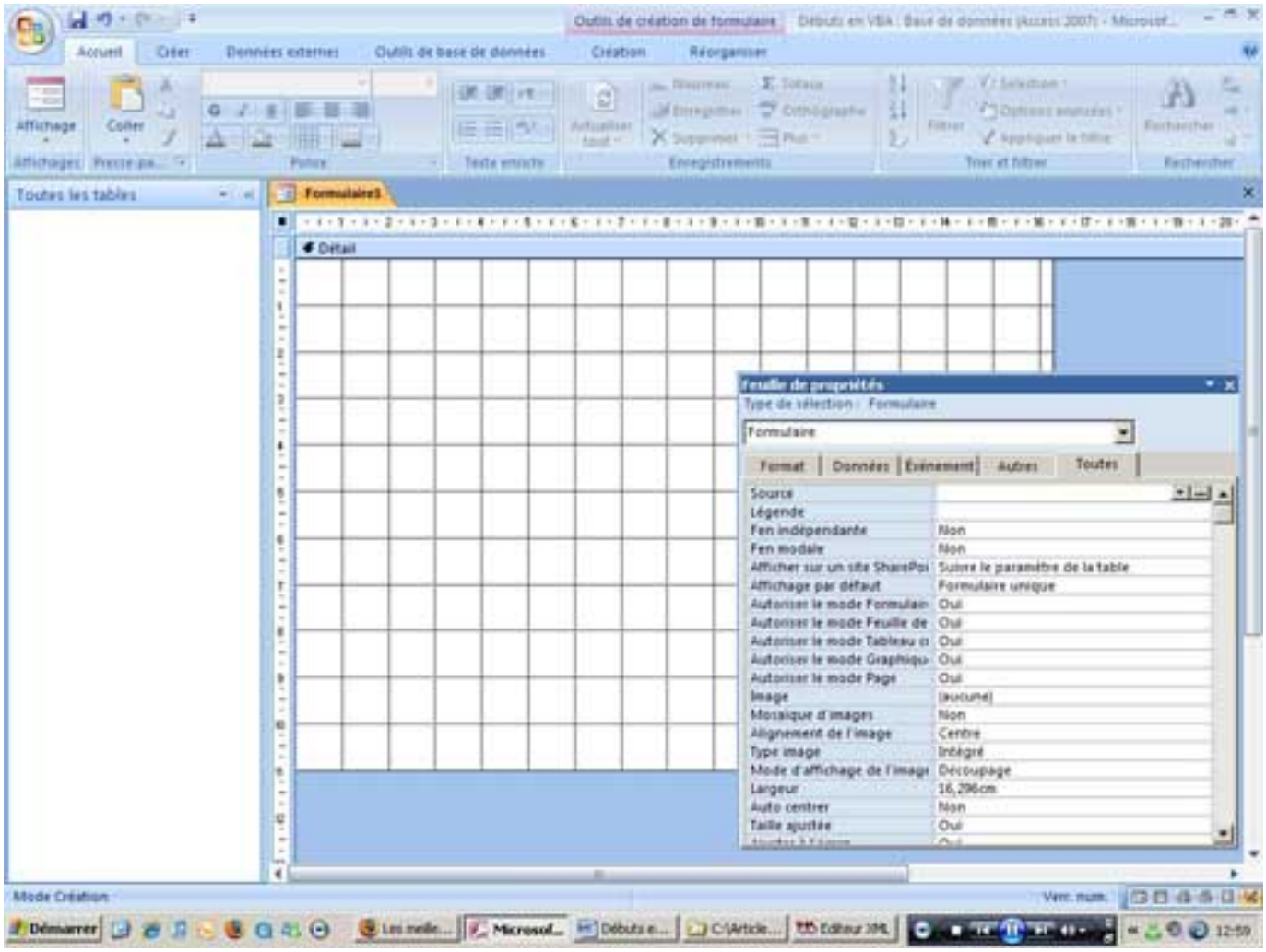
Ce tuto n'a pas la prétention de faire de vous des programmeurs de haut vol, mais juste vous permettre de comprendre les bases de la programmation sous Access en VBA.

VBA est l'abréviation de Visual Basic for Application. Ce langage ne va pas vous permettre de concevoir des applications autonomes, mais d'automatiser certaines actions dans Access ou un autre logiciel de la suite office. VBA est un complément indispensable à la suite office.

Access réagit aux évènements qui peuvent se produire sur un formulaire. C'est la raison pour laquelle il est primordial de toujours utiliser les formulaires pour l'accès aux données.

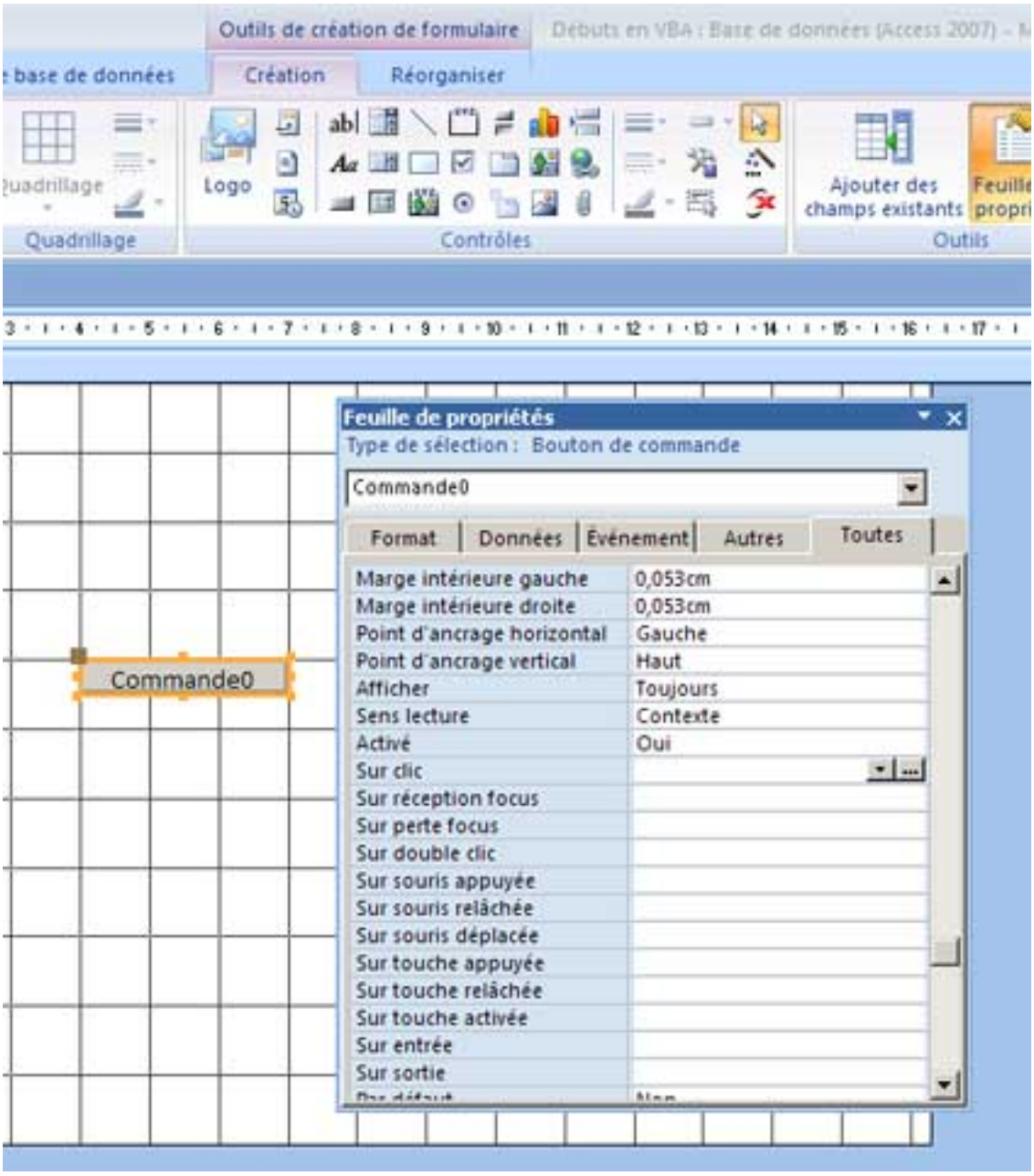
II - Ma première procédure "Helo World"

Pour cette première procédure, nous allons avoir besoin de juste un formulaire et d'un bouton.



Formulaire en mode création

Une fois le formulaire en mode création, nous allons y déposer un bouton. Afin de rester maître des opérations, on va désactiver l'assistant. La prédure est relativement simple, un clic sur l'outil et on dessine le bouton sur le formulaire.

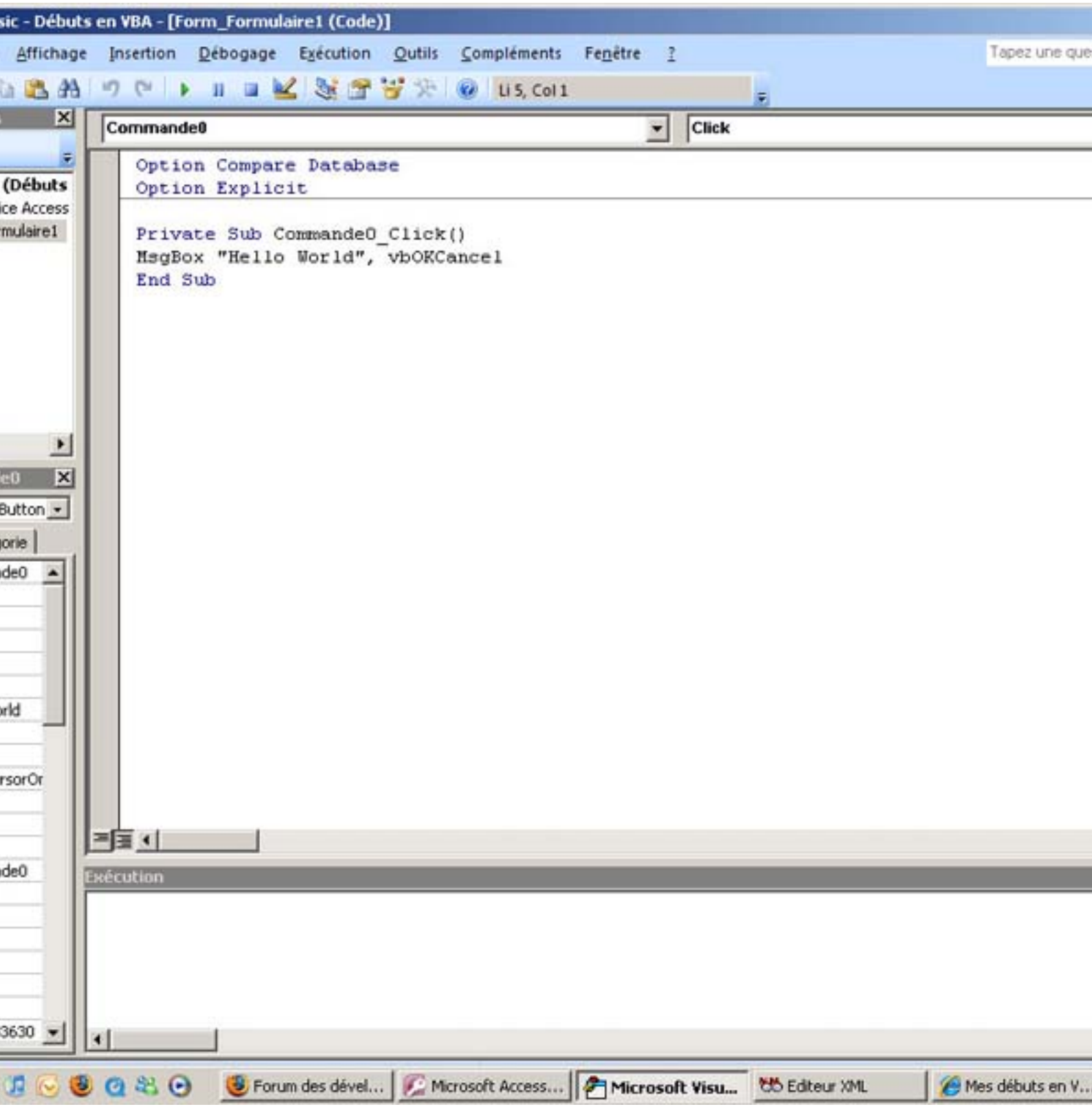


Le bouton

On va changer la légende du bouton par un double clic sur le bouton et on va lui donner comme légende "Hello World".

Bouton pour générateur de code

L'action sur le bouton générateur de code va ouvrir l'éditeur VBA.



Editeur VBA

Ma première procédure

```
Private Sub Commande0_Click()
MsgBox "Hello World"
End Sub
```

On revient alors dans Access, on affiche le formulaire en mode formulaire et on clique sur le bouton.

Voilà le résultat.

Image de notre boîte de message

Lors de l'écriture de notre message, on fait appel à une fonction qui a pour but d'afficher un message.

"MsgBox"

Le texte du message se trouve juste derrière l'appel de la fonction "MsgBox". L'aide nous donne quelques renseignements sur cette fonction :

La fonction MsgBox

```
Syntaxe
MsgBox(prompt[, buttons] [, title] [, helpfile, context])
```

Elément	Utilité
Prompt	Le text qui sera affiché dans la boîte de message. Ce texte ne peut avoir une taille suppérieure à 1024 caractères.
Buttons	Ce sont les bouton que l'on va retrouver sur le boîte de message. Par défaut on y retrouve que le bouton OK. On peut y mettre d'autres boutons.
Title	Le texte qui sera affich dans la bare de texte de la boîte de message.
HelpFile Context	

	Ces deux arguments vont de pair et sont utilisés pour rediriger l'utilisateur vers un fichier d'aide personnalisé.

Si des boutons autres que OK sont choisis, un clic sur ces boutons renvoie une valeur. Les valeurs les plus courantes sont :

Constante	Valeur retournée	Affichage
vbOK	1	OK
vbCancel	2	Annuler
vbAbort	3	Abandonner
vbRetry	4	Réessayer
vbIgnore	5	Ignorer
vbYes	6	Oui
vbNo	7	Non

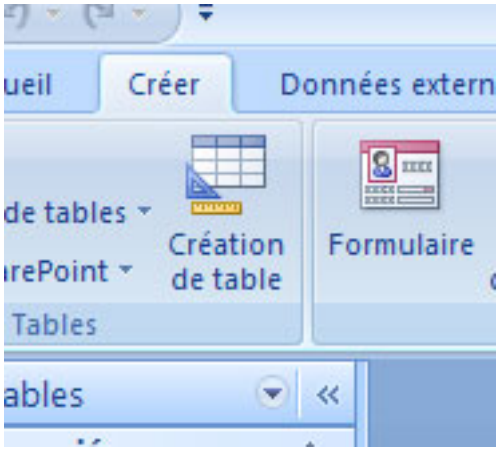
Voilà, nous venons de réaliser notre toute première procédure VBA.

III - Notre boîte de message dans un module

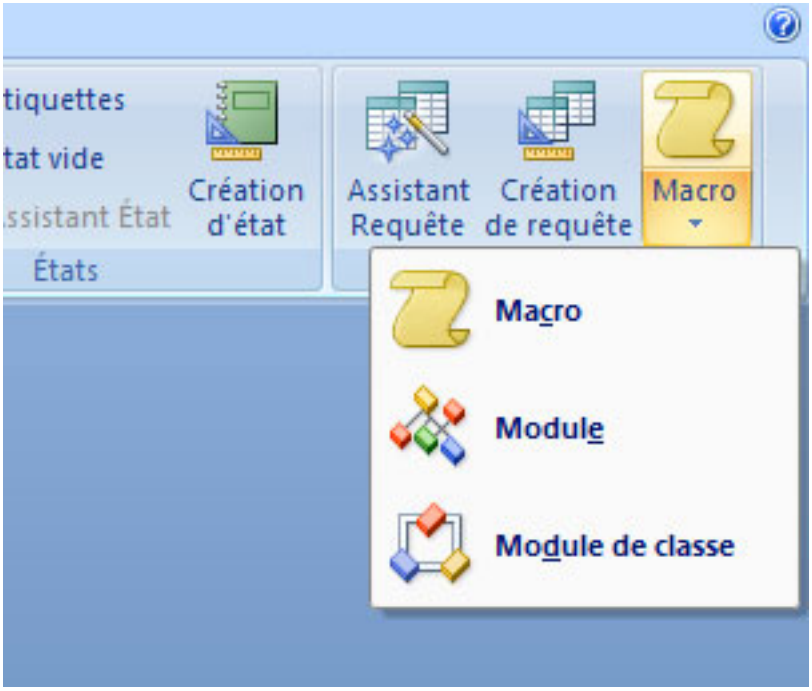
III-A - Ajout d'un procédure dan sun module

L'avantage d'une procédure ou d'une fonction appartenant à un module, est la facilité d'appel de n'importe où dans notre application. Dans l'exemple prévédent, nous avions une procédure stockée dans un formulaire. Cette procédure n'était utilisable que dans ce formulaire. La déclaration en Private en atteste. Si nous voulons utiliser cette procédure d'un autre endroit de notre application, la procéduire doit être déclarée en Public. Mais dans ce cas, il impératif d'avoir une référence vers le formulaire qui contient la procédure.

Access nous propose une solution plus facile, l'utilisation d'un module pour le stockage de nos procédures.



création



Nouveau Module

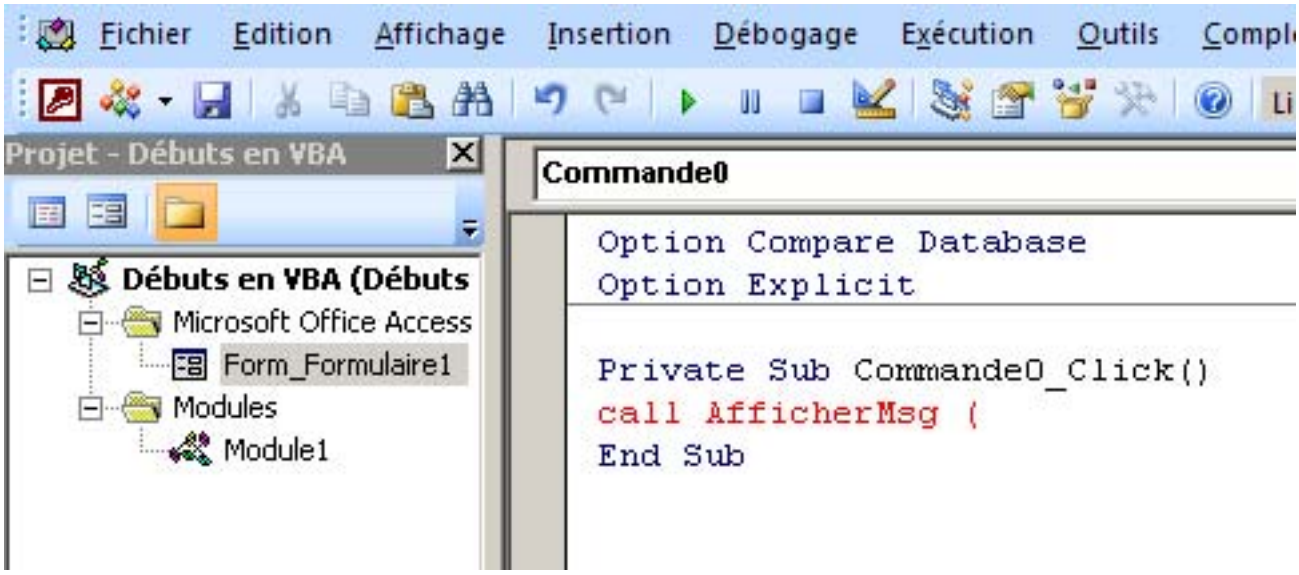
Dans cette procédure, nous allons passer des paramètres en argument. Le premier est "monMsg" et le second "monTitre" Cela signifie que lors de l'appel de la procédure dans du code, on va être amené à mettre des arguments. Ces arguments vont être utilisés pendant l'exécution de la procédure.

Voici le code que nous allons insérer dans le module. AfficherMsg est le nom de la procédure.

mon nouveau module

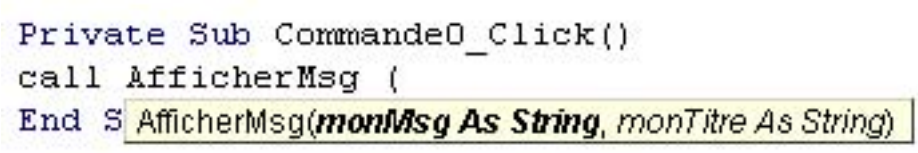
```
Public Sub AfficherMsg(monMsg As String, monTitre As String)
MsgBox monMsg, vbOKOnly, monTitre
End Sub
```

Pour pouvoir utiliser cette procédure, il faut l'appeler. Revenons à notre formulaire, où nous allons modifier le code existant. Toujours dans l'éditeur VBA, un double clic sur Formulaire1 affiche le code contenu dans le formulaire.



dbl clic

Nous allons modifier le code. Lors de la frappe, Access avec son aide à la saisie nous facilité le travail.



Aide à la saisie

Cette aide ressemble currieusement aux paramètres que nous avons mis à notre procédure.

Code modifié

```
Private Sub Commande0_Click()
Call AfficherMsg("Hello World", "Boujour à tous")
```

Code modifié
```
End Sub
```

En retournat dans notre formulaire, un petit clic sur le bouton produit le même effet.

Notre nouveau message

Ce qu'on peut tire comme conclusion c'est que le texte affiché dans notre boîte de message ne se trouve plus dans la procédure d'affichage, mais dans la ligne d'appel. Ces texte sont donc transmis.

III-B - Ajout d'une fonction

Nous allons maintenant modifier notre procédure pour en faire une fonction. La différence entre un procédure et un fonction est de taille. Un procédure s'exécute, elle "fait" quelque chose. Alors qu'une fonction retourne un résultat. Ce résultat peut-être la valeur d'un clic sur un bouton ou un résultat beaucoup plus complexe. On peut faire appel à une fonction dans une table en valeur par défaut par exemple, dans une requête, dans un formulaire, ...

Nous allons simplement remplacer Sub par Function. Ajouter à la fin de la ligne une déclaration (Single) Affecter la valeur du bouton cliqué à AfficherMsg Modifier vbOKOnly par vbOkCancel pour avoir au moins deux choix.

Notre fonction
```
Public Function AfficherMsg(monMsg As String, monTitre As String) As Single
AfficherMsg = MsgBox (monMsg, vbOKCancel, monTitre)
End Function
```

Dans notre formulaire, nous n'allons plus utiliser Call, mais juste le nom de la fonction. Cette fonction, nous allons l'utiliser pour obtenir un résultat.

Appel de la fonction
```
Private Sub Commande0_Click()
Dim resultat As Single

resultat = AfficherMsg("Hello World", "Boujour à tous")
MsgBox resultat

End Sub
```

Comme cette fonction va retourner un résultat, nous allons utiliser une variable de type réel simple pour stocker et pouvoir réutiliser la valeur renvoyée par la fonction.

Nous aurions pu utiliser une autre fonction pour afficher la valeur obtenue, nous allons juste afficher une boîte de message simple.

IV - Interaction avec une zone de texte