TUTORIAL : LES INSTRUCTIONS DE TEST

NIVEAU : DEBUTANTS / INITIES

Pré requis pour comprendre ce tutorial :

- Savoir lancer Visual Basic et créer un nouveau projet
- Connaître l'environnement de développement de Visual Basic 4.0 ou supérieur

Auteur : Dark sidious
Date de création : 26/03/2004
Version : 1.0

SOMMAIRE

INTRODUCTION

Ce tutorial est destiné à vous présenter les instructions conditionnelles (ou tests) de VB, à quoi elles servent, comment les utiliser, etc.
Je me suis efforcé de vous donner quelques petites astuces qui vous simplifieront la programmation, et vous dire des remarques pratiques.
J'espère que la lecture de ce tutorial vous sera plaisante, et que cela vous donnera envie de continuer en Visual Basic qui est un langage merveilleux pour apprendre la programmation.
Pour avoir des informations supplémentaires sur les variables, si vous voyez des erreurs, si vous avez des problèmes ou que vous ne comprenez pas trop bien ce que j'essaie de vous apprendre dans ce tutorial, ou si vous avez des idées pour un nouveau tutorial, vous pouvez m'envoyer un message privé sur le site www.ProgOtoP.com à DarK Sidious.

Bonne lecture, et bonne compréhension.

A QUOI CA SERT ?

Lorsque vous programmez, vous devez interagir avec l'utilisateur du programme et interpréter ses manipulations et ainsi, faire réagir votre programme selon ses attentes.

Si vous ne pouvez pas faire de tests dans vos programmes, votre programme ne pourra faire qu'une seule chose, et aura une utilité très restreinte !

Grâce aux instructions conditionnelles ou tests, vous pouvez aisément tester des valeurs et indiquer au programme ce qu'il doit faire selon telle condition rencontrée.

Les tests permettent d'éviter une programmation linéaire qui est très limitée, et apporte une sorte d'intelligence à vos programmes.

Les tests sont énormément utilisés dans la programmation, et je vous conseille de bien comprendre comment s'en servir avant de vous lancer dans la programmation d'un logiciel complet !

L'INSTRUCTION IF

L'instruction If est la plus utilisée et la plus commune dans les langages de programmation. VB n'y échappe pas !

L'instruction If est une instruction de test très complète et permet de faire ce que toutes les autres instructions de tests permettent de faire ! Elle est relativement concise mais peut être remplacée par des instructions beaucoup plus pratique, lisible ou concise (que nous verrons par la suite). Elle est utilisée dans bon nombre d'algorithme.

L'instruction If permet surtout de tester la valeur d'une variable ou d'une condition, ou une plage de valeur. Voici la syntaxe du test If-Then :

```
If (condition) Then (expression)
```

« Condition » est la condition qui doit être remplie pour que « expression » soit exécutée. « Expression » n'est exécutée que si la condition est remplie.

Exemple classique :

```
If i = 0 Then i = 1
```

Dans cet exemple, si i vaut 0 alors la valeur de i devient 1.

La condition peut également être une instruction, un appel à une fonction, ou toute autre expression qui renvoie une valeur, par exemple :

```
If MsgBox (« Voulez-vous quitter ? », vbYesNo) = vbYes Then End
```

Cet exemple affiche tout d'abord la boîte de dialogue « Voulez-vous quitter ? », et si l'utilisateur appuie sur le bouton « Oui » de celle-ci, alors l'instruction « End » est exécutée, ce qui a pour effet de fermer le programme.

Il est également possible d'exécuter toute une série d'instruction si le test est réussi. A ce moment là, il faut utiliser le mot clé End If à la fin du bloc de test If pour informer VB que le test se termine. Voici la syntaxe du test If-Then-End If :

```
If MsgBox (« Voulez-vous quitter ? », vbYesNo) = vbYes Then

  Msgbox « Vous avez appuyez sur oui. Le programme va donc s'arrêter»

  End

End If
```

Dans ce code, une boite de dialogue est tout d'abord affichée pour demander à l'utilisateur s'il veut quitter le programme. S'il clique sur le bouton « Oui », alors une nouvelle boîte de dialogue s'affiche pour indiquer à l'utilisateur qu'il a bien cliqué sur « Oui » et que le programme va s'arrêter. Dès que l'utilisateur clique sur le bouton « Ok » de cette boîte de dialogue, alors l'instruction End s'éxécuter se qui a pour conséquence de fermer le programme.

Note : Remarquez que si l'utilisateur clique sur le bouton « Non » dans la boite de dialogue de confirmation de sortie du programme, il ne se passe rien.

Bien entendu, l'instruction If ne se limite pas à un simple test et permet de faire des tests un peu plus approfondis. Vous pouvez par exemple combiner deux expressions avec les opérateurs binaires « And » ou « Or », « Xor », not, etc. Par exemple, si vous voulez tester la valeur d'un nombre tout en sachant que cette condition n'est réalisée que dans des cas précis :

```
If n And (i / n = 1) Then MsgBox "Le résultat de la division est 1"
```

Cet exemple teste la valeur de n : si n est non nul, alors l'intruction « i / n » est effectuée, sinon, le test est tout de suite terminé : la condition n'est pas remplie.

Note : Lorsque deux expressions sont liées avec l'opérateur Or, les deux expressions sont évaluées pour connaître leur résultat qui sera alors testé. Par contre, si elles sont liées avec l'opération Or, si la première expression n'est pas valide lors du test, la deuxième instruction ne sera pas exécutée.

Encore plus pratique, en utilisant le mot clé else vous pouvez combiner des instructions if pour définir des tests sur plusieurs valeurs par exemple. Le mot clé else (« sinon » en français) permet de tester le cas où « condition » n'est pas remplit. En fait, l'utilisation de else est l'inverse de If. Voici la syntaxe d'un test If-Then-Else :

```
If (condition) Then (expression)

Else

End if
```

Remarque : Notez que lorsque vous utilisez le mot clé Else, les mots clés End if sont obligatoires pour fermer la condition.

Tout ce qui suit le mot clé Else sera exécuté seulement si la condition du test if n'est pas remplie. Ainsi, vous pouvez traiter les cas où la condition est remplie, et les cas où elle ne l'est pas :

```
If Not (Nombre Mod 2) Then

Msgbox "Nombre est un nombre pair"

Else

Msgbox « Nombre est un nombre impair »

End If
```

Dans cet exemple, si Nombre est un nombre pair (ce que l'on teste selon le résultat du calcul du modulo du nombre avec 2), alors on affiche une boite de dialogue indiquant que le nombre est pair, sinon, il est forcément impair.
Ce genre de test est très souvent utilisé pour dissocier l'exécution normale d'un programme à une instruction particulière qui doit se passer que si une certaine condition est remplie. Pour l'affectation d'une valeur selon une condition, vous pouvez faire plus court en utilisant le test Iif que nous verrons plus loin.

Enfin, que serait le test if si nous ne pouvions pas imbriquer les tests facilement ! Imaginez que vous ayez 10 valeurs à tester, l'écriture de 10 tests if imbriqués serait vraiment fastidieuse et très peu lisibles :

```
If (Nombre = 0) Then

'…

Else

If (Nombre = 1) Then

'…

Else

'tout cela pour 10 test !!!

End If

End If
```

Heureusement, l'instruction If permet de raccourcir l'écriture et ainsi d'oublier de compter le nombre de End If à écrire à la fin de chaque bloc :

```
If (Nombre = 0) Then

'…

ElseIf (Nombre = 1) Then

'tout cela pour 10 test !

End If
```

L’utilisation d’une instruction If combinée avec des instructions ElseIf puis terminée bien souvent par une instruction Else est un cas qu’on retrouve très souvent en VB pour tester des conditions. Cependant, lorsque ce test permet de définir une plage de valeur pour une variable (comme dans notre dernier exemple), il est préférable d’utiliser l’instruction de test Select Case qui est bien plus lisible.

L'INSTRUCTION SELECT CASE

L'instruction Select Case permet de tester les valeurs d'une variable. Cette instruction est particulièrement préconisée lorsque vous voulez tester plusieurs valeurs possibles d'une variable. Cette instruction peut être plus lisible et plus compacte que son équivalent obtenu avec des tests If-Else.

Voici la syntaxe de l'instruction Select Case :

```
Select Case (expression)

Case (valeur)

[Case (valeur)]

[Case Else]

End Select
```

Expression est une expression dont on veut tester la valeur. Cela peut être une variable, une propriété, une expression algébrique, etc.
Valeur est une valeur que l'on veut tester : si expression retourne la valeur du cas, alors le Case est exécuté.
Vous pouvez remarquer que ce test est un peu plus long à taper qu'une simple instruction if. C'est pour cette raison que l'utilisation d'un test Select Case est surtout préconisée lors du test de plusieurs valeurs pour une expression.

Voici un exemple pratique d'utilisation : le test de l'âge d'une personne avec le test If classique :

```
If (Age_Jean = 10) Then

MsgBox « Jean a 10 ans. C'est donc encore un jeune homme. »

ElseIf (Age_Jean = 20) Then

MsgBox « Jean a 20 ans, il est donc majeur. »

ElseIf (Age_Jean = 70) Then

MsgBox « Jean a 70 ans, c'est donc un retraité. »

End If
```

Voici le même test, mais avec l'instruction Select Case :

```
Select Case Age_Jean

Case 10

  MsgBox « Jean a 10 ans. C'est donc encore un jeune homme. »

Case 20

  MsgBox « Jean a 20 ans, il est donc majeur. »

Case 70

  MsgBox « Jean a 70 ans, c'est donc un retraité. »

  End Select
```

La quantité de code à taper est bien plus mince avec l'utilisation d'un test Select case plutôt qu'un test If. De plus, le test est beaucoup plus lisible pour un test avec Select Case !
Si vous vouliez tester les valeurs intermédiaires, telle que 25 ans par exemple, cela est aussi possible avec l'instruction Select Case :

```
Select Case Age_Jean

Case 0 To 18

  MsgBox « Jean a entre 0 et 18 ans. C'est donc encore un jeune homme. »

Case 19 To 60

  MsgBox « Jean a entre 19 et 20 ans, il est donc majeur. »

Case 61 To 100

  MsgBox « Jean a plus de 61 ans, c'est donc un retraité. »

  End Select
```

L'instruction Select case est donc très pratique pour tester des intervalles de valeurs, ou tester quelques valeurs que peux prendre une expression. Ce test peut aussi porter sur des chaînes de caractères, des dates, etc.

L'INSTRUCTION IIF

L'instruction Iif est un test qui est utile pour l'affectation de valeur selon la valeur d'une expression. Par exemple, si vous voulez définir la valeur de l'âge de Jean selon la valeur de l'âge de Julie, une instruction Iif le permet en une seule ligne de code. Il s'agit ni plus ni moins d'une instruction If en plus compacte.
Cette instruction est apparue avec l'arrivée du langage VBA (Visual Basic for Application) qui est intégré dans les logiciels de la suite Microsoft Office.

Voici sa syntaxe :

```
Iif (Condition, ValeurVraie, ValeurFausse)
```

Condition est une expression à tester.
ValeurVraie est la valeur renvoyée par la fonction Iif si Condition est vraie.
ValeurFausse est la valeur renvoyée par la fonction Iif si Condition est fausse.

Grâce à cette petite fonction, il est très rapide d'affecter une valeur selon la valeur d'une variable :

```
Age_Jean = Iif(Age_Julie = 10, 20, 15)
```

Dans cet exemple, Age_Jean vaudra 20 si Age_Julie = 10 sinon, Age_Jean vaudra 15.

Nous aurions pu obtenir le même résultat avec un test If classique, mais il aurait fallut plus de lignes de codes, et la lisibilité est améliorée avec le test Iif.

L'INSTRUCTION CHOOSE

L'instruction Choose est une sorte d'instruction Select Case un peu plus compacte et plus ciblée. Elle permet d'affecter une valeur à une variable selon le résultat d'une condition.
Cette instruction est apparue avec l'arrivée du langage VBA (Visual Basic for Application) qui est intégré dans les logiciels de la suite Microsoft Office.

Voici sa syntaxe :

```
Choose (Index, sélection1[, sélection2, sélection3, …, sélectionN)
```

L'instruction Choose permet de renvoyer l'une des sélections selon la valeur de l'index.
Index est une valeur entière qui peut être une expression à évaluer, une variable ou autre.
Sélection1 est la valeur renvoyée par l'instruction choose si l'index est égal à 1.
Sélection2 à N sont facultatives. Il s'agit des valeurs renvoyées si index est égal à 2 ou 3 ou … jusqu'à N.

Un exemple pratique d'utilisation de cette instruction est l'affectation d'une valeur à une variable selon la valeur d'un calcul, par exemple :

```
Age_Julien = Choose(Age_Jean – Age_Julie, 10, 20, 40, 60)
```

Dans cet exemple, Age_Julien prendra la valeur 10 si Age_Jean – Age_Julie vaut 1, 20 si Age_Jean – Age_Julie vaut 2, etc.

Nous aurions pu obtenir le même résultat avec un test If classique, mais il aurait fallut plus de lignes de codes, et la lisibilité est améliorée avec le test Choose. Il s'agit ni plus ni moins d'une instruction Iif avec une plage de retour plus étendue.

L'INSTRUCTION SWITCH

L'instruction Switch est une sorte d'instruction Choose un peu plus complète mais moins concise. Elle permet d'affecter une valeur à une variable selon le résultat de plusieurs conditions.
Cette instruction est apparue avec l'arrivée du langage VBA (Visual Basic for Application) qui est intégré dans les logiciels de la suite Microsoft Office.

Voici sa syntaxe :

```
Switch (Condition1, Valeur1 [, Condition2, Valeur2, ..., ConditionN, ValeurN)
```

L'instruction Choose permet de renvoyer l'une des valeurs si la condition la précédant est vraie.

Condition1 jusqu'à N sont des expressions à tester, et si l'une d'elle est vérifiée, l'instruction Switch renvoie la Valeur correspondante, et ne teste pas les autres Condition.
Cette instruction est particulièrement adaptée pour affecter une valeur à une variable lors d'un test d'une variable de type chaine de caractère par exemple :

```
Age_Jean = Switch(Age_Julie = « vingt », 20, Age_Julie = « trente », 30)
```

Nous aurions pu obtenir le même résultat avec un test If classique, mais il aurait fallut plus de lignes de codes, et la lisibilité est améliorée avec le test Iif.

CONCLUSION

Comme vous pouvez le voir, il existe beaucoup d'instruction de test en VB, et cela ne facilite pas le choix lors de la programmation. Cependant, cette variété permet de vite trouver celle qu'il vous faut ! Alors laquelle choisir ? Cela dépend beaucoup du type de test à effectuer.

Tout est réalisable avec l'instruction If classique, mais elle n'est pas très appropriée pour quelques cas, et dans ces cas là, il vaut mieux utiliser les autres instructions qui sont souvent plus compactes et lisibles.

Les principales instructions de tests utilisés restent les plus classiques que l'on retrouve dans de nombreux langages : If et Select Case, cependant, les autres instructions ne sont pas à négliger car elle apporte un gain certains en concision !

Maîtriser les instructions de test permet de développer des programmes qui réagissent à des conditions, et c'est la clé de voûte à un programme « normal » qui peut effectuer plus d'une seule chose.

J'espère que la lecture de ce tutorial vous a satisfaite, et que vous continuerez à programmer sous Visual Basic. Et si vous avez des commentaires à me faire, n'hésitez pas à m'envoyer vos messages sur le site www.ProgOtoP.com.