ANNEXE B

Conventions de codage de Visual Basic Document MICROSOFT

Cette annexe suggère un ensemble de conventions de codage pour les programmes Visual Basic.

Les conventions de codage constituent des règles de programmation qui ne concernent pas la logique du programme mais sa structure physique et sa présentation. Elles facilitent la lecture, la compréhension et la gestion du code. Ces conventions de codage peuvent regrouper :

- Les conventions d'affectation de noms aux objets, aux variables et aux procédures.
- Les formats standard des étiquettes et des commentaires ajoutés au code.
- Les conventions relatives à l'espacement, la mise en forme et la mise en retrait.

Ces différents points sont présentés dans les sections suivantes, qui fournissent également des exemples d'utilisation.

Sommaire

Filename: Ms_ConvCod.DOC Project: Conventions de codage
Template: naetugp.dot Author: Mary B. Last Saved By: Claude
Revision #: 2 Page: 929 of 13 Printed: 12/10/01 17:12 0112/P12

Pourquoi respecter des conventions de codage ?

Le principal avantage d'un ensemble de conventions de codage cohérent est d'obtenir une application ayant une structure et un style de codage standard, dont le code peut ainsi être lu et compris sans problème par vous-même ou par d'autres utilisateurs.

Le respect des conventions de codage permet de créer un code source précis, lisible et exempt d'ambiguïtés. Ce code sera, de plus, cohérent avec les conventions respectées dans d'autres langages et aussi intuitif que possible.

Conventions de codage minimales

Un ensemble de conventions générales de codage permettra de définir les conditions minimales nécessaires pour écrire un code doté des qualités évoquées ci-dessus et permettre ainsi au programmeur de se concentrer sur la logique et le flux fonctionnel du programme.

L'objectif est d'obtenir un programme facile à lire et à comprendre sans gêner la créativité du programmeur avec des contraintes excessives et des restrictions arbitraires.

L'objectif de cette annexe est de présenter quelques conventions simples. Elle ne réglemente donc pas tous les objets ou contrôles existants et ne signale pas tous les cas où un commentaire serait utile pour apporter des informations complémentaires. En fonction de votre projet et des exigences de votre société, vous pourrez l'enrichir en y incluant, par exemple :

- Des conventions relatives à des objets et composants spécifiques, développés en interne ou acquis auprès de fournisseurs tiers.
- Des variables décrivant les activités ou les services de votre société.
- Tout autre élément jugé important par votre société ou dans le cadre de votre projet pour la lisibilité et la clarté du programme.

Pour plus d'informations sur les restrictions relatives aux noms de procédures, variables et constantes, reportez-vous à la section « Principes d'écriture du code » du chapitre 5, « Éléments fondamentaux de programmation ».

Filename: Ms_ConvCod.DOC Project: Conventions de codage
Template: naetugp.dot Author: Mary B. Last Saved By: Claude
Revision #: 2 Page: 930 of 13 Printed: 12/10/01 17:12 0112/P12

Conventions d'affectation de noms aux objets

Les noms d'objets doivent comporter un préfixe cohérent, permettant d'identifier facilement leur type. Vous trouverez ci-dessous la liste des conventions relatives à certains objets pris en charge par Visual Basic.

Suggestions de préfixes de contrôles

Type de contrôle	Préfixe	Exemple
Barre de défilement horizontale (HScrollBar)	hsb	hsbVolume
Barre de défilement verticale (VScrollBar)	vsb	vsbRate
Bouton animé (AniPushButton)	ani	aniMailBox
Bouton de commande (CommandButton)	cmd	cmdExit
Cadre (Frame)	fra	fraLanguage
Case à cocher (CheckBox)	chk	chkReadOnly
Boites de dialogue communes (CommonDialog)	dlg	dlgFileOpen
Communications (MSComm)	com	comFax
Plan (Outline)	out	outOrgChart
Contrôle (Control) (utilisé lorsque le type spécifique est inconnu)	ctr	ctrCurrent
Contrôle de données (Data)	dat	datBiblio
État (CrystalReport)	rpt	rptQtr1Earnings
État de la touche (Mhstate)	key	keyCaps
Étiquette (Label)	lbl	lblHelpMessage
Feuille (Form)	frm	frmEntry
Feuille MDI fille (MDI child form)	mdi	mdiNote
Forme (Shape)	shp	shpCircle
Graphique (Graph)	gra	graRevenue
Grille (Grid)	grd	grdPrices
Grille dépendante (DBGrid)	dbgrd	dbgrdQueryResult
Image	img	imgIcon
Image (Picture)	pic	picVGA
Liste d'images (ImageList)	ils	ilsAllIcons
Indicateur (Gauge)	gau	gauStatus
Ligne (Line)	lin	linVertical
Liste de fichiers (FileListBox)	fil	filSource

Filename: Ms_ConvCod.DOC Project: Conventions de codage
Template: naetugp.dot Author: Mary B. Last Saved By: Claude
Revision #: 2 Page: 931 of 13 Printed: 12/10/01 17:12 0112/P12

Type de contrôle	Préfixe	Exemple
Liste de dossiers (DirListBox)	dir	dirSource
Liste de lecteurs (DriveList Box)	drv	drvTarget
Liste modifiable dépendante (DBCombo)	dbcbo	dbcboLanguage
Affichage de listes (ListView)	lvw	lvwHeadings
Multimédia MCI	mci	mciVideo
Menu	mnu	mnuFileOpen
Message MAPI (MAPI message)	mpm	mpmSentMessage
Minuterie (Timer)	tmr	tmrAlarm
MS Flex grid	msg	msgClients
Onglet (MSTab)	mst	mstFirst
OLE	ole	oleWorksheet
Panneau 3D (3D Panel)	pnl	pnlGroup
Pen BEdit	bed	bedFirstName
Pen Hedit	hed	hedSignature
Pen ink	ink	inkMap
Jeu d'image (PictureClip)	clp	clpToolbar
Barre de progression (ProgressBar)	prg	prgLoadFile
Éditeur RTF (RichTextBox)	rtf	rtfReport
Session MAPI (MAPI session)	mps	mpsSession
Curseur (Slider)	sld	sldScale
Compteur (SpinButton)	spn	spnPages
Barre d'état (StatusBar)	sta	staDateTime
Barre d'onglets (TabStrip)	tab	tabOptions
Barre d'outils (Toolbar)	tlb	tlbActions
Affichage de l'arborescence (TreeView)	tre	treOrganization
Barre de défilement (UpDown)	upd	updDirection
Zone de liste (ListBox)	lst	lstPolicyCodes
Zone de liste dépendante (DBlist)	dblst	dblstJobType
Zone de liste modifiable, zone de liste déroulante (ComboBox)	cbo	cboEnglish
Zone de texte (TextBox)	txt	txtLastName

Filename: Ms_ConvCod.DOC Project: Conventions de codage
Template: naetugp.dot Author: Mary B. Last Saved By: Claude
Revision #: 2 Page: 932 of 13 Printed: 12/10/01 17:12 0112/P12

Suggestions de préfixes d'objets d'accès aux données (DAO)

Utilisez les préfixes suivants pour signaler des objets d'accès aux données.

Objet de base de données	Préfixe	Exemple
Container	con	conReports
Database	db	dbAccounts
DBEngine	dbe	dbeJet
Document	doc	docSalesReport
Field	fld	fldAddress
Group	grp	grpFinance
Index	idx	idxAge
Parameter	prm	prmJobCode
QueryDef	qry	qrySalesByRegion
Recordset	rec	recForecast
Relation	rel	relEmployeeDept
TableDef	tbd	tbdCustomers
User	usr	usrNew
Workspace	wsp	wspMine

Exemples :

```
Dim dbBiblio As Database
Dim recPubsInNY As Recordset, strSQLStmt As String
Const DB_READONLY = 4          ' Définit la constante.
'Ouvre la base de données.
Set dbBiblio = OpenDatabase("BIBLIO.MDB")
' Définit le texte de l'instruction SQL .
strSQLStmt = "SELECT * FROM Publishers WHERE _
   State = 'NY'"
' Crée l'objet Recordset.
Set recPubsInNY = db.OpenRecordset(strSQLStmt, _
   dbReadOnly)
```


Filename: Ms_ConvCod.DOC Project: Conventions de codage
Template: naetugp.dot Author: Mary B. Last Saved By: Claude
Revision #: 2 Page: 933 of 13 Printed: 12/10/01 17:12 0112/P12

Suggestions de préfixes de menus

Les applications utilisent généralement de nombreux contrôles de menus ; il est donc utile d'adopter pour ces contrôles un ensemble unique de conventions d'affectation de noms. Nous vous conseillons d'ajouter à l'étiquette initiale « mnu » des préfixes de contrôles de menus un préfixe supplémentaire pour chaque niveau d'imbrication, le libellé de menu final figurant à la fin de la chaîne de nom. Le tableau suivant fournit quelques exemples.

Séquence de libellés de menu	Nom du gestionnaire de menu
Fichier Ouvrir	mnuFileOpen
Fichier Envoyer Email	mnuFileSendEmail
Fichier Envoyer télécopie	mnuFileSendFax
Format caractère	mnuFormatCharacter
Index d'Aide	mnuHelpContents

Si vous utilisez cette convention d'affectation de noms, tous les membres d'un menu donné figurent les uns à côté des autres dans la fenêtre Propriétés de Visual Basic. De plus, les noms de contrôles de menus permettent d'identifier facilement les éléments de menu auxquels ils sont associés.

Choix de préfixes pour les autres contrôles

Pour les contrôles qui ne figurent pas ci-dessus, utilisez un préfixe standard, constitué de deux ou trois caractères. Utilisez plus de trois caractères uniquement si cela s'impose pour des raisons de clarté.

Pour les contrôles dérivés ou modifiés, par exemple, ajoutez des lettres aux préfixes ci-dessus afin d'éviter toute confusion quant au contrôle réellement utilisé. Pour les contrôles provenant de fournisseurs tiers, ajoutez au préfixe une abréviation en minuscules du nom du fournisseur. Par exemple, une instance d'un contrôle créé à l'aide du cadre 3D Visual Basic Édition Professionnelle peut être signalée par le préfixe fra3d afin d'éviter toute confusion quant au contrôle réellement utilisé.

Filename: Ms_ConvCod.DOC Project: Conventions de codage
Template: naetugp.dot Author: Mary B. Last Saved By: Claude
Revision #: 2 Page: 934 of 13 Printed: 12/10/01 17:12 0112/P12

Conventions d'affectation de noms aux constantes et aux variables

Outre les objets, les constantes et les variables nécessitent également l'adoption de conventions d'affectation de noms standard. Cette section présente les conventions prises en charge par Visual Basic que nous vous recommandons d'utiliser. Elle traite également des problèmes d'identification du type de données et de la portée.

Il est conseillé de toujours définir les variables avec la plus petite portée possible. Les variables globales (Public) peuvent entraîner des situations d'une complexité extrême et rendre la logique d'une application très difficile à comprendre. Elles peuvent également rendre la réutilisation et la gestion de votre code beaucoup plus délicates.

Dans Visual Basic, les variables peuvent être dotées de la portée suivante :

Portée	Déclaration	Visible dans
De niveau procédure	'Private' dans la procédure, la Sub ou la Fonction	La procédure dans laquelle elle est déclarée
De niveau module	'Private' dans la section des déclarations d'une feuille ou d'un module de code (.frm, .bas) Déclarations	Toute procédure dans la feuille ou le module de code
Globale	'Public' dans la section des déclarations d'un module de code	Partout dans l'application

Dans une application Visual Basic, il est conseillé de n'utiliser des variables globales que s'il n'existe aucun autre moyen adéquat de partager des données entre des feuilles. Si vous devez utiliser des variables globales, il est recommandé de les déclarer toutes dans un seul module, regroupées par fonction. Attribuez au module un nom évocateur, indiquant son but, tel que Public.bas.

Il est conseillé d'écrire un code modulaire chaque fois que cela est possible. Par exemple, si votre application affiche une boîte de dialogue, placez la totalité des contrôles et du code nécessaires à la tâche prévue pour la boîte de dialogue dans une feuille unique. Vous pouvez ainsi organiser le code de l'application en composants significatifs et gagner du temps au moment de l'exécution.

À l'exception des variables globales (qui ne doivent pas être passées), les procédures et les fonctions ne doivent traiter que des objets qui leur sont transmis. Les variables globales utilisées dans des procédures doivent être indiquées dans la section des déclarations au début de la procédure. De plus, il est conseillé de transmettre des arguments aux procédures Sub et aux fonctions à l'aide du mot clé ByVal, à moins que vous ne soyez obligé de modifier explicitement la valeur de l'argument transmis.

Filename: Ms_ConvCod.DOC Project: Conventions de codage
Template: naetugp.dot Author: Mary B. Last Saved By: Claude
Revision #: 2 Page: 935 of 13 Printed: 12/10/01 17:12 0112/P12

Préfixes de portée de variables

Plus le projet grandit, plus il est important de déterminer rapidement la portée des variables. Un préfixe de portée constitué d'une lettre précédant le préfixe de type permet d'arriver à ce résultat sans pour autant augmenter de manière significative la taille des noms de variables.

Portée	Préfixe	Exemple
Globale	g	gstrUserName
De niveau module	m	mblnCalcInProgress
Locale à une procédure	Aucun	dblVelocity

Une variable est dotée d'une portée globale si elle est déclarée comme Public dans un module standard ou un module de la feuille. Une variable est dotée d'une portée *de niveau module* si elle est déclarée comme Private, respectivement dans un module standard ou un module de feuille.

Note Pour que cette technique vous apporte tous les avantages escomptés, il est absolument nécessaire d'être cohérent ; en effet, le vérificateur de syntaxe de Visual Basic ne s'arrêtera pas sur des variables de niveau module commençant par la lettre « p ».

Constantes

Le corps des noms de constantes doit être constitué de lettres de casse différente, chaque mot commençant par une lettre majuscule. Bien que les constantes Visual Basic standard ne comprennent pas d'informations sur le type de données et la portée, des préfixes tels que i, s, g et m peuvent s'avérer très utiles pour comprendre la valeur et la portée d'une constante. Pour les noms de constantes, respectez les mêmes conventions que pour les variables. Exemple :

```
mintUserListMax     'Limite d'entrée maximale de la
                    'liste des utilisateurs
                    '(nombre entier, local à un
                    'module)
gstrNewLine         'Caractère de saut de ligne
                    '(chaîne, globale à une
                    'application)
```


Filename: Ms_ConvCod.DOC Project: Conventions de codage
Template: naetugp.dot Author: Mary B. Last Saved By: Claude
Revision #: 2 Page: 936 of 13 Printed: 12/10/01 17:12 0112/P12

Variables

Le fait de déclarer toutes les variables réduit le temps de programmation en limitant le nombre d'erreurs causées par des fautes de frappe (par exemple : aUserNameTmp, sUserNameTmp et sUserNameTemp). Dans l'onglet Éditeur de la boîte de dialogue Options, cochez l'option Requiert la déclaration de variables. L'instruction Option Explicit vous oblige à déclarer toutes les variables dans votre programme Visual Basic.

Les variables doivent comporter un préfixe indiquant leur type de données. Ce préfixe peut éventuellement être complété pour indiquer la portée de la variable, en particulier dans le cas de programmes volumineux.

Type de données des variables

Utilisez les préfixes suivants pour indiquer le type de données d'une variable.

Type de données	Préfixe	Exemple
Boolean	bln	blnFound
Byte	byt	bytRasterData
Collection (Objet)	col	colWidgets
Currency	cur	curRevenue
Date (Time)	dtm	dtmStart
Double	dbl	dblTolerance
Error	err	errOrderNum
Integer	int	intQuantity
Long	lng	lngDistance
Object	obj	objCurrent
Single	sng	sngAverage
String	str	strFName
Type défini par l'utilisateur	udt	udtEmployee
Variant	vnt	vntCheckSum

Filename: Ms_ConvCod.DOC Project: Conventions de codage
Template: naetugp.dot Author: Mary B. Last Saved By: Claude
Revision #: 2 Page: 937 of 13 Printed: 12/10/01 17:12 0112/P12

Noms de procédures et de variables descriptifs

Le corps d'un nom de variable ou de procédure doit comporter des lettres de casse différente et être aussi long que la description de sa fonction le demande. De plus, les noms de fonction doivent commencer par un verbe, tel que InitNameArray ou CloseDialog.

Pour les termes fréquemment utilisés ou particulièrement longs, nous vous recommandons d'utiliser des abréviations standard afin de réduire la longueur des noms. En règle générale, des noms de variables supérieurs à 32 caractères peuvent devenir difficiles à lire en affichage VGA.

Si vous utilisez des abréviations, vérifiez qu'elles sont cohérentes dans la totalité de l'application. Par exemple, l'utilisation de Cnt et Count indifféremment dans un projet risque de semer la confusion.

Types définis par l'utilisateur

Dans un gros projet comportant de nombreux types définis par l'utilisateur, il est très utile d'attribuer à chacun d'entre eux un préfixe de trois caractères distincts. Si ces préfixes commencent par la lettre « u », ils seront aisément identifiables lorsque vous travaillerez avec un type défini par l'utilisateur. Par exemple, le préfixe « ucli » peut être utilisé pour les variables dotées du type défini par l'utilisateur Client.

Conventions de structuration du code

Outre les conventions d'affectation de noms, les conventions de structuration du code, tels que les commentaires ajoutés au code et une mise en retrait cohérente, peuvent améliorer de façon considérable la lisibilité du code.

Conventions relatives aux commentaires ajoutés au code

Il est conseillé de faire commencer toutes les procédures et fonctions par un bref commentaire décrivant les caractéristiques fonctionnelles de la procédure (les opérations qu'elle effectue). Cette description ne doit pas inclure les détails d'implémentation (autrement dit, comment les opérations sont effectuées) car ces derniers varient souvent dans le temps et entraîneraient un travail de gestion des commentaires superflus ou, pire encore, des commentaires erronés. Le code proprement dit et les commentaires des lignes nécessaires suffiront à décrire l'implémentation.

Les arguments transmis à une procédure doivent être décrits lorsque leur fonction n'est pas explicite et lorsque la procédure s'attend à ce que les arguments se trouvent dans une plage spécifique. Les valeurs renvoyées par la fonction et les variables globales modifiées par la procédure, en particulier par l'intermédiaire d'arguments de référence, doivent également être décrites au début de chaque procédure.

Filename: Ms_ConvCod.DOC Project: Conventions de codage
Template: naetugp.dot Author: Mary B. Last Saved By: Claude
Revision #: 2 Page: 938 of 13 Printed: 12/10/01 17:12 0112/P12

Les blocs de commentaires situés en en-tête de procédure doivent comprendre les en-têtes de section suivants. Des exemples figurent dans la section suivante, « Mise en forme de votre code ».

En-tête de section	Description du commentaire
Objet	Opérations effectuées par la procédure (et non la manière dont elle les effectue).
Commentaires	Liste de chaque variable, contrôle, fichier ouvert ou autre élément externe qui n'est pas explicite.
Description	Liste de chaque variable, contrôle, fichier ouvert externe et son effet (uniquement si cela n'est pas explicite).
Entrées	Tout argument qui n'est pas explicite. Les arguments figurent sur des lignes distinctes avec des commentaires de ligne.
Retours	Explication des valeurs renvoyées par les fonctions.

Gardez à l'esprit les points suivants :

- Toute déclaration de variable importante doit inclure un commentaire de ligne décrivant l'utilisation de la variable déclarée.
- Les variables, contrôles et procédures doivent être dotés de noms suffisamment explicites pour qu'un commentaire de ligne ne soit nécessaire que dans le cas d'une implémentation complexe.
- Au début du module .bas qui contient les déclarations de constantes génériques de Visual Basic utilisées dans le projet, nous vous conseillons d'inclure une présentation de l'application, énumérant les objets d'accès aux données primaires, les routines, les algorithmes, les boîtes de dialogue, les bases de données et les dépendances du système. Quelques lignes de pseudocode décrivant l'algorithme peuvent parfois s'avérer utiles.

Filename: Ms_ConvCod.DOC Project: Conventions de codage
Template: naetugp.dot Author: Mary B. Last Saved By: Claude
Revision #: 2 Page: 939 of 13 Printed: 12/10/01 17:12 0112/P12

Mise en forme de votre code

Étant donné que de nombreux programmeurs utilisent encore le mode d'affichage VGA, il est conseillé de conserver autant d'espace d'affichage que possible tout en préservant une mise en forme du code qui reflète la structure logique et les niveaux d'imbrication. Quelques conseils :

- Les blocs imbriqués standard, basés sur des tabulations, doivent être mis en retrait à l'aide de quatre espaces (par défaut).
- Le commentaire relatif à la présentation fonctionnelle d'une procédure doit être mis en retrait à l'aide d'un espace. Les instructions de plus haut niveau suivant le commentaire de présentation doivent être mises en retrait à l'aide d'une tabulation, chaque bloc imbriqué devant être mis en retrait à l'aide d'une tabulation supplémentaire. Par exemple :

```
'*************************************************
' Objet: Recherche la première occurrence d'un
'        utilisateur indiqué dans le tableau
'        UserList.
' Entrées:
'  strUserList(): liste des utilisateurs à rechercher.
'  strTargetUser: nom de l'utilisateur à rechercher.
' Retours:  L'index de la première occu rrence de
'           rsTargetUser dans le tableau rasUserList.
'           Si l'utilisateur cible est introuvable,
'           renvoie -1.
'*************************************************

Function intFindUser (strUserList() As String, strTargetUser As _
   String)As Integer
   Dim i As Integer           ' Compteur de boucle.
   Dim blnFound As Integer    ' Indicateur cible trouvé.
   intFindUser = -1
   i = 0
   While i <= Ubound(strUserList) and Not blnFound
      If strUserList(i) = strTargetUser Then
         blnFound = True
         intFindUser = i
      End If
   Wend
End Function
```

Regroupement de constantes

Les variables et les constantes définies doivent être regroupées par fonction et non disséminées dans des zones isolées ou des fichiers spéciaux. Les constantes génériques de Visual Basic doivent être regroupées au sein d'un module unique afin d'être séparées des déclarations spécifiques à l'application.

Filename: Ms_ConvCod.DOC Project: Conventions de codage
Template: naetugp.dot Author: Mary B. Last Saved By: Claude
Revision #: 2 Page: 940 of 13 Printed: 12/10/01 17:12 0112/P12

Opérateurs & et +

Utilisez toujours l'opérateur & pour relier des chaînes et l'opérateur + lorsque vous traitez des valeurs numériques. L'utilisation de l'opérateur + pour la concaténation peut causer des problèmes s'il est employé avec deux variants. Par exemple :

```
vntVar1 = "10.01"
vntVar2 = 11
vntResult = vntVar1 + vntVar2          'vntResult = 21.01
vntResult = vntVar1 & vntVar2  'vntResult = 10.0111
```

Création de chaînes pour MsgBox, InputBox et des requêtes SQL

Pour créer une longue chaîne de caractères, utilisez le caractère de continuité de ligne (trait de soulignement) afin d'insérer plusieurs lignes de code et ainsi être en mesure de lire et de déboguer facilement la chaîne. Cette technique est particulièrement utile pour afficher un message (MsgBox) ou une zone d'entrée (InputBox) ou lors de la création d'une chaîne SQL. Par exemple :

```
Dim Msg As String
Msg = "Paragraphe qui apparaîtra " _
& "dans un message. Le texte est" _
& " divisé en plusieurs lignes de code" _
& " dans le code source, le programmeur pouvant" _
& " ainsi le lire et le déboguer facilement."
MsgBox Msg

Dim QRY As String
QRY = "SELECT *" _
& " FROM Titles" _
& " WHERE [Year Published] > 1988"
TitlesQry.SQL = QRY
```


Filename: Ms_ConvCod.DOC Project: Conventions de codage
Template: naetugp.dot Author: Mary B. Last Saved By: Claude
Revision #: 2 Page: 941 of 13 Printed: 12/10/01 17:12 0112/P12